

palgrave
macmillan

Control engineering

An introductory course

Jacqueline Wilkie, Michael Johnson
and Reza Katebi





Control engineering

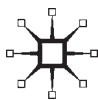
An introductory course

Jacqueline Wilkie

Michael Johnson

Reza Katebi

palgrave



© Jacqueline Wilkie, Michael Johnson and Reza Katebi 2002

All rights reserved. No reproduction, copy or transmission of this publication may be made without written permission.

No paragraph of this publication may be reproduced, copied or transmitted save with written permission or in accordance with the provisions of the Copyright, Designs and Patents Act 1988, or under the terms of any licence permitting limited copying issued by the Copyright Licensing Agency, 90 Tottenham Court Road, London W1P 0LP.

Any person who does any unauthorised act in relation to this publication may be liable to criminal prosecution and civil claims for damages.

The authors have asserted their right to be identified as the authors of this work in accordance with the Copyright, Designs and Patents Act 1988.

First published 2002 by
PALGRAVE

Houndmills, Basingstoke, Hampshire RG21 6XS and
175 Fifth Avenue, New York, N. Y. 10010

Companies and representatives throughout the world

PALGRAVE is the new global academic imprint of
St. Martin's Press LLC Scholarly and Reference Division and
Palgrave Publishers Ltd (formerly Macmillan Press Ltd).

ISBN 0-333-77129-X

This book is printed on paper suitable for recycling and
made from fully managed and sustained forest sources.

A catalogue record for this book is available
from the British Library.

Typeset by Ian Kingston Editorial Services, Nottingham, UK

10 9 8 7 6 5 4 3 2 1
11 10 09 08 07 06 05 04 03 02

Printed and bound in Great Britain by
Antony Rowe Ltd, Chippenham, Wiltshire

To my husband, Patrick, for his continual supply of tea and understanding; to my daughters, Róisín and Erin, for not playing on mummy's computer; and to my three-day-old son, Rory, who arrived just in time for this dedication.

Jacqueline Wilkie

For my mother and father – whose love and support have never failed me, and for my Glaswegian friend, Stanley.

Michael Johnson

To Raha and Imaon.

Reza Katebi

This page intentionally left blank



Contents

Dear Student	xi
1 Introduction: Control engineering is a part of our life	1
Wind turbine systems	3
Wastewater treatment plant control	4
Flight control systems	6
Coordinate measuring machines	8
Ship autopilot design	10
Hot strip rolling mills in the steel industry	11
Industrial heaters	14
2 Tools for the control engineer	16
How many ways of writing complex numbers are there?	17
What is the complex exponential and how do I use it?	18
I just want to know how to add, subtract, multiply and divide complex numbers. Is this easy?	20
Control engineers talk about transfer functions. What is a transfer function?	22
What are the magnitude or gain values of a complex number?	22
How do I work out the phase of a complex number?	22
I need to practise solving quadratic equations: are there some simple methods?	23
Parameter dependent complex numbers! That sounds hard – what are they?	24
How do I define a Laplace transform?	25
What is the Laplace variable s and the s -plane?	26
Where do the poles and zeros come from in a Laplace transform?	27
How do I use transform tables?	28
Oh dear, I need to work out Laplace transforms from first principles. How do I begin?	30
The transforms get even harder: what are the exponential–trigonometric signals?	32
When I multiply a signal by a constant, what happens to the Laplace transform?	34
I need to transform a combinations of signals. How do I do this?	34
What Laplace transform operations should I learn?	35
Why is a differentiator like multiplying by s ?	36
How do we represent an integrator using Laplace transforms?	37

I need to use Laplace transforms to represent a differential equation.	
How do I do this?	38
My lecturer says the system is linear. What does she mean?	39
I want to know what the superposition property is. Is it useful?	42
The tutorial sheet mentions causality and physical realisability.	
They sound difficult. What are they?	43
Can I use partial fractions to help me with Laplace transform derivations?	44
What is the Final Value Theorem?	45
How do I use the Initial Value Theorem?	46
How do I find the frequency content of a signal?	47
3 Software toolkit: MATLAB	52
3.1 Introduction to MATLAB	52
3.2 Starting MATLAB	53
3.3 Basic operations	54
3.4 Vectors	55
3.5 Vector manipulation	56
3.6 Polynomials	57
3.7 Matrices	58
3.8 Functions	59
3.9 Help window/tips	60
3.10 Plotting	61
3.11 Transfer functions in MATLAB	64
3.12 MATLAB environment	68
3.13 M-files and functions	70
3.14 SISO Design Tool: <code>r1tool</code>	75
General MATLAB commands	78
4 Software toolkit: Simulink	83
4.1 Using Simulink for analysis	83
4.2 Detailed house model	86
4.3 Building a simple Simulink model	91
4.4 Development and analysis of the house heating model	96
5 Modelling for control engineering	102
5.1 Signals, systems and block diagrams	103
5.2 Actuator–Process–Transducer device structure	106
5.3 Modelling summary	109
5.4 Chemical process engineering: liquid level control	110
5.5 Mechanical systems: model of a shaker table	122
5.6 Modelling of a manufacturing process component	135
6 Simple systems: first-order behaviour	147
6.1 First-order system model	150
6.2 First-order step response	151
6.3 Positive and negative step signals ('up and down' step signals)	154
6.4 Use of Simulink to find the step response	155
6.5 General first-order system time response	155

6.6	System parameters and system behaviour	158
6.7	Speed of response	163
6.8	Process time delays	165
6.9	Modelling of deadtime process	167
6.10	MATLAB function: pade	168
7	Simple systems: second-order systems	173
7.1	Second-order systems: model of a trailer suspension system	174
7.2	Second-order system parameters	177
7.3	Second-order transfer function forms	181
7.4	Solving general second-order equations	183
7.5	Modelling of second-order systems with deadtime	189
7.6	Simulink model of second-order system with deadtime	190
8	Feedback improves system performance	196
8.1	Open and closed-loop systems	197
8.2	Introducing feedback into control	200
8.3	Practical closed-loop control systems	202
8.4	Block diagram manipulation	204
8.5	Feedback changes the closed-loop performance	211
9	Design specifications on system time response	219
9.1	Design specifications: steady state and transient behaviour	220
9.2	Steady state performance	221
9.3	Transient performance	226
9.4	Specifications for disturbance rejection	233
10	Poles, zeros and system stability	242
10.1	Poles and zeros	243
10.2	System parameters and their relationship to pole locations	248
10.3	The link between pole position and system step response	251
10.4	How do the zeros of a transfer function model arise?	253
10.5	Further analysis and interpretation of the role of zeros in a system	257
10.6	Open- and closed-loop poles and zeros	259
10.7	What do we mean by bounded signals?	264
10.8	System stability	267
11	Three-term control: PID control	277
11.1	Controller assessment framework	279
11.2	Proportional control	282
11.3	Integral control	290
11.4	Derivative control	299
11.5	PI and PID controller formula	307
12	PID control: the background to simple tuning methods	312
12.1	Choice of controller structure	313
12.2	Manual tuning method	318
12.3	Proportional control of a system with a first-order model	321

12.4	Proportional and integral control of a system with a first-order model	327
12.5	Proportional and derivative control procedures	331
12.6	PID controller design by pole placement	340
13	Root locus for analysis and design	351
13.1	The relationship between the poles and system dynamic response: a summary	352
13.2	Introducing the root locus	353
13.3	Preliminary MATLAB root locus investigations	358
13.4	Some useful root locus rules	360
13.5	Second-order system performance: root locus contours	362
13.6	Effects of adding a pole or a zero to the root locus of a second-order system	364
13.7	Time delays and inverse response systems	368
13.8	Parameter root locus	370
13.9	Using MATLAB <code>r1tool</code> and <code>rlocus</code> routines	375
14	The frequency domain	383
14.1	Identification of magnitude and phase values from a sinusoidal signal	386
14.2	Frequency and logarithmic frequency scales	390
14.3	Presentation of gain and phase information	392
14.4	The frequency response and system features	397
14.5	Special frequency points	405
14.6	Interpretation of frequency response plot	409
14.7	Performance specification: gain and phase margins	410
15	Frequency response using Bode plot presentation	426
15.1	The Bode plot	428
15.2	Gain and phase calculation without using a computer	428
15.3	Using computers to form a frequency response plot	430
15.4	What are low, middle and high frequencies?	432
15.5	Transfer function components	434
15.6	Magnitude and phase of transfer function components	436
15.7	Introducing a sketching table	439
15.8	Elementary examples	445
15.9	Second-order underdamped system	451
15.10	Effect on gain and phase plots of increasing the damping ratio	454
15.11	Further examples using MATLAB plots	457
15.12	Magnitude plots of closed-loop and sensitivity transfer functions	461
16	Controller design using the Bode plot	469
16.1	Design specifications	470
16.2	Design example 1: proportional control with lag term added	471
16.3	Design example 2: PI control	477
16.4	Phase lag and phase lead elements	482
16.5	Phase lag controller	486

16.6	Phase lead control design	491
16.7	Design is iterative: a cautionary tale	498
16.8	Summary of the effects of phase lag and phase lead controllers on system responses	500
17	Analysis and simple design using the Nichols chart	505
17.1	Adding closed loop information to a Nichols plot	507
17.2	The Nichols chart	513
17.3	Design specifications on the Nichols chart	516
17.4	Reading gain and phase margins from the Nichols plot	517
17.5	Altering the loop gain to achieve design specifications	519
18	The practical aspects of PID control	529
18.1	Understanding common notation for industrial PID controllers	531
18.2	Industrial PID control technology	535
18.3	The issues in implementing an industrial PID controller	538
18.4	Integral wind-up and anti-wind-up circuits	539
18.5	Implementing the derivative term	545
18.6	Industrial PID controller structures	547
18.7	Different forms of industrial PID controllers	553
18.8	Reverse acting controllers	556
18.9	Digital PID control	565
19	PID controller tuning methods	577
19.1	Understanding PID tuning procedures	579
19.2	Process reaction curve PID tuning method	584
19.3	Sustained oscillation PID tuning	592
19.4	Damped oscillation or quarter amplitude decay PID tuning procedure	600
19.5	The relay experiment	605
19.6	Conclusions; or is the PID tuning problem solved?	613
20	Introducing a state variable description of a system	620
20.1	What is a state variable?	621
20.2	State vectors and matrices of coefficients	622
20.3	General procedure for writing a state variable representation	623
20.4	State variable diagram	629
20.5	MATLAB–Simulink representation of state variable models	630
20.6	Example of the development of a state variable model	632
20.7	State variable free and forced responses	635
20.8	Modelling and simulation in Simulink using an ABCD form	640
20.9	State variable model to transfer function model	643
20.10	MATLAB function <code>ss2tf</code> : state space to transfer function conversion	645
20.11	From transfer function to state space model	647
20.12	MATLAB command <code>tf2ss</code> : transfer function to state space conversion	653

21	Linearisation of systems from the real nonlinear world	659
21.1	What do we mean by <i>linear</i> ?	660
21.2	Nonlinear examples	662
21.3	Working regions and operating points	664
21.4	Linear approximation through Taylor series	667
21.5	Where do we apply linearisation?	668
21.6	Linearisation of a simple nonlinear dynamic equation	670
21.7	Linearising the model equations for a liquid level process	673
21.8	Linearisation of a more general nonlinear state variable model	675
21.9	Linearising a nonlinear state variable model to produce a linear ABCD model	676
22	Analysis of state variable systems	683
22.1	Matrix revision	684
22.2	Eigenvalues and eigenvectors	688
22.3	Poles, eigenvalues and system stability	691
22.4	More on state variable system time responses	696
22.5	Case study: Eigenvalues, eigenvectors and time responses	699
23	An introduction to control using state variable system models	710
23.1	State variable system structure	711
23.2	State variable controller structure	712
23.3	A state variable investigation of output feedback	713
23.4	Pole placement design with output feedback	716
23.5	Investigating state feedback: using the state vector directly	722
23.6	At the signpost of advanced control	733
	Answers to multiple choice questions	738
	Answers to selected questions	740
	Index	745
	MATLAB and Simulink index	751



Dear Student

As engineers, our interests lie in the specification, analysis, design, construction and testing of different systems. We often enjoy seeing an 'end product' to our efforts; this may be the part we played in the development of a new device, the software code that we wrote to upgrade a system or the report we constructed which analysed the possible failure modes for one part of a complex system. As *control* engineers, we find that many engineers from different disciplines need to design and implement control systems: the chemical engineer may need to provide level control for a tank of liquid; the manufacturing engineer may need to control the speed of a conveyor belt regardless of the load put upon it, the mechanical engineer may need to control the position of a machine's cutting tool; and the electrical engineer may need to maintain a constant electrical power output regardless of the number of electrical machines being used in the factory. To design these systems, we need to combine process and control engineering knowledge with design procedures, mathematical skills and analysis tools.

This book provides a first-level introduction to control engineering. Although we take a guided route through the basics of control, we have still covered sufficient material to enable you to understand the design of many simple control loops found in industry. The book can be considered to be in two halves: the first covers simple design in the *time domain*, while the second covers frequency domain material and introduces state variables and simple analysis for state variable models.

In the first part (see Figure 1), we start by providing three chapters on the analysis and software kits which are commonly used to represent control systems. Even if you have already covered some of the mathematics or software tools before, you can use these chapters for revision. We have not placed this material in an appendix because we believe that mathematics is a key skill in control engineering and the software tools we use are now a commonplace design aid even in industry. This emphasis on mathematical understanding is necessary because we use particular software (Simulink) models from an early stage of analysis and Simulink icons often include mathematical notation. However, we will find that these mathematical skills and the new software lead to easy-to-use 'graphical' control design packages and tools. MATLAB, Simulink and similar software are similar to a powerful calculator for today's control engineer.

We use our analysis tools for modelling simple industrial control systems. Here we are interested in introducing basic block diagrams and identifying the *actuator-process-transducer* model. We illustrate this modelling format for three processes from differing engineering disciplines – chemical, mechanical and manufacturing. Throughout the book we have used simple control examples from different engineering disciplines that do not require extensive knowledge of another engineering field. This is because control engineering is genuinely applied in many different fields and at many different levels. The mechanism for switching off an automatic kettle

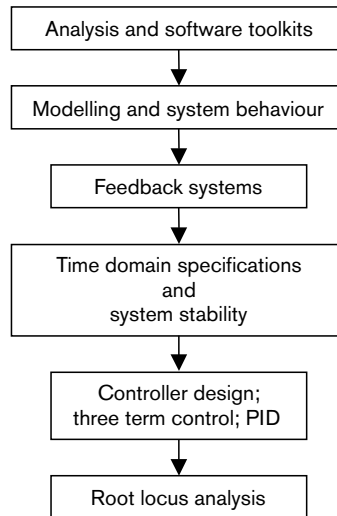


Figure 1 Route to simple control design.

once it has boiled and the autopilot on your luxury cruise liner (which you will enjoy as a successful control engineer!) are both examples of control systems.

Having produced some basic models we then look at how these systems behave. We examine first- and second-order models in detail, as these are the class of models most representative of the simple systems studied at this level. Indeed, many complex industrial systems can also be approximated by models of this form.

Once we have some idea of how these systems respond to input signals, we look at how we can change and improve a system's performance; for this we introduce the concept of feedback. We acquire some more skills when we study elementary closed loop block diagram analysis. Following on from this, we need to look at the performance we might require from a control system. How much do we allow the output to exceed the desired setpoint? Should the system respond slowly or quickly? We also need to introduce the important concept of the stability of systems and how this is related to features (*poles* and *zeros*) of the system model. We find that once the concept of a '*pole-zero*' map is understood, the plotting of the poles and zeros and determination of the system stability can be done easily using MATLAB commands.

We then look at designing a controller – a three-term controller or Proportional–Integral–Derivative (PID) controller. Since this controller is one of the most common controllers found in industry, we have made PID a strong theme in this book. We discuss the differences in the three terms and we provide some elementary manual tuning methods. At this place in the book we are almost at a breakpoint. As shown in Figure 1, we have learnt some analysis and looked at the modelling and behaviour of some simple systems. We now understand the benefits of feedback and can formulate some specifications for a control system. We can also design a suitable PID controller for the system. To complete our set of new skills and ideas, it remains to introduce a more theoretical chapter which ties together some of the ideas – we introduce the root locus technique.

Most of this first stage material has been presented in the *time domain*. This is because we can often understand more easily how a system behaves in time. It is sometimes less obvious to understand its behaviour in the *frequency domain*. The second part of this book (Figure 2) then

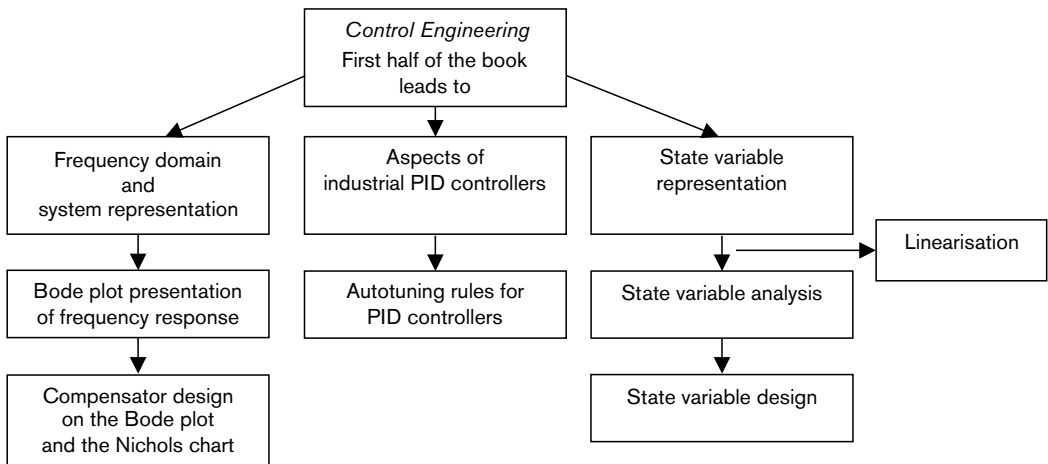


Figure 2 Outline of the second half of the book.

introduces the frequency domain and first of all reminds us how to read the logarithmic scales which are often found on system frequency plots.

System responses in the frequency domain require us to examine the ideas of system gain and phase and the graphical ways in which these features may be represented. We introduce the three common plots – Bode, Nyquist and Nichols – and learn to identify certain features on these plots. After this, we mainly use the Bode plot for control system design. If you are an electrical engineering student you might already have met frequency response plots when analysing the frequency response of simple analogue circuits.

After some elementary Bode plot analysis, we look at control design on the Bode plot and Nichols chart. We use the Bode plot for many control designs, but we find that for more complex systems, we can gain more information by using the other graphical techniques available to us: root locus, Nichols chart and time domain analysis.

It is important that your newly acquired control engineering skills should relate to common industrial practice. Therefore we look again at PID controllers, although this time we move away from the textbook representation of the earlier chapters and look at the practical aspects of controllers in an industrial context. This is followed by a chapter on the autotuning of PID controllers which highlights their advantages and disadvantages and the systems which would be appropriate for the three different autotuning methods discussed.

We end the book with several chapters on state variable models. State variable representation is an important way of expressing more complex or high-order control systems in a concise manner. Many system descriptions are given in state variable format, so it is appropriate at this level of textbook to ensure that we know how to use the representation and how to perform some elementary analysis. It is also often used in the MATLAB and Simulink software as one option for the representation of systems. Therefore we introduce state variable notation and the ABCD model format. We also present a separate chapter on linearisation. Many of the systems we deal with in the book are represented by linear models, but we acknowledge that the real world is nonlinear and we demonstrate the common linearisation procedure. Since state variable models are becoming more common due to the ease of analysis, design and implementation, we provide the elementary background in state variable model analysis and state feedback design in order to provide you with an initial set of skills in this design format.

It remains to say that we hope to meet you someday, sketching block diagrams and Bode plots on a notepad or entering control systems into your laptop and becoming excited about a control system that you have designed or had a part in designing. If you have any comments on the book or the exercises, we should be very pleased to hear from you, so do email us if you wish.

Best wishes

Jacqueline Wilkie, Michael Johnson and Reza Katebi
Glasgow, Scotland, UK
wjk@eee.strath.ac.uk

Some special features

There never seems to be enough time to do assignments; coursework seems to take forever and then those assessed tutorials! To help we have used some study and learning aids in this book. The future in many engineering areas will belong to software like MATLAB or Simulink, which are becoming industry standards, so we feel you should learn about these tools. We introduce them in Chapters 3 and 4. Throughout the book we have used software examples where appropriate. Examples using these software tools have been indicated with the

M

and

S

symbols respectively. Sometimes we have included smaller references to appropriate MATLAB or Simulink commands directly in the text where they represent an easy way of providing a solution. You should try out all these calculation and simulation opportunities. You will improve your MATLAB skills if you do.

Skill sections

In this book you will learn that control engineering is a combination of knowledge, skills and tools, but especially skills. Practice in these skills is essential and to help, we have marked certain areas as skill sections.

At the beginning of most chapters we have included a block diagram outline of the chapter with some genies (described below) to guide you through the book. We hope that this helps you find the important bits and lets you set aside the more demanding sections for that long train ride!



Trying things out/progress being made



Answering specific questions or discussing a particular problem



More advanced material



Skill or practice section



Read carefully: new knowledge at this point

The end of each chapter has three types of exercises; multiple choice questions for a quick test of your knowledge, practical skills questions where there are some simple rote exercises and finally problems which use the skills learnt in the chapter.

This page intentionally left blank



1

Introduction

Control engineering is a part of our life

Control engineering is an exciting discipline: without the advances in sensors, actuator devices and control design, space travel would be impossible, music systems would be mono, energy production would be smog-bound and computers would remain the territory of science fiction writers! Control engineering uses the measurements of process variables and the power of actuators to provide better control of a system's outputs. This could be the accurate positioning of components in manufacturing processes or the maintenance of disc speed in a DVD drive. The application of control theory and technology is so widespread today that you can hardly find a working engineering system around you without some form of control circuitry, logic or control procedure.

An example of a simple everyday control system is the electronic shower. If you choose a desired temperature for the water, you expect the water to be delivered at that temperature whatever the variations in the cold water flow caused by others in the household drawing off cold water from the system. We expect control systems to produce the desired outputs even when the process is subject to variations or disturbances. Apart from these fundamental performance requirements, we find that there are many other strategic reasons why control techniques are widely used in modern computer-based engineering systems.

Improvements in production processes

In today's highly competitive global economy, control engineers have to produce the best quality products at the lowest cost. Improving product quality and minimising cost in the manufacturing and process industries have always been major tasks for industrial control engineers. Why? We find that most industrial installations require huge amounts of capital expenditure to build. So every time a company improves product quality or reduces cost per unit, all the competitors have to follow suit or lose their market. To do this, companies cannot simply rebuild their plant; it would cost too much. So instead they call in the control engineer. Control engineers look at the control systems, at the latest advances in sensors, actuators, control methods and computer systems, and then recommend how plant performance can be improved for a much lower capital outlay. One day it may be necessary to rebuild the plant and introduce the next generation of production technology, but until that day arrives, as a control engineer you are likely to be

travelling the world improving and enhancing your company's factories and production facilities.

Improvement in quality of life

Our health services have undergone a radical change in recent years due to advances in control, sensors, actuators and information technology. Medical equipment and aids for the disabled have been two beneficiaries of these technology improvements. In the medical area, we find kidney dialysis machines, pacemakers, artificial hearts, automated procedures in operations, and automated treatment equipment all using control methods. For example, control engineering is helping anaesthesia procedures and radiation treatment to become more reliable, more accurate, and safer. Assistive technology is the field where we find control engineering helping to give the physically and sensory impaired person a better quality of life. Automatic stair lifts, motorised wheelchairs, digital hearing aid electronics and the voice-controlled computer mouse are all examples where understanding the principle of feedback is crucial to a successful design.

Improvements in natural resource usage

If we make a quick comparison of our manufacturing and process industries with the factories of 20 years ago we will find a dramatic change in working conditions and the technology of production processes. Computer control systems are widespread, and factories with several thousand manual workers and huge chimneys polluting the environment have long since disappeared. Control engineering and its technology can monitor the use of resources and energy in production systems. We can use this information to identify control loops that need re-tuning, or processes that need better control. We can look again at the performance levels that production should be achieving and compare (or benchmark, as it is called nowadays) these against our competitors to see if we need to introduce new control schemes altogether. Environmental control is also of critical importance in these audits. We can design control schemes to minimise the production of harmful emissions. As an example, we can cite gas turbine control schemes designed to reduce NO_x emissions. Even processes designed to eliminate harmful atmospheric gases like CO_2 scrubbers at coal-fired power stations will use control engineering principles.

Control engineering plays an important role in many other areas of industry, such as the processes in pharmaceutical plants, steel works, and the utility industries. Specialised control systems are also very important and these are often associated with particular products. For example, in cars we find engine management systems, in surface vessels we find ship autopilots, and sub-sea we find remotely operated vehicle (ROV) control systems. An aircraft control system is a vital control engineering application. Going just a little higher, much of our space technology would not be where it is now without control techniques and technology to help it stay in place! In the following sections we will give some more detail of examples of control engineering applications that we have worked with in our own careers or simply just admire. At the end of each case we identify the particular control objectives for the application and list some of the features that make this a demanding control problem.

Wind turbine systems

Wind turbine systems are probably the most widely applied renewable energy method for generating electric power. You may have seen one of the wind turbine farms that are becoming an increasingly familiar feature of the landscape in windy regions. The tower, hubs and blades of a wind turbine system undergo large stresses during operation. By designing an efficient and reliable control system, we can reduce these effects as well as provide improved energy capture from the wind.

For several centuries, windmills were a common sight in our countryside, and they were used for grinding corn or pumping water. In the 1900s they largely disappeared, but, in recent years, new types of windmill, or 'wind turbines', have been built and installed in groups, called wind farms, to convert wind energy into electricity.

These new developments have occurred because wind energy is currently viewed as one of the most promising forms of renewable energy sources. The turbines are deployed as a single unit, in small groups of four or five, or, increasingly, in wind farms of up to 50 machines. The most distinctive feature of a wind turbine is its rotor, with, typically, two or three blades (Figure 1.1 (a)). The aerodynamic forces, which are dependent on the incident air speed, act upon the blades and cause the rotor to turn. The power of a wind turbine is proportional to the swept area of its rotor. We can find machines of different size, from 600 mm diameter, rated at about 50 W and used for example for recharging batteries, to 60 m diameter, rated at about 3 MW, which can be connected to an electric power transmission network. The rotational speed is roughly inversely proportional to rotor diameter, so that blade tip speeds lie somewhere between 50 ms^{-1} and 100 ms^{-1} for large machines.

For fixed speed wind turbines, the rotational energy of the main shaft is passed to the generator, often through a gearbox and a high-speed shaft (Figure 1.1(b)). The shafts and associated gearboxes are referred to as the 'drive train'. The high-speed shaft is connected to a generator, which produces electrical energy. To connect a wind turbine to the grid, the rotor speed is controlled until the high-speed shaft speed reaches the grid frequency.

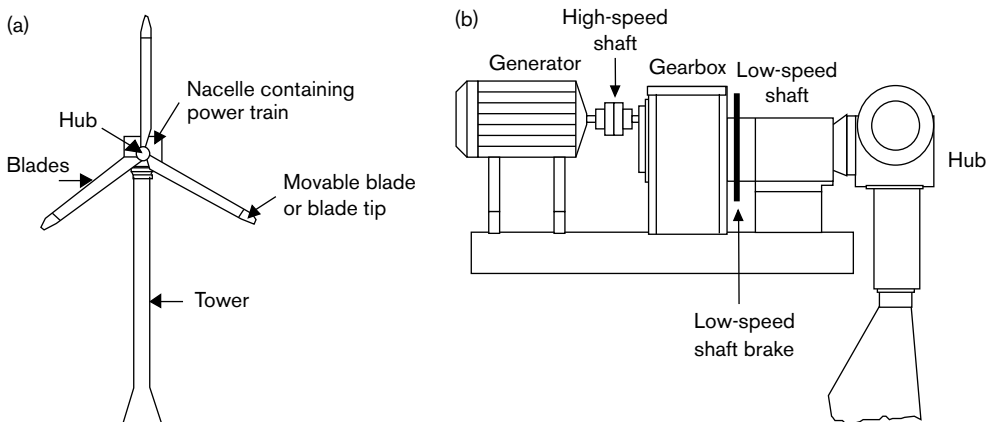


Figure 1.1 A diagram of an actively regulated wind turbine. (a) Horizontal axis wind turbine; (b) the power train.

The turbine is then connected to the grid and the speed of rotation is maintained by the grid frequency. As we can see, these control design and implementation problems call for control, mechanical and electrical engineering skills, but it is the control engineer who will integrate the various aspects to produce a working system.

Example of a control problem: Wind turbine systems

Control objective: To regulate the power output and to minimise the stresses and torque disturbances, also called the load disturbances by control engineers, on the rotor, the drive train and the tower installation.

Manipulated variable: Altering the angles of the blades – or the blade tips – to the incoming wind controls the rotational speed of the wind turbine.

Enabling technologies: System modelling, control design, mechanical system design, power electronics and electrical power systems knowledge.

Wastewater treatment plant control

Environmental concerns have had a large impact on the regulations for the effluent disposal from wastewater treatment plants. For example, recent European Union directives no longer allow the dumping of untreated sewage to sea. So, throughout Europe, operators of wastewater treatment plants have to ensure that sewage influent is treated to the highest standards before the effluent can be discharged to the environment. To meet these stringent new regulations, new advanced supervisory and control systems are being used in the wastewater treatment plants.

Wastewater treatment is a long-standing and essential task for many sectors of human activity. We use the term ‘wastewater treatment’ to cover the processing of effluent from agricultural activities and all forms of industrial and manufacturing processes as well as the disposal of domestic or urban sewage. In fact, many large industrial plants will have their own specialised wastewater treatment processes and plants. The particular process we shall describe here is that for the treatment of urban sewage. Urban sewage is characterised by three parameters: the Biochemical Oxygen Demand (BOD), the concentration of suspended solids, and the bacteriological content. Fairly limited instrumentation is available to monitor effluent quality before discharge, and advanced supervisory and control systems are needed to ensure that the effluent attains the quality values specified by EU directives and regulations.

An urban sewage wastewater treatment plant incorporates a number of treatment processes. We can describe these using a sequential process framework: Preliminary, Primary, Secondary, and Tertiary stages, as shown in Figure 1.2.

In the *preliminary* and *primary* stages, mechanical and hydraulic methods are used to extract and remove the larger sized particulate content from the incoming sewage, known as the influent. The preliminary processes usually include a mechanical grid filter to remove the large-scale debris, typically sanitary products and solid waste, from the influent. In the case of grit, long channels are used to settle the grit while leaving the

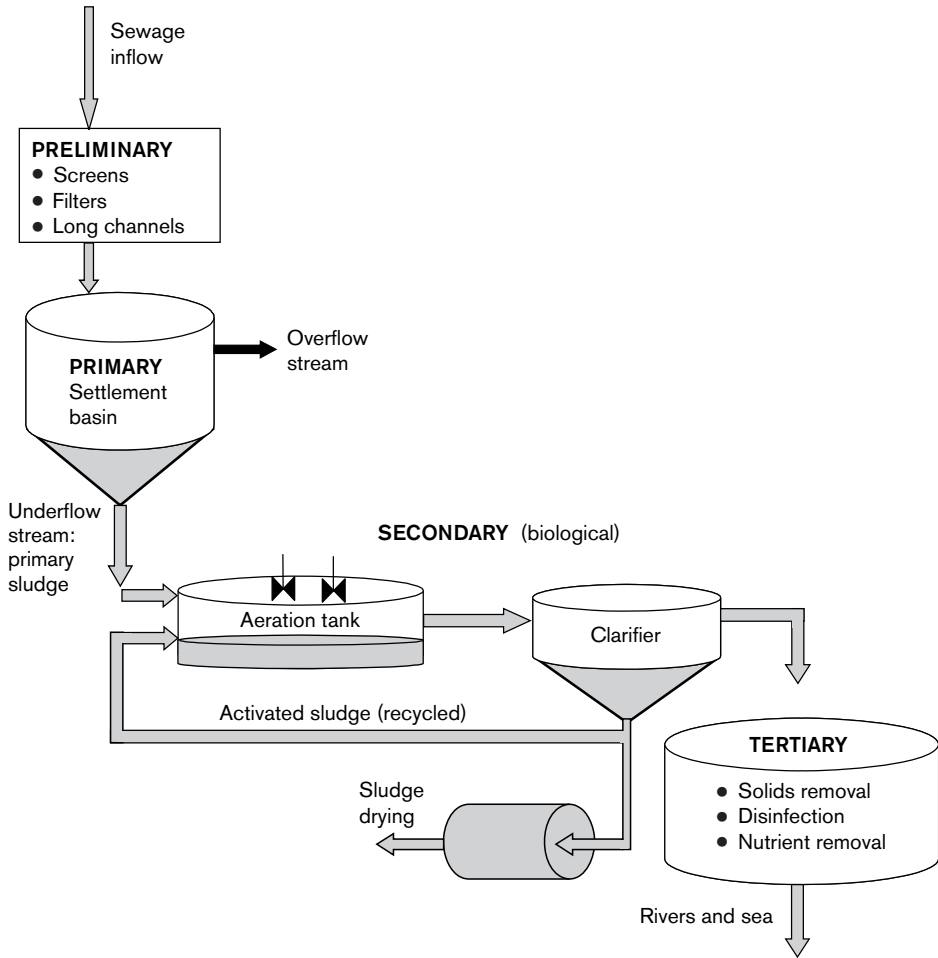


Figure 1.2 The stages in wastewater treatment.

organic material suspended. Moving on to the primary stage, large tanks are used to settle out some of the larger sized organic waste. These tanks are often termed the primary settlement tanks.

In the *secondary* processes, biotechnology comes to the fore and the processes to treat wastewater are really different ways of engineering a biological reactor. One very common secondary process is known as the *Activated Sludge Process*, where the waste in the sewage is transformed to sludge and separated from the clarified liquid. Aerated sewage is a sewage–liquid mixture that has been mechanically exposed to the atmosphere in order to increase the dissolved oxygen (DO) in the sewage mixture. This aerated mixture is then able to support a population of free-floating biological organisms that are able to use the organic matter and the dissolved oxygen in the sewage as a food source. Eventually the population of bio-organisms reaches a stage where the live and dead organisms form an activated sludge. The activated sludge process is the name given to the use of this sludge in a continuous bio-reactor to process the organic material content of urban

sewage. This continuous bio-reactor is engineered as a sludge recycle process comprising an aeration tank and a gravity clarifier tank. This can be seen in the middle part of Figure 1.2. The activated sludge is encouraged to grow in the aeration tank via the control of the dissolved oxygen content of the sewage–liquid mixture. This mixture is then transferred by flow to a gravity settler or clarifier, where the live and dead organic matter falls to the bottom and clear effluent is withdrawn. The sludge at the bottom of the clarifier is bled off for recycling or disposal. The part-sludge off-take which is fed back in the recycle path is used to maintain the growth of the biological population in the aeration tank. We recognise this recycle as an internal feedback mechanism in the process. The major part of the sludge off-take is, however, dried and then disposed of. If the control systems are working well, then the clear effluent may be directly discharged to the sea or river basin receiving waters. However, in some cases treatment by tertiary processes may be necessary.

The tertiary processes form a group of operations that take place at the end of the secondary wastewater treatment stage to achieve further treatment objectives. Sometimes these processes are called *polishing* processes because they are designed to raise effluent quality to one extra grade. Typically these tertiary processes might achieve finer suspended solids removal or specific nutrient (nitrate) removal.

Example of a control problem: Waste water treatment plant control

Control objective: To maintain the population level of micro-organisms in the aerated sludge tanks it is essential to control the concentration of dissolved oxygen (DO) in the sewage–liquid mixture in the tank.

Manipulated variable: The concentration level of dissolved oxygen in the sludge tank is controlled by switching on more aerators and changing the rotational speed of the individual aerator devices.

Enabling technologies: Analytical sensors to measure DO, online models to predict sludge and effluent quality variables, SCADA systems to provide good overall treatment plant control, and Programmable Logic Controllers (PLCs) to implement control loops at the individual process stages.

Flight control systems

Many of the most advanced control and instrumentation systems are found in the aerospace and space industries. Almost all of the basic manoeuvres performed by an aircraft, such as landing, take-off and directional control, are automated and implemented by computer systems. Highly accurate aircraft models allow the use of the most advanced control design techniques.

The Flight Control System (FCS) on any aircraft, whether a passenger jet or a high performance fighter, is a highly automated computerised system that incorporates almost all the control and monitoring functions. These will include the routine manoeuvres such as landing and take-off. A fully integrated flight control system maximises aircraft performance while ensuring crew and passenger safety. We find that such a system reduces the

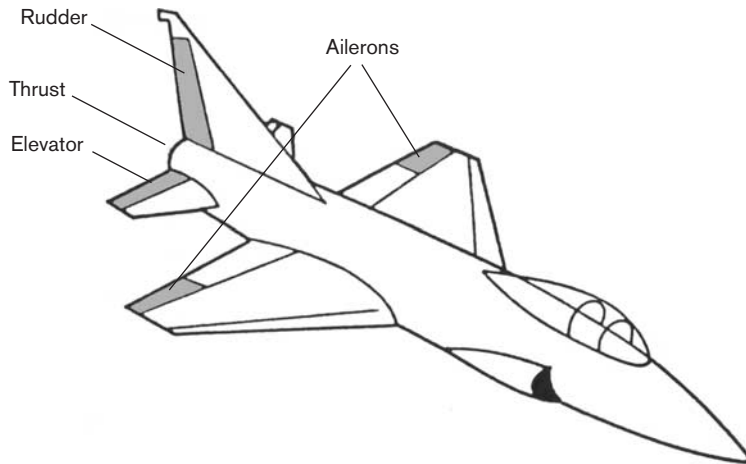


Figure 1.3 Flight control surfaces.

burden of decision making and eases the task faced by the human crew when piloting the increasingly complex aircraft designs being built. Aircraft like Concorde, Boeing 737, Eurofighter and so on would simply not be possible without a significant technological input from control engineers.

In military jet fighters, the advantage will lie with the aircraft that has the faster speed and the pilot who has the most manoeuvrability. To exploit these features, the pilot faces both aircraft structural limitations and aerodynamic constraints. We can take an example and look at the aircraft design. The pilot has three primary control surfaces to manipulate. These are the ailerons, rudder, and elevator. Each of the three primary control surfaces moves an aircraft about one of the three flight axes: the ailerons roll the aircraft about its longitudinal axis; the rudder turns the aircraft about its vertical axis and the elevator moves the aircraft about its lateral axis. We can see the flight control surfaces in Figure 1.3. A control system is required for each axis, to process the sensor measurements and to change the control surfaces to compensate for the effect of wind and other disturbances and keep the aircraft in the air.

In previous generations of fighter aircraft technology and at a slower speed, the human pilot functioned as an expert system and learnt to manipulate and coordinate some of these flight surfaces *instinctively* to achieve the sort of manoeuvrability that would win a duel of fighters in the air. But from Figure 1.3, we can see immediately that any human pilot is going to have great difficulty doing all the computations necessary (even instinctively) to move a complicated set of flight control surfaces for a fighter aircraft. Indeed, it might be not be too surprising to learn that a fighting advantage can be obtained by making jet fighter manoeuvres unstable for a very short period of time before moving back into the stable flight regime again! This is the only case we have ever come across where we design for an unstable closed-loop system, and we can only be thankful that passenger jet aircraft controls are not designed this way otherwise every journey would be just too exciting for most of us.

In a fighter aircraft, the various types of manoeuvres will be identified and categorised. For each type of manoeuvre a different control strategy will be devised. In fighting mode

the pilot can demand an evasive or attacking manoeuvre, knowing that the flight control computers will perform all the necessary computations to coordinate and move the flight control surfaces to deliver the manoeuvre demanded by the pilot. The very same computers will be ensuring that the aircraft response will be within the limits of a safe operating flight envelope. The flight control system will incorporate models of the aircraft and other flight formulas that will use onboard sensors measuring flight parameters, such as speed, acceleration, altitude, aircraft configuration (noting features like how much of the aircraft remains operational), aircraft mass and balance. These models will then be used to define the current structural and aerodynamic limits and if necessary limit the response of the aircraft to pilot demands. This will prevent the pilot from overstressing the aircraft structure or flying in a way that leads to completely uncontrolled flight while in the middle of combat activities.

The autopilot is the name we give to the control technology used in aircraft to keep to a simple heading direction in long-range cruising mode. In a fighter like the Eurofighter, the autopilot can be used to help the pilot in tactical situations to provide basic track, heading, altitude and airspeed modes. Even optimum attack profiles can be implemented automatically. More advanced modes such as auto-climb, auto-attack and auto-approach control systems are also integrated with the autopilot system.

Example of a control problem: Flight control systems

Control objective: To keep the aircraft in a horizontal position and maintain a desired heading.

Manipulated variable: Flight control surfaces.

Enabling technologies: Onboard smart sensors to measure aeronautical variables. Advanced electro-hydraulic actuation equipment, aircraft models and formulas, advanced control algorithms, control and manoeuvre visualisation, and computer systems.

Coordinate measuring machines

A coordinate measuring machine is a robotic machine with high-precision movement in three dimensions which is able to take measurements to a fraction of a micron on machined products. These machines are used for quality control in the automotive, aerospace and other component manufacturing industries. The precise control of the arm positions in three directions is essential for operation of the system.

Coordinate measuring machines (CMMs) are reliable and robust tools for obtaining measurements at micrometre accuracy on machined products. Machined products are invariably metal goods cut and ground to a high precision by numerically controlled machine tools. Typically, we find that these products are used in a wide range of high-precision applications, from gas turbine components to spacecraft construction. Speed is of the essence in the production and manufacturing cycle for these machined product. The coordinate measuring machine is needed to ensure that the manufacturing process is producing goods which have the requisite dimensional accuracy. We have to be able to identify when the production setup is going astray and prevent the production of any off-

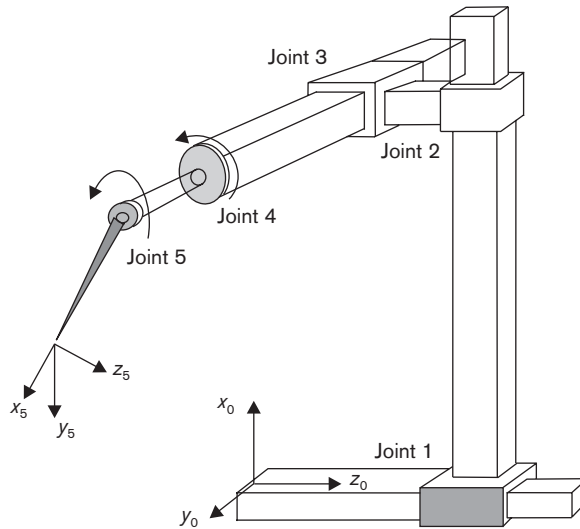


Figure 1.4 The coordinate measuring machine.

spec. products. We also have to be able to prove to our customers that they are actually receiving goods that meet the specification agreed on the order form.

The demand for enhanced product quality and reduced manufacturing cycle time has continued in recent years. This has meant that the coordinate measuring machine has to be more accurate and faster in operation than ever before. Originally built from mechanical and analogue components, the advent of high-speed computing machines and the development of high-power electronic components have made it possible to improve the speed and accuracy of the modern coordinate measuring machines significantly.

The elementary components of a coordinate measuring machine are a table for mounting the workpiece, a set of mechanical linked arms, a probe for component location, contact/measuring scales to determine the component position and the power drive systems. As shown in Figure 1.4, there are three orthogonal arms in a coordinate measuring machine: Joint 1 to Joint 2, Joint 2 to Joint 3 and Joint 3 to Joint 4. The probe at the end of Joint 5 is capable of rotating about the $(xyz)_5$ axis. The tip of the probe operates like a switch and when it touches a surface, it sends back a signal to be recorded by the coordinate measuring machine supervisory control computer.

If we wished to measure the length of a workpiece to micrometre accuracy, we would first set a target starting position in the supervisory computer. This starting position is a small distance from the actual edge of the workpiece. The probe moves to this target position at the highest acceleration and velocity possible. From this target position the probe is then moved at a slow constant speed until contact is made with the edge of the workpiece. A triggering mechanism in the probe sends a signal to indicate the actual initial edge position. From there the probe tracks the workpiece until it reaches the other edge, when it sends a signal to indicate the end point of the product length. The coordinate measuring machine requires a supervisory control system which has been programmed to measure the specific product. We can imagine that for a complicated, non-flat machined product we would need some complex programming to store away the shape being measured.

The success of the high-level control supervisor depends on the performance of the low-level control loops; that is, those loops controlling the basic arm servo devices, and variables like current, velocity and position. The control problem is complicated by a requirement for high positioning accuracy. It is also necessary to ensure that the control system does not use motions that trigger even the smallest structural vibrations of the coordinate measurement machine arms, since this would deteriorate the measurement accuracy.

Example of a control problem: Coordinate measuring machine

Control objective: To move the coordinate measurement machine probe to a desired position quickly and accurately.

Manipulated variable: Current in motor drives.

Enabling technologies: Measurement sensors, power drives, trajectory planning at the supervisory control system level, and interacting control loop designs.

Ship autopilot design

Proportional–Integral–Derivative (PID) controllers are common in industry today. They are also called three-term controllers, since the controller output is the combination of terms relating to a proportion of the error, integral of the error and derivative of the error in a feedback system. The PID controller was first implemented in analogue form to maintain the heading of a ship. Today, digital control systems prevail, but they still use the traditional PID format for the controller design. Most ships also have additional control systems to compensate for ship motions such as roll, yaw and pitch.

If you have travelled on a passenger ferry you will remember that due to the environment disturbances, like wind, wave and sea current, the ferry and you will experience roll, yaw and heave. The more severe the inclement weather, the more you will have experienced these phenomena, which can be quite unpleasant. In Figure 1.5 we can find the precise details of the coordinate systems and the disruptive directions of ship motions. We have designed different types of control systems to reduce and compensate for the effects of these disturbances and make the sea transport more safe and comfortable. For example, fin-roll stabilisers and their control systems are used to reduce ship roll effects, while ship autopilots are used for heading control problems.

Ship autopilots have two modes of operation: course keeping and course changing. In course keeping the control objective is to maintain a fixed heading. The main actuator is the rudder, whose angle can be varied to change the heading of the ship. The heading angle is measured using a gyrocompass and the control objective is to keep the ship on a specified heading or a specified path. In this situation the controller must compensate for the disturbance effects of wind, waves and ship loading. At the same time, the high-frequency motion of the ship caused by the waves and detected through the gyrocompass must be blocked to minimise the rudder activity and wear and tear on the steering mechanism. By way of contrast, the control objective in course changing is to perform a manoeuvre quickly and accurately. The manoeuvre should have a clear start and it should

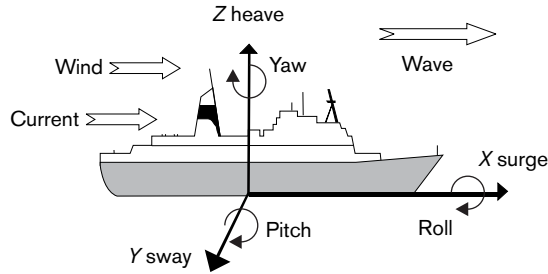


Figure 1.5 Sea-going vessel.

be completed without overshoot. This ‘no overshoot’ performance specification is an interesting one which will be met again as a critically damped response later in this book.

The problems of ship autopilot design are classical examples of control engineering applications. The performance specifications are given in both the time and frequency domains. The design has to accommodate a certain amount of model inaccuracy, because we do not have exceptionally accurate models of ship motion. The controllers to be used must be simple because all the commissioning and testing takes place at sea, which provides a rather hostile environment for engineering work and few repeatable sets of test conditions. We find that the controller methods are classical three-term or PID controllers. Such a PID controller solution was first installed on a Russian ship. Even today, fin stabiliser control systems on ferries and modern luxury cruise liners also use PID technology. This type of controller has been used throughout many engineering disciplines, and motivated by this widespread industrial use, we will study its structure and performance in detail in this book.

Example of a control problem: Ship autopilot design

Control objective: To reduce the roll motions on a cruise vessel by using high-frequency commands to the rudder while still using the low-frequency commands for course keeping.

Manipulated variable: Hydraulic systems controlling the rudder.

Enabling technologies: Roll sensors, advanced control algorithms and digital computer control systems.

Hot strip rolling mills in the steel industry

The modern hot strip rolling mill is an example of a complex production facility that uses advanced control and instrumentation technology. The slab of steel enters the mill at 25 cm thickness and leaves the last stand at 2.5 mm, a reduction by a factor of 100. It takes about twenty minutes to walk from one end of the process to the other, so it is a very large-scale industrial installation.

A hot strip rolling mill produces rolled steel sheet. This sheet is used in car bodies and white goods like washing machines. So the next time that you turn on your washing

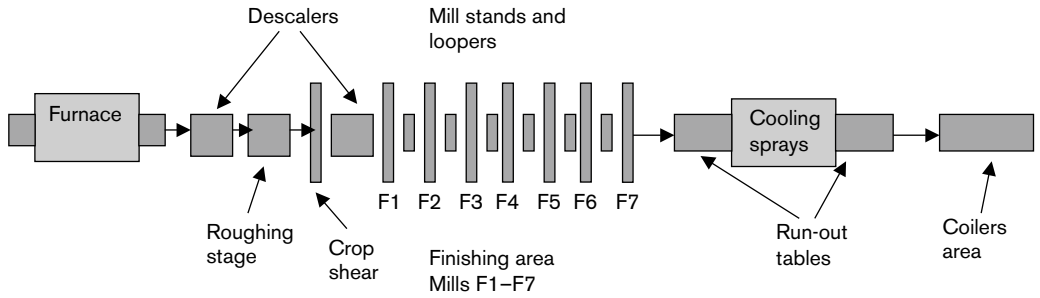


Figure 1.6 Hot strip rolling process.

machine, just think of the steps that the steel shell of your washing machine has been through before it ended up in your kitchen. Passing through the hot strip mill is only one of these. The whole hot strip mill process sequence can be found in Figure 1.6. In a hot strip mill the sheet product starts out as a slab at the furnaces with dimensions 25 cm thick, 9 m to 12 m in length and between 600 mm and 1600 mm in width, and with a temperature of around 1320 °C. At the finish, the hot rolling process will have changed the slab into a steel strip about 1 km long and 1.5 to 2.5 mm thick.

We can use Figure 1.6 to track the progress of the slab as it passes through the hot strip rolling mill.

1. *Descaling process*

On leaving the furnace, the slab undergoes a descaling process. There are two stages. Initially the slab goes through a vertical rolling operation. This breaks up the surface scale, which is then washed away by strong jets of water in the secondary stage.

2. *Roughing stage*

The thickness of the slab is then reduced by a factor of 10 to about 3 cm in thickness. This is the first rolling process, which gets the slab into a ‘rough’ strip shape ready for the finishing rolling process. Because the mill produces a ‘rough’ strip shape this rolling mill is called a roughing mill.

3. *Crop shear*

From the roughing mill, the slab passes along a length of roller table to a crop shear that trims the ends of the now much thinner slab.

4. *Second descaling process*

A second scale breaker process finishes this first stage of the hot strip mill process.

5. *Hot strip rolling process (finishing area)*

In the middle part of the hot strip rolling process, the slab enters the hot strip finishing mill. The hot strip finishing mill usually has from four to seven mill stands operating in tandem. The mill in Figure 1.6 is shown with seven stands, labelled F1 to F7. A roll stand (Figure 1.7) is a set of stacked rollers that have their roll speed and roll gap width controlled to impart a desired thickness to the steel strip at each stand. A mechanical arm, called a looper, supports the hot strip between the stands. The looper is used to keep the strip tension constant. At each mill stand there are very few instruments to measure the main strip variables of thickness (gauge), profile, temperature and width.

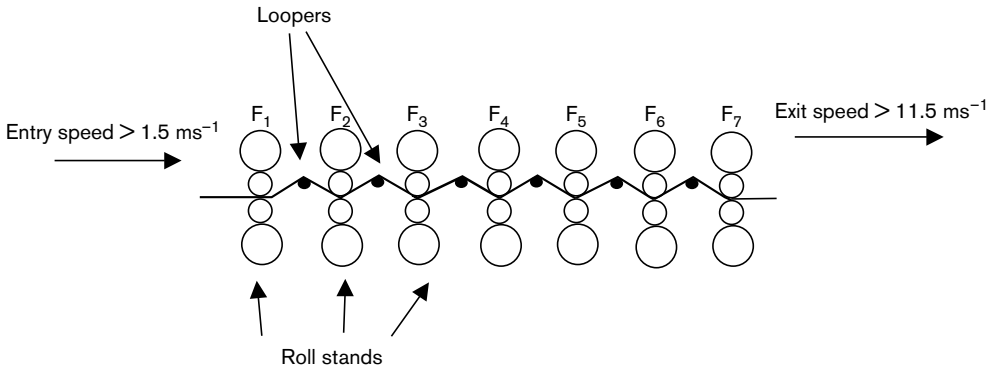


Figure 1.7 Roll stand and looper in a hot strip finishing mill.

The roll stand speed derives from some very large roll drive motors, one per stand. These are rated at about 5–6 MW and they generate the torque to transport and compress the strip. Each stand has a load cell sensor to measure the roll force, and of course there is a measurement of roll speed available. But, as with so many industrial control problems, we cannot actually directly measure the variable that we wish to control, which in this case is the thickness of the steel strip. At the inter-stand looper, we wish to control strip tension; again, this is a variable that we cannot measure directly. It is problems like this that make control engineering such a fascinating and challenging discipline.

6. Run out tables

On leaving the finishing mill, the strip is moved along on a conveyor belt consisting of a set of rollers, which move it at a linear speed dependent on the section it is in. As the steel becomes thinner and longer, the roller table has to become faster towards the end of the process. At this point metallurgical requirements also make an appearance. The steel has to be rolled within a certain temperature range around 800 °C so that it can be cooled at a specified rate before coiling. The cooling process occurs at the so-called run-out table. The run-out table has a distributed sprinkler system designed to cause the desired controlled cooling of the strip down to a temperature of about 350 °C.

7. Coilers area

At the very end of the process the finished strip is coiled ready for further processing in a cold strip rolling mill. Needless to say, the cold mill is a story saved for another book!

Example of a control problem: Hotstrip rolling mill

Control objective: By controlling the roll force and the strip speed, the strip gauge can then be controlled to a desired thickness.

Manipulated variable: Roll gap setting and current in motor drives.

Enabling technologies: Coordinated supervisory computer control system, advanced multi-loop control algorithms, process models in control systems, and robust sensor technology for harsh industrial environments.

Industrial heaters

Industrial heaters are extremely large-scale furnaces that act as heat suppliers for chemical feed stock (the materials entering the chemical process and pharmaceutical industries). The fuel used may be oil or gas, and it is used at very high rates. These heaters require control and monitoring of internal and external variables in order to minimise fuel consumption, reduce furnace maintenance and improve heat transfer. With this type of application the chemical and petrochemical industry is a major application area for control engineering.

The rapid growth of modern chemical and petrochemical industries has led to close scrutiny of fuel consumption and the level of CO₂ emissions to the atmosphere within the industry. The benchmarking of process control units to discover where it is possible to save costs and reduce environmental pollution has been widespread in the industry. Control engineering is an enabling technology for achieving better and more efficient control of chemical and petrochemical processes. An industrial heater burning either oil or gas, as shown in Figure 1.8, is an example of a common petrochemical process that consumes a great deal of energy. This type of heater is usually used as a heat exchanger for a down stream process.

We have two control priorities in this system: firstly, to maintain the outlet temperature of the feedstock liquid with as little fluctuation as possible, and secondly, to reduce the oxygen percentage in the exhaust gases in the duct. Since the thermal efficiency of the heater depends on the outgoing percentage oxygen concentration, the lower the percentage, the higher the thermal efficiency. To achieve these objectives, we can vary the oil fuel flow rate to the burners and the change the orientation of the *damper* in the exhaust duct. Reducing the rate of oil flow reduces the outlet feedstock temperature but increases the percentage of oxygen in the duct. This in turn will reduce the thermal efficiency of the heater. On the other hand the outlet temperature will increase as the damper is closed and the oxygen percentage will also reduce. We are faced with designing a control system to balance the effect of changing the fuel oil flow and the damper position on the outlet temperature and efficiency of the boiler. As a result, we have a complex control design problem in which we have to balance several performance objectives. We

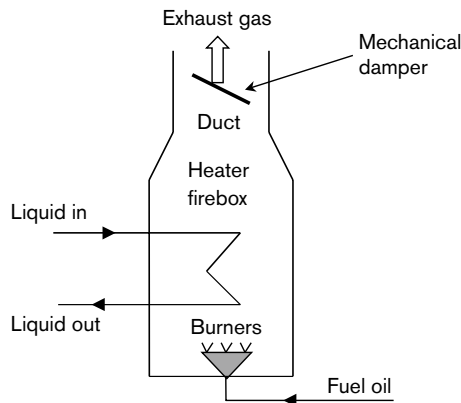


Figure 1.8 An industrial heater burning oil fuel.

call this type of problem a *multivariable* control problem due to the interactive nature of the system inputs on the output variables, in this case, the feedstock output temperature and percentage oxygen.

Example of a control problem: Industrial heater control

Control objective: To control the temperature of the outgoing liquid and the oxygen content in the duct exhaust gases.

Manipulated variable: Fuel oil flow and position of control damper.

Enabling technologies: Sensors for measuring percentage of oxygen and other exhaust gases, flow meters, valve technology, control systems for processes with multivariable objectives, and processes with actuator limitations.

Onwards

From the many examples we have described here we find that control engineering is an important feature in many aspects of our lives. Putting it simply, control is everywhere: in household goods themselves, in the manufacture of the material from which we make household goods, and in the domestic environment with which we surround ourselves. In the wider picture, our ability to measure and then control temperature, velocity and energy use leads directly to the urban, industrial and transport infrastructure on which our society depends. In the lifestyle of today, we rely heavily on technology and its applications. Control engineers are needed to analyse these technological systems and provide the crucial control systems without which modern society would simply not be possible. This book is a first stage control book that brings together all the key elements in an easy manner to be able to analyse and design effective controllers for simple systems.

2

Tools for the control engineer Analysis kit

Today's engineer is a problem-solver, and the engineering method splits into problem analysis and design followed by solution implementation. More and more the engineer must:

- Decide upon the right questions to be asked.
- Formulate a scientific and engineering description of the questions.
- Use appropriate analysis and software tools to solve the problem.
- Engineer and implement the solution.

The control engineer needs *analysis tools* to prepare a precise problem formulation and *software tools* to find a solution to the problem posed. Logical reasoning is needed by the engineer to ensure sound judgement and to work with some sophisticated mathematical methods. In control engineering many of the ideas, concepts and mathematical tools are advanced, *but* software is available to make the use of these tools *easy*.

The control engineer must not be fazed by the use of precision mathematics and brilliant software packages. What is important is being able to concentrate on the real issues of the control problem. The *toolkit of analysis methods* for control engineering is small, but should be thoroughly understood and practised. The software kit used – products like MATLAB (Chapter 3) for example – should be treated like a sophisticated calculator. What is important is the control problem to be formulated and what the solution says about the control issues being studied. In this chapter we describe a small set of tools which form a Control Engineer's Analysis Toolkit. We can use this chapter either to learn about and practise using the tools or simply to refer to the appropriate section when we need to.

We can even work our way through the whole chapter on the grounds that the key to understanding the theoretical aspects of control engineering and system dynamics is a thorough knowledge of complex numbers and Laplace Transforms. In this case we need to organise our work around the five steps that are needed to acquire a sound knowledge of Laplace transforms for control engineering studies. These steps are shown in Figure 2.1.

The chapter is ordered according to these steps, but we have set it in the form of a set of frequently asked questions on the analysis found in control engineering studies. In this way we can also use the chapter to answer worrying questions.

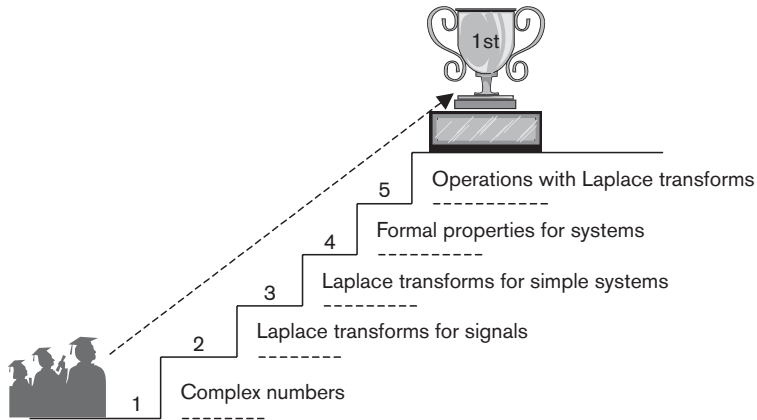


Figure 2.1 The five steps to success with Laplace transforms.

Learning objectives

- To revise complex number operations.
- To introduce the Laplace transform for signals and systems.
- To introduce an alternative Laplace transform representation for differentiation and integration.
- To examine some formal Laplace transform manipulations.

Step 1 Complex numbers



How many ways of writing complex numbers are there?

There are three representations for complex numbers, these are called *Cartesian*, *polar* and *complex exponential*.

Cartesian (or rectangular) representation:

$$z = a + jb \quad \text{where } a \text{ and } b \text{ are real numbers}$$

Polar representation:

$$z = r(\cos\theta + j \sin\theta) \quad \text{where } r \geq 0 \text{ and } -\pi < \theta \leq \pi$$

Complex exponential representation:

$$z = re^{j\theta}$$

We should note that we can move from one representation to another and also that we have two *graphical* interpretations available for use: the Cartesian and the polar versions. We can represent $z = a + jb$ using the Cartesian plane, as shown in Figure 2.2.

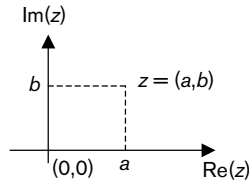


Figure 2.2 Cartesian representation for $z = a + jb$.

We identify the real part of z using the notation $a = \text{Re}\{z\}$ and for the imaginary part of z we use $b = \text{Im}\{z\}$. One Cartesian property is the distance between the origin and the point z ; this length is called the modulus, denoted $|z|$, and is given by:

$$|z| = \sqrt{a^2 + b^2}$$

We can now impose the trigonometric relations onto the Cartesian framework to obtain the polar representation for z . Using Figure 2.3 we use the polar angle, θ , where $-\pi < \theta \leq \pi$ and the radial length $r = |z|$.

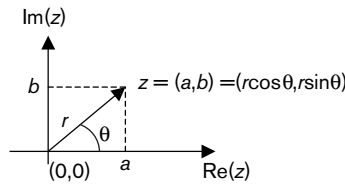


Figure 2.3 Polar representation of $z = a + jb$.

We have $r = |z| = \sqrt{a^2 + b^2}$, and from Figure 2.3,

$$a = r \cos\theta \text{ and } b = r \sin\theta$$

so that $z = a + jb$

and $z = r \cos\theta + jr \sin\theta$

giving $z = r(\cos\theta + j \sin\theta)$ where $r \geq 0$ and $-\pi < \theta \leq \pi$.

This is the familiar polar representation for a complex number z .

Q

What is the complex exponential and how do I use it?

The link between the polar form for z and the complex exponential is established formally by the use of series for e^x , $\cos x$ and $\sin x$. Any student mathematical handbook gives:

$$e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \dots + \frac{x^n}{n!} + \dots$$

$$\cos x = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \dots + (-1)^n \frac{x^{2n}}{(2n)!} + \dots$$

$$\sin x = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \dots + (-1)^n \frac{x^{2n+1}}{(2n+1)!} + \dots$$

Let us now evaluate e^x for $x = j\theta$; hence:

$$e^{j\theta} = 1 + j\theta - \frac{\theta^2}{2!} - j\frac{\theta^3}{3!} + \frac{\theta^4}{4!} + j\frac{\theta^5}{5!} + \dots$$

and if we partition this, we obtain

$$e^{j\theta} = \left\{ 1 - \frac{\theta^2}{2!} + \frac{\theta^4}{4!} - \dots \right\} + j \left\{ \theta - \frac{\theta^3}{3!} + \frac{\theta^5}{5!} - \dots \right\}$$

and using the series for cosine and sine, we obtain

$$e^{j\theta} = \cos\theta + j \sin\theta$$

The link is now almost established, since the polar representation for a complex number is

$$z = r(\cos\theta + j \sin\theta)$$

and this can now be written in complex exponential form as

$$z = re^{j\theta} \quad r \geq 0 \text{ and } -\pi < \theta \leq \pi$$

As an exercise, if e^x is similarly evaluated for $x = -j\theta$ you should be able to show:

$$e^{-j\theta} = \cos\theta - j \sin\theta$$

We show some properties and different ways to use the complex exponential in Table 2.1. For example, we will find that the power rule is used quite frequently.

Example Two complex numbers z_1 and z_2 have unity modulus. If the polar angle of z_1 is $\theta_1 = +25^\circ$ and the polar angle of z_2 is $\theta_2 = -1.396$ rad, give the complex exponential form for $z = z_1 z_2$.

Solution The general exponential form is $z = re^{j\theta}$. Thus, z_1 has $r = 1$ and $\theta_1 = 25^\circ$. We change θ_1 to radians to give $\theta_1 = (25/180) \times \pi \text{ rad} = 0.436 \text{ rad}$. Then $z_1 = e^{j0.436}$. The complex number z_2 has $r = 1$ and $\theta_2 = -1.396 \text{ rad}$. Thus $z_2 = e^{-j1.396}$. We use the power rule to find $z = z_1 z_2$ as follows:

$$z = z_1 z_2 = e^{j0.436} e^{-j1.396} = e^{j(0.436 - 1.396)} = e^{-j0.96}$$

Table 2.1 Useful properties of the complex exponential.

1	Unit modulus	$ e^{j\theta} = 1$
2	Inverse exponential	$e^{j\theta} e^{-j\theta} = 1$
3	Power rule	$e^{j\theta_1} e^{j\theta_2} = e^{j(\theta_1 + \theta_2)}$
4	The Euler formulas	$\cos\theta = \frac{1}{2}(e^{j\theta} + e^{-j\theta})$ and $\sin\theta = \frac{1}{2j}(e^{j\theta} - e^{-j\theta})$



I just want to know how to add, subtract, multiply and divide complex numbers. Is this easy?

Addition

The addition of two or more complex numbers is easiest in the Cartesian form. Consider two complex numbers:

$$z_1 = a + jb \text{ and } z_2 = c + jd$$

Then their sum $z = z_1 + z_2$ is given by

$$\begin{aligned} z &= z_1 + z_2 \\ &= (a + jb) + (c + jd) \\ &= (a + c) + j(b + d) \end{aligned}$$

Subtraction

The subtraction of two or more complex numbers is easiest in the Cartesian form. Consider two complex numbers:

$$z_1 = a + jb \text{ and } z_2 = c + jd$$

Then their difference $z = z_1 - z_2$ is given by

$$\begin{aligned} z &= z_1 - z_2 \\ &= (a + jb) - (c + jd) \\ &= (a - c) + j(b - d) \end{aligned}$$

Example: Addition

$$z_1 = 4 + j9 \text{ and } z_2 = 5 + j(-5)$$

Their sum, $z = z_1 + z_2$ is given by

$$\begin{aligned} z &= z_1 + z_2 \\ &= (4 + j9) + (5 + j(-5)) \\ &= (4 + 5) + j(9 - 5) \\ &= 9 + j4 \end{aligned}$$

Example: Subtraction

$$z_1 = 4 + j9 \text{ and } z_2 = 5 + j(-5)$$

Their difference, $z = z_1 - z_2$ is given by

$$\begin{aligned} z &= z_1 - z_2 \\ &= (4 + j9) - (5 + j(-5)) \\ &= (4 - 5) + j(9 + 5) \\ &= -1 + j14 \end{aligned}$$

Multiplication

The multiplication of two or more complex numbers is easiest in the polar form. Consider two complex numbers:

$$z_1 = r_1 e^{j\theta_1} \text{ and } z_2 = r_2 e^{j\theta_2}$$

Then their product $z = z_1 z_2$ is given by

$$\begin{aligned} z &= z_1 z_2 \\ &= r_1 e^{j\theta_1} r_2 e^{j\theta_2} \\ &= (r_1 r_2) e^{j(\theta_1 + \theta_2)} \\ &= r e^{j\theta}, \text{ where } r = (r_1 r_2) \text{ and } \theta = (\theta_1 + \theta_2) \end{aligned}$$

Division

The division of two or more complex numbers is easiest in the polar form. Consider two complex numbers:

$$z_1 = r_1 e^{j\theta_1} \text{ and } z_2 = r_2 e^{j\theta_2}$$

Then their ratio $z = z_1 / z_2$ is given by

$$\begin{aligned} z &= z_1 / z_2 \\ &= \frac{r_1 e^{j\theta_1}}{r_2 e^{j\theta_2}} = \left(\frac{r_1}{r_2} \right) e^{j\theta_1} e^{-j\theta_2} \\ &= \left(\frac{r_1}{r_2} \right) e^{j(\theta_1 - \theta_2)} \\ &= r e^{j\theta}, \text{ where } r = (r_1 / r_2) \text{ and } \theta = (\theta_1 - \theta_2) \end{aligned}$$

Example: Multiplication

$$z_1 = 2e^{j0.1} \text{ and } z_2 = 3.5e^{j1.4}$$

Their product, $z = z_1 z_2$, is given by

$$\begin{aligned} z &= z_1 z_2 = 2e^{j0.1} 3.5e^{j1.4} \\ &= 7e^{j1.5} \end{aligned}$$

and if $z = a + jb$, $a = r \cos \theta$, $b = r \sin \theta$, then,

$$z = 0.49 + j6.98$$

Example: Division

$$z_1 = 2e^{j0.1} \text{ and } z_2 = 3.5e^{j1.4}$$

Their ratio, $z = z_1 / z_2$, is given by

$$\begin{aligned} z &= z_1 / z_2 = \frac{2e^{j0.1}}{3.5e^{j1.4}} \\ &= 0.57e^{-j1.3} \end{aligned}$$

and if $z = a + jb$, $a = r \cos \theta$, $b = r \sin \theta$, then,

$$z = 0.15 - j0.55$$

Q Control engineers talk about transfer functions. What is a transfer function?

As control engineers, we will often meet expressions of the form

$$G(s) = \frac{n(s)}{d(s)}$$

where s is a complex variable which is written as

$$s = \sigma + j\omega$$

with a real component σ and an imaginary component ω , or frequency. The terms $n(s)$ and $d(s)$ are numerator and denominator polynomials in s . The function $G(s)$ is commonly referred to as a transfer function, since it will determine how the information or energy in the input signal is *transferred* to the output signal.

Q What are the magnitude or gain values of a complex number?

The magnitude or gain are terms we often apply to the modulus of complex numbers. For example, we might have that a transfer function has been worked out for a frequency of $\omega = 2 \text{ rad s}^{-1}$ to give:

$$G(j2) = \frac{3+j4}{2+j3}$$

The magnitude or gain of the complex number $G(j2)$ is simply its modulus and we calculate it as follows:

$$|G(j2)| = \frac{|3+j4|}{|2+j3|} = \frac{|3+j4|}{\sqrt{2^2+3^2}} = \frac{\sqrt{3^2+4^2}}{\sqrt{13}} = \frac{\sqrt{25}}{\sqrt{13}} = 1.3867$$

Q How do I work out the phase of a complex number?

Similarly, the phase is a term we often apply to a ratio of complex numbers as might arise from a transfer function that has been worked out for a particular frequency. For example, we might have a transfer function that has been worked out at a frequency of $\omega = 2 \text{ rad s}^{-1}$ to give:

$$G(j2) = \frac{3+j4}{2+j3}$$

The phase of the complex number $G(j2)$ is simply its polar angle. Therefore we need to be able to find the polar angle from a complex number.

Finding the phase (polar angle) from a complex number

Given $z = a + jb$

we can write equivalently $z = r \cos \theta + jr \sin \theta$
 $= r(\cos \theta + j \sin \theta)$

Therefore $a = r \cos \theta$ and $b = r \sin \theta$

and
$$\tan \theta = \frac{\sin \theta}{\cos \theta} = \frac{r \sin \theta}{r \cos \theta} = \frac{b}{a}$$

so that
$$\theta = \tan^{-1} \left(\frac{b}{a} \right)$$

In general, for a complex number we can write

$$\theta = \tan^{-1} \frac{\text{Imag}}{\text{Real}}$$

One strategy for finding the phase of $G(j\omega)$ is to write it as $G(j\omega) = a + jb$. Thus

$$\begin{aligned} G(j2) &= \frac{3+j4}{2+j3} = \left(\frac{3+j4}{2+j3} \right) \left(\frac{2-j3}{2-j3} \right) \\ &= \frac{6-j9+j8+12}{2^2+3^2} = \frac{18-j1}{13} = \frac{18}{13} + j\frac{-1}{13} \end{aligned}$$

giving

$$\theta = \tan^{-1} (-1/18), \text{ so that } \theta = -3.179^\circ$$

Equivalently we can evaluate the expression by noting that the angle or phase, denoted \angle , of a fraction is merely the phase of the numerator minus the phase of the denominator:

$$\angle G(j2) = \angle \frac{3+j4}{2+j3} = \angle(3+j4) - \angle(2+j3) = \tan^{-1}(4/3) - \tan^{-1}(3/2) = -3.179^\circ$$

One last topic which may involve complex numbers is that of solving a quadratic equation. This procedure will appear time and time again in control engineering, since many systems can be represented as second-order processes.

Q

I need to practise solving quadratic equations: are there some simple methods?

There are two common methods for finding the roots of a quadratic equation. Here we give both. Firstly, assume that we have a second-order polynomial equation of the form:

$$as^2 + bs + c = 0$$

We wish to find the values of s that solve this equation.

Method 1: Formula

The roots of the quadratic equation are given by

$$s = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

Example Find the roots of $3s^2 + 2s + 4 = 0$.

$$s = \frac{-2 \pm \sqrt{2^2 - 4 \times 3 \times 4}}{2 \times 3} = \frac{-2 \pm \sqrt{-44}}{6} = \frac{1}{3} \pm j \frac{\sqrt{11}}{3} = 0.33 \pm j1.105$$

Method 2: Separating real and imaginary parts to give a 'sum of squares' form

This method is more clearly understood by an example.

Example Find the roots of $3s^2 + 2s + 4 = 0$.

Divide through by the coefficient of s^2 :

$$s^2 + \frac{2}{3}s + \frac{4}{3} = s^2 + 0.66s + 1.33 = 0$$

We note that we can rewrite the terms in s^2 and s as follows:

$$s^2 + 0.66s = (s + 0.33)^2 - 0.33^2$$

Substituting this into the original equation gives:

$$\{(s^2 + 0.66s) + 1.33\} = \{(s + 0.33)^2 - 0.33^2\} + 1.33 = (s + 0.33)^2 + 1.21 = 0$$

We find that the square root of 1.21 is 1.105, and this gives us the alternative 'sum of squares' form:

$$s^2 + 0.66s + 1.33 = (s + 0.33)^2 + 1.105^2 = 0$$

The final steps are simple:

$$(s + 0.33)^2 + 1.105^2 = 0$$

$$(s + 0.33)^2 = -1.105^2$$

$$s + 0.33 = \pm j1.105$$

$$s = -0.33 \pm j1.105$$

We can see that the real and imaginary parts of the roots can easily be read from the 'sum of squares' form when the original equation is rearranged into this convenient form.

Q

Parameter dependent complex numbers! That sounds hard – what are they?

In control engineering we often meet complex numbers which are dependent on an independent parameter. In our case this parameter is usually angular frequency, ω . Let us begin using the Cartesian form for complex numbers z :

$$z = a + jb$$

We introduce a dependency on the parameter ω as:

$$z(\omega) = a(\omega) + jb(\omega) \quad -\infty < \omega < \infty$$

For example, let

$$a(\omega) = 1 + \cos \omega \quad \text{and} \quad b(\omega) = 1 + \sin \omega$$

then

$$z(\omega) = (1 + \cos \omega) + j(1 + \sin \omega) \quad -\infty < \omega < \infty$$

The graphical implication of this dependency is that the point $z(\omega) = (a(\omega), b(\omega))$ is no longer fixed but changes position as the parameter ω travels from $-\infty$ to ∞ . We say that the point z *traces out a locus*. Indeed, it is not difficult to see that:

$$a(\omega) = 1 + \cos \omega \quad \text{and hence } (a(\omega) - 1) = \cos \omega$$

$$b(\omega) = 1 + \sin \omega \quad \text{and hence } (b(\omega) - 1) = \sin \omega$$

giving

$$(a(\omega) - 1)^2 + (b(\omega) - 1)^2 = 1$$

and the locus or path, given by $z(\omega) = a(\omega) + jb(\omega)$, is a unit circle, centre $(1, 1)$. We show this in Figure 2.4.

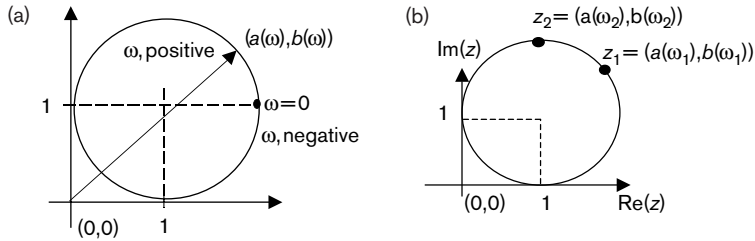


Figure 2.4 Plot for $z(\omega) = a(\omega) + jb(\omega)$, where $a(\omega) = 1 + \cos\omega$, $b(\omega) = 1 + \sin\omega$, $-\infty < \omega < \infty$.

The step to a polar parametric representation is easily taken:

$$z(\omega) = r(\omega)(\cos\theta(\omega) + j \sin\theta(\omega))$$

In the example of the circle locus, the formulas for $r(\omega)$ and $\theta(\omega)$ are not so convenient:

$$r(\omega) = (3 + 2(\cos\omega + \sin\omega))^{1/2}$$

$$\theta(\omega) = \tan^{-1} \left\{ \frac{1 + \sin\omega}{1 + \cos\omega} \right\} \quad \text{where } -\infty < \omega < \infty$$

This characterisation of $z(\omega)$ through the pair $(r(\omega), \theta(\omega))$ motivates a potential representation as two graphs: (1) modulus $r(\omega)$ versus parameter, and (2) angle $\theta(\omega)$ versus parameter ω . This type of representation will be seen to be quite important as the basis of our frequency response plots, or as they are called in control engineering, Bode plots. When we use these types of plot we usually use logarithmic scales.

Step 2 The Laplace transforms of signals

The Laplace transform is a useful tool for the analysis of control systems. We discuss its definition and some of its important properties in this step on the route to success. The emphasis is on gaining some confidence with defining Laplace transforms and understanding something of their structure.

Q

How do I define a Laplace transform?

Let us assume that $x(t)$ is a time signal for $0 \leq t < \infty$ and $x(t) = 0$ for $t < 0$. The operation of applying the Laplace transform is indicated by the symbol $\mathcal{L}\{x(t)\}$ and is defined as:

$$X(s) = \mathcal{L}\{x(t)\} = \int_0^{\infty} x(t)e^{-st} dt$$

where $s = \sigma + j\omega$ is a complex variable.

As can be seen, the Laplace transform takes the time signal, $x(t)$ and integrates out the time variable to produce a function $X(s)$ of the complex variable, s . We normally use the

lower-case letter for a time signal, x , and the upper-case letter for its Laplace transform, X . We also use the notation $\text{Re}(s)$ for the real part of s and $\text{Im}(s)$ for the imaginary part of s ; thus if $s = \sigma + j\omega$, then $\text{Re}(s) = \sigma$ and $\text{Im}(s) = \omega$.

Example Find the Laplace transform of the decaying exponential signal:

$$x(t) = e^{-4t} \quad \text{where } 0 < t < \infty$$

Solution Substitute for $x(t)$ in the definition of the Laplace transform to obtain:

$$X(s) = \mathcal{L}\{e^{-4t}\} = \int_0^{\infty} e^{-4t} e^{-st} dt = \int_0^{\infty} e^{-(s+4)t} dt$$

Evaluating the transform, we have

$$X(s) = \left[\frac{e^{-4(s+4)t}}{-(s+4)} \right]_{t=0}^{t=\infty} = \left[\frac{0}{-(s+4)} - \frac{1}{-(s+4)} \right]$$

and

$$X(s) = \frac{1}{(s+4)}$$

Fortunately, we do not often work out Laplace transforms from first principles, but use a table of common Laplace transforms, as shown in Table 2.2 (pp. 29–30).

Q

What is the Laplace variable s and the s -plane?

The Laplace variable s is a complex variable defined in the formula of the Laplace transform. It is given by:

$$s = \sigma + j\omega$$

where σ is the real part of s , which we also denote as $\text{Re}(s)$. We will see later that σ is related to the boundedness of a signal or the stability of a system. Variable ω is the imaginary part of s , which we also denote as $\text{Im}(s)$. This variable will be shown to be related to the frequency content of a signal.

We can show the range of variation of s or the domain of s schematically as a plane. We call the horizontal axis of this plane the real axis, since it is defined by:

$$s = \sigma + j0$$

The vertical axis is known as the imaginary axis and is represented by:

$$s = 0 + j\omega$$

The origin is represented by the complex zero $s = 0 + j0$. The region to the left of the imaginary $j\omega$ axis (for $-\infty < \sigma < 0$) is known as the Left Half Plane (LHP) and the region to the right of the imaginary axis (for $0 < \sigma < \infty$) is known as the Right Half Plane (RHP). We call the whole complex plane the s -plane (Figure 2.5).

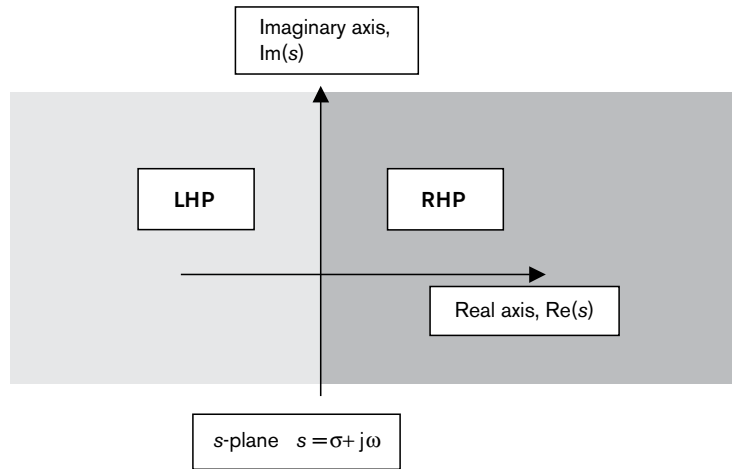


Figure 2.5 The s-plane.

Q

Where do the poles and zeros come from in a Laplace transform?

Very often Laplace transforms work out as the ratio of two constant coefficient polynomials, for example:

$$G(s) = \frac{s+6}{s^2+4s+3}$$

This can be written in general form as

$$X(s) = \frac{b_m s^m + b_{m-1} s^{m-1} + \dots + b_1 s + b_0}{a_n s^n + a_{n-1} s^{n-1} + \dots + a_1 s + a_0} \quad n \geq m$$

We write this in the format:

$$X(s) = \frac{\text{num}(s)}{\text{den}(s)}$$

and identify the following:

Numerator polynomial, num(s)

$$\text{num}(s) = b_m s^m + b_{m-1} s^{m-1} + \dots + b_1 s + b_0$$

with the degree of num(s) given by the integer m .

Transform zeros

The Laplace transform zeros are the values of s for which the transform is zero. They are found by setting the numerator polynomial to 0 and solving the resulting equation.

$$\text{num}(s) = b_m s^m + b_{m-1} s^{m-1} + \dots + b_1 s + b_0 = 0$$

Denominator polynomial, den(s)

$$\text{den}(s) = a_n s^n + a_{n-1} s^{n-1} + \dots + a_1 s + a_0$$

with the degree of $\text{den}(s)$ given by the integer n . Parameter n is also called the order of the transform, $X(s)$.

Transform poles

The Laplace transform poles are the values of s for which the transform is infinite. The poles are found by setting the denominator polynomial to 0 and solving the resulting equation.

$$\text{den}(s) = a_n s^n + a_{n-1} s^{n-1} + \dots + a_1 s + a_0 = 0$$

Example

$$G(s) = \frac{s+6}{s^2+4s+3}$$

We write this in the form:

$$G(s) = \frac{\text{num}(s)}{\text{den}(s)} = \frac{s+6}{s^2+4s+3}$$

and identify the following:

Numerator polynomial: $\text{num}(s) = s + 6$, with the degree of $\text{num}(s)$ given by $m = 1$.

Transform zeros: Solve $\text{num}(s) = s + 6 = 0$ and $s = -6$.

Denominator polynomial: $\text{den}(s) = s^2 + 4s + 3$, with the degree of $\text{den}(s)$ given by $n = 2$. This is also the *order* of the transform, $G(s)$.

Transform poles: Solve $\text{den}(s) = s^2 + 4s + 3 = 0$ and the poles are $s_1 = -1$ and $s_2 = -3$.

Q

How do I use transform tables?

We have seen that the Laplace transform definition was given as:

$$X(s) = \int_0^{\infty} x(t) e^{-st} dt \quad s = \sigma + j\omega$$

This shows how the time variable, t , is integrated out to leave a transform function $X(s)$ which is a function of complex variable s . If we wish to *reverse* this process and go from a given signal transform $X(s)$ to a time domain signal $x(t)$ then there is a complex inversion integral formula available, as shown below.

The inverse Laplace transform

$$x(t) = \begin{cases} \frac{1}{2\pi j} \int_{\sigma-j\infty}^{\sigma+j\infty} e^{st} X(s) ds & 0 \leq t < \infty \\ 0 & -\infty < t < 0 \end{cases}$$

As can be seen, this is a fairly formidable looking operation. Fortunately we have no reason to use this formula in this book; instead we rely on tables and computers to assist

us. Table 2.2 shows common Laplace transform pairs. The tables available of the Laplace transforms cover a wide range of signals and these can be consulted in two different ways.

From a time function to its Laplace transform:

Look down the time function column until the required time function is found and then read off the equivalent Laplace transform.

From a Laplace transform to its time function:

Look down the Laplace transform column until the required Laplace transform is found and then read off the time function.

Table 2.2 Common Laplace transform pairs.

	$X(s)$	$x(t), t \geq 0$
1.	1	$\delta(t)$: unit impulse at $t = 0$
2.	$\frac{1}{s}$	1: unit step, $t > 0$
3.	$\frac{1}{s^2}$	t : ramp function
4.	$\frac{1}{s^n}$	$\frac{1}{(n-1)!} t^{n-1}$
5.	$\frac{1}{(s+a)}$	e^{-at}
6.	$\frac{1}{(s+a)^n}$	$\frac{1}{(n-1)!} t^{n-1} e^{-at}$
7.	$\frac{1}{s(s+a)}$	$\frac{1}{a}(1 - e^{-at})$
8.	$\frac{1}{s(s+a)(s+b)}$	$\frac{1}{ab} \left(1 - \frac{b}{b-a} e^{-at} + \frac{a}{b-a} e^{-bt} \right)$
9.	$\frac{(s+\alpha)}{s(s+a)(s+b)}$	$\frac{1}{ab} \left(\alpha - \frac{b(\alpha-a)}{b-a} e^{-at} + \frac{a(\alpha-b)}{b-a} e^{-bt} \right)$
10.	$\frac{1}{(s+a)(s+b)}$	$\frac{1}{b-a} (e^{-at} - e^{-bt})$
11.	$\frac{s}{(s+a)(s+b)}$	$\frac{1}{a-b} (ae^{-at} - be^{-bt})$
12.	$\frac{(s+\alpha)}{(s+a)(s+b)}$	$\frac{1}{b-a} [(\alpha-a)e^{-at} - (\alpha-b)e^{-bt}]$
13.	$\frac{\omega_n}{s^2 + \omega_n^2}$	$\sin \omega_n t$
14.	$\frac{s}{s^2 + \omega_n^2}$	$\cos \omega_n t$
15.	$\frac{1}{s(s^2 + \omega_n^2)}$	$\frac{1}{\omega_n^2} (1 - \cos \omega_n t)$

Table 2.2 Common Laplace transform pairs (*continued*).

16.	$\frac{1}{(s+a)^2 + b^2}$	$\frac{1}{b}(e^{-at} \sin bt)$
17.	$\frac{1}{s[(s+a)^2 + b^2]}$	$\frac{1}{a^2 + b^2} + \frac{1}{b\sqrt{a^2 + b^2}} e^{-at} \sin(bt - \phi)$ $\phi = \tan^{-1}\left(\frac{b}{-a}\right)$
18.	$\frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} \quad \zeta < 1$	$\frac{\omega_n}{\sqrt{1-\zeta^2}} e^{-\zeta\omega_n t} [\sin \omega_n(\sqrt{1-\zeta^2})t]$
19.	$\frac{\omega_n^2}{s(s^2 + 2\zeta\omega_n s + \omega_n^2)} \quad \zeta < 1$	$1 - \frac{1}{\sqrt{1-\zeta^2}} e^{-\zeta\omega_n t} \sin[\omega_n(\sqrt{1-\zeta^2})t + \phi]$ where $\phi = \tan^{-1} \frac{\sqrt{1-\zeta^2}}{\zeta}$
20.	$\frac{s+\alpha}{(s+a)^2 + b^2}$	$\frac{\sqrt{(\alpha-a)^2 + b^2}}{b} [e^{-at} \sin(bt + \phi)]$ where $\phi = \tan^{-1} \frac{b}{a}$
21.	$\frac{s+\alpha}{s[(s+a)^2 + b^2]}$	$\frac{\alpha}{a^2 + b^2} + \frac{1}{b} \sqrt{\frac{[(\alpha-a)^2 + b^2]}{(a^2 + b^2)}} e^{-at} \sin(bt + \phi)$ where $\phi = \tan^{-1} \frac{b}{\alpha-a} - \tan^{-1} \frac{b}{-a}$

Although we will find that there are transform tables and computer programs to help us find the Laplace transform of signals, it is often useful to derive one or two examples ourselves, so that we understand more fully how the information in the tables is arrived at. The following examples are the transforms of the most common signals we use in control engineering: the step, the ramp, sine and cosines, growing and decaying exponential signals and growing and decaying sinusoidal signals.

Q

Oh dear, I need to work out Laplace transforms from first principles. How do I begin?

(a) *The step signal transform*

Time domain definition: $x(t) = \begin{cases} A, & 0 < t < \infty \\ 0, & -\infty < t < 0 \end{cases}$

This signal is a step of height A units.

Laplace transform: $X(s) = \frac{A}{s}$

Derivation: Directly from the transform definition:

$$X(s) = \int_0^{\infty} A e^{-st} dt = \left[\frac{A}{-s} e^{-st} \right]_{t=0}^{t \rightarrow \infty} = 0 - \frac{A}{-s} = \frac{A}{s}$$

and

$$X(s) = \frac{A}{s}$$

(b) *The ramp signal transform*

$$\text{Time domain definition: } x(t) = \begin{cases} At, & 0 < t < \infty \\ 0, & -\infty < t < 0 \end{cases}$$

This signal is a ramp signal with slope A .

$$\text{Laplace transform: } X(s) = \frac{A}{s^2}$$

Derivation: Directly from the definition we have

$$X(s) = \mathcal{L}\{At\} = \int_0^{\infty} Ate^{-st} dt = A \int_0^{\infty} te^{-st} dt$$

Using integration by parts:

$$\int u dv = [uv] - \int v du$$

$$\text{Set } u = t, dv = e^{-st} dt, du = dt, v = \frac{-e^{-st}}{s}$$

Then

$$X(s) = A \left[-\frac{te^{-st}}{s} \right]_{t=0}^{t \rightarrow \infty} - A \int_0^{\infty} \left(-\frac{e^{-st}}{s} \right) dt$$

Evaluating the transform:

$$X(s) = A \left[-\frac{te^{-st}}{s} - \frac{e^{-st}}{s^2} \right]_{t=0}^{t \rightarrow \infty} = A \left(-\frac{0}{s} - \frac{0}{s^2} \right) - A \left(-\frac{0}{s} - \frac{1}{s^2} \right)$$

and

$$X(s) = \frac{A}{s^2}$$

(c) *Decaying exponential signal*

$$\text{Time domain definition: } x(t) = \begin{cases} Ae^{-at}, & 0 < t < \infty \\ 0, & -\infty < t < 0 \end{cases}$$

This is a decaying exponential signal where $A > 0$ and $a > 0$.

Derivation: Substitute for $x(t)$ in the definition of the Laplace transform to obtain:

$$X(s) = \mathcal{L}\{Ae^{-at}\} = A \int_0^{\infty} e^{-at} e^{-st} dt = A \int_0^{\infty} e^{-(s+a)t} dt$$

Evaluating the transform, we have

$$X(s) = A \left[-\frac{1}{(s+a)} e^{-(s+a)t} \right]_{t=0}^{t \rightarrow \infty} = A \left[0 - \frac{1}{-(s+a)} \right] = \frac{A}{(s+a)}$$

and

$$X(s) = \frac{A}{(s+a)}$$

(d) *The trigonometric signals: the sine signal*

$$\text{Time domain definition: } x(t) = \begin{cases} A \sin \omega_n t, & 0 < t < \infty \\ 0, & -\infty < t < 0 \end{cases}$$

This is a sine waveform where amplitude $A > 0$ and $\omega_n > 0$.

$$\text{Laplace transform: } X(s) = \frac{A\omega_n}{(s^2 + \omega_n^2)}$$

Derivation: Directly in the definition

$$X(s) = \int_0^{\infty} A \sin \omega_n t e^{-st} dt$$

use

$$\sin \omega_n t = \frac{1}{2j} (e^{j\omega_n t} - e^{-j\omega_n t})$$

Then,

$$\begin{aligned} X(s) &= \frac{A}{2j} \int_0^{\infty} (e^{j\omega_n t} - e^{-j\omega_n t}) e^{-st} dt \\ &= \frac{A}{2j} \left[\int_0^{\infty} e^{-(s-j\omega_n)t} dt - \int_0^{\infty} e^{-(s+j\omega_n)t} dt \right] \\ &= \frac{A}{2j} \left[\frac{-e^{-(s-j\omega_n)t}}{(s-j\omega_n)} \right]_{t=0}^{t \rightarrow \infty} - \frac{A}{2j} \left[\frac{-e^{-(s+j\omega_n)t}}{(s+j\omega_n)} \right]_{t=0}^{t \rightarrow \infty} \end{aligned}$$

A careful look at the conditions where $t \rightarrow \infty$ shows that

$$X(s) = \frac{A}{2j} \left[0 + \frac{1}{(s-j\omega_n)} \right] - \frac{A}{2j} \left[0 + \frac{1}{(s+j\omega_n)} \right]$$

and

$$X(s) = \frac{A\omega_n}{(s^2 + \omega_n^2)}$$



The transforms get even harder: what are the exponential–trigonometric signals?

In this section we look at the several types of exponential–trigonometric signal which can arise in some control problems. We classify them by whether the exponential envelope is of decaying or growing type.

Growing exponential–trigonometric signal transforms

$$\text{Time domain definition: } \begin{aligned} x_s(t) &= \begin{cases} Ae^{at} \sin \omega_n t, & 0 < t < \infty \\ 0, & -\infty < t < 0 \end{cases} \\ x_c(t) &= \begin{cases} Ae^{at} \cos \omega_n t, & 0 < t < \infty \\ 0, & -\infty < t < 0 \end{cases} \end{aligned}$$

These are growing exponential sine and cosine waveforms where amplitude $A > 0$, $a > 0$ and $\omega_n > 0$.

$$\text{Laplace transforms: } X_s(s) = \frac{A\omega_n}{(s-a)^2 + \omega_n^2}, \quad X_c(s) = \frac{A(s-a)}{(s-a)^2 + \omega_n^2}$$

Decaying exponential–trigonometric signal transforms

$$\text{Time domain definition: } \begin{aligned} x_s(t) &= \begin{cases} Ae^{-at} \sin \omega_n t, & 0 \leq t < \infty \\ 0, & -\infty < t < 0 \end{cases} \\ x_c(t) &= \begin{cases} Ae^{-at} \cos \omega_n t, & 0 \leq t < \infty \\ 0, & -\infty < t < 0 \end{cases} \end{aligned}$$

These are decaying exponential sine and cosine waveforms where amplitude $A > 0$, $a > 0$ and $\omega_n > 0$

$$\text{Laplace transforms: } X_s(s) = \frac{A\omega_n}{(s+a)^2 + \omega_n^2}, \quad X_c(s) = \frac{A(s+a)}{(s+a)^2 + \omega_n^2}$$

Derivation of the decaying cosine–exponential transform:

Use the relationship $\cos \omega_n t = (e^{j\omega_n t} + e^{-j\omega_n t}) / 2$ and the definition of the Laplace transform to obtain:

$$\begin{aligned} X_c(s) &= \int_0^{\infty} Ae^{-at} \cos \omega_n t e^{-st} dt \\ &= A \int_0^{\infty} e^{-at} \left(\frac{e^{j\omega_n t} + e^{-j\omega_n t}}{2} \right) e^{-st} dt \\ &= \frac{A}{2} \int_0^{\infty} \{e^{-(s+a+j\omega_n)t} + e^{-(s+a-j\omega_n)t}\} dt \\ &= \frac{A}{2} \left[\frac{e^{-(s+a+j\omega_n)t}}{-(s+a+j\omega_n)} + \frac{e^{-(s+a-j\omega_n)t}}{-(s+a-j\omega_n)} \right]_{t=0}^{t \rightarrow \infty} \end{aligned}$$

A careful consideration of the upper limit behaviour as $t \rightarrow \infty$ shows that

$$X_c(s) = \frac{A}{2} \left[\frac{1}{(s+a+j\omega_n)} + \frac{1}{(s+a-j\omega_n)} \right] = \frac{A(s+a)}{(s+a)^2 + \omega_n^2}$$



When I multiply a signal by a constant, what happens to the Laplace transform?

When we multiply a signal, $x(t)$, by a number, k , its Laplace transform, $X(s)$, is also multiplied by the number k . We prove this simple property by direct use of the Laplace definition; recall,

$$\mathcal{L}\{x(t)\} = X(s) = \int_0^{\infty} x(t)e^{-st} dt$$

Consider $y(t) = kx(t)$; then,

$$Y(s) = \mathcal{L}\{kx(t)\} = \int_0^{\infty} kx(t)e^{-st} dt = k \int_0^{\infty} x(t)e^{-st} dt = kX(s)$$

so that $Y(s) = kX(s)$

Note that if two signals, $x(t)$ and $y(t)$, are equal so that ($k = 1$) and $x(t) = y(t)$, then their Laplace transforms are also equal: $X(s) = Y(s)$.

Example From Table 2.2, if $x(t) = e^{-4t}$, $0 < t < \infty$, then

$$X(s) = \frac{1}{s+4}$$

If $y(t) = 5e^{-4t} = 5x(t)$ then

$$Y(s) = \mathcal{L}\{y(t)\} = \mathcal{L}\{5x(t)\} = 5\mathcal{L}\{x(t)\} = 5X(s)$$

and

$$Y(s) = 5X(s) = \frac{5}{(s+4)}$$



I need to transform a combinations of signals. How do I do this?

When we add or subtract two signals, the Laplace transform of the resulting signal will be the sum or the difference of the Laplace transforms of the two signals. For example, consider two signals $r(t)$ and $y(t)$. If we form the signal $e(t)$ by subtracting $y(t)$ from $r(t)$, we have

$$e(t) = r(t) - y(t)$$

In Laplace transforms, take the transform of both sides of this equation; then

$$\begin{aligned} E(s) &= \mathcal{L}\{e(t)\} = \mathcal{L}\{r(t) - y(t)\} \\ &= \int_0^{\infty} (r(t) - y(t))e^{-st} dt = \int_0^{\infty} r(t)e^{-st} dt - \int_0^{\infty} y(t)e^{-st} dt \\ &= \mathcal{L}\{r(t)\} - \mathcal{L}\{y(t)\} \\ &= R(s) - Y(s) \end{aligned}$$

We may write this operation in more general form for two signals $f_1(t)$ and $f_2(t)$.

Consider

$$h(t) = k_1 f_1(t) \pm k_2 f_2(t) \text{ for constants } k_1 \text{ and } k_2$$

Then

$$H(s) = \mathcal{L}\{h(t)\} = \mathcal{L}\{k_1 f_1(t) \pm k_2 f_2(t)\} = \mathcal{L}\{k_1 f_1(t)\} \pm \mathcal{L}\{k_2 f_2(t)\}$$

Using the Laplace transform property $\mathcal{L}\{kf(t)\} = k\mathcal{L}\{f(t)\}$ gives

$$\begin{aligned} H(s) &= k_1 \mathcal{L}\{f_1(t)\} \pm k_2 \mathcal{L}\{f_2(t)\} \\ &= k_1 F_1(s) \pm k_2 F_2(s) \end{aligned}$$

Q

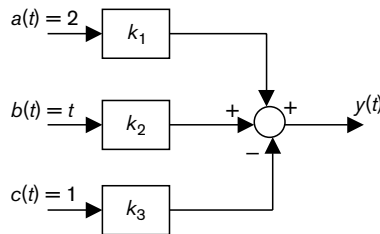
What Laplace transform operations should I learn?

The list of Laplace transform properties is quite extensive but those in Table 2.3 are often useful.

Table 2.3 Formal Laplace transform properties.

Operation on signals	Formal transform operation
Laplace transform of simple signal	$\mathcal{L}\{x(t)\} = X(s)$
The multiplication of a signal by an exponential	$\mathcal{L}\{e^{-at}x(t)\} = X(s+a), \quad a > 0$
Delayed signals	$\mathcal{L}\{x(t-T)\} = e^{-Ts}X(s)$
Scaling a signal	$\mathcal{L}\left\{x\left(\frac{t}{a}\right)\right\} = aX(as)$
Multiplying a signal by a constant	$\mathcal{L}\{kf(t)\} = k\mathcal{L}\{f(t)\} = kF(s)$
Combinations of signals	$\begin{aligned} \mathcal{L}\{k_1 f_1(t) \pm k_2 f_2(t)\} &= k_1 \mathcal{L}\{f_1(t)\} \pm k_2 \mathcal{L}\{f_2(t)\} \\ &= k_1 F_1(s) \pm k_2 F_2(s) \end{aligned}$

Problem Find the Laplace transform of the signal $y(t)$ as shown in the figure below:



Solution First the Laplace transforms of the individual input signals are determined from Table 2.2:

Signal $a(t) = 2$ is a step of height 2 units $\therefore A(s) = \frac{2}{s}$

Signal $b(t) = t$ is a ramp of unit slope $\therefore B(s) = \frac{1}{s^2}$

Signal $c(t) = 1$ is a unit step $\therefore C(s) = \frac{1}{s}$

The block diagram shows the signals being combined together:

$$y(t) = k_1 a(t) + k_2 b(t) - k_3 c(t)$$

$$\begin{aligned}
Y(s) &= \mathcal{L}\{y(t)\} = \mathcal{L}\{k_1 a(t) + k_2 b(t) - k_3 c(t)\} \\
&= k_1 \mathcal{L}\{a(t)\} + k_2 \mathcal{L}\{b(t)\} - k_3 \mathcal{L}\{c(t)\} \\
&= k_1 A(s) + k_2 B(s) - k_3 C(s) \\
&= k_1 \frac{2}{s} + k_2 \frac{1}{s^2} - k_3 \frac{1}{s}
\end{aligned}$$

Hence, putting this over a common denominator leads to:

$$Y(s) = \frac{(2k_1 - k_3)s + k_2}{s^2}$$

Our main aim in this section was to become familiar with the derivation of Laplace transforms for signals. An introduction to the s -plane was also made. This initial look at the formal algebraic and manipulation issues is now to be followed by the more important interpretative aspects of Laplace transforms.

Step 3 Transforms for simple systems

When we begin to use Laplace transforms to analyse systems and their controllers, we will need to manipulate the transforms of signals and the time derivatives of signals. The number of useful properties and formulas that we are required to know is relatively small, and some practice will soon ensure a good familiarity with their use.

Q Why is a differentiator like multiplying by s ?

We can find the Laplace transform of the derivative of a signal $x(t)$ by using the relationship:

$$\mathcal{L}\left\{\frac{d}{dt}x(t)\right\} = sX(s) - x(0)$$

where $x(0)$ is the initial value of $x(t)$, evaluated at $t = 0$. We can prove this relationship using integration by parts as follows:

$$\mathcal{L}\left\{\frac{d}{dt}x(t)\right\} = \int_0^{\infty} \frac{dx}{dt} e^{-st} dt$$

Let $u = e^{-st}$ and $dv = dx/dt$; then $du = -se^{-st}$ and $v = x(t)$. Then use

$$\int u dv = [uv] - \int du v$$

so that

$$\begin{aligned}
\mathcal{L}\left\{\frac{d}{dt}x(t)\right\} &= \left[e^{-st}x(t)\right]_0^{\infty} - \int_0^{\infty} (-se^{-st})x(t) dt \\
&= (0 - x(0)) + s \int_0^{\infty} e^{-st}x(t) dt \\
&= sX(s) - x(0)
\end{aligned}$$

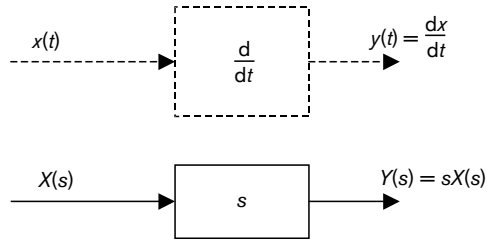


Figure 2.6 Operational equivalence between time domain differentiation and multiplication by the Laplace variable s .

This simple formula has a very useful system analogy. Assume that the signal $x(t)$ has zero initial condition. Then the result becomes $\mathcal{L}\{dx/dt\} = sX(s)$, so that a derivative operating on a signal can be represented as multiplying the signal Laplace transform by s . We see this in block diagram form as Figure 2.6. We have shown by dashed lines the equivalent time domain differentiation.

We find a similar relationship for the second derivative of $x(t)$ with respect to time, t , and it is very useful to see the two results side by side.

Laplace transform of the first derivative of the time signal $x(t)$:

$$\mathcal{L}\left\{\frac{dx}{dt}\right\} = sX(s) - x(0)$$

Laplace transform of the second derivative of the time signal $x(t)$:

$$\mathcal{L}\left\{\frac{d^2x}{dt^2}\right\} = s^2X(s) - sx(0) - \dot{x}(0)$$

where the initial conditions are $x(0)$ and $\dot{x}(0)$. Observe that the *second* derivative is equivalent to operating on the Laplace transform by s^2 . If the initial conditions are defined as zero, then $\mathcal{L}\{d^2x/dt^2\} = s^2X(s)$.

Q

How do we represent an integrator using Laplace transforms?

The Laplace transform of the integral of a signal $x(t)$ results in the relationship

$$\mathcal{L}\left\{\int_0^t x(\tau) d\tau\right\} = \left(\frac{1}{s}\right)X(s)$$

and we can also prove this using integration by parts as follows:

$$\mathcal{L}\left\{\int_0^t x(\tau) d\tau\right\} = \int_0^\infty \left(\int_0^t x(\tau) d\tau\right) e^{-st} dt$$

Let

$$u = \int_0^t x(\tau) d\tau \quad \text{and} \quad dv = e^{-st}$$

Then $du = x(t)$ and $v = (-1/s)e^{-st}$ so that

$$\int u \, dv = [uv] - \int v \, du$$

It is always necessary to be careful when evaluating integral limits. The result is as follows:

$$\begin{aligned} \mathcal{L}\left\{\int_0^t x(\tau) \, d\tau\right\} &= \left[\left(\int_0^t x(\tau) \, d\tau\right)\left(-\frac{1}{s}e^{-st}\right)\right]_0^\infty - \int_0^\infty x(t)\left(-\frac{1}{s}e^{-st}\right) \, dt \\ &= (0-0) + \left(\frac{1}{s}\right)\int_0^\infty x(t)e^{-st} \, dt \\ &= \left(\frac{1}{s}\right)X(s) \end{aligned}$$

As is the case with the differentiator, there is a useful system analogy in this result. The Laplace transform for the integrator is:

$$\mathcal{L}\left\{\int_0^t x(\tau) \, d\tau\right\} = \left(\frac{1}{s}\right)X(s)$$

Thus the effect of an integrator operating on a signal is equivalent to operating on or multiplying the signal Laplace transform by $1/s$. We see this in block diagram form as Figure 2.7, where, once again, we have shown the integral time-domain version using dashed lines.

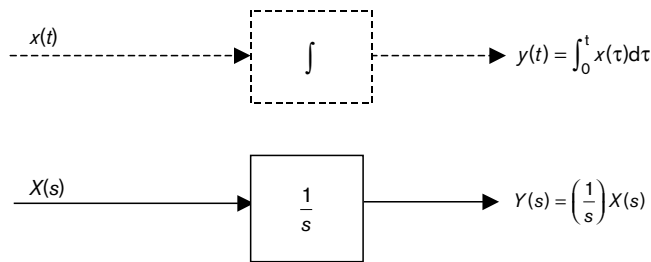


Figure 2.7 Operational equivalence between an integrator and multiplication by $(1/s)$.

Q

I need to use Laplace transforms to represent a differential equation. How do I do this?

Many physical systems involve a rate of change equation, so the operations of differentiation and integration arise naturally when we describe this physical property mathematically. We have already seen how the operations of differentiation and integration can be interpreted as operations on the input signal Laplace transform, and now we start from the rate-of-change equations to extend this to the idea of a physical process as an operation on the system’s input variables. We do this by first looking at a simple example.

Example A system operates on an input signal $u(t)$ to produce an output signal $x(t)$. The system is described by a first-order differential equation:

$$\dot{x}(t) + ax(t) = bu(t)$$

with an initial condition of $x(0) = 0$. Find the Laplace transform relationship for the system.

Solution We take the Laplace transforms of the system differential equation and use standard formulas to find

$$\mathcal{L}\{\dot{x}(t) + ax(t)\} = \mathcal{L}\{bu(t)\}$$

$$\mathcal{L}\{\dot{x}(t)\} + \mathcal{L}\{ax(t)\} = \mathcal{L}\{bu(t)\}$$

$$\mathcal{L}\{\dot{x}(t)\} + a\mathcal{L}\{x(t)\} = b\mathcal{L}\{u(t)\}$$

$$sX(s) - x(0) + aX(s) = bU(s)$$

Solve for $X(s)$:

$$X(s) = \frac{b}{s+a}U(s) + \frac{1}{s+a}x(0)$$

But $x(0) = 0$, giving

$$X(s) = \frac{b}{s+a}U(s) = G(s)U(s) \quad \text{where} \quad G(s) = \frac{b}{s+a}$$

We call $G(s)$ the Laplace transform or transfer function description of the system given by the equation $\dot{x}(t) + ax(t) = bu(t)$.

We often find that the differential equation arises directly from modelling the physical system. The next example shows how a tank-filling exercise can be written in terms of Laplace transforms.

Example Figure 2.8 shows a tank which is being filled with liquid and which has an outflow pipe for the liquid to drain from the tank.

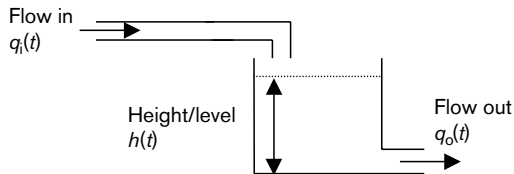


Figure 2.8 Tank being filled with liquid.

The rate of change equation for this system can be given by

$$\{\text{Rate of change in liquid volume in tank}\} = \{\text{Inflow of liquid}\} - \{\text{Outflow of liquid}\}$$

Defining some system variables:

$$v(t) = \text{volume of liquid in the tank, m}^3$$

$$q_i(t) = \text{inflow of liquid, m}^3/\text{s}$$

$$q_o(t) = \text{outflow of liquid, m}^3/\text{s}$$

Then, substituting these variables in the rate of change equation gives

$$\frac{d}{dt}v(t) = q_i(t) - q_o(t)$$

If the tank is a closed vessel, there is no outflow, so $q_o(t) = 0$ and the rate-of-change equation becomes:

$$\frac{d}{dt}v(t) = q_i(t)$$

Suppose now that the formula for the liquid volume is given by $v(t) = Ah(t)$, where A is the *constant cross-sectional area* of the tank and $h(t)$ is the *height* of the liquid in the tank, the rate of change equation becomes:

$$\begin{aligned}\frac{d}{dt}(Ah(t)) &= A \frac{d}{dt}(h(t)) = q_i(t) \\ \frac{d}{dt}h(t) &= \left(\frac{1}{A}\right)q_i(t)\end{aligned}$$

We use our Laplace transform formula for the derivative to find

$$\mathcal{L}\left\{\frac{dh(t)}{dt}\right\} = sH(s) - h(0) = \mathcal{L}\left\{\frac{1}{A}q_i(t)\right\} = \frac{1}{A}\mathcal{L}\{q_i(t)\} = K_T Q_i(s)$$

where $H(s)$, $Q_i(s)$ are the Laplace transforms of the liquid height and the inflow variables, and $K_T = 1/A$.

If we assume that the tank was empty at time $t = 0$ then $h(0) = 0$ and the Laplace equation can be simplified to:

$$H(s) = \left(\frac{K_T}{s}\right)Q_i(s)$$

and we write this as a physical system *transfer function* relation, namely:

$$H(s) = G_T(s)Q_i(s)$$

where the tank's transfer function, $G_T(s)$, is given by

$$G_T(s) = \left(\frac{K_T}{s}\right)$$

Remarks

1. The transfer function $G_T(s)$ is a relation from the system input to a system output. In the example above the input was the inflow variable, $q_i(t)$ and the chosen system output was the liquid height, $h(t)$.
2. The system transfer function, $G_T(s)$, is dependent on the system physical parameters, in this case $K_T = 1/A$. This parameter K_T tells us something about the tank filling process. If the cross-sectional area A is large the tank will be slow to fill, while if the value of A is small, then the tank will fill at a much faster pace. Think about the difference in filling a bath and a swimming pool. The tap is the same but the areas are very different, and so is the time it takes to fill them. So we see that all this abstract analysis *is* firmly linked to the real world.
3. We categorise the different types of physical system behaviour using the type of transfer function that we obtain for our physical process. In the example of the tank filling process, the system transfer function $G_T(s) = K_T/s$ tells us that we have an *integrating* process.

The example was typical of the use of *transfer function analysis*, and because the analysis gave more insight into the physical process the method is a very powerful tool in our analysis toolkit.

Step 4 Formal properties for systems

The links between systems, Laplace transforms and their transfer function representation run very deep in the world of control engineering. To appreciate the full technical detail requires plenty of mathematical background and as engineers we do not want to become too engrossed in all the technical niceties. In this section we look briefly at some technical concepts we are likely to meet and use. Beyond these concepts we have to recognise that if knowledge of the full technical theory of Laplace transforms is going to be necessary, then it is time to call in the experts – the mathematicians!

Q My lecturer says the system is linear. What does she mean?

For system linearity we are considering whether our physical system is going to behave in a fairly straightforward manner or whether we can expect some unusual effects in the system behaviour. The fairly straightforward manner we are looking for is *linear system* behaviour, whereas the counter-intuitive behaviour is *nonlinear system* behaviour. Systems that behave as a linear system can be analysed much more easily than those that do not fit this useful category. So how does linear and nonlinear system behaviour arise? For this we are going to use the tank filling exercise described in the Step 3 section on physical system transfer functions.

In filling a tank with liquid the rate of change equation was given by:

$$\frac{d}{dt} v(t) = q_i(t) - q_o(t)$$

where $v(t)$ = volume of liquid in the tank, $q_i(t)$ = inflow and $q_o(t)$ = outflow. We assumed the tank was a closed vessel, so that there is no outflow, and so $q_o(t) = 0$. In this case, the rate-of-change equation becomes:

$$\frac{d}{dt} v(t) = q_i(t)$$

The next step in the analysis was to supply a formula for the liquid volume; we used $v(t) = Ah(t)$, where A is the *cross-sectional area* of the tank and $h(t)$ is the *height* of the liquid in the tank. It was at this point that a very crucial assumption was introduced, for if the area was dependent on height in any way then instead of having a well-behaved linear system, the analysis becomes complicated and nonlinear. Let us put the two developments side by side to see the difference.

Linear tank model	Nonlinear tank model
Let the formula for the tank volume be $v(t) = Ah(t)$, where A is the cross-sectional area of the tank	
A is regarded as a constant.	A is dependent on say h , so that we can write $A = A(h)$. This might arise because the tank has sloping sides rather like a funnel.
The rate of change equation becomes: $\frac{d}{dt}(Ah(t)) = q_i(t)$ $A \frac{dh}{dt} = q_i(t)$ $\frac{dh}{dt} = \frac{1}{A} q_i(t)$	The rate of change equation becomes: $\frac{dA(h)h(t)}{dt} = q_i(t)$ $\frac{\partial A}{\partial h} \frac{dh}{dt} h(t) + A(h) \frac{dh}{dt} q_i(t)$ $\frac{dh}{dt} = \frac{1}{A(h) + (\partial A / \partial h)h(t)} q_i(t)$
The Laplace transform for the derivative gives $sH(s) - h(0) = \frac{1}{A} Q_i(s) = K_T Q_i(s)$	Clearly this will not allow the use of the Laplace transformations very easily and a simple physical transfer function is not going to emerge. This is a nonlinear system.
If we assume that the tank was empty at time $t = 0$ then $h(0) = 0$ and the Laplace equation can be simplified to: $H(s) = \frac{K_T}{s} Q_i(s) = G_T(s) Q_i(s)$ $G_T(s)$ is the transfer function of the tank. This is a linear system.	Further analysis can be applied to reduce it to a linear system case – or we can use more difficult nonlinear analysis methods.

Unfortunately, our world is nonlinear and even the most simple systems are really quite difficult to describe fully. However, we have to make the most of approximating the actual physical models of real systems, and these approximations usually lead to linear system models. This is very convenient, since we are very good at analysing linear systems, and this is why we concentrate on having a good background in control engineering theory for linear systems. The linear system assumption or hypothesis is fundamental to many of the properties used in this book (for example the next question is on the superposition property and in turn this needs a linear system assumption), so read on.

Q

I want to know what the superposition property is. Is it useful?

The superposition principle asks a very simple question about the output of a system to a sum of inputs. Suppose we have two system inputs, u_1 , u_2 and a combined input defined by: $u_{12} = u_1 + u_2$, and let the system output, y , be obtained as the operation $y = S(u)$. We look at two cases.

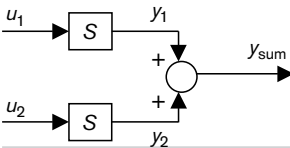
Case 1: Separate inputs

In this case the inputs pass through the system separately and are formed as y_{sum} . Mathematically, define the outputs to the two inputs u_1, u_2 as

$$y_1 = S(u_1), y_2 = S(u_2)$$

and define a summed output as

$$y_{\text{sum}} = y_1 + y_2$$



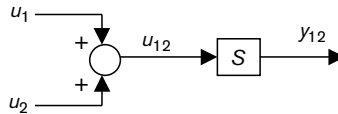
Case 2: Summed inputs

In this case the inputs are first summed and passed through the system as a single input. Mathematically, define the summed input as

$$u_{12} = u_1 + u_2$$

The output to the summed inputs is defined as y_{12} , where

$$y_{12} = S(u_{12})$$



Do the system outputs satisfy $y_{\text{sum}} = y_{12}$? If yes, the principle of superposition holds.

One of the important implications of system linearity is that the principle of superposition holds. This follows from the linearity property that the general output equation satisfies:

$$y = S(u) = S \times u$$

From this we can easily get:

Case 1: Separate inputs

$$y_{\text{sum}} = y_1 + y_2 = S(u_1) + S(u_2) = S \times u_1 + S \times u_2 = S \times (u_1 + u_2) = S \times u_{12} = y_{12}$$

Case 2: Summed inputs

$$y_{12} = S(u_{12}) = S \times u_{12} = S \times (u_1 + u_2) = S \times u_1 + S \times u_2 = y_1 + y_2 = y_{\text{sum}}$$

so that $y_{12} = y_{\text{sum}}$ and the principle of superposition holds.

Is this useful? The answer is yes, because if there are several inputs to a system arising from different physical sources, the principle of superposition allows us to decompose the system output into corresponding output components due to each of the different physical inputs. We can then analyse the effect of the system on these different inputs separately. System linearity and the principle of superposition are extremely powerful properties.

Q

The tutorial sheet mentions causality and physical realisability. They sound difficult. What are they?

Causal and *physically realisable* systems are two terms for a property of systems in the real world. The concept turns on whether the output of a real system depends on future input values. To put this formally in a real practical system, does the output $y(t)$ at time t depend on future values of the system input, $u(t_{\text{future}})$, where $t_{\text{future}} > t$ as well as the past system input, $u(t_{\text{past}})$, where $t_{\text{past}} \leq t$? For example, does an RC circuit have an output

voltage which depends on input voltages which have not yet been applied to the circuit? The answer is no, because the system is causal or physically realisable, in which case the output $y(t)$ depends only on the past system input, $u(t_{\text{past}})$, where $t_{\text{past}} \leq t$. There are several consequences of causality, one of the most notable being that we can find the time response of a casual system if we know the past inputs to the system.

Step 5 Useful operations with Laplace transforms

Although it must be said that the day of easily portable symbolic manipulation software is not so far away now, and many manual mathematical skills currently taught for control courses will disappear, it can be argued that the need for an understanding of these techniques will not disappear. This means that some of the skills reviewed in this section still form a significant part of the tricks of the trade and are often needed in examinations, so there is a good incentive to practise and polish your performance.

Q Can I use partial fractions to help me with Laplace transform derivations?

A convenient method for obtaining the inverse Laplace transform is to use the partial fraction expansion method. This method breaks down the signal transform $X(s)$ into simple transforms for which the inverse Laplace transform is readily available from tables (see Step 2). We present this method by using the following example where we wish to find the inverse Laplace transform of a given signal:

$$X(s) = \frac{b}{s(s+a)}$$

The solution strategy we adopt is to write $X(s)$ as the summation of two simple transforms for which the time function can be found easily; we write:

$$X(s) = \frac{b}{s(s+a)} = \frac{A}{s} + \frac{B}{s+a}$$

To find the values of A and B , multiply both sides of the above equation by $s(s+a)$ to obtain:

$$b = 0s^1 + bs^0 = A[s+a] + Bs = (A+B)s^1 + Aas^0$$

Now equate the coefficients of this polynomial equation to find A and B , namely:

Coefficient of s^1 :

$$A + B = 0$$

Coefficient of s^0 :

$$Aa = b$$

Solving these two equations to find A and B gives:

$$A = \frac{b}{a} \quad \text{and} \quad B = -\frac{b}{a}$$

We can now write the partial fraction expansion of $X(s)$ as

$$X(s) = \frac{b}{s(s+a)} = \frac{b/a}{s} - \frac{b/a}{(s+a)}$$

A look at the Laplace transform pairs in Table 2.2 shows that

$$\mathcal{L}\{1\} = \frac{1}{s} \quad \text{and} \quad \mathcal{L}\{e^{-at}\} = \frac{1}{(s+a)}$$

and we can identify these transforms in $X(s)$:

$$X(s) = [b/a] \frac{1}{s} + [-b/a] \frac{1}{(s+a)}$$

The corresponding time functions are thus:

$$x(t) = [b/a]1 - [b/a]e^{-at}, \quad 0 < t < \infty$$

Hence

$$x(t) = \begin{cases} 0, & t < 0 \\ \frac{b}{a}(1 - e^{-at}), & 0 \leq t \end{cases}$$

This step of going from $X(s)$ to $x(t)$ is called performing the operation of inverse Laplace transformation, sometimes denoted $ILT\{X(s)\}$ and sometimes as $\mathcal{L}^{-1}\{X(s)\}$.

Q

What is the Final Value Theorem?

We are often interested to know the final value or steady state value of a signal, $x(t)$, as time approaches infinity. This we might write as

$$x_{ss} = \lim_{t \rightarrow \infty} x(t)$$

The final value x_{ss} is firstly assumed to exist and be finite. This requires at least a bounded signal, a condition which is tested by checking that all the poles of $X(s) = \mathcal{L}\{x(t)\}$ lie in the left half plane. The $j\omega$ axis is, however, excluded except for a single pole at origin. This is because purely sinusoidal signals, whose transforms will have poles on the $j\omega$ axis, do not settle to a constant value x_{ss} as $t \rightarrow \infty$. Multiple poles at the origin are also excluded because as we have already seen in the table of Laplace transforms, these correspond to unbounded signals like ramps. With these conditions noted:

1. $X(s)$ should represent a bounded signal, having poles only in the left half plane,
2. $X(s)$ should have no purely complex poles,
3. $X(s)$ should not have multiple poles at the origin,

then the Final Value Theorem is:

$$x_{ss} = \lim_{t \rightarrow \infty} x(t) = \lim_{s \rightarrow 0} sX(s)$$

This relationship states that the infinite-time limit of $x(t)$ can be obtained from the limit of the Laplace transform of its derivative, $sX(s)$, as $s \rightarrow 0$.

Problem Find the infinite-time value of the signal $y(t) = ke^{-at}$, $a > 0$.

Solution From the Laplace transform pairs in Table 2.2, we find that the Laplace transform of the signal $y(t)$ is given by

$$Y(s) = \mathcal{L}\{y(t)\} = \frac{k}{s+a}$$

The signal transform has a single pole on LHP, at $s = -a < 0$; it has no purely complex poles or multiple poles at the origin. Hence we may use the Final Value Theorem:

$$y_{ss} = \lim_{t \rightarrow \infty} y(t) = \lim_{s \rightarrow 0} sY(s) = \lim_{s \rightarrow 0} \frac{sk}{s+a} = 0$$

Note that, since $a > 0$, this signal is a decaying exponential and $y(t) \rightarrow 0$ as $t \rightarrow \infty$.

Problem A signal, $x(t)$, is described by the transform

$$X(s) = \frac{b}{s+a}U(s)$$

where $a > 0$ and $U(s)$ is the transform of input signal, $u(t)$. Find the infinite-time value of $x(t)$ when the signal $u(t)$ is a unit step.

Solution For a unit step

$$U(s) = \frac{1}{s}$$

Replace this in the relationship for $X(s)$; hence:

$$X(s) = \frac{b}{(s+a)s}$$

The signal transform $X(s)$ has a single pole at the origin and a pole at $s = -a < 0$ hence we may apply the Final Value Theorem:

$$x_{ss} = \lim_{t \rightarrow \infty} x(t) = \lim_{s \rightarrow 0} sX(s) = \lim_{s \rightarrow 0} \frac{sb}{(s+a)s} = \frac{b}{a}$$

Q How do I use the Initial Value Theorem?

By analogy with the Final Value Theorem, we can relate the initial value of a signal to the Laplace transform of its derivative as $s \rightarrow \infty$, using the Initial Value Theorem:

$$x_0 = \lim_{t \rightarrow 0} x(t) = \lim_{s \rightarrow \infty} sX(s)$$

Problem A signal $x(t)$, is described by the transform:

$$X(s) = \frac{b}{s+a}U(s)$$

where $a > 0$. Find the initial value $x(t)$ when the signal $U(s)$ is a unit step.

Solution For a unit step:

$$U(s) = \frac{1}{s}$$

Substitute this in the relationship for $X(s)$:

$$X(s) = \frac{b}{(s+a)s}$$

The transform has poles at $s = 0$ and $s = -a$, and we may apply the Initial Value Theorem as:

$$\begin{aligned} x_0 &= \lim_{t \rightarrow 0} x(t) = \lim_{s \rightarrow \infty} sX(s) \\ &= \lim_{s \rightarrow \infty} s \frac{b}{(s+a)s} \\ &= \lim_{s \rightarrow \infty} \frac{b}{(s+a)} = 0 \end{aligned}$$

Q How do I find the frequency content of a signal?

If the signal $x(t)$ is bounded and has Laplace transform $X(s)$ then the frequency content of the signal $x(t)$ can be recovered by setting $s = 0 + j\omega$ in $X(s)$. This is equivalent to evaluating $X(j\omega)$. We demonstrate this by writing the Laplace transform integral and setting $s = \sigma + j\omega$:

$$X(s) = X(\sigma + j\omega) = \int_0^{\infty} x(t) e^{-(\sigma + j\omega)t} dt$$

so that if we now set $\sigma = 0$ we obtain:

$$X(j\omega) = \int_0^{\infty} x(t) e^{-j\omega t} dt$$

We define $X(j\omega)$ as the spectral density function of the signal $x(t)$. In fact, $X(j\omega)$ is the Fourier transform of the signal $x(t)$ and reveals the frequency content of the signal. We should note that $X(j\omega)$ is a frequency-dependent complex function and we will be able to graph gain and phase as frequency varies.

Problem Use the Laplace transform to determine the spectral density function or spectrum for the exponential signal $x(t) = e^{-t}$, $0 < t < \infty$.

Solution We use the Laplace transform result of Table 2.2, where we find:

$$\mathcal{L}\{e^{-at}\} = \frac{1}{(s+a)}$$

Thus setting $a = 1$, the Laplace transform $X(s) = \mathcal{L}\{e^{-t}\}$ is

$$X(s) = \frac{1}{(s+1)}$$

This has a pole at $s = -1$ and hence $X(s)$ corresponds to a bounded signal. To recover the frequency content, set $s = j\omega$, then

$$X(j\omega) = \frac{1}{j\omega + 1} = \frac{1}{(j\omega + 1)} \frac{(-j\omega + 1)}{(-j\omega + 1)} = \frac{1 - j\omega}{(1 + \omega^2)} = \left[\frac{1}{(1 + \omega^2)} \right] + j \left[\frac{-\omega}{(1 + \omega^2)} \right]$$

We see that the spectral density function $X(j\omega)$ is a parameter-dependent complex function where the independent parameter is frequency, ω .

To obtain the magnitude plot:

$$|X(j\omega)| = \left\{ \frac{1^2 + (-\omega)^2}{(1 + \omega^2)^2} \right\}^{1/2} = \frac{1}{(1 + \omega^2)^{1/2}}$$

and the phase plot:

$$\text{Arg}(X(j\omega)) = \tan^{-1} \left\{ \frac{-\omega / (1 + \omega^2)}{1 / (1 + \omega^2)} \right\} = \tan^{-1}(-\omega)$$

Thus for positive frequency $0 \leq \omega < \infty$, the frequency content of the signal $x(t) = e^{-t}$ is given in magnitude and phase as:

$$A_x(\omega) = |X(j\omega)| = \frac{1}{(1 + \omega^2)^{1/2}} \quad \text{and} \quad \phi_x(\omega) = \text{Arg}(X(j\omega)) = \tan^{-1}(-\omega)$$

As plots, this information is shown in Figure 2.9.

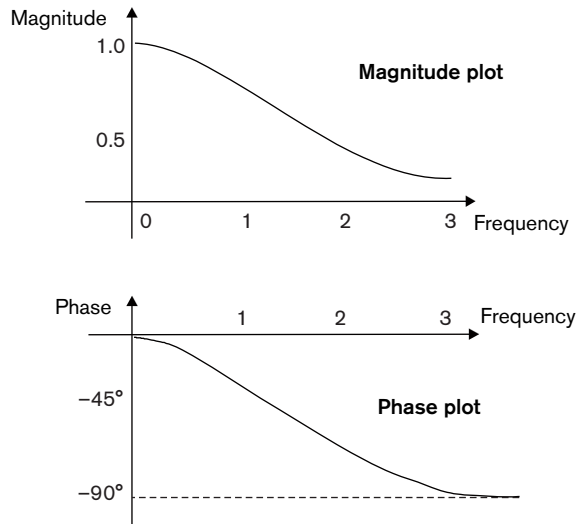


Figure 2.9 The magnitude/phase plots for a negative exponential signal transform.

As can be seen, the frequency content lies mostly in the low-frequency region, and the maximum phase shift $\phi_x(\omega)$ is -90° .

What we have learnt

- ✓ To add, subtract, multiply and divide complex numbers.

- ✓ To represent complex numbers in cartesian, polar and complex exponential forms:

$$\text{cartesian form:} \quad z = a + jb$$

$$\text{polar form:} \quad z = r \cos \theta + j r \sin \theta = r(\cos \theta + j \sin \theta)$$

$$\text{complex exponential form} \quad z = re^{j\theta}$$

- ✓ To realise that the Laplace variable s is a complex variable:

$$s = \sigma + j\omega$$

- ✓ To be able to find a Laplace transform representation of a system given its differential equation description and using the following transforms.

$$\mathcal{L}\{y(t)\} = Y(s)$$

$$\mathcal{L}\left\{\frac{dy}{dt}\right\} = sY(s) - y(0)$$

- ✓ To be able to combine simple systems together by using combinations of their Laplace transforms.

$$\mathcal{L}\{a(t) + b(t)\} = \mathcal{L}\{a(t)\} + \mathcal{L}\{b(t)\} = A(s) + B(s)$$

- ✓ To use the Final Value Theorem as applied to Laplace transforms

$$x_{ss} = \lim_{t \rightarrow \infty} x(t) = \lim_{s \rightarrow 0} sX(s)$$

Multiple choice

M2.1 The Cartesian or rectangular form for the complex number, z , is given by

- (a) $z = re^{j\theta}$
- (b) $z = a + jb$
- (c) $z = r(\cos\theta + j \sin\theta)$
- (d) $z = (a^2 + b^2)^{1/2}$

M2.2 Using tables or otherwise, the Laplace transform of $f(t) = 4e^{-5t}$ is given by:

- (a) $F(s) = 4/(s + 5)$
- (b) $F(s) = 5/(s + 4)$
- (c) $F(s) = 4/(s - 5)$
- (d) $F(s) = 5/(s - 4)$

M2.3 The Laplace transform of the signal $f(t) = 2a(t) - 7b(t)$ is given by:

- (a) $F(s) = 14A(s) + B(s)$
- (b) $F(s) = 2A(s) - 7B(s)$
- (c) $F(s) = 2sA(s) - 7sB(s)$
- (d) $F(s) = 2A(s) - 7B(s)$

M2.4 The roots of the quadratic equation $s^2 + 3s + 5 = 0$ are:

- (a) $s_{1,2} = -3 \pm j\sqrt{11}$
- (b) $s_{1,2} = -1.5 \pm j\sqrt{11}/2$
- (c) $s_{1,2} = -5 \pm j\sqrt{13}/2$
- (d) $s_{1,2} = -2.5 \pm j\sqrt{13}/2$

M2.5 The Laplace transform for the following operation: $y(t) = \dot{x}(t)$ is given by:

- (a) $Y(s) = sX(s)$
- (b) $Y(s) = X(s)/s$
- (c) $Y(s) = aX_1(s) + bX_2(s)$
- (d) $Y(s) = s^2X(s)$

M2.6 Laplace transform methods can be used to:

- (a) do calculations with complex numbers
- (b) represent linear system operations as transfer functions
- (c) manipulate parameter-dependent complex numbers
- (d) solve algebraic equations

M2.7 Linear system descriptions have many useful features and one of these is:

- (a) we can determine the system stability from the system zeros
- (b) we can apply the principle of superposition to summed inputs
- (c) all derivatives can be set to zero in the system description
- (d) we can multiply signals $x(t)$ and $y(t)$ together

M2.8 Partial fraction expansions can be used to:

- (a) find out whether a system is physically realisable
- (b) establish whether the principle of superposition applies
- (c) decompose the transform to allow easy use of Laplace tables
- (d) solve differential equations directly

M2.9 When using the Final Value Theorem, the transform should not have poles on the imaginary axis because:

- (a) these poles represent sinusoids which do not settle out to a steady value as $t \rightarrow \infty$
- (b) these poles lead to unbounded signals like ramps which are not finite as $t \rightarrow \infty$
- (c) these poles lead to decaying signals which have a final value of zero
- (d) the partial fraction method cannot then be applied

M2.10 Setting $s = j\omega$ in a signal Laplace transform enables:

- (a) the steady state value of the signal to be calculated
- (b) the frequency content of the signal to be specified
- (c) the initial value of the signal to be found
- (d) the final value of the signal to be found

Questions: practical skills

Q2.1 Convert the following complex numbers to polar representation (r, θ) :

- (a) $2 + j3$
- (b) $4 - j6$
- (c) $-1 + j0.5$
- (d) $-j7$

Q2.2 Convert the following complex numbers in polar representation to rectangular form:

- (a) $(6, 30^\circ)$
- (b) $(2, -50^\circ)$
- (c) $(2.5, 0.6 \text{ radians})$
- (d) $(1, -0.3 \text{ radians})$

Q2.3 Given the following values of z_1 and z_2 , what are the values of $z_1 \times z_2$?

- (a) $z_1 = (2, -45^\circ)$ and $z_2 = (3, 50^\circ)$
- (b) $z_1 = (4, 20^\circ)$ and $z_2 = (6, 30^\circ)$
- (c) $z_1 = (3, -10^\circ)$ and $z_2 = (20, -50^\circ)$

Q2.4 Given the transfer function $G(j\omega) = (1 + j2\omega)/(3 + j\omega)$ and the following frequency values:

- (i) $\omega_1 = 1 \text{ rad/s}$ and (ii) $\omega_2 = 0.5 \text{ rad/s}$, what are $|G(j\omega)|$ and $\angle G(j\omega)$?

Q2.5 For the following second-order equations:

- (i) what are the roots of the equations?
- (ii) express the equations in the sum of squares form.

- (a) $s^2 + 2s + 3 = 0$
- (b) $s^2 + 4s + 6 = 0$
- (c) $3s^2 - s + 2 = 0$

Q2.6 Using the Laplace transform tables, give the transform $X(s)$ of the following time domain signals:

- (a) $x_1(t) = 6e^{-t}$
- (b) $x_2(t) = 2t$

- (c) $x_3(t) = 4t + 7e^{-2t}$
 (d) $x_4(t) = \sin 5t$
 (e) $x_5(t) = -\cos 2t$

Q2.7 Using Laplace transform tables, find the time domain signals from the following Laplace transforms:

- (a) $X_1(s) = 3/(2 + s)$
 (b) $X_2(s) = 6/(s - 3)$
 (c) $X_3(s) = 7/(s^2 + 16)$
 (d) $X_4(s) = 1/s(s + 3)$
 (e) $X_5(s) = 3/s^2$

Q2.8 Given the following transfer functions, what are the final values of the signals?

- (a) $X_1(s) = 6/(s^2 + 2s + 1)$
 (b) $X_2(s) = 6/(s^3 + s^2 + 3s)$

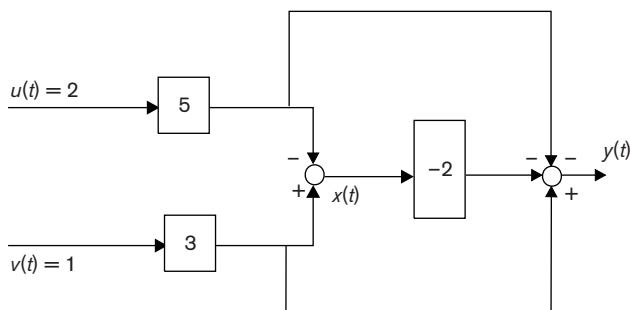
Problems

P2.1 The Laplace transform of the signal $x(t)$ is defined as:

$$X(s) = \frac{1}{(2s+1)} U(s)$$

- (a) Find $x(t)$ for $U(s) = 1$.
 (b) Using partial fraction expansion, repeat (a) for $U(s) = 1/s$.
 (c) Plot the time signals found in (a) and (b) on the same graph.
 (d) Which time signal exhibits an exponential rise and which an exponential fall?

P2.2 Consider the signal diagram shown below.



- (a) Find the Laplace transform of $u(t)$ and $v(t)$.
 (b) Find the Laplace transform of $x(t)$ and $y(t)$.
 (c) Find the poles and zeros of $Y(s)$.

P2.3. Consider the Laplace transform of the signals $x_1(t)$ and $x_2(t)$:

$$X_1(s) = \frac{2s+3}{(s^2+3s+2)}, \quad X_2(s) = \frac{1}{(s^2+2s+2)}$$

- (a) Find the poles of $X_1(s)$ and $X_2(s)$.
 (b) Find the final values of $x_1(t)$ and $x_2(t)$.

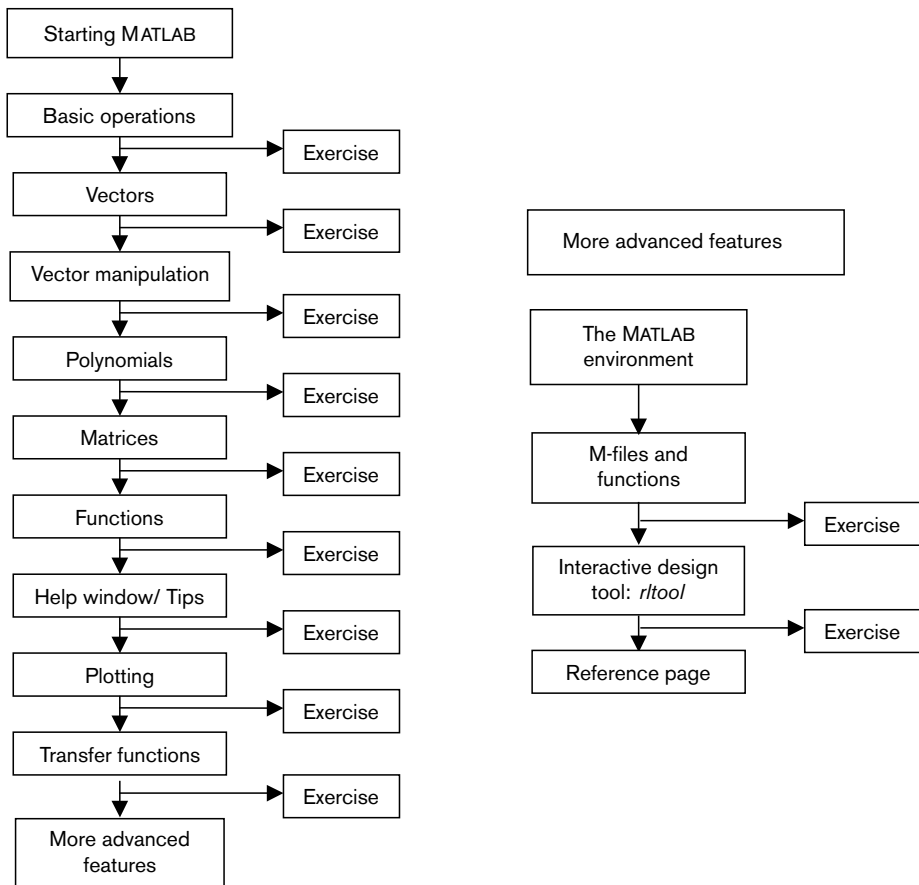
P2.4. Find the spectral density function for $x(t) = e^{-3t}$.

3

Software toolkit: MATLAB

3.1 Introduction to MATLAB **M**

MATLAB is a high-level language for technical computing which is often used by engineers to help them design systems or analyse a system's behaviour. We present the following material in a tutorial style which can therefore be used as a self-teaching exercise, or as a reminder for those who may have used MATLAB and forgotten some of the notation. The following represents a beginner's route into the MATLAB package. Although we use version 6, almost all the basic commands are valid in MATLAB version 5.x.



Throughout the text of this book, MATLAB exercises will be marked by **M**.

3.2 Starting MATLAB

To run MATLAB, move the cursor to the MATLAB icon and double-click on the left-hand mouse button.



The MATLAB desktop will open. This is an integrated development environment for working with MATLAB suite of toolboxes and programs. It initially looks fairly complex. We see in Figure 3.1 that there are three open windows, which represent:

- the *Command Window*
- the *Launch Pad and Workspace*
- the *Command History and Current Directory*

We can make a particular window active by clicking anywhere inside its borders.

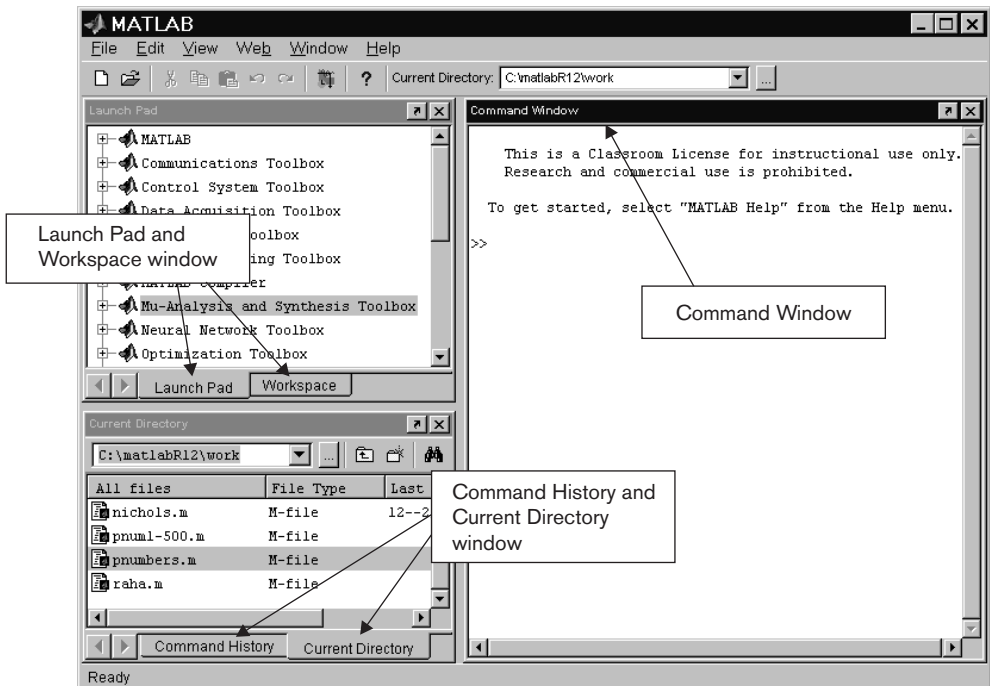


Figure 3.1 MATLAB Desktop (version 6).

For this chapter, we use only the Command Window. To simplify our display, we can view only the Command window. This can be achieved by going to *View, Desktop Layout* and then to *Command Window Only*. The result is shown in Figure 3.2. This is the window that would open if we had used MATLAB 5.x.

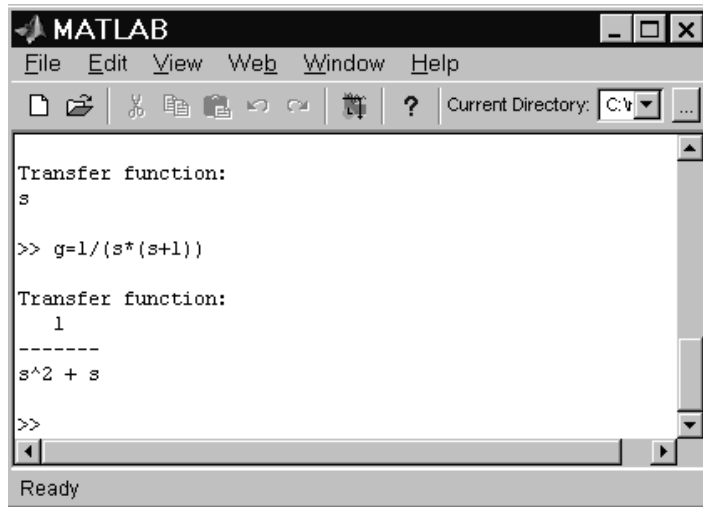


Figure 3.2 Command window only on display.

However, we will use the other windows once we become more proficient and will give a brief summary of their purpose later in the chapter.

Command Window: We type all our commands in this Window at the prompt (`>>`) and press return (`↵`) to see the results of our operations.

3.3 Basic operations

As we go through this tutorial, we introduce new MATLAB commands and functions. We have used the following two symbols in the tutorial:

1. The symbol `↵` is used at the end of a command line to indicate that we need to press the *Return Key* to execute the command.
2. The symbol `⇒` shows what MATLAB returns to the Command Window when we issue our command.

We can assign a numerical value (data) to a variable using the equal (=) sign. For example, type:

```
a = 2
```

and press return (`↵`). MATLAB returns (`⇒`):

```
⇒ a =
    2
```

or `b = -5.2 ↵`

```
⇒ b =
   -5.2
```

M Exercise 1: Basic operations

Complete the table below. The basic arithmetic operations use the operators shown. Use values of $a = 4$ and $b = -2.1$

Operation	Operator	Example	Result
Plus	+	$a+b$	↻
Minus	-	$a-b$	↻
Multiply	*	$a*b$	↻
Power	^	a^b	↻
Divide	/	a/b	↻

3.4 Vectors

Enter each element of the row vector (separated by a space) between square brackets, and set it equal to a variable. For example, to create the row vector x , enter into MATLAB:

```
x=[1 2 3 4] ↻
↻ x =
    1    2    3    4
```

To enter a column vector, separate the elements by a semicolon ';'. For example:

```
y=[1;2;3;4] ↻
↻ y =
     1
     2
     3
     4
```

We can determine the size of the vectors x and y by using the size command.

```
size(x) ↻
↻ ans =
     1    4
```

```
size(y) ↻
↻ ans =
     4    1
```

If we want to create a vector with elements between 0 and 5 evenly spaced in increments of 0.5, we can use:

```
t = 0:0.5:5 ↻
↻ t =
Columns 1 through 7
    0    0.5000    1.0000    1.5000    2.0000    2.5000    3.0000
Columns 8 through 11
    3.5000    4.0000    4.5000    5.0000
```


M Exercise 2: Vectors

- (a) Enter the first three prime numbers as a row vector.
- (b) Repeat (a), but use a column vector.
- (c) The following table shows some measurements of voltage across a resistor as a function of time.

Time (s)	0.0	1.0	2.0	3.0	4.0	5.0	6.0	7.0	8.0	9.0
Voltage (V)	1.1	1.2	1.3	1.4	1.5	1.4	1.3	1.2	1.1	1.0

- (d) Enter the time as a column vector called `tim`.
- (e) Enter the voltage as a column vector called `vol`.
- (f) Enter the command `plot(tim,vol)`.

Which variable is on the independent axis?

Which variable is on the dependent axis?

3.5 Vector manipulation

Manipulating vectors is almost as easy as creating them. Try the following operations:

```
a=[1 2 3 4]
x=a+2      Result:
x=a-2     Result:
x=a*2     Result:
x=a/2     Result:
```

Let $b = [1\ 2\ 3]$. What happens if we try to add two variables (vectors: $a+b$) of different dimensions? What is the error message?

(We note that, in the example $x=a+2$, variable a is size 1×4 and 2 is size 1×1 . Technically they could not be added together, as the dimensions are not compatible. However, MATLAB assumes, when you add/subtract/multiply/divide a variable by a number, that all elements of the vector should be operated on. This is not true with two 'variables', as we see now.)

For $x = [1\ 2\ 3\ 4]$ and $y=[5;6;7;8]$, would the following operations be acceptable?

```
x + y ? 
x*y  ? 
y*x  ? 
```

In this case we note that we cannot add $x + y$ since they do not have appropriate dimensions. However, if we transpose $x \rightarrow x'$ (where $'$ indicates the transpose), we can then evaluate:

```
x+y'
```

The size command could be used to verify the dimensions of the vectors.

M Exercise 3: Vector manipulation

Try the following vector operations and write down the results. Ensure that the dimensions of x and y are appropriate for the operations attempted. Note that some of the operators are different from scalar operators.

Operation	Operator	Example	Notes
Plus	+	$x+y$ ↵	Must be same dimension
Minus	-	$x-y$ ↵	Must be same dimension
Multiply	*	$x*y$ ↵	Must be appropriate dimension
Multiply element by element	.*	$x.*y$ ↵	Must be same dimension
Divide element by element: second vector elements by first vector elements	./	$x./y$ ↵	Must be same dimension
Divide element by element: first vector elements by second vector elements	.\	$x.\y$ ↵	Must be same dimension

Explain in words the three operations: `.*`, `./` and `.\`.
Think of an example where we might need this type of operation.

Remark

Note that if we put a semi-colon (;) at the end of a MATLAB command, MATLAB does not return the values to the window. Try:

```
a=[1 2 3]; ↵
b=[-2 -3 4]; ↵
c=a+b ↵
c=a-b ↵
```

3.6 Polynomials

We can enter the coefficients of a polynomial as a row vector. The coefficients should be entered in descending order of the powers of x . For example

$$p(x) = x^2 + 10x + 3$$

can be represented by

$$p = [1 \ 10 \ 3]; ↵$$

How would we enter the following polynomial, q : $q(t) = 6t^3 + 4t^2 + 3t + 1$?

MATLAB can interpret a vector of length $n + 1$ as the coefficients of an n th order polynomial. Thus, if the polynomial is missing any coefficients, we must enter zeros in the appropriate place in the vector. For example

$$y(s) = 6s^4 + 1$$

would be represented in MATLAB as:

$$y = [6 \ 0 \ 0 \ 0 \ 1]; ↵$$

Finding roots: Use the following command:

```
roots([6 0 0 0 1]) ↵ or roots(y) ↵
```

How would we represent the following polynomial: $m(s) = 4s^4 + s^3 + 2s$?

Determine the roots of $m(s) = 0$.

Multiplying/dividing polynomials: The product of two polynomials is found by taking the convolution of their coefficients. The function `conv` will do this for us.

Polynomial: $X(s) = s + 2$

MATLAB representation: `x = [1 2];` ↵

Polynomial: $Y(s) = s^2 + 4s + 8$

MATLAB representation: `y = [1 4 8];` ↵

Polynomial: $Z(s) = (s + 2)(s^2 + 4s + 8)$

MATLAB representation: `z = conv(x,y)` ↵

$= X(s) * Y(s)$

Dividing two polynomials is just as easy. If $Z(s) = Q(s)Y(s) + R(s)$, then given $Z(s)$ and $Y(s)$ the `deconv` function will return the result, Q , as well as the remainder, R . Note that if there is more than one output for a function, they should be put inside square brackets and be separated by commas. Try to divide $Z(x)$ by $Y(x)$:

```
[Q, R] = deconv(z,y) ↵
```

How do we check the result?

M Exercise 4: Polynomials

We wish to plot a second-order polynomial $y(t) = 4t^2 + 2t - 3$ as a function of time and check the roots of the equation. We can do this in two ways.

1. We can enter the polynomial coefficients as a vector and use the `roots` command.
2. We can plot the function y against time and note the points where the function crosses the time axis (that is, points where $y = 0$). To do this, follow these steps.
 - (a) Create a vector t which has values evenly spaced from -5 to 5 , in steps of 0.5 .
 - (b) Calculate $y(t) = 4t^2 + 2t - 3$ for all values of the time vector, t . Since t is a vector, think carefully how you will calculate $4t^2 + 2t - 3$.
 - (c) Request a plot using `plot(t,y)`. Find the roots on the graph. Are the roots on the graph similar to the answer in Item 1 above?

3.7 Matrices

Entering matrices into MATLAB is the same as entering a vector, except that each row of elements is separated by a semicolon. Try:

```
B = [1 2 3 4;5 6 7 8;9 10 11 12] ↵
```

Matrices in MATLAB can be manipulated in many ways. For example, we can find the transpose of a matrix using the apostrophe key (`'`). Try:

```
C = B' ↵
```

Remember that the order of multiplication is important when dealing with matrices. Would the following operations be allowed?

$$D = B + C \downarrow$$

$$D = B + C' \downarrow$$

$$D = C * B \downarrow$$

$$D = B * C \downarrow$$

$$E = B^3 \downarrow \text{ (Power of a matrix)}$$

$$X = \text{inv}(B) \downarrow \text{ (Inverse of a matrix)}$$

M Exercise 5: Matrices

Find the solution of the following set of linear equations:

$$2x_1 + 5x_2 - 3x_3 = 6$$

$$3x_1 - 2x_2 + 4x_3 = -2$$

$$x_1 + 6x_2 - 4x_3 = 3$$

Hint: We write this in the matrix form, that is, $AX = B$, where

A is the matrix of coefficients of x_1 , x_2 and x_3

X is the column vector which will contain the solutions x_1 , x_2 and x_3

B is the column vector of values on the right-hand side

Answer: $X = A^{-1}B = ?$

3.8 Functions

MATLAB includes many standard functions (and constants). Each function is a block of code that accomplishes a specific task.

Common functions

\sin , \cos , \log (\log_e), \log_{10} (\log_{10}), \exp , sqrt , mean , std (standard deviation)

Common constants

$\text{pi} = \pi$ returns 3.1416. The variables i or j represent the square root of -1 (complex numbers).

Try the following sine function:

$$x = \sin(\text{pi}/4) \downarrow \text{Result} = ?$$

We wish to plot the sine wave of frequency 3 rad/s and amplitude 1 over a period of time between 0 to 20 seconds (in steps of 0.1 seconds):

(a) Write down the sine wave expression: $y(t) = ?$

(b) Calculate the vector y for the values of time given. Graph the results using an appropriate plot command.

Function inputs and outputs

Each function in MATLAB has a number of inputs and output variables. We should define all the necessary inputs before we use a function. We can use the `help` function to check the inputs and outputs of any function we are going to use. For example, if we wanted to find out about the use of the `abs` function, we could type

```
help abs↵
```

MATLAB returns:

```
ABS Absolute value.
  ABS(X) is the absolute value of the elements of X. When
  X is complex, ABS(X) is the complex modulus (magnitude) of
  the elements of X.
```

Therefore by typing

```
y=abs(x);
```

the variable `y` would hold the absolute values of `x`.

If we want to find the modulus and the angle of a complex number, we can enter the following:

```
s = j*pi/4↵ define complex number s
g = 1/s↵ define complex function g
mag=abs(g)↵ find the magnitude:   output: mag   input: g
ang=angle(g)↵ find the angle:     output: ang   input: g
```

Once we have defined the output variables, they will be stored in MATLAB workspace and we can manipulate them as we wish.

M Exercise 6: Functions

A sine wave of amplitude 5 and frequency 0.1 Hz is applied to the input of an analogue device, which has a constant gain of 1.5 over all frequencies. Calculate the output signal for the interval 0 to 10 seconds in increments of 0.1 seconds.

Use the command `plot(t,y)↵` to see a graph of this signal. What is the frequency of the sine wave on the plot? (Use the `grid` command to help work this out (approximately)).

Does it agree with the frequency of 0.1 Hz given in the question?

3.9 Help window/tips

To determine the usage of any function, enter:

```
help function name
```

Try

```
help sin↵
```

To see the help window, type:

```
helpwin↵
```

It lists the directories for help for many of the other toolboxes that come with MATLAB that we may use later in this book.

For example, click on MATLAB\elfun to see some basic functions. With more practice we can even write special MATLAB files (M-files) to save re-typing the same commands.

Tips

1. MATLAB variables are case-sensitive, that is, temp and Temp would be two different variables.
2. We can use the command `who` to see the variables in the MATLAB workspace.
3. We can get the value of a particular variable at any time by typing its name. Use `who` to list all the variables in the workspace. Type the name of one of these variables to show its current value.
4. We can use `clear all` to clear all the variables from the workspace or `clear (variable name)` to clear only a single variable.
5. We can also have more than one statement on a single line, so long as we separate them with either a semicolon or comma. Try:

```
gain = 20*log10(2), ang=angle(0.5*i);
```

Now, enter `gain` and `ang` to see the results.

6. If we do not assign a variable to a specific operation or result, MATLAB will store the outcome of the operation in a temporary variable called `ans`. For example:

```
3+6
ans
= 9
```

M Exercise 7: Help/tips

- (a) Enter the complex number $3 + j4$.
- (b) Find the modulus and angle of this complex number.
- (c) Change the value of magnitude calculated to dB and change the value of angle to degrees.

Write down the MATLAB commands that were entered.

3.10 Plotting

It is easy to create plots in MATLAB. Suppose we make a time vector and then compute the sine values of the vector at each time point.

```
t=0:0.05:10;
y = sin(t);
plot(t,y)
```

Basic plotting is very easy in MATLAB, and the `plot` command has extensive add-on capabilities. Some useful features are given here.

3.10.1 figure command

When we plot a graph, MATLAB opens a window called the Figure Window. Every time we plot a graph, this figure window is updated. If we want to keep the old graphs, we can open a new window by using the command:

```
figure↵
```

Try:

```
plot(t,y) ↵
figure↵
plot(t,2*y)↵
```

3.10.2 Plotting several responses on same axis

Plot the first response graph required and then enter the command:

```
hold on↵
```

Try these commands:

```
figure(2); ↵
plot(t,y) ↵
hold on ↵
plot(t,2*y) ↵
plot(t,0.5*y) ↵
```

Enter hold off to return to the default mode.

3.10.3 Finding the coordinates of a point on a graph

For graphs using the plot command, use

```
ginput(N)
```

N is the number of points at which coordinates may be found.

This command will give a cursor which we can move to a point on the figure using the mouse, and then press the left-hand button to see the coordinates in MATLAB windows. Alternatively, click on any point on the graph and a text window pops up with the coordinates' values.

Example Type the following:

```
t=0:0.05:10; ↵
y = sin(t); ↵
plot(t,y) ↵
ginput(1) ↵
```

Use the mouse to click on the maximum value of the sine response. Look in the MATLAB Command Window to see the values of x and y .

3.10.4 Labelling plots and axes

We can use the *Insert* pull-down menu to place text, labels and lines on a graph (Figure 3.3).

We use the sine wave as previously generated:

```
t=0:0.05:10; y = sin(t); plot(t,y) ↵
```

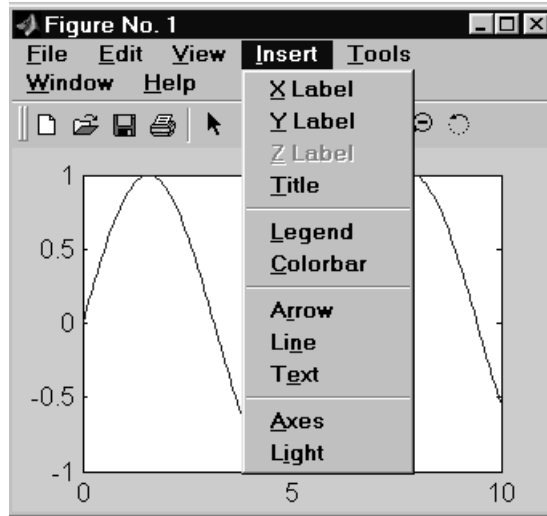


Figure 3.3 Example of Insert command (MATLAB v6).

1. Go to Insert and then choose X Label. A text box will appear under the x-axis. Type 'Time (sec)' and then click on any point on the Figure Window.
2. Go to Insert and then choose Y Label. A text box will appear along the y-axis. Type 'Amplitude' and then click on any point on the Figure Window.
3. Go to Insert and then choose Title. A text box will appear on the top of the graph box. Type 'Sine Wave Plot' and then click on any point on the Figure Window.
4. Go to Insert and then choose Text. A text box will appear on the graph. Type 'Example of using labels' and then click on any point on the Figure Window. Click on the text box and move it to the top right-hand corner.
5. Go to Insert and then choose Line. The mouse cursor changes to a cross. Click on zero and then click on the point (0,10). A line will appear parallel to the x-axis at the $y = 0$ level.

The resulting plot should look like Figure 3.4.

3.10.5 Frequency response graphs

The command `semilogx` is the same as `plot` except that a logarithmic (base 10) scale is used for the x-axis. This command is used in the exercise below.

M Exercise 8: Plotting

The input-output relationship of an RC circuit can be represented by the following complex function:

$$g(j\omega) = \frac{V_o}{V_i} = \frac{K}{j\tau\omega + 1}$$

where K is the gain and τ is the time constant.

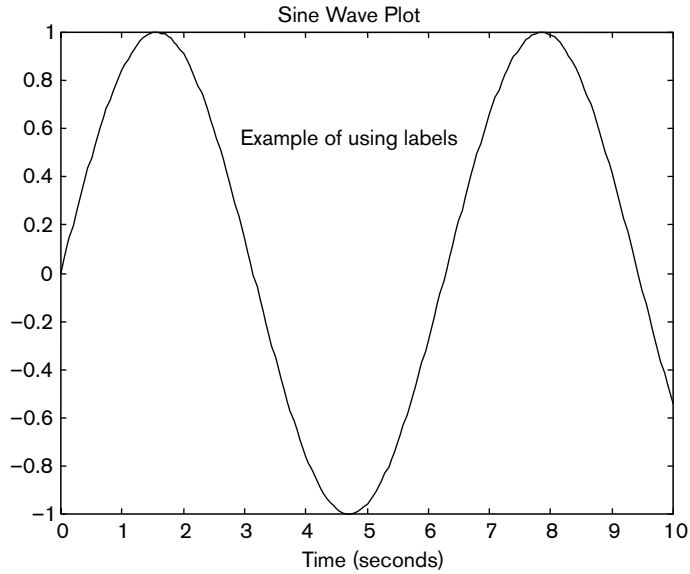


Figure 3.4 Result of labelling and annotating a graph.

- (a) For $K = 10$ and $\tau = 5$, calculate the gain and phase of V_o/V_i for $\omega = 0.01, 0.05, 0.1, 2, 5, 10$ (rad/s).
- (b) Plot the gain and phase using the semi logx command.

Hints:

1. Enter the values K and τ as `K` and `tau`. Define a frequency vector, `w`.
2. Calculate `g=K ./ (j*tau*w+1)`. Note the operation `./`.
3. Calculate the gain in dB and the phase in degrees.
4. Plot the response using `semilogx(w,gain)` and `semilogx(w,phase)`.
5. Label the axes and insert a title for the plot.

3.11 Transfer functions in MATLAB

A transfer function can be entered into MATLAB using different commands.

Method 1

1. We can use the command `tf(num,den)`, where `num` and `den` are vectors of coefficients of the numerator and denominator polynomials, respectively. Enter the transfer function:

$$g_1 = \frac{3s+1}{s^2+3s+2}$$

```
num = [3 1] ↵
den = [1 3 2] ↵
g1 = tf(num,den) ↵ enter the transfer function
```

2. Alternatively, we can define s as a transfer function and then use it to 'write' the transfer function directly.

```
s=tf('s') ↵
g1= (3*s+1)/(s^2+3*s+2) ↵
```

Method 2

1. We can use the command `zpk(zeros,poles,gain)`, where zeros, poles and gain are vectors of the zeros, poles and gain of the transfer function. For example, we can enter the following transfer function in the form:

$$g_2 = \frac{4(s+5)}{(s+3)(s+10)}$$

```
zeros = [-5] ↵
poles = [-3 -10] ↵
gain = 4 ↵
g2 = zpk(zeros,poles,gain) ↵ enter the transfer function
```

or equivalently

```
g2 = zpk([-5], [-3 -10], 4) ↵
```

2. Alternatively, we can define s as a transfer function and then use it to 'write' the transfer function directly:

```
s=zpk(' s') ↵
g2= 4*(s+5)/((s+3)*(s+10)) ↵
```

If we have a factor with complex poles in the transfer function,

$$g_3 = \frac{4(s+5)}{(s+2+3j)(s+2-3j)} = \frac{4(s+5)}{s^2+4s+13}$$

the complex roots of the denominator can be entered easily as:

```
g3=4*(s+5)/((s+2+3*j)*(s+2-3*j)) ↵
```

Try entering the following transfer functions:

$$g_4 = \frac{12(3s+2)}{(s+3+7j)(s+3-7j)} \quad g_5 = \frac{2(s+8)}{(s+3)(s+0.1)(4s+6)}$$

3.11.1 Transfer function manipulation

Once various transfer functions have been entered, we can combine them together.

Use the transfer functions

$$g_6 = \frac{1}{s+2} \quad \text{and} \quad g_7 = \frac{5}{s+3}$$

to complete the following table.

Operation	Result
Addition	$g_6 + g_7$
Subtraction	$g_6 - g_7$
Multiplication	$g_6 * g_7$
Division	g_6/g_7
Combination of operations	$g_6/(1 + g_6)$

Now that we have entered various transfer functions, it is easy to use some of the control analysis commands available from the various MATLAB toolboxes. The following commands represent some of the common control analysis operations that we will meet in this book.

3.11.2 Time responses

If g represents a system transfer function, then:

1. For a unit impulse response use the command: `impulse(g)` ↵
2. For a unit step response, use the command: `step(g)` ↵
3. For a step response of magnitude K , simply multiply the transfer function, g , by the numerical value of K : `step(K*g)` ↵

For some functions, such as `step` or `bode`, MATLAB graphs are automatically generated. For these graphs, we can use the cursor to find the coordinates of different points.

Example Type the following:

```
s=tf('s') ↵
g=1/(s+1) ↵
step(g) ↵
```

Click on a point on the graph. A cursor will appear as shown in Figure 3.5. We can hold the cursor using the mouse and move over the graph to see the coordinates at different points. In MATLAB v6, if we release the mouse the text window containing the coordinates will remain on the graph. We can delete the text window by clicking on it using the right-hand mouse button.

3.11.3 Poles and zeros

1. To see the roots of the numerator (called the zeros) of $g(s)$ use the command: `zero(g)` ↵
2. For the roots of the denominator (called the poles) of a transfer function use the command: `pole(g)` ↵
3. For a pole-zero map use the command: `pzmap(g)` ↵

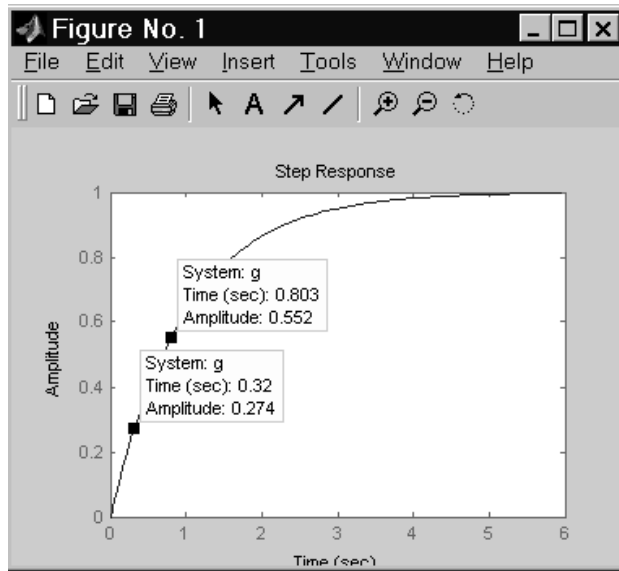


Figure 3.5 Finding coordinates using the cursor.

3.11.4 Feedback control

To determine the closed-loop control transfer function from a system with negative feedback we can use the command:

```
g=feedback(g,h) ↵
```

where g is the forward transfer function and h represents the transfer function of the components in the feedback path. If positive feedback is required then the following command can be used:

```
g=feedback(g, h, 1) ↵
```

A general quick-reference MATLAB help sheet to functions that we will meet throughout this book is included at the end of this chapter.

M Exercise 9: Transfer functions

Given the following transfer functions:

$$g_1(s) = \frac{3(s+1)}{s^2 + 3s + 1}, \quad g_2(s) = \frac{2s+3}{s^3 + 1}$$

- Enter $g_1(s)$ using row vectors.
- Enter $g_2(s)$ using the `s=tf('s')` notation.

Record from the screen the MATLAB commands that were entered.

- Find the roots of the numerator and denominator polynomials using the `roots` command.
- Compare with the results using the `zero` and `pole` commands.
- Examine where the roots are using the `pzmap` function.

3.12 MATLAB environment

We return now to examining the MATLAB window environment. This can be achieved by choosing View, Desktop Layout and then Default. The default Desktop windows are displayed once more.

Launch Pad: We use this window (Figure 3.6) to execute MATLAB demos and programs. It has a tree structure similar to Windows Explorer. Double-clicking on a displayed icon runs the program associated with the icon.

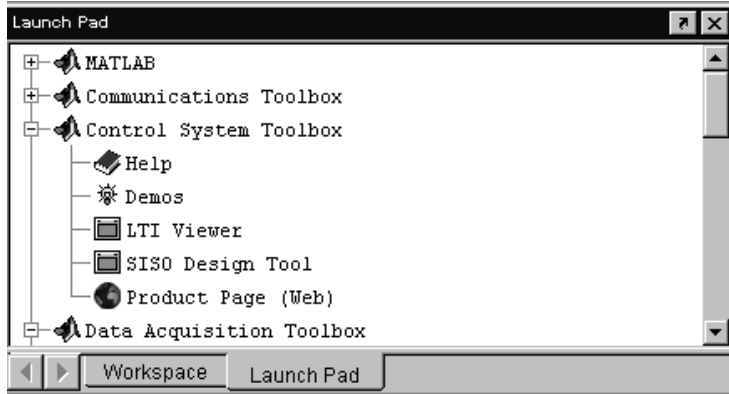


Figure 3.6 Launch Pad window.

Workspace window: This shows the name, size, bytes and class of any variable we define in MATLAB environment. For example, in Figure 3.7 we can find that we have used a variable x in the command window and the variable x is a double array of size 1×3 . We can therefore check the size of variables or the names of variables that we have been using.

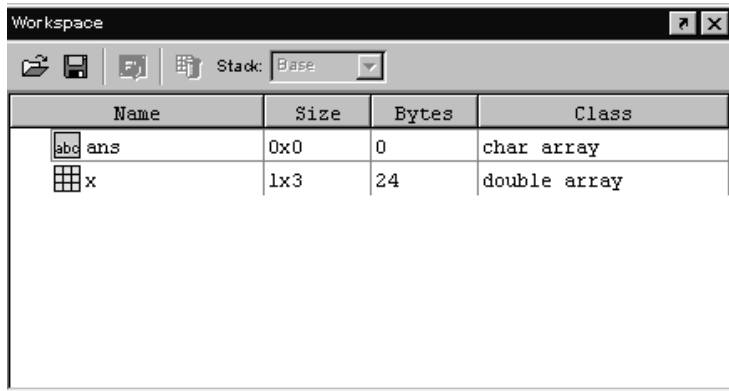


Figure 3.7 Example of entries in the Workspace.

If we double-click on the name of the variable, the array editor window appears (Figure 3.8). We can change the format of the data, for example from integer to floating point, and the size of the array. We can do this by changing the options in the window.

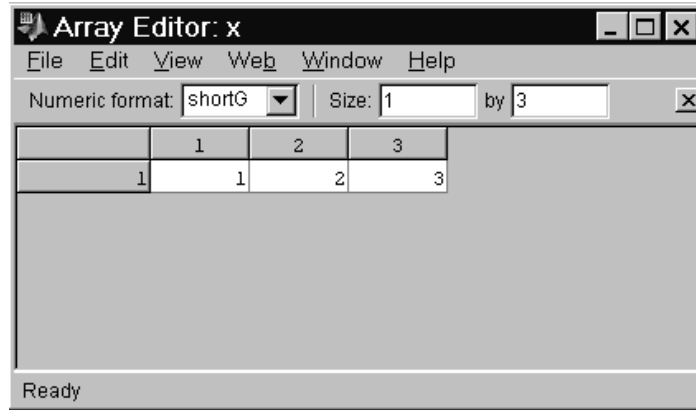


Figure 3.8 Array editor window.

Command History window: This window (Figure 3.9) contains a record of all the commands that we type in the command window. By double-clicking on any command, we can execute it again.

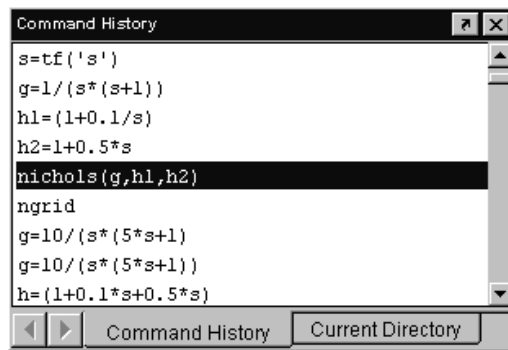


Figure 3.9 Command History window.

Current Directory window: This window (Figure 3.10) shows the directory on the hard disk which we use to run or save our programs. The default directory is the `\MATLAB\work`

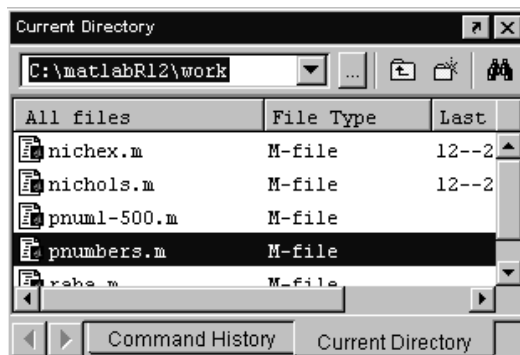


Figure 3.10 Current Directory window.

directory. We can change this directory to our own directory on the hard disk or a floppy drive by typing the name in the pull-down window.

Although these are the windows immediately available on the desktop, we also introduce the Help window. This can be activated by choosing Help from View menu. We can use this window (Figure 3.11) to search for help on any commands or functions.

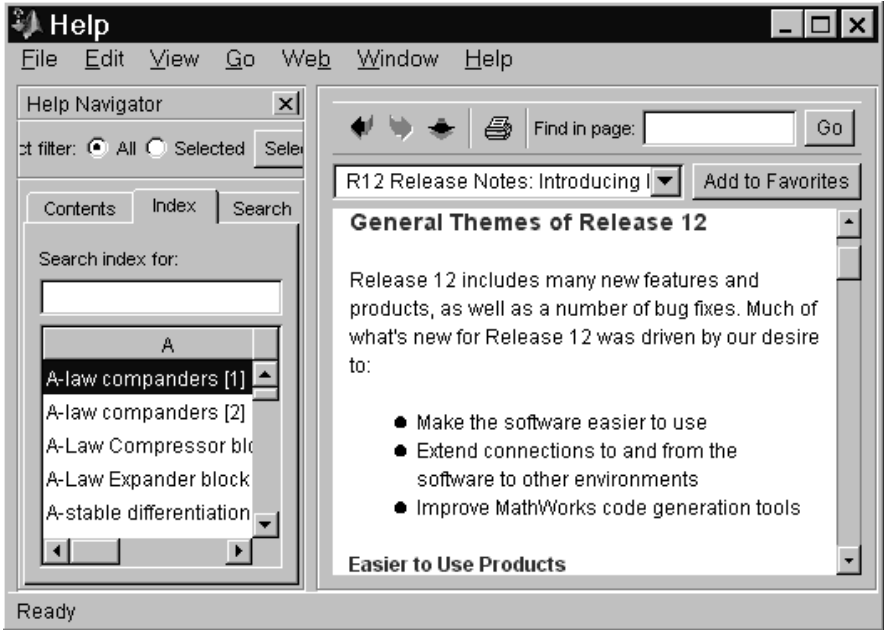


Figure 3.11 Help window (MATLAB v6).

3.13 M-files and functions

We often repeat sequences of commands, or need to enter large amounts of data that we may need to alter. We can create files that contain MATLAB code that save us rewriting the commands. These files are called *M-files*. We can create M-files using the MATLAB editor or any other text editor. Using the MATLAB editor has the advantage that it can also be used as a debugger to find any possible errors in our programs.

The MATLAB editor can be invoked by selecting File from the pull-down menu, then New and then M-file, as shown in Figure 3.12. The editor window will open and we can type in our program.

After typing the program, we should save it on the hard disk. If we select File from the pull-down menu on the editor window, and then Save, the program will be saved in the current directory. If we want to save the file in a specific directory, we should change the working directory of MATLAB using the Directory window and then save the program.

M-files can be run like any MATLAB commands or functions by typing the name of the file and the input data if needed in the MATLAB Command window and then pressing return.

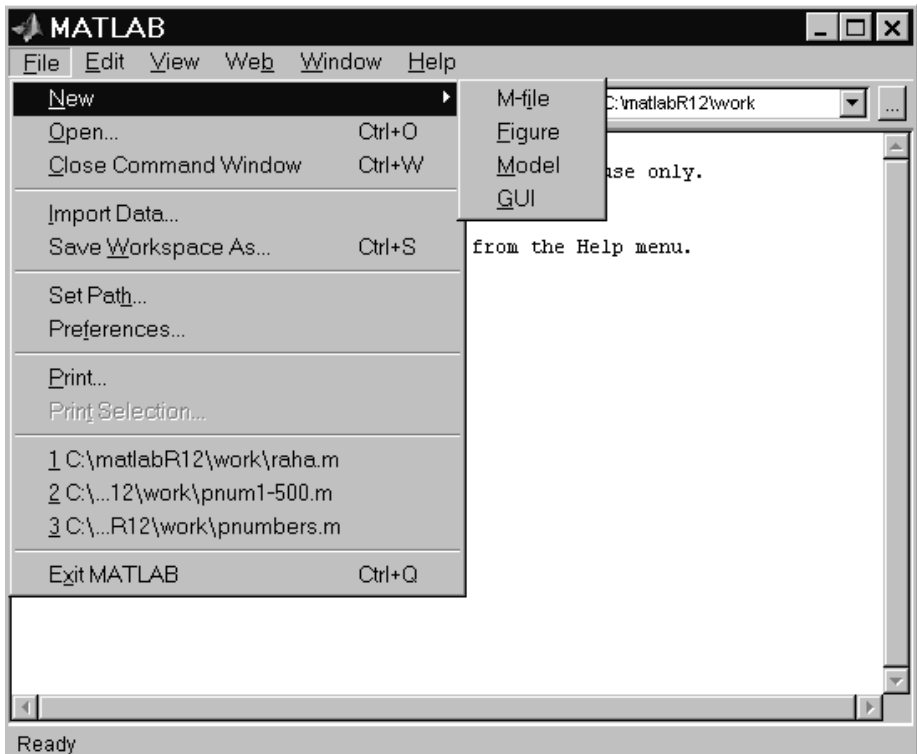


Figure 3.12 Opening an M-file from the MATLAB Command window.

We can write two types of M-file:

1. Programs that do not need any input or output arguments. They operate on the data in the workspace.
2. Functions which require input data and return output arguments. Any internal data in the function will then be local to the function and cannot be accessed from the workspace.

Example Write an M-file which enters the following transfer function, plots the two step responses corresponding to $k = 1$ and $k = 2$ on the same plot and titles the plot:

$$G_1(s) = \frac{k}{s+1}$$

Select File, New and the M-file to open the editor.

Enter the following code

```
s=tf('s');
k=1;
g=k/(s+1);
step(g);
hold;
k=2;
```



```

g=k/(s+1);
step(g);
title ('Step response for k=1 and k=2');

```

Now select File, Save As and save the file as `twoplots.m` in the working or your own directory. Return to the MATLAB Command window. By typing

```
twoplots
```

at the MATLAB prompt the file should run and produce the plot. If it does not run, the most likely error, apart from a typing error, is that the directory you have saved the file into is not in the MATLAB path. If you save to `MATLAB\work`, then the file will be found. Alternatively you can add your own working directory to the MATLAB path.

Example We would like to write a MATLAB program to add four sine waves of amplitudes $A_1 = 1$, $A_2 = 0.5$, $A_3 = 0.333$ and $A_4 = 0.25$ and frequencies in rad/s of $\omega_1 = 1$, $\omega_2 = 2$, $\omega_3 = 3$ and $\omega_4 = 4$, respectively and then plot the result for the time horizon $t = 0$ to $t = 10$ seconds with a sample time of 0.01 s.

We first write a function to automate the process of calculating the sine wave given its amplitude, A , its frequency w rad/s, the time period T and the sample interval DT .

We can then call this function four times to produce the final program we required.

Creating the function for the sine wave calculation

Select File, New and the M-file to open the editor.

Writing a function

Functions are defined by writing an M-file with a specific format. The first line of the M-file must be of the form:

```
function [x1out, x2out, ...] = function-name(data1, data2, ..., datan)
```

where

`x1out, x2out, ...` are the names of the variables that will be returned from the function

`function-name` is the name of the function that can be used from the command line (or a script M-file). *It must also be the name of the M-file that the function is saved as.*

`data1, data2, ..., datan` are input variables that should be entered at the command line and are required by the function to complete the calculation.

After the first line, a function M-file is written as a regular script M-file.

Remarks

1. The function name should be different from any used by MATLAB and its toolboxes.
2. We can add comments to an M-file by typing `%`. Anything on the same line after the `%` will be ignored by MATLAB when executing the M-file. In the MATLAB Editor, comments are coloured green.

Type in the program shown in the MATLAB Editor window (Figure 13.13) and then save it as `myexample.m`.

We can now test the function by typing in the following commands:

```
t = 0:0.01:2;
```

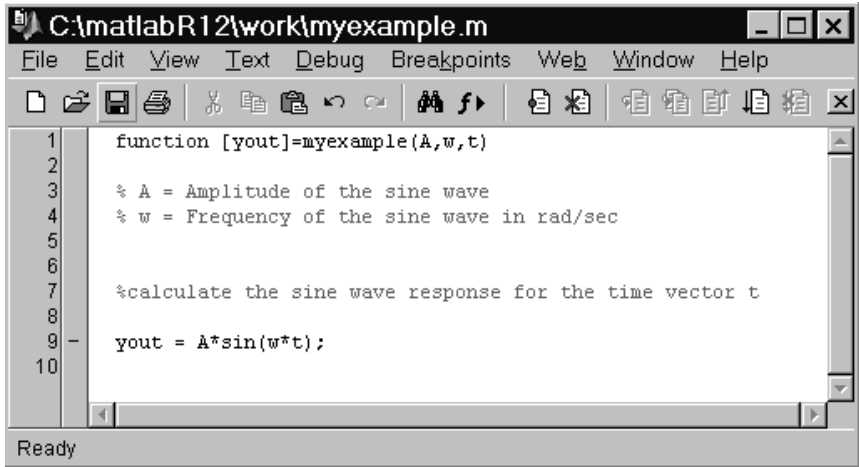


Figure 13.13 Function for calculating sine wave response.

```

y1=myexample(1,10,t);
plot(t,y1)

```

This should give the response in Figure 3.14.

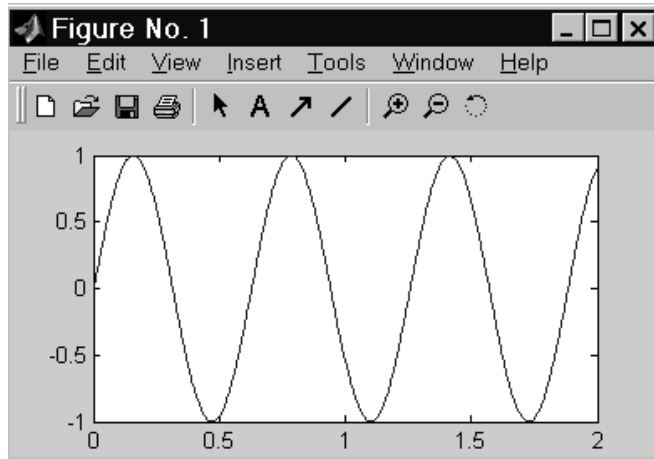


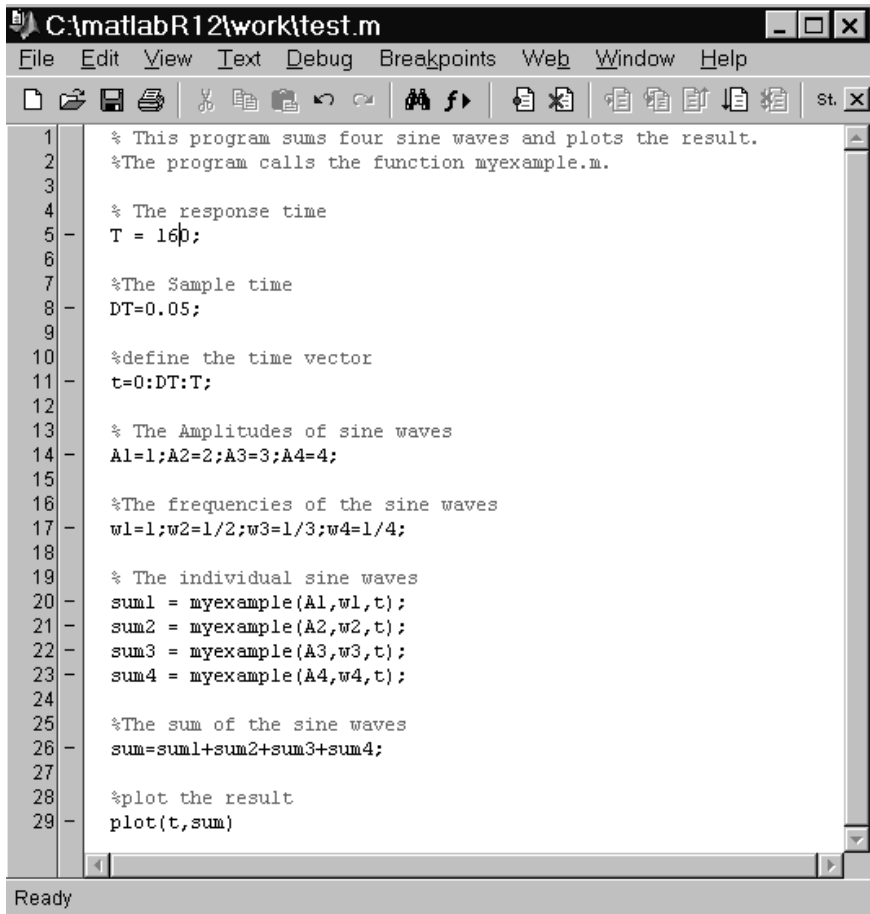
Figure 3.14 Output of function myexample.

Writing the main programs

We have to define all the data in the main programs and then call the function `myexample.m` four times and add the results. We can then plot the response of the sum of the sine waves.

Type in the program shown in the command window of Figure 3.15 and save it as `mytest.m`.

Run `mytest` to obtain the response in Figure 3.16.



```

C:\matlabR12\work\test.m
File Edit View Text Debug Breakpoints Web Window Help
[Icons] st. X
1 % This program sums four sine waves and plots the result.
2 %The program calls the function myexample.m.
3
4 % The response time
5 T = 160;
6
7 %The Sample time
8 DT=0.05;
9
10 %define the time vector
11 t=0:DT:T;
12
13 % The Amplitudes of sine waves
14 A1=1;A2=2;A3=3;A4=4;
15
16 %The frequencies of the sine waves
17 w1=1;w2=1/2;w3=1/3;w4=1/4;
18
19 % The individual sine waves
20 sum1 = myexample(A1,w1,t);
21 sum2 = myexample(A2,w2,t);
22 sum3 = myexample(A3,w3,t);
23 sum4 = myexample(A4,w4,t);
24
25 %The sum of the sine waves
26 sum=sum1+sum2+sum3+sum4;
27
28 %plot the result
29 plot(t,sum)
Ready

```

Figure 3.15 Program to sum four sine waves.

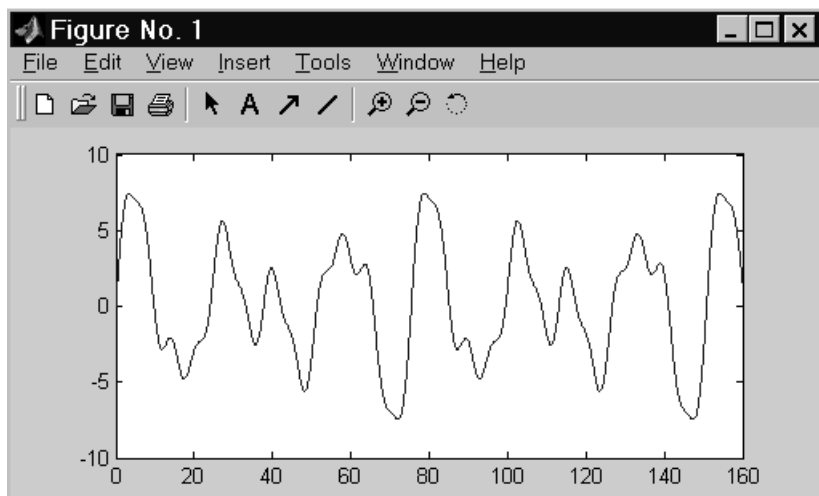


Figure 3.16 Sum of four sine waves; output of program mytest.

3.14 SISO Design Tool: r1tool

MATLAB includes an interactive design tool called SISO Design Tool which is activated by typing `r1tool` in the MATLAB command window. This provides a convenient method for linking time, frequency and root locus design together. We briefly describe its use here and will return to it in the appropriate sections throughout the text. By typing `r1tool` in the MATLAB window, the SISO Design Tool window (Figure 3.17) appears.

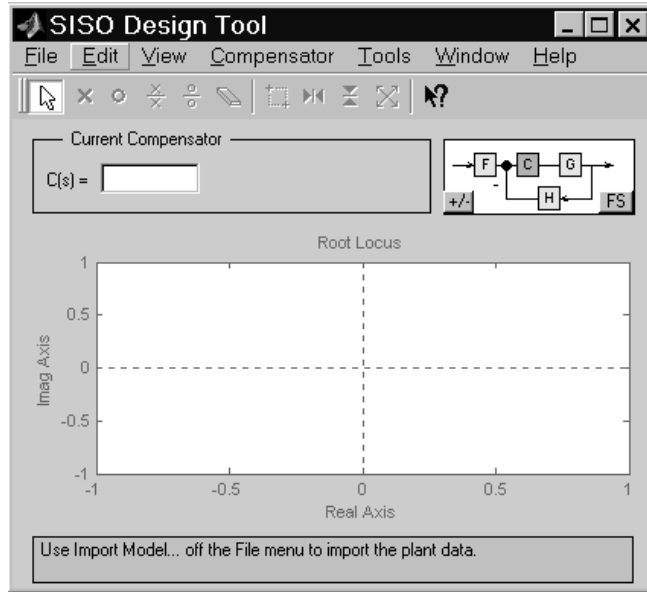


Figure 3.17 SISO Design Tool window.

We can import models into the design tool by clicking on the File menu and then choosing Import. The Import System Data window will appear (Figure 3.18).

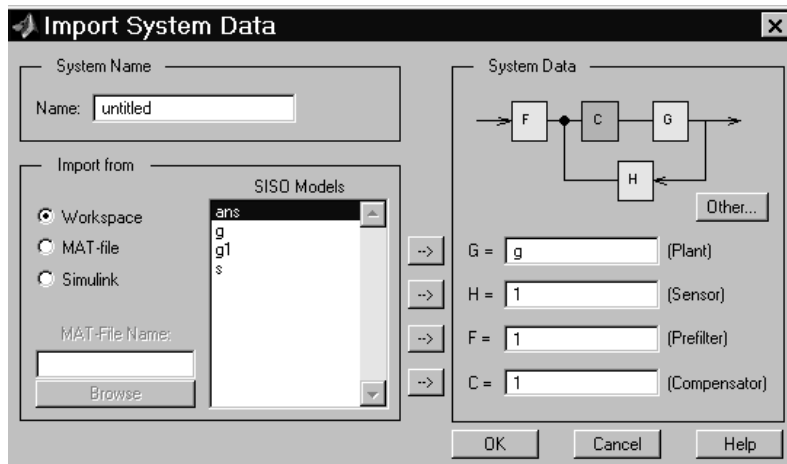


Figure 3.18 Import System Data window.

We find that on the left-hand side we can choose which area we would wish to import our data from. If we choose the Workspace, we find a list of SISO models in our workspace which we could import. On the right-hand side we see that there is a basic feedback loop with transfer function blocks representing the plant or process, G , the sensors or measurement, H , the compensator or controller, C , and a prefilter, F , which acts on the reference input.

We can highlight one of the SISO models listed. Then, by clicking on the arrow opposite G and then OK the model will be imported and the SISO Design Tool window will change to that shown in Figure 3.19.

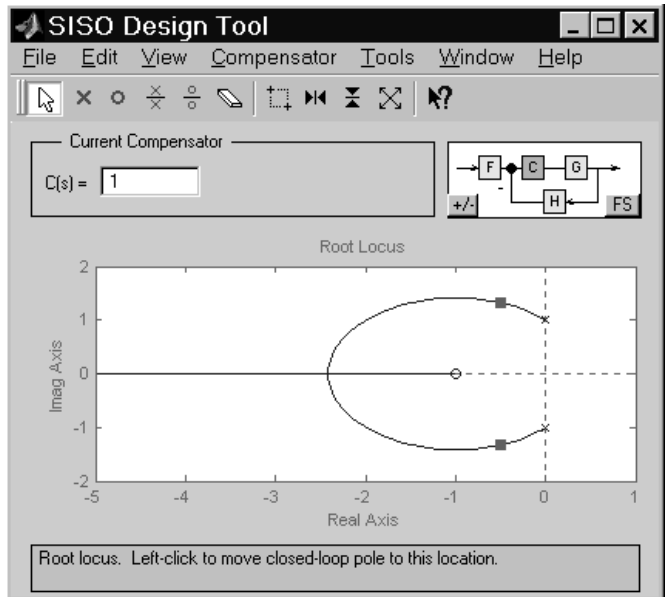


Figure 3.19 File imported to SISO Design Tool window.

Note that the models should be entered into the MATLAB environment before we can import them into the design tool.

We can either import the compensator (controller) or enter it using the red cross (pole) or the circle (zero) on the top left-hand corner of the SISO design tool window.

By choosing the Tools menu and Loop responses, we can display the Step, Impulse, Bode, Nyquist or Nichols plots for the system, as listed in Figure 3.20.

There are several features which can be examined by clicking on various points:

1. By clicking on any point on the graph, we can see the loop gain in the text box at the bottom of the r1 tool window.
2. By clicking on the small red box, we can also see the damping and the natural frequency of the closed-loop system.
3. We can display the various plots, such as Bode or step response. By *right clicking* on each plot, we can find the values of various characteristics that are associated with each of the individual plots.

We will discuss the SISO Design Tool in more detail in Chapter 13.

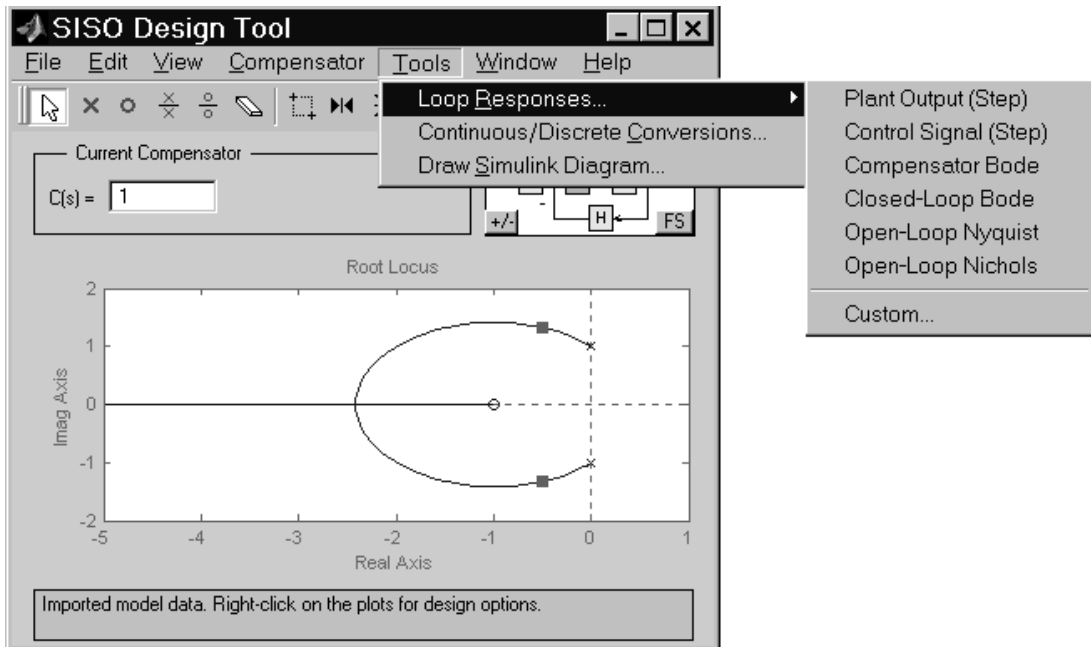


Figure 3.20 Response available from SISO design tool.

What we have learnt

- ✓ To enter numbers vectors, matrices and transfer functions easily into the package.
- ✓ To perform several mathematical calculations and operations on different variables.
- ✓ To plot graphs of results.
- ✓ To form a step response of a transfer function and plot the output response.
- ✓ To write M-files and M-functions.

General MATLAB commands

MATLAB command	Usage	Description
'	x'	Transpose
.*	$x.*y$	Element by element multiplication
./	$x./y$	Element by element division
log10	$b=\log_{10}(a)$	$b = \log_{10}a$
Plotting		
plot	$\text{plot}(t,y)$	Plots y against t
	$\text{plot}(t,x(:,2))$	Plots second column of matrix x against t
hold	hold on	hold is a toggle switch which will hold or release plot on current figure
	hold off	
ginput	$\text{ginput}(n)$	LH mouse click n times on graph and n pairs of (x,y) coordinates are given
zoom		LH mouse click on graph to zoom in, RH mouse click to return to original
gtext	$\text{gtext}('text')$	LH mouse click on graph to place 'text'
figure	$\text{figure}(n)$	Changes (or creates) current figure as figure number n
close	$\text{close}(n)$	Closes figure number n
Transfer functions		
tf	$g=\text{tf}(\text{num},\text{den})$	Creates transfer function from numerator and denominator coefficient vectors
	$s=\text{tf}('s')$	Creates the Laplace transform s , which can be used to write the transfer function directly
zpk	$[\text{num},\text{den}]=\text{zpk}(\text{zeros},\text{poles},\text{gain})$	Creates transfer function from vectors of zeros and poles and the multiplying gain
	$s=\text{zpk}('s')$	Creates the Laplace transform s , which can be used to write the transfer function directly
feedback	$gcl=\text{feedback}(g,h)$	gcl is closed loop transfer function if unity feedback used and g is forward transfer function and h is feedback transfer function
conv	$n3=\text{conv}(n1,n2)$	$n3$ creates the numerator coefficient vector of the multiplication of two numerator polynomials $n1$ and $n2$
deconv	$[\text{xx},\text{R}]=\text{deconv}(z,y)$	Divides polynomial given by z by polynomial y , puts answer in xx and remainder in R

General MATLAB commands (*continued*)

pole	pole(g)	Provides poles of transfer function g
zero	zero(g)	Provides zeros of transfer function g
roots	roots(den)	Provides roots of polynomial equation den(s)=0
pzmap	pzmap(g)	Pole-zero map of transfer function g
Time response		
step	step(g)	Plots the step response of transfer function g
	step(num,den)	Plots the step response of transfer function given by numerator and denominator
	step(K*g)	Plots the step response of transfer function K*g
InputDelay	set(g, 'InputDelay', 3)	Sets the time delay of the transfer function g to 3 seconds
Bode plots		
max (min)	[xmax,i]= max(x)	Give maximum (minimum) output of vector x at position i in vector
margin	margin(g)	Gives gain margin and phase margin
	margin(num,den)	Gives gain margin and phase margin
bode	bode(g)	Plots frequency response plot of transfer function g
	bode(num,den)	Plots frequency response plot of transfer function given by numerator (num) and denominator (den)
	[mag,phase,w] = bode(g)	Magnitude (in absolute value) and phase (degrees) and frequency vector of frequency response plot
	[mag,phase,w]= bode(num,den)	Magnitude (in absolute value) and phase (degrees) and frequency vector of frequency response plot
Nichols		
nichols	nichols(g)	Plots Nichols plot of transfer function g
	[mag,phase,w]=nichols(g)	Magnitude (in absolute value) and phase (degrees) and frequency vector of frequency response plot
ngrid	ngrid	Plots the Nichols chart on the Nichols plot
Root locus		
rlocus	rlocus(g)	Produces a root locus plot for transfer function g
sgrid	sgrid	Overlays the lines of constant damping and natural frequency on the root locus plot

Multiple choice

M3.1 Find the correct expression to enter matrix **A** into MATLAB.

$$\mathbf{A} = \begin{bmatrix} 1 & 2 & 3 \\ 0 & -1 & 1 \\ 1 & 0 & 2 \end{bmatrix}$$

- (a) $\mathbf{A}=[1\ 2\ 3, 0\ -1\ 1, 1\ 0\ 2]$
- (b) $\mathbf{A}=[1\ 2\ 3; 0\ -1\ 1; 1\ 0\ 2]$
- (c) $\mathbf{A}=(1\ 2\ 3; 0\ -1\ 1; 1\ 0\ 2)$
- (d) $\mathbf{A}=[1; 0; 1; 2; -1; 0; 3; 1; 2]$

M3.2 Which MATLAB expression can be calculated for the vectors $\mathbf{x}=[1\ 2]$ and $\mathbf{y}=[1; 2]$?

- (a) \mathbf{y}/\mathbf{x}
- (b) $\mathbf{x}*\mathbf{y}$
- (c) $\mathbf{y}*\mathbf{x}$
- (d) $\mathbf{x}*\mathbf{y}$

M3.3 Find the correct expression to enter transfer function $g(s)$ into MATLAB.

$$g(s) = \frac{(s+1)}{(s+2)(s+3)}$$

- (a) $g(s) = ((s+1)/(s+2))*(s+3)$
- (b) $g(s) = ((s+1)/(s+2)(s+3))$
- (c) $g = ((s+1)/(s+2))*(s+3)$
- (d) $g = (s+1)/((s+2)*(s+3))$

M3.4 A function is useful to simplify:

- (a) Repeated calculation
- (b) Difficult expressions
- (c) Large matrices
- (d) (a) and (b)

M3.5 The correct expression for entering the function $y(t) = (3t^3 + 1)$ is:

- (a) $y(t) = (3t^3 + 1)$
- (b) $y = (3t^3 + 1)$
- (c) $y = (3*t.*t.*t + 1)$
- (d) $y = (3*t^3 + 1)$

M3.6 To find the step response of transfer function $g(s)$ and $H(s)$, we use:

- (a) `step(g(s),H(s))`
- (b) `step(H(s),G(s))`
- (c) `step(g,H)`
- (d) `step(g,s)`

M3.7 If a transfer function $G(s)$ is entered as $g = 1/(s+1)$

the command to find the frequency response of the system $G(s)$ is:

- (a) `bode(G)`
- (b) `bode(g)`
- (c) `BODE(g)`
- (d) `Bode(G)`

M3.8 If we have complex poles in the transfer function:

$$g_3 = \frac{4(s+5)}{(s+2+3j)(s+2-3j)}$$

the transfer function can be entered easily as:

- (a) $g_3 = 4*(s+5)/((s+2+3*j)*(s+2-3*j))$
- (b) $g_3 = 4*(s+5)/((s+2+3j)*(s+2-3j))$
- (c) $g_3 = 4*(j+5)/((s+2+3*j)*(s+2-3*j))$
- (d) $g_3 = 4(s+5)/((s+2+3*j)*(s+2-3*j))$

M3.9 To enter the transfer function:

$$g_1 = \frac{3s+1}{s^2+3s+2}$$

we can use:

- (a) `num = [3 1]; den = [1 3 2];`
`g1 = tf(num,den);`
- (b) `num = [3 1]; den = [1 3 2];`
`g1 = tf(den,num);`
- (c) `num = [3; 1]; den = [1 3 2];`
`g1 = tf(num,den);`
- (d) `num = (3 1); den = (1 3 2);`
`g1 = tf(num,den);`

M3.10 Which expression would produce a sine wave plot?

- (a) `t=0:0.25:10; y = sin(t); plot(t,y)`
- (b) `t=0:0.25:10; y = sin(t); plot(y,t)`
- (c) `y=0:0.25:10; t = sin(y); plot(t,y)`
- (d) `t=10:0.25:0; y = sin(t); plot(ty)`

Questions: practical skills

Q3.1 Create a time vector t with points spaced at 0.1 seconds between 0 and 10 seconds. Plot the function $y = 2 \sin t + 3 \cos t$ and use the max function to find the maximum value of the function. Verify using the cursor.

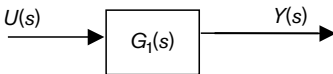
Q3.2 Find the roots of the following polynomial equations:

- (a) $s^3 + 2s^2 + 1 = 0$
- (b) $p^3 + 1 = 0$
- (c) $-3s^2 - 5 + 2 = 0$

Q3.3 For the following transfer function descriptions, enter the MATLAB commands to provide the appropriate transfer function expression.

- (a) System G_A has a unity gain at $\omega = 0$ rad/s, a pole at -3 and a zero at -4 .
- (b) System G_B has a gain of 10 at $\omega = 0$ rad/s, a zero at -1 and two poles at $-1 + j0.5$ and $-1 - j0.5$.
- (c) System G_C has an infinite gain at $\omega = 0$ rad/s, a zero at -6 , two poles at the origin and a pole at -4 .

Q3.4 For the following system, where $G_1(s) = 1/(\tau s + 1)$, and $U(s)$ is a unit step input, write an M-file to plot the output $y(t)$ for the three cases $\tau = 2$, $\tau = 4$ and $\tau = 6$ seconds. Plot the three responses on the same graph.



Q3.5 Given the following transfer functions:

$$g_1(s) = \frac{s^2 + 2s + 1}{s^3 + 3s^2 + 1}, \quad g_2(s) = \frac{2s^2 - 3}{(s^3 + 1)(s^2 + s + 1)}$$

- (a) Enter $g_1(s)$ using row vectors.
- (b) Enter $g_2(s)$ using the `s=tf('s')` notation. Record from the screen the MATLAB commands that were entered.
- (c) Find the zeros of the numerator and denominator of $g_1(s)$ using the `roots` command.
- (d) Compare with the results using the `zero` and `pole` commands. Examine where the roots are using the `pzmap` function.
- (e) Find the zeros and poles of $g_2(s)$ using appropriate commands. Use the `pzmap` function to ascertain if the system is stable (*poles* must lie in the left-half plane).

Q3.6 Write an M-file to solve the following set of equations.

$$\begin{bmatrix} 2 & 1 & 0 & 0 \\ 1 & 2 & 1 & 0 \\ 0 & 1 & 2 & 1 \\ 0 & 0 & 1 & 2 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ t \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ -2 \end{bmatrix}$$

Q3.7 The Laplace transform of a time delay is e^{-Ts} . The following transfer function has a time delay of 2 seconds.

$$g(s) = \frac{10e^{-2s}}{5s + 1}$$

- (a) Enter a transfer function $g1$ which represents $g(s)$ without the time delay.
- (b) Enter `get(g1)`; we see in the list of attributes of $g1$ that `InputDelay` is zero.
- (c) Use the following commands to enter $g(s)$:

```
g=g1;
set (g, 'InputDelay', 2)
```

(d) Plot the step response of the system with and without the time delay and compare the result.

Problems

P3.1 Write a MATLAB function to check whether the number N is prime. (Hint: investigate the mod function.)

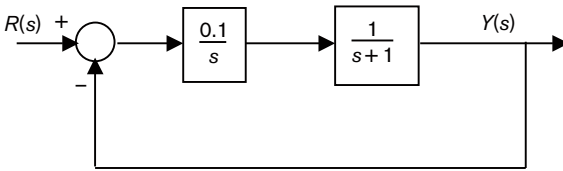
P3.2 The input–output relationship of a servo system can be represented by the following complex function:

$$g(j\omega) = \frac{\theta}{V_i} = \frac{K}{j\omega(j\tau\omega + 1)}$$

where K is the gain and τ is the time constant.

- (a) For $K = 10$ and $\tau = 2.5$, calculate the gain and phase of θ/V_i for $\omega = 0.01, 0.05, 0.1, 2, 5, 10$ (rad/s).
 (b) Plot the gain and phase using the semi logx command.

P3.3 Write an M-file to find and plot the response of the following system to an input signal of $R(s) = 10/s$.



P3.4 Plot the frequency response of the following system for $K = 1$.

$$G(s) = \frac{10K}{s(s+5)}$$

- (a) Determine the frequency at which the magnitude plot crosses the 0 dB line.
 (b) What value of K would ensure that the magnitude plot crosses 0 dB at 5 rad/s?

4

Software toolkit: Simulink

What is Simulink? **S**

Simulink is a software package with a graphical user interface for modelling, simulating and analysing dynamical systems. We can implement the block diagrams of the models and the control systems we study in this book and see how they perform. Firstly we *run* a Simulink model and show how a model can be used to give us information about the behaviour of a system.

4.1 Using Simulink for analysis

By typing Simulink in the MATLAB command window we display the Simulink menu, from which we can create a new model or open an existing one. We can also click on the Simulink icon shown in Figure 4.1.

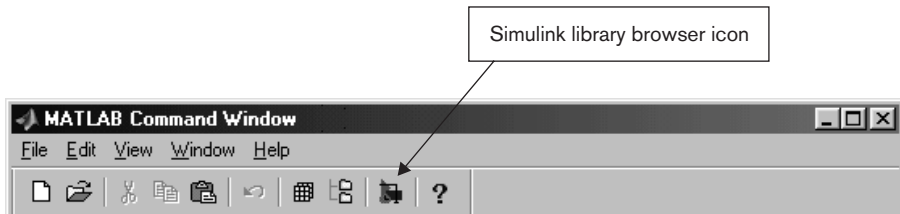


Figure 4.1 MATLAB command window showing Simulink icon.

The Simulink Library Browser will pop up and we can load a previous Simulink model by clicking on the icon shown on the top left-hand corner of the Simulink window in Figure 4.2. We will open a model which represents a house heating system. It is a simplified and modified version of one of the Simulink demos: 'Thermodynamic Model for a House'. It can be downloaded from <http://www.palgrave.com/science/engineering/wilkie/> and stored in your own directory. The file can then be opened from the Simulink command window.

In the model (Figure 4.3), the internal temperature, T_i , will vary depending on the heat input from the house heating system and the external temperature T_o . The mathematical equations which represent the dynamics of the house are 'hidden' from us at the moment; we could access them by double-clicking on the House icon, and we will do so later in this chapter. In the meantime, we use Figure 4.3 to identify some components of this system:

1. The House icon containing the dynamics explaining how the house heats up and cools down.

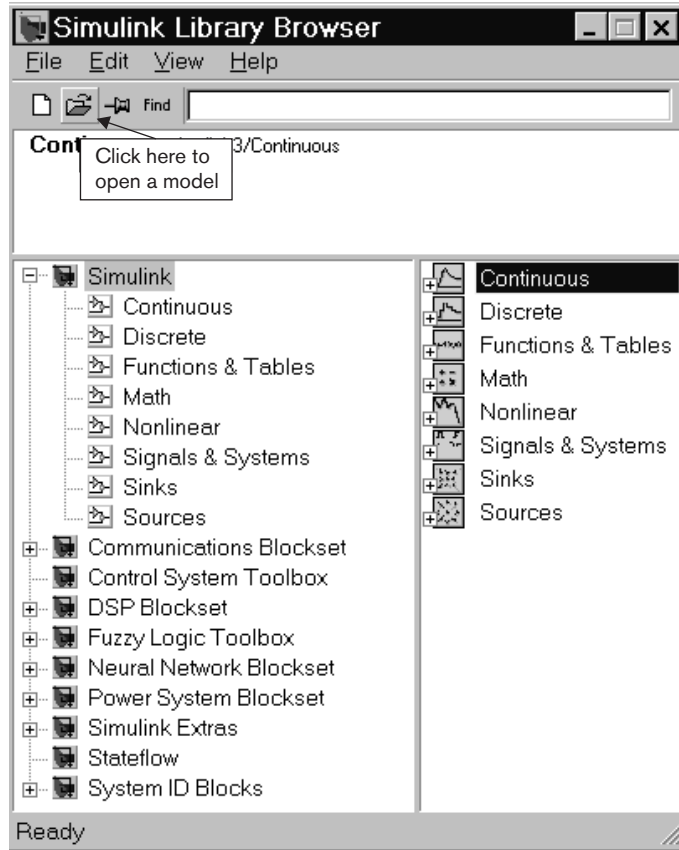


Figure 4.2 Simulink library.

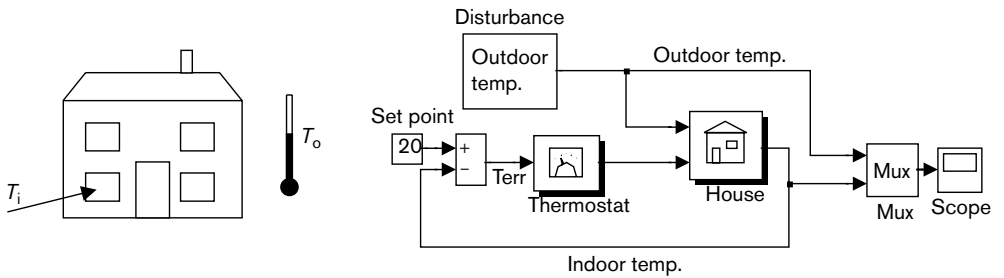


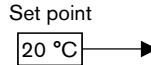
Figure 4.3 Simulink model of house heating system.

2. The Set point block, *comparator* and thermostat block. The set point indicates the desired temperature for the system. The comparator compares the set point with the temperature of the house and the thermostat switches the heating on when the house temperature falls below the set point.
3. The Outdoor temperature, which acts as a *disturbance* and which will affect the House temperature.

- The Mux and Scope blocks. The Mux block takes the two signals (Outdoor temperature and House temperature) and displays them on the 'Scope'.

S Exercise 1: Temperature set point and control

We would like to change the temperature reference of the system.



To change the set point follow these steps:

- Double-click on the set point box and the window shown in Figure 4.4 appears.

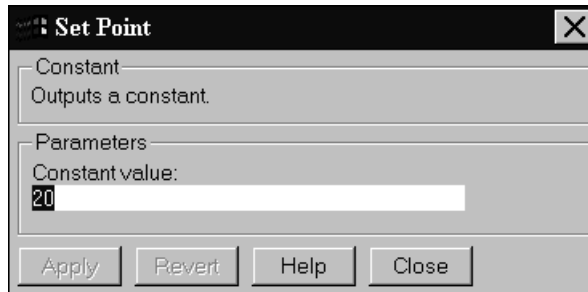


Figure 4.4 Pop-up box for changing the set point.

- Move the pointer to the white box and click the left mouse button.
- Use backspace to clear 20.
- Type in the new set point, say 18.
- Close the window.
- Use the pull-down menus headed Simulation and click on Start to run the simulation.
- Double-click on the Scope to produce a running plot showing how the House temperature and Outdoor temperature are changing.

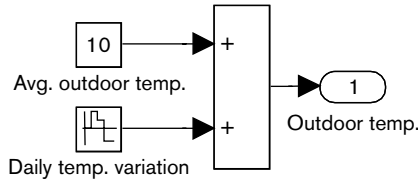
Questions:

- Which line corresponds to the outdoor temperature and which to the indoor temperature?
- Can you explain the behaviour of the House temperature?

An important feature of simulation is to be able to predict what would happen given certain input conditions.

- How would the House temperature behave if the average Outdoor temperature was higher than the current value? Lower than the current value?

We can check our predictions by double-clicking on the Outdoor temperature block. This produces a slightly more complex model:



We note that the original average daily temperature was 10 °C. We can double-click on this and change it to a higher value and re-run the Simulation to see how this would affect the House temperature.

We can do several test runs in this manner. For real industrial processes, it is often impossible to *play* with the system and simulations are necessary for engineers to test out their control designs before applying them to the real process.

On finishing running the simulation, we close all the models by choosing Close from the file menu.

Although we can *use* a Simulink model without knowing what components make up the individual icons, very often we have to derive our own models and then do the analysis. The next section presents the House model in detail. All modelling requires some engineering knowledge, and here we use models of each component in transfer function form (using the Laplace transforms of Chapter 2).

4.2 Detailed house model

This example shows how Simulink uses icons to represent common systems and signals in control engineering. The example in Figure 4.5 is a model of the thermodynamic behaviour of a house whose interior temperature changes due to heat supplied by the heating system. The house temperature is also affected by external environmental changes, such as the change in daily temperature.

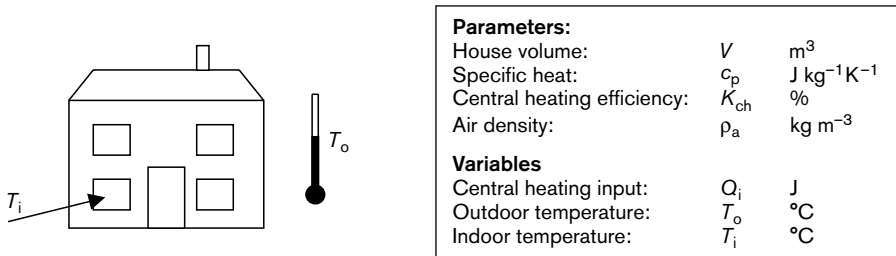


Figure 4.5 House model and parameters.

When we enter a simulation model of a system, we must use some form of representation, or model, for each of the systems or subsystems. These models may already have been provided from previous analysis or information from other engineers. However, the production of a model may require some preliminary analysis. For the house heating system, we can derive a representation using Laplace transforms.

4.2.1 Simple house model

Equation for conservation of heat energy:

$$\text{Rate of change of heat content} = \text{heat from central heating} \\ + \text{heat loss to environment}$$

$$\frac{dQ}{dt} = K_{ch}Q_i + Q_e$$

We note that the heat content, Q , is given by:

$$Q = c_p V T_i \rho_a$$

and that the heat loss to the environment can be expressed as

$$Q_e = K_e(T_o - T_i)$$

where K_e is the parameter which represents the energy loss per °C difference in outdoor and indoor temperature. (This equation shows that if the outdoor temperature is greater than the indoor temperature, then there is a net heat gain (Q_e is positive). Similarly, if $T_o < T_i$, there is a net heat loss, $Q_e < 0$.)

The conservation equation can now be written as

$$\rho_a c_p V \frac{dT_i}{dt} = K_{ch}Q_i + K_e(T_o - T_i)$$

Manipulating this equation gives:

$$\frac{\rho_a c_p V}{K_e} \frac{dT_i}{dt} = \frac{K_{ch}}{K_e} Q_i + (T_o - T_i)$$

or

$$\frac{\rho_a c_p V}{K_e} \frac{dT_i}{dt} + T_i = \frac{K_{ch}}{K_e} Q_i + T_o$$

This is a first-order differential equation and can be written in Laplace transforms as:

$$(\tau s + 1)T_i(s) = KQ_i(s) + T_o(s)$$

or

$$T_i(s) = \frac{K}{(\tau s + 1)} Q_i(s) + \frac{1}{(\tau s + 1)} T_o(s)$$

where $\tau = \rho_a c_p V / K_e$ and $K = K_{ch} / K_e$.

The equation shows how the indoor temperature is affected by both the heat input due to the central heating system and the outdoor temperature. This can be shown as a block diagram (Figure 4.6).

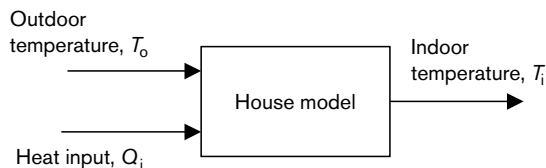


Figure 4.6 Block diagram of simple house model.

4.2.2 Simulink model of house heating control system

The house model and the control for the heating system can be modelled in Simulink as shown in Figure 4.7.

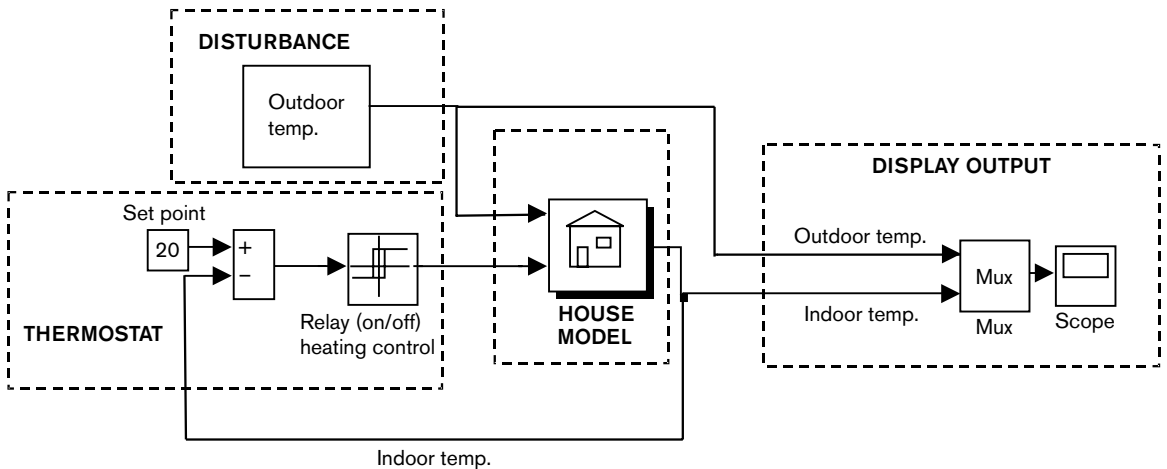


Figure 4.7 Block diagram for house and control system.

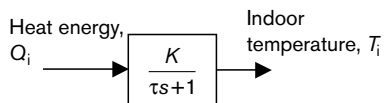
This Simulink model of the thermodynamic behaviour of a house can be divided into four subsystems:

- A The House model**, which contains:
 - (i) the effect of the central heating system on the indoor temperature
 - (ii) the effect of the outdoor temperature on the indoor temperature
- B The Disturbance**, which models the variation of the outdoor temperature
- C The Thermostat**, which provides the control for the central heating system based on the reading of the indoor temperature
- D The Display output**, which produces a graph against time of the indoor and outdoor temperatures.

A The house model

- (i) The effect of a heat source on indoor temperature

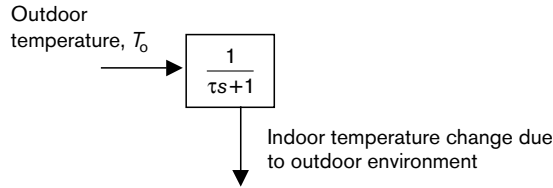
The house is heated by a central heating system or a heater whose output energy is represented by Q_i . As we have seen from our simple house model analysis, we can model the effect of the heat source on the indoor temperature of the house as a first-order system with system gain of K and a time constant of τ minutes.



- (ii) The effect of the outdoor temperature on the indoor temperature

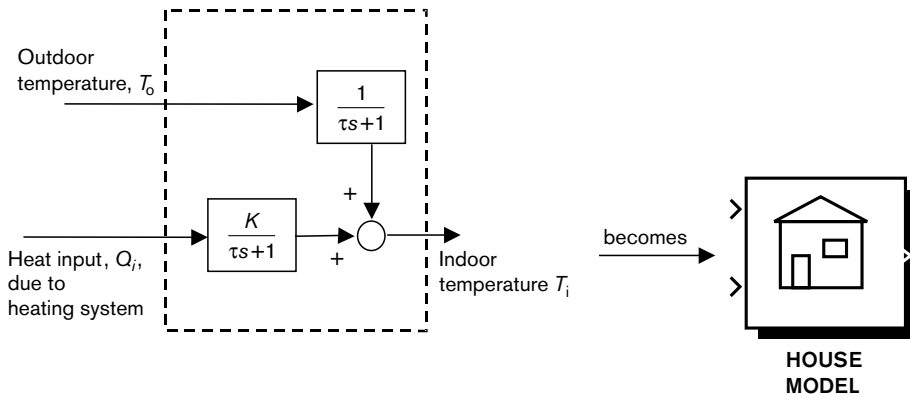
We model the indoor temperature of the house as the summation of the outdoor effect (outdoor temperature) and the heat supplied by the heat source. From the house model

analysis, the effect of the outdoor temperature on energy supplied to the building has first-order dynamics, unity system gain and a time constant of τ minutes.



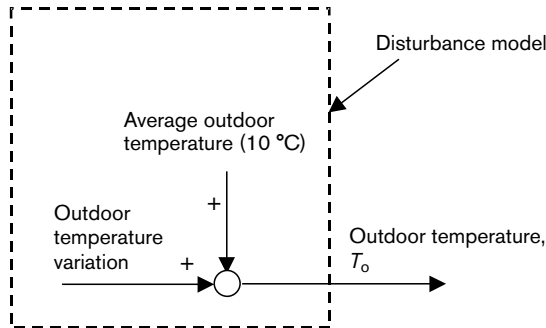
Total change in indoor temperature

We can now connect this block to the indoor temperature block and create a new icon to represent the combined house model.

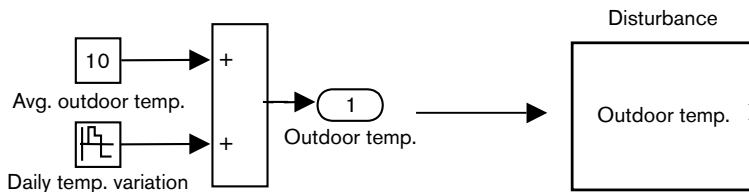


B The outdoor temperature

We assume that the outdoor temperature varies about an average value:



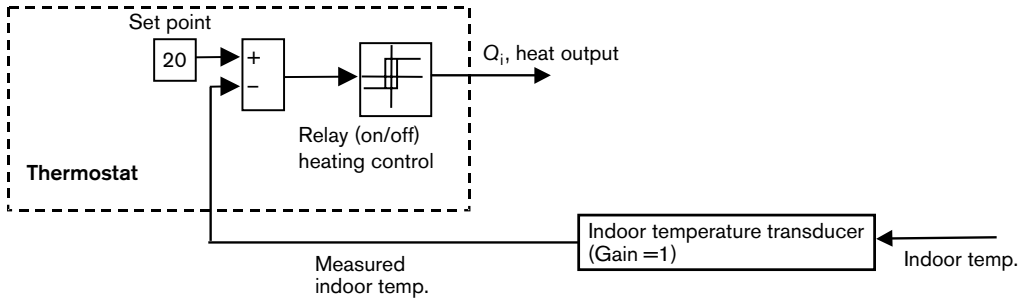
A representation of this model in Simulink is:



Simulink has the capability to group the sub-blocks of this model into a new block for better visualisation of the overall structure of the complicated models.

C The thermostat

The control of the heating system is performed by a thermostat which produces an on-off signal depending on the size of the incoming signal. This is implemented using a relay:



We usually set a desired temperature (the reference temperature) on the thermostat. This value is compared with the measured level of indoor temperature. If the indoor temperature is too low, the error is positive and the thermostat switches on the heater. When the error reduces below a certain threshold, the thermostat switches off the heater.

Note that most thermostats have a sensor to measure the temperature and a knob or dial to set the reference. For simplicity, we have assumed that the sensor has unity gain and the reference is as shown in the block diagram.

The overall model

The overall model brings together the above component models (A, B and C) and has the structure shown in Figure 4.8.

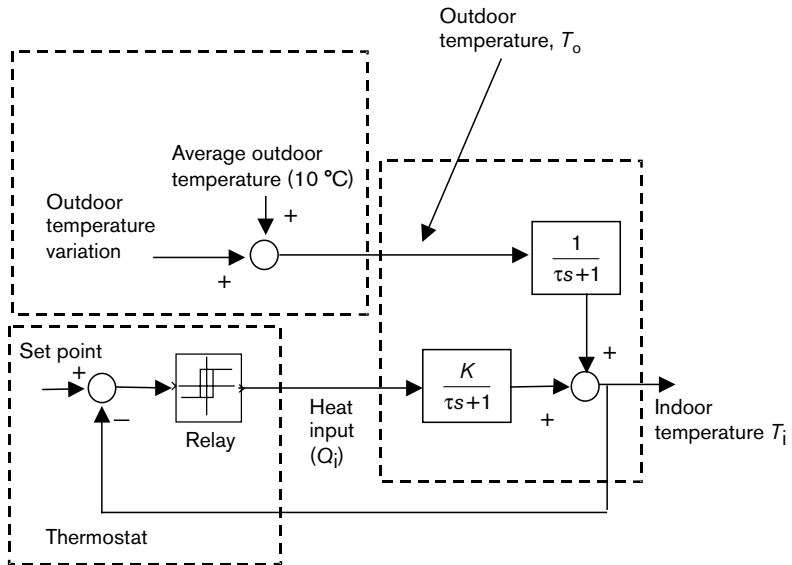


Figure 4.8 Overall model.

These three subsystems (A, B and C) provide the model for the simulation. However, we add the fourth subsystem (display output) in order that we can monitor the change in indoor and outdoor temperatures visually on a plot.

D Display output

When running a simulation, we usually monitor several variables, such as the indoor and outdoor temperatures, so that we can determine if the simulation model is working. To do this we use a 'multiplexing' unit which, in this example, receives two signals and produces a single vector, containing these signals, as the output. This output vector is used as the input to a *scope* which can show graphs against time of the input signals.

The resulting Simulink diagram looks like that in Figure 4.9.

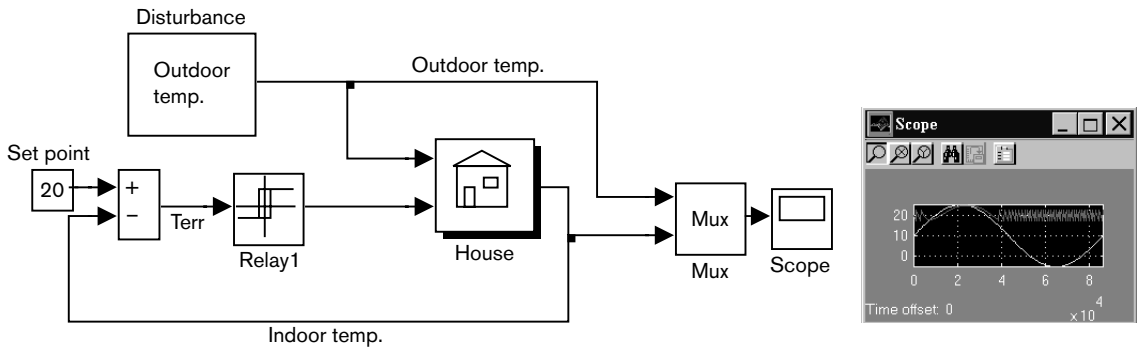


Figure 4.9 High-level model of house heating system.

The house heating system contains several components. We start to construct Simulink models by considering how to build a simpler example of a system: a simple feedback loop.

4.3 Building a simple Simulink model

We use the example in this section to build a model using many of the model building icons available in Simulink. The block diagram of the model is shown in Figure 4.10. This is the model of a first-order system with a proportional controller of gain K_p .

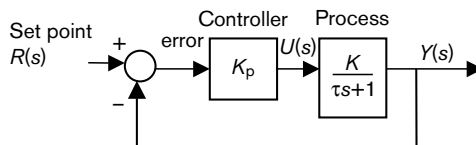
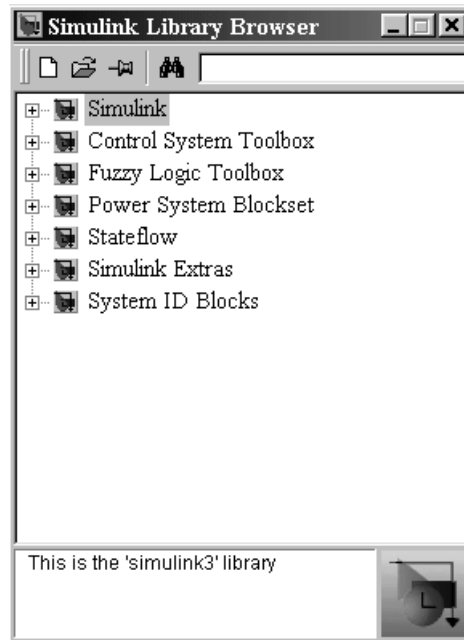


Figure 4.10 Simple feedback system.

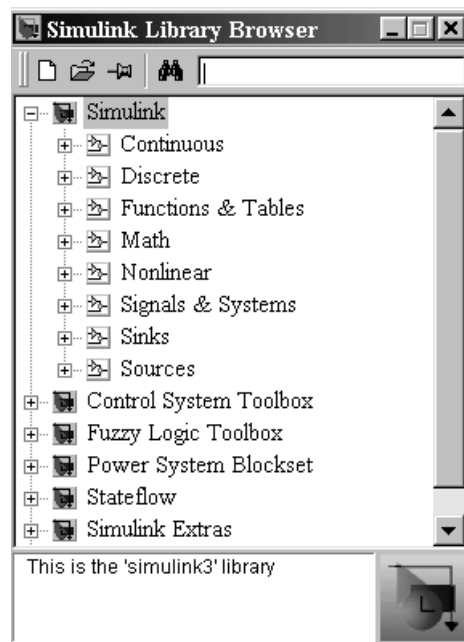
4.3.1 Simulink libraries

To construct a Simulink simulation for the closed-loop system, we use the following steps.

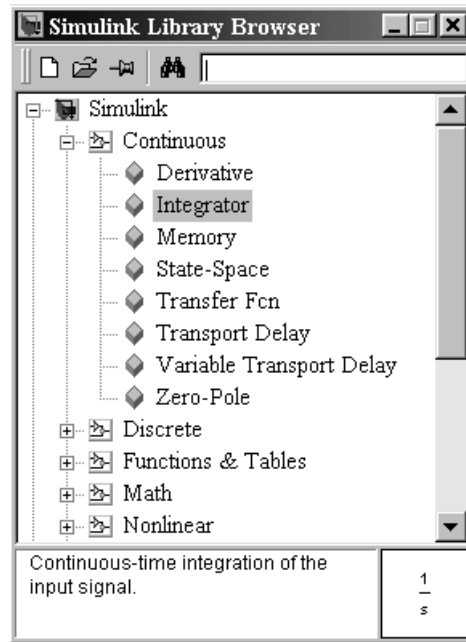
1. Type Simulink in the MATLAB command window to display the Simulink library browser. The Simulink library browser window looks like this:




2. Double-click on Simulink and a number of Simulink Libraries appear, which can be used to build a model.



3. Double-click on Continuous and a number of specific Simulink blocks will appear.



4. To open a modelling window (called *untitled*), click on: 
5. Click on Transfer Fcn, hold the mouse and drag the block onto the modelling window and release the mouse. The transfer function block will appear on the window.

Example Each signal and block of the control system can be represented by one of the Simulink blocks. There are many Simulink blocks that we can use, but for this example we need only the blocks shown in the following figure. The libraries where the blocks can be found have also been shown in Figure 4.11.

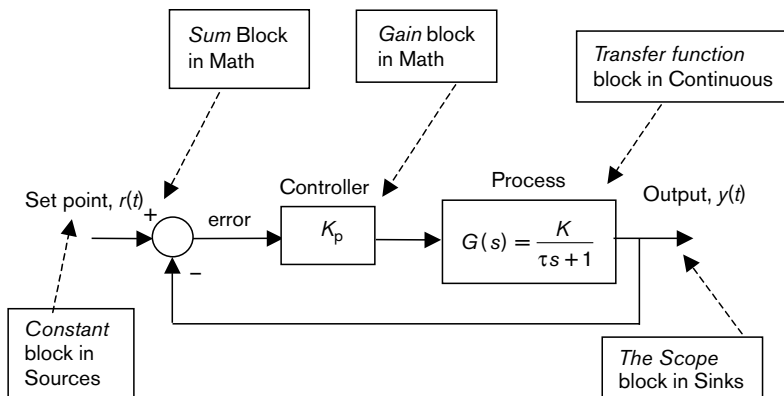


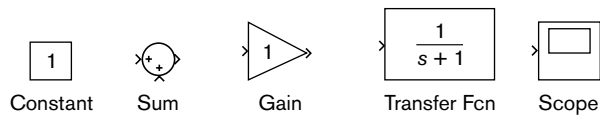
Figure 4.11 Simulink blocks.

The blocks needed are summarised here.

1. *Constant* block in the Sources library
2. *Sum* block in the Math library
3. *Gain* block in the Math library
4. *Transfer function* block in the Continuous library
5. *The Scope* block in the Sinks library

We need to copy all these blocks into the *untitled* window. Follow these steps to copy the blocks.

1. Open the Sources library to access the Constant block. To open a block library, double-click on the library menu. Simulink displays a list that contains all the blocks in the library.
2. To copy the Constant block, click over the Constant block, then press and hold down the mouse button.
3. Now, drag the block into the model window.
4. When the pointer is where you want the block to be in the model window, release the mouse button. A copy of the Constant block is now in the model window. Close the Sources window.
5. In the same way copy the rest of the blocks in the model window. The model window should contain these components:



Note that a block can be moved from one place in the model window to another using the same technique that was used to copy the block.

The addition or subtraction operations for the Sum symbol can be changed by double-clicking on the icon. The default values are given as '| + +', which can be changed to '| + -' to provide the comparator needed in the example for negative feedback. The string provides the number and entry positions (anticlockwise) for the incoming signals on the summation icon, with the '|' symbol indicating no entry point. For example, entering the string '+ + | - +' produces the following icon:



The summation symbol can also be changed to a rectangular shape by choosing the appropriate option after double-clicking on the summation icon.

If we examine the block icons, we can see an angle bracket on the right of the Gain block or two on the left of the Sum block. The $>$ symbol pointing to a block is an **input port**; if the symbol points out of a block, it is an **output port**. A signal travels out from an output port and to the input port of another block through a connecting line.

When the blocks are connected, the port symbol disappears.

Connecting the blocks

To connect the Constant block to the top input port of the Sum block, we use the following steps:

1. Position the pointer over the output port on the right side of the Constant block. Note that the cursor shape changes to cross hairs.
2. Hold down the mouse button and move the cursor to the input port of the Sum block.
3. Release the mouse button. The blocks are connected.

Now connect the other blocks as follows:

1. Output port of the Sum block to the input port of the Gain block.
2. Output port of the Gain block to the input port of the Transfer function block.
3. Output port of the Transfer function block to the input port of the Scope block.
4. To connect the feedback path, follow these steps:
 - (a) Position the pointer on the line between the Transfer function block and the Scope block.
 - (b) Press and hold down the Ctrl key while pressing the left-hand mouse button (or, alternatively, do not use the Ctrl key, but hold down the right-hand mouse button).
 - (c) Drag the pointer down for about 2 cm.
 - (d) Release the mouse button, and line No. 1 appears.
 - (e) Starting from the end of line No. 1, draw line No. 2. Use the same technique.
 - (f) Draw line No. 3.
 - (g) Draw line No. 4. The feedback loop is now connected (Figure 4.12).

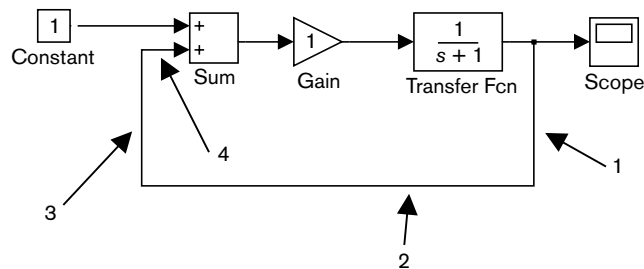


Figure 4.12 Connecting up the feedback loop.

Model data

The block diagram of the simulation model is now complete. Before running the model, we need to enter the model parameters for all blocks. Follow these steps:

1. Double-click on the Constant block: change 1 to 10. Close the window.
2. Double-click on the Sum block: change $+$ $+$ to $+$ $-$. Note that $-$ is for the negative feedback.
3. Double-click on the Gain block: change 1 to 2. Close the window.
4. Double-click on the Transfer function: change the numerator polynomial to [0.5]. This is the gain K . Change the denominator to [10 1]. This is for $\tau s + 1$. Close the window.
5. Double-click on the Scope block. Change Ymax to 10 and Ymin to 0. Close window.

Simulation

From the Simulation menu choose Parameters. Change Stop time to 10. Close the window. To run the simulation, choose Start from the Simulation menu and run the simulation. When the simulation is completed, click on the scope. We can now examine how the system output, $y(t)$, tracks the set point, $r(t)$, in the Scope block.

S Exercise 2

Using the block diagram that we have just entered, we will change the controller gains and examine the effect on the output response.

For the controller gains given in the table, measure the time constant and the steady state error of the closed-loop system. Discuss the effect of changing the controller gain on the system output response.

Controller gain	1	3	5	7	9	10
Time constant						
SS error						

4.4 Development and analysis of the house heating model

Open the house model in Simulink.

4.4.1 Analysis: Effect of thermostat parameters

The thermostat block represents the controller used to control the indoor temperature. To see how the controller is implemented, double-click on the *thermostat* block. The heating control is a relay. The input signal to the relay is the deviation, or error, from the temperature set point and the output signal is the output energy from the heating system which works on an on-off switch. Hence the thermostat switches the heating system on and off depending on the input and the relay parameters.

S Exercise 3

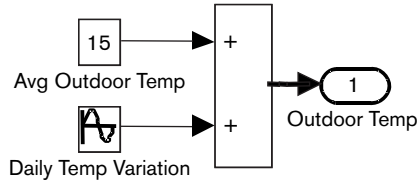
- Run the simulation and explain the behaviour of the room temperature in terms of the varying outdoor temperature signal.
- Double-click on the Relay block and change the limits to ± 5 . Run the simulation and comment on the difference with the previous response.

4.4.2 Analysis: Effect of disturbance (outdoor temperature)

The temperature of the house is affected by the outside temperature, which varies by applying a sine wave with amplitude of 8 °C to a base temperature of 15 °C. This simulates daily temperature fluctuations. To see how this is implemented, follow these steps.

S Exercise 4

- (a) Double-click on the Disturbance block. The following window will open.



- (b) Double-click on the Daily Temp Variation to see how the sine wave is generated.
 (c) Change the value of the frequency of the sine wave by a factor of 2. Close the window.
 (d) Run the simulation to see the effect of the changes made.
 (e) Close the Disturbance window.

The following exercise can be done if you have covered the material in the chapters on modelling of systems and block diagrams.

S Exercise 5: Modelling diagrams

- (a) Draw the block diagram of the house model section of the simulation model.
 (b) Identify the controller, the actuator and the process. Is there a temperature transducer in the diagram? If there is, what does it measure? Note that the thermostat effectively contains a temperature transducer and an on-off controller.

We can save the model by choosing Save from the file menu.

S Exercise 6: Control system study

A pacemaker to regulate the speed of a human heart has a block diagram as shown in Figure 4.13.

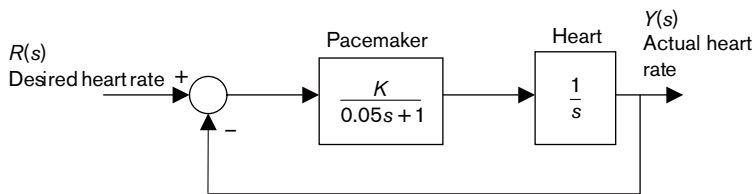


Figure 4.13 Pacemaker system.

- (a) Implement the model in Simulink.
 (b) The overshoot of a second-order system is related to the damping by the following equation:

$$\text{OS}(\%) = 100 \times \exp\left(\frac{-\pi\zeta}{\sqrt{1-\zeta^2}}\right)$$

By changing the value of K in the Simulink model, find a value for which the closed-loop response has a damping factor of 0.35.

- (c) Use transfer function analysis to verify that the value of K found in (b) does give a closed-loop response with a damping factor of 0.35.

Throughout this book we will be illustrating some of the examples with results taken from Simulink models. The simulation package becomes easy to use and enables us to obtain a plot of a system output, which we can then analyse using the tools in MATLAB. In this way both MATLAB and Simulink become commonplace software tools for the control engineer.

What we have learnt

- ✓ How to run a system simulation to predict a system's behaviour.
- ✓ How easy it is to build a simple Simulink simulation.
- ✓ How to analyse and interpret the results that we obtain in simulation runs.
- ✓ How simulations are often structured in layers and that although the top layer looks simple, it often hides some careful modelling analysis.

Multiple choice

- M4.1** Simulink is a tool for:
- (a) Dynamic simulation
 - (b) Control design
 - (c) Data collection
 - (d) (b) and (c)
- M4.2** Simulink files have the extension:
- (a) .m
 - (b) .mdl
 - (c) .mld
 - (d) .dlm
- M4.3** The data in the MATLAB workspace:
- (a) can be used in Simulink blocks
 - (b) cannot be used in Simulink blocks
 - (c) can be used if they are re-entered in the MATLAB Command window
 - (d) can be use if they are re-entered in the Simulink window
- M4.4** The name of the Simulink block for multiplying two signals is:
- (a) Multiply
 - (b) Times
 - (c) Product
 - (d) Transfer Function
- M4.5** The blocks in Sources in the Simulink library can be used to provide:
- (a) the system inputs
 - (b) the system outputs
 - (c) the system inputs and outputs
 - (d) none of the above
- M4.6** Implementing a controller, $K(s)$, in Simulink:
- (a) requires the Sources block
 - (b) requires the Sinks block
 - (c) requires the Transfer function block
 - (d) cannot be done
- M4.7** The output of a Simulink model can be viewed:
- (a) on the scope
 - (b) in the workspace
 - (c) as a plot in a figure
 - (d) all of the above
- M4.8** To change the simulation time horizon in a Simulink model:
- (a) we change the parameter of a transfer function in the model
 - (b) we change the Simulation Parameters under the Simulation menu
 - (c) we cannot change this parameter
 - (d) the model should be scaled accordingly

- M4.9** To create a vector of signals in Simulink:
- (a) we use MATLAB commands
 - (b) we use a multiplexer block
 - (c) we cannot implement vectors in Simulink
 - (d) we can use both (a) and (b)

- M4.10** To implement the trigonometric functions in Simulink, we can use:
- (a) the Transfer Function block
 - (b) the Signal Generator block
 - (c) the Function block
 - (d) (b) or (c)

Questions: practical skills

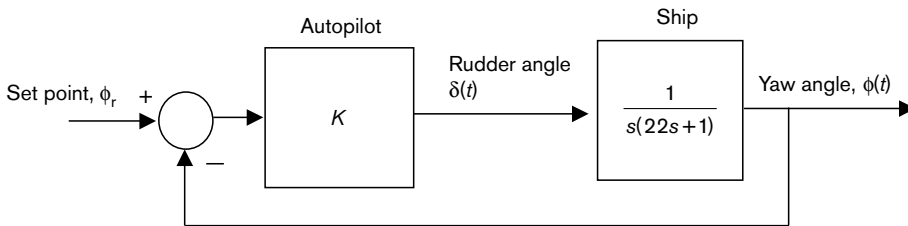
Q4.1 Consider a system described by the following differential equations:

$$Y_1(s) = \frac{2}{3s+1} U_1(s)$$

$$Y_2(s) = 6Y_1(s) + \frac{5}{6s+1} U_2(s)$$

- (a) Implement the model in Simulink.
- (b) Plot $y_2(t)$ for $u_1(t) = u_2(t) = 1$.

Q4.2 A simplified block diagram of a ship autopilot control system is shown below:



- (a) Implement the model in Simulink for $K = 1$ and ϕ_r of 30° . Use a 'slider gain' to implement K .
- (b) Change the value of K to give 10° overshoot.

Q4.3 A tank system is represented by the following first-order differential equation:

$$11.5\dot{h}(t) + h(t) = q(t)$$

where $h(t)$ is the liquid level and $q(t)$ is the input flow rate in m^3/min .

- (a) Develop a Laplace transform model for the system.
- (b) Use Simulink to find the response of $h(t)$ to a step change of $1 \text{ m}^3/\text{min}$, but where there is a slowly varying frequency on the input signal of amplitude 0.1 and frequency 0.05 Hertz.

Q4.4 Stefan's Law states that the rate of change of temperature of a body due to radiation of heat is

$$\frac{dT}{dt} = -k(T^4 - T_0^4)$$

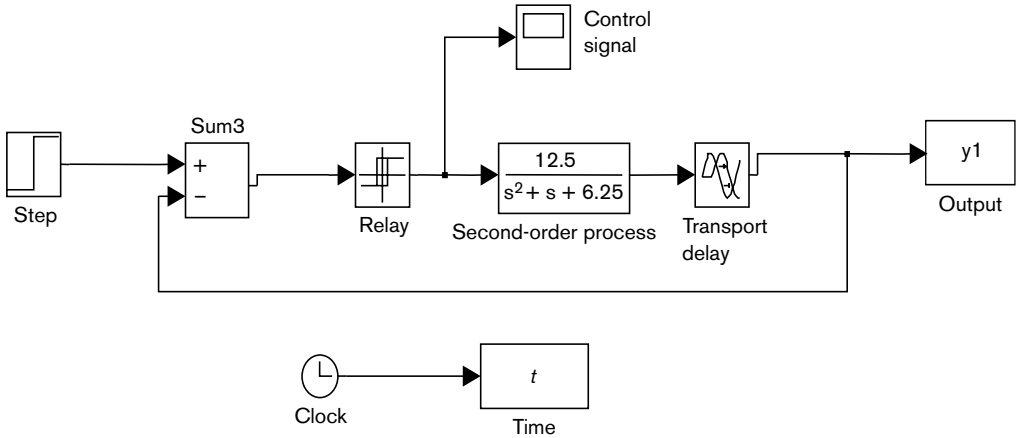
where T is the temperature of the body and T_0 is the temperature of the surrounding medium in K.

- (a) Implement the nonlinear model in Simulink. Hint: use the function block 'Fcn', and 'Transfer Fcn with initial states' (from Simulink Extras).
- (b) Assuming $T_0 = 280 \text{ K}$, $k = 1.0 \times 10^{-11}$ and the initial temperature $T(0) = 600 \text{ K}$, how long does it take for the body temperature to reach steady state?

Problems

P4.1 Simple on/off control

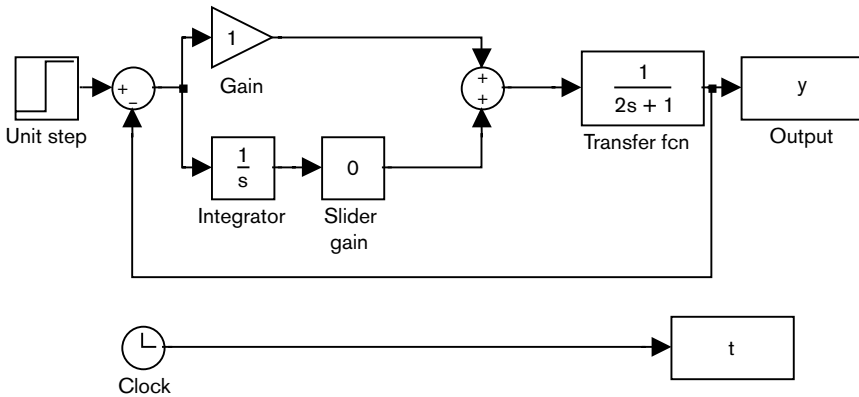
The following model shows how a simple on-off controller can be implemented and tested in Simulink:



- (a) Set the transport delay to 1 second, the relay on/off points to ± 1 and the relay output to ± 2 . Run the simulation for a step input of 5.0. Observe the highly oscillatory response.
- (b) Double the step input size to 10 units to see the oscillations disappear.

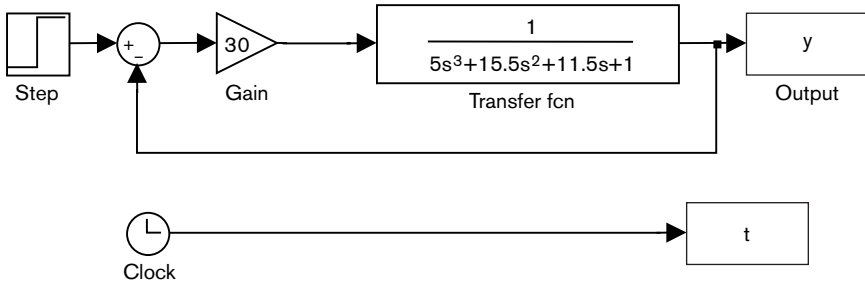
P4.2 Three-term control: demonstration of effect of integral action

Implement the following model in Simulink which shows a process transfer function with a controller which is formed by a gain and an integrator-plus-slider-gain.



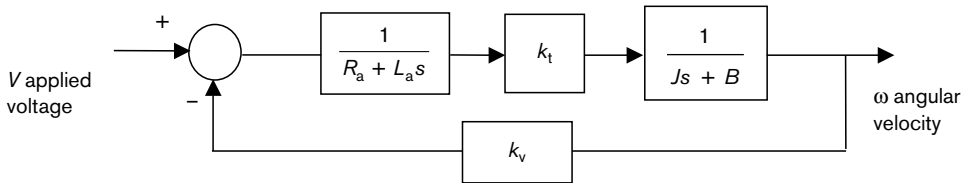
Run the simulation for values of integral gain from zero to 1.0 in steps of 0.1. Note the change in error between the reference signal and the output.

P4.3 The following model shows how a proportional controller can be tuned using Simulink.



- (a) Increase the gain from 30 in steps of 1 until the response becomes oscillatory. Call this gain the ultimate gain, K_U .
- (b) Set the value of proportional gain to be $K_p = 0.45K_U$.
- (c) Use this value in the simulation and investigate the closed-loop response.

P4.4 The block diagram of the angular velocity of a d.c. motor is:



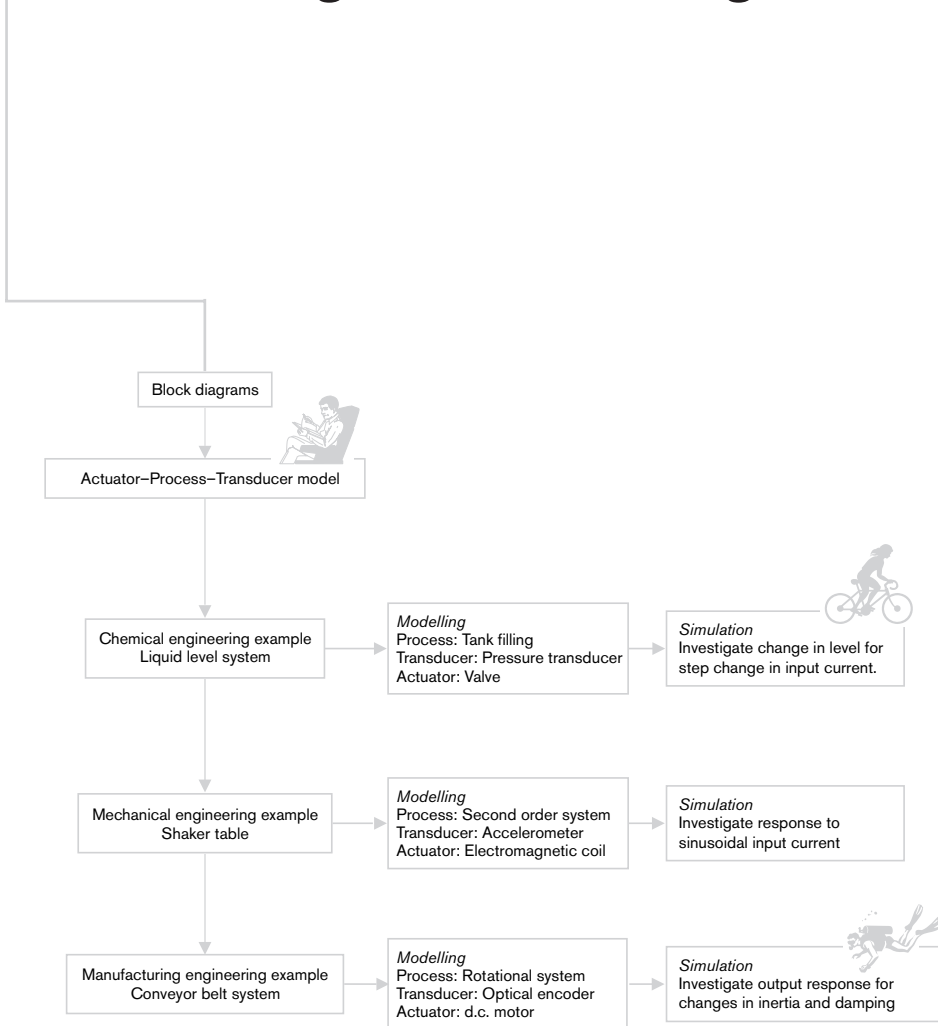
where

- R_a is the resistance of the motor armature (ohms) = 1.75
- L_a is the inductance of the motor armature (H) = 2.83×10^{-3}
- k_v is the velocity constant (V sec/rad) = 0.093
- k_t is the torque constant (Nm/A) = 0.0924
- J is the inertia seen by the motor (includes inertia of the load) (kg m^2) = 30×10^{-4}
- B is the mechanical damping coefficient associated with rotation (Nm s) = 5.0×10^{-3}

- (a) Implement the model in Simulink.
- (b) From a step response test, determine the time for the step response to increase from 10% to 90% of its final value.
- (c) The angular position is obtained by integrating the angular velocity. Implement this in the model.
- (d) Implement a closed-loop position controller assuming a proportional controller of gain K_p . (Hint:
 1. add a reference input signal for angular position
 2. add a summing block which calculates the error between the angular position reference and the measured angular position
 3. multiply this error by the gain K_p and let this signal be the applied voltage V .)
- (e) Find the value of K_p for the system to be critically damped, that is, when the system response is approximately at the point of reducing the overshoot value to zero.

5

Modelling for control engineering



Gaining confidence



Help?



Going deeper



Skill section



Time to read

Control engineering systems, whether containing chemical, mechanical or electrical sub-processes, are of varying sizes and complexity. We would like to use a standard schematic form to represent all the different systems. We will then be able to examine common features and develop a general framework and understanding of control systems.

Learning objectives

- To identify control inputs and outputs of subsystems.
- To construct simple control block diagrams.
- To recognise the actuator, process and measurement subsystems and create an actuator–process–transducer block diagram.
- To develop simple system models from mathematical equations or technical information.
- To form Laplace transform representations from simple system models.
- To implement simple Laplace transform representations in Simulink.

5.1 Signals, systems and block diagrams

The common form of representation in control systems is to use *block diagrams*; the *system* is represented by a box or block and the signals of temperature, voltage, flow etc. are represented by directed lines. Figure 5.1 shows a simple block diagram representation of a hot water heating system. The inlet water temperature *causes* the temperature of the radiator to rise which *causes* convection currents of warm air which in turn *cause* the temperature of the room to change.

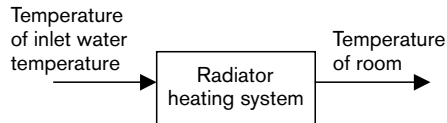


Figure 5.1 Hot water heating system.

Note that the lines do not actually represent hard-wired connections; they represent the flow of signals in the system diagram. We can define the block diagram structure more formally as follows.

5.1.1 Single-input single-output system block

We define a system in a general sense as a *cause and effect* relationship between one or more input signals and one or more output signals. A simple system structure is the single-input single-output (SISO) system. This simple system is described as the operation which transforms one input signal to produce one output signal, as shown in Figure 5.2.

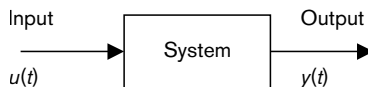


Figure 5.2 Single-input single-output block diagram.

This block diagram represents a cause and effect relationship between the input signal, $u(t)$, and the output signal, $y(t)$. It is usual in our control engineering field to find that the input $u(t)$ is a control input; that is, a signal that can be altered or *manipulated* to achieve a desired system behaviour. The output signal $y(t)$ in this context is a variable that can be measured and is often a system variable to be controlled in some desired manner.

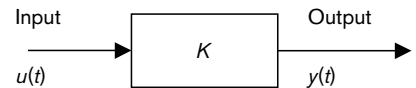
Very often the system blocks may represent quite complex processes. The system may have more than one input and more than one output, in which case it is referred to as a *multi-input multi-output* (MIMO) system. We can also then define SIMO and MISO systems as single-input multi-output and multi-input single-output systems, respectively.

We can study system representation by examining some simple general block diagram representations. We start by looking at the Constant gain block.

The Constant gain block

One simple example of a SISO system is a constant gain multiplier which we represent as a SISO block diagram.

Input signal: $u(t)$
 Output signal: $y(t)$
 Multiplier operation: $y(t) = Ku(t)$



It is important to remember that the gain, K , has physical units:

$$K\text{'s units} = \frac{y\text{'s units}}{u\text{'s units}}; [K] = \frac{[y]}{[u]}$$

Remark

Note that there is often an unusual mix of units associated with the constant gain multiplier. This is a common phenomenon which will be met in control systems work when defining input–output quantities for a system. For example, if there was a voltage-controlled heater causing the room temperature to rise, the units of the process gain would be °C/volt.

Processes containing more than one system block

Industrial processes may contain many subsystems, each of which could be represented by a SISO block. We must learn how to manipulate the signals and systems within our existing block diagram framework to enable us to deal with multiple blocks and many signals. Figure 5.3 shows a diagram where K_1 and K_2 represent two processes; the first may represent a control valve and the second may represent a heating system. The valve controls the flow of liquid which physically connects the two systems. Let signal $u(t)$ be

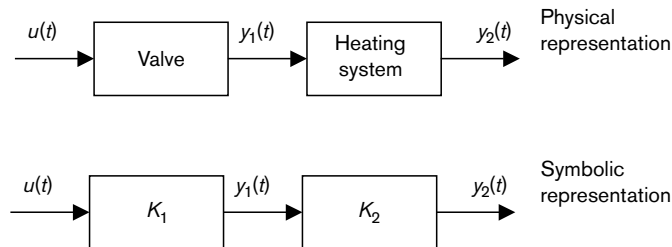


Figure 5.3 Cascaded systems.

the input to the first process, $y_1(t)$ the output from the first process (and the input to the second process) and signal $y_2(t)$ be the output of the total process. We refer to these systems as *cascaded* systems, or that K_1 is *in cascade* with K_2 .

We often wish to combine blocks to form a simpler diagram. For the simple linear blocks that we will be mainly dealing with in this text, we can evaluate the output of the system as follows. We often start at the output signals and work back to the input signals:

$$\text{For the second process: } y_2(t) = K_2 y_1(t)$$

$$\text{For the first process: } y_1(t) = K_1 u(t)$$

$$\text{Combining both: } y_2(t) = K_2 K_1 u(t)$$

or

$$y_2(t) = Gu(t) \text{ where } G = K_2 K_1$$

Hence the system can be re-drawn as Figure 5.4.

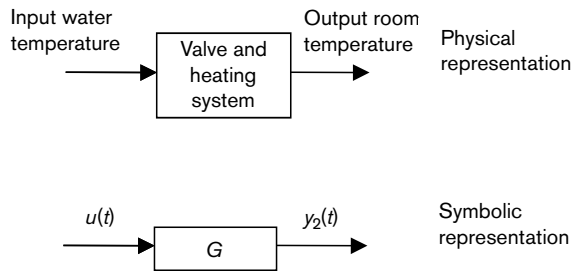


Figure 5.4 Combined block diagram.

5.1.2 Adding and subtracting signals: the summation symbol

Some processes, such as the mixing of two fluid streams, or the effect of a disturbance in temperature to a heating system, require the addition and subtraction of signals. These operations may be described as follows:

(i) Addition operation

Example:

Input signals: $x(t)$ and $y(t)$

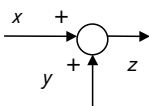
Output signal: $z(t)$

Addition operation:

$$z(t) = x(t) + y(t)$$

Representation:

$$z = x + y$$



(ii) Subtraction operation

Example:

Input signals: $x(t)$ and $y(t)$

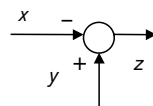
Output signal: $z(t)$

Subtraction operation:

$$z(t) = y(t) - x(t)$$

Representation:

$$z = y - x$$



Signals, x , y and z must be the same type and have the same units. For example, x and y might both represent velocity, but we cannot add them directly if one is measured in ms^{-1} and the other in miles/hr.

General usage

The use of the summation symbol to represent the addition or subtraction of signals of consistent type and units is fairly straightforward. The addition or subtraction operation is set up so that the summation has only *one* output or resulting signal. The symbol is then used as follows:

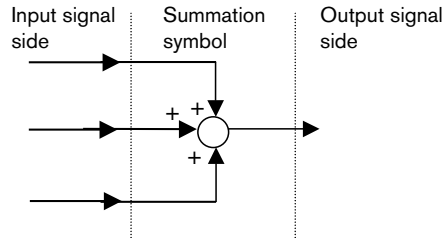


Figure 5.5 Summation sign.

We now introduce the structure for a typical process description: the Actuator–Process–Transducer diagram.

5.2 Actuator–Process–Transducer system structure

Many of the examples we have looked at can be generalised into the structure shown in Figure 5.6, with three basic elements: the actuator system, the process and the transducer.

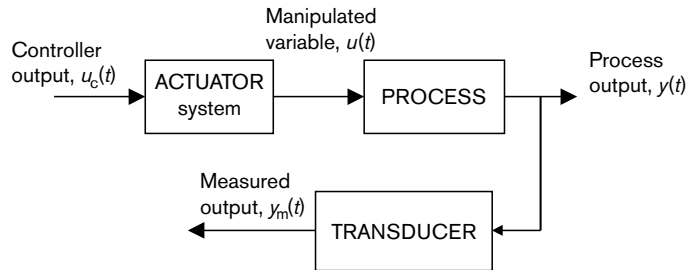


Figure 5.6 Actuator–Process–Transducer structure.

5.2.1 The process and the process model

The *process* or *plant* (from *process plant*) is represented by a mathematical model which shows how the input signal $u(t)$ affects the output $y(t)$. The input signal $u(t)$ is often called the *manipulated variable*, since it is this value that is changed ultimately by the control system to produce a corresponding output $y(t)$.

The mathematical model may be derived from differential equations relating the inputs and outputs of the process. These models can be tested against process data to

confirm their suitability. Alternatively, if experimental input–output data is available, the process model can be *identified* from this data.

5.2.2 The transducer

Different industrial processes require different measurements. If we consider the typical measurements in the food industry, we may need to measure temperature, moisture, flow, weight, and level, or even pressure, force, pH, humidity, density and colour. The measurements are taken for the following reasons.

1. Monitoring

Processes or sensor outputs are monitored for alarm/warning purposes, quality control aims, mandatory legal and technical requirements, or calibration of the measuring instruments themselves.

2. Control

By measuring a variable, we can find out how the process is behaving and take corrective action if necessary; the output signals are therefore monitored for control purposes.

Thus the output signals of any system may have to be monitored, displayed, recorded or used by the control system. All these considerations require that the measured output (measurand) of the system is as true a representation of the signal as is possible. The relationship is shown in Figure 5.7.

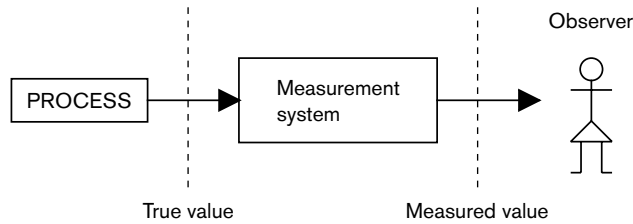


Figure 5.7 True and measured value.

The device used to sense any changes in the output is called a *sensor*. The output of the sensor is usually linked to other, often electrical or electronic, equipment to provide a signal which can be used for control or monitoring purposes. Sometimes, additional filtering or processing of the measured signal is needed (Figure 5.8). The sensor and associated electrical interface are called the *transducer*.

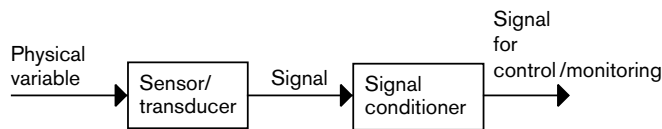


Figure 5.8 The transducer block diagram.

The transducing equipment itself may cause errors and therefore the measured value and the true value will not be exactly the same. These errors may occur by digitally sampling a signal or from environmental effects on the transducer such as temperature

changes having an effect on electrical resistance. It may also take additional time for the transducer to sample the process and produce an output. It is therefore useful to model the transducer as a separate system block.

Very often the transducer can be modelled by a simple gain block. This is because the time taken for the measurement to take place is often far smaller than the time periods involved in the actual process. By this we mean that although a temperature transducer may take 0.5 seconds to respond, this can often be neglected if the process is slow; for example, of the order of hours. Transducers that sense and transmit a change in a physical variable, such as pressure or temperature, can often be scaled to reflect the range of the signal being monitored. This will alter the gain of the transducer as follows.

Transducer span

The *span* of a signal is the range of that signal. Although transducers may have fixed maximum output ranges such as 0–10 bar or 4–20 mA, very often the transducer gain is scaled to accommodate the range of the actual measurement signal. The gain K can then be calculated as

$$K = 100\%/\text{span}$$

where ‘span’ is the range of the input signal. This equation states that the output signal will change by 100% for the full span of the input signal. For example, if a transducer were calibrated for a range of 0–5 bar, then the transducer gain would be 20%/bar. The transducer would then pass through 20% of its output range for one bar change in measurement signal.

5.2.3 The actuator system

The actuator system can also be subdivided into two components: the *actuating* equipment and the *final controlling element* (Figure 5.9(a)).

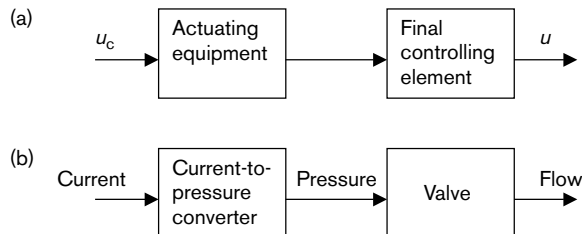


Figure 5.9 Actuator structure.

From the control input, $u_c(t)$, the actuating equipment produces a change in the final controlling element. The signal manipulated by the final controlling element is the input signal, $u(t)$, to the process. For example, as shown in Figure 5.9(b), a current to pressure converter attached to a pneumatic control valve will firstly convert the input control signal in mA to an equivalent pressure, which will ultimately open or close a valve, altering the flow of liquid or gas to the process. The valve is the final controlling element in this particular example.

We can see that the actuator system has two purposes:

1. to translate the output of the controller signal into one that can be applied to the piece of equipment, such as a valve or a heating implement

2. to apply this signal to the piece of equipment that directly changes the manipulated variable, $u(t)$.

Example: An actuator system for flight control

Actuator system:

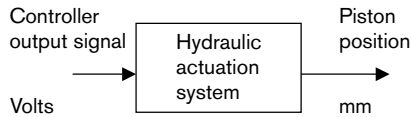
The hydraulic system which changes the angle of the control surfaces

Input to actuator system:

Output signal (in volts) from the controller

Output from actuator system:

A hydraulic ram or piston position (in mm) that causes a change in control surface angle



5.2.4 Complete block diagram representation for the Actuator–Process–Transducer system

Figure 5.10 represents an Actuator–Process–Transducer system using standard control system notation:

Actuator system:	system block G_a
Process:	system block G
Transducer:	system block H
Control signal:	$u_c(t)$
Manipulated signal:	$u(t)$
True output signal:	$y(t)$
Measured output signal:	$y_m(t)$

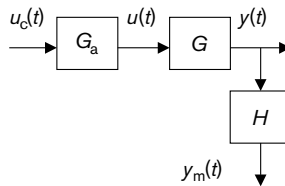


Figure 5.10 Actuator–Process–Transducer system.

We shall continue the analysis of systems and their signals by examining examples from different engineering disciplines. Before we proceed with the examples, it is useful to provide a summary of what we have covered.

5.3 Modelling summary

For any system, whether it be mechanical, electrical or chemically based, we will follow the same procedure: we need to determine:

1. the process model
2. the description of the transducer
3. the description of the actuator system

Block diagram description

For each component of the system (process, measurement, actuator) we need to:

1. draw the block diagram representing the sub-system
2. determine what the *input* and *output* signals for each component block are

Laplace transform models

To form a Laplace transform model, we need to remind ourselves of the following.

Laplace transform rules:

Transform of a derivative:
$$\mathcal{L}\left(\frac{dy}{dt}\right) = sY(s) - y(0)$$

Transform of a combination of signals: $\mathcal{L}(ax(t) + by(t)) = aX(s) + bY(s)$

We now examine three case studies, one based on chemical process engineering, one based on a mechanical system and one describing part of a manufacturing process. We note, however that most control systems cannot be modelled solely by referring to one discipline; control systems contain many components from electrical transducers to mechanical valves to chemical reactors, and the control engineer has to have a good understanding of the actual system.

5.4 Chemical process engineering: liquid level control

Fluid flow systems are very common in process control. These types of system involve components such as

- Process: mixing or blending processes, liquid-holding tanks
- Actuators: valves, pumps
- Measured values: level, flow, composition

Figure 5.11 shows a tank containing liquid. The flow of liquid into the tank is controlled by a valve. The control input signal to the valve is a current signal in mA which is converted into a pressure signal. This pressure is applied to a valve and changes the valve stem position (in mm). The valve position dictates the amount of flow passing through the valve into the tank. The height of liquid in the tank is measured by a transducer (gauge pressure) which produces an output in mA. The parameters of the system are given in Table 5.1.

- **Process:** The *process* is the change in level of the liquid in the tank. The *input* signal is the flow into the tank, $q_i(t)$ (in m^3/s). The *output* signal is the height of liquid in the tank. Combining these give the process block diagram of Figure 5.12.

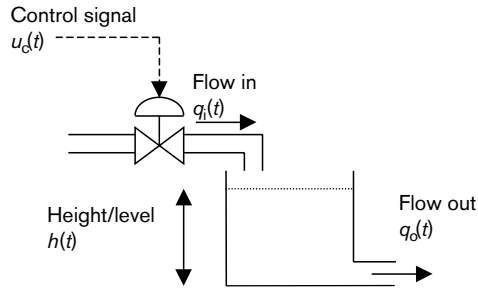


Figure 5.11 Liquid level process.

Table 5.1 Parameters of liquid level system.

Physical and design parameters	
Height of tank	4 m
Maximum fill level	3 m
Radius of tank	2 m
Tank capacity (volume)	37.68 m ³
Process valve position	0–25 mm
Pressure input signal	0–6 bar
Exit pipe restrictance parameter:	140 s/m ²

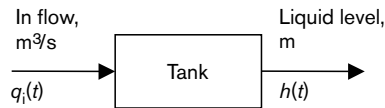


Figure 5.12 Process block diagram.

- Transducer:** We are interested in measuring the height of liquid in the tank. The transducer does not measure level directly, but measures pressure. From physics, $pressure = density \times g \times head\ of\ liquid$, and therefore a value for the head, or level of liquid, can be calculated. The pressure measurement transducer converts the gauge pressure reading (pressure reading relative to atmospheric) to an equivalent electrical current signal (in mA). The transducer block diagram is shown in Figure 5.13.

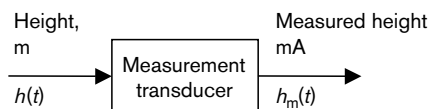


Figure 5.13 Measurement transducer.

- **Actuator:** The actuation system takes the control signal $u_c(t)$, a current in mA and applies this to a *current to pressure* transducer which in turn produces a valve position, (in mm). The position of the valve stem determines the flow (in m^3/s). The actuator block diagram is given in Figure 5.14

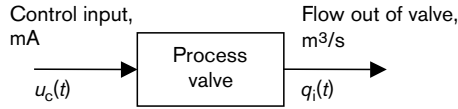


Figure 5.14 Actuator block.

By combining the Actuator–Process–Transducer block diagrams we find the total process block diagram can be represented by Figure 5.15.

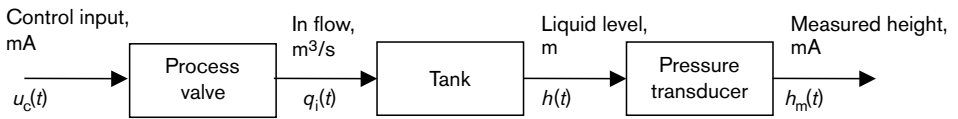


Figure 5.15 Combined block diagram for liquid level process.

Summarising the above:

- Process: a liquid level process
- Transducer: a gauge pressure transducer
- Actuator system: a process valve controlled by a current signal through a current to pressure transducer

We now need to introduce some models for the processes within our blocks.

5.4.1 Actuator–Process–Transducer: the Process block

Modelling of systems requires us to determine the relationship between input and output signals. In the case of the liquid level process (Figure 5.11), we must determine the relationship between the input flow and the output level. The liquid level system shows the liquid inflow as $q_i(t)$, and the outflow as $q_o(t)$. The height is given by $h(t)$ and the constant cross-sectional area of the tank by A . In control systems we often refer to the process input signal as $u(t)$ and the process output signal to be controlled as $y(t)$ (Figure 5.16). As we have only one manipulated input, $q_i(t)$, and one controlled output, $h(t)$, we classify this as a SISO system.

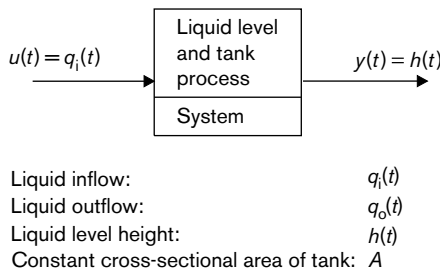


Figure 5.16 Block diagram of liquid level system.

We model the system using physical principles. The physical equation governing the change in liquid volume is:

$$\text{rate of change of volume of liquid} = \text{inflow} - \text{outflow}$$

Thus, if the inflow was equal to the outflow then there would be no change in the volume of liquid retained by the tank. We have, using the above physical principle:

$$\frac{d}{dt}(h(t)A) = q_i(t) - q_o(t)$$

We will assume

1. a constant cross-sectional area for the tank (A)
2. the outflow is proportional to the height of liquid: $q_o(t) = h(t)/R$, where R represents a parameter due to pipe restrictance.

Applying these assumptions to the differential equation for rate of change of volume gives:

$$A \frac{dh}{dt} = q_i(t) - q_o(t) = q_i(t) - \frac{h(t)}{R}$$

or

$$RA \frac{dh}{dt} + h(t) = Rq_i(t)$$

We have an equation with one first-order derivative, dh/dt ; hence the system is modelled by a first-order differential equation. Letting $AR = \tau$ and $K = R$, then we can write the equation in a standard form:

$$\tau \frac{dh}{dt} + h(t) = Kq_i(t)$$

This system equation has two parameters associated with it: K and τ . K is referred to as the system gain and τ is called the time constant for the system with units of time.

Using the information on the properties of the tank, we can work out the values of K and τ :

$$K = R = 140 \text{ s/m}^2$$

$$\tau = AR = \pi r^2 R = 12.6 \times 140 = 1758 \text{ s} = 29.3 \text{ min}$$

Key result: Laplace transformation of process equations

We can apply Laplace transforms to the first-order differential equation

$$\tau \frac{dh}{dt} + h(t) = Kq_i(t)$$

This gives

$$\mathcal{L}\left\{\tau \frac{dh}{dt} + h(t)\right\} = \mathcal{L}\{Kq_i(t)\}$$

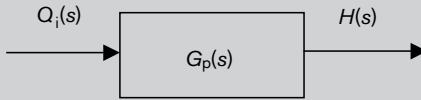
$$\tau \mathcal{L}\left\{\frac{dh}{dt}\right\} + \mathcal{L}\{h(t)\} = \mathcal{L}\{Kq_i(t)\}$$

$$\tau(sH(s) - h_0) + H(s) = KQ_i(s)$$

Assuming in this example that the tank level starts at zero ($h_0 = 0$) and rearranging gives

$$H(s) = \frac{K}{\tau s + 1} Q_i(s) = G_p(s) Q_i(s)$$

In block diagram format it can be written as:



where $G_p(s) = K/(\tau s + 1)$

5.4.2 Actuator–Process–Transducer: the Transducer block

Level transducers can take many forms:

- Measuring the differential pressure or gauge pressure (pressure relative to atmospheric) in a vessel will give an indication of level.
- Ultrasonic meters can detect depth by measuring the time taken for the ultrasonic beam to be deflected from the boundaries in the medium, for example from an air/liquid surface.
- Capacitive level transducers will change their output voltage depending on the change in overlap area of the capacitive plates, in the separation of the plates or in the dielectric material (here, liquid) between the plates.

We will discuss the pressure transducer in greater detail and provide the system block diagram for this form of level measurement.

Pressure gauge transducer

Figure 5.17 shows a tank containing a liquid of density ρ . The pressure, $p(t)$, at the measurement transducer is dependent on the head of liquid, $h(t)$, above the transducer. Since the transducer is conveniently placed at the base of the tank, the head will directly represent the level in the tank. The output of the electrical transducer is given by a current in mA, $h_m(t)$, which represents the measurement of liquid head.

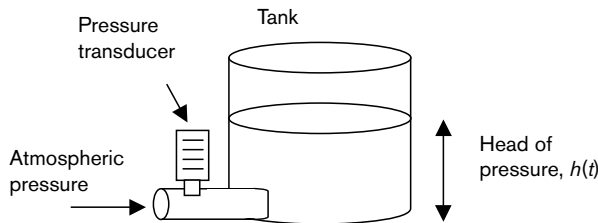


Figure 5.17 Gauge pressure transducer.

Transducer modelling

To model this transducer, we need to consider:

1. the relationship between the level in the tank and the pressure, and
2. the relationship between the pressure and the current signal

We can represent this measurement system by the block diagram of Figure 5.18.

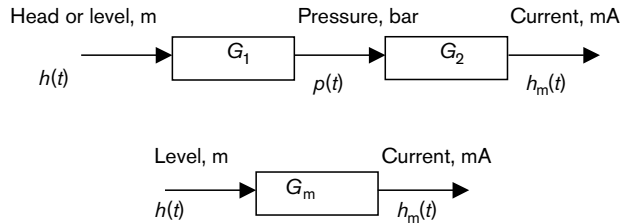


Figure 5.18 Transducer block diagram.

The values, G_1 and G_2 represent the relationship between the signals of level and pressure, or pressure and current, respectively. Since the component blocks are linear representations, the total transducer block is given by G_m , where $G_m = G_2 G_1$.

First Transducer block: G_1 (relationship between level and pressure)

Consider the physical law relating the input head of liquid to the output pressure:

$$\text{pressure} = \text{density} \times \text{gravitational constant} \times \text{head}$$

or, assuming the density remains constant,

$$p_p(t) = \rho g h(t)$$

where $p_p(t)$ is the pressure in pascals. For water, the value of ρ is approximately 1000 kg/m^3 . Hence, for a liquid level system with water as the liquid, the model required is:

$$p_p(t) = 1000 \times 9.81 h(t) = 9810 h(t)$$

The units of pressure are pascals; however, since $10^5 \text{ Pa} \approx 1 \text{ bar}$,

$$p(t) = 0.0981 h(t)$$

and the pressure $p(t)$ is in bar. Hence the first transducer block can be represented by Figure 5.19.

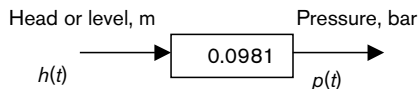


Figure 5.19 First Transducer block.

We can see that the value of G_1 has been given by $G_1 = 0.0981$. We can determine the dimensional units of G_1 by considering the units of the input and output signals:

$$[G_1] = \frac{[\text{units of output signal}] [p(t)]}{[\text{units of input signal}] [h(t)]} = \frac{\text{bar}}{\text{m}}$$

Therefore $G_1 = 0.0981 \text{ bar/m}$.

Table 5.2 Technical specification for pressure transducer.

Functional specification	
Output signal	4 to 20 mA
Span limit (max)	130 kPa (1.3 bar)
Range limit	-100 to 130 kPa (-1 to 1.3 bar)
Performance specifications	
Accuracy rating	Spans greater than 10% of FS ± 0.1% of span
Temperature effect	Zero shift: ± 0.5% Total effect: ± 1%
Step response	Time constant 0.2 s Dead time ≈ 0.3 s

Second Transducer block: G_2 (relationship between pressure and current)

To calculate the value of G_2 we consider the range (span) of the input and output signals. This is given by the technical specification for the pressure transducer (Table 5.2). The technical specification provides some functional information on the type of signal and ranges of signals, while the performance specification provides an indication for an instrument engineer of the accuracy of the readings and any effects that temperature will have on the output.

The functional specification confirms that the output signal range is 4–20 mA (the industrial standard current range) and tells us that, for this transducer, we can have a maximum input range of 1.3 bar (which would correspond to approximately $h(t) = p(t)/\rho g = 13.3$ m in our example). However, the output span on the instrument can be set to be between -1 and 1.3 bar. (The negative allows for systems which may require readings below atmospheric pressure.) For a maximum head in our system of 3 m (Table 5.1), we can set the lower and upper limits of the instrument to be:

- lower limit: 0 bar 4 mA
- upper limit: (1.3/13.3) × 3 = 0.294 bar 20 mA

To calculate the value of gain, G_2 , we look at the table of input and output signals:

Control block inputs and outputs	Physical variable	Range	Physical units
System input, $u(t) = p(t)$	pressure	0–0.294	bar
System output, $y(t) = h_m(t)$	current	4–20	mA

The measurement gain, G_2 , is found by considering the ratio of the change in output signal to the change in input signal:

$$G_2 = \frac{y(t)}{u(t)} = \frac{i(t)}{p(t)} = \frac{20-4}{0.294-0} \frac{\text{mA}}{\text{bar}} = 54.4 \frac{\text{mA}}{\text{bar}}$$

The two component blocks of the transducer can be cascaded together, as shown in Figure 5.20.

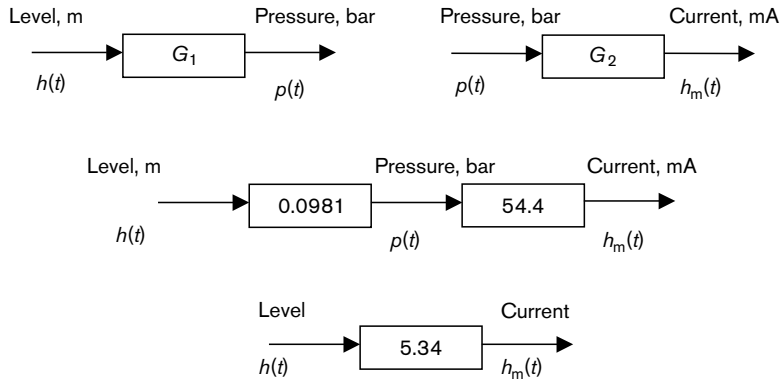


Figure 5.20 Cascaded transducer block diagram.

The equations for the transducer can be written as

$$h_m(t) = G_2 p(t) \quad p(t) = G_1 h(t)$$

Therefore

$$h_m(t) = G_2 G_1 h(t) = 54.4 \times 0.0981 h(t)$$

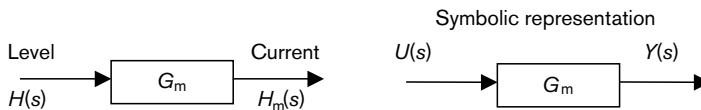
$$h_m(t) = G_m h(t) = 5.34 h(t)$$

and $G_m = 5.34$ with units of

$$[G_m] = \frac{[h_m(t)]}{[h(t)]} = (\text{mA})/\text{m}$$

Laplace transformation of measurement system

In Laplace transforms, the block diagram becomes:



The Laplace equations become:

$$Y(s) = G_m U(s) \quad \text{with } G_m = 5.34 \text{ mA/m}$$

5.4.3 Actuator–Process–Transducer: the Actuator block

The main actuators used in process systems are valves (to regulate flow) and pumps (to inject materials). In this example, we shall look at a diaphragm valve which controls the flow of liquid in to the tank in our liquid level system.

Figure 5.21 is a diagram of a process valve. The valve is inserted into the pipework such that liquid flows through the body of the valve. The size of opening that the liquid flows through is given by the position of the valve stem. This is controlled by changing the pressure on one side of the diaphragm which causes a change in the position of the plug. The pressure signal can be electrically actuated by means of a current-to-pressure transducer.

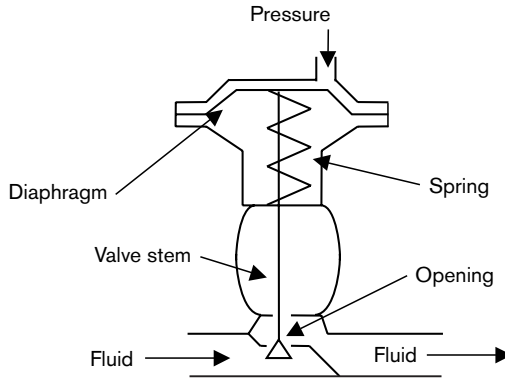


Figure 5.21 Process valve.

Actuator block diagram

The actuator can be represented by the cascaded block diagram of Figure 5.22.

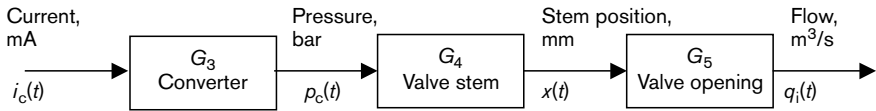


Figure 5.22 Actuator block diagram.

The values G_3 , G_4 and G_5 represent the relationship between the signals of current, pressure, valve stem position and flow, respectively.

First actuator block: G_3 (current-to-pressure converter)

A typical current-to-pressure specification may give the following information on the range of input and output signals:

Control block inputs and outputs	Physical variable	Range	Physical units
System input, $u(t) = i_c(t)$	current	4–20	mA
System output, $y(t) = p_c(t)$	pressure	0–6	bar

Hence the gain, G_3 , is given by:

$$G_3 = \frac{p_c(t)}{i_c(t)} = \frac{6-0}{20-4} = 0.375 \text{ mA/bar}$$

Second actuator block: G_4 (relationship between pressure and valve stem position)

We can relate the pressure on the diaphragm to the valve stem position by equating the forces on either side of the diaphragm:

$$\begin{aligned} \text{pressure} \times \text{diaphragm area} &= \text{spring stiffness constant} \times \text{change in stem position} \\ p_c(t)A_d &= K_s x(t) \end{aligned}$$

Hence

$$x(t) = \frac{A_d}{K_s} p_c(t)$$

Let the diameter of the diaphragm be 100 mm and the spring stiffness, K_s , be 188 400 kg/m; then we can calculate G_4 as

$$G_4 = \frac{\pi \times (0.05)^2}{188400} = 4.17 \times 10^{-8} \text{ m/N} \approx 4.17 \text{ mm/bar}$$

Third actuator block: G_5 (relationship between valve stem position and flow)
The flow, $f(t)$, that passes through a valve is given by:

$$f(t) = \alpha(t) C_v \sqrt{\frac{\Delta p(t)}{\rho}}$$

where

$\alpha(t)$ is the fractional opening of the valve

C_v is the flow coefficient of the valve (a number that relates to the design of the valve)

Δp is the pressure drop across the valve

ρ is the density of the liquid

For a linear valve, the value of α is the stem position, $x(t)$, of the valve. If we assume in this example that there is a constant pressure drop across the valve (which would mean that the flow had negligible effect on the stem position of the valve), then the flow through the valve is linearly related to the valve stem position. Hence the process valve model is given by

$$q_i(t) = G_5 x(t)$$

with the units of the gain, G_5 , as:

$$[G_5] = \frac{[y]}{[u]} = \frac{[\text{m}^3/\text{s}]}{[\text{mm}]}$$

For a valve that has a linear characteristic over its operating range, the value of the valve gain becomes

$$G_5 = \text{Rated flow}/100\% \text{ change in input}$$

For example, for a rated flow of $200 \text{ m}^3/\text{hr} = 0.056 \text{ m}^3/\text{s}$,

$$G_5 = 0.56 \times 10^{-3} \text{ m}^3/\text{s} \text{ for } 1\% \text{ change in input signal}$$

In our example, the stem position is not given in percentage change of input but in change in mm. By knowing the full range of input (say 0–25 mm), we can determine the valve gain for a change of 1 mm (= 4% of input range) in stem position:

$$\begin{aligned} G_5 &= 0.56 \times 10^{-3} \text{ m}^3/\text{s} \text{ for } 1\% \text{ change in input} \\ &= 2.24 \times 10^{-3} \text{ m}^3/\text{s} \text{ for } 1 \text{ mm } (4\%) \text{ change in input} \\ &= 0.00224 \text{ m}^3/\text{s}/\text{mm} \end{aligned}$$

Total actuator block diagram

The full actuator block diagram given in Figure 5.22 can be reduced to Figure 5.23.

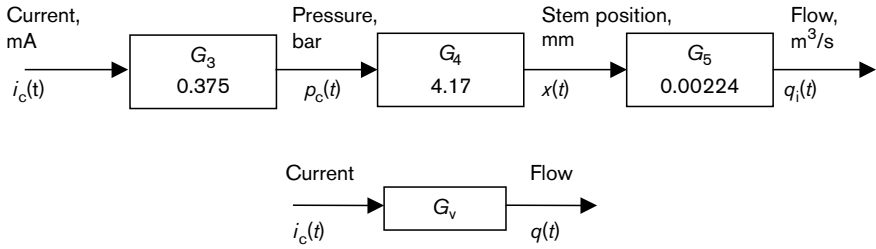


Figure 5.23 Actuator block diagram.

The equations for the actuator system can be written as:

$$q(t) = G_5 x(t) \quad x(t) = G_4 p_c(t) \quad p_c(t) = G_3 i_c(t)$$

Therefore

$$q(t) = G_5 G_4 G_3 i_c(t) = 0.0022 \times 4.17 \times 0.375 i_c(t) = 0.0034 i_c(t)$$

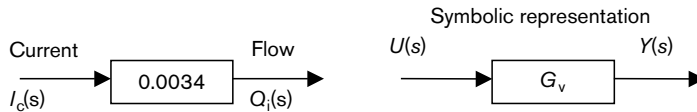
$$q(t) = G_v i_c(t)$$

and $G_v = 0.0034$ with units of

$$[G_v] = \frac{[q(t)]}{[i_c(t)]} = \text{m}^3/\text{s}/\text{mA}$$

Laplace transformation of actuator system

In Laplace transforms, the block diagram becomes:



The Laplace equations become:

$$Y(s) = G_v U(s) \quad \text{with } G_v = 0.0034 \text{ m}^3/\text{s}/\text{mA}$$

5.4.4 Complete Actuator–Process–Transducer block diagram

We can combine all our blocks to provide an overall model for the liquid level system (Figure 5.24).

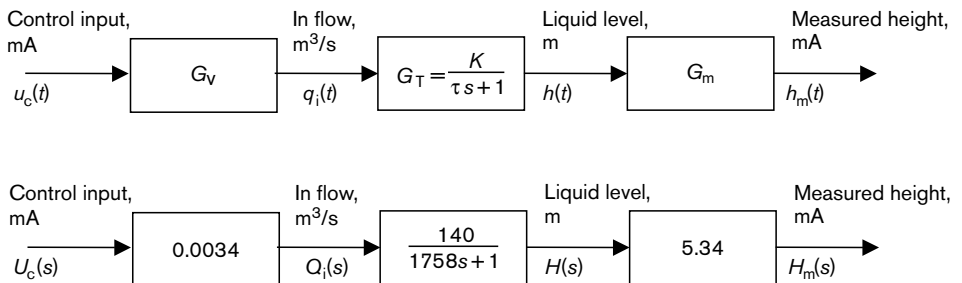


Figure 5.24 Actuator–Process–Transducer for liquid level system.

5.4.5 Simulink representation of liquid level system

The liquid level system shown in Figure 5.24 can now be implemented in a Simulink model (Figure 5.25). We use a first-order transfer function block to represent the first-order differential equation which represents the tank level process. The actuator and measurement blocks are, in this example, simple gain blocks which can be added to the model. The Simulink model requires an input signal block; we have used a step change signal for the input current, $u_c(t)$.

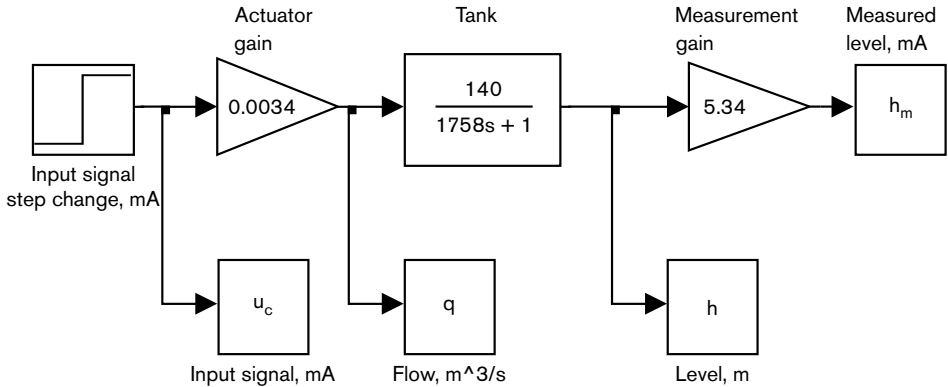


Figure 5.25 Simulink representation of liquid level system.

One of the advantages of using Simulink is that we can examine signals within the model which are not actually measured in practice. In this study, we have written variables $q(t)$, $h(t)$ and $h_m(t)$ to the workspace in order that we can examine the signals. We can inject a step change of 1 mA in the input signal and we can see (Figure 5.26) that the change in level in the tank is 0.476 m. The process input signal actually lies in the range

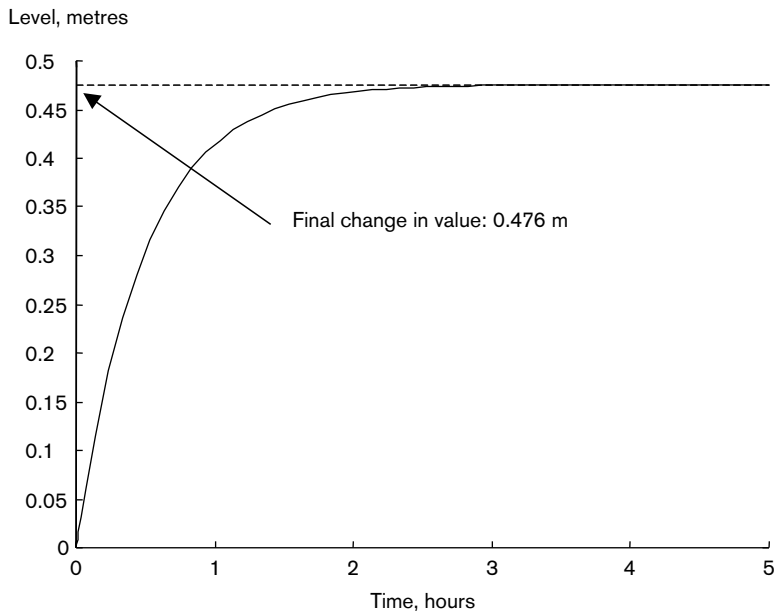


Figure 5.26 Response of liquid level to step change in input current of 1 mA.

4–20 mA. If we wished to monitor the actual level, we would need to add the offset to the signal. However, it is very common in control systems to examine the *change* in system behaviour from a steady operating condition. The reason for this is that we often assume that the system behaves in a linear manner for small movements around its steady operating condition – this allows us to use linear models and to use many analysis and control design techniques which are suitable for linear systems.

If we monitor the measured output, given by $h_m(t)$ in mA (Figure 5.27), we see in this example that the measured output signal has the same form of response as the actual level since there is only a constant gain difference between the two. Once again, it is important to note that the magnitude of the change in current is 2.54 mA maximum. This is the *change* in current which would register a change of 0.476 m. The actual current being measured lies in the range 4–20 mA, and therefore there would be an offset of 4 mA to be added if we were to read the value from a real process.

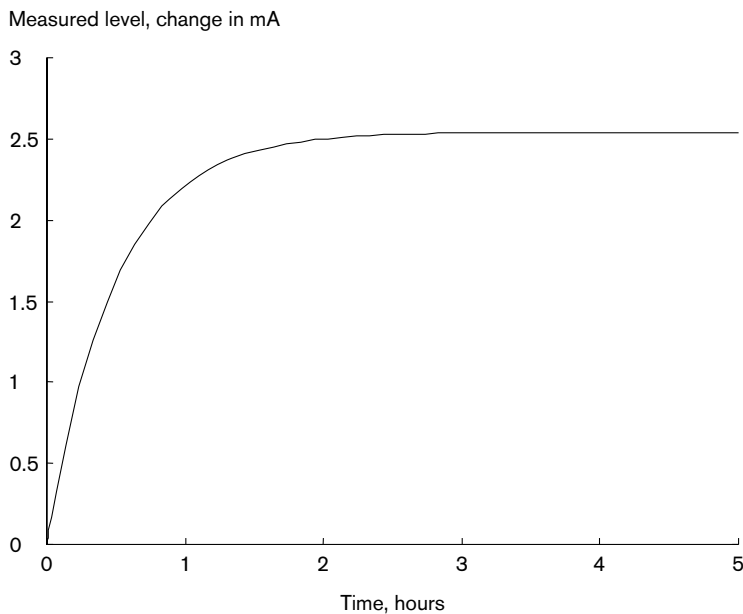


Figure 5.27 Measured level, change in mA.

5.5 Mechanical systems: model of a shaker table

Mechanical systems tend to be characterised by the movement (linear or rotational) of solid components. This movement may result in position, velocity or acceleration changes, or indeed in changes to the output force or torque on an object. In this example we will examine a *shaker table* system which can be set to vibrate and then used to calibrate measurement instruments or to test the resilience of manufactured products.

A shaker table system

Consider that an object, for example a camera, is placed on a shaker table and an input force applied through an electromagnetic coil (which has a current input). The effect of this force is to cause a vertical displacement and acceleration, $x(t)$ and $\ddot{x}(t)$, of the

combined shaker table and camera mass (Figure 5.28). The acceleration of the actual table is measured by an accelerometer, which provides an output signal in mV.

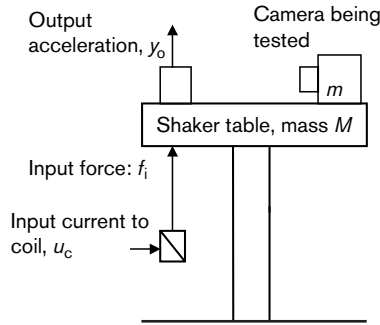


Figure 5.28 Shaker table with camera being tested.

First of all we shall determine the overall process diagram by examining the individual process, measurement and actuator blocks.

- **Process:** The process is represented by the dynamic behaviour of the combined shaker table and test object. The input signal is a force, $f_i(t)$ and the output we are interested in is the resultant acceleration of the shaker table (Figure 5.29).

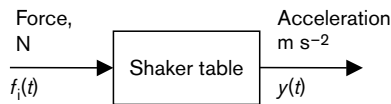


Figure 5.29 Process block.

- **Transducer:** The transducer is an accelerometer which produces a representation of acceleration in mV (Figure 5.30).

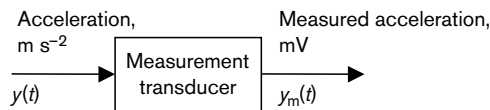


Figure 5.30 Measurement block.

- **Actuator:** The actuator which drives the shaker table is an electromagnetic coil. This provides an oscillating force on the table. The input to the coil is a control signal in amps (Figure 5.31).

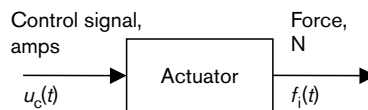


Figure 5.31 Actuator block.

The combined shaker table process is shown in Figure 5.32.

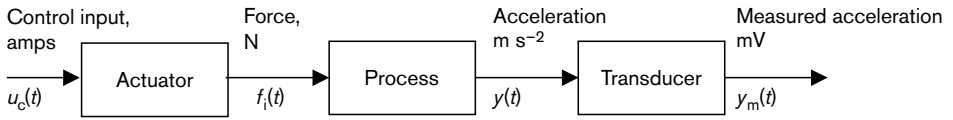


Figure 5.32 Shaker table block diagram.

We now take a closer look at the modelling of mechanical systems. Since mechanical process systems are often modelled as a combination of springs, masses and dampers, we will discuss these representations first.

5.5.1 Actuator–Process–Transducer: the Process block

We start by examining the representations for two mechanical elements: the spring element and the damper element.

Key result: A spring element

A spring element (Figure 5.33) behaves as a linear mechanical device if the spring restoring force $f_s(t)$ is related to the extension $x(t)$ by the law

$$f_s(t) = -K_s x(t) \quad K_s \text{ is called the spring constant.}$$

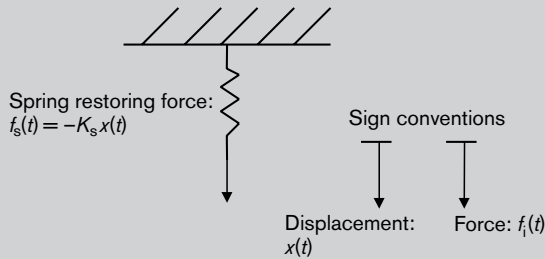


Figure 5.33 Spring element.

If a mass, m , is hung on the end of the spring, it provides an input force, $f_i(t)$. The system forces in balance would be:

$$f_i(t) + f_s(t) = 0 \quad \text{or} \quad mg - K_s x(t) = 0$$

A damper element

As the name implies, a damper is a mechanical element used to damp out mechanical motion. A very simple example is the common door closure damper where the restoring force is used to close a door smoothly and quietly. A common industrial example employs the damper principle to dampen out unwanted machinery vibrations.

The *damper*, or *dashpot* as it is sometimes called, is an oil-filled piston arrangement (Figure 5.34). The applied force $f_d(t)$ is resisted by a force caused by the oil flowing from one side of the piston to the other. The viscous oil will pass through orifices in the piston, or around the circumference of the piston.

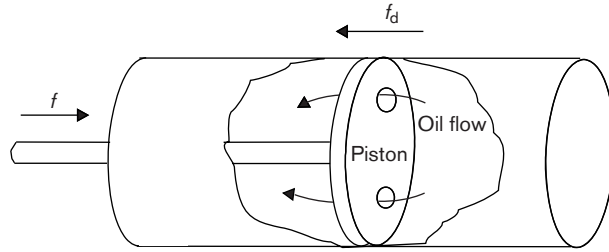


Figure 5.34 Damper element.

Key result: A damper element

The linear damper law is

$$f_d(t) = -B\dot{x}(t)$$

where $f_d(t)$ is the damper force, $\dot{x}(t)$ is the rate of change of displacement of $f_d(t)$ and B is the *viscous friction constant*.

Model of shaker table system

The spring and damper elements enable us to develop a simple model of the shaker table. The model usefully turns out to be that of a second-order system.

A camera, mass m , is placed on a shaker table and an input force applied through a solenoid valve (Figure 5.35). The input force is given by $f_i(t)$, and this is resisted by the spring force, $f_s(t)$, and the damper force $f_d(t)$. The net effect of these forces is a vertical displacement and acceleration, $x(t)$ and $\ddot{x}(t)$, of the combined shaker table and camera mass.

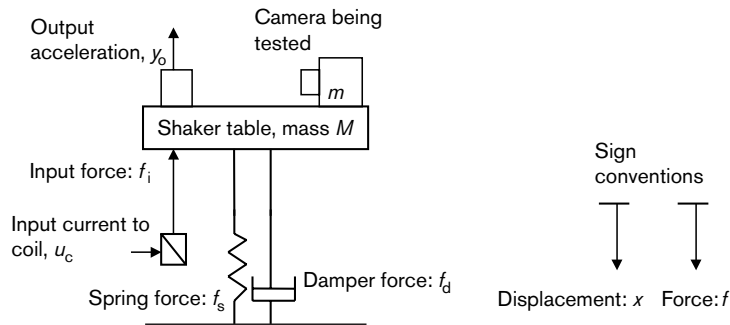


Figure 5.35 Shaker table and forces acting on it.

The physical principles are:

Input force from coil: $f_i(t)$

Spring force: $f_s(t) = -K_s x(t)$

Damper force: $f_d(t) = -B\dot{x}$

and these are all drawn together using Newton's second law of motion:

$$(M + m) \frac{d^2x}{dt^2} = f_i(t) + f_d(t) + f_s(t)$$

hence

$$(M + m) \frac{d^2x}{dt^2} - f_d(t) - f_s(t) = f_i(t)$$

giving

$$(M + m) \frac{d^2x}{dt^2} + B \frac{dx}{dt} + K_s x(t) = f_i(t)$$

Dividing through by the spring constant, K_s gives

$$\frac{(M + m)}{K_s} \frac{d^2x}{dt^2} + \frac{B}{K_s} \frac{dx}{dt} + x(t) = \frac{1}{K_s} f_i(t)$$

This is a second-order differential equation whose input is $f_i(t)$ and whose output is a resultant vertical displacement of the shaker table (Figure 5.36). By differentiating (twice) the displacement signal, $x(t)$, we can determine the acceleration of the table, $x_a(t)$.

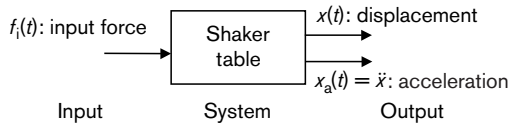


Figure 5.36 Shaker table modelling block.

We can calculate the parameters given the information on the system in Table 5.3.

Table 5.3 Physical parameters for shaker table system.

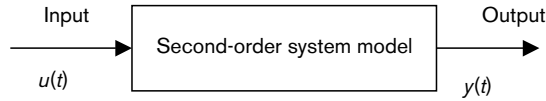
Mass of shaker table, M	2.5 kg
Mass of camera, m	250 g
Stiffness constant, K_s	1.1×10^7 N/m
System damping, B	1100 N/m s ⁻¹

If the maximum acceleration that we wish the table to experience is, for example, 1000m/s², then the maximum force (neglecting the mass of the accelerometer) is given by $F = ma = 2750$ N.

The second-order modelling equation can be written in a standard second-order form

$$\frac{1}{\omega_n^2} \frac{d^2y(t)}{dt^2} + \frac{2\zeta}{\omega_n} \frac{dy(t)}{dt} + y(t) = Ku(t)$$

where $u(t)$ represents the input, $y(t)$ the output and K , ζ and ω_n represent the model parameters. The general block diagram has the form:



The model parameters K , ζ and ω_n , are related to the system's physical parameters: such as mass, viscous friction constant and spring constant, or resistance, capacitance and inductance. We will soon discover that ω_n represents the natural frequency (undamped) of a system, ζ represents the damping *ratio* and K represents the system gain. In some textbooks we find ζ referred to colloquially as the damping *factor*. However, the damping factor is more correctly defined as the product $\zeta\omega_n$.

In our first-order model description (for the single tank level system) we found a general first-order model with two parameters, K and τ . For our second-order general model we have increased the number of parameters to three: K , ζ and ω_n .

Laplace transformation of process equations

Second-order equation:

$$\frac{(M+m)}{K_s} \frac{d^2x}{dt^2} + \frac{B}{K_s} \frac{dx}{dt} + x(t) = \frac{1}{K_s} f_i(t)$$

Assuming zero initial conditions for the position and velocity of the shaker table gives

$$\frac{(M+m)}{K_s} s^2 X(s) + \frac{B}{K_s} sX(s) + X(s) = \frac{1}{K_s} F_i(s)$$

which can be rearranged to give the transfer function between force and position:

$$\frac{X(s)}{F_i(s)} = \frac{1/K_s}{[(M+m)/K_s]s^2 + (B/K_s)s + 1}$$

The transfer function between force and acceleration is given by

$$\frac{X_A(s)}{F_i(s)} = \frac{s^2/K_s}{[(M+m)/K_s]s^2 + (B/K_s)s + 1}$$

Using the parameters for our system this becomes

$$\frac{X_A(s)}{F_i(s)} = \frac{9.1 \times 10^{-8} s^2}{2.5 \times 10^{-7} s^2 + 1.0 \times 10^{-4} s + 1}$$

5.5.2 Actuator–Process–Transducer: the Transducer block

The measurement device for the shaker table system is an accelerometer which provides a measurement, $x_m(t)$, of the acceleration signal, $x_a(t)$. The accelerometer is, in an ideal case, another example of a mechanical oscillating system (Figure 5.37). The measurement of the displacement is picked up by a piezoelectric device. This is a crystal lattice where the displacement, $z(t)$, of the atoms is proportional to the force, $F(t)$. The piezoelectric sensor itself has a second-order dynamic equation. However the damping ratio ζ is small and the natural frequency of the device chosen to be much higher than the frequency range we are interested in. Hence, in steady state,

$$F(t) = k_s z(t)$$

where:

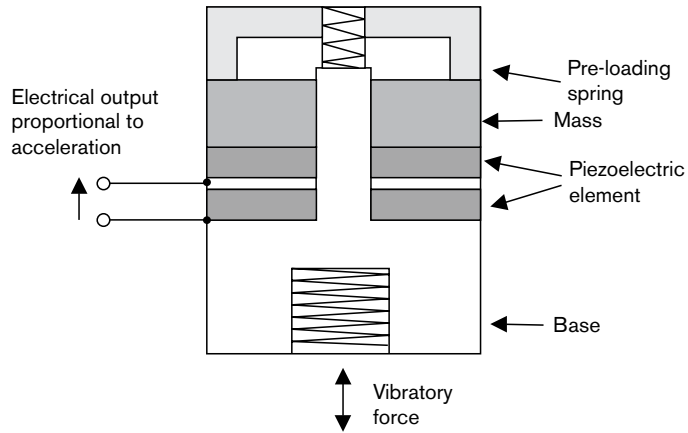


Figure 5.37 Compression type accelerometer.

$F(t)$ is the applied force

k_s is the stiffness of piezoelectric sensor (typical value $\sim 2 \times 10^9 \text{ Nm}^{-1}$)

$z(t)$ is the displacement due to applied force

The crystal acquires a net charge $q(t)$ which is proportional to $z(t)$:

$$q(t) = K_q z(t) = \frac{K_q}{k_s} z(t)$$

with K_q being the constant of proportionality. This gives a direct electrical output which can be measured by appropriate electrical circuitry.

As a general rule the accelerometer mass should be no more than one tenth of the dynamic mass of the vibrating part to which it is attached. A general purpose accelerometer will measure down to 0.01 m s^{-2} (limited due to electrical noise from connecting cables) and up to $50\,000$ to $100\,000 \text{ m s}^{-2}$. It also has a linear output within the working range of interest. A typical meter would also use amplifiers and filters within the measurement to reduce the component of error in the system.

A specification for our accelerometer is given in Table 5.4.

Table 5.4 Accelerometer specification.

Accelerometer specification	
Voltage range	0–1.7 volts
Max. acceleration	1000 m s^{-2}
Weight	50 g
Frequency range	0–12 000 Hz

A block diagram representation for the transducer is given in Figure 5.38.

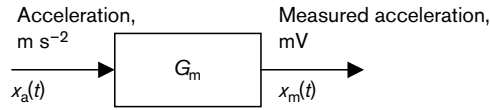


Figure 5.38 Transducer block diagram.

To calculate the value of the gain, G_m , we examine the input and output signals:

Control block inputs and outputs	Physical variable	Range	Physical units
System input, $u(t) = x_a(t)$	acceleration	0–1000	$m\ s^{-2}$
System output, $y(t) = x_m(t)$	current	0–1.7	V

The transducer gain, G_m (Figure 5.39), is found by considering the ratio of output to input signals.

$$G_m = \frac{y(t)}{u(t)} = \frac{1.7-0}{1000-0} \frac{V}{m\ s^{-2}} = 0.017\ V/m\ s^{-2}$$

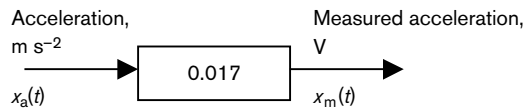


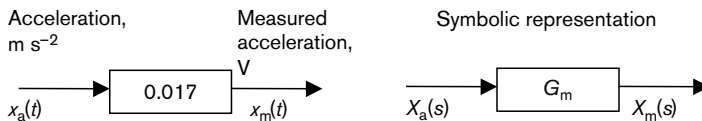
Figure 5.39 Transducer block diagram.

The equations can be written as

$$x_m(t) = G_m x(t)$$

Laplace transformation of measurement system

In Laplace transforms, the block diagram becomes:



The Laplace equations become:

$$X_m(s) = G_m X(s), \quad \text{with } G_m = 0.017$$

5.5.3 Actuator–Process–Transducer: the Actuator block

The actuator for the shaker table is an electromagnetic coil (Figure 5.40). Current is supplied to the coil, which is wound round the pole of a permanent magnet. The electromagnetic force induced causes the coil to move up and down. The end of the coil is attached to the shaker table to pass the force. By modulating the current to be, for example, $u_c(t) = A \sin \omega t$, the input force to the shaker table would be an equivalent oscillating force.

The block diagram for this system is shown in Figure 5.41. Although the electromagnetic coil is a complex system, we can represent it for control purposes here by a simple gain block, G_A .

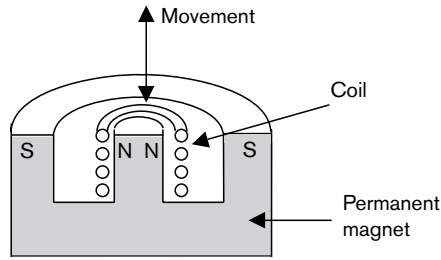


Figure 5.40 Electromagnetic coil.

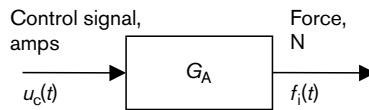


Figure 5.41 Actuator block diagram.

To find the value of G_A , we examine the input and output ranges for this actuator.

Control block inputs and outputs	Physical variable	Range	Physical units
System input, $u(t) = u_c(t)$	current	0–10	amps
System output, $y(t) = f_i(t)$	force	0–2750	N

Hence the gain G_A is given by:

$$G_A = \frac{2750-0}{10-0} \frac{\text{N}}{\text{amps}} = 275 \text{ N/amp}$$

The equation for the actuator system is

$$f_i(t) = 275u_c(t)$$

Laplace transformation of actuator system

Actuator system equation:

$$f_i(t) = 275u_c(t)$$

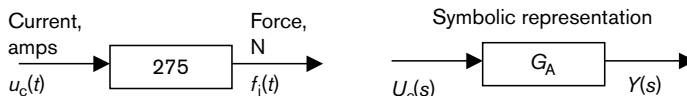
Actuator system equation in Laplace transforms:

$$F_i(s) = 275U_c(s)$$

or

$$F_i(s) = G_A U_c(s) \text{ with } G_A = 275$$

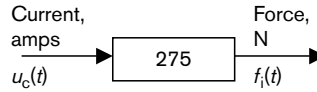
The block diagram becomes:



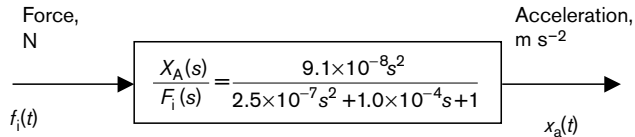
5.5.4 Complete Actuator–Process–Transducer block diagram

The individual actuator, process and transducer blocks are:

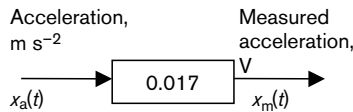
Actuator:



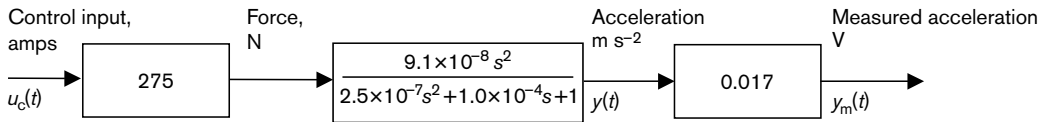
Process:



Transducer:



This gives the final model as



5.5.5 Simulink representation of shaker table system

The Simulink model (Figure 5.42) shows an input sinusoid being passed into the actuator gain block (which represents the electromagnetic coil). The force signal output from this block drives the shaker table (whose Laplace transform representation is given). The output acceleration signal of the shaker table is measured using the accelerometer to provide an output in volts. The advantage of using a Simulink model is that we have access to some of the signals which we do not measure on the real system, for example the force, $f_i(t)$ and the actual acceleration in $m s^{-2}$, $x_a(t)$.

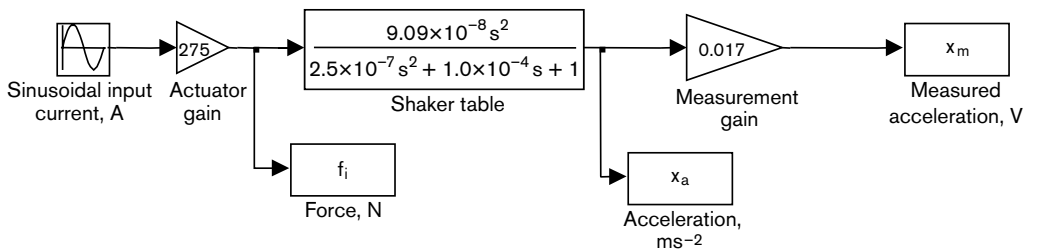


Figure 5.42 Simulink representation of shaker table system.

Alternative Simulink model

However, we can also model this system in Simulink in an alternative manner by using the differential equations that represent the process:

$$(M + m) \frac{d^2x}{dt^2} + B \frac{dx}{dt} + K_s x(t) = f_i(t)$$

or

$$\frac{d^2x}{dt^2} = \frac{1}{(M + m)} \left(f_i(t) - B \frac{dx}{dt} - K_s x(t) \right)$$

We remember that velocity is the integration of acceleration and displacement is the integration of velocity. Therefore, once we have determined the acceleration, we can integrate to find the velocity and position. An integrator block in Simulink is given by the Laplace representation: $1/s$. Figure 5.43 shows this Simulink representation. Once again we can use the features of Simulink to examine signal behaviour of signals that are not actually monitored in the real system.

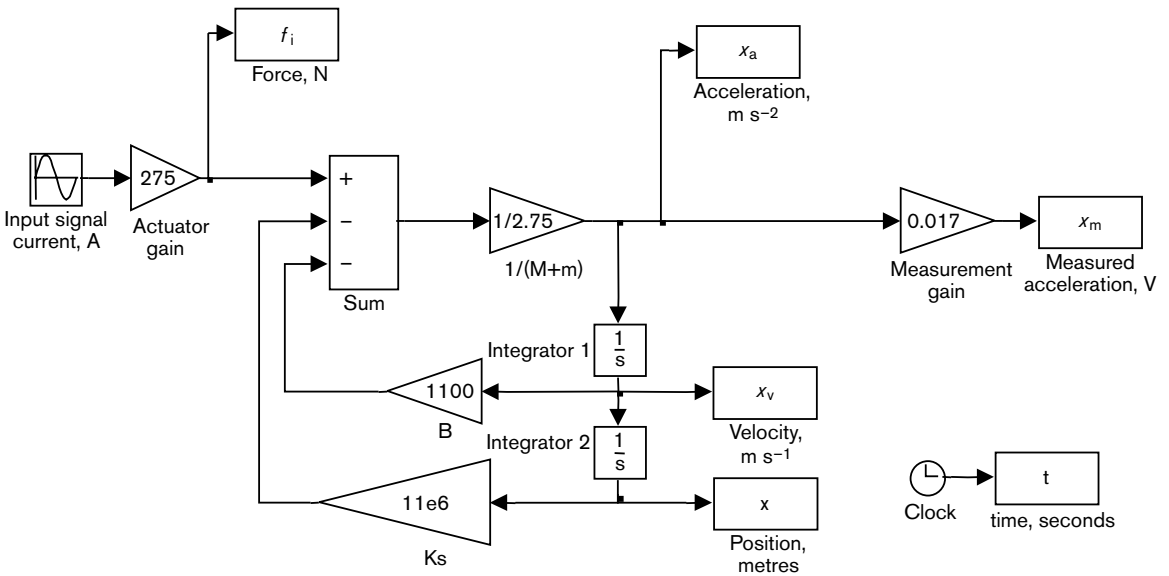


Figure 5.43 Alternative Simulink representation: Simulink representation from differential equations.

Simulation test: response to sinusoidal input

If we inject a sinusoidal input signal of 5 amps at 200 rad s^{-1} , we can use MATLAB to plot any of the signals we have written to the workspace. For example, let us look at the position of the shaker table, $x(t)$, in Figure 5.44. We notice that the output position is very small, less than $\pm 0.15 \text{ mm}$. This is correct, since the movement of the shaker table is not designed to be large. The table is designed to oscillate and test the object (camera) on the table at different frequencies to determine how it responds when stimulated near its resonant frequency.

This is a good example to show how you must still apply some intuition to the results output from a Simulink model. If we look at the acceleration and position together

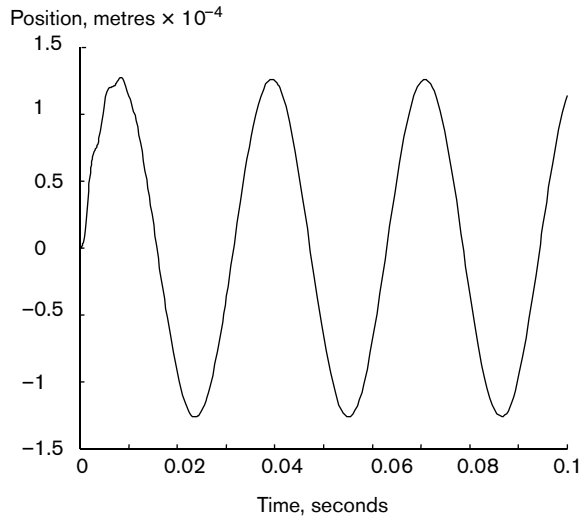


Figure 5.44 Position of shaker table.

(Figure 5.45: the acceleration has been scaled by a factor of 10^{-4} to enable them to be shown together), we see that the acceleration appears very oscillatory. This would not be expected, since if the displacement followed pure sinusoidal behaviour, the velocity and acceleration would do likewise:

$$y = \sin \omega t$$

The velocity and acceleration are given by

$$\frac{dy}{dt} = \omega \cos \omega t \quad \frac{d^2y}{dt^2} = -\omega^2 \sin \omega t$$

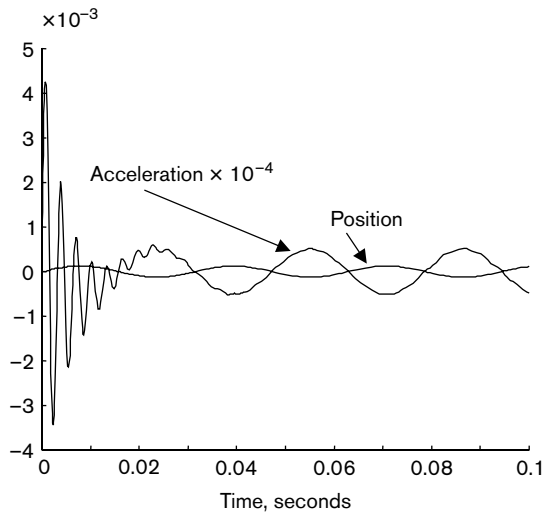


Figure 5.45 Acceleration and position of shaker table.

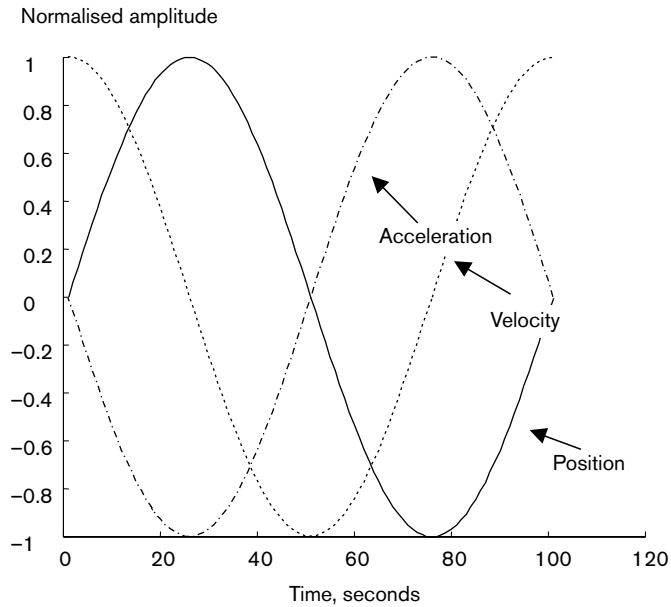


Figure 5.46 Position, velocity and acceleration.

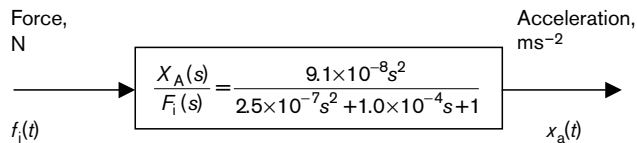
If we refer to Figure 5.46, we would expect the velocity to be zero when the shaker table is at its maximum position (and changing direction) and the acceleration to be at a maximum value but in the opposite direction. We do see this clearly in Figure 5.45 as time progresses. The initial large oscillations in Figure 5.45 are the natural transient response of the system to the input signal. The initial transient decays, leaving the underlying sinusoidal behaviour. We can verify this using MATLAB to determine the approximate *damped* frequency of oscillation of the system.

1. Measuring the damped frequency of oscillation from the plot

By using the function `ginput(2)` and clicking twice on two consecutive peaks on the MATLAB plot (Figure 5.45), we find that the approximate time for one cycle is $(0.0135 - 0.0103) = 0.0032$ seconds. The number of cycles/second (Hertz) is given by $1/0.0032 = 312.5$ Hz. Finally, the answer in rad s^{-1} is given as

$$\text{Damped frequency of oscillation: } 312.5 \times 2\pi = 1963 \text{ rad s}^{-1}.$$

2. Using Laplace transform analysis to find the roots of the denominator of the transfer function:



we enter the denominator polynomial as a vector of coefficients, `den`, and then request the roots.

```

den = [ 2.5e-7 1.0e-4 1];
p = roots (den)
ans =
    1.0e+003 *
    -0.2000 + 1.9900i
    -0.2000 - 1.9900i

```

This gives us two complex roots. The real part gives the rate of decay, $\zeta\omega_n$, and the imaginary part gives the value of the *damped* oscillation, ω_d . In this case $\omega_d = 1990 \text{ rad s}^{-1}$. Bearing in mind the precision with which we might have used the cursor on the MATLAB graph and the simulation step size, it does seem that the initial oscillations were due to the dynamics within the shaker system. With more detailed and accurate analysis, the two results for the damped oscillation could become closer.

5.6 Modelling of a manufacturing process component

Manufacturing systems often include conveyor belts, 'pick and place' systems and robot systems. Almost all will use some form of motor to drive, or actuate, the process. The measurements will vary depending on the process requirement, for example for accurate positioning in robotic welding and for accurate velocity in conveyor systems. As with our other process systems, the first important task is to be able to evaluate what the inputs and outputs of any particular system are and to be able to produce a process diagram.

Conveying system

Manufacturing systems use conveyor systems to transport goods from different areas of the production facility: from processing to filling to labelling to packaging and so on. A typical conveyor system (Figure 5.47) may use a belt on rollers to carry products. The rollers are driven by motors. The velocity of the system would have to be kept constant to optimise the process. The speed of the conveyor belt may be determined using a tachogenerator or an optical encoding system. The physical parameters are given in Table 5.5.

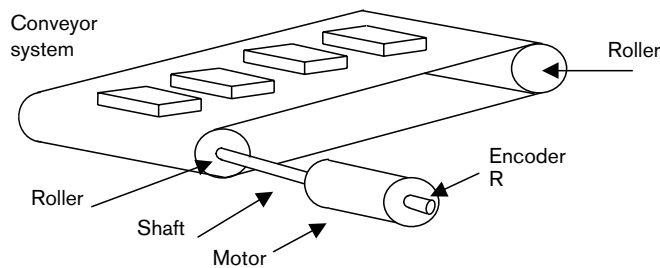


Figure 5.47 Conveying system.

We can represent this system in a control block diagram format. Firstly, determine the process, measurement and actuator components.

- Process:** In this example, the process is represented by the rotating shaft which moves the *load* (the conveyor belt and products). The *input* signal is the torque, $q(t)$, (in N m), from the motor which drives the load. The *output* signal is the rotational or angular velocity of the system (Figure 5.48).

Table 5.5 Conveyor parameters.

Physical and design parameters	
Combined inertias (relative to drive shaft), J	0.1 kg m ²
Friction factor, B	0.5 N m s ⁻¹
Roller diameter	200 mm
Max. belt speed	1 m s ⁻¹
Max. angular velocity	10 rad s ⁻¹
Maximum applied torque	10 N m

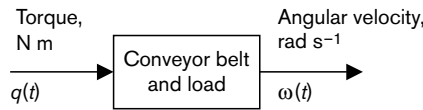


Figure 5.48 Conveyor Process block.

- Transducer:** We are interested in measuring the rotational velocity of the drive shafts to ensure accurate control of the conveying system. The measurement of the velocity of the rotating shaft is being performed by an optical encoder which produces a number of pulses per second, $\omega_m(t)$ (Figure 5.49).

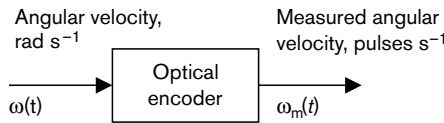


Figure 5.49 Conveyor Transducer block.

- Actuator:** The shaft is driven by a d.c. servo motor. The input to the motor is a control signal, $u_c(t)$, in volts. The motor provides an output torque, $q(t)$, which drives the system (Figure 5.50).

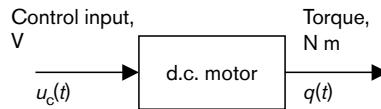


Figure 5.50 Conveyor Actuator block.

By combining the Actuator–Process–Transducer block diagrams we find that the total process block diagram can be represented by Figure 5.51.

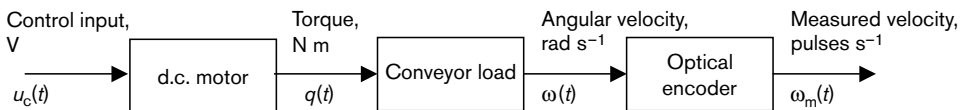


Figure 5.51 Model of conveyor belt system.

5.6.1 Actuator–Process–Transducer: the Process block

As with the mechanical system, which was modelled by a spring–mass–damper system, the modelling of a rotational system comprises three similar components: a torsional spring, a rotational inertia and a frictional torque effect. The total applied torque, T_A , on the shaft can be expressed as:

$$T_A(t) = T_s(t) + T_i(t) + T_f(t)$$

The three components, $T_s(t)$, $T_i(t)$ and $T_f(t)$ arise from the following:

- a torsional spring effect arising from a stiffness factor K_s which is associated with the mechanical property that the shaft twists through an angle $\theta_s(t)$ under applied rotational load or torque
- a torque arising from the sums of inertias, J , (referred to the motor shaft) and the related angular acceleration, $d^2\theta/dt^2$ or $\ddot{\theta}(t)$, of the shaft.
- frictional effects leading to a damping constant or friction constant, B , which is associated with the rotating load (angular velocity $d\theta/dt$ or $\dot{\theta}(t)$ or $\omega(t)$)

In this conveyor system, we will assume that

- (a) the torque produced by the motor passes directly to the load
- (b) the drive shaft is very stiff and therefore the value of $\theta_s(t)$ is negligible

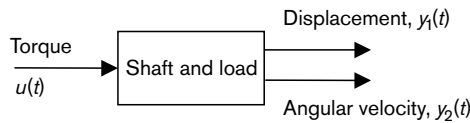
This results in an equation for the applied torque, $T_A(t)$:

$$T_A(t) = J \frac{d^2\theta(t)}{dt^2} + B \frac{d\theta(t)}{dt}$$

This is a second-order differential equation. We can use our standard control notation and define inputs and outputs of the system:

- input signal $u(t)$: applied torque, $T_A(t)$
- output signals $y_1(t)$, $y_2(t)$: position and velocity of shaft, $\theta(t)$ and $d\theta/dt$

The differential equation can be written in a control system block diagram:



Laplace transformation of process equation

Second-order differential equation: $T_A(t) = J\ddot{\theta}(t) + B\dot{\theta}(t)$.

Assuming that the initial conditions on position and velocity are zero ($\theta(0) = 0$, $\dot{\theta}(0) = 0$), this can be written in Laplace transforms as:

$$T_A(s) = Js^2\theta(s) + Bs\theta(s)$$

Hence the relationship between the input applied torque and the shaft position is given by

$$\frac{\theta(s)}{T_A(s)} = \frac{1}{s(Js + B)}$$

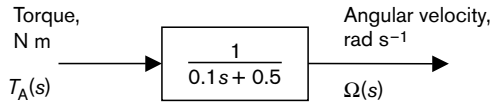
The velocity can be written as $\omega(t) = d\theta/dt$, and $\omega(s) = s\theta(s)$ in Laplace transforms. Hence the torque to velocity transfer function can be written as:

$$\frac{\Omega(s)}{T_A(s)} = \frac{1}{Js + B}$$

Using the information in the system properties table (Table 5.5), the transfer function can be written as

$$\frac{\Omega(s)}{T_A(s)} = \frac{1}{0.1s + 0.5}$$

The block diagram for the process is then given by



5.6.2 Actuator–Process–Transducer: the Transducer block, an optical encoder

Shaft encoders are often used in manufacturing and robotics to provide a measurement of angular movement, such as angular displacement or angular velocity. The encoder is mounted on the drive shaft and rotates at the same velocity as the shaft. A simple rotary encoder, which is used to measure angular velocity, is shown in Figure 5.52.

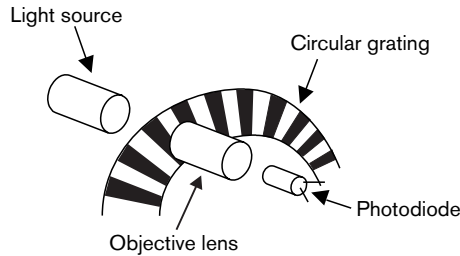


Figure 5.52 Rotary encoder.

A fixed source of light is provided on one side of the disc and a means of monitoring this light, such as a photodiode, is placed on the opposite side of the encoder disc. This disc is marked with a precise circular pattern of alternating clear and opaque segments. As the disc rotates, light falls on the photodiode in a distorted sinusoidal manner. The photodiode therefore produces a sinusoidal electrical output signal. By passing this through an electrical circuit, (a Schmitt trigger circuit), the signal can be converted into a square wave. This square wave can be converted electrically into a series of pulses (Figure 5.53) which will be directly proportional to the shaft velocity. Sometimes it is necessary to determine the direction of the shaft rotation, and in this case two photodiodes can be used, with an auxiliary grating system on the encoder.

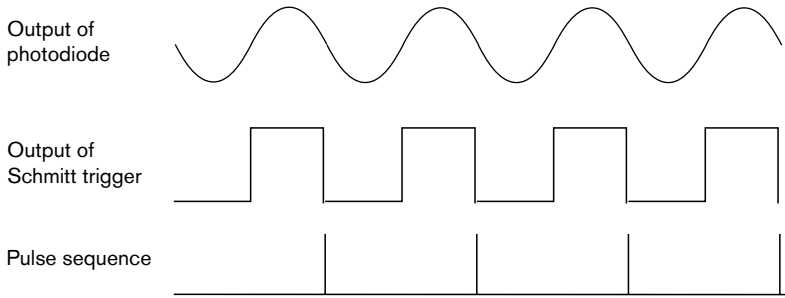
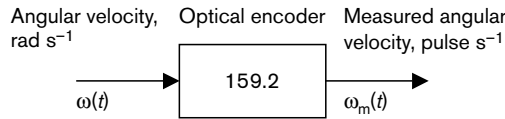


Figure 5.53 Output of encoder and electrical circuitry.

By examining the table of inputs and outputs for the system, we find that the measurement gain, G_m , relates the measured electrical pulses to the input angular velocity.

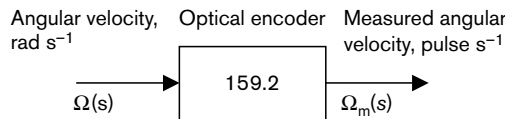
Control block inputs and outputs	Physical variable	Physical units
System input, $u(t) = w(t)$	angular velocity	rad s ⁻¹
System output, $y(t) = w_m(t)$	electrical pulses	pulse s ⁻¹

The value of the transducer gain, G_m , is found from the parameters of the encoder disc. The encoder disc in our example gives 1000 pulses/revolution, or equivalently, 1000 pulses/ 2π radians, which gives 159.2 pulse/rad. This value is the value of G_m . Therefore the transducer block diagram is given by:



Laplace transformation of transducer equations

In Laplace transforms, the transducer block diagram becomes:



The Laplace equations become:

$$\Omega_m(s) = G_m \Omega(s) \quad \text{with } G_m = 159.2 \text{ pulse/rad}$$

5.6.3 Actuator–Process–Transducer: the Actuator block, a field controlled d.c. motor

The operation of electric motors is based on the following principles:

1. If a current-carrying conductor is placed in a magnetic field, there will be a force exerted on the conductor.

2. If a conductor is moved through a magnetic field there will be a voltage induced in the conductor.

Electric motors can be classified according to their electrical configuration. This produces two main categories: d.c. motors and a.c. motors. The d.c. motor consists of a rotating cylinder called the *armature* which is placed in the magnetic field between two stationary magnetic poles, called the *field*. The a.c. motor consists of a rotating cylinder called the rotor which rotates in the rotating magnetic field produced by the windings of the stationary stator.

Traditionally, a.c. motors with no additional speed control were used primarily for operations that require a constant single operating speed. Single-phase motors are used for low-power applications, with three-phase motors being used for applications requiring greater power. D.c. motors have traditionally been used for variable-speed applications, since the control circuits for varying a.c. motor speed are more complex. However, with the application of modern electronic circuits to a.c. motor control, variable-speed a.c. drives have become more attractive.

Variable-speed d.c. motor

There are two methods of controlling the speed of a d.c. motor:

1. Field control: hold the armature current constant and control the torque via the field voltage.
2. Armature control: hold the field constant and control the torque via the armature voltage.

In this example (Figure 5.54), we control the field voltage $V_f(t)$ in order to control the speed and position of the output rotating shaft. The motor variables and parameters are given in Table 5.6.

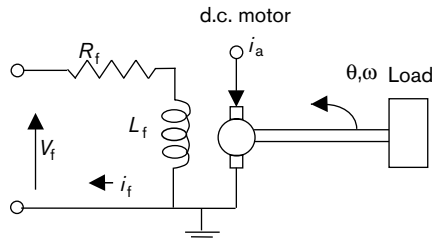


Figure 5.54 D.c. motor schematic.

Table 5.6 D.c. motor variables and parameters.

Motor variables	Motor parameters
$V_f(t)$: Field voltage	R_f : Field winding resistance = 2 Ω
$i_f(t)$: Field current	L_f : Field winding inductance = 0.5 H
$i_a(t)$: Armature current	

In a d.c. motor, the field provides the magnetic flux and the armature circuit is provided with a constant current $i_a(t)$ by a supply voltage, $V_a(t)$. By supplying the field windings with a varying voltage, the magnetic field varies and this provides, through the motor, the changing torque.

1. Field circuit

$$V_f(t) = R_f i_f(t) + L_f \frac{di_f(t)}{dt}$$

2. A mechanical effect in the form of an applied rotational torque, $T_A(t)$, where

$$T_A(t) = K_1 \phi i_a(t)$$

3. However, since the armature current is constant, the torque is proportional to the flux, and the magnetic flux satisfies

$$\phi(t) = K_2 i_f(t)$$

Combining these gives the torque as

$$T_A(t) = (K_1 K_2 I_a) i_f(t)$$

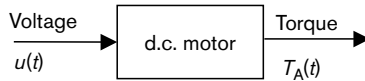
The maximum applied torque is 10 N m for a field current of 5 A, and therefore this gives the combined constants of proportionality, $K_1 K_2 I_a$, equal to 2 N m/A.

We can now express the d.c. motor as a system with appropriate input and outputs:

input signal, $u_c(t)$: field voltage, $V_f(t)$

output signal, $y(t)$: applied torque, $T_A(t)$

The control block diagram is:



Laplace transformation of d.c. motor equations

Field circuit:
$$V_f(t) = R_f i_f(t) + L_f \frac{di_f(t)}{dt}$$

Applied torque:
$$T_A(t) = (K_1 K_2 I_a) i_f(t)$$

Assuming zero initial conditions for the field current and voltage gives

$$V_f(s) = (R_f + sL_f)I_f(s)$$

and

$$T_A(s) = K_3 I_f(s) \quad \text{where} \quad K_3 = K_1 K_2 I_a$$

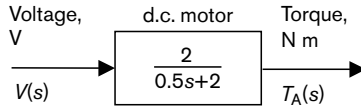
This gives the transfer function as:

$$\frac{T_A(s)}{V_f(s)} = \frac{K_3}{(R_f + sL_f)}$$

Given the system parameters in Figure 5.54, the transfer function can be written as:

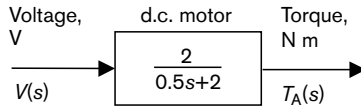
$$\frac{T_A(s)}{V_f(s)} = \frac{2}{(2+0.5s)}$$

and the Laplace block diagram representation is given by:

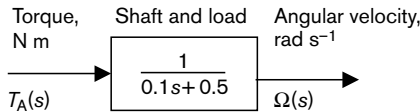


5.6.4 Complete Actuator–Process–Transducer block diagram

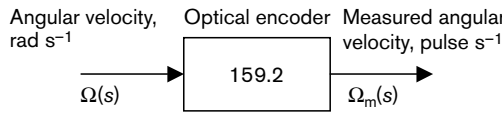
Actuator block diagram:



Process block diagram:



Transducer block diagram:



The combined block diagram for the conveyor belt system is given in Figure 5.55.

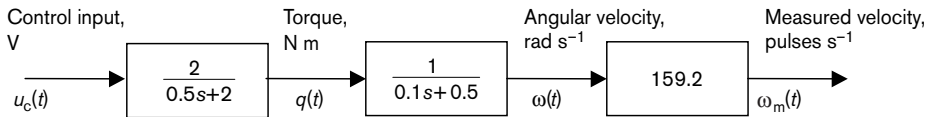


Figure 5.55 Final block diagram of conveyor belt system.

5.6.5 Simulink representation of conveyor belt system

This can be represented in a Simulink block diagram as shown in Figure 5.56. The motor and drive shaft and load are represented by two first-order systems (in Laplace transforms) and the measurement is given by a gain block. In Simulink (which does not use Greek letters and subscripts), the measured value is indicated by *wm* and we can also monitor the input voltage, *uc*, the input torque, *q* and the actual velocity, *w*. The input signal is a step change in voltage to the d.c. motor.

We can therefore examine how this system behaves for a change in input voltage or for changes of system parameters. Figure 5.57 shows three responses. In each case, the plot represents the change in angular velocity, $\omega(t)$, of the system for a step change of 1 V in input signal to the drive motor. However, in the second and third cases the nominal

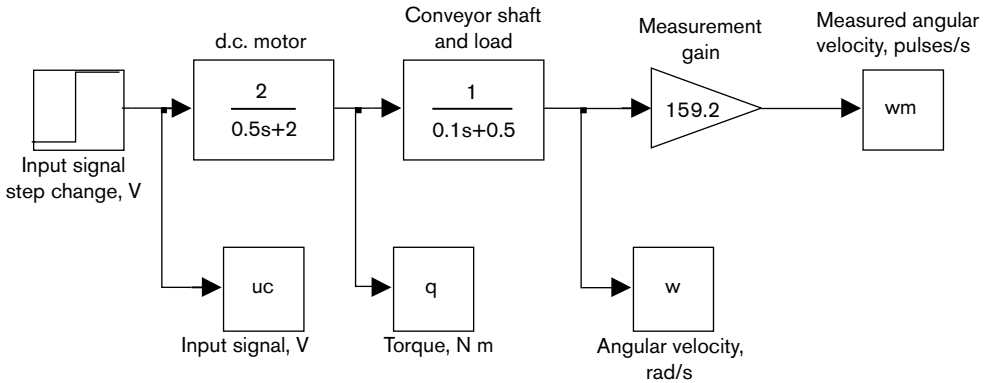


Figure 5.56 Simulink representation of conveyor system.

parameter values have been changed. By changing the inertia value J we see that the speed of response is slower, though the drive shaft does reach the same angular velocity eventually. By increasing the friction constant, B , we find that the steady state angular velocity attained is less than the original. Simulink has provided us with a means for determining the system behaviour for a range of parameter values. This could be useful when designing systems or experimenting with different system components.

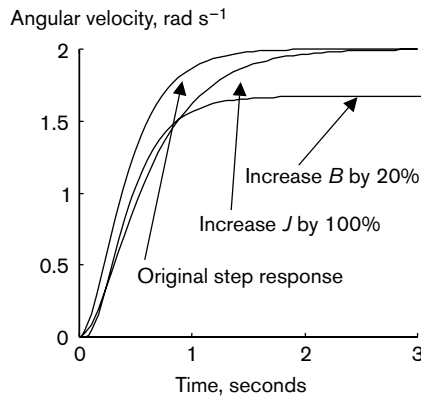


Figure 5.57 Angular velocity of shaft of conveying system.

Although engineering systems may often look complex, many can be broken into subsystems which can be approximated by simple linear block models. In this chapter we have examined three such systems from different disciplines, a liquid level system, a shaker table and a manufacturing conveying system. All the systems use a combination of knowledge from different areas: chemical, mechanical, manufacturing and electrical. We now summarise what we have covered in this chapter.

What we have learnt

- ✓ To recognise the control inputs and outputs of a system.
- ✓ To represent a process by a block diagram, with control inputs and outputs as directed lines.
- ✓ To model a simple gain block within a control system.
- ✓ To examine a control system and to recognise the Actuator–Process–Transducer units within it.
- ✓ To develop simple models for actuator, process and measurement units.
- ✓ To produce a Laplace transform representation of the complete system.
- ✓ To simulate the system using Simulink and provide a first simple analysis of results.

Multiple choice

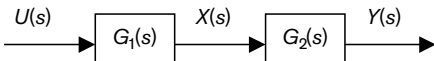
M5.1 In a control block diagram, signals are represented by

- (a) Boxes
- (b) Arrows
- (c) Directed lines
- (d) No symbol, just their letters

M5.2 The common control notations for the input and output signals of a process block are, respectively:

- (a) $p(t)$, $y(t)$
- (b) $u(t)$, $y(t)$
- (c) $u(t)$, $x(t)$
- (d) $x(t)$, $y(t)$

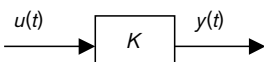
M5.3 If a system has the following representation:



which of the following is correct?

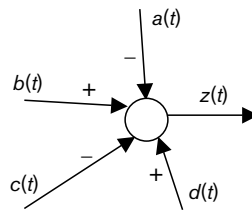
- (a) $U(s) = G_1(s)X(s)$
- (b) $Y(s) = X(s)U(s)$
- (c) $U(s) = G_1(s)G_2(s)Y(s)$
- (d) $Y(s) = G_2(s)G_1(s)U(s)$

M5.4 If $u(t)$ has units of °C and $y(t)$ has units of mm, what are the units of K in the following diagram?



- (a) °C
- (b) mm
- (c) mm/°C
- (d) °C/mm

M5.5 For the following diagram, which statement is correct?



- (a) $z(t) = a(t) + b(t) - c(t) + d(t)$
- (b) $z(t) = a(t) + b(t) + c(t) + d(t)$
- (c) $z(t) = -(a(t) + c(t)) + (b(t) + d(t))$
- (d) $z(t) = -a(t) - b(t) - c(t) + d(t)$

M5.6 Which statement describes the purpose of an actuator?

- (a) equipment which measures the controller signal
- (b) equipment which converts the controller signal into an action on the process
- (c) equipment whose sole purpose is to amplify the controller signal before being applied to the process
- (d) equipment used to measure the output of the process

M5.7 The Laplace transform of dy/dt is

- (a) $sY(s) - y_0$
- (b) $sY(s) + y_0$
- (c) $Y(s) - y_0$
- (d) $Y(s) + y_0$

- M5.8** The output of a measurement transducer:
- (a) can suffer from environmental effects
 - (b) will have the same units as the output of the process
 - (c) is always calculated using a simple gain block
 - (d) gives the exact value of the process output

- M5.9** A damper is an element:
- (a) that amplifies the output of a process
 - (b) used in the modelling of mechanical components
 - (c) used to stop the conveyor belt in a manufacturing system
 - (d) used in chemical process engineering

- M5.10** Modelling refers to the process of:
- (a) defining a set of mathematical equations for a process
 - (b) finding a block diagram representation for a system
 - (c) finding an input–output relationship for a system
 - (d) all of the above

Questions: practical skills

- Q5.1** For the following transducer descriptions, what are the values of the simple gain and the units of the gain?

- (a) a flow meter: for every litre of liquid that flows it records a change of 2 V
- (b) a pressure transducer: for every 0.75 bar change it records a change of 1 mA
- (c) a temperature transducer: for every +2 °C change in temperature, the output changes by 80 μ V

- Q5.2** What is the Laplace transform of the following differential equations?

- (a) $4 \frac{dy}{dt} + 3y(t) = 10q(t)$ $y(t)$ is output, $q(t)$ is input, $y(0) = 0$
- (b) $6 \frac{dm}{dt} + m(t) = 2p(t)$ $m(t)$ is output, $p(t)$ is input, $m(0) = 3$
- (c) $\tau \frac{dy}{dt} + y(t) = Ku(t)$ $y(t)$ is output, $u(t)$ is input, $y(0) = y_0$

- Q5.3** The following two differential equations represent second-order equations of motion in linear and rotational forms. What is the Laplace transform of the differential equations? Note the similarities in the general equations.

- (a) $M \frac{d^2y}{dt^2} + B \frac{dy}{dt} + K_s y(t) = Kf(t)$
 $y(t)$ is output position, $f(t)$ is input force, $y(0) = 0$, $\dot{y}(0) = 0$.
- (b) $J \frac{d^2\theta}{dt^2} + B \frac{d\theta}{dt} + K_s \theta(t) = KT(t)$
 $\theta(t)$ is output angular position, $T(t)$ is input torque, $\theta(0) = 0$, $\dot{\theta}(0) = 2 \text{ rad / s}$.

Problems

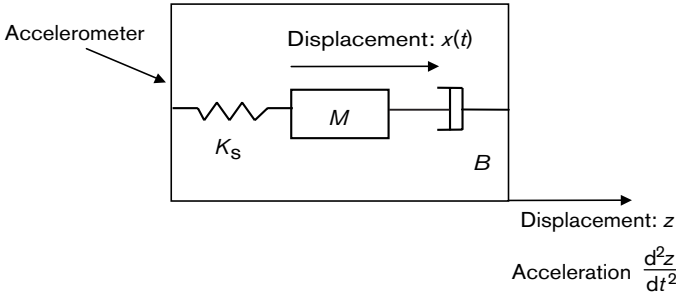
- P5.1** Given the following information about a system, find the complete block diagram description using Laplace transforms.

A tank system is represented by the following first order differential equation:

$$\tau \dot{h}(t) + h(t) = Kq(t)$$

where $h(t)$ is the liquid level in m and $q(t)$ is the input flow rate in m^3/s . K is given as 0.01 s/m^2 and $\tau = 15$ minutes. The output signal comes from a measurement transducer whose gain is 5 V/m . The pump controlling the flow in is driven by a current and has a gain of $2 \text{ m}^3/\text{s/A}$.

P5.2 A picture of a mechanical accelerometer is shown. The position and acceleration of the housing are given by $z(t)$ and d^2z/dt^2 . The position and acceleration of the mass inside the housing are given by $x(t)$ and d^2x/dt^2 . Find the equation that relates the relative displacement of the mass to the housing to the acceleration of the housing.



P5.3 The following description represents a temperature monitoring system.

A thermocouple with a gain of $40 \mu\text{V}/^\circ\text{C}$ is connected to an amplifier which has a nominal setting of 100. This voltage drives a chart recorder which produces 0.01 mm change for 0.5 mV . The chart recorder is also fairly slow and can be represented by the following first-order differential equation:

$$3 \frac{dx}{dt} + x(t) = K v_a(t)$$

where $x(t)$ is the position, K is the chart recorder gain and $v_a(t)$ is the input driving voltage in mV . The maximum length of paper is 300 mm .

- (a) Produce a block diagram for the system.
- (b) If the temperature changes by up to $35 \text{ }^\circ\text{C}$, investigate the maximum value of the amplifier gain such that the chart recorder position remains within the boundary of the paper.

P5.4 A process contains a storage vat, v_1 , which is fed by the outflow of a vat, v_2 , of similar dimensions. The cross-sectional area of both vats is constant at 15 m^2 . The output flow from v_1 is measured by a flow transducer which provides an output of 1.2 V for a change of $1 \text{ m}^3/\text{s}$ in flow. The input to vat v_2 is controlled by a pump which produces a flow of $2 \text{ m}^3/\text{s}$ for every 1 A . The gravity outflow from the first tank is controlled by a variable restriction, K , on the outlet pipe.

- (a) Produce a schematic diagram for the system.
- (b) Develop the modelling equations for the process.
- (c) Write down the block diagram using Laplace transform notation for the mathematics.

6

Simple systems: first-order behaviour



Gaining confidence



Help?



Going deeper



Skill section



Time to read

Systems behave differently, as do humans. Some people respond quickly to changing information, others take much longer to react. Some people are more extrovert and quickly become over-excited, while others take any change to their life in their stride. If we inject different input signals to systems, we also see that the system behaviour changes – some systems respond quickly, others respond more slowly; some become over-excited and ‘bounce’ around, while others sluggishly follow the input signals. What determines their behaviour is referred to as the ‘dynamics’ of the system. Although the dynamics of many systems are quite complex, we can often approximate complicated system dynamics by simpler representations (which are often quite sufficient for control design).

In our studies we will be examining the response of systems to an input signal which is usually a step change in input level. We find a wide variety of responses such as those shown in Figure 6.1, and these responses depend on the nature of the physical process and its model. In particular, we are interested in the responses of first-order and second-order process models, and we examine these in detail in Chapters 6 and 7.

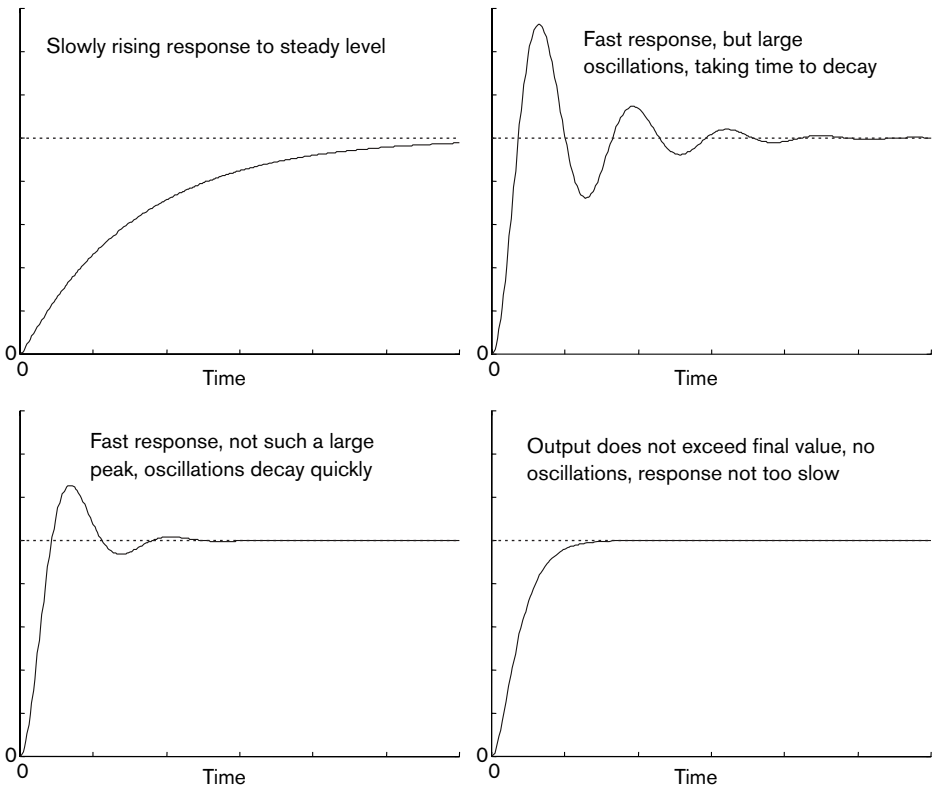


Figure 6.1 Typical system responses.

However, regardless of the particular model, we need to be able to identify different parts of the response; in particular, the regions called the *transient* and *steady state* portions of the response. Transient behaviour in a step response is the early part of the response, which is possibly still rapidly changing before the response settles to its final steady state level (or steady state oscillation). Figure 6.2 shows the regions for a slowly responding process and a process which oscillates in response to a step change in input signal.

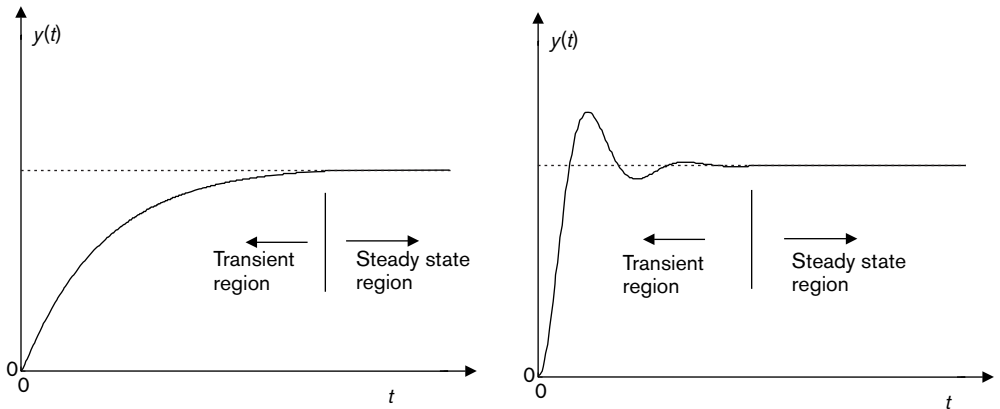


Figure 6.2 Steady state and transient regions of a response.

Therefore, in this chapter and the following one we will examine first-order and second-order systems, so called because the systems are described by first- and second-order differential equations. In fact, we will limit our detailed study to the following categories:

Models and behaviour of simple systems

Chapter 6	Chapter 7
First-order system	Second-order systems:
First-order lag plus deadtime	Underdamped
	Critically damped
	Overdamped
	Second-order system plus deadtime

Learning objectives

- To become familiar with first-order system behaviour.
- To find parameters K and τ from a first-order time response graph.
- To use MATLAB/Simulink to investigate system responses.
- To recognise how the parameter τ affects the speed of response of the system.
- To introduce the idea of deadtime in a system.
- To illustrate the step response of a first-order lag-plus-deadtime system.

6.1 First-order system model

Two different process systems are shown in Figures 6.3(a) and (b).

- (a) In the first system, the level of liquid in the tank is dependent on the flow into the tank.
- (b) In the second system, the voltage across the capacitor is altered by changing the input voltage.

Both these systems can be represented by first-order differential equations.

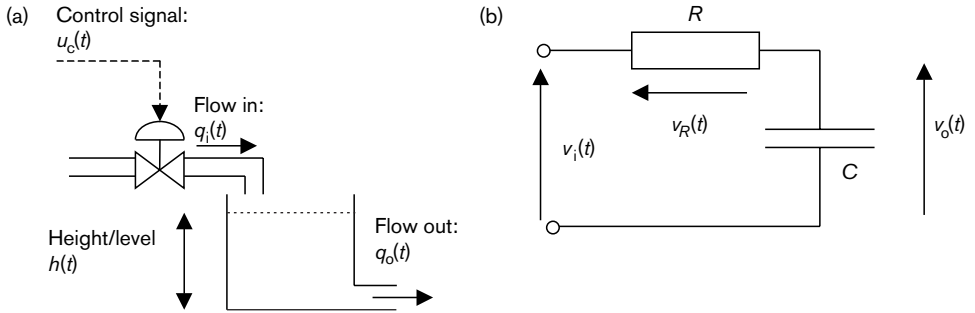


Figure 6.3 Liquid level system and electrical RC circuit.

The equations and their input and output signals are shown in Table 6.1.

Table 6.1 First-order system descriptions.

System	Liquid level system	Electrical RC circuit
Input signal	Flow $q_i(t)$	Input voltage $v_i(t)$
Output signal	Height $h(t)$	Output voltage $v_o(t)$
Differential equation	$RA \frac{dh}{dt} + h(t) = Rq_i(t)$	$RC \frac{dv_o}{dt} + v_o(t) = v_i(t)$
General differential equation	$\tau \frac{dy}{dt} + y(t) = Ku(t)$	$\tau \frac{dy}{dt} + y(t) = Ku(t)$
Parameter K	$K = R = 140$	$K = 1$
Parameter τ	$\tau = RA = 29$ minutes	$\tau = RC = 0.3$ seconds

Both these systems can be written in the form

Key result: First-order parameters

K is referred to as the **system gain** and τ is called the **time constant** for the system.

Laplace transform of first-order differential equation

The Laplace transform for this equation can be found as follows:

$$\mathcal{L}\left\{\tau \frac{dy}{dt} + y(t)\right\} = \mathcal{L}\{Ku(t)\}$$

$$\tau(sY(s) - y_0) + Y(s) = KU(s)$$

where y_0 is the initial value of signal $y(t)$.

$$(\tau s + 1)Y(s) = KU(s) + \tau y_0$$

$$Y(s) = \frac{K}{\tau s + 1} U(s) + \frac{\tau}{\tau s + 1} y_0$$

$U(s)$ is the Laplace transform of the input signal, $u(t)$, and $Y(s)$ is the Laplace transform of the output signal, $y(t)$. The value of y_0 is the initial value of the output (the level in the tank or the voltage across the capacitor). This initial value is often zero, and with this assumption the systems can both be written in block diagram format as



where $G(s) = K/(\tau s + 1)$.

6.2 First-order step response

Let us look at giving a step change in input signal to a general first-order system. We often use a *unit step*, or we can inject a step of size r_0 . If we inject a step of size r_0 to a first-order system (with a time constant of $\tau = 10$) we find that the graph of the output $y(t)$ against time is as shown in Figure 6.4.

The important points that can be read from this graph are the steady state (or infinite time) level of the output, y_{ss} , and the time constant, τ .

Key result: Values from first-order response plot

Steady state value of output: the value of the output reaches a constant level, y_{ss} (steady state value). This value is equal to

$$y_{ss} = Kr_0$$

Value of output at the time constant: The time constant represents the time taken for the output to rise to 63% of the change in output. In the plot, the output changes from zero to a steady state value of Kr_0 . The change is then given by $(Kr_0 - 0) = Kr_0$. The value of the system output when the time reaches the value of the time constant, y_τ

$$y_\tau = 0.63Kr_0$$

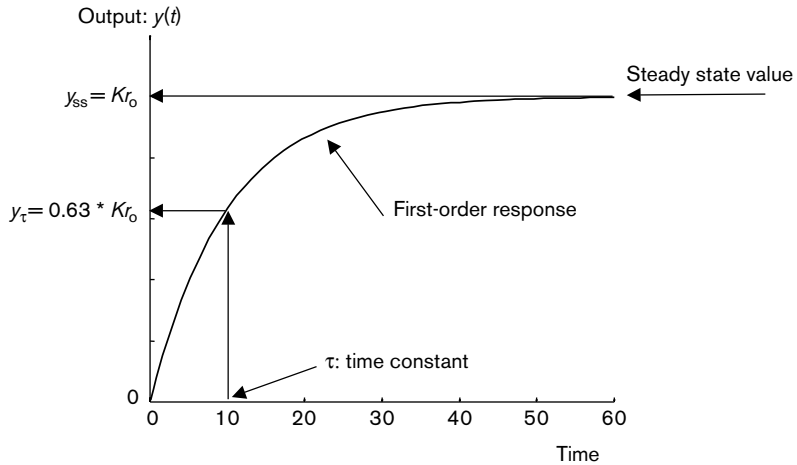


Figure 6.4 First-order response of system.

The exponential rise of the system resembles the typical response for *all* first-order (linear) systems. Hence if you can put the system in the form given by

$$Y(s) = \frac{K}{\tau s + 1} U(s)$$

where we know the values of K and τ , and we know the step size, r_o , of the input signal, we can sketch the output response.

Skill section

Finding K and τ from system plots

We can also work in reverse: given an unknown system, we can inject a step into the system of value r_o and record the output response. If the output plot resembles that in Figure 6.4 and we make the assumption that it is a first-order response, we can use the plot to determine the values of K and τ .

Example

Examine the graph given in Figure 6.5. This plot represents the output level in a tank given that the input signal undergoes a step change of value $r_o = 2$.

Calculation of system gain, K : Read off the steady state value of y_{ss} from the graph. Given the input step size r_o , use the equation $y_{ss} = Kr_o$ to work out the system gain, K .

From the figure we can estimate that $y_{ss} = 280$. Given the fact that $r_o = 2$, we find that $K = 140$.

Calculation of time constant, τ : Calculate the value of y_τ (63% of the change in output). Mark the value on the vertical axis. From the vertical (y -axis) track across to the output curve. Descend to the time axis to determine the time taken for the output to have altered by 63% of its change in value.

From the figure, we can calculate that $y_\tau = 0.63 (280 - 0) = 176.4$. Find this point on the y -axis and track across to the output curve. From this intersection, track down to the time axis to find the time taken to reach this point: approximately 29 minutes. Hence $\tau = 29$.

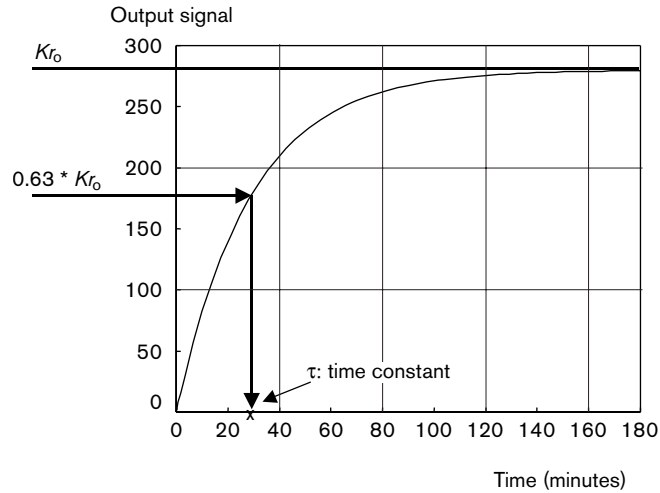


Figure 6.5 Output response of unknown first-order system.

The system, $G(s)$, is then given by

$$G(s) = \frac{140}{29s+1}$$

Remark

We note that in these systems:

The system parameters do not change for different input signals.

Therefore, even if the input signal changes, the values of K and τ , which are related to physical properties of the system, will not alter. Formally, we call these types of system *linear time-invariant* systems.

Example Look at the output responses in Figure 6.6. The input step sizes were 0.5, 1.0 and 1.5 respectively. Find the value of K and τ for plots 1, 2 and 3 and compare the values of K and τ for each plot. (Since the system has not changed, the parameters K and τ should work out to be the same!)

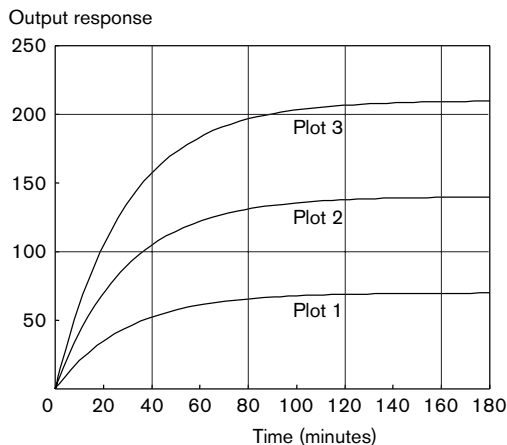


Figure 6.6 Different magnitudes of step size into the same system.

6.3 Positive and negative step signals ('up and down' step signals)

Although we often examine the 'unit step response', which is customarily taken in a positive direction, we can also use step signals in a negative direction. We are effectively injecting input signals which move the process output value up and down from its starting point. For our linear constant coefficient systems we will find that the values of K and τ are found in the same manner for a negative step as for a positive step. For more complex *nonlinear* systems, the size of step and the direction *will* result in different responses.

Example Figure 6.7 shows the output response to a negative step for two different systems:

$$G_A(s) = \frac{K_A}{\tau_A s + 1} \quad G_B(s) = \frac{K_B}{\tau_B s + 1}$$

The step sizes were -20 and -3 respectively. The calculation of K_A and τ_A is done for plot A. The evaluation of K_B and τ_B for plot B is left as an exercise.

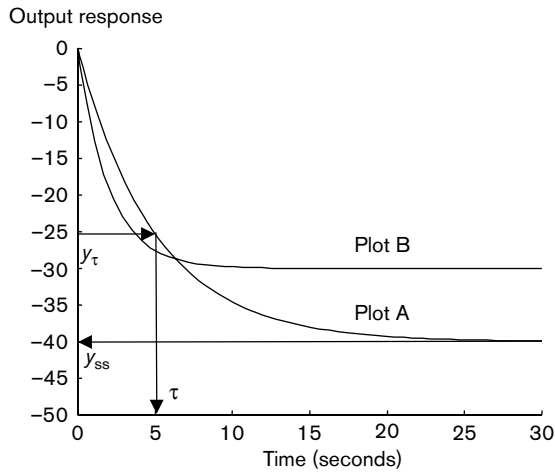


Figure 6.7 Negative step responses.

Plot A:

$$y_{ss} = -40 = K_A r_o$$

Since $r_o = -20$, $K_A = 2$.

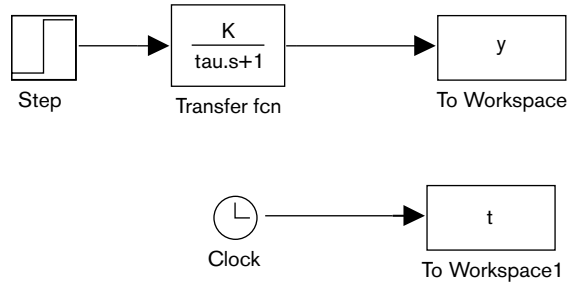
$$y_\tau = 0.63 \times (-40 - 0) = -25.2$$

Time taken for output to reach -25.2 is $t = \tau_A = 5$ seconds.

System A is
$$G_A(s) = \frac{2}{5s + 1}$$

S 6.4 Use of Simulink to find the step response

We can use a Simulink representation to help us examine the output response of $y(t)$ for different sizes of input signal.



Notes:

1. The values of K and τ (tau in the Simulink icon) can be set at the MATLAB command line.
2. The step size is altered by opening the Step icon and entering the new value there.
3. The output response is plotted by using the command `plot(t,y)`.

It is very convenient that, given any first-order system defined by the parameters K and τ , we know the shape of the response and vice versa. We now look at the mathematics behind this and start with the differential equation.

6.5 General first-order system time response

We consider a first-order model which can be represented by the first-order differential equation:

$$\tau \frac{dy}{dt} + y(t) = Ku(t)$$

where

- $y(t)$ represents a varying quantity in the system
- $u(t)$ represents an input
- τ and K represent model parameters (they depend on physical parameters)

The differential equation describes how the system behaves over time given certain inputs. However, to find the equation for the output signal, $y(t)$, we must 'solve the differential equation for $y(t)$ '. We can then plot this signal against time and see how the system behaves. Complex systems are more difficult to integrate, but in control engineering we often use first- and second-order systems as approximations. We have just seen that all first-order systems described by K and τ behave similarly. Let us solve the general first-order differential equation and explain why.

6.5.1 Solution of first-order linear differential equation

We note that the equation is a linear, first-order, constant coefficient, time-invariant differential equation, and therefore the differential equation can be written as

$$\frac{dy}{dt} + \frac{y(t)}{\tau} = \frac{K}{\tau} u(t)$$

This differential equation is valid for all the time points t_1 where $0 < t_1 < t$. To solve this differential equation, we need to integrate it from time 0 to time t .

Common practice for solving differential equations of this form is to introduce an integrating factor, α , given by

$$\alpha = e^{\int_0^t 1/\tau dt_1} = e^{t/\tau}$$

We multiply the left-hand side (LHS) and right-hand side (RHS) of the differential equation by the integrating factor:

$$\text{LHS} = e^{t/\tau} \frac{dy}{dt} + e^{t/\tau} \frac{y(t)}{\tau} = \frac{d}{dt} (e^{t/\tau} y(t)) = \frac{d}{dt} (\text{integrating factor} \times \text{output})$$

$$\text{RHS} = e^{t/\tau} \frac{K}{\tau} u(t)$$

We evaluate the LHS and RHS at $t = t_1$, then integrate both sides over the range $0 < t_1 < t$, and equate:

$$\begin{aligned} \left[e^{t_1/\tau} y(t_1) \right]_0^t &= \frac{K}{\tau} \int_0^t e^{t_1/\tau} u(t_1) dt_1 \\ \Rightarrow e^{t/\tau} y(t) - y(0) &= \frac{K}{\tau} \int_0^t e^{t_1/\tau} u(t_1) dt_1 \end{aligned}$$

This gives

$$y(t) = e^{-t/\tau} y(0) + e^{-t/\tau} \frac{K}{\tau} \int_0^t e^{t_1/\tau} u(t_1) dt_1$$

The solution for $y(t)$ depends on two component signals:

1. the input signal $u(t)$
2. the initial condition $y(0)$; that is, the output at time $t = 0$

Let us now look at two cases:

A putting a step change of value r_o into the model equations:

$$u(t) = r_o, \quad y(0) = 0$$

B no input signal, but non-zero initial conditions

$$u(t) = 0, \quad y(0) = y_o$$

A: Output to a step change, r_o , in input signal

The general solution is given by

$$y(t) = e^{-t/\tau}y(0) + e^{-t/\tau} \frac{K}{\tau} \int_0^t e^{t_1/\tau} u(t_1) dt_1$$

We would like to look at the case when the system is at rest (that is $y(0) = 0$, zero initial conditions) and we supply an input signal, that is we *force* the system. The solution in this case is known as the **forced response**.

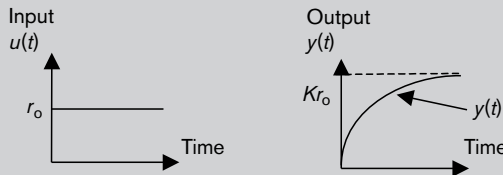
$$y(t) = e^{-t/\tau} \frac{K}{\tau} \int_0^t e^{t_1/\tau} u(t_1) dt_1$$

Let $u(t) = r_0$:

$$\begin{aligned} y(t) &= e^{-t/\tau} \frac{K}{\tau} \int_0^t e^{t_1/\tau} r_0 dt_1 = e^{-t/\tau} \frac{K}{\tau} r_0 \left[\tau e^{t_1/\tau} \right]_0^t = e^{-t/\tau} \frac{K}{\tau} r_0 (\tau e^{t/\tau} - \tau) \\ &= Kr_0 (1 - e^{-t/\tau}) \end{aligned}$$

Key result: Step response of first-order system

The forced response of the system is given by $y(t) = Kr_0(1 - e^{-t/\tau})$.



We see that the output rises exponentially to a steady state value and is dependent on the parameters K and τ and the input signal r_0 .

We will also consider the case where the response of the system has initial conditions other than zero. For example, consider the basic steps in boiling up water in a kitchen kettle. Cold water goes into the kettle at about 4°C . This is the rest or zero initial condition for the boiling process. When we switch on the kettle, heat is supplied at a constant rate by an electrical element. This drives the temperature of the water up to a value of 100°C . If we plotted this temperature rise we would be looking at the *forced response* of the system. After the kettle has been boiling for a few minutes we can safely assume that all the water is at 100°C . If we now switch off the kettle, then the system has no forcing input, and an initial condition of 100°C . A plot of the water temperature against time as it cools down from 100°C would be the *free response* of the system.

B: Output response for no input signal but non-zero initial conditions

Once again the general solution is given by

$$y(t) = e^{-t/\tau}y(0) + e^{-t/\tau} \frac{K}{\tau} \int_0^t e^{t_1/\tau} u(t_1) dt_1$$

The initial condition is given by $y(0) = y_0$ and the input signal is zero, $u(t) = 0$.

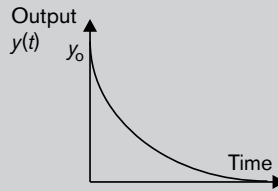
The general solution becomes

$$y(t) = y_0 e^{-t/\tau}$$

and is dependent on τ , but not on K or r_0 . The graph of this output is shown in the Key Result below.

Key result Free response of first-order systems

The free response of the system is given by $y(t) = y_0 e^{-t/\tau}$.



The behaviour of any system is found as the sum of its free response and its forced response:

$$y(t) = y_{\text{free}}(t) + y_{\text{forced}}(t)$$

6.6 System parameters and system behaviour

At the beginning of this chapter, we found that the behaviour of the system depended on the system parameters, K and τ , and that these could be calculated from the response plots. Let us look at the mathematics for the reason why.

Model parameters, K and τ

The output response for a step input of magnitude r_0 and zero initial conditions is given by:

$$y(t) = Kr_0(1 - e^{-t/\tau})$$

Steady state value

We must find the output $y(t)$ as $t \rightarrow \infty$. Clearly as $t \rightarrow \infty$, $e^{-t/\tau} \rightarrow 0$ and

$$y_{\text{ss}} = \lim_{t \rightarrow \infty} y(t) = Kr_0$$

This is what we found from our examples. No matter what the size of r_0 , the steady state value was given by $y_{\text{ss}} = Kr_0$.

Time constant

The value τ is called the *time constant* of the system, with units of time (seconds, minutes, etc.). The actual value of τ will depend on the particular physical system parameters.

When the time reaches the value of the time constant, τ , that is $t = \tau$, we can determine what the output value, $y(t = \tau)$, should be. We use the non-unity step response equation:

$$y(t) = Kr_0(1 - e^{-t/\tau})$$

Then for $t = \tau$:

$$y(\tau) = y_{\tau} = Kx_0(1 - e^{-1})$$

and using $y_{ss} = Kx_0$

$$y_{\tau} = (1 - e^{-1})Kx_0 = 0.632Kx_0 = 0.632y_{ss}$$

Thus when $t = \tau$, the output has climbed to 63.2% of its final or steady state value, y_{ss} . This is why the figure of 63% appeared in our first examination of the first-order time responses. The time constant is a measure of speed of response for first-order systems.

Problem: RC circuit

For the circuit given in Figure 6.3(b) we will

- (a) model the physical system
- (b) derive a Laplace transform model
- (c) form a Simulink model
- (d) evaluate the responses of the system for different input conditions: Case A: $y_0 = 0$, $u(t) = 1$; Case B: $y_0 = 3$, $u(t) = 2$.

The parameters of our system are given as $R = 6 \text{ k}\Omega$ and $C = 50 \text{ }\mu\text{F}$.

Solution (a) Modelling

Voltage law (sum of voltages round a loop = 0):

$$v_i(t) - v_R(t) - v_o(t) = 0$$

Hence,

$$\begin{aligned} v_i(t) &= v_R(t) + v_o(t) = i(t)R + v_o(t) \\ &= RC \frac{dv_o}{dt} + v_o(t) \end{aligned}$$

or, by rewriting in terms of v_o :

$$RC \frac{dv_o}{dt} + v_o(t) = v_i(t)$$

Letting $RC = \tau$ and $K = 1$ gives

$$\tau \frac{dv_o}{dt} + v_o(t) = Kv_i(t)$$

Use standard input–output variables: $u(t) = v_i(t)$, $y(t) = v_o(t)$:

$$\tau \frac{dy}{dt} + y(t) = Ku(t)$$

This differential equation represents the RC circuit for different values of τ ($= RC$).

The value of K for the RC circuit was identified as $K = 1$. The value of τ is calculated from the values of R and C to be 0.3 seconds.

(b) Laplace transformation of the system equation

The Laplace transform of the first-order differential equation was derived as:

$$Y(s) = \frac{K}{\tau s + 1} U(s) + \frac{\tau}{\tau s + 1} y_0$$

Given $K = 1$ and $\tau = 0.3$, the system equation is:

$$Y(s) = \frac{1}{0.3s + 1} U(s) + \frac{0.3}{0.3s + 1} y_0$$

(c) Simulink model

Since we have initial conditions on the output, $y(t)$, of our model we can use the Simulink block 'Transfer function with initial outputs' which is found in the Simulink Extras, 'Additional Linear' library. The Simulink representation is quite simple (Figure 6.8).

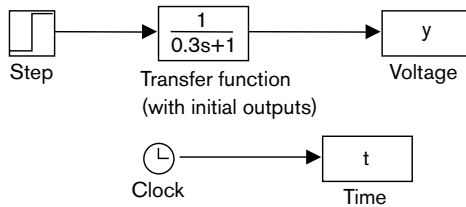


Figure 6.8 Simulink representation of RC circuit.

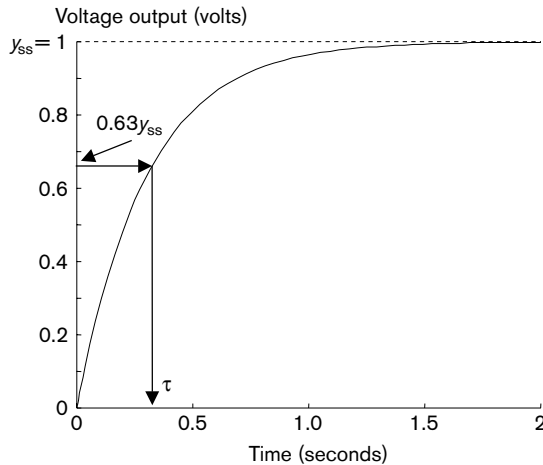
(d) Response analysis

Let us examine the output of the circuit under the two different sets of conditions:

Case A

- (i) No initial voltage on the capacitor so that $y_0 = v_0(0) = 0$.
- (ii) A constant voltage is applied: $u(t) = v_1(t) = 1$ V from time $t = 0$.

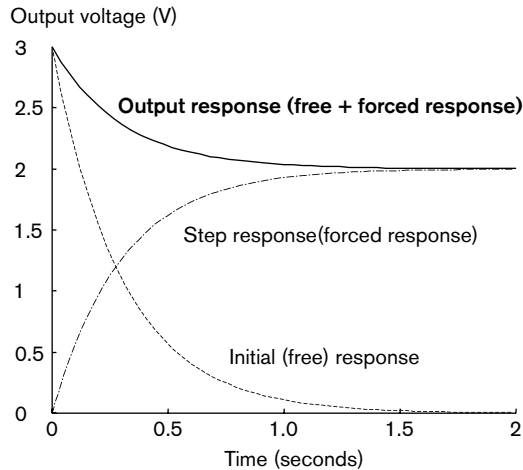
The output response shows the voltage across the capacitor rising to $y_{ss} = Kr_0 = 1$, where $K = 1$ and $r_0 = 1$. The time constant can be read from the graph as 0.3 seconds.



Case B

- (i) The capacitor has an initial voltage of 3 V: $y_0 = v_0(0) = 3$.
- (ii) A constant voltage is applied: $u(t) = v_1(t) = 2$ V from time $t = 0$.

In this case the output is the sum of the free response, due to the initial condition, and the forced response, due to the step voltage applied. Therefore, the capacitor voltage, which starts at 3 V, decreases to the value of the step voltage applied to the system.

**Remark****Simulink response with initial conditions**

The Laplace transform of the system with initial conditions can be expressed by:

$$Y(s) = \frac{K}{\tau s + 1} U(s) + \frac{\tau}{\tau s + 1} y_0$$

This can be represented by the block diagram of Figure 6.9.

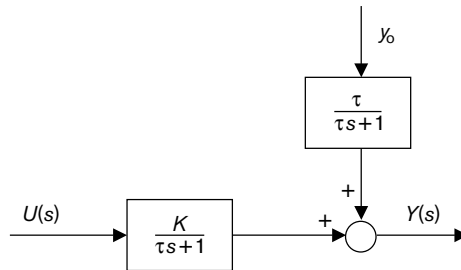


Figure 6.9 Output response with initial conditions.

Although we can represent the system in the block diagram, we must be careful when representing this in Simulink. The input signal, $U(s)$, is often a step signal, $U(s) = r_0/s$, and can be represented by the step signal block in Simulink. However, in Laplace transforms the initial condition y_0 is only active at the initial time, usually $t = 0$. It is incorrect to represent this by a step signal or a constant block. It should be represented by an impulse. There is no single icon block for this and it requires a more complex representation. Therefore, at this stage, for implementing initial conditions on output signals, we recommend using the specific transfer function block (transfer function with initial outputs) if possible, as shown in the *RC* circuit example.

Skill section

Finding different transfer function representations

We have looked at the first-order example and found that the parameters K and τ provide information about the system response. We now examine the different forms of this transfer function and how the parameter τ provides information on how fast the system responds to an input.

The transfer function we have used so far has the form

$$G(s) = \frac{K}{\tau s + 1}$$

From this we can determine the *system gain* K , which is the steady state output value to a *unit* step input. If we write the denominator polynomial in the form

$$d(s) = \tau s + 1$$

where the constant term of $d(s)$ is unity, the time constant, τ , can be clearly identified as the coefficient of s .

Problem The transfer function between output $Y(s)$ and input $U(s)$ is given by

$$G(s) = \frac{10}{6s + 1}$$

- (a) Identify the system gain and its units.
- (b) Identify the time constant and its units.

Solution (a) Since the transfer function is in standard gain–time constant form, we can read off the value of K as

$$K = 10$$

where

$$K \text{ units} = \left[\frac{Y\text{'s units}}{U\text{'s units}} \right]$$

- (b) The time constant τ is identified as $\tau = 6$. The time constant's units depend on the physical definition of τ . However, time constant units will be time units.

Because the gain and time constant can be read easily from the transfer function, this form is known as the gain–time constant form. The transfer function can be rearranged to be put in *pole–zero* form. Although this may look different, there is only some algebraic manipulation between the two forms and the system behaviour will not change.

Key result: First-order transfer function forms

Gain–time constant form: $G(s) = \frac{K}{\tau s + 1}$

Pole–zero form: $G_{pz}(s) = \frac{K_{pz}}{s + a}$

Since both forms may represent the same system, we should be able to find a relationship between K , τ and K_{pz} , a . Let us take the gain–time constant form and divide both numerator and denominator by τ to obtain:

$$G(s) = \frac{K}{\tau s + 1} = \frac{K/\tau}{s + 1/\tau} = \frac{K_{pz}}{s + a}$$

This is now in the form where we can identify K_{pz} and a as:

$$K_{pz} = K/\tau$$

$$a = 1/\tau$$

This shows us that each first-order system transfer function can be written in two equivalent ways using either the (K, τ) format or the (K_{pz}, a) format. However, the system gain and the time constant will remain the same no matter which algebraic form is used for the transfer function.

Problems Find the open-loop gain and time constants for the following:

$$(a) \frac{10}{4s+1} \quad (b) \frac{10}{s+4} \quad (c) \frac{3}{4s+6}$$

Solutions (a) The first is in the gain–time constant form. We can therefore read off the time constant as 4 and the gain as 10.

(b) The second is in pole–zero form. To put it in the gain–time constant form we need to divide the numerator and denominator by 4 to ensure that we have a unity constant coefficient on the denominator, giving $\tau s + 1$ on the denominator. Hence,

$$G(s) = \frac{10}{s+4} = \frac{10/4}{(1/4)s+1} = \frac{2.5}{0.25s+1}$$

The gain is $K = 2.5$ and the time constant $\tau = 0.25$.

(c) This is not in pole–zero form (no unity coefficient of s) and not in gain–time constant form (no unity constant coefficient on denominator); divide numerator and denominator by 6 to obtain the denominator in the form $(\tau s + 1)$:

$$G(s) = \frac{3}{4s+6} = \frac{3/6}{(4/6)s+1} = \frac{0.5}{0.67s+1}$$

The time constant $\tau = 0.67$ [= $1/a$] and the gain $K = 0.5$.

6.7 Speed of response

We have found that the model parameter K has an effect on the steady state or final value of the output, but has no effect on the *shape* of the response. Only the system time constant influences the shape or speed of the output response. We consider the following three systems:

$$(a) G_1(s) = \frac{10}{s+1} \quad (b) G_2(s) = \frac{10}{2s+1} \quad (c) G_3(s) = \frac{10}{5s+1}$$

The time constants of the system transfer functions are given by

$$\tau_1 = 1, \tau_2 = 2, \tau_3 = 5$$

If we plot the *unit* step responses for the three systems we obtain the graph in Figure 6.10. The steady state output levels for all three systems are the same : $y_{ss} = Kr_o = 10$. We insert a line at $0.632y_{ss}$ on the graph so that the time constants are identified.

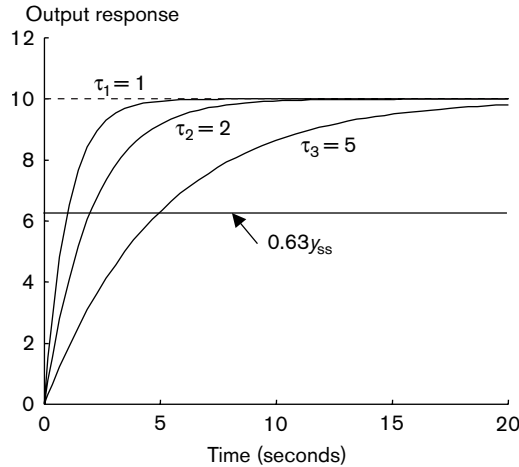


Figure 6.10 Speed of response for different time constants.

As τ increases the response becomes slower; that is, it takes longer to reach 63.2% of the final output. The step response in the time domain is given as

$$y(t) = (1 - e^{-at})Kr_o$$

where again $a = 1/\tau$. A small value for the time constant, for example $\tau = 0.1$, yields a large negative exponent $-a = -1/\tau = -10$, which in turn provides a large negative exponential e^{-10t} in the output: one that will decay quickly. Similarly a large value for τ , say $\tau = 5$, results in a slowly decaying exponential, $e^{-0.2t}$, in the output. Plotting out the step responses as in Figure 6.11 shows the speed of responses.

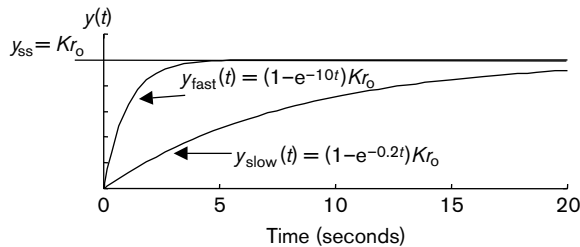


Figure 6.11 Fast and slow step responses.

The difference in the speed of response can be seen clearly.

In this section on first-order systems, we have derived a general description for a single-input single-output first-order system; that is, a system with one input and one output described by a first-order differential equation. We have shown two simple physical systems that can be characterised by this model, although there are many other systems or devices which can have a first-order response, such as some temperature measuring devices.

6.8 Process time delays

One modelling feature which we should include in our collection of simple models is the feature known as *deadtime* or *transport delay*. Figure 6.12 shows a steel-rolling process. The rolls in the stand press on the incoming steel slab to reduce its dimension from H to h . The output strip thickness is measured by an X-ray gaugemeter. However, this is placed at some distance from the rolling stand. Therefore the information that the transducers provide does not coincide with the *current* status of the mill. For example, if the strip travels at a velocity v and the measurement device is a distance L away from the stand, the information is delayed by:

$$T_d = \frac{v}{L}$$

The variable T_d represents the *deadtime* or *delay* in a system; in this example it represents the delay in the measurement system. The measured output of the process will therefore be delayed by a period of time, T_d , called the *deadtime* or *transport delay*.

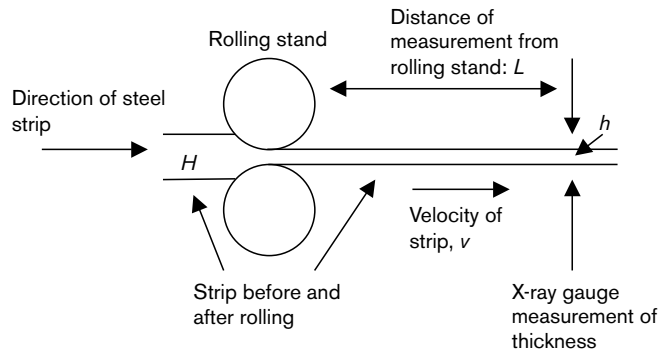


Figure 6.12 Rolling mill process.

S 6.8.1 Simulink model of a heating process

We can illustrate the transport delay using an example of a first-order heating process system. Consider a small vessel which contains liquid which is being heated. The transfer function for the heating system is a first-order process given by:

$$G(s) = \frac{3}{2s+1}$$

However, the measurement of the temperature of the liquid flowing out of the vessel is taken downstream of the vessel (Figure 6.13), causing a delay of 0.5 minutes in the measurement of the temperature. Therefore temperature measurement T_{v2} represents the temperature T_{v1} delayed by 0.5 minutes.

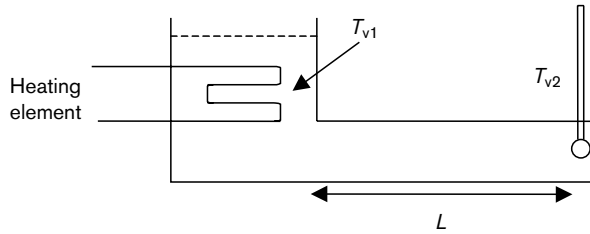


Figure 6.13 Heating system with deadtime in measurement.

The Simulink model of the process is shown in Figure 6.14. The graph of the rise in temperature due to a step change in input is shown in Figure 6.15.

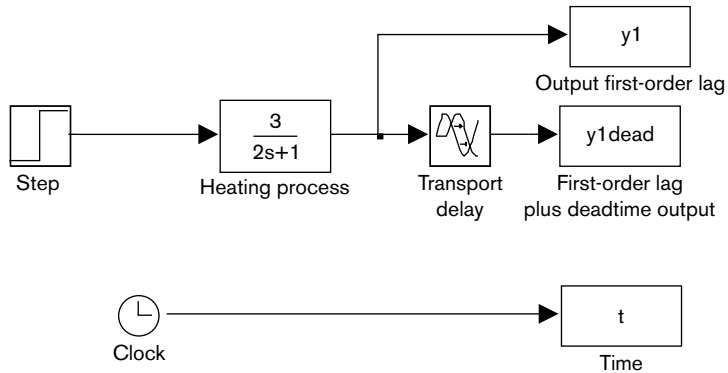


Figure 6.14 Simulink model of transport delays in system models.

The transport delay for the first-order system was 0.5 minutes, which can be read clearly from the output response plot.

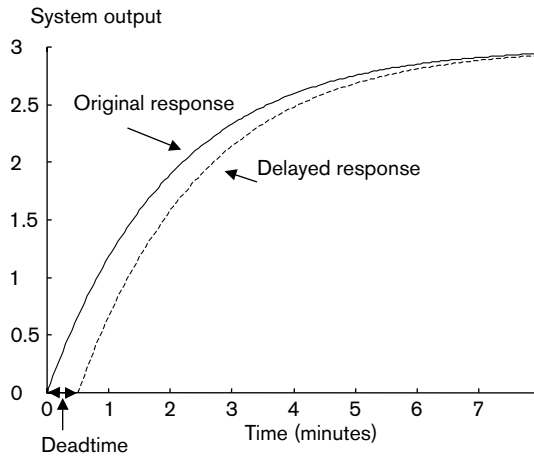
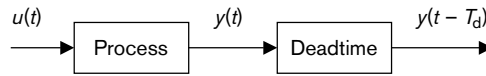


Figure 6.15 Step response of first system with deadtime.

6.9 Modelling of deadtime process

The mathematical equation for the modelling of the deadtime process is given by:



Input to process: $u(t)$

Output from process: $y(t)$

Output from deadtime: $y(t - T_d)$

Key result: Laplace transform of deadtime model

The Laplace transform for the deadtime model is given by

$$\mathcal{L}\{y(t - T_d)\} = e^{-sT_d}Y(s)$$

Derivation of Laplace transform of deadtime model

$$\mathcal{L}\{y(t - T_d)\} = \int_0^{\infty} y(t - T_d)e^{-st} dt$$

Multiply inside the integral by e^{+sT_d} and outside the integral sign by e^{-sT_d} . This leaves the integral expression unchanged.

$$\mathcal{L}\{y(t - T_d)\} = e^{-sT_d} \int_0^{\infty} y(t - T_d)e^{-s(t-T_d)} dt$$

Let $t_1 = t - T_d$. Then $dt_1 = dt$, and

$$\mathcal{L}\{y(t - T_d)\} = e^{-sT_d} \int_0^{\infty} y(t_1)e^{-st_1} dt_1$$

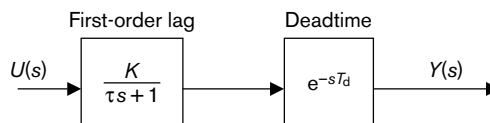
Noting that the right-hand side contains the definition for the transform of $y(t_1)$:

$Y(s) = \int_0^{\infty} y(t_1)e^{-st_1} dt_1$, then

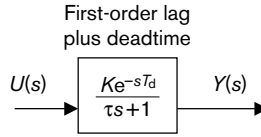
$$\mathcal{L}\{y(t - T_d)\} = e^{-sT_d}Y(s)$$

6.9.1 First-order lag plus deadtime model

Very often in control we meet what is known as the **first-order lag-plus-deadtime** model. This is shown in diagram form as:



These subsystems can be combined to give the total transfer function system representation:



6.9.2 Padé approximation to deadtime model

In our analysis of process systems using linear polynomial Laplace transform representations, it is sometimes inconvenient to have this exponential representation for the delay. We can use an approximation to the delay called a **Padé Approximation**. The approximation is usually first- or second-order, though can be higher order if required:

$$e^{-sT_d} \approx \frac{1 - (T_d / 2)s}{1 + (T_d / 2)s} \quad \text{first-order approximation}$$

$$e^{-sT_d} \approx \frac{1 - (T_d / 2)s + (T_d^2 s^2) / 12}{1 + (T_d / 2)s + (T_d^2 s^2) / 12} \quad \text{second-order approximation}$$

M 6.10 MATLAB function: pade

The MATLAB function pade will provide the Padé approximation to a specified delay. The form of the function is

$$[\text{num}, \text{den}] = \text{pade}(T, \text{order})$$

where T represents the delay and order is the order of the approximation.

```
T=0.5;
order = 1;
[n,d]=pade(T,order) % or you can use [n,d] = pade(0.5,1)
n =
    -1  4
d =
     1  4
```

This gives the solution

$$e^{-sT_d} \approx \frac{-s+4}{s+4} = \frac{4-s}{4+s} = \frac{1-\frac{1}{4}s}{1+\frac{1}{4}s}$$

This is equivalent to the general expression for the above first-order approximation.

What we have learnt

The first-order differential equation with constant coefficients is given by:

$$\tau \frac{dy}{dt} + y(t) = Ku(t)$$

- ✓ The general solution of this equation is:

$$y(t) = e^{-t/\tau} y(0) + e^{-t/\tau} \frac{K}{\tau} \int_0^t e^{t_1/\tau} u(t_1) dt_1$$

- ✓ The solution for a step input of magnitude r_0 and zero initial conditions is:

$$y(t) = Kr_0(1 - e^{-t/\tau}) \quad y(0) = 0$$

- ✓ The solution for a step input of magnitude r_0 and non-zero initial conditions:

$$y(t) = Kr_0(1 - e^{-t/\tau}) + y_0 e^{-t/\tau} \quad y(0) = y_0$$

- ✓ The output equations rely on the two system parameters, K and τ , the initial condition and the input signal.
- ✓ The two equivalent common forms of first-order transfer function are:

$$\text{Gain-time constant form: } G(s) = \frac{K}{\tau s + 1}$$

$$\text{Pole-zero form} \quad G_{pz}(s) = \frac{K_{pz}}{s + a}$$

- ✓ A small value for the time constant, τ , gives a fast response, and a large value produces a slow response.
- ✓ The deadtime in a system represents the pure time delay, T_d , within the system:

$$y(t - T_d)$$

- ✓ The Laplace transform for the deadtime model is given by

$$\mathcal{L}\{y(t - T_d)\} = e^{-sT_d} Y(s)$$

- ✓ A **first-order lag-plus-deadtime model** is given by

$$\frac{Ke^{-sT_d}}{\tau s + 1}$$

Multiple choice

M6.1 What are the gains of the following transfer functions?

$$G_1(s) = \frac{3}{4s+1} \quad G_2(s) = \frac{10}{4s+2}$$

- (a) 0.75, 10
- (b) 3, 10
- (c) 3, 5
- (d) 0.75, 2.5

M6.2 What are the time constants of the following transfer functions?

$$G_1(s) = \frac{2}{0.5s+1} \quad G_2(s) = \frac{6}{3s+2}$$

- (a) 0.5, 1.5
- (b) 0.5, 3
- (c) 4, 2
- (d) 0.5, 2

M6.3 What is the gain and time constant of the following transfer function?

$$G_1(s) = \frac{a}{bs+c}$$

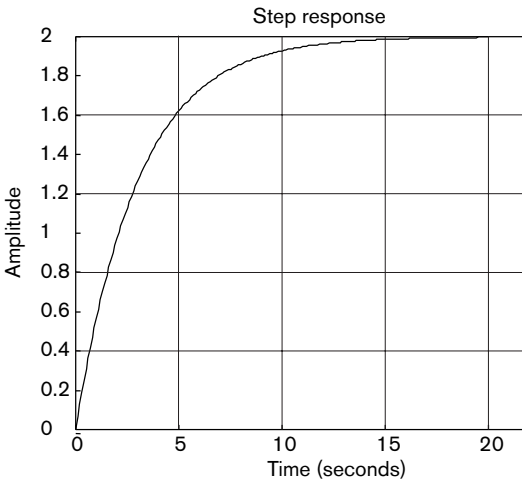
- (a) a, b
- (b) $a/b, b/c$
- (c) $a/c, b/c$
- (d) $c/a, c/b$

M6.4 The standard first-order linear differential equation is given by

- (a) $K \frac{dy}{dt} + \tau y(t) = u(t)$
- (b) $K \frac{d^2y}{dt^2} + \tau = y(t)$
- (c) $\tau \frac{dy}{dt} + Ky(t) = u(t)$
- (d) $\tau \frac{dy}{dt} + y(t) = Ku(t)$

M6.5 The plot shows the unit step response of a first-order system. What is the transfer function of the system?

- (a) $\frac{2}{3s+1}$
- (b) $\frac{1}{5s+1}$
- (c) $\frac{2}{15s+1}$
- (d) $\frac{1}{15s+1}$



M6.6 The value of the time constant is found at what percentage change in output value?

- (a) 50%
- (b) 66%
- (c) 63%
- (d) 100%

M6.7 For an input step of r_0 to a first-order system of standard form, what is the steady state level achieved?

- (a) r_0
- (b) K
- (c) Kr_0
- (d) K/r_0

M6.8 A free response of a system is when:

- (a) the value of $u(t)$ is zero
- (b) the value of $u(t)$ is constant
- (c) the system is allowed to respond freely to input signals, $u(t)$
- (d) the system has zero initial conditions

M6.9 $G_1(s)$, $G_2(s)$ and $G_3(s)$ have the same gain, but time constants of 3, 4 and 5 seconds respectively. Which responds more quickly to a step input?

- (a) They all respond the same since the gain is the same
- (b) $G_1(s)$
- (c) $G_2(s)$
- (d) $G_3(s)$

M6.10 Another name for a process time delay is:

- (a) the time constant
- (b) the integration time
- (c) the deadtime
- (d) the Padé time

Questions: practical skills

Q6.1 Identify the open-loop gain and time constant of the following first-order system transfer functions:

(a) $G_1(s) = \frac{3}{s+10}$

(b) $G_2(s) = \frac{6}{s+1}$

(c) $G_3(s) = \frac{2}{s+3}$

Q6.2 What is the steady state value of the output from the following systems when a step of magnitude 3 is injected?

(a) $\frac{10}{2s+1}$

(b) $\frac{3}{s+4}$

(c) $\frac{10}{2s+3}$

Q6.3 What is the first-order Padé approximation for delays of

(a) 2 seconds

(b) 10 seconds

Q6.4 Sketch the output time response for the system represented by the transfer function:

$$G(s) = \frac{10}{3s+1}$$

to a step input of magnitude 2. Show clearly the time constant of the system on the sketch.

Q6.5 Use Simulink to produce the output time response for the system represented by the transfer function:

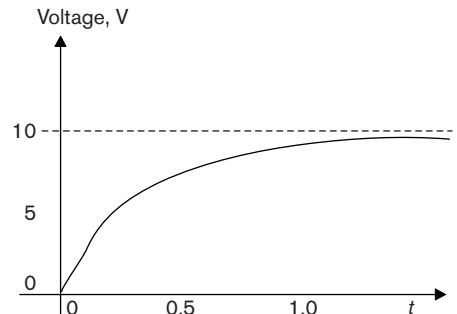
$$G(s) = \frac{6e^{-2s}}{3s+1}$$

to a step input of magnitude 1. Show clearly the time constant of the system on the sketch.

Problems

P6.1 First-order systems can be described by a gain and a time constant. A first-order temperature measuring device has a gain of 2 mm/°C and a time constant of 0.3 seconds. What is the differential equation defining the system? What are the input and output of the system? Draw a block diagram of the system, labelling the input and output clearly.

P6.2 The graph shows the response of an RC circuit to a step in voltage of 10 V. It is known that the capacitance is 50 μF. What is the resistance R for the circuit?



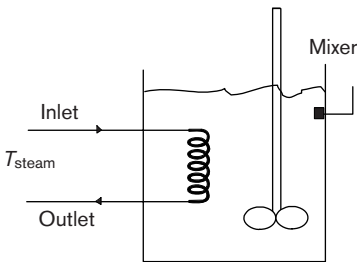
P6.3 The inflow to a liquid level system changes after 8 min from 0 m³/min to

$$q_i(t) = 20 \text{ m}^3/\text{min}$$

The liquid level system has a gain, K , of 0.5 min/m² and a time constant of 5 mins. Sketch the response of the liquid height, assuming the tank is empty at $t = 0$.

P6.4 A mercury thermometer has a time constant of 0.8 seconds and the oven it is being used with has a time constant of 3 hours. Will the time constant of the thermometer have an effect on the plot of temperature against time for the oven?

P6.5 A system comprises a tank of liquid heated by a coil containing saturated steam, as shown in the figure.



The energy balance is given by:

$$\rho c_p V \frac{d}{dt} (T_{LIQ}) = UA_{coil} (T_{steam} - T_{LIQ})$$

where

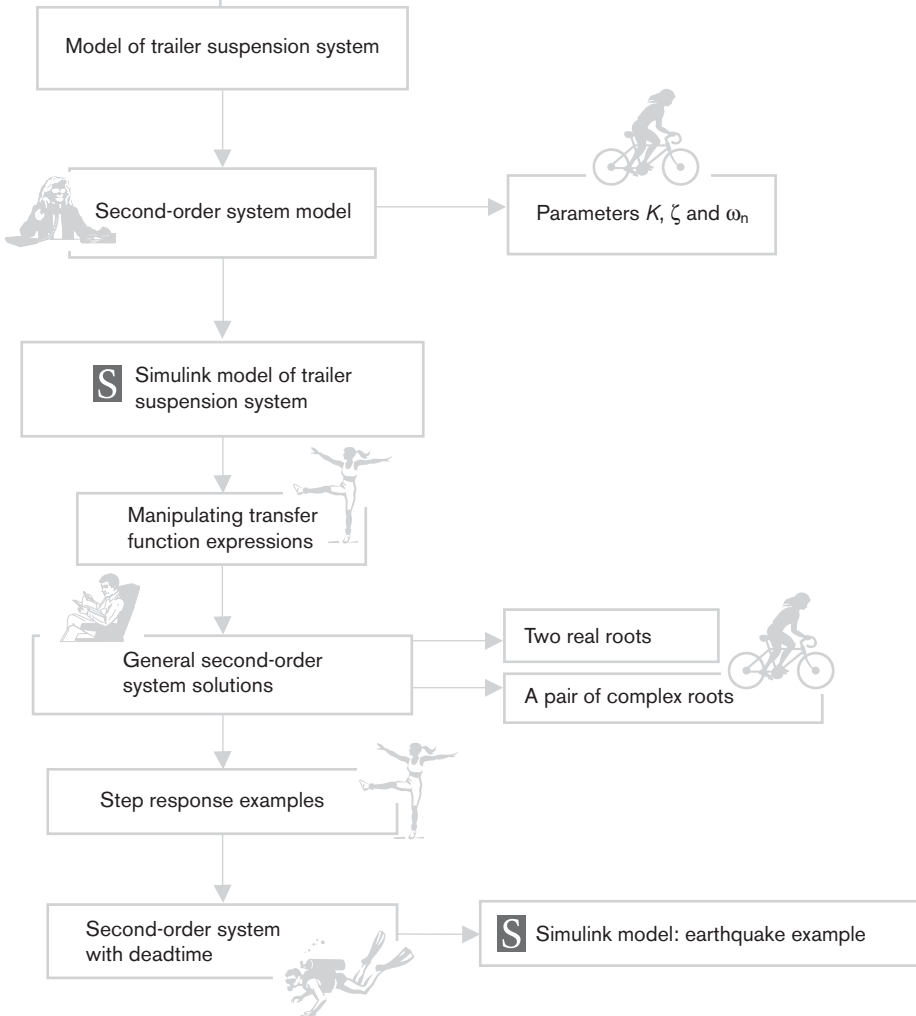
- V = volume of the tank
- c_p = liquid heat capacity
- ρ = liquid density
- A_{coil} = heat transfer area
- U = overall heat transfer between steam and liquid respectively.
- T_{steam}, T_{LIQ} = temperature coefficient of steam and liquid

Explain the energy balance equation on which the model is based.

- (a) Determine the purpose of the mixer in this tank
- (b) Identify system input and output variables
- (c) Identify the associated time constant
- (d) The initial liquid temperature is 20 °C, and the steam is supplied at a constant temperature of 200 °C. Explain with suitable sketch graphs the behaviour of the liquid temperature T_{LIQ} .

7

Simple systems: second-order systems



Gaining confidence



Help?



Going deeper



Skill section



Time to read

Physical systems, like an RC circuit, a liquid level process or a heating process, can be modelled as first-order systems. The type of step input response we obtain from these systems is a smooth exponential rise to a steady output value. For such systems we do not observe either overshoots or oscillations in the output signals. Physical systems which oscillate have models with at least second-order system dynamics. The fact that a physical process can be represented by at least a second-order dynamic model indicates that there are at least two physical effects within the process being described by the model. It is the competition between these two physical effects which gives rise to oscillations in process variables. In mechanical systems, dominant spring effects compete against less effective damping effects, leading to oscillating motions. In hydraulic systems, different liquid levels in interconnected tanks lead to flows between tanks, resulting in oscillating fluctuations in liquid level measurements. In electrical circuits, adding an inductance to an RC circuit to form an RLC circuit will lead to oscillation in voltage measurements. In all of these physical systems, second-order dynamics are present.

We have another set of reasons for studying second-order system models. Very few real systems are exactly second-order; almost all processes are actually of much higher order. However, for many practical control engineering design studies, second-order systems models are a useful tool because:

1. Many systems can be approximated by a process with dominant second-order dynamics.
2. There is a whole toolkit of results based on second-order systems available.
3. The mathematics involved in analysing third or higher order systems is more daunting
4. Control design methods will allow for the modelling inaccuracy in using a second-order approximation to represent a high-order system.

Learning objectives

- To become familiar with second-order system behaviour.
- To use MATLAB/Simulink to investigate system responses.
- To find parameters K , ζ and ω_n from a second-order response graph.
- To understand the effect of changes in these three parameters on the step response of the system.
- To recognise underdamped, overdamped and critically damped system behaviour and relate the behaviour to the system's pole positions.

7.1 Second-order systems: model of a trailer suspension system

We introduce second-order systems by way of an example.

7.1.1 System equations

To test the effectiveness of the trailer's dampers, a force is applied to the stationary system and the resulting motion examined. The input force is given by $f_i(t)$, and this is resisted by the spring force, $f_s(t)$, and the axle damper force, $f_d(t)$. It is assumed that the

trailer wheel is solid. The net effect of these forces is a vertical displacement, $y(t)$, of the trailer mass (Figure 7.1).

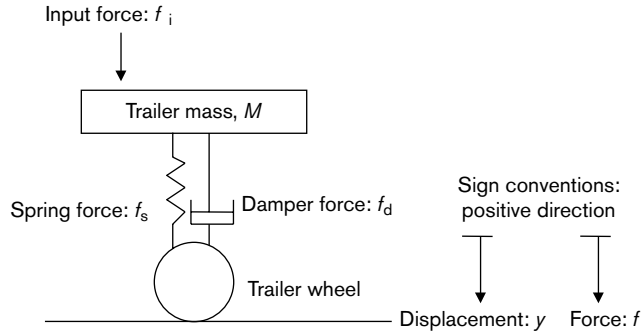


Figure 7.1 Trailer suspension system.

The physical principles are:

Input force: $f_i(t)$
 Spring force from the tyre: $f_s(t) = -K_s y(t)$
 Damper force: $f_d(t) = -B \frac{dy}{dt}$

and these are all drawn together using Newton’s second law of motion:

$$M \frac{d^2y}{dt^2} - f_i(t) - f_d(t) - f_s(t) = 0$$

Hence

$$M \frac{d^2y}{dt^2} - f_d(t) - f_s(t) = f_i(t)$$

giving

$$M \frac{d^2y}{dt^2} + B \frac{dy}{dt} + y(t) = f_i(t)$$

Dividing through by the spring constant, K_s , gives

$$\frac{M}{K_s} \frac{d^2y}{dt^2} + \frac{B}{K_s} \frac{dy}{dt} + y(t) = \frac{1}{K_s} f_i(t)$$

This is a second-order differential equation whose input is $f_i(t)$ and whose output is a resultant vertical displacement of the trailer mass position. The block diagram for this system would be given by Figure 7.2.

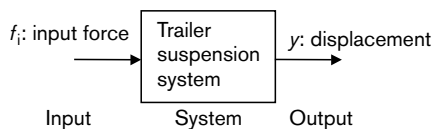


Figure 7.2 Second-order systems.

The system variables and parameters are summarised in Table 7.1.

Table 7.1 Second-order system description.

System	Trailer suspension system
Differential equation	$\frac{M}{K_s} \frac{d^2y}{dt^2} + \frac{B}{K_s} \frac{dy}{dt} + y(t) = \frac{1}{K_s} f_i(t)$
Input signal	Force $f_i(t)$
Output signal	Position $y(t)$
Parameter K_s	80 000 N/m
Parameter B	3464 N/m s ⁻¹

This system can be written in the form

$$\frac{M}{K_s} \frac{d^2y}{dt^2} + \frac{B}{K_s} \frac{dy}{dt} + y(t) = \frac{1}{K_s} f_i(t)$$

and we can move from here to a more general second-order form:

$$\frac{1}{\omega_n^2} \frac{d^2y}{dt^2} + \frac{2\zeta}{\omega_n} \frac{dy}{dt} + y(t) = Ku(t)$$

where $u(t)$ and $y(t)$ represent the input and output signals and where the three parameters K , ζ and ω_n are dependent on system properties. In this case

$$\begin{aligned} \frac{1}{\omega_n^2} &= \frac{M}{K_s} \Rightarrow \omega_n = \sqrt{\frac{K_s}{M}} \\ \frac{2\zeta}{\omega_n} &= \frac{B}{K_s} \Rightarrow \zeta = \frac{\omega_n B}{2K_s} = \frac{B}{2\sqrt{K_s M}} \\ K &= \frac{1}{K_s} \end{aligned}$$

K is referred to as the **system gain**, ζ is referred to as the **damping ratio** and ω_n is the **natural frequency** of the system. We note that the units of ζ are given by

$$\begin{aligned} [\zeta] &= \left[\frac{B}{2\sqrt{K_s M}} \right] = \frac{\text{N}}{\text{m s}^{-1}} \frac{1}{\sqrt{\text{N kg/m}}} = \frac{\text{kg m s}^{-2}}{\text{m s}^{-1}} \frac{1}{\sqrt{\text{kg m s}^{-2} \text{kg/m}}} \\ &= \text{kg s}^{-1} \frac{1}{\text{kg s}^{-1}} = 1 \end{aligned}$$

Hence the damping ratio is dimensionless.

7.1.2 Laplace transformation of second-order differential equation

The Laplace transform of the general second-order differential equation can be found as follows:

$$\mathcal{L}\left\{\frac{1}{\omega_n^2}\frac{d^2y}{dt^2} + \frac{2\zeta}{\omega_n}\frac{dy}{dt} + y(t)\right\} = \mathcal{L}\{Ku(t)\}$$

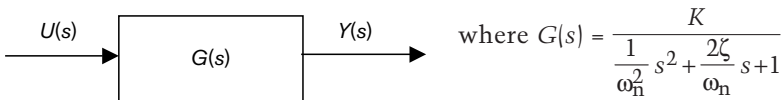
$$\frac{1}{\omega_n^2}(s^2Y(s) - sy_0 - \dot{y}_0) + \frac{2\zeta}{\omega_n}(sY(s) - y_0) + Y(s) = KU(s)$$

where y_0 is the initial value of signal $y(t)$ and \dot{y}_0 is the initial condition for dy/dt . If we let the initial conditions be zero ($y_0 = 0, \dot{y}_0 = 0$):

$$\left(\frac{1}{\omega_n^2}s^2 + \frac{2\zeta}{\omega_n}s + 1\right)Y(s) = KU(s)$$

$$Y(s) = \frac{K}{(1/\omega_n^2)s^2 + (2\zeta/\omega_n)s + 1}U(s)$$

where $U(s)$ is the Laplace transform of the input signal and $Y(s)$ is the Laplace transform of the output signal. The value of y_0 is the initial value of the output, which in this example is the position of the trailer. This initial value is often zero and with this assumption, and the fact that $\dot{y}(0) = 0$ since the trailer is at rest, the system can be written in block diagram form as



In a first-order system description we characterised a general first-order model with two parameters, K and τ . For a second-order general model we need three parameters: K , ζ and ω_n . We now look at each of these parameters individually and examine what effect they have on the step response of a system.

7.2 Second-order system parameters

7.2.1 System gain, K

If the input signal to the second-order system is a step signal of height r_0 , then the input signal transform is $U(s) = r_0/s$. The output signal is given by the transfer function equation

$$Y(s) = G(s)U(s) = \frac{K}{\frac{1}{\omega_n^2}s^2 + \frac{2\zeta}{\omega_n}s + 1} \frac{r_0}{s}$$

The Final Value Theorem (Chapter 2) may be used to determine the steady state value of the output signal, y_{ss} . We have:

$$\begin{aligned} y_{ss} &= \lim_{t \rightarrow \infty} y(t) = \lim_{s \rightarrow 0} sY(s) = \lim_{s \rightarrow 0} sG(s)U(s) \\ &= \lim_{s \rightarrow 0} s \frac{K}{\frac{1}{\omega_n^2}s^2 + \frac{2\zeta}{\omega_n}s + 1} \frac{r_0}{s} \end{aligned}$$

$$y_{ss} = Kr_0$$

Thus, in steady state, the system output will reach a level of Kr_0 . If $K < 1$, then y_{ss} will reach a numerical level lower than that of the input, r_0 . If $K > 1$, the opposite occurs and the input level will be magnified by a factor of K to reach the increased level of Kr_0 .

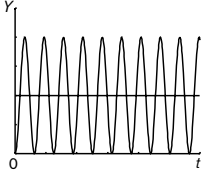
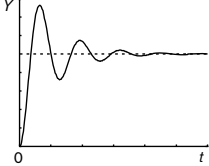
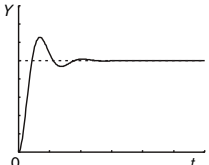
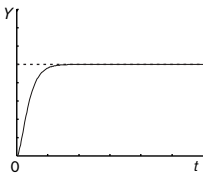
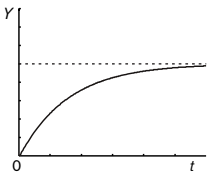
7.2.2 Damping ratio, ζ

The damping ratio, ζ , is an indicator of the type of transient behaviour (or basic shape) expected in a system's response. The damping ratio is a dimensionless parameter and we can classify second-order systems depending on its value:

- If $\zeta > 1$ then the system is *overdamped*.
- If $\zeta = 1$ then the system is *critically damped*.
- If $\zeta < 1$ then the system is *underdamped*.

In fact, if $\zeta = 0$, there would be no damping in the system and any oscillations would continue indefinitely.

Table 7.2 Range of second-order response for different values of damping.

Damping value	Description of output response	Output step response
$\zeta_1 = 0.0$	No damping: output continues to oscillate about the steady state level. There is no decay in the magnitude of the oscillations	
$\zeta_2 = 0.2$	Underdamped: output has large overshoot on steady state level, output is oscillatory (though not unstable)	
$\zeta_3 = 0.4$	Underdamped: output is less oscillatory, but still overshoots the steady state level, to a lesser degree	
$\zeta_4 = 1.0$	Critically damped: output has no overshoot and rises quickly to steady state level	
$\zeta_5 = 4.0$	Overdamped: output has no overshoot, the response is slow and sluggish, takes a longer time to reach the steady state level	

The damping ratio is the ratio of the actual system damping (given by the damping constant B in the trailer's physical equations) to the damping required for a *critically damped* response. We have found that the damping ratio is sometimes referred to, colloquially, as the damping factor. However, we note that in many books, some authors use *damping factor* or *damping coefficient* to refer to the specific term $\zeta\omega_n$. We shall refer to ζ as the damping ratio throughout this text. The terms *overdamped*, *critically damped* and *underdamped* become clearer through the following example.

Example Consider the system step response (Table 7.2) for five different systems which are described by the parameters K , ω_n and ζ . We will keep K and ω_n constant ($K = 1$, $\omega_n = 1$) and vary ζ ($\zeta_1 = 0.0$, $\zeta_2 = 0.2$, $\zeta_3 = 0.4$, $\zeta_4 = 1.0$, $\zeta_5 = 4.0$). For $\zeta = 0.0$, the output is oscillatory and the oscillations do not decay. This is because there is no damping in the system; in practice, although the damping ratio may be small, it will not be zero. For $\zeta = 0.2$ and $\zeta = 0.4$ the systems show an oscillatory response. They are both *underdamped* ($\zeta < 1$) and the response oscillates around the final value. The lower the damping value, the more oscillatory is the system response. As the damping approaches unity the system no longer oscillates and there is no overshoot visible. When $\zeta = 1$, the system is *critically damped*. It does not overshoot the final value.

For values of $\zeta > 1$, the system has a slow response due to the fact that the system is *overdamped*. There are no oscillations in this response.

7.2.3 Natural frequency, ω_n

If there were no damping in a system, $\zeta = 0$, the output response would contain oscillations that did not decay. The frequency of these oscillations would be the natural frequency (sometimes referred to as the *undamped natural frequency*) of the system. Since all systems have some damping, even if they are *lightly damped*, the frequency of the oscillations is given by the **damped natural frequency**, ω_d :

$$\omega_d = \omega_n \sqrt{1 - \zeta^2}$$

For values of $\zeta \leq 0.5$, the value of the damped natural frequency approaches that of the natural frequency. We can see this in Table 7.3, which shows how the value of ω_d becomes closer to $\omega_n = 1 \text{ rad s}^{-1}$ as ζ becomes smaller.

Table 7.3 Range of damped natural frequency for different values of damping.

ζ	ω_n	$\omega_d = \omega_n \sqrt{1 - \zeta^2}$
0.9	1.0	0.4359
0.7	1.0	0.7141
0.5	1.0	0.8660
0.3	1.0	0.9359
0.1	1.0	0.9950
0.05	1.0	0.9987

S 7.2.4 Simulink model of trailer suspension system

We could test the trailer suspension by

- (a) applying a test mass to the trailer body and noting the change in the deflection of the trailer position
- (b) applying an initial force to the trailer body which will produce an initial deflection of the trailer body. When we release the trailer, it will return to its resting position.

The first example provides a step response of the system, whereas the second test shows us the free response of the system from an initial position.

Both output signals are determined from the model of the system that we have obtained. The only difference is in the initial conditions and the input signals.

We will examine the initial condition response of the system (test (b)) and examine how damped the system is.

We use the following Simulink model (Figure 7.3) to analyse the trailer suspension. We have used the ‘Transfer function with initial outputs’ block from Simulink.

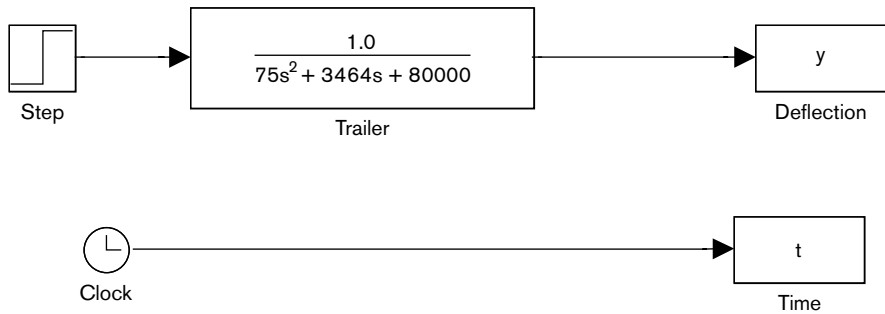


Figure 7.3 Simulink representation of trailer suspension model.

The parameters are given in Table 7.1 and therefore the Laplace transform representation is given by:

$$Y(s) = \frac{1}{(Ms^2 + Bs + K_s)} U(s) = \frac{1}{75s^2 + 3464s + 80000} U(s)$$

We have decided to deflect the trailer by 10 mm (= 0.01 m) and monitor the output position, $y(t)$. Although we have put the step icon as input signal to the trailer, we have set all the step parameters to zero since we are only interested in the system’s *free response*. Figure 7.4 shows the trailer deflection returning to its resting (‘0’) position.

We note that although the trailer overshoots the resting position, it returns quickly with no further overshoot. This is typical of a system with a damping ratio of approximately 0.7. We can check the damping ratio of our system by rearranging the equation to have a unity constant coefficient in the denominator and then equate terms in powers of s .

$$Y(s) = \frac{1}{(Ms^2 + Bs + K_s)} U(s) = \frac{1/K_s}{(M/K_s)s^2 + (B/K_s)s + 1} U(s) = \frac{K}{(1/\omega_n^2)s^2 + (2\zeta/\omega_n)s + 1} U(s)$$

Hence

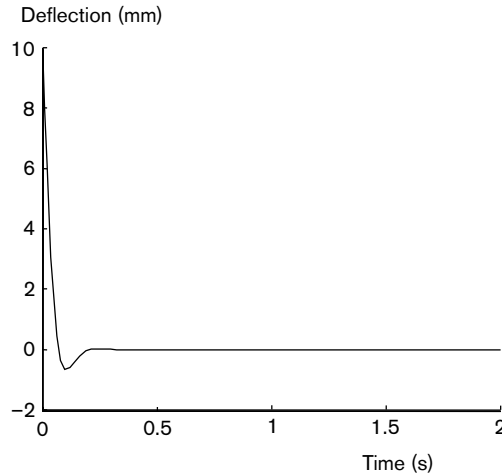


Figure 7.4 Trailer response after release from initial deflection of 10 mm.

$$\text{Parameter } K: \quad \frac{1}{K_s} = 0.0000125 \text{ m/N}$$

$$\text{Parameter } \zeta: \quad \frac{B}{2\sqrt{K_s M}} = 0.707$$

$$\text{Parameter } \omega_n: \quad \sqrt{\frac{K_s}{M}} = 32.66 \text{ rad/s}$$

and we note that the damping ratio is indeed 0.707.

7.3 Second-order transfer function forms

In deriving our physical model for the trailer system, we arrived at a second-order system model which we then turned into the standard (K, ζ, ω_n) second-order transfer function form. Sometimes, as in the trailer suspension example, it is useful to work with other forms of second-order model.

Skill section

Equivalent transfer function expressions

The three common forms are

$$\text{General polynomial form} \quad G_1(s) = \frac{b_0}{a_2 s^2 + a_1 s + a_0}$$

$$\text{Unity constant coefficient form} \quad G_2(s) = \frac{K}{(1/\omega_n^2)s^2 + (2\zeta/\omega_n)s + 1}$$

$$\text{Unity } s^2 \text{ coefficient form} \quad G_3(s) = \frac{K\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2}$$

The first transfer function $G_1(s)$ is formed from numerator and denominator polynomials. However, it is the other two forms that we commonly work with. The **Unity constant coefficient**

form has a '1' as the constant coefficient on the denominator. The **Unity s^2 coefficient form** has a '1' as the coefficient of the s^2 term in the denominator. Any one of these transfer function forms can be turned into the other forms by simple algebraic manipulation. You should be able to recognise each form and determine the parameters K , ζ and ω_n .

Problems Determine K , ζ and ω_n for each of the following transfer functions:

$$(a) \quad G_A(s) = \frac{6}{2s^2 + 3s + 1}$$

$$(b) \quad G_B(s) = \frac{24}{s^2 + 2s + 36}$$

$$(c) \quad G_C(s) = \frac{6}{24s^2 + 12s + 48}$$

Solution Consider each example in turn:

- (a) The transfer function $G_A(s)$ is in the unity constant coefficient form because the constant term in the denominator is 1. We can determine the values of K , ζ , ω_n by comparing the coefficients of s^2 , s^1 and s^0 in the denominator.

Compare

$$\frac{K}{(1/\omega_n^2)s^2 + (2\zeta/\omega_n)s + 1} \quad \text{with} \quad \frac{6}{2s^2 + 3s + 1}$$

Hence

$$\frac{1}{\omega_n^2} = 2, \quad \frac{2\zeta}{\omega_n} = 3 \quad \text{and} \quad K = 6$$

This gives

$$\omega_n = \frac{1}{\sqrt{2}}, \quad \zeta = \frac{3\omega_n}{2} = \frac{3}{2\sqrt{2}} \quad \text{and} \quad K = 6$$

- (b) The transfer function $G_B(s)$ is in unity s^2 coefficient form.

Compare

$$\frac{K\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} \quad \text{with} \quad \frac{24}{s^2 + 2s + 36}$$

Hence $2\zeta\omega_n = 2$, $\omega_n^2 = 36$ and $K\omega_n^2 = 24$

Consequently

$$\omega_n = 6, \quad \zeta = \frac{1}{\omega_n} = \frac{1}{6} \quad \text{and} \quad K = \frac{24}{\omega_n^2} = \frac{24}{36}$$

- (c) Finally, $G_C(s)$ is seen to be of general polynomial form.

Compare

$$\frac{b_0}{a_2s^2 + a_1s + a_0} \quad \text{with} \quad \frac{6}{24s^2 + 12s + 48}$$

It is our experience that fewer errors are made if the general form is converted into the standard form where the constant term in the denominator is unity. For this particular example, it

is necessary to divide all the terms in the numerator and denominator by 48 to give a unit constant coefficient:

$$\frac{6}{24s^2 + 12s + 48} = \frac{6/48}{(24/48)s^2 + (12/48)s + (48/48)}$$

$$= \frac{0.125}{0.5s^2 + 0.25s + 1}$$

Compare

$$\frac{K}{(1/\omega_n^2)s^2 + (2\zeta/\omega_n)s + 1} \text{ with } \frac{0.125}{0.5s^2 + 0.25s + 1}$$

Hence

$$\frac{1}{\omega_n^2} = 0.5, \quad \frac{2\zeta}{\omega_n} = 0.25 \quad \text{and} \quad K = 0.125$$

giving $\omega_n = \sqrt{0.5}$, $\zeta = 0.1768$ and $K = 0.125$.

This example showed how to re-scale the transfer function to be able to extract K , ζ and ω_n correctly. Once these are known they can be used to predict the general shape of a second-order system's step response.

7.4 Solving general second-order equations

Control engineers do not in general solve differential equations in the manner taught in mathematics classes. We use the Laplace transform tables technique to make it simpler. An excerpt from the Laplace transform tables showing the second-order time and Laplace transform functions as well as the transforms for a second-order systems with a step ($1/s$) input is given in Table 7.4.

Table 7.4 Excerpt from Laplace transform tables.

	Laplace transform	Time domain function
Second-order (overdamped)	$\frac{1}{(s+a)(s+b)}$	$\frac{1}{b-a}(e^{-at} - e^{-bt})$
Second-order system with step signal input (overdamped)	$\frac{1}{s(s+a)(s+b)}$	$\frac{1}{ab}\left(1 - \frac{b}{b-a}e^{-at} + \frac{a}{b-a}e^{-bt}\right)$
Second-order (critically damped)	$\frac{1}{(s+a)^2}$	te^{-at}
Second-order system with step signal input (critically damped)	$\frac{1}{s(s+a)^2}$	$\frac{1}{a^2}(1 - e^{-at} - ate^{-at})$
Second-order (underdamped, oscillatory)	$\frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} \quad \zeta < 1$	$\frac{\omega_n}{\sqrt{1-\zeta^2}}e^{-\zeta\omega_n t} \sin(\omega_d t)$ $\omega_d = \omega_n\sqrt{1-\zeta^2}$
Second-order system with step signal input (underdamped, oscillatory)	$\frac{\omega_n^2}{s(s^2 + 2\zeta\omega_n s + \omega_n^2)} \quad \zeta < 1$	$1 - \frac{1}{\sqrt{1-\zeta^2}}e^{-\zeta\omega_n t} \sin(\omega_d t + \phi)$ $\phi = \tan^{-1} \frac{\sqrt{1-\zeta^2}}{\zeta}, \quad \omega_d = \omega_n\sqrt{1-\zeta^2}$

The solution procedure for a general second-order linear differential equation with constant coefficients:

$$a \frac{d^2y}{dt^2} + b \frac{dy}{dt} + cy = Ku(t)$$

entails finding the auxiliary or characteristic equation and solving it to find the roots which are then used as the exponential rate constants in the solutions. Table 7.5 shows the equations required, both in typical mathematical notation and in the engineering notation we use in this book.

Table 7.5 Summary of equations required in solution of second-order linear differential equation with constant coefficients.

	General mathematical form	Engineering equations
General second-order equation	$a \frac{d^2y}{dt^2} + b \frac{dy}{dt} + cy = Ku(t)$	$\frac{1}{\omega_n^2} \frac{d^2y}{dt^2} + \frac{2\zeta}{\omega_n} \frac{dy}{dt} + y = Ku(t)$
Auxiliary equation (or characteristic equation)	$ap^2 + bp + c = 0$	$\frac{1}{\omega_n^2} p^2 + \frac{2\zeta}{\omega_n} p + 1 = 0$
Roots of auxiliary equation	$p_{1,2} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$	$p_{1,2} = -\zeta\omega_n \pm \omega_n \sqrt{\zeta^2 - 1}$

The solution to the second-order equation is dependent on the two roots of the auxiliary or characteristic equation. The two roots (p_1 and p_2) will either be two real numbers or a complex conjugate pair, depending on whether ($\zeta^2 - 1$) is greater or less than zero; or equivalently, whether ζ is greater or less than 1 (Table 7.6).

Table 7.6 Roots of characteristic equation and relation to system damping.

Roots, p_1, p_2	Value of ζ	System damping
Two real roots	$\zeta > 1$	Overdamped
Two equal real roots	$\zeta = 1$	Critically damped
Two complex roots	$\zeta < 1$	Underdamped

You should note the link between the auxiliary equation and the equation which represents the denominator of the transfer function/Laplace transform of the signal or system. For **overdamped** systems, the roots of the auxiliary equation provide the *exponential rate constant* in the solution of the differential equation. In **underdamped** systems (complex roots), the *real* part of the roots gives the exponential rate constant. In control engineering terms we refer to the roots of the denominator of the transfer function as **poles**.

Problem: Two real roots

Given the second-order differential equation

$$\frac{1}{4} \frac{d^2y}{dt^2} + 3 \frac{dy}{dt} + y = u(t)$$

find the output step response of the system to a unit step change in input signal: $u(t) = 1$, where $y(0) = 0$ and $\dot{y}(0) = 0$.

Solution Firstly we form the Laplace transform expression for this system:

$$Y(s) = \frac{1.0}{0.25s^2 + 3s + 1} U(s)$$

Since the input signal is a step, we find that the output $Y(s)$ is given by

$$Y(s) = \frac{1.0}{s(0.25s^2 + 3s + 1)}$$

We note that the transfer function is the unity constant coefficient form and that

$$K = 1$$

$$1/\omega_n^2 = 0.25, \quad \omega_n = 2$$

$$2\zeta/\omega_n = 3, \quad \zeta = 3$$

Since $\zeta > 1$:

(i) the system is overdamped, and

(ii) we can therefore expect two real roots from the characteristic equation.

The output response of the system to a step input is:

$$y(t) = \mathcal{L}^{-1}\{Y(s)\} = \mathcal{L}^{-1}\left\{\frac{1.0}{s(0.25s^2 + 3s + 1)}\right\}$$

From transform tables we find

$$\mathcal{L}^{-1}\{Y(s)\} = \mathcal{L}^{-1}\left\{\frac{1}{s(s+a)(s+b)}\right\} = y(t) = \frac{1}{ab} \left(1 - \frac{b}{b-a} e^{-at} + \frac{a}{b-a} e^{-bt}\right)$$

We can rewrite the transfer function to give a unity coefficient in front of the s^2 term.

$$\frac{1.0}{s(0.25s^2 + 3s + 1)} = \frac{1.0}{0.25s[s^2 + (3/0.25)s + (1/0.25)]} = \frac{4.0}{s(s^2 + 12s + 4)}$$

The roots of the denominator of the transfer function are given by

$$s(s - a)(s - b) = 0$$

giving

$$a = -11.66 \text{ and } b = -0.34$$

These will give rise to components in the step response of the form $e^{-11.66t}$ and $e^{-0.34t}$.

$$y(t) = (1 + 0.03e^{-11.66t} - 1.03e^{-0.34t})$$

The overdamped output response is shown in Figure 7.5.

Problem: two complex roots

Using the Laplace transform expression for the second-order response to a unit step input, write down the output signal, $y(t)$, for the following system:

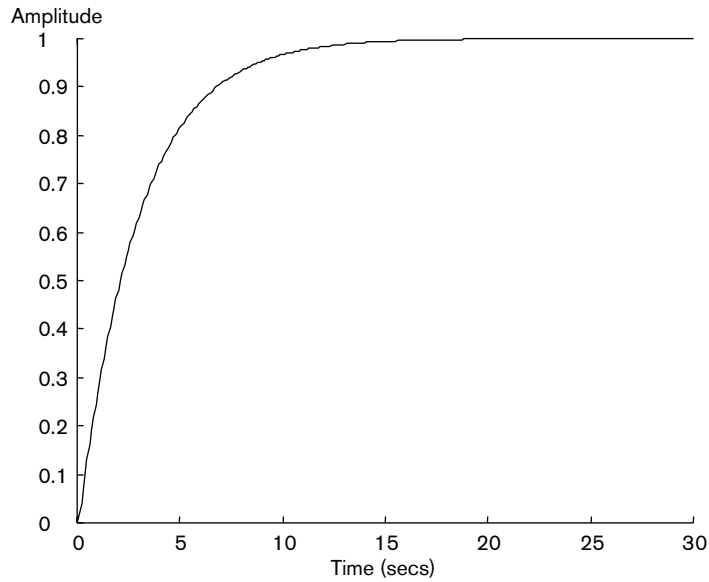


Figure 7.5 Overdamped step response.

$$0.11 \frac{d^2y}{dt^2} + 0.2 \frac{dy}{dt} + y = 6u(t)$$

where we assume $y(0) = 0$ and $\dot{y}(0) = 0$.

Solution Once again we find that the Laplace transform of the system is given by:

$$Y(s) = \frac{6.0}{0.11s^2 + 0.2s + 1} U(s)$$

Since the input signal is a step, we find that the output $Y(s)$ is given by

$$Y(s) = \frac{6.0}{s(0.11s^2 + 0.2s + 1)}$$

The system parameters are given by

$$K = 6$$

$$1/\omega_n^2 = 0.11, \quad \omega_n = 3$$

$$2\zeta/\omega_n = 0.2, \quad \zeta = 0.3$$

Since $\zeta < 1$:

- (i) the system is underdamped, and
- (ii) we can therefore expect two complex roots from the auxiliary equation.

The output response is given by

$$y(t) = \mathcal{L}^{-1}\{Y(s)\} = \mathcal{L}^{-1}\left\{\frac{6.0}{s(0.11s^2 + 0.2s + 1)}\right\}$$

Using the appropriate entry in the Laplace transform tables, we find that the general expression for the output of a second-order underdamped system to a step input is given by

$$y(t) = K \left(1 - \frac{1}{\sqrt{1-\zeta^2}} e^{-\zeta\omega_n t} \sin(\omega_d t + \phi) \right) \quad \text{where } \phi = \tan^{-1} \left(\frac{\sqrt{1-\zeta^2}}{\zeta} \right) \text{ and } \omega_d = \omega_n \sqrt{1-\zeta^2}$$

For our system this becomes

$$y(t) = 6 \left(1 - \frac{1}{\sqrt{0.91}} e^{-0.9t} \sin(3\sqrt{0.91}t + \phi) \right) \quad \text{where } \phi = \tan^{-1} \left(\frac{\sqrt{0.91}}{0.3} \right)$$

We note that the complex poles are given by

$$p_{1,2} = -\zeta\omega_n \pm j\omega_n\sqrt{1-\zeta^2} = -\zeta\omega_n \pm j\omega_d = -0.9 \pm j3\sqrt{0.91}$$

Real part of pole = -0.9

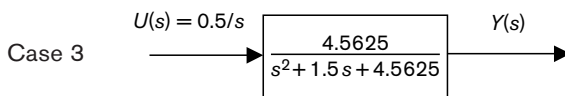
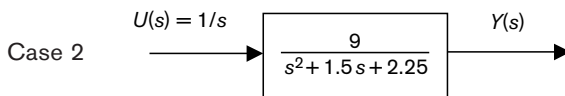
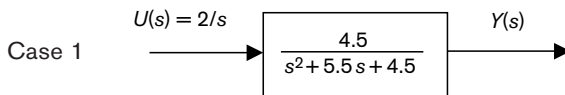
Imaginary part of pole = $3\sqrt{0.91}$

We can see that the real part of the pole ($-\zeta\omega_n = -0.9$) gives the exponential rate constant $e^{-\zeta\omega_n t} = e^{-0.9t}$. The imaginary part of the roots, $3\sqrt{0.91}$ provides the damped natural frequency of oscillation (ω_d).

Skill section

Step response examples

In control system engineering it is invaluable to be able to predict the type of step response which is likely from a system. The behaviour of second-order systems, as established through their unity s^2 coefficient form is an essential means of developing this skill. We might be faced with the problem of predicting the step response for the following three cases:



The three examples are chosen to demonstrate the link between the gain, damping ratio and the steady state output and response shape.

Evaluate steady state value y_{ss}

To evaluate the steady state value for the step response two items are needed: the height of the step input, r_0 , and the d.c. system gain, K . Through the use of the Final Value theorem, the steady state output $y_{ss} = Kr_0$.

Case No.	Gain K	Input step height	Steady state output, y_{ss}
$G_1(s)$	1	1	1
$G_2(s)$	4	0.5	2
$G_3(s)$	1	0.5	0.5

Determine the response shape

To determine the basic shape of the step response, we first need to determine whether the system is underdamped, critically damped or overdamped. Then the roots of the transfer function denominator and the value of damping ratio are used together to give an indication of the speed of decay of the exponential components and the frequency of oscillation within the response when this is appropriate.

System transfer	Roots of characteristic equation	Damping factor, ζ	Response shape	Comments
$G_1(s)$	$p_1 = -1$ $p_2 = -4.5$	1.296	Overdamped	Of the components e^{-t} and $e^{-4.5t}$, the e^{-t} component is the slowest
$G_2(s)$	$p_1 = -0.75$ $p_2 = -0.75$	1	Critically damped	$p_1 = -0.75, p_2 = -0.75$ indicates slow exponential terms $e^{-0.75t}, te^{-0.75t}$
$G_3(s)$	$p_1 = -0.75 + j2$ $p_2 = -0.75 + j2$	0.35	Underdamped	Real(s) = -0.75 indicates slow exponentials Imag(s) = $\pm 2j$ indicates damped frequency Damping $\zeta = 0.35$ is low

We have used all the information contained in the denominator poles and the value of damping ratio, ζ to provide guidance in determining the shape of the transient portion of the step response.

Plotting the step response

M Matlab plots

Use the matlab tf command to form the transfer function

```
g=tf(num,den)
```

then use the step command to give a unit step input (if you require a step of greater than one unit, you need to multiply the output, or the transfer function, g, by the magnitude of the step):

```
step(2*g)
```

The final step is to list the features which can be used to sketch a step response or which can be identified from a step response plot. The output responses are shown in Figure 7.6. Matlab has been used to produce these plots.

Case 1: Transfer function $G_1(s)$

Input step 1/s, d.c. gain $K = 1$; steady state output $y_{ss} = 1$

Damping $\zeta = 1.296 > 1$. Overdamped with slowest component of step response at e^{-t} .

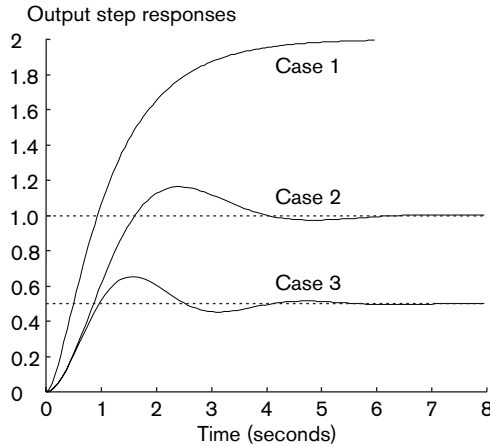


Figure 7.6 Output responses for three cases.

Case 2: Transfer function $G_2(s)$

Input step $0.5/s$, d.c. gain $K = 4$; steady state output $y_{ss} = 2$

Damping $\zeta = 1$. Critically damped with slow exponential terms $e^{-0.75t}$, $te^{-0.75t}$, but no overshoot or oscillation.

Case 3: Transfer function $G_3(s)$

Input step $0.5/s$, d.c. gain $K = 1$; steady state output $y_{ss} = 0.5$

Damping $\zeta = 0.35 \ll 1$. Underdamped with complex roots at $-0.75 \pm j2$, giving slow exponential envelope $e^{-0.75t}$, oscillatory content $\omega_d = 2$ rad/s

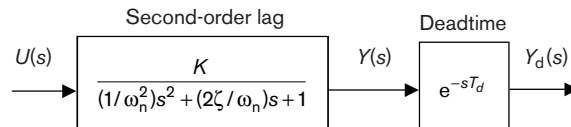
7.5 Modelling of second-order systems with deadtime

As we can have a deadtime with a first-order system, so we can have a deadtime associated with a second-order process.

We recall from Chapter 6 that the Laplace transform for the deadtime model is given by

$$\mathcal{L}\{y(t - T_d)\} = Y_d(s) = e^{-sT_d} Y(s)$$

The block diagram for the modelling of the deadtime process is given by:



and can be combined to give the total transfer function system representation. The output $y_d(t)$ will be a delayed signal with the same 'shape' as $y(t)$. We illustrate this by an example.

S 7.6 Simulink model of second-order system with deadline

We can illustrate the transport delay using an example of a second-order model of a building oscillating under an external force, such as an earthquake.

7.6.1 Earthquake input to building model

Consider a building which oscillates slowly when it receives a force due to an earthquake phenomenon. The transfer function representing the building oscillations (from input force to output position) is given by a second-order process:

$$G(s) = \frac{1}{s^2 + 0.6s + 4}$$

The output position of the system is measured, but there is a processing delay in the measurement system, which gives a delay of 1 second. The Simulink model of this process is shown in Figure 7.7.

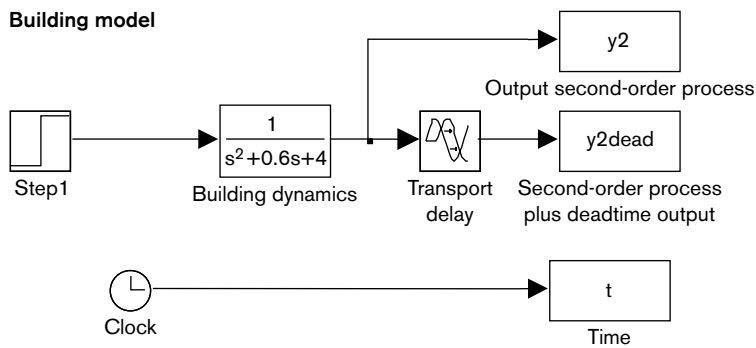


Figure 7.7 Simulink model of transport delays in system models.

The output response of the system is given in Figure 7.8 (opposite). The transport delay was 1 second which can be read clearly from the output response plot. The delay of 1 second is quite large compared with the time for the initial transient. This can cause problems for systems, since the information on the process behaviour has been delayed in reaching a monitoring or control system. This can cause instability within the system (Chapter 10).

What we have learnt

- ✓ The second-order differential equation with constant coefficients is given by:

$$\frac{1}{\omega_n^2} \frac{d^2y}{dt^2} + \frac{2\zeta}{\omega_n} \frac{dy}{dt} + y = Ku(t)$$

- ✓ The Laplace transform representation of this second-order system is given by:

$$G(s) = \frac{K}{(1/\omega_n^2)s^2 + (2\zeta/\omega_n)s + 1}$$

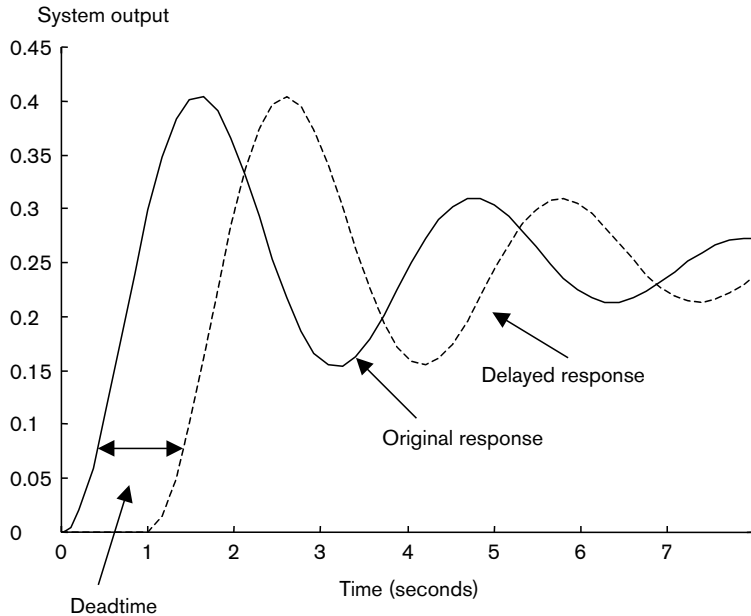


Figure 7.8 Step response of second-order system with deadline.

- ✓ We need three parameters to describe the system:

Gain K

Natural frequency ω_n (damped natural frequency $\omega_d = \omega_n \sqrt{1 - \zeta^2}$)

Damping ratio ζ

- ✓ The denominator of the transfer function is called the *characteristic equation* or *auxiliary equation*.
- ✓ The step response of the system depends on the roots, p_1 and p_2 , of the denominator of system transfer function:

$$p_{1,2} = -\zeta\omega_n \pm \omega_n \sqrt{\zeta^2 - 1}$$

- ✓ The response of a second-order system (with complex roots) to a unit step input is:

$$y(t) = K \left(1 - \frac{1}{\sqrt{1 - \zeta^2}} e^{-\zeta\omega_n t} \sin(\omega_d t + \phi) \right) \quad \text{where } \phi = \tan^{-1} \left(\frac{\sqrt{1 - \zeta^2}}{\zeta} \right)$$

$$\text{and } \omega_d = \omega_n \sqrt{1 - \zeta^2}$$

- ✓ The value of the roots, the value of ζ and the nature of the system step response are all related, as shown in the following summary table:

Roots, p_1, p_2	System damping	System step response	Value of ζ
Two real roots	Overdamped	Slow exponential rise, no overshoot, no oscillations	$\zeta > 1$
Two real equal roots	Critically damped	Exponential rise, no overshoot, no oscillations	$\zeta = 1$
Two complex roots	Underdamped	Overshoot and oscillatory response (how much depends on value of ζ)	$\zeta < 1$

Multiple choice

- M7.1** If a second-order system has a damping ratio of $\zeta = 0.3$, is it:
 (a) underdamped?
 (b) critically damped?
 (c) overdamped?
 (d) a system with no damping?

- M7.2** What are the gain and natural frequency of the following system transfer function?

$$G(s) = \frac{36}{s^2 + 3s + 36}$$

- (a) 36, 6
 (b) 6, 6
 (c) 1, 6
 (d) 6, 1

- M7.3** What is the damping ratio of the following system transfer function?

$$G(s) = \frac{27}{s^2 + 1.8s + 9}$$

- (a) 3
 (b) 0.9
 (c) 0.1
 (d) 0.3

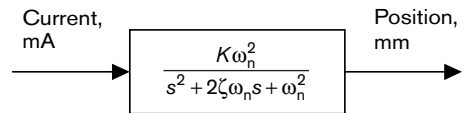
- M7.4** Which of these descriptions is true of the step response of an overdamped system?

- (a) it rises to a steady state value with no overshoot
 (b) it rises to a steady state value with little overshoot
 (c) it rises to a steady state value with a large overshoot
 (d) it does not settle to a steady state value

- M7.5** If a system has two complex roots in the denominator, it is described as:

- (a) having no damping
 (b) being underdamped
 (c) being critically damped
 (d) being overdamped

- M7.6** For the following second-order system, what are the dimensions of ζ ?



- (a) mm/mA
 (b) mA/mm
 (c) mm
 (d) it is dimensionless

- M7.7** An equivalent representation of

$$G(s) = \frac{3}{2s^2 + 4s + 1}$$

in unity s^2 coefficient form is:

- (a) $G(s) = \frac{3}{s^2 + 2s + 0.5}$
 (b) $G(s) = \frac{1}{0.66s^2 + 1.33s + 0.33}$
 (c) $G(s) = \frac{1.5}{s^2 + 2s + 0.5}$
 (d) $G(s) = \frac{1.5}{s^2 + 4s + 1}$

M7.8 The roots of the characteristic equation for

$$G(s) = \frac{4}{12s^2 + 11s + 2}$$

are:

- (a) $s_1 = -4, s_2 = -3$
- (b) $s_1 = -0.25, s_2 = -0.67$
- (c) $s_1 = -0.25, s_2 = -2$
- (d) $s_1 = +4, s_2 = +0.67$

M7.9 The steady state output of the system

$$G(s) = \frac{32}{s + 16}$$

to an input step of magnitude 3 is:

- (a) 3
- (b) 96
- (c) 6
- (d) 48

M7.10 If the roots of characteristic equation are given by $s_{1,2} = -3 \pm 2j$, what is the damped natural frequency in rad/s?

- (a) e^{-2t}
- (b) e^{-3t}
- (c) 2
- (d) 3

Questions: practical skills

Q7.1 A standard second-order system has the form

$$G(s) = \frac{K}{(1/\omega_n^2)s^2 + (2\zeta/\omega_n)s + 1}$$

Three systems have the parameters given by:

System 1: $\omega_n = 0.5, \zeta = 0.1, K = 1$

System 2: $\omega_n = 2, \zeta = 2, K = 2$

System 3: $\omega_n = 1.0, \zeta = 0.3, K = 3$

For each system, determine if it is under- or overdamped and write down the *damped* natural frequency for each system.

Q7.2 Convert the following transfer functions into the unity s^2 coefficient form.

(a) $G_1(s) = \frac{3}{7s^2 + 4s + 9}$

(b) $G_2(s) = \frac{4}{2s^2 + 3s + 1}$

(c) $G_3(s) = \frac{6}{3s^2 + 2s + 36}$

Q7.3 What are the values of K, ζ and ω_n for the following systems. Also state the form of the second-order transfer function (standard, unity s^2 coefficient, unity constant coefficient).

(a) $G_1(s) = \frac{3}{s^2 + 2s + 9}$

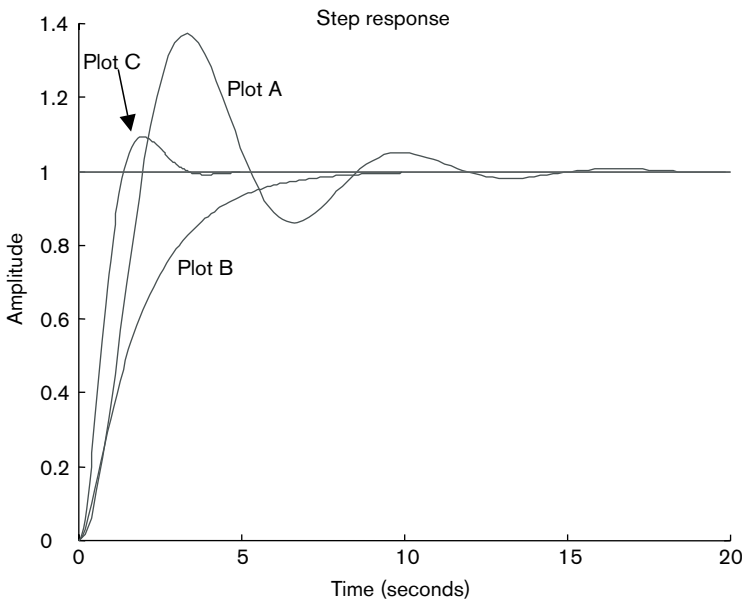
(b) $G_2(s) = \frac{4}{2s^2 + 2s + 1}$

(c) $G_3(s) = \frac{100}{100s^2 + 10s + 1}$

Q7.4 For the following systems, determine the model parameters, K , ζ and ω_n , and state whether the system is overdamped, underdamped or critically damped.

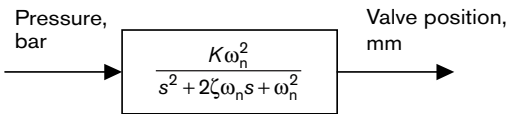
- (a) $36 \frac{d^2x}{dt^2} + 6 \frac{dx}{dt} + x = 6u$
- (b) $\frac{d^2x}{dt^2} + 2 \frac{dx}{dt} + x = 14u$
- (c) $\frac{d^2z}{dt^2} + 48 \frac{dz}{dt} + 6z = 12F$
- (d) $60u - 4y - 8 \frac{dy}{dt} = \frac{d^2y}{dt^2}$

Q7.5 Which of the following plots represents an underdamped system? Will the roots of the characteristic equation be real or complex?



Problems

P7.1 A pneumatic control valve has the following block diagram representation:



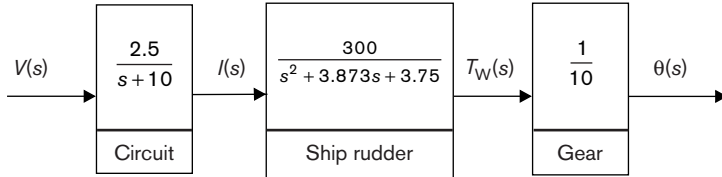
- (a) What are the units of gain K ?
- (b) If $\omega_n = 2 \text{ rad s}^{-1}$, $\zeta = 0.7$ and K has a value of 2.36, what is the transfer function of system $G(s)$ in unity constant coefficient form?

P7.2 A d.c. motor has the transfer function description relating angular velocity, $\omega(t)$, to input voltage, $v(t)$:

$$\frac{\omega(s)}{V(s)} = \frac{20}{0.001s^2 + 0.6s + 1}$$

- (a) What are the values of K , ω_n , and ζ ?
 (b) Using the value for ζ , predict the type of velocity response to a step input voltage change.
 (c) Using the value for ζ , predict the expected roots of the characteristic equation. Verify your prediction by direct evaluation.

P7.3 Consider a cascade of systems given by:



- (a) Assuming a step of magnitude 8 is applied to $V(s)$, sketch the signal $i(t)$ indicating the steady state values.
 (b) A control engineer wishes to predict the type of step response from $V(s)$ to $\theta(s)$. Using your step response sketch for intermediate signal $i(t)$ would it be reasonable to approximate $i(t)$ by a step input function. If so, what height step should be selected for $I(s)$.
 (c) Using this step input for $I(s)$, sketch the heading response $\theta(s)$.
 (d) What evidence could be offered that the ship's rudder system has been designed to be critically damped?

P7.4 Given a system represented in transfer function form as:

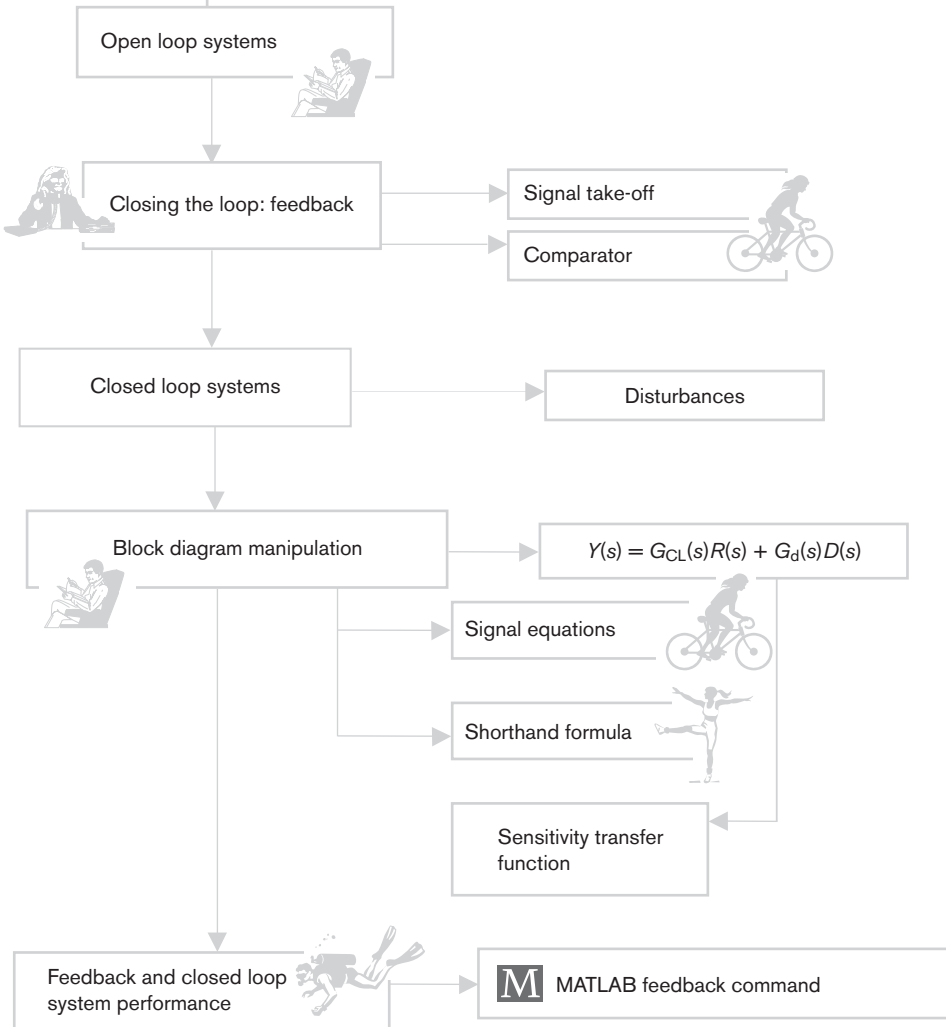


where $U(s)$, $I(s)$, $T_q(s)$ and $\Omega(s)$ are the Laplace transforms of the signals, $u(t)$, $i(t)$, $t_q(t)$ and $\omega(t)$ respectively.

- (a) What is the transfer function from $U(s)$ to $\Omega(s)$.
 (b) Find the steady state values of the signals $i(t)$, $t_q(t)$ and $\omega(t)$ if the applied input $U(s)$ is a unit step.
 (c) Draw the signal $\omega(t)$ given that the applied input $U(t)$ is step of magnitude two units.
- P7.5** A liquid level system comprises two non-interacting tanks which can be represented by two first-order transfer functions in cascade having time constants $\tau_1 = 100$ s and $\tau_2 = 300$ s.
- (a) Draw a block diagram for the system. Use inflow $q(t)$ as input and the level of the second tank, namely $h_2(t)$, as output.
 (b) What would you expect the response of the output height to a step change in flow input to be?
 (c) Given that the overall gain of the system is $1 \text{ m/m}^3 \text{ s}^{-1}$ determine the system transfer function. Work out the damping factor and natural frequency of the system. Do they verify your answer to part (b)?
 (d) If the system had different time constants $\tau_1 = 100$ s and $\tau_2 = 400$ s, how would this affect the system response?

8

Feedback improves system performance



Every day we use feedback. For example, when we drive a car, we constantly adjust the steering or speed depending on what our senses are telling us about the current road situation. In this way, our eyes are acting as a measuring device and our brain as a controller and *closing the loop* from the car's position or speed back to the steering or accelerating inputs. This feedback concept is also used in engineering systems; we often use electronic measurement and transducing equipment to *feed back* the information about how well the system is performing so that the controller can make an adjustment and correct for any errors. We have already looked at modelling the Actuator–Process–Transducer components of engineering systems; now, we look at how to extend our system knowledge and control engineering skills to create a transfer function representation of a closed-loop system and to understand what advantages a closed-loop system has over an open-loop system.

Learning objectives

- To understand the limitations of an open-loop control system.
- To appreciate the advantages of feedback control.
- To construct a closed-loop block diagram of a system using the controller, process, actuator and transducer components.
- To perform block diagram manipulation of a closed-loop process.
- To perform some simple closed-loop analysis.

8.1 Open and closed-loop systems

Consider the example of a vehicle's windscreen wipers. The driver switches the windscreen wipers to 'intermittent wipe' which gives a fixed time interval between the clearing of the windscreen. Then, regardless of whether the rain has eased or if the windscreen has been soaked by a deluge of water from a passing truck, the wipers remain at their preset level. It requires a *manual* adjustment to allow for any increase in frequency of wipers clearing the screen. This is an open-loop system (Figure 8.1(a)) since there is no *automatic* adjustment of the wipe timing.

Recent developments in sensor technology have provided sensors which can detect the moisture level on windscreens. This information can be passed electronically to the motor timing circuits which can then alter the wipe frequency. By doing this we are *feeding back* information from the current situation to the wiper controller (Figure 8.1(b)). We are *closing the loop*.

To 'close the loop' for a system we require a measurement of a system variable which can be used by the controller to make decisions on how to alter the controller output signal to improve the system performance. In the windscreen wiper example, we can see that the system would work *without* feedback, but by including the closed loop we can improve the system performance in terms of keeping the windscreen free of water. Some systems may perform adequately with open-loop control; for example, there may be no disturbing influences which cause the output to move from its manually set level. However, many systems do require feedback to compensate for disturbances affecting the processes or variations in the system parameters.

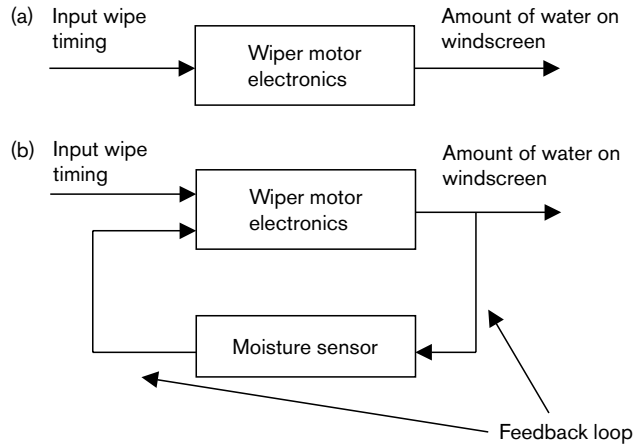


Figure 8.1 Open- and closed-loop systems for windscreen wipers.

We use the example of an open-loop heating system to demonstrate why feedback is useful.

8.1.1 Open-loop heating system

An open-loop control scheme is one where the manipulated input signal does not *automatically* depend on the actual process output. For example, consider a heating system in a university building (Figure 8.2): the output temperature of a particular lecture room rises 0.5 °C for every kW of power applied.

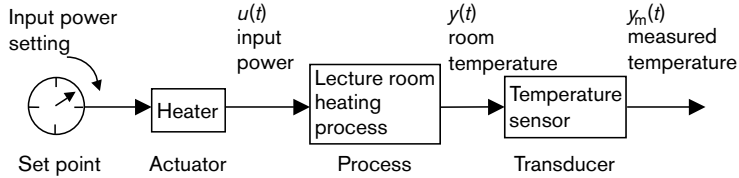


Figure 8.2 Room heating system.

The system operator can then determine that if a temperature rise of 4 °C is required, 8 kW will need to be applied to the system. Therefore, for any particular output level, the operator must manually reset the input level. Although it is tiresome for operators to make continual changes, this system will work well when we know exactly how the output changes for specific input levels.

However, there are often **disturbance** effects that can alter the temperature: external weather, opening and closing doors providing draughts, 100 students sitting in a lecture theatre providing bodily warmth, etc. These disturbances are represented in Figure 8.3 by the signal $d(t)$.

In this diagram we have *two* input signals: $u(t)$ and $d(t)$. Both signals will affect the output $y(t)$ but each input will have no effect on the other input signal: the inputs are independent of each other. We can consider a simplified model (Figure 8.4) where the output signal is given by

$$y(t) = 0.5(u(t) + d(t))$$

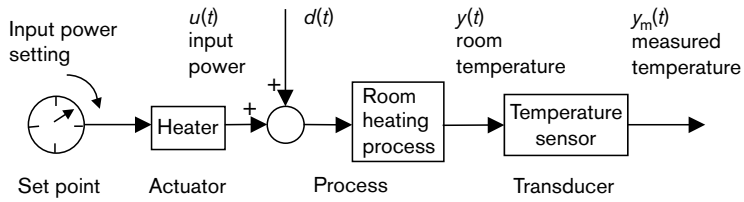


Figure 8.3 Heating system plus disturbance.

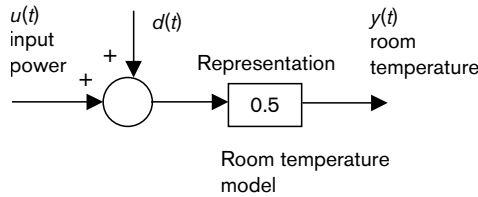


Figure 8.4 Simplified model of heating system.

If we let $d(t)$ be modelled as an effective heating power of 3 kW due to the combined heat of the students, then the output temperature for an input power of $u(t) = 40$ kW is:

$$y(t) = 0.5 \times (40 + 3) = 21.5 \text{ }^\circ\text{C}$$

giving an error of 1.5 °C. The system is an open-loop system, since there is no automatic adjustment for any errors in output temperature that may occur. If there is any error this must first be spotted by the heating system operator and then any alteration must be done manually.

We note that we can also represent the heating system in Laplace transforms (Figure 8.5). The system transfer function is given by $G(s) = 0.5$.

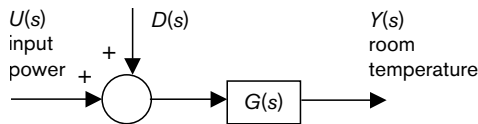


Figure 8.5 Laplace transform block diagram of heating system.

Problem A plant uses a pump on a pipeline to increase pipeline flow. The plant in Figure 8.6 represents the flow, $y(t)$, along the pipe which also suffers from a leakage after the pump. The leakage is indicated by the disturbance flow $d(t)$ on the output flow signal $y(t)$.

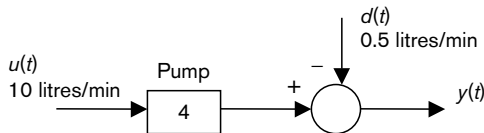


Figure 8.6 Model of pipeline flow.

- (a) What is the output flow?
- (b) The flow is required to be 40 litres/min. What would the operator have to set $u(t)$ to maintain $y(t)$ at this value?
- (c) The output flow is required to increase to 46 litres/min, but at the same time it is noticed that the leakage has increased to 1.5 litres/min. What would the operator have to set the input flow to be to achieve an output of 46 litres/min?

Solution (a) From Figure 8.6 the output flow is given by

$$\begin{aligned} y(t) &= 4u(t) - d(t) \\ &= 39.5 \text{ litres/min} \end{aligned}$$

(b) We wish to maintain $y(t) = 40$ litres/min:

$$\begin{aligned} y(t) &= 4u(t) - d(t) \\ 40 &= 4u(t) - 0.5 \\ u(t) &= 40.5/4 = 10.125 \text{ litres/min} \end{aligned}$$

(c) We wish to achieve $y(t) = 46$ litres/min:

$$\begin{aligned} y(t) &= 4u(t) - d(t) \\ 46 &= 4u(t) - 1.5 \\ u(t) &= 47.5/4 = 11.875 \text{ litres/min} \end{aligned}$$

In many instances the size or source of the disturbance acting on a process is unknown, but the effect is observed on the output signal. To obtain the required output, we must manually reset the input to the process. This can be time-consuming and inefficient if we have a process that suffers from many disturbance effects.

8.2 Introducing feedback into control

Returning to the temperature control problem, if there were no disturbances acting on the heating process we would not need to change the input power to achieve the desired **set point** or output value. In open-loop control we must reset the controller input values or live with the consequences – overheated lecture rooms, excessive use of energy etc. The way to rectify this is to inform the control system what is going wrong. We could use a temperature measurement to determine how far the *actual* temperature is from the set point, or **reference** temperature. This will produce an error signal

$$\text{error} = \text{reference value} - \text{measured value}$$

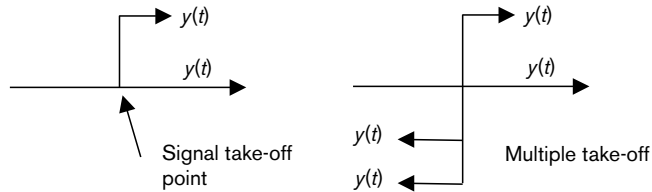
which can be used by the control system in what is called a **feedback** control system. The feedback is said to be **negative** when the signal which is fed back is used to correct the difference between the reference value and the actual value of the controlled variable. Positive feedback occurs when the signal fed back reinforces the error, usually causing instability.

We need to introduce some additional block diagram notation.

8.2.1 Signal notation

1. Signal take-off

Often a signal is required for use in another part of a system, such as for monitoring the information or for control purposes. Symbolically this is represented by a take-off point.



2. A comparator

A comparator is the special name we give to a subtraction operation which generates an *error* between the desired value of a signal and its actual measured value. Common notation used in this situation is:

$e(t)$: error signal

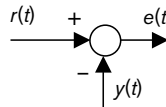
$r(t)$: reference or set point

$y(t)$: output signal

The error relationship, which is often implemented in control systems, is then given by

$$e(t) = r(t) - y(t)$$

and the symbolic representation is:



We can therefore add a feedback loop to the heating system problem (Figure 8.7):

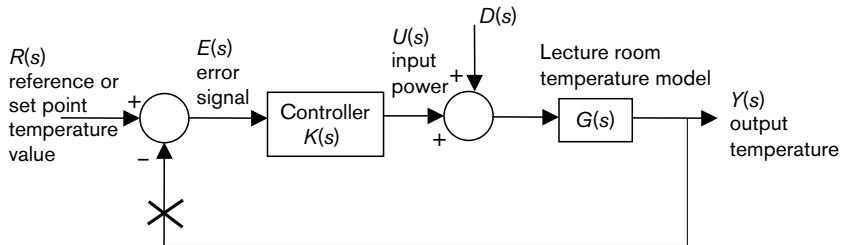
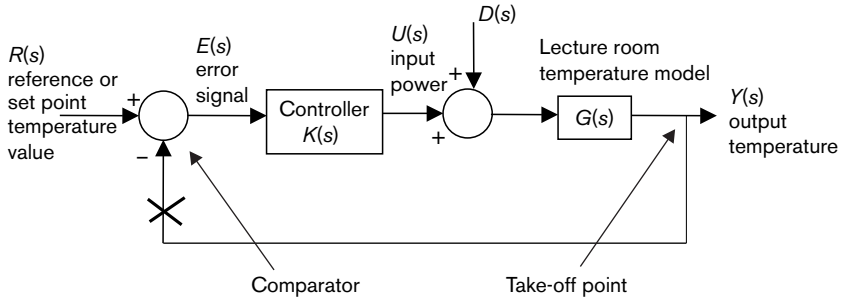


Figure 8.7 Feedback loop added to heating system.

- Problem**
1. Identify the signal take-off point and the comparator in the closed-loop system of Figure 8.7.
 2. Assuming the feedback loop is broken at 'X', identify the forward transfer functions, $G_a(s)$ and $G_b(s)$, given by:

- (a) from $R(s)$ to $Y(s)$: $Y(s) = G_a(s)R(s)$
- (b) from $D(s)$ to $Y(s)$: $Y(s) = G_b(s)D(s)$

Solution



- (a) $G_a(s) = G(s)K(s)$
- (b) $G_b(s) = G(s)$

In this problem we met components of the feedback loop, such as the take-off point, the comparator, the reference input and the controller. However, to represent a more general closed-loop system we introduce some additional blocks. This is illustrated next.

8.3 Practical closed-loop control systems

A practical closed-loop control system often has many components. We look at the example of a ship’s autopilot control system (Figure 8.8).

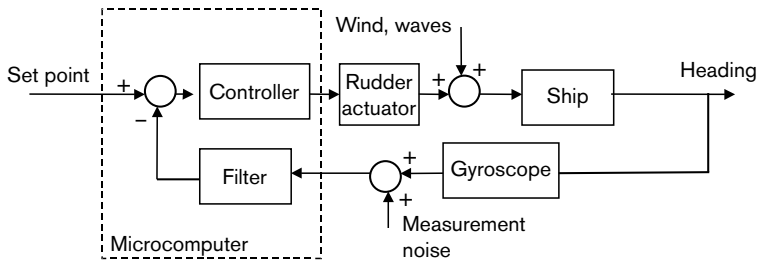


Figure 8.8 Ship autopilot feedback system.

In the ship autopilot example, the **process** (or **plant** or **system**) is the ship itself. The **measurement** system is the gyroscope, which gives a measurement of the ship’s heading. This is subtracted from the reference heading to obtain an error signal which is fed to the **controller**. The controller will send a signal to the **actuator** to change the rudder angle and hence the ship’s direction. It is the change of the rudder direction which is the input to the process. The disturbances acting on the ship which could lead to heading errors are wind and wave forces from the sea environment. We note that the measurement system and/or the computer software may include filters to remove such unwanted low- or high-frequency signals.

Many control systems have similar components and can be written in a similar block diagram. This more general form is given in Figure 8.9.

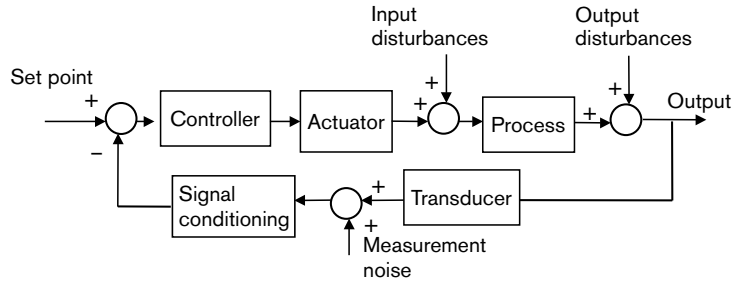


Figure 8.9 General feedback system and loop components.

We can represent each block by a transfer function to give a general system block diagram using Laplace transform notation (Figure 8.10).

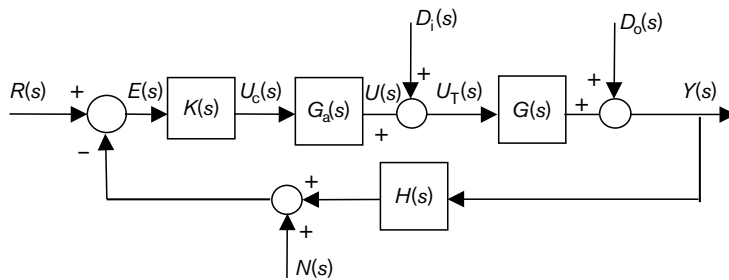


Figure 8.10 General system block diagram.

We can identify the following signals and system components of the closed loop of Figure 8.10.

■ Signals:

- $R(s)$: reference signal or set point
- $E(s)$: process error signal
- $U_c(s)$: controller output signal
- $U(s)$: actuator output signal
- $U_T(s)$: process input signal (actuator input plus disturbance)
- $D_i(s)$: input disturbance signal
- $D_o(s)$: output disturbance signal
- $Y(s)$: process output
- $N(s)$: measurement noise signal

■ Systems:

- $K(s)$: controller transfer function
- $G_a(s)$: actuator system
- $G(s)$: plant or process model
- $H(s)$: measurement system, comprising transducers and signal conditioning units

8.3.1 Input-output disturbances

There are different forms of disturbances that can act on a system: for example, *supply* or *load* disturbances. These are described more fully in Chapter 11. However, we note that

an input disturbance to a stable process may be *moved* to the process output. We show these alternative forms in Figure 8.11.

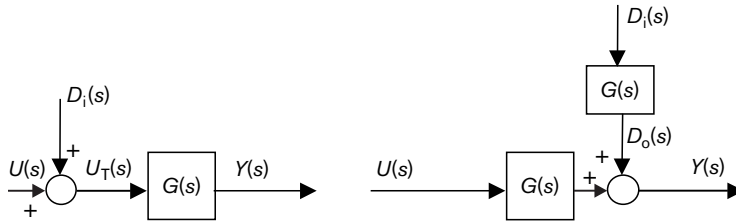


Figure 8.11 Moving a disturbance representation from input to output.

The systems we will study may contain some or all of the signal and system components from the general system block diagram. When we learn about systems and their stability, we often reduce the number of component blocks in order to make the analysis more simple.

8.4 Block diagram manipulation

When we analyse closed-loop systems we still use the input–output relation of a system and we must therefore develop the block diagram reduction method for this closed-loop system. We will practise our analyses on various system types, but, in particular, ones with the structure of Figure 8.12.

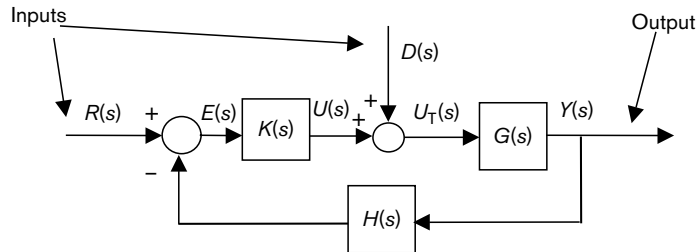


Figure 8.12 General feedback loop.

The system has two inputs – a reference signal and a disturbance signal – and one output. We note that the reference and disturbance signals are independent of each other (they will not affect each other), but that both signals will have an effect on the output signal. Therefore we view the block diagram as two single-input single-output processes which are related by the blocks within the closed-loop process, as shown in Figure 8.13.

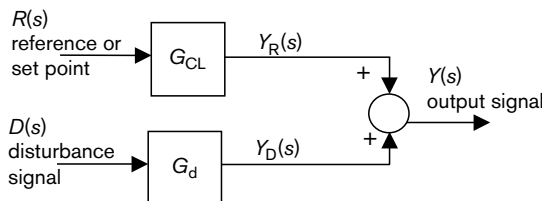


Figure 8.13 Output signal formed from two transformed input signals.

We use the superposition principle (Chapter 2), which states that the total response of the system, $Y(s)$, is the summation of responses for two cases:

1. Input signal $R(s)$, with $D(s) = 0$
2. Input signal $D(s)$, with $R(s) = 0$

that is,

$$\begin{aligned} Y(s) &= Y_R(s) + Y_D(s) \\ &= G_{CL}(s) R(s) + G_d(s) D(s) \end{aligned}$$

The problem we now face is how to determine the transfer functions $G_{CL}(s)$ and $G_d(s)$, given that we have a closed-loop process.

8.4.1 Closed-loop block diagram reduction

The closed-loop description will take the form

$$Y_R(s) = G_{CL}(s)R(s)$$

$$Y_D(s) = G_d(s)D(s)$$

where $G_{CL}(s)$ represents the closed-loop relationship between the reference input and the output signal and $G_d(s)$ represents the closed-loop relationship between the disturbance input and the output signal

We can form the closed-loop transfer functions in two ways:

- A:** By following the signals round the loop and developing a series of equations
- B:** By using a shorthand formula, which is a quick way of forming the closed-loop transfer function

A: Closed-loop transfer function: signal equations

We can formalise the procedure for finding the closed-loop transfer function as:

1. Start at the output signal whose response we are interested in.
2. Working from right to left (anticlockwise), write down the equations relating the input and outputs for each process block or each summation sign as you meet them.
3. Continue round the loop until your equations include the signal you started with.
4. Gather all terms in the output signal to the left-hand side of the equation and all terms involving any inputs to the right-hand side.
5. By algebraic manipulation, determine the final system transfer function equation. For example, for one output and two inputs, you would find

$$\begin{aligned} \text{Output signal} &= [\text{transfer function 1}] \times \text{input signal 1} \\ &\quad + [\text{transfer function 2}] \times \text{input signal 2} \end{aligned}$$

Example: Signal equations

Calculate the closed-loop transfer functions from both $R(s)$ and $D(s)$ to $Y(s)$ for the system shown in Figure 8.14.

For brevity, in the following analysis we have not shown the dependence on the Laplace variable s .

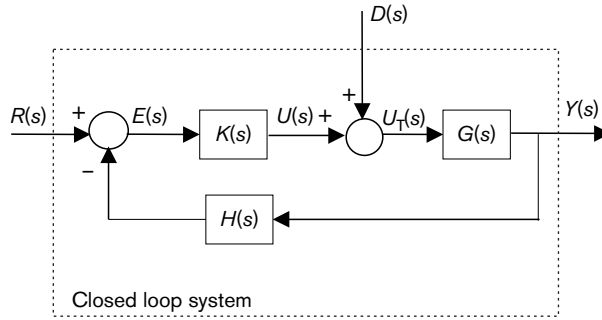


Figure 8.14 General closed-loop block diagram.

Starting from the output signal $Y(s)$:

From the block diagram: $Y = GU_T$

The combined process and disturbance signal is: $U_T = KE + D$

From the comparator we can deduce that: $E = R - HY$

We now have an equation containing Y , the variable we started with, so we can begin elimination.

Combining the equations together, we find: $Y = G(KE + D)$

$$Y = GK(R - HY) + GD$$

Gathering terms in Y to the left-hand side: $(1 + GKH)Y = GKR + GD$

$$\text{or } Y = \frac{GK}{(1+GKH)}R + \frac{G}{(1+GKH)}D$$

This can be written as: $Y = G_{CL}R + G_dD$

where $G_{CL} = GK/(1 + GKH)$ represents the closed-loop transfer function between the input R and the output Y and $G_d = G/(1 + GKH)$ represents the closed-loop transfer function between the disturbance, D , and the output, Y .

There are two transfer functions, representing how the reference signal is passed to the output, $Y(s)$, and how the disturbance signal is passed to the output:

$$Y(s) = Y_R(s) + Y_D(s)$$

where

$$Y_R(s) = \frac{G(s)K(s)}{1+G(s)K(s)H(s)}R(s) = G_{CL}(s)R(s)$$

and

$$Y_D(s) = \frac{G(s)}{1+G(s)K(s)H(s)}D(s) = G_d(s)D(s)$$

We might think that we could force the transfer function analysis to give one overall transfer function for this process. However, this is not possible since the transfer functions relate how the output signal is affected by *each* input. Since there are two input

signals in this diagram, there should be two transfer functions showing the effect of each input signal on the output.

We could equally well have determined the closed-loop transfer function from $R(s)$ to $Y(s)$ by letting $D(s)$ be zero in the above signal equations. This would have simplified the algebra and would have resulted in only one transfer function: $R(s)$ to $Y(s)$. Likewise, we could evaluate the closed-loop transfer from $D(s)$ to $Y(s)$ by letting $R(s)$ be zero in our signal equations. Again, this would simplify the algebra and produce the transfer function we required. (However, although $R(s) = 0$, the negative sign at the summation point remains within the loop. This is important, as the negative sign is the indication of negative feedback.)

By *closing the loop*, the transfer function from input to output has been altered; the closed-loop transfer function need not resemble the system transfer function, $G(s)$. This is desirable since it is through this ability to modify the input–output relationship that we can change (for the better!) the system performance. Control systems use a feedback loop in conjunction with a controller to provide the desired system behaviour.

Remark

In our examples, multiplication of transfer functions G , H and K would give the same final transfer function, no matter what order we multiplied the three together, since they all represent single-input single-output blocks. However, this would not be true if we were dealing with multivariable systems where the systems can be described using *transfer function matrices*. As with any algebra involving matrices, the order is important. The following provides an example of manipulation of transfer function matrices.

Example: Transfer function multiplication: push-through rule

Firstly we note the following *push-through* rule:

$$G_1(I + G_2G_1)^{-1} = (I + G_1G_2)^{-1}G_1$$

In this identity, the matrix, or transfer function, G_1 has been moved from one side of the inverse term to the other, and in doing so, the order of G_1 and G_2 has been reversed.

We can apply this to the development of the closed-loop transfer function from input R to output Y . We note that the development of G_{CL} included the equation:

$$(1 + GKH)Y = GKR$$

which gave

$$Y = (1 + GKH)^{-1}GKR = \frac{GK}{1+GKH} = G_{CL}R$$

By applying the identity above, we can write this equation as:

$$Y = GK(1 + HGK)^{-1}R = \frac{GK}{1+HGK} = G_{CL}R$$

Remark

Unity feedback systems

We often assume that we have a *unity feedback* system; that is, a closed-loop system such as that in Figure 8.14, where the transfer function $H(s) = 1$. In this case (and, for example, by letting $D(s)$ be zero), we can write

$$Y(s) = G(s)K(s)E(s)$$

and

$$E(s) = R(s) - H(s)Y(s) = R(s) - (1 \cdot Y(s)) = R(s) - Y(s)$$

Therefore

$$Y(s) = G(s)K(s)[R(s) - Y(s)]$$

which gives ultimately

$$Y(s) = \frac{G(s)K(s)}{1 + G(s)K(s)} R(s)$$

B: Closed-loop transfer function: shorthand formula

First we define some common transfer functions, referring to Figure 8.15.

- *Forward path*: the transfer function relating the input signal to the output signal as though the feedback path was not included
- *Feedback path*: the transfer function connecting the output signal to the comparator
- *Open loop*: the transfer function from the input signal to the loop, round the loop and back to the input signal

Key result: Closed loop transfer function

$$\text{Closed-loop transfer function: } \frac{\text{forward path}}{1 + \text{open loop transfer function}}$$

The above closed-loop transfer function assumes that there is negative feedback within the closed loop.

We can form a procedure for determining the shorthand form:

1. Break the loop where the feedback signal returns to the summation sign at the input to the loop.
2. Calculate the open-loop transfer function from the point at which the loop is broken back (anticlockwise) round the loop to the summation sign.
3. Calculate the forward path transfer function from the input to the output required (remembering to work from right to left when multiplying transfer functions).
4. Form the closed-loop transfer function as:

$$\text{closed-loop transfer function: } \frac{\text{forward path}}{1 + \text{open loop transfer function}}$$

Skill section

Example 1 We apply the block diagram reduction methods to Figure 8.15 to evaluate the following transfer function for the input given by $R(s)$ and the output given by $Y(s)$:

- (a) The forward, feedback and open loop transfer functions, assuming the loop is broken at X.
- (b) The closed-loop transfer function.

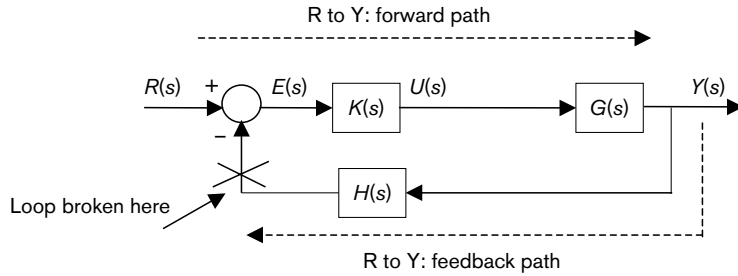


Figure 8.15 Closed-loop transfer function calculations for Example 1.

The transfer functions are given by:

Forward path:	$G(s)K(s)$
Feedback path:	$H(s)$
Open-loop transfer function:	$H(s)G(s)K(s)$
Closed-loop transfer function:	$G_{CL}(s) = \frac{G(s)K(s)}{1+H(s)G(s)K(s)}$

Example 2 For the input $D(s)$ and the output $Y(s)$ in Figure 8.16, determine the following transfer functions:

1. The forward, feedback and open loop transfer functions, assuming the loop is broken at X.
2. The closed-loop transfer function.

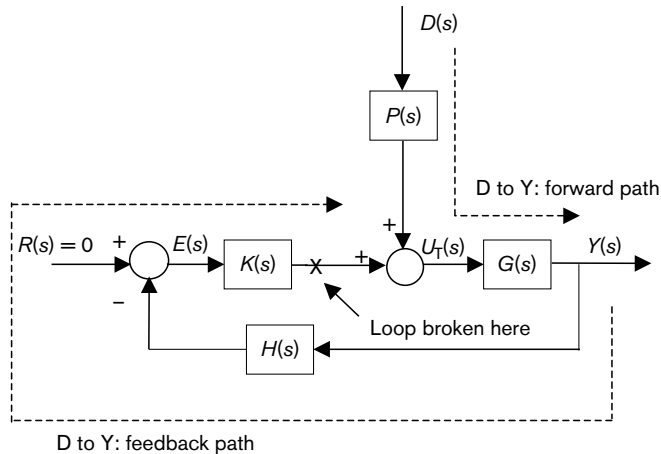


Figure 8.16 Closed-loop transfer function calculations for Example 2.

Forward path:	$G(s)P(s)$
Feedback path:	$K(s)H(s)$
Open-loop transfer function:	$K(s)H(s)G(s)$

Closed-loop transfer function:
$$G_{CL}(s) = \frac{G(s)P(s)}{1+K(s)H(s)G(s)} = \frac{G(s)P(s)}{1+G(s)K(s)H(s)}$$

We note that the forward path from the input $D(s)$ to the output $Y(s)$ passes through only $G(s)P(s)$ and that $P(s)$ does not appear in the open loop transfer function since $P(s)$ is outside the loop. The open loop transfer function is the same as the open loop transfer function in the previous example.

Example: Sensitivity transfer function $S(s)$

For the input $D(s)$ and the output $Y(s)$ in Figure 8.17, determine the following transfer functions:

1. The forward, feedback and open loop transfer functions, assuming the loop is broken at X.
2. The closed-loop transfer function

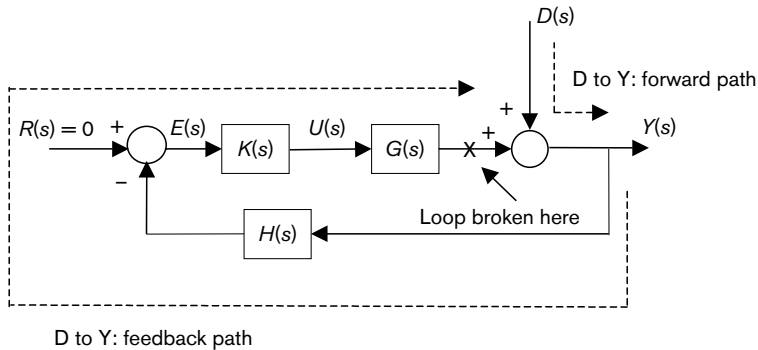


Figure 8.17 Transfer function calculations for sensitivity transfer function.

- Forward path: 1
- Feedback path: $G(s)K(s)H(s)$
- Open-loop transfer function: $G(s)K(s)H(s)$
- Closed-loop transfer function: $\frac{1}{1+G(s)K(s)H(s)}$ (sensitivity transfer function, $S(s)$)

This transfer function has a particular significance when we examine issues of disturbance rejection, particularly in the frequency domain (Chapter 15). It is known as the sensitivity transfer function and denoted by $S(s)$.

Remark

Algebra

We have performed the closed-loop block diagram reduction with symbols for the transfer functions. These symbols will represent the Laplace transform descriptions of the systems. When we replace the symbols with Laplace transforms we will need to be careful with the algebra to find the final system transfer function. We will often find control systems with closed-loop expressions such as

$$G_{CL}(s) = \frac{G}{1+G} = \frac{\frac{6}{3s+1}}{1+\frac{6}{3s+1}}$$

(or often much worse!). We need to be able to manipulate these expressions algebraically:

$$G_{CL}(s) = \frac{6}{3s+1} \frac{1}{1 + \frac{6}{3s+1}} = \frac{6}{3s+1} \frac{1}{\frac{3s+1+6}{3s+1}} = \frac{6}{3s+1+6} = \frac{6}{3s+7}$$

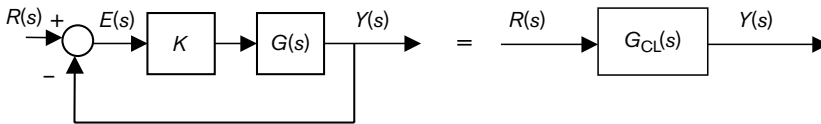
8.5 Feedback changes the closed-loop performance

We now consider an example in which we show that including the feedback loop gives us the freedom to choose a controller to change the speed of the system response.

Example Consider the following example of a process which has a time constant of 4 seconds and a system gain of 3.

$$Y(s) = G(s)U(s) \quad G(s) = \frac{3}{4s+1}$$

We would like to add a constant gain controller, K , *in cascade* with the system transfer function in a unity feedback configuration in order to change the speed of response of the system.



The forward transfer function from $R(s)$ to $Y(s)$ is given by $G(s)K = 3K/(4s + 1)$. The open-loop transfer function is also given by $G_{OL}(s) = 3K/(4s + 1)$. These in turn will give a closed-loop transfer function of:

$$G_{CL}(s) = \frac{G(s)K}{1 + G(s)K} = \frac{3K}{4s+1} \frac{1}{1 + \frac{3K}{4s+1}} = \frac{3K}{4s+1} \frac{1}{\left(\frac{4s+1+3K}{4s+1}\right)} = \frac{3K}{4s+1+3K}$$

The time constant of the process has changed. We can see this by putting the closed-loop transfer function in the standard gain–time constant form. Starting with

$$G_{CL}(s) = \frac{3K}{4s+1+3K}$$

we divide by $1 + 3K$ to get a unit constant coefficient on the denominator:

$$G_{CL}(s) = \frac{\frac{3K}{1+3K}}{\frac{4}{1+3K}s + 1}$$

The gain and time constant can be read as:

$$\text{Gain: } \frac{3K}{1+3K}$$

$$\text{Time constant: } \frac{4}{1+3K}$$

By changing the value of the controller K we can change the time constant of the system. For example, if we wished a value of $\tau = 0.5$ seconds:

$$\tau = \frac{4}{1+3K} = 0.5$$

$$\Rightarrow 4 = 0.5 + 1.5K$$

$$\Rightarrow K = 3.5/1.5 = 2.33$$

There are practical limits on the choice of controller K , but these and the formal controller design process are discussed in the later chapters on control design.

M Problem

Standard MATLAB commands can be used to manipulate transfer functions. Using MATLAB and Figure 8.18, do the following exercises.

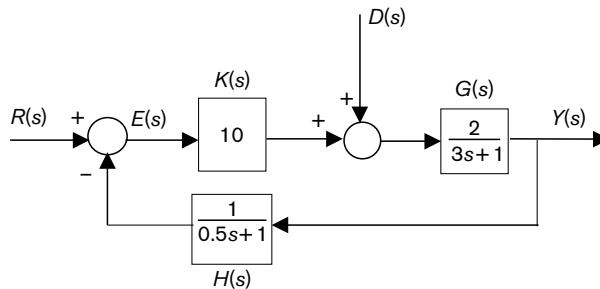


Figure 8.18 Closed-loop system.

- Enter the transfer function $K(s)$, $G(s)$ and $H(s)$.
- Calculate the *forward* transfer function from $R(s)$ to $Y(s)$.
- What is the *forward* transfer function from $D(s)$ to $Y(s)$?
- What is the open-loop transfer function in both cases (b) and (c)?
- Calculate the closed-loop transfer function $G_{CL}(s)$ and $G_d(s)$ using the shorthand formula:

$$\frac{\text{forward path}}{1 + \text{open loop transfer function}}$$

- Use the MATLAB feedback command to verify your answer. (General usage: `g = feedback(forward, feedback)`, where forward and feedback are the appropriate transfer functions for the particular input and output signals.)

Solution (a) `s=tf('s');`

$$k=10; g=4/(2*s+1); h=1/(0.5*s+1);$$

(b) `gfwd = g*k;`

(c) `gfwd2= g;`

- (d) $\text{openloop} = h \cdot g \cdot k$; (and will be the same for both R to Y and D to Y)
- (e) $gc1 = \text{gfwd} / (1 + \text{openloop})$
 $gd = \text{gfwd}2 / (1 + \text{openloop})$
- (f) $gc1 = \text{feedback}(\text{gfwd}, h)$
 $gd = \text{feedback}(\text{gfwd}2, h \cdot k \cdot g)$

What we have learnt

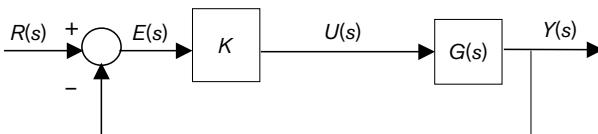
- ✓ Open-loop systems can be used if the relationship between the input and output signals is known exactly and there are no major disturbances which will upset the process.
- ✓ The closed-loop system involves the use of a *comparator* to calculate the error between the reference and the measured output value.
- ✓ By closing the loop, the overall system transfer function will change.
- ✓ The closed-loop transfer function can be calculated from the loop transfer function:
 - by using signal equations, or
 - by using the shorthand formula:

$$\text{Closed-loop transfer function} = \frac{\text{forward path}}{1 + \text{open loop transfer function}}$$

Multiple choice

M8.1 The following diagram shows:

- (a) an open-loop system
- (b) a system with a feedback transfer function equal to zero
- (c) a negative feedback loop
- (d) a positive feedback loop



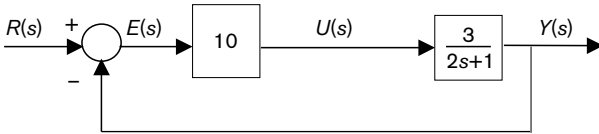
M8.2 An open-loop control system:

- (a) has a controller $K = 1$ in the closed-loop system
- (b) corrects for disturbances acting on the system
- (c) responds well to incorrect system models
- (d) must be altered manually to deal with disturbances

M8.3 The shorthand formula for calculating the closed-loop transfer function for simple systems is:

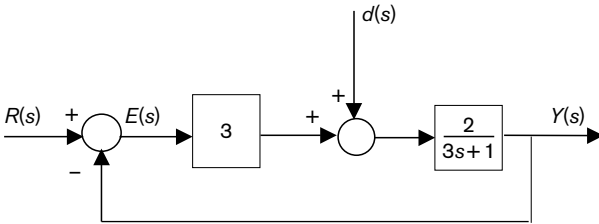
- (a) $\frac{\text{forward}}{1+\text{open loop}}$ (b) $\frac{\text{forward} \times \text{feedback}}{1+\text{open loop}}$
 (c) $\frac{\text{forward}}{1+\text{forward}}$ (d) $\frac{\text{loop}}{1+\text{open loop}}$

M8.4 The forward, feedback and closed-loop transfer function for the following system are:



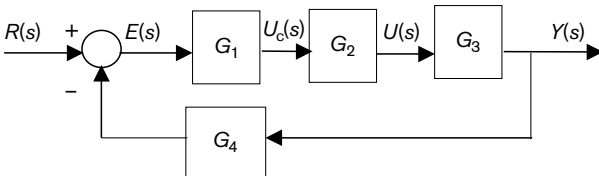
- (a) $\frac{3}{2s+1}$, 0, $\frac{30}{2s+31}$ (b) $\frac{30}{2s+1}$, 1, $\frac{30}{2s+30}$
 (c) $\frac{30}{2s+1}$, 1, $\frac{30}{2s+31}$ (d) $\frac{30}{2s+1}$, 0, $\frac{30}{2s+31}$

M8.5 The transfer function from $d(s)$ to $Y(s)$ is



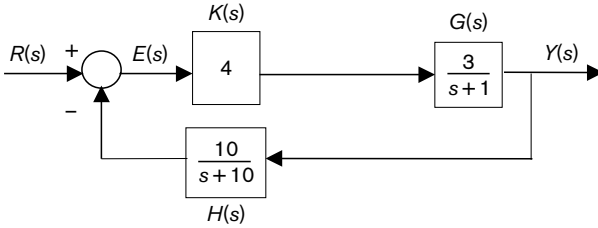
- (a) $\frac{2}{3s+7}$ (b) $\frac{2}{3s+1}$
 (c) $\frac{6}{3s+7}$ (d) $\frac{2}{3s+6}$

M8.6 In the following typical control system block diagram, which block would represent the controller unit?



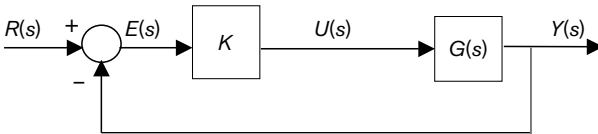
- (a) G_1
 (b) G_2
 (c) G_3
 (d) G_4

M8.7 What is the characteristic equation of the following closed-loop system?



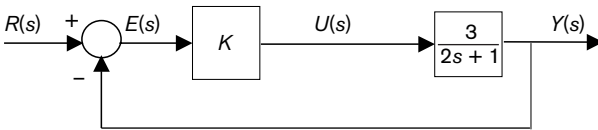
- (a) $s^2 + 11s + 10$
- (b) $s^2 + 11s + 130$
- (c) $s^2 + 10s + 120$
- (d) $s^2 + 10s + 12$

M8.8 In a typical control loop, the aim is to control:



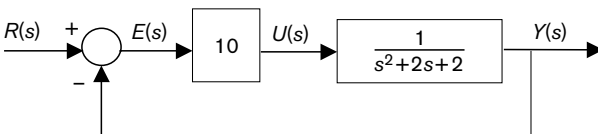
- (a) $R(s)$
- (b) $E(s)$
- (c) $U(s)$
- (d) $Y(s)$

M8.9 For what values of K is the time constant of the closed-loop system less than 0.2 seconds?



- (a) $K > 3$
- (b) $K > 5$
- (c) $K > 7$
- (d) $K > 9$

M8.10 Which of the following describes the step response of this closed-loop system?



- (a) underdamped
- (b) critically damped
- (c) overdamped
- (d) the output does not reach a steady state value

Questions: practical skills

Q8.1 (i) Draw symbolic summation diagrams for the following:

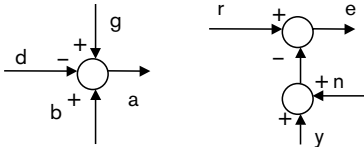
(a) $z(t) = a(t) + b(t)$

(b) $z(t) = -a(t) + b(t) - c(t)$

(c) $\alpha(t) = r(t) + s(t) + \beta(t)$

$\beta(t) = a(t) - b(t)$

(ii) Define the operations represented by the following summation diagrams:

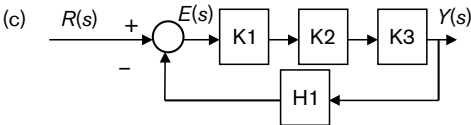
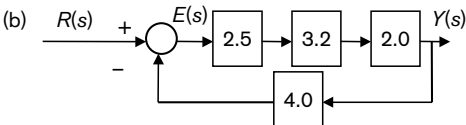
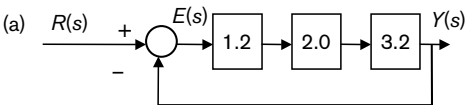


Q8.2 For the following block diagrams, determine the following:

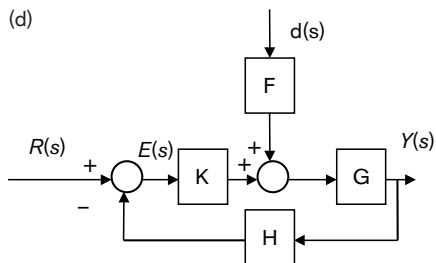
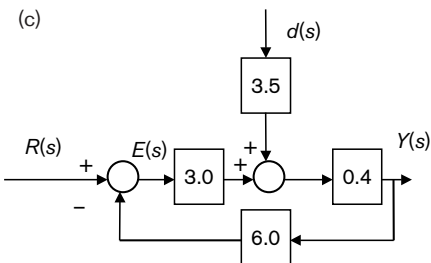
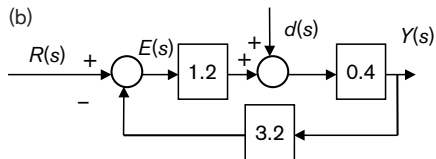
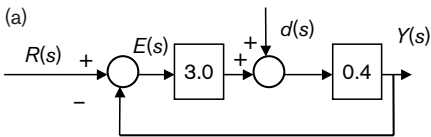
(i) the transfer function of the *forward path*

(ii) the transfer function of the *feedback path*

(iii) the closed-loop transfer function; that is, the transfer function from $R(s)$ to $Y(s)$



Q8.3 For the following block diagrams, determine the forward, feedback and closed-loop transfer functions from $R(s)$ to $Y(s)$. Determine also the closed loop transfer function from the disturbance input $d(s)$ to the output $Y(s)$.

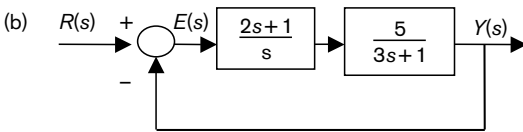
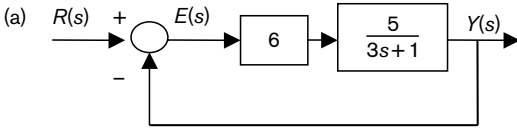


Q8.4 For the following systems evaluate:

$$G_{CL}(s), \text{ where } Y(s) = G_{CL}(s)R(s)$$

and

$$G_E(s), \text{ where } E(s) = G_E(s)R(s)$$



In each case (a) and (b), what do you notice about the denominator and numerator of each transfer function?

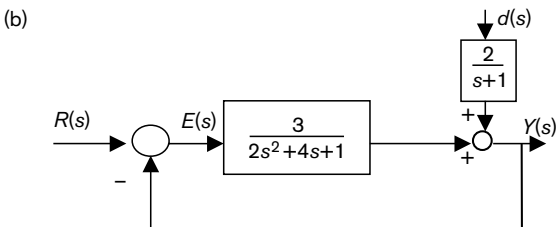
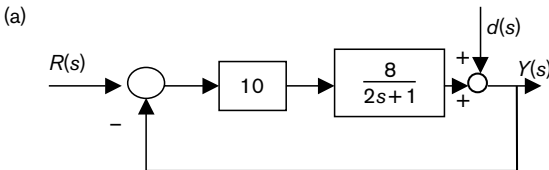
Q8.5 For the following systems evaluate:

$$G_{CL}(s), \text{ where } Y(s) = G_{CL}(s)R(s)$$

$$G_E(s), \text{ where } E(s) = G_E(s)R(s)$$

and

$$G_d(s), \text{ where } d(s) = G_d(s)R(s)$$



In each case (a) and (b), what do you notice about the denominator and numerator of each transfer function?

Problems

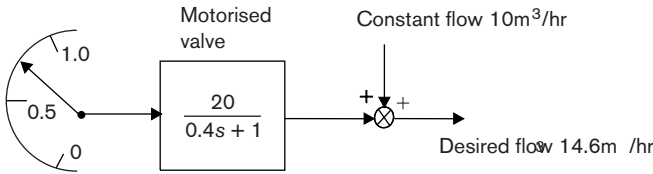
P8.1. Liquid flows into a tank whose height is due to be controlled. The input flow is regulated by a valve. A flow of $3 \text{ m}^3/\text{s}$ is produced for a 1 mm change in valve position. The tank is such that an input flow of $1 \text{ m}^3/\text{s}$ produces a liquid height of 2 m .

218 Feedback improves system performance

(a) Sketch the block diagram of the system.

(b) There is a leak from the connecting pipes of $0.02 \text{ m}^3/\text{s}$. Re-draw your block diagram showing where this would appear.

P8.2 An open-loop control system requires an operator to set a motorised valve setting, c , where $0 < c < 1$, so that two fluids are mixed together. One stream has a constant flow of $10 \text{ m}^3/\text{hr}$, and the outflowing stream should be $14.6 \text{ m}^3/\text{hr}$. The configuration is shown below, where motorised valve dynamics are given.

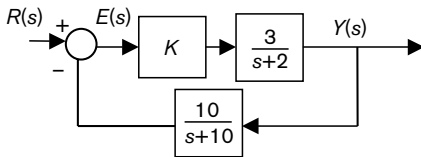


Calculate the value of the valve setting to achieve the desired steady state outflow of $14.6 \text{ m}^3/\text{hr}$.

P8.3 For a first-order process plant with a time constant of 5 and gain of 2, determine a controller gain K in a unity feedback system such that the closed-loop time constant is $\tau_{CL} = 0.5$. Sketch the open and closed-loop responses to a unit step, noting carefully the gain and time constant of both systems.

P8.4 In the previous problem, let $K(s) = K/s$ and examine what happens to the roots of the characteristic equation for varying values of K . Comment on the performance for a step input for $K = 0.01$ and $K = 1$.

P8.5 The system shown represents a process control plant with an output measurement transducer



Find the range of values of K such that the roots of the characteristic equation are

(a) overdamped

(b) underdamped

9

Design specifications on system time response

What are the steady state and transient parts of the response?



Steady state response

Error transfer function



Error design specification, Final Value Theorem, e_{ss}

S Simulink test

Transient response

Percentage overshoot: OS(%)

Maximum peak value: y_{max}

Settling time: $t_s(5\%), t_s(2\%)$

Rise time: $t_r(10\%, 90\%), t_r(0\%, 100\%)$

S Simulink test

Disturbance rejection

Peak disturbance: D_{peak}

Disturbance settling time: D_{ts}

S Simulink test



Gaining confidence



Help?



Going deeper



Skill section



Time to read

We can use the modelling and transfer function analysis which we have learnt in previous chapters when we go on to design controllers. However, before we look at control design, we need to specify what performance we require from the controlled system. Do we wish the response to move rapidly to its set point? Do we accept that the system may take some time to settle down? Do we wish to react quickly to disturbances on the process? These requirements lead to a set of *design specifications* which can be expressed in terms of a number of parameters which are related to the process's step response.

This chapter presents the common time domain specifications: firstly the steady state error, then common specifications on the output for changes in reference input. We also look at the specification of the system for when there is a change in disturbance. Although the actual change in disturbance input may be unknown in practice, we often design systems to cope with a certain level of disturbance input.

Learning objectives

- To calculate the steady state error to a specified input for a system.
- To appreciate that both step changes in reference and disturbances may produce steady state offsets.
- To calculate the specifications for transient performance: the percentage overshoot (OS(%)), the settling time ($t_s(5\%)$ or $t_s(2\%)$) and the rise time ($t_r(10\%,90\%)$ or $t_r(0\%,100\%)$).
- To calculate the specifications for transient performance after a change in disturbance: peak change resulting from the disturbance (D_{peak} , $D_{\text{peak}}(\%)$) and settling time due to a change in disturbance ($D_{ts}(5\%)$, $D_{ts}(2\%)$).

9.1 Design specifications: steady state and transient behaviour

We often use controllers to stabilise and change the performance of a system. In doing so, we need to specify the behaviour we would actually like to see from the system. This is done by providing a set of **design specifications** for the inputs and outputs. These are based on the desired steady state behaviour of the system (steady state error from a reference) and the transient performance of the system (Figure 9.1). If we recall from Chapter 8 that the error is given by $e(t) = r(t) - y(t)$ where $r(t)$ is the reference level or set point and $y(t)$ is the output, we can determine from Figure 9.1 that the steady state level of the output, y_{ss} , is 0.8. The resulting steady state error will be given by

$$e_{ss} = r_{ss} - y_{ss} = 1 - 0.8 = 0.2$$

where the steady state reference level is 1.

Example: Robot arm/cutting device

A robot arm has at its extremity a cutting tool for speed cutting of leather for shoes. It is required to cut a variety of pieces of leather and to cut at varying speeds depending on the complexity of the reference shape. In this situation it is important for the robot arm to satisfy the following requirements:

- a quick response to new commands
- not too much overshoot (cutting past the required outline), since this would cause wastage

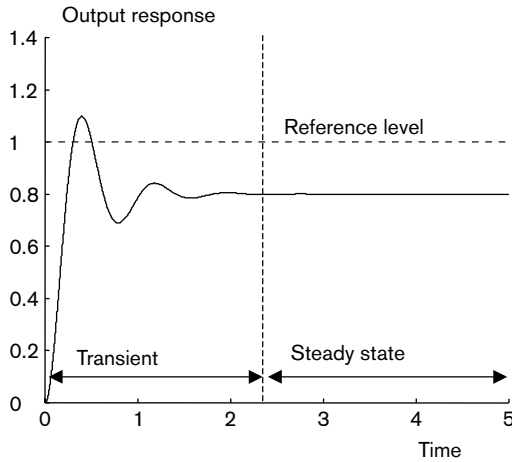


Figure 9.1 Steady state and transient parts of step response.

- quick to settle to a new cutting speed or line
- no steady state offset when the arm moves from one position to another

These requirements on the steady state and transient performance can be defined more rigorously using criteria from the system's output performance in response to changes in reference signals or disturbance inputs.

9.2 Steady state performance

Key result: Steady state error

Just as we calculated the final output value y_{ss} , we can use the Final Value Theorem to work out the final error value, e_{ss} , for stable systems:

$$e_{ss} = \lim_{t \rightarrow \infty} e(t) = \lim_{s \rightarrow 0} sE(s)$$

Skill section

Error transfer function evaluation

We wish to find the transfer function that relates the error signal $E(s)$ in Figure 9.2 to the input $R(s)$ and the disturbance $D(s)$. We can do this in two ways:

A work out the error transfer function directly in terms of the input signal: for example, $E(s) = G_e R(s)$

B given the transfer function relationship $Y(s) = G_{CL}(s) R(s)$, calculate $E(s) = R(s) - Y(s) = R(s) - G_{CL}(s)R(s) = [1 - G_{CL}(s)]R(s)$

To make the manipulations easier to read we have removed the dependency on the Laplace variable s .

A: Transfer function by calculating $E(s) = G_e(s)R(s)$

We practise our skills in manipulation by considering both the signal equation formulation and the shorthand formula.

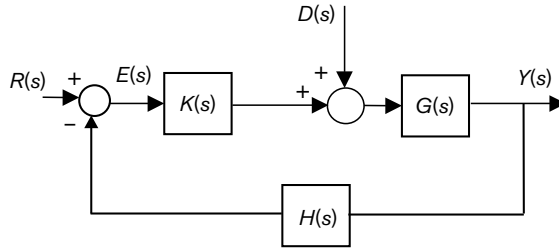


Figure 9.2 Block diagram for error analysis.

1. Transfer function from signal equations

Since we wish $E(s)$ as the output signal, we start the block diagram manipulation at $E(s)$ and work our way round the loop, back to $E(s)$:

$$E = R - HY$$

$$Y = G[KE + D]$$

Combine these to give:

$$E = R - H[GKE + GD]$$

$$E = R - HGKE + HGD$$

$$[1 + HGK]E = R + HGD$$

$$E = \frac{1}{1+HGK} R + \frac{HG}{1+HGK} D$$

Total error: $E = E_R R + E_d D$

2. Shorthand form

(i) Set $D(s)$ to zero: evaluate the transfer function from $R(s)$ to $E(s)$:

forward path (from R to E): 1

open-loop transfer function: HGK

Therefore

$$E = \frac{1}{1+HGK} R$$

(ii) Set $R(s)$ to zero: evaluate the transfer function from $D(s)$ to $E(s)$:

forward path (from D to E): HG

open-loop transfer function: KHG

Therefore

$$E = \frac{HG}{1+KHG} D = \frac{HG}{1+HGK} D$$

B: Transfer function by calculating $E(s) = R(s) - Y(s)$

If we know the closed-loop transfer function,

$$Y(s) = G_{CL}(s)R(s)$$

we can easily calculate the error transfer function. For the above system,

$$G_{CL} = \frac{GK}{1+HGK}$$

and the error transfer function is given by

$$\begin{aligned} E &= R - HY = R - HG_{CL}R = [1 - HG_{CL}]R \\ &= \left(1 - \frac{HGK}{1+HGK}\right)R \\ &= \frac{1+HGK - HGK}{1+HGK}R \end{aligned}$$

9.2.1 Steady state error design specification

We found in Chapter 8 that the output signal, $Y(s)$, can be composed of two components: one from the system's response to a changing reference signal and the other from the system's response to a disturbance input:

$$Y(s) = Y_R(s) + Y_D(s)$$

We can use the Final Value Theorem to work out the steady state level of $y(t)$.

Final Value Theorem

$$\begin{aligned} y_{ss} &= \lim_{t \rightarrow \infty} y(t) = \lim_{s \rightarrow 0} sY(s) \\ &= \lim_{s \rightarrow 0} s(Y_R(s) + Y_D(s)) \\ &= Y_{Rss} + Y_{Dss} \end{aligned}$$

The steady state value has two components: one derived from the transfer function from $R(s)$ to $Y(s)$ and one derived from the transfer function from $D(s)$ to $Y(s)$. We can use these values to specify the requirements on the closed-loop system design. For example, we may wish our steady state error to satisfy one of the following criteria

- (i) Steady state error to a unit step increase in $R(s)$ to be less than 0.1: $e_{ss} < 0.1$
- (ii) Steady state error to a change in disturbance to be zero.

Problem Consider the system shown in Figure 9.3. What are the values of gain K which will achieve the following design specifications?

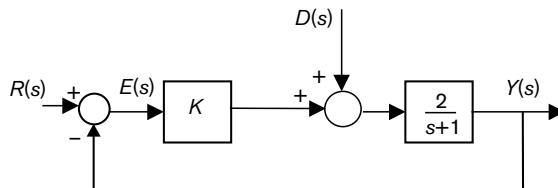


Figure 9.3 Control system for design problem.

- (a) $e_{ss} \leq 0.1$ for a unit step input in $R(s)$
- (b) $e_{ss} \leq 0.2$ for a step change of 2 in the disturbance $D(s)$

Solution (a) We need to evaluate $Y_{R_{ss}}$. The closed-loop transfer function is evaluated using the shorthand formula (and setting $D(s)$ to zero).

$$\text{forward transfer function: } \frac{2K}{s+1}$$

$$\text{open-loop transfer function: } \frac{2K}{s+1}$$

$$\text{closed-loop transfer function: } Y_R(s) = G_{CL}(s)R(s)$$

$$G_{CL}(s) = \frac{\text{forward path}}{1 + \text{open loop transfer function}} = \frac{\frac{2K}{s+1}}{1 + \frac{2K}{s+1}} = \frac{2K}{s+1+2K}$$

Hence

$$\begin{aligned} Y_{R_{ss}} &= \lim_{s \rightarrow 0} sY_R(s) = \lim_{s \rightarrow 0} sG_{CL}(s)R(s) = \lim_{s \rightarrow 0} s \frac{2K}{s+1+2K} \frac{1}{s} \\ &= \frac{2K}{1+2K} \end{aligned}$$

Since the input signal was a unit step ($R_{ss} = 1$), the error will be given by

$$e_{ss} = R_{ss} - Y_{R_{ss}} = 1 - \frac{2K}{1+2K} = \frac{1}{1+2K}$$

The error will therefore become smaller as our controller gain increases. We meet this again as proportional control (Chapter 11). Our design requires

$$\begin{aligned} e_{ss} &\leq 0.1 \\ \Rightarrow \frac{1}{1+2K} &\leq 0.1 \\ \Rightarrow K &\geq 4.5 \end{aligned}$$

A value of K greater than 4.5 will satisfy the design criterion. However, in practice the actual equipment has hard limits and this will determine the maximum amount of controller gain that can be applied.

(b) We need to evaluate $Y_{D_{ss}}$. The closed-loop transfer function is evaluated using the shorthand formula (and setting $R(s)$ to zero for this calculation).

$$\text{forward transfer function: } \frac{2}{s+1}$$

$$\text{open-loop transfer function: } \frac{2K}{s+1}$$

$$\text{closed-loop transfer function: } Y_R(s) = G_d(s)D(s)$$

$$G_d(s) = \frac{\text{forward path}}{1 + \text{open loop transfer function}} = \frac{\frac{2}{s+1}}{1 + \frac{2K}{s+1}} = \frac{2}{s+1+2K}$$

Hence

$$\begin{aligned} Y_{Dss} &= \lim_{s \rightarrow 0} sY_D(s) = \lim_{s \rightarrow 0} sG_d(s)D(s) \\ &= \lim_{s \rightarrow 0} s \frac{2}{s+1+2K} \frac{2}{s} \\ &= \frac{4}{1+2K} \end{aligned}$$

Since the input signal $r(t) = 0$, R_{ss} will be zero, and the error will be given by

$$|e_{ss}| = |0 - Y_{Dss}| = \left| 0 - \frac{4}{1+2K} \right| = \left| \frac{-4}{1+2K} \right| = \frac{4}{1+2K}$$

Our design requires

$$\begin{aligned} e_{ss} &\leq 0.2 \\ \Rightarrow \frac{4}{1+2K} &\leq 0.2 \\ \Rightarrow K &\geq 9.5 \end{aligned}$$

A value of $K \geq 9.5$ will satisfy the design criterion. We note that we cannot always satisfy design criteria on the disturbance and reference signal simultaneously. This will become more apparent in Chapter 11.

Problem Given the system in Figure 9.4, what would be the value of the controller gain K to achieve a steady state error of $e_{ss} \leq 0.2$ to a unit step input?

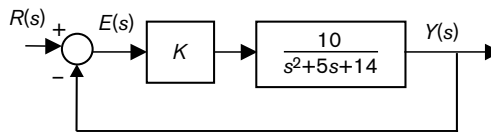


Figure 9.4 Control system for second-order design problem.

Solution The closed-loop transfer function is given by:

$$Y(s) = \frac{10K}{s^2+5s+14+10K} R(s)$$

Therefore the steady state output to a unit step input can be calculated as:

$$\begin{aligned} y_{ss} &= \lim_{s \rightarrow 0} sY(s) = \lim_{s \rightarrow 0} s \frac{10K}{s^2+5s+14+10K} \frac{1}{s} \\ &= \lim_{s \rightarrow 0} \frac{10K}{s^2+5s+14+10K} = \frac{10K}{14+10K} \end{aligned}$$

We asked for a reference value of 1.0; the error in the final value is $e_{ss} = R_{ss} - Y_{ss}$:

$$e_{ss} = 1 - \frac{10K}{14+10K} = \frac{14}{14+10K} \leq 0.2$$

$$K \geq 5.6$$

S We can use a Simulink model (Figure 9.5), to verify these results.

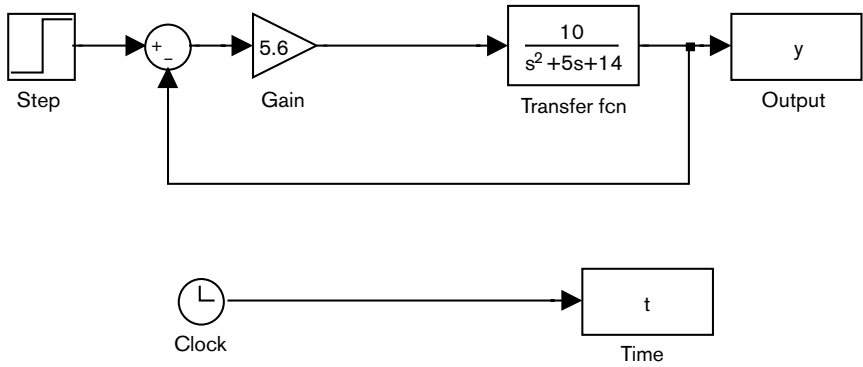


Figure 9.5 Simulink diagram for second-order design problem.

The output response (Figure 9.6) settles down to a steady state value of 0.8, giving an error of 0.2. If we increased the value of the gain, we would notice that the error would decrease. However, the value of $K = 5.6$ satisfies the design specification for this system.

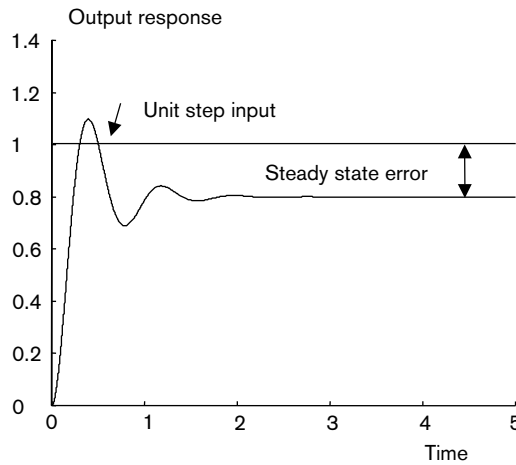


Figure 9.6 Output response for $K = 5.6$.

9.3 Transient performance

Steady state specifications are concerned with the accuracy with which a closed-loop system can track a specified input reference trajectory. Transient performance corresponds to the initial behaviour of the system before it has begun to settle down into its steady state behaviour. Transient performance is usually specified on the basis of the system unit step response.

Time domain performance indices have to be appropriate to all the different type of responses. However, second-order systems are of considerable interest to the control engineer because:

1. Many physical systems can be characterised by a second-order model (even systems of higher order can often be approximated by a second-order system).
2. The mathematics involved in developing design specifications for second-order systems is not too daunting.

Although we could set demands on the second-order parameters of ζ and ω_n , we often set specifications on the time (step) response of the system, and in particular on:

- the system overshoot
- the system settling time
- the system rise time

When we discuss each of these, we indicate whether we are referring to a change in reference input or disturbance input.

9.3.1 System overshoot (change in reference)

In general, the overshoot is an indication of the largest error between the reference input and the output. It is usually given as a percentage.

Key result: System overshoot

$$\text{percent overshoot} = \text{OS} (\%) = \frac{\text{peak value} - \text{final value}}{\text{final value}} \times 100\%$$

On an overdamped system's step response there is no overshoot value. Figure 9.7 shows the overshoot for an underdamped case. The percentage overshoot can be calculated as

$$\text{OS} (\%) = \frac{0.97 - 0.75}{0.75} \times 100 = 29.33\%$$

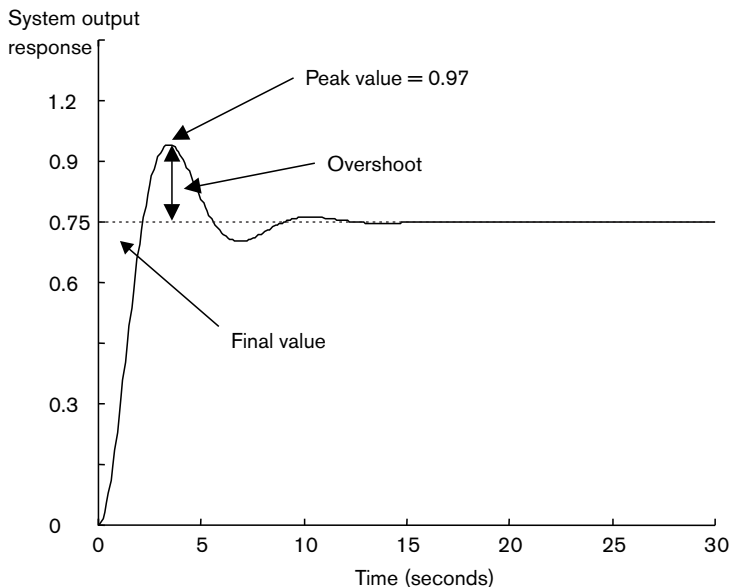


Figure 9.7 Overshoot in system response.

What is a tolerable overshoot depends on the application. It might be as low as 0% for ship turning manoeuvres, 5% for a tracking system or as high as 20% for a position control system where steady state accuracy is considered more important.

For a second-order system where we know the system output equation we can find the maximum and minimum turning points using the techniques of calculus. The maximum overshoot is determined by taking the derivative of $y(t)$ and evaluating the values of t which give turning points of the output response.

Maximum peak value of second-order system

We recall that the unit step response for a second-order system of the following form (note the unity gain):

$$G(s) = \frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2}$$

is

$$y(t) = 1 - \frac{1}{\sqrt{1-\zeta^2}} e^{-\zeta\omega_n t} (\sin \omega_d t + \phi)$$

where $\omega_d = \omega_n \sqrt{1-\zeta^2}$ and $\phi = \tan^{-1}[(\sqrt{1-\zeta^2})/\zeta]$. The derivative of the function is given by

$$\frac{dy}{dt} = -\frac{\omega_n}{\sqrt{1-\zeta^2}} e^{-\zeta\omega_n t} [-\zeta \sin(\omega_d t + \phi) + \sqrt{1-\zeta^2} \cos(\omega_d t + \phi)]$$

Combining sine and cosine expressions gives

$$\frac{dy}{dt} = -\frac{\omega_n}{\sqrt{1-\zeta^2}} e^{-\zeta\omega_n t} \sin \omega_d t$$

The turning point times can be calculated to be the times when $dy/dt = 0$:

$$t_{\max} = k\pi/\omega_d \quad k = 1, 3, 5, \dots$$

$$t_{\min} = k\pi/\omega_d \quad k = 2, 4, 6, \dots$$

By substituting the value of t_{\max} for $k = 1$ in the equation for $y(t)$:

$$y(t) = 1 - \frac{1}{\sqrt{1-\zeta^2}} e^{-\zeta\omega_n t} \sin(\omega_d t + \phi) \quad t = t_{\max} = \pi/\omega_d$$

and with some reduction, we can determine the maximum value of the function $y(t)$:

$$y_{\max} = 1 + e^{-\zeta\pi/\sqrt{1-\zeta^2}}$$

Hence we find that the maximum percentage overshoot is given by

$$OS(\%) = \frac{1 + e^{-\zeta\pi/\sqrt{1-\zeta^2}} - 1}{1} \times 100$$

which reduces to

$$OS(\%) = 100 \exp\left(\frac{-\zeta\pi}{\sqrt{1-\zeta^2}}\right)$$

Remark The percentage overshoot of a step response of a second-order system is a function of ζ only.

9.3.2 Settling time (change in reference)

This is the time required for the system to rise and settle within a given percentage of its final value. Figure 9.8 shows the settling time for the output response to lie within 5% bounds of the final value. Since in this case the final value is '1', the 5% bounds are 1.05 and 0.95. The usual bounds that we refer to are either 5% or 2% and the settling time is denoted by $t_s(5\%)$ or $t_s(2\%)$ respectively.

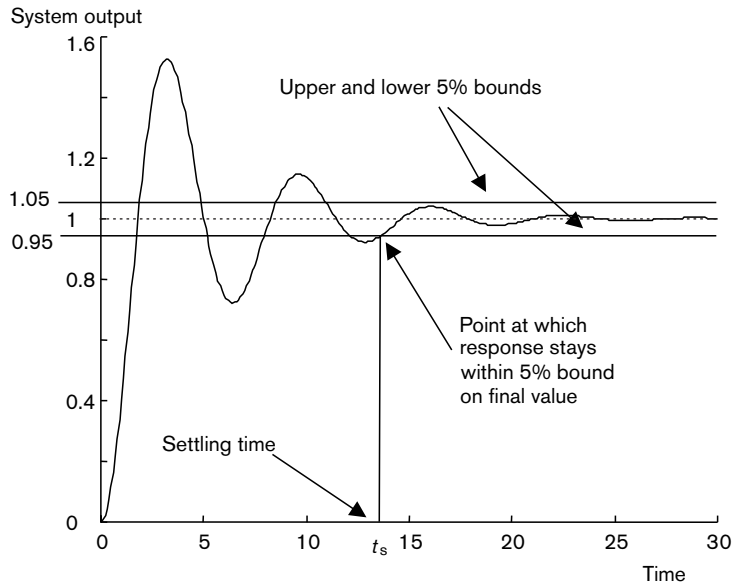


Figure 9.8 Settling time in system response.

We can determine an approximate value for the settling time for a second-order system by examining the system response:

$$y(t) = 1 - \frac{1}{\sqrt{1-\zeta^2}} (\sin \omega_d t + \phi)$$

We note that the output comprises a decaying exponential and a sinusoidal oscillating term. Consequently, the maxima and minima touch an upper and lower envelope:

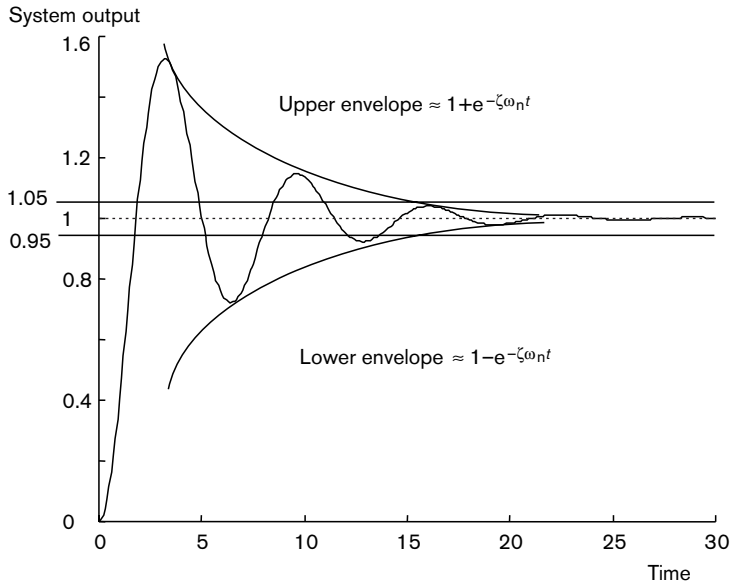
$$y_{\max}(t) = 1 + \frac{1}{\sqrt{1-\zeta^2}} e^{-\zeta\omega_n t}$$

$$y_{\min}(t) = 1 - \frac{1}{\sqrt{1-\zeta^2}} e^{-\zeta\omega_n t}$$

If the damping is small this can be approximated by:

$$y_{\max}(t) = 1 + e^{-\zeta\omega_n t}$$

$$y_{\min}(t) = 1 - e^{-\zeta\omega_n t}$$



This envelope can be used to determine a relationship for the settling time. Let X be the percentage (0.05, 0.02) that we would like to calculate a settling time for. By equating the value of the envelope to the 5% or 2% settling bound, we can find an approximate value for t_s .

$$1 + X = 1 + e^{-\zeta\omega_n t_s}$$

or $\ln(X) = -\zeta\omega_n t_s$, giving

$$t_s = \frac{-\ln(X)}{\zeta\omega_n}$$

By evaluating this for the 2% bound ($X = 0.02$) and 5% bound ($X = 0.05$), we find:

Key result	Settling time: 2% and 5%
Settling time bound	Settling time
2% ($X = 0.02$)	$t_s(2\%) \sim \frac{4}{\zeta\omega_n}$
5% ($X = 0.05$)	$t_s(5\%) \sim \frac{3}{\zeta\omega_n}$

9.3.3 Rise time (change in reference)

The rise time is the time required for the system to change from, say, 10% to 90% of its final value (Figure 9.9). In some textbooks you will find that the definition is ‘0 to 100%’ for underdamped cases and ‘10 to 90%’ for overdamped cases. We will denote the rise time by $t_r(10\%,90\%)$ or $t_r(0\%,100\%)$ for clarity. A short rise time indicates a fast

response, but this may also cause a large peak value. We often find that when we design control systems, the result is a trade-off between the different design specifications.

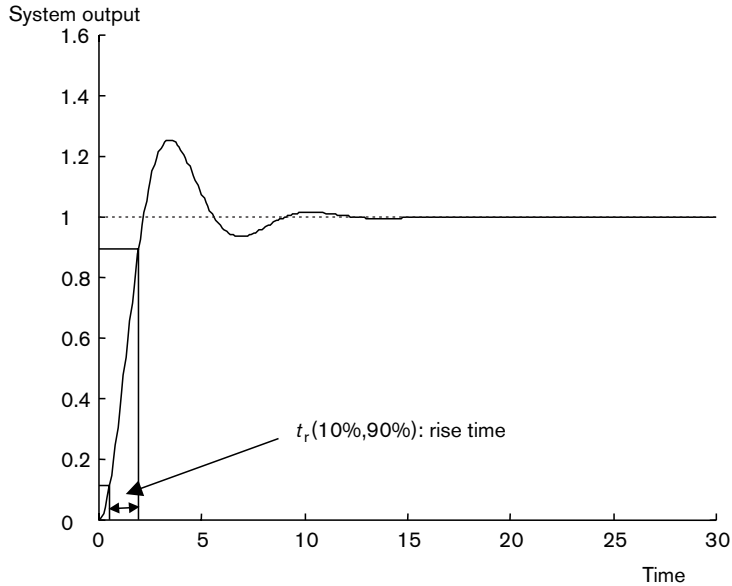


Figure 9.9 Rise time in system step response.

Estimation of transfer function model from time response

Using our knowledge of final value, peak value and overshoot, combined with our knowledge of general second-order systems, we can estimate the transfer function of a system given its step response.

Problem Figure 9.10 shows the step response of a second-order system of the standard form to a unit step. Determine approximately from the plot the gain, damping and natural frequency of the system.

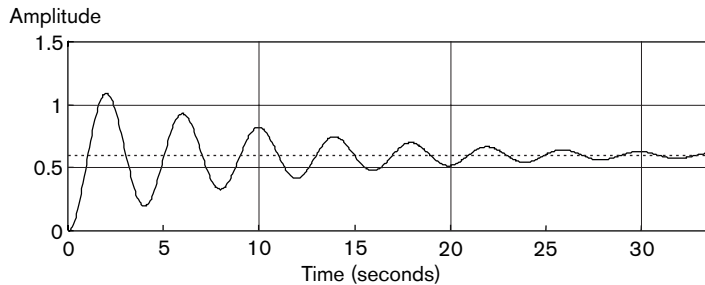


Figure 9.10 Second-order step response.

Solution 1. We know from our knowledge of second-order systems that the gain can be calculated from:

$$y_{ss} = Kr_o$$

where y_{ss} is the steady state value, r_o is the magnitude of the step input and K is the system gain. In this case the final value = 0.6 and the magnitude of r_o is 1. This gives the gain K as 0.6.

2. Since the overshoot is a function of damping ratio only, we can determine the value of ζ by first calculating the percentage overshoot. The formula for the overshoot is given by:

$$\text{OS (\%)} = \frac{\text{peak} - \text{final}}{\text{final}} \times 100 = 100 \exp(-\pi\zeta/\sqrt{1-\zeta^2})$$

From the plot, the peak value is approximately 1.15 and the final value is 0.6. This gives the percentage overshoot as

$$\text{OS (\%)} = \frac{1.15 - 0.6}{0.6} 100 = 91.7\%$$

Therefore we find that

$$0.917 = \exp(-\pi\zeta/\sqrt{1-\zeta^2})$$

or

$$\ln(0.917) = \frac{-\pi\zeta}{\sqrt{1-\zeta^2}}$$

giving

$$(\ln(0.917))^2 = (0.0866)^2 = 0.00750 = \frac{\pi^2\zeta^2}{(1-\zeta^2)}$$

Solving for ζ^2 gives

$$0.00750 (1 - \zeta^2) = \pi^2\zeta^2$$

or

$$\zeta^2 = \frac{0.00750}{0.00750 + \pi^2} = 0.000760$$

$$\zeta = 0.0276$$

3. The natural frequency, ω_n , can be calculated from the frequency of oscillation of the response, ω_d :

$$\omega_d = \omega_n\sqrt{1-\zeta^2}$$

The calculation of ω_d requires us to read the period of one oscillation from the response plot. In this case we take the time between the first and second peaks, t_1 , to be

$$t_1 = 6 - 2 = 4 \text{ seconds}$$

The frequency in hertz (cycles/second) is given by $f = 1/t_1 = 0.25$ Hz. The angular frequency is then:

$$\omega_d = 2\pi f = 2 \times 3.14 \times 0.25 = 1.57 \text{ rad/s}$$

The natural frequency can then be easily calculated using our value for damping, $\zeta = 0.0276$:

$$\omega_n = \omega_d/\sqrt{1-\zeta^2} = 1.57 \times \sqrt{0.99924} \cong 1.57 \text{ rad/s}$$

In this case, the damping is so small that the natural frequency and damped natural frequency take almost the same value.

The resulting transfer function is

$$G(s) = \frac{0.6}{(1/1.57^2)s^2 + [(2 \times 0.027)/1.57]s + 1} = \frac{0.6}{0.406s^2 + 0.034s + 1}$$

9.4 Specifications for disturbance rejection

Let us consider the robot arm/cutting system whose aim is to cut leather for shoes given a reference track to follow (Figure 9.11). The robot cutting arm is driven by a motor and suffers from disturbances in the form of varying qualities of leather causing variations in the process input signal.

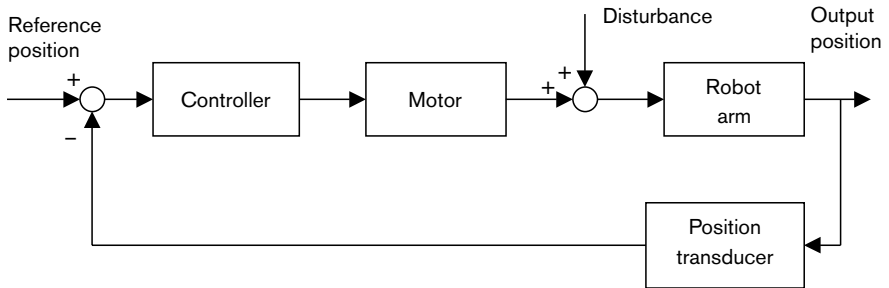


Figure 9.11 Closed-loop system for robot arm position system.

Figure 9.12 shows the output from the position-measuring system. There is an initial change in reference position to an operating level of 10 mm. After 10 minutes we notice a disturbance affecting the system due to the change in quality of thickness of leather. The control system acts to reduce this disturbance, but in this case is not sufficient to remove the steady state error caused by the disturbance. This error is approximately 0.4 mm.

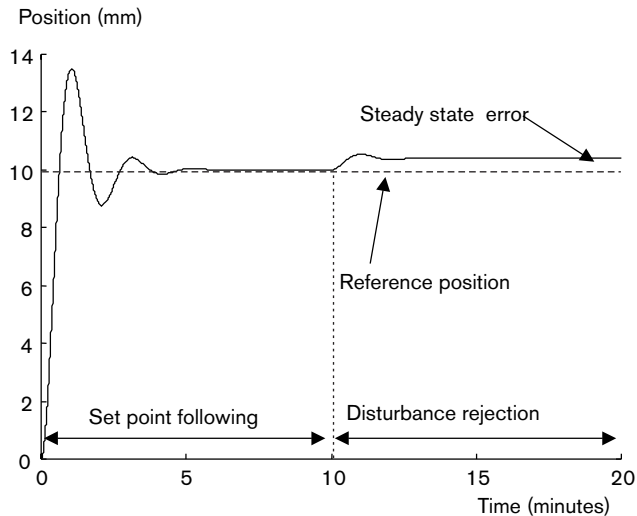


Figure 9.12 Robot arm position.

In this case we can see that a disturbance has caused the output to move from the desired operating level. Very often we will see the disturbance response graph shown as in Figure 9.13. This is the same disturbance as before, but we have shown the output response as the deviation from its steady state (or 'zero') level.

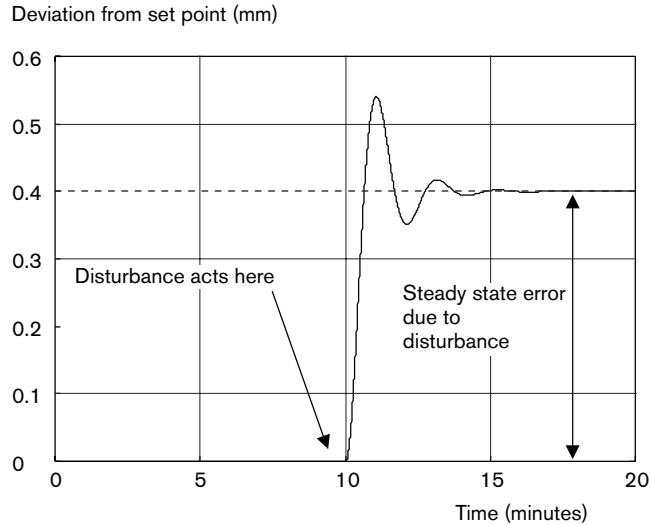


Figure 9.13 Robot position referenced to steady state level.

We often wish to design a control system to reduce the effect of a disturbance on the system output. In these circumstances we are responding to a *disturbance rejection* situation. We may often meet systems which operate at a steady state operating condition. These systems are not required to track a changing reference level and therefore the specifications on the rise time and percentage overshoot are not applicable in these circumstances. However, it is important to design control systems to reduce the effect of the disturbance which may cause the system to move from its operating condition.

What we do here is provide a means of quantifying the effect that the disturbance may have. We assume that the system is operating at a nominal or steady state condition and the disturbance will cause a movement from this steady state level.

9.4.1 Peak disturbance

Key result: Peak disturbance

We can define the peak disturbance as:

$$D_{\text{peak}} = \max_t |y(t) - y_{\text{ss}}|$$

Or, as a percentage of the (non-zero) steady state level, as:

$$D_{\text{peak}} (\%) = \frac{D_{\text{peak}}}{y_{\text{ss}}} \times 100$$

Problem In the robot cutting example, the engineer has a maximum allowable specification for the D_{peak} of 0.5 mm. The output response is shown in Figure 9.12, and is re-plotted as the deviation from the steady state level in Figure 9.13.

- (a) Determine whether the system meets the design specification.
- (b) Also determine the percentage overshoot, $D_{\text{peak}}(\%)$.

Solution (a) Figure 9.13 shows the deviation

$$|y(t) - y_{\text{ss}}|$$

We find that the maximum deviation is given by

$$D_{\text{peak}} = \max |y(t) - y_{\text{ss}}| = 0.54 \text{ mm}$$

This is larger than the required specification of 0.5 mm and the system does not meet the design requirements.

- (b) The percentage of steady state level requires us to refer to the first plot to determine the (non-zero) steady state level. This is read as 10 mm. Therefore the peak overshoot as a percentage of the system's steady state is

$$D_{\text{peak}}(\%) = \frac{D_{\text{peak}}}{y_{\text{ss}}} \times 100 = 5.4\%$$

Remark

We note that if the system were operating around a zero level, the value of $D_{\text{peak}}(\%)$ would not be calculable and we would refer only to D_{peak} .

9.4.2 Settling time (disturbance input)

We recall the definition for a 5% settling time (reference input):

$t_s(5\%)$: time required for the system to rise and settle within 5% of its final value

We can use a similar definition for the settling time after a disturbance input. Once again, we presume that the system is operating at a (non-zero) steady state condition and the 5% band is referred to this steady state condition.

Disturbance settling time = $D_{t_s(5\%)}$: time required for the system to rise and settle within 5% of its steady state value

From Figure 9.14, we can calculate the 5% settling time as:

$$D_{t_s(5\%)} = 11.3 - 10.0 = 1.3 \text{ minutes.}$$

If we look at the 'zero-reference' plot (Figure 9.14), we notice that this system would not satisfy any design specification on a 2% disturbance settling time due to the steady state offset caused by the disturbance.

Remark

Once again, the disturbance settling time can also be expressed as a value in absolute units. For example, it may have been a requirement for this system to stay within a bound of ± 0.45 mm after 3 minutes. This manner of specification avoids the necessity of referring the settling time to a specific steady state level.

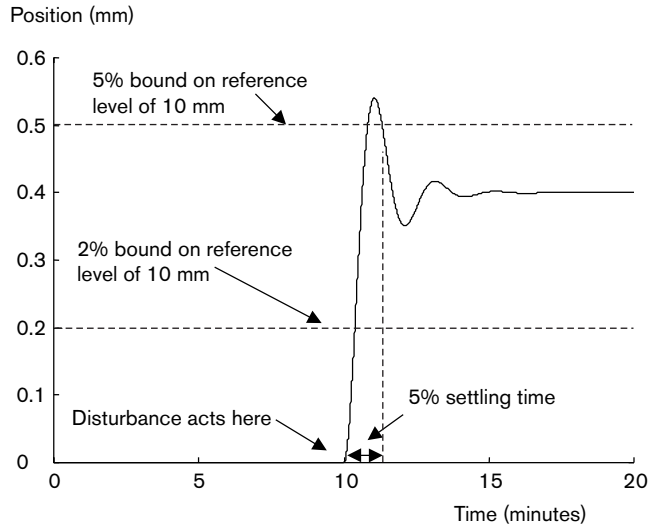


Figure 9.14 Settling time bound on offset due to disturbance.

What we have learnt

- ✓ To analyse the system behaviour to changes in input reference signals or disturbances using steady state and transient performance indices.
- ✓ To evaluate the steady state error for a system using

$$e_{ss} = \lim_{t \rightarrow \infty} e(t) = \lim_{s \rightarrow 0} sE(s)$$

- ✓ To calculate the percentage overshoot for change in reference:

$$OS (\%) = \frac{\text{peak value} - \text{final value}}{\text{final value}} \times 100$$

- ✓ To measure a peak disturbance from a step response plot:

$$D_{\text{peak}} = \max_t |y(t) - y_{ss}|$$

$$D_{\text{peak}} (\%) = \frac{D_{\text{peak}}}{y_{ss}} \times 100$$

- ✓ To measure the settling time from the plot:

$$t_s(5\%) \text{ or } t_s(2\%) \text{ and } (D_{ts}(5\%), D_{ts}(2\%))$$

- ✓ To calculate an approximate settling time from the system parameters:

$$t_s(2\%) \sim \frac{4}{\zeta\omega_n}$$

and

$$t_s(5\%) \sim \frac{3}{\zeta\omega_n}$$

✓ To measure the rise time from the step response plot

$$t_r(10\%,90\%) \text{ or } t_r(0\%,100\%)$$

We use these specifications when we go on to do control design.

Multiple choice

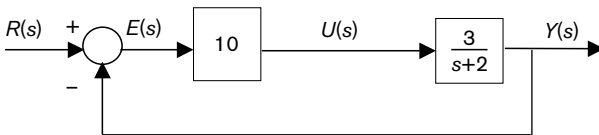
M9.1 A control design specification is required:

- (a) when the controller is just a simple device
- (b) to remove disturbances
- (c) when the final control system must meet certain requirements
- (d) there is no pre-existing knowledge of the system output

M9.2 The 'steady state error to a unit step input' refers to:

- (a) the initial error at the beginning of a step input
- (b) the final output value of the system after a step input
- (c) the magnitude of the step input
- (d) the difference between the final output and the step input

M9.3 The error transfer function between $R(s)$ and $E(s)$ for the following system is:



- (a) $\frac{s+2}{s+32}$
- (b) $\frac{s+32}{30s+32}$
- (c) $\frac{30}{s+2}$
- (d) $\frac{3}{s+2}$

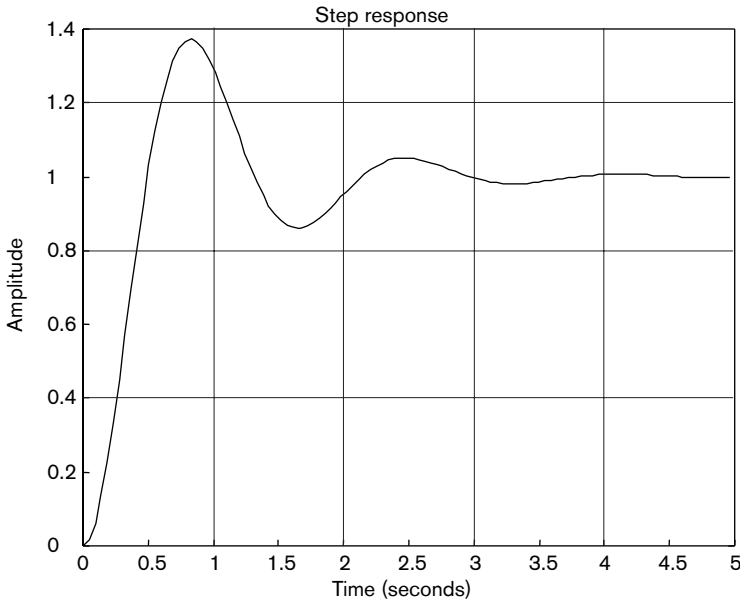
M9.4 If

$$E(s) = \frac{6}{s^2 + 3s + 4} R(s)$$

what is e_{ss} if the input signal is a step input of magnitude 2?

- (a) 3
- (b) 6
- (c) 1.5
- (d) 2

M9.5 The figure shows the output of a system to a unit step response. What is the percentage overshoot in the figure?



- (a) 3.7%
- (b) 1.37%
- (c) 13.7%
- (d) 37%

M9.6 In the previous figure, what is the rise time, $t_r(10\%, 90\%)$?

- (a) approx. 0.5 seconds
- (b) approx. 0.8 seconds
- (c) approx. 0.3 seconds
- (d) approx. 0.6 seconds

M9.7 In the previous figure, what is the damped natural frequency of the system?

- (a) approx. 2.65 rad/s
- (b) approx. 1.85 rad/s
- (c) approx. 3.55 rad/s
- (d) approx. 5.75 rad/s

M9.8 A disturbance peak value, D_{peak} , is:

- (a) the error in the steady state output of the plot
- (b) the percentage of the peak to the final output value
- (c) the actual magnitude of the peak value on the output graph
- (d) the time it takes to reach the peak value on the output graph

M9.9 The settling time, $t_s(5\%)$, is given as:

- (a) the time taken for the response to stay within 5% of its final value
- (b) the time taken to reach the final output value
- (c) the time taken to reach 95% of the final output value
- (d) the time taken to reach the 5% overshoot

M9.10 The percentage overshoot of a second-order system to a step input depends only on:

- (a) the value of the step input
- (b) the value of the damping ratio
- (c) the value of the gain K
- (d) the parameter ω_n

Questions: practical skills

Q9.1 What is the steady state value of the output from the following systems when a step of magnitude 3 is injected?

(a) $G(s) = \frac{10}{2s+1}$ (b) $G(s) = \frac{3}{s+4}$ (c) $G(s) = \frac{10}{2s-1}$

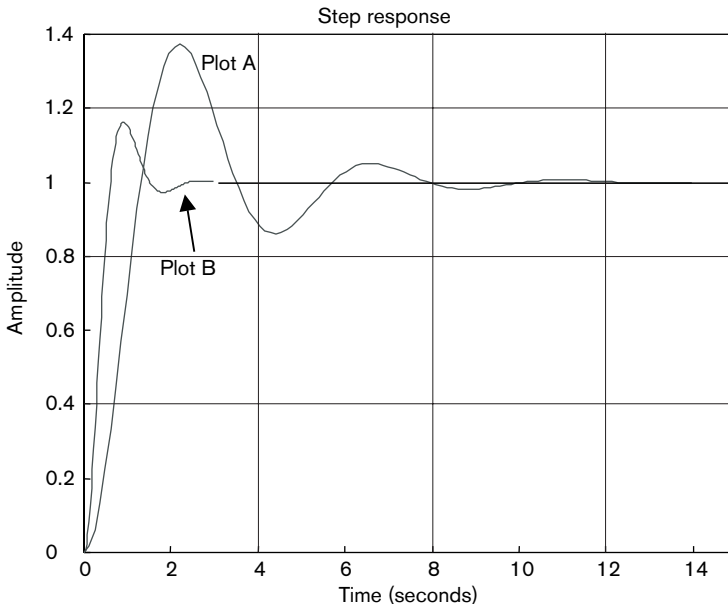
Q9.2 What is the steady state value of the output from the following systems when a step of magnitude 4 is applied?

(a) $G(s) = \frac{6}{3s^2+2s+1}$ (b) $G(s) = \frac{10}{5s^2+2s+3}$ (c) $G(s) = \frac{7}{s^2+4s-2}$

Q9.3 For plots A and B, determine

- (a) the percentage overshoot
- (b) the damping ratio
- (c) $t_r(10\%,90\%)$
- (d) $t_s(5\%)$
- (e) the damped frequency, ω_d
- (f) the natural frequency, ω_n

Hence write down the second-order transfer functions for Plot A and Plot B.



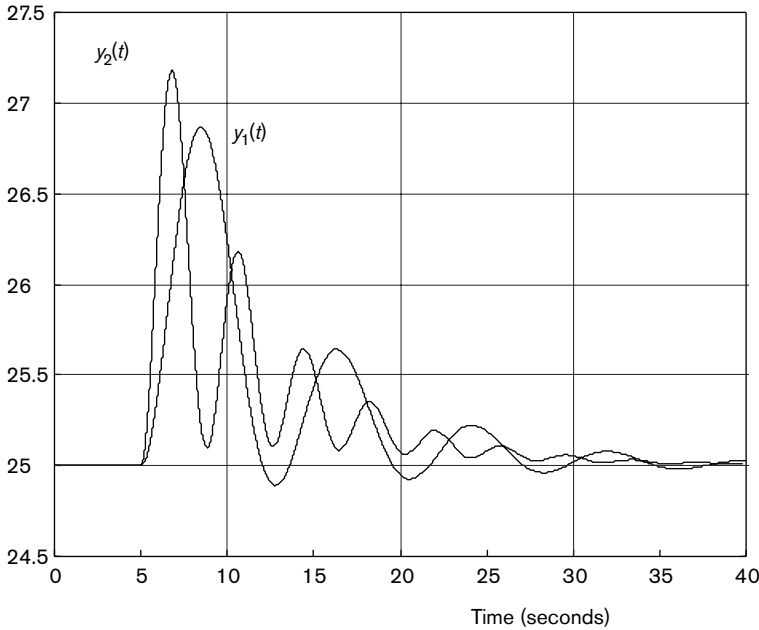
Q9.4 The plot shows two output responses, $y_1(t)$ and $y_2(t)$, from electrical systems which were at rest at a value of $y_1(t) = y_2(t) = 25$ V. An input disturbance voltage of 1 V occurs at 5 seconds.

For both systems:

- (a) What is D_{peak} ?

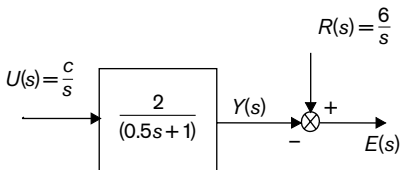
- (b) What is $D_{ts}(2\%)$?
- (c) What is the approximate value of ω_d .

Output response,
V



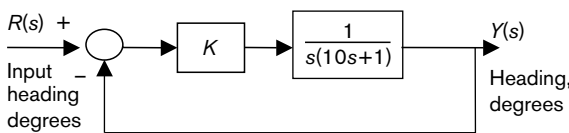
Problems

P9.1 Identify the order of the system shown below.



Determine the magnitude of a constant input signal to ensure that the steady state error, $e_{ss} = 0$. Sketch the time responses of the system output $y(t)$ and the error signal.

P9.2 For the comfort of passengers on board a ship, it is important not to have large sudden changes in position and that any oscillations should be reduced quickly. The following system represents an auto-pilot. It is desirable to switch from one heading to another with the closed-loop system satisfying $OS(\%) \leq 5\%$.

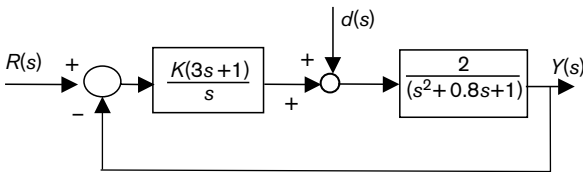


(a) Choose a value of K to satisfy the above criterion.

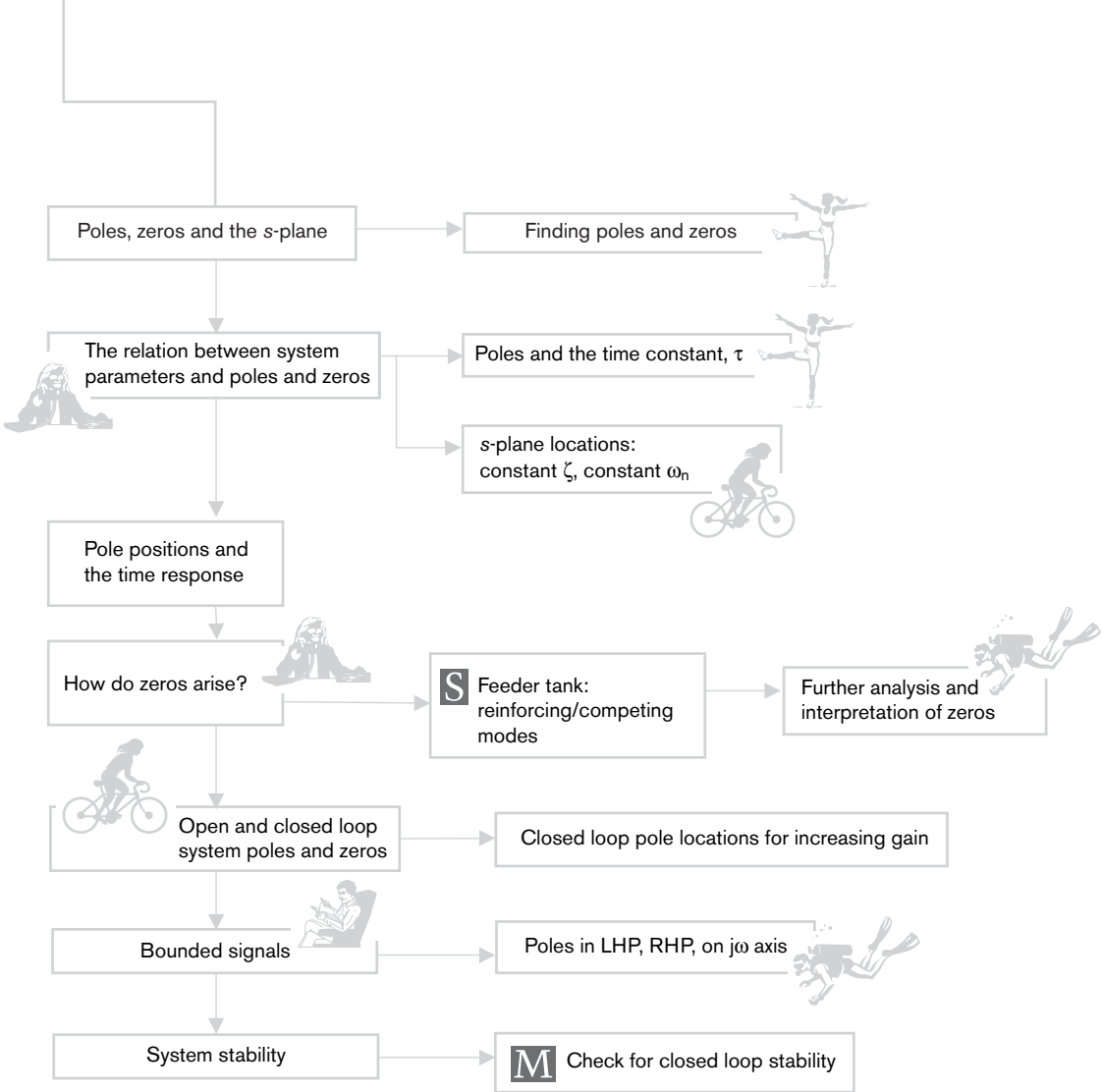
- (b) Implement your final system in Simulink and check the output response for a 10° reference heading meets the specification. What is the settling time?
- (c) What would happen if you changed K to meet a second requirement on the settling time of the system?

P9.3 The system shown represents a second-order position control system. The output $Y(s)$ gives the position in mm and is subject to disturbances acting at the input to the system. The controller is a specific type of controller called a Proportional–Integral controller, which we shall learn more about later. In this exercise we shall vary only the gain K in the controller.

- (a) Implement the system in Simulink. Let $K = 0.5$. Plot the output $y(t)$ for a disturbance input of magnitude 2. Presume that the system is already controlled to a reference level of 5 mm. What does the plot show?
- (b) Let K take the following values: $K = 0.5, 1.0, 1.5$ and 2.0 . Draw up a table showing how the change in K relates to D_{peak} .
- (c) If the reference level of the output was initially 5 mm, what values of K satisfy a 5% settling time of less than than 4 seconds.



10 Poles, zeros and system stability



When we represent systems in terms of transfer functions, the numerator and denominator of the transfer function contain key information on the system parameters which inform us about the *performance* of the system. We find that the denominator and numerator polynomials also give us the poles and zeros of the system.

The poles will provide information on the system *stability*. Stability is a main consideration when we design control systems since we do not wish to have any spiralling or unbounded output signals. Luckily for us, most everyday processes are stable. We do not have house heating systems that produce unbounded heat, nor do we have cars that accelerate in an unbounded manner. We will find out in this chapter that the essence of stability depends on whether a bounded input signal produces a bounded or unbounded output signal. We will also find that for the type of systems described by transfer function models, we do not need to examine the input and output signals, but will be able to find out about a system's stability from the locations of its poles.

The zeros of a system are also important, since they will affect the performance of a system. We will find out how zeros arise in a system and the 'odd' effects that they can cause in the system response.

Learning objectives

- To be able to determine the poles and zeros of a system.
- To relate the position of the system poles to the time domain performance indices.
- To understand how zeros arise within a system.
- To examine what is meant by a *bounded* or *unbounded* signal.
- To determine the stability of a system and its relationship to the poles of the system.

Before we study how the stability of a system is related to the poles of the system transfer function, we introduce the definitions for the poles and the zeros of a system and examine how the first and second-order system parameters (τ , ω_n and ζ) are related to pole positions.

10.1 Poles and zeros

We often express our system models in transfer function representation. For example, let the transfer function of a general system, $G(s)$, be expressed as the ratio of two polynomials:

$$G(s) = \frac{n(s)}{d(s)} = \frac{b_m s^m + b_{m-1} s^{m-1} + \dots + b_1 s^1 + b_0}{a_n s^n + a_{n-1} s^{n-1} + \dots + a_1 s^1 + a_0}$$

where $n(s)$ is the numerator polynomial of degree m and $d(s)$ is the denominator polynomial of degree n . We define the order of the system, $G(s)$, as the degree, n , of the denominator polynomial, $d(s)$. We term the roots of the denominator polynomial equation, the *poles* of the system and the roots of the numerator polynomial equation as the *zeros* of the system.

Key result: Definition of poles

The poles of the system $G(s)$ are defined as the roots of the denominator polynomial equation:

Poles: All the values of s for which $d(s) = 0$

We note that the polynomial, $d(s)$, of degree n has n roots and a system of order n has n poles. The system poles may be real and/or complex. If the poles are complex, the pole locations will occur in complex conjugate pairs.

Key result: Definition of zeros

The zeros of the system $G(s)$ are defined as the roots of the numerator polynomial equation:

Zeros: All the values of s for which $n(s) = 0$

The numerator polynomial, $n(s)$, of degree m will have m roots. (However, the order of the *system* is still determined by the order of the denominator.) The system zeros may be real and/or complex. If the zeros are complex, the zero locations will occur in complex conjugate pairs.

We recall that the variable s is the independent complex variable defined by:

$$s = \sigma + j\omega$$

where σ is the real part of s , which we also denote as $\text{Re}(s) = \sigma$ and variable ω is the imaginary part of s , which we denote as $\text{Im}(s) = \omega$. We show the range of variation of s or the domain of s as a plane (Figure 10.1).

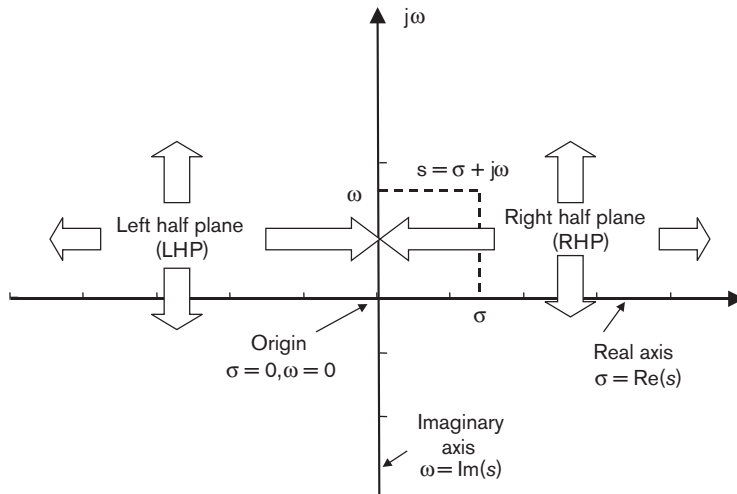


Figure 10.1 s -plane for pole and zero location.

We call the horizontal axis of this plane the real axis, since it is defined by:

$$s = \sigma + j0$$

We denote the vertical axis as the imaginary axis, since it is defined by:

$$s = 0 + j\omega$$

The origin is represented by the complex zero $s = 0 + j0$. We call the region to the left of the imaginary $j\omega$ axis (for $-\infty < \sigma < 0$) the Left Half Plane (LHP), and the region to the right of imaginary axis for ($0 < \sigma < \infty$) the Right Half Plane (RHP). The $j\omega$ axis is the border between the LHP and RHP. We call the whole complex plane the *s-plane*.

Skill section

After we have found the poles and zeros of a system's transfer function, we can locate them on the *s-plane*. This visual representation of the poles and zeros can help us in several ways. For example, we will see that the stability of system depends on the location of the poles on *s-plane*. Also, the time constant of a first-order system or the damping ratio and natural frequency of a second-order system are related to the position of the poles on the *s-plane*.

Problem Find the poles and zeros for the system model:

$$G(s) = \frac{s+1}{s^2+s-6}$$

Solution Poles

We identify the denominator polynomial as $d(s) = (s^2 + s - 6)$. Since the order of $d(s)$ is $n = 2$, $d(s)$ is a second-order polynomial, and we expect to find two poles p_1 and p_2 . To find the poles solve $d(s) = 0$:

$$d(s) = (s^2 + s - 6) = (s - 2)(s + 3) = 0$$

The poles of the system are at $s = 2$ and $s = -3$, hence the poles are $p_1 = 2$ and $p_2 = -3$.

Zeros

We identify the numerator polynomial as $n(s) = (s + 1)$. Since the order of $n(s)$ is $m = 1$, $n(s)$ is a first-order polynomial and we expect one zero, z_1 . To find the zero, solve $n(s) = 0$:

$$n(s) = (s + 1) = 0$$

The solution is $s = -1$. Hence the zero of the system is $z_1 = -1$.

We now show the positions of poles and zeros on the complex *s-plane*. For our example all the poles and zeros are real, so they are located on the real axis, $\text{Re}(s)$. We represent the poles by a cross (x) and the zeros by a circle (O), as shown in Figure 10.2.

We call this schematic representation the *pole-zero map*. For high-order transfer functions we can use the MATLAB functions `poles`, `zeros` and `pzmap` to find the poles and zeros or to draw the pole-zero map.

Problem Find the poles and zeros of the transfer function

$$G(s) = \frac{s+3}{(s-4)(s^2+12s+52)}$$

Draw the pole-zero map of $G(s)$.

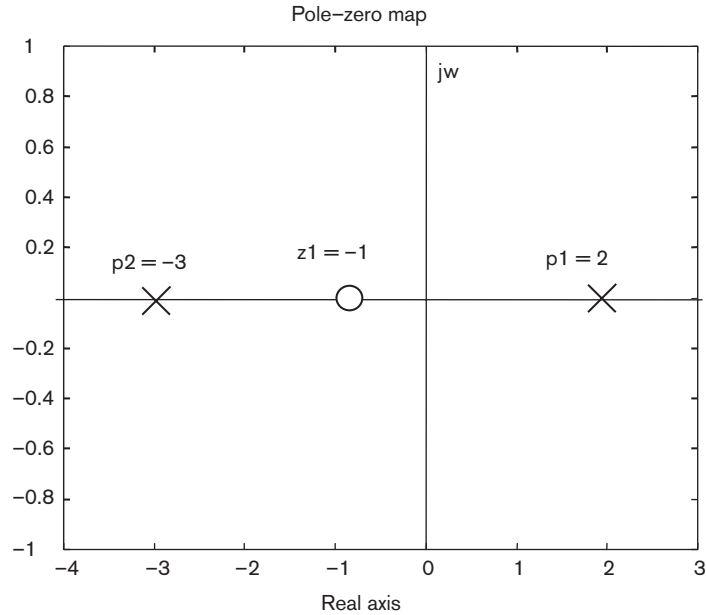


Figure 10.2 Pole-zero diagram for $G(s) = (s + 1)/(s^2 + s - 6)$.

Solution We identify the numerator and denominator polynomials as:

$$n(s) = (s + 3)$$

$$d(s) = (s - 4)(s^2 + 12s + 52)$$

Poles

Solve $d(s) = (s - 4)(s^2 + 12s + 52) = 0$.

Hence $(s - 4) = 0$ and $(s^2 + 12s + 52) = 0$.

$$(s - 4) = 0 \rightarrow s = 4$$

Thus the pole is $p_1 = 4$.

$$(s^2 + 12s + 52) = (s + 6)^2 + 16 = 0$$

$$(s + 6)^2 = -16 = -4^2 \rightarrow (s + 6) = \pm\sqrt{-(4)^2} = \pm 4j \rightarrow s = -6 \pm 4j$$

so that $p_2 = -6 + 4j$ and $p_3 = -6 - 4j$

Therefore $p_1 = 4$, $p_2 = -6 + j4$ and $p_3 = -6 - j4$ are the poles of $G(s)$. Note that $G(s)$ has a pair of complex poles $p_{2,3} = -6 \pm j4$ which are symmetric with respect to the real axis. These poles are known as complex conjugate poles.

Zeros

Solve $n(s) = 0$:

$$n(s) = (s + 3) = 0 \rightarrow s = -3$$

Therefore $z_1 = -3$ is a zero of $G(s)$.

We can now draw the pole-zero map as shown in Figure 10.3.

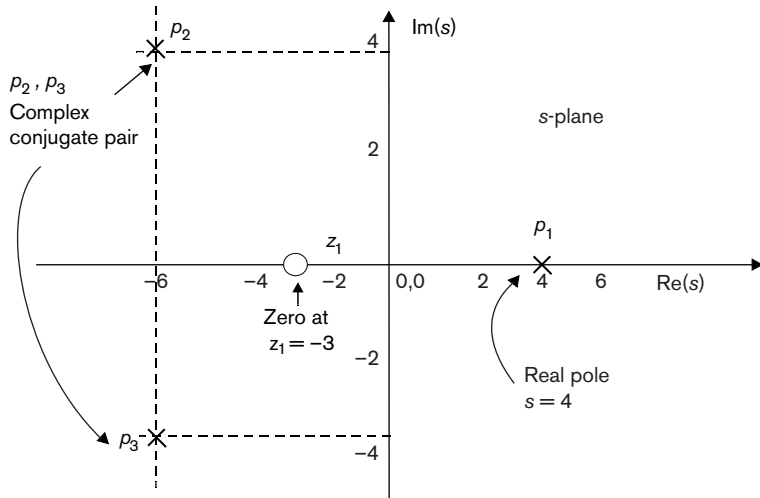


Figure 10.3 Pole-zero diagram for $G(s) = (s + 3)/[(s - 4)(s^2 + 12s + 52)]$.

Multiple poles

When the system $G(s)$ has more than one pole or one zero at the same coordinates on the s -plane, we say that $G(s)$ has multiple poles or multiple zeros. For example, consider the following system transfer function:

$$G(s) = \frac{1}{s^2}$$

To find the poles, identify the denominator polynomial, $d(s)$, and solve

$$d(s) = s^2 = 0$$

This has two roots both at $s = 0$, hence $p_1 = 0$ and $p_2 = 0$, since it is of degree two. Thus the transform has a multiple pole at the origin of the s -plane, and we indicate this by the double cross (\otimes) in Figure 10.4.

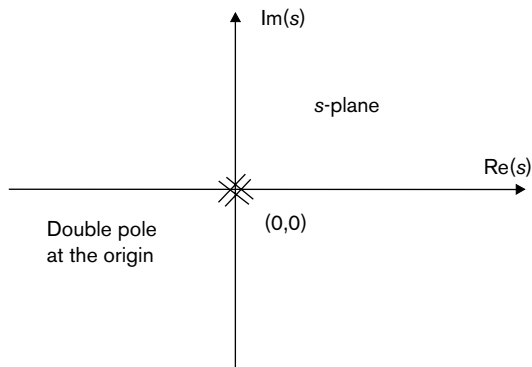


Figure 10.4 Pole-zero map for $G(s) = 1/s^2$.

10.2 System parameters and their relationship to pole locations

When we study control systems, we are often interested to see how the performance of the system changes as the damping ratio or the natural frequency are varied. Alternatively, it is often useful to know where the poles of a system should be located in order to obtain a desired set of control design specifications. To do this we need to relate the performance parameters (first-order time constant, second-order damping ratio and natural frequency) to the location of the poles of the system.

Skill section

Relationship between the first-order system poles and its time constant

For the first-order system transfer function, $K/(\tau s + 1)$, the system pole is at $s = -1/\tau$. We notice that the model parameter K does not affect the poles of the system. It will have an effect on the steady state or final value of the output but has no effect on the speed of the response. Only the system pole location influences the dynamical shape or speed of response of the output through the time constant parameter τ .

Problems Find the poles and the time constants of the following first-order systems:

(a) $G_1(s) = \frac{1}{2s+1}$

(b) $G_2(s) = \frac{1}{5s+1}$

(c) $G_3(s) = \frac{1}{10s+1}$

(d) $G_4(s) = \frac{1}{20s+1}$

Solution The poles and time constants of the system transfer functions are given by

(a) $s_1 = -0.5, \quad \tau_1 = 2$

(b) $s_2 = -0.2, \quad \tau_2 = 5$

(c) $s_3 = -0.1, \quad \tau_3 = 10$

(d) $s_4 = -0.05, \quad \tau_4 = 20$

We can compare the unit step responses and the poles of the systems as shown in Figure 10.5.

As τ increases the response becomes slower; that is, it takes longer to reach 63.2% of the final output. We can see that the pole corresponding to the fastest step response lies furthest from the origin, while the pole closest to the origin results in the slowest step response. This correlation of speed of response and the location of the system pole is a useful effect to appreciate.

Key result: Time constants, poles and speed of response

A large time constant produces a slow response and its pole will lie close to the imaginary axis.

A small time constant gives a fast response and its pole will lie further from the imaginary axis.

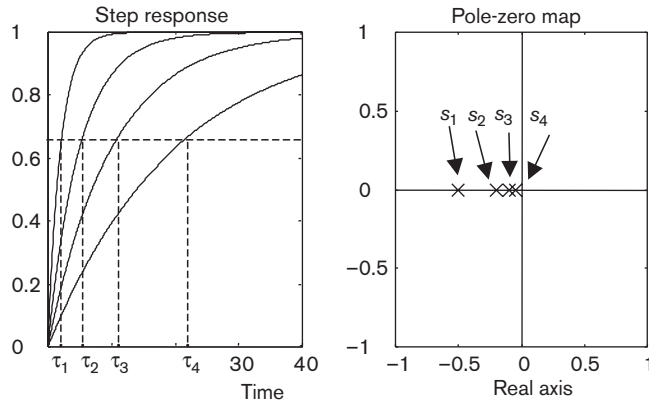


Figure 10.5 Step responses and pole locations for different first-order systems.

10.2.1 Second-order systems

Consider the second-order system:

$$G(s) = \frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2}$$

This system will have different step responses (underdamped, critically damped and overdamped), depending on the value of damping ratio ζ . We will now try to find a relationship between ζ and ω_n and the position of the poles for a second-order system.

For underdamped systems, the damping ratio satisfies $0 < \zeta < 1$, and the two poles for the underdamped system are:

$$p_1 = -\zeta\omega_n + j\omega_n\sqrt{1-\zeta^2}$$

$$p_2 = -\zeta\omega_n - j\omega_n\sqrt{1-\zeta^2}$$

We can represent these poles on the s -plane as shown in Figure 10.6.

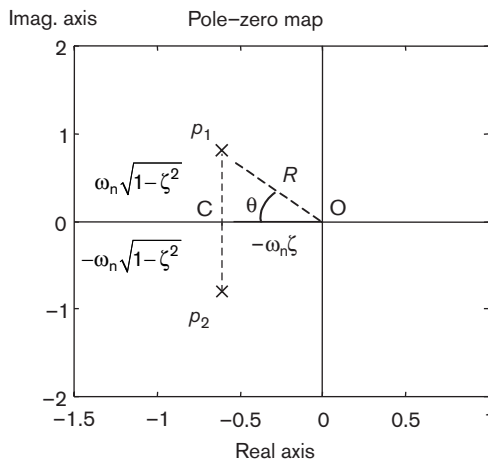


Figure 10.6 Example of second-order underdamped system pole locations.

We can apply trigonometry to the triangle OCp_1 to determine the angle θ .

$$\tan \theta = \frac{\omega_n \sqrt{1-\zeta^2}}{\zeta \omega_n} = \frac{\sqrt{1-\zeta^2}}{\zeta}$$

This states that the angle, θ , that the poles make with the real axis position will depend entirely on the damping ratio of the system. If $\zeta = 0$ then $\theta = \pi/2$ and if $\zeta = 1$ then $\theta = 0$. If $0 < \zeta < 1$, then $\theta =$ will lie between 0 and $\pi/2$.

We now calculate the radial value of R in Figure 10.6, which is the distance from the origin to the pole. By trigonometry again

$$R^2 = (\zeta \omega_n)^2 + (\omega_n \sqrt{1-\zeta^2})^2 = \zeta^2 \omega_n^2 + \omega_n^2 - \zeta^2 \omega_n^2 = \omega_n^2 \quad \text{or} \quad R = \omega_n$$

Therefore all poles with constant value for ω_n will lie on a semicircle, radius $R (= \omega_n)$ from the origin.

We can plot the loci of the poles on the pole-zero map:

- (i) For different values of ω_n , we obtain a number of semicircles.
- (ii) For different values of ζ , we find a number of straight lines making different angles with the real axis. We can then easily establish a relationship between the location of poles and ζ and ω_n . This is shown in Figure 10.7.

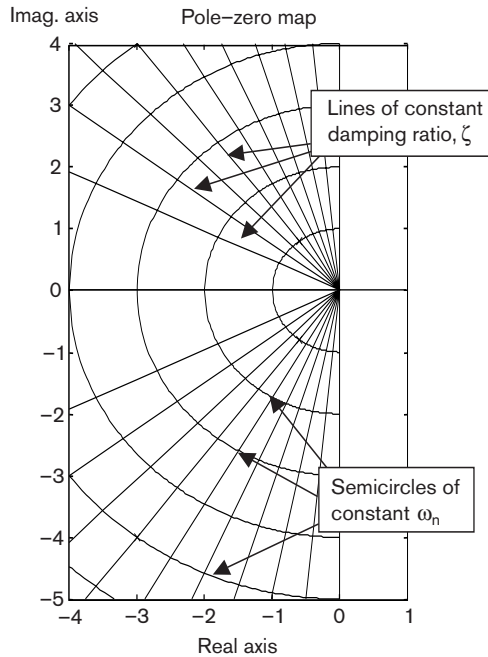


Figure 10.7 Contours of constant damping and natural frequency.

If we keep ω_n constant and change ζ from zero to one, the poles of the system move on a semicircle starting from the $j\omega$ axis and end on the real axis where the two poles meet and the system becomes critically damped.

As we increase the value of $\zeta > 1$, the system becomes overdamped and the poles become real. One pole will move to the origin of the s -plane and the other pole moves to

$-\infty$ as the value of ζ approaches infinity. We call this semicircle and the negative part of the real axis, the *locus* of all the poles of a second-order system.

Remark

- The imaginary axis represents the locus of all the poles for constant $\zeta = 0$ (the oscillatory systems).
- The negative real axis represents the pole locations for constant values of $\zeta \geq 1$.

10.3 The link between pole position and system step response

We look at the four system categories: oscillatory, underdamped, critically damped and overdamped.

Oscillatory systems

Oscillatory systems have zero damping factor. By substituting for $\zeta = 0$ in $G(s)$, we obtain:

$$G(s) = \frac{\omega_n^2}{s^2 + \omega_n^2}$$

We find the poles by setting the denominator polynomial equal to zero and solving for the roots:

$$s^2 + \omega_n^2 = 0 \rightarrow s = \pm j\omega_n$$

The system has a pair of complex conjugate poles on the $j\omega$ axis ($p_1 = j\omega_n$, $p_2 = -j\omega_n$) (Figure 10.8(b)). The unit step response of the system is a sine wave of amplitude one and frequency ω_n (Figure 10.8(a)).

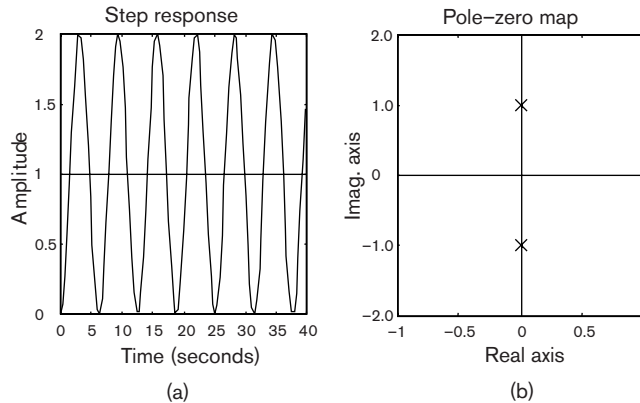


Figure 10.8 Oscillatory system. (a) Step response; (b) pole-zero map.

Underdamped system

The system has a pair of complex conjugate poles:

$$p_1 = -\zeta\omega_n + j\omega_n\sqrt{1-\zeta^2}$$

$$p_2 = -\zeta\omega_n - j\omega_n\sqrt{1-\zeta^2}$$

The step response of an underdamped system is also shown in Figure 10.9(a). As ζ approaches zero, the system response becomes more oscillatory. The poles have negative real parts ($\zeta > 0$ and $\omega_n > 0$), and hence they always lie on the LHP for $0 < \zeta < 1$ (Figure 10.9(b)).

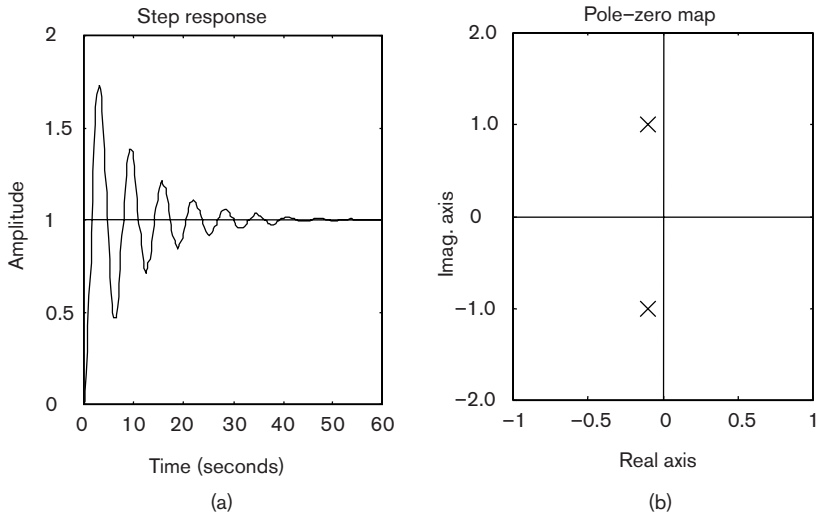


Figure 10.9 Step response and pole-zero map for underdamped system.

Critically damped system

For a critically damped system, the damping ratio is $\zeta = 1$. Setting $\zeta = 1$, we can write $G(s)$ as:

$$G(s) = \frac{\omega_n^2}{(s + \omega_n)^2}$$

The system has two real poles at $s = -\omega_n$ and $s = -\omega_n$. If we let $\omega_n = 1$, we see in Figure 10.10(a) that the unit step response is similar to a first-order exponential-like function,

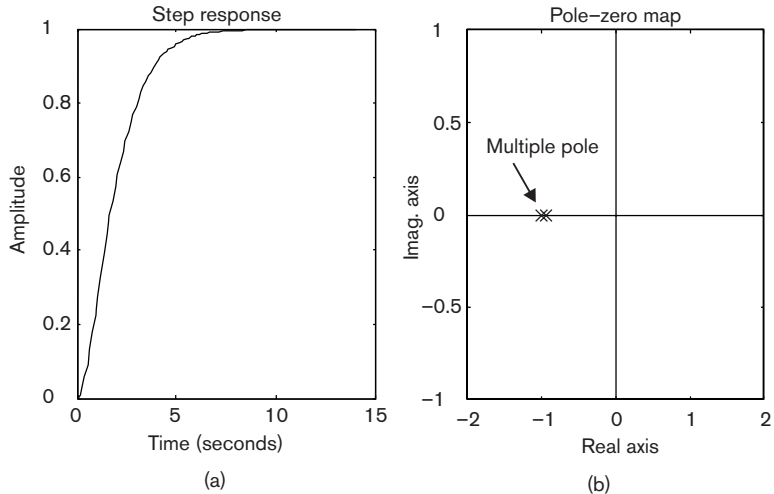


Figure 10.10 Critically damped system. (a) Step response; (b) pole-zero map.

which settles to a steady state value as time increases. Figure 10.10(b) shows the pole locations for the case $\omega_n = 1$.

Overdamped system

For an overdamped system, the damping ratio is greater than one, $\zeta > 1$. The poles are:

$$s^2 + 2\zeta\omega_n s + \omega_n^2 = 0 \rightarrow s = -\zeta\omega_n \pm \omega_n\sqrt{\zeta^2 - 1}$$

Since $\zeta > 1$, both poles are real: $p_1 = -\zeta\omega_n - \omega_n\sqrt{\zeta^2 - 1}$ and $p_2 = -\zeta\omega_n + \omega_n\sqrt{\zeta^2 - 1}$. Both poles have negative real parts, and therefore the poles lie in the LHP on the negative real axis. If we consider the example $s^2 + 5.2s + 1 = 0$, we find that the poles lie at $s = -5$ and $s = -0.2$ (Figure 10.11(b)). The corresponding unit step response is shown in Figure 10.11(a).

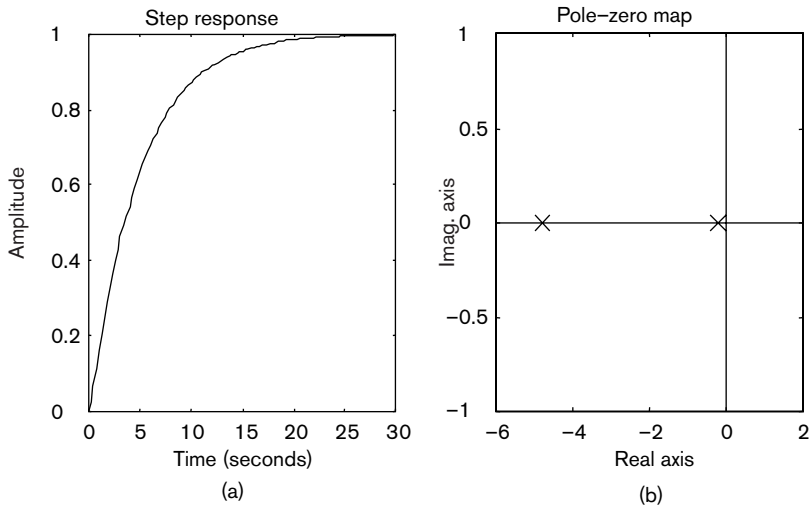


Figure 10.11 Overdamped system. (a) Step response; (b) pole-zero map.

10.4 How do the zeros of a transfer function model arise?

Earlier we looked at how the poles of a system transfer function arose and we were quite willing to accept that the poles represented the dynamical motions of a system. When we ask the same question about the transfer function zeros we do not usually receive such a straightforward answer to the question. This is surprising, because system zeros are a very common feature of system transfer function models. Firstly, we look at how the zeros arise before analysing their effect on a system.

10.4.1 Origin of zeros

Zeros arise from the internal physical pathways of a process and represent where these internal effects are adding together or competing (subtracting) with one another. Figure 10.12 shows examples where the internal pathways are being added or subtracted to produce zeros.

We now look at an example of the causes and consequences of a zero in the LHP and the RHP. The example we use is of a feeder tank used in process control.

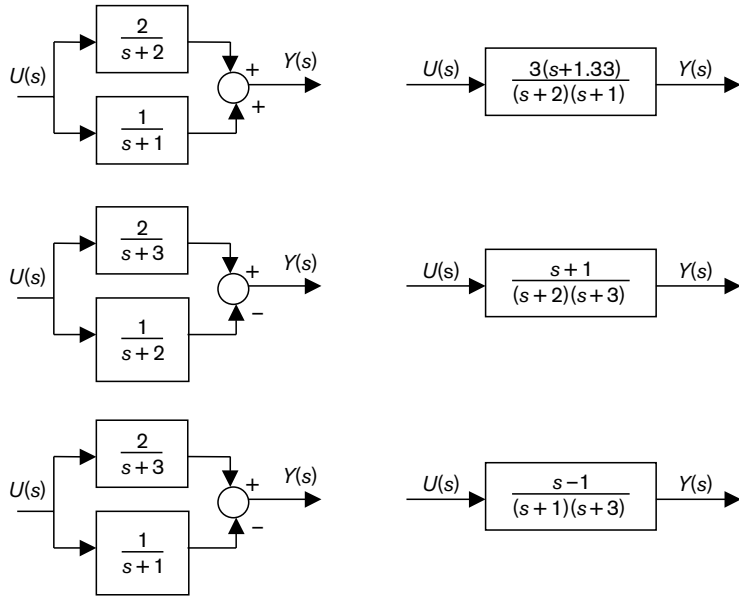


Figure 10.12 Zeros arising from the addition or subtraction of different pathways.

10.4.2 Feeder tank simulation

The feeder tank is used to supply a steady flow of liquid feed to downstream processes. This tank is shown in Figure 10.13 and it has the length, width and depth dimensions of 4 m, 3 m and 3 m respectively, giving a cross-sectional area of 12 m².

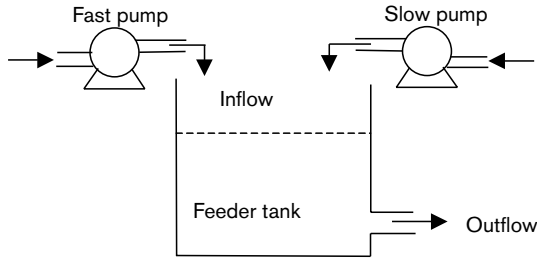


Figure 10.13 Feeder tank example.

There are two pumps in the system: a fast-responding pump and a slow but more powerful pump.

System modelling: general tank model

The height of the feed liquid is assumed to be measured from the level of the outflow pipes.

$$\text{Rate of change of volume} = \text{inflow} - \text{outflow}$$

Thus

$$A \frac{dh}{dt} = 12 \frac{dh}{dt} = f_{\text{T}}(t) - f_{\text{o}}(t)$$

where the cross-sectional area is $A = 12 \text{ m}^2$, the total inflow $= f_T(t)$ and the outflow is determined from $f_o(t) = 1.2h(t)$ with the coefficient $1.2 \text{ m}^2/\text{min}$ being determined from flow experiments.

Thus

$$12 \frac{dh}{dt} = f_T(t) - 1.2h(t)$$

or, equivalently

$$12 \frac{dh}{dt} + 1.2h(t) = f_T(t)$$

Assuming $h(0) = 0$, and taking Laplace transforms

$$12sH(s) + 1.2H(s) = F_T(s)$$

giving

$$H(s) = \frac{0.833}{10s+1} F_T(s)$$

The pumps have been calibrated and their details are:

$$\text{Fast pump} \quad F_f(s) = \frac{0.5}{0.75s+1} V_{in}(s)$$

with $\tau_f = 0.75 \text{ min}$, $K_f = 0.5 \text{ m}^3/\text{min}/\text{volt}$ and $V_{in}(s) = V/s$.

$$\text{Slow pump} \quad F_s(s) = \frac{1.0}{5s+1} V_{in}(s)$$

with $\tau_s = 5 \text{ min}$, $K_s = 1.0 \text{ m}^3/\text{min}/\text{volt}$ and $V_{in}(s) = V/s$.

Thus the fast pump is less powerful than the slow pump, and the same step voltage input drives both.

Case 1: Reinforcing mode connection

In this connection we make the reasonably sensible suggestion that the two pumps should be made to work together.

The control of level will be

$$\begin{aligned} F_T(s) &= F_f(s) + F_s(s) \\ &= \frac{0.5}{0.75s+1} V_{in}(s) + \frac{1.0}{5s+1} V_{in}(s) \end{aligned}$$

giving

$$F_T(s) = \frac{1.5(2.167s+1)}{(0.75s+1)(5s+1)} V_{in}(s)$$

and

$$H(s) = \frac{0.833}{10s+1} F_T(s) = \frac{1.25(2.167s+1)}{(10s+1)(0.75s+1)(5s+1)} V_{in}(s)$$

- (i) We can see that in this connection the system has a zero at $s = -1/2.167 = -0.46$. This is a Left Half Plane zero.

- (ii) The steady state value for level is calculated as $h_{ss} = 1.25$.
- (iii) We can use the Simulink model in Figure 10.14 to plot the output response for the system. We have marked the summing point on the Simulink model, since this represents the flow into the tank.

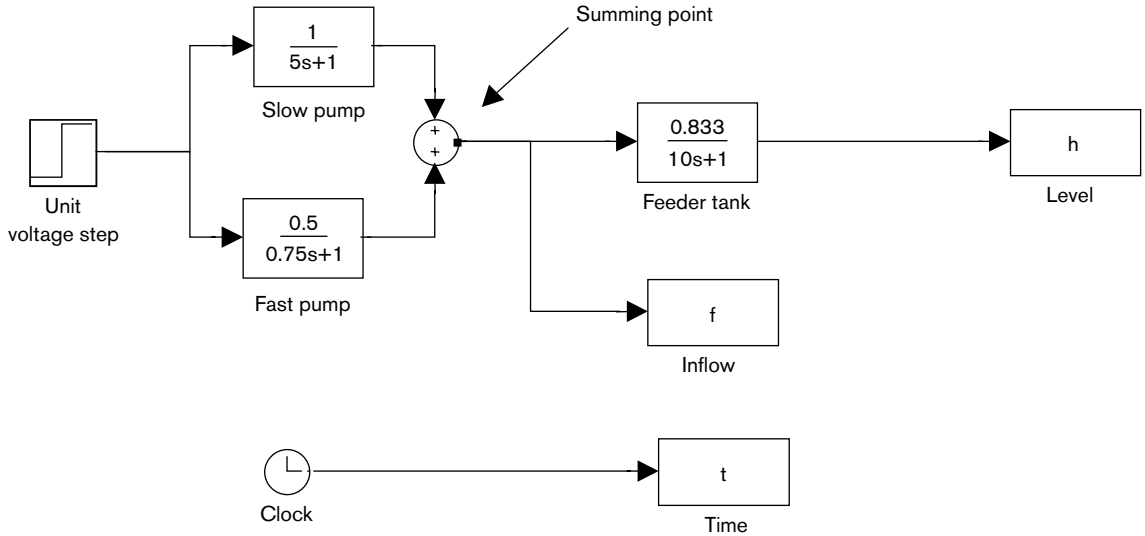


Figure 10.14 Simulink representation of feeder tank.

The Simulink response to a 1 volt signal on $v_{in}(t)$ shows a reasonably well behaved over-damped response (Figure 10.15). This response may be a little slow but it is unremarkable.

Case 2: Competing mode connection

In this connection we make the pumps work in opposing modes against one another. We think that this is an unreasonable method, but it is interesting to see what happens. The control of the level will be given by

$$\begin{aligned}
 F_T(s) &= F_s(s) - F_f(s) \\
 &= \frac{1}{5s+1} V_{in}(s) - \frac{0.5}{0.75s+1} V_{in}(s)
 \end{aligned}$$

giving

$$F_T(s) = \frac{0.5(-3.5s+1)}{(0.75s+1)(5s+1)} V_{in}(s)$$

and

$$H(s) = \frac{0.833}{10s+1} F_T(s) = \frac{0.4165(-3.5s+1)}{(10s+1)(0.75s+1)(5s+1)} V_{in}(s)$$

- (i) We can see that in the competing mode, the connection has given rise to a zero at $s = 1/3.5 = +0.286$. This is a zero in the Right Half Plane.

- (ii) The steady state value for level is calculated at $h_{ss} = 0.4165$, which is less than for the reinforcing connection mode.
- (iii) The Simulink model can represent the competing mode by changing the operations at the summing point to '+' and '-'. The plot for a 1 volt signal on $v_{in}(t)$ shows quite a remarkable response (Figure 10.15). We see that the response goes negative before going positive, and attaining a positive steady state level. This initial negative-going response is due to the fast pump *removing* feed liquid from the tank before the slow but more powerful pump has had time to respond. The result is that the level *falls* before the slow pump is able to recover control of the level. In the long run the slow pump dominates, but is less effective due to the initial response of the fast pump. The first valuable lesson we should learn is that the presence of Right Half Plane zeros in the system transfer function model means that the system will have possible control problems and unusual effects in the system responses. The second lesson to be learnt is that we should try to understand why a Right Half Plane zero has occurred, since there are usually good physical reasons for its presence.

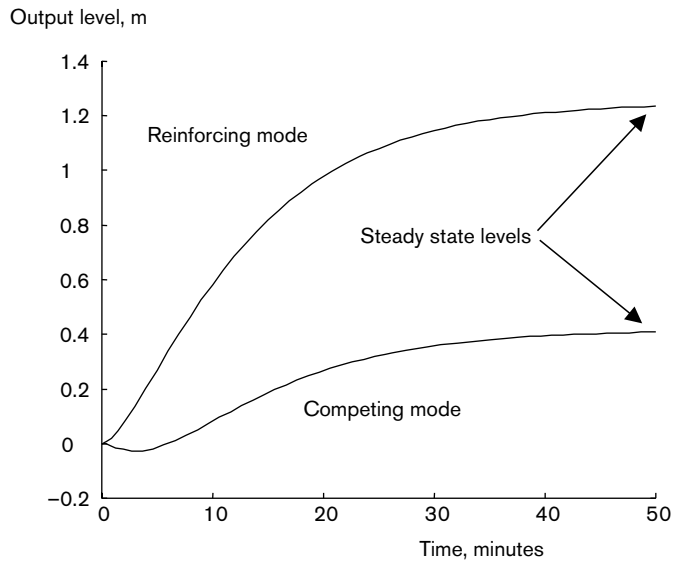


Figure 10.15 Step responses for feeder tank systems with reinforcing and competing flows.

10.5 Further analysis and interpretation of the role of zeros in a system

- **Analytical feature:** If we have a system transfer function, $G(s)$, which has a zero at $s = s_z$, then $G(s_z) = 0$. In other words, zeros are called zeros because they cause the transfer function to be zero at the position of the zero!
- **Interpretation:** This one is a little subtle. If an input signal to the plant has a pole at the zero location, then the physical system does not transmit this signal into the output. This means that we have lost the ability to transmit *all* input signals; zero locations are identified with those signals we can no longer transmit through the system.

So zeros are really very important to our understanding of system models and they play a role in deciding whether we shall be able to achieve good performance from the control system.

10.5.1 A computational example

An electrical circuit has a transfer function description which represents the input voltage, $V_i(s)$, to output voltage, $V_o(s)$:

$$V_o(s) = \frac{(s+1)}{(s+2)(s+3)} V_i(s)$$

To find the poles and zeros we identify denominator and numerator polynomials and set up the pole and zero equations as follows:

$$G(s) = \frac{n(s)}{d(s)} = \frac{(s+1)}{(s+2)(s+3)}$$

For poles $d(s) = (s+2)(s+3) = 0$ and $s_1 = -2, s_2 = -3$ defines the poles. For zeros $n(s) = (s+1) = 0$ and $s_z = -1$. Hence we have poles at $s_1 = -2, s_2 = -3$, and one zero at $s_z = -1$. If we look at the transfer function value at the zero location we find:

$$G(s_z) = \frac{s_z+1}{(s_z+2)(s_z+3)} = \frac{-1+1}{(-1+2)(-1+3)} = \frac{0}{(1)(2)} = 0$$

Thus the transfer function has zero gain at the zero location, $s_z = -1$.

To understand the subtle interpretation about not transmitting certain types of input signal, let us look at the step response. The input voltage is a 10 V step, hence $V_i(s) = 10/s$.

The output voltage is evaluated as:

$$V_o(s) = G(s)V_i(s) = \frac{(s+1)}{(s+2)(s+3)} \frac{10}{s}$$

Using partial fractions we find

$$V_o(s) = \frac{1}{6s} + \frac{57}{6(s+2)} - \frac{58}{6(s+3)}$$

and

$$v_o(t) = \frac{1}{6}(1 + 57e^{-2t} + 58e^{-3t}) \quad t > 0$$

Notice how the output voltage contains three components, one from the input term, $(1/s)$, one from the system pole at $s = -2$, (the term $1/(s+2)$) and one from the other system pole at $s = -3$, (the term $1/(s+3)$). Clearly the output does have a time component from the input step.

By way of contrast, look at the following. The system zero location is $s_z = -1$. Consider an input signal whose pole lies at this zero location, namely set $V_i(s) = 10/(s+1)$, or in the time domain $v_i(t) = 10e^{-t}$.

How does the circuit respond to this particular input? The voltage output is evaluated as

$$V_o(s) = G(s)V_i(s) = \frac{s+1}{(s+2)(s+3)} \frac{10}{s+1} = \frac{10}{(s+2)(s+3)} = \frac{10}{(s+2)} - \frac{10}{(s+3)}$$

and the output time voltage response is $v_o(t) = 10e^{-2t} - 10e^{-3t}$. Unlike the step response calculation, we appear to have lost the input signal $v_i(t) = 10e^{-t}$ completely within the

circuit, and a component has not appeared in the output voltage $v_o(t)$. This is a very clear demonstration of how the zero at $s_Z = -1$ has blocked the transmission of a particular signal, e^{-t} , since at $s = s_Z = -1$ the system gain is zero, $G(-1) = 0$.

10.6 Open- and closed-loop poles and zeros

We have found that open-loop systems have limitations, for example, in rejecting disturbance signals, and we learned how we can use feedback to improve the performance of systems. We will now study the effect of feedback control systems on the position of the poles and zeros of the closed-loop system. Consider the unity feedback system shown in Figure 10.16.

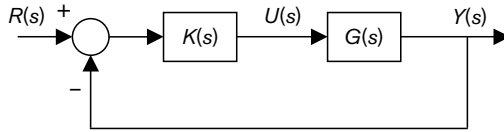


Figure 10.16 Unity feedback system.

10.6.1 Some transfer function analysis

We assume

$$G(s) = \frac{n_G(s)}{d_G(s)}$$

$$K(s) = \frac{n_K(s)}{d_K(s)}$$

We can then find the closed-loop transfer function as:

$$G_{CL}(s) = \frac{Y(s)}{R(s)} = \frac{n_{CL}(s)}{d_{CL}(s)} = \frac{K(s)G(s)}{1 + K(s)G(s)}$$

We replace for $K(s)$ and $G(s)$ to obtain:

$$G_{CL}(s) = \frac{n_{CL}(s)}{d_{CL}(s)} = \frac{\frac{n_K(s)}{d_K(s)} \frac{n_G(s)}{d_G(s)}}{1 + \frac{n_K(s)}{d_K(s)} \frac{n_G(s)}{d_G(s)}} = \frac{n_K(s)n_G(s)}{d_K(s)d_G(s) + n_K(s)n_G(s)}$$

Key result: Open- and closed-loop poles

The open-loop poles will be the zeros of the denominator of the system transfer function

$$d_G(s) = 0$$

We can find the closed-loop poles by setting the denominator of the closed-loop transfer function $G_{CL}(s)$ to zero:

$$d_{CL}(s) = d_K(s)d_G(s) + n_K(s)n_G(s) = 0$$

We can see that the two equations for the open- and closed-loop poles are different.

Key result: Open- and closed-loop zeros

The open-loop zeros are the roots of the numerator equation of the system transfer function:

$$n_G(s) = 0$$

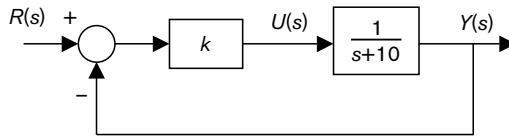
The closed-loop zeros are the roots of the numerator polynomial equation of the closed-loop transfer function:

$$n_{CL}(s) = n_K(s)n_G(s) = 0 \rightarrow n_K(s) = 0, n_G(s) = 0$$

We see that the zeros have not changed and can conclude that feedback does not change the zeros of the system. Although we can introduce extra zeros through the design of controller $K(s)$, the original system zeros will remain unchanged.

Problem: First-order system

Study the effect of the controller gain $k = 0, 1, 4, 8, 10$ on the closed-loop pole position of the following unity feedback system.



Solution System poles:

$$(s + 10) = 0 \rightarrow s = -10 \rightarrow p_1 \text{ (system)} = -10$$

Closed-loop transfer function:

$$G_{CL}(s) = \frac{k / (s+10)}{1 + [k / (s+10)]} = \frac{k}{s+10+k}$$

Closed-loop poles:

$$(s + 10 + k) = 0 \rightarrow s = -10 - k \rightarrow p_1 \text{ (closed-loop)} = -10 - k$$

We can calculate the closed-loop poles for different values of k . This is summarised in Table 10.1.

Table 10.1 Closed-loop pole locations for increasing K (first-order system).

k	System pole	Closed-loop pole
0	-10	-10
1	-10	-11
4	-10	-14
8	-10	-18
10	-10	-20

We can see that as we increase the gain k , the closed-loop poles moves further to the left in the LHP. Thus the system becomes faster, as the magnitude of the first-order system pole is inversely proportional to its time constant. The locus of the poles is shown in Figure 10.17 and confirms the result of the table.

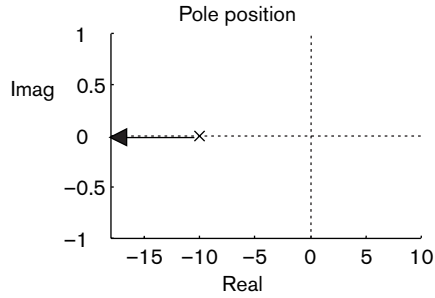
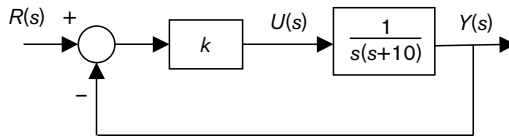


Figure 10.17 Movement of poles as the controller gain k increases.

Problem: Second-order system

Study the effect of the controller gain $k = 0, 1, 3, 5, 6, 8, 10$ on the closed-loop poles of the following unity feedback system.



Solution System poles:

$$s(s + 10) = 0 \rightarrow s = -10 \text{ and } s = 0 \rightarrow p_1(\text{system}) = -10 \text{ and } p_2(\text{system}) = 0$$

Closed-loop transfer function:

$$G_{CL}(s) = \frac{k / [s(s+10)]}{1 + k / [s(s+10)]} = \frac{k}{s^2 + 10s + k}$$

Closed-loop poles:

$$(s^2 + 10s + k) = 0 \rightarrow s = -5 \pm \sqrt{25 - k}$$

We analyse three cases depending on the value of k .

(i) $0 < k < 25$: $s = -5 \pm \sqrt{25 - k}$

closed-loop poles:

$$p_1 = -5 + \sqrt{25 - k} \text{ and } p_2 = -5 - \sqrt{25 - k} \text{ (overdamped system)}$$

(ii) $k = 25$: $s = -5$ (twice)

$$p_1 = -5 \text{ and } p_2 = -5 \text{ (critically damped system)}$$

(iii) $k > 25$: $s = -5 \pm j\sqrt{k - 25}$

closed-loop poles:

$$p_1 = -5 + j\sqrt{k - 25} \text{ and } p_2 = -5 - j\sqrt{k - 25} \text{ (underdamped system)}$$

The closed-loop poles are given for different values of k in Table 10.2.

Table 10.2 Closed-loop pole location for increasing K (second-order system).

k	System poles	Closed-loop poles
0	-10, 0	-10, 0
3	-10, 0	-9, -1
5	-10, 0	-5, -5
10	-10, 0	$-5 \pm j8.6$

Increasing gain k has a dramatic effect on the position of the closed-loop poles for the second-order system, as shown in Table 10.2 and the pole-zero map (Figure 10.18). The system is overdamped for $k = 0$ (no feedback). As we increase k to 25, the closed-loop poles both move to -5 and the system becomes critically damped. Increasing k to values above 25 makes the poles complex and they move away from the real axis (underdamped system). We can therefore change the position of the poles by changing the gain k until a desired controller performance is obtained.

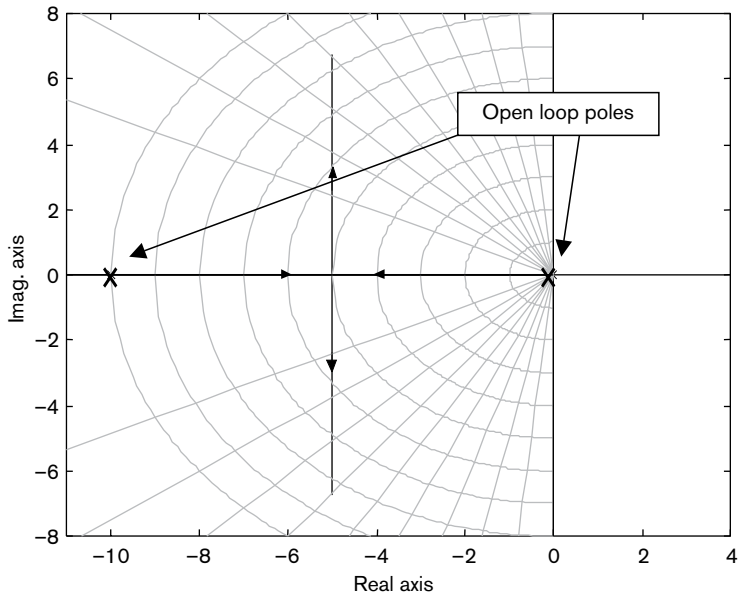


Figure 10.18 Pole-zero map for second-order system.

Example Consider the system shown in Figure 10.19.

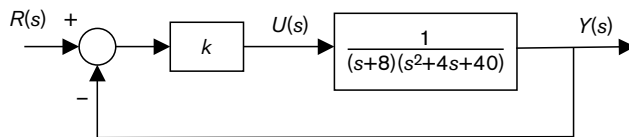


Figure 10.19 Closed-loop block diagram for $G(s) = 1/(s + 8)(s^2 + 4s + 40)$.

We set the denominator of the system transfer function to zero to find the open-loop poles:

$$(s + 8)(s^2 + 4s + 40) = 0$$

The poles are at

$$s_1 = -8$$

$$s_2 = -2 + 6j$$

$$s_3 = -2 - 6j$$

All the poles are in the LHP. The closed-loop transfer function is given by

$$Y(s) = \frac{k}{(s+8)(s^2+4s+40)+k} \quad R(s) = \frac{k}{s^3+12s^2+72s+320+k} \quad R(s)$$

The poles of the closed-loop transfer function are given by the roots of the following equation:

$$s^3 + 12s^2 + 72s + 320 + k = 0$$

For $k=0$, we get the poles of the original system transfer function. We can use MATLAB to calculate the poles for us for different values of k .

k	Poles: roots of $s^3 + 12s^2 + 72s + 320 + k$	
200	-10.0000	-1.0000 + 7.1414i -1.0000 - 7.1414i
400	-11.2770	-0.3615 + 7.9822i -0.3615 - 7.9822i
600	-12.2521	0.1261 + 8.6645i 0.1261 - 8.6645i
800	-13.0559	0.5279 + 9.2470i 0.5279 - 9.2470i

MATLAB commands:

```
k=200;
p=[1 12 72 320+k];
roots(p)
```

We see that we still have one real pole and two complex poles for the values of k chosen. The real pole stays in the LHP but moves away from the imaginary axis. The complex poles actually cross from the LHP to the RHP. We can see this in a plot by using the analysis and design tool `r1tool` in MATLAB. Figure 10.20, produced from `r1tool`, shows the three system poles and the path or locus that the poles make as the gain k increases. We can identify the point at which the two loci related to the complex poles cross the imaginary axis and become unstable poles. This tells us that if you increase the gain to a high enough value the closed-loop system will become unstable.

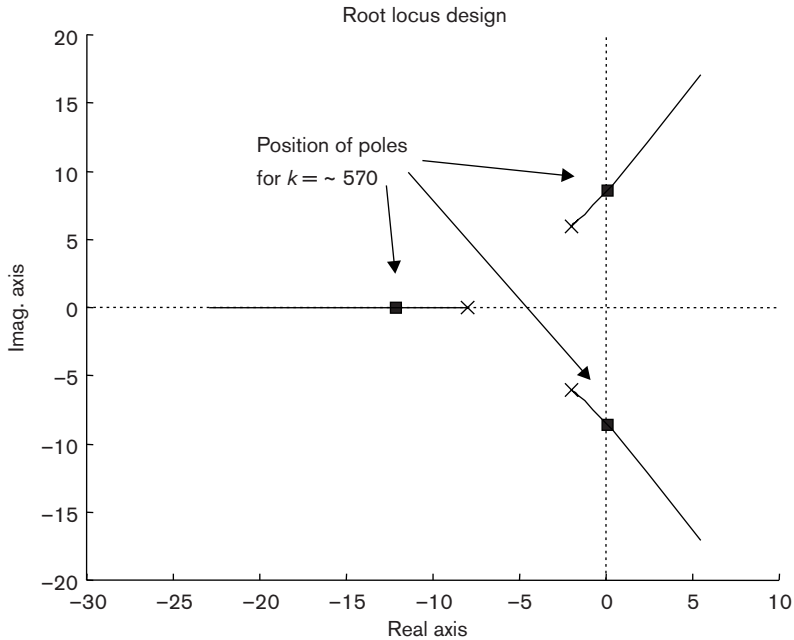


Figure 10.20 Position of closed-loop poles as gain k increases.

The definition of system stability can be related to the poles of a system. However, its strict interpretation also relies on our knowledge of the system's response to *bounded* signals. We now look at what is meant by *bounded* and *unbounded* signals before we address the question of system stability.

10.7 What do we mean by bounded signals?

A time domain signal, $x(t)$, is assessed by the behaviour of its magnitude over an infinite time interval. As time tends to infinity, the absolute value of the signal magnitude can either:

- (a) continuously decrease and/or increase (or stay constant) but remain within a bounded range
- (b) continuously increase to very large values without any bound

Figures 10.21 and 10.22 show examples of bounded exponential signals and bounded sinusoidal signals. We can see that the magnitude of an exponential function, e^{at} , with $a < 0$, will decrease to zero as time tends to infinity. The magnitude of a unit step function is finite since its value is 1, even when time tends to infinity. We call these types of signals *bounded*.

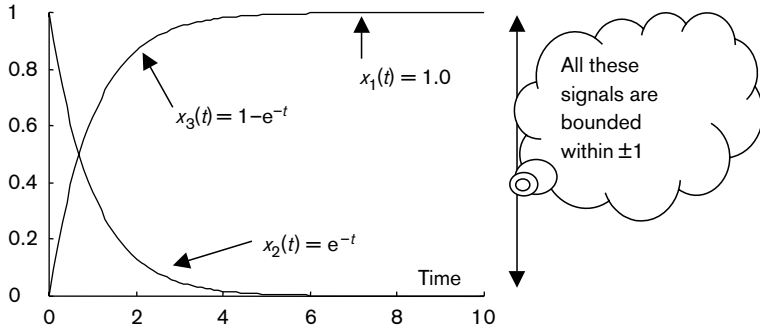


Figure 10.21 Examples of bounded exponential signals.

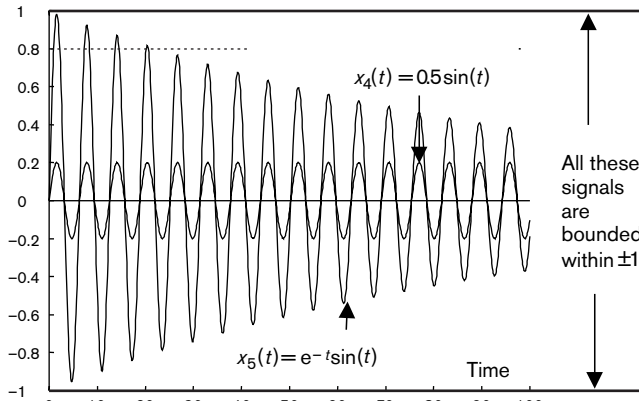


Figure 10.22 Example of bounded sinusoidal signals.

Figures 10.23 and 10.24 show examples of unbounded signals. If the exponent in the exponential signal e^{at} is positive, $a > 0$, the signal will increase to infinity as time tends to infinity. We categorise these signals as *unbounded* signals.

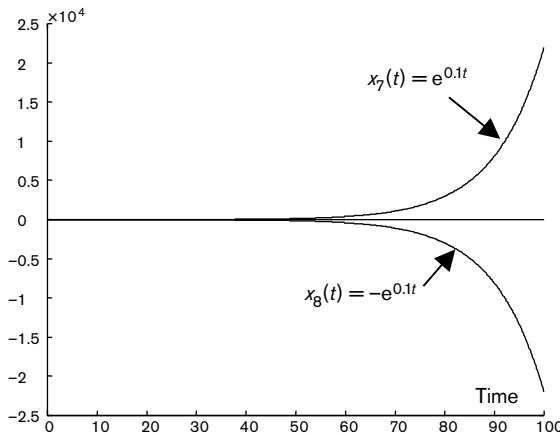


Figure 10.23 Unbounded exponential signal.

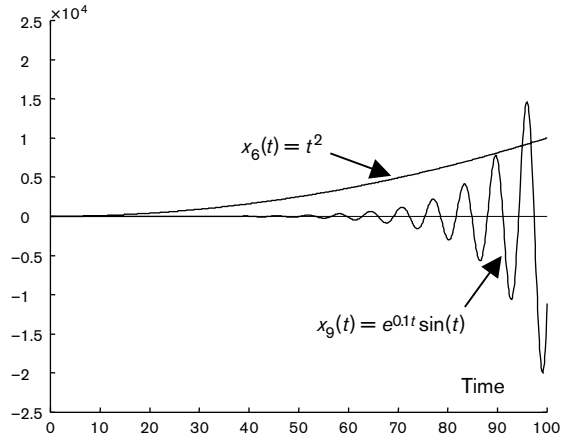


Figure 10.24 Unbounded parabolic and sinusoidal signal.

Table 10.3 examines the exponential signals and their boundedness by studying the location of their poles.

Table 10.3 Location of poles and boundedness of signals.

Exponential signal	Laplace transform	Poles	Bounded/unbounded?
$x_1(t) = e^{-10t}$	$X_1(s) = \frac{1}{s+10}$	$s = -10$	Bounded
$x_2(t) = e^{-6t}$	$X_2(s) = \frac{1}{s+6}$	$s = -6$	Bounded
$x_3(t) = e^{-at}$	$X_3(s) = \frac{1}{s+a}$	$s = -a$ $a > 0$	Bounded
$x_4(t) = e^{-0.5t}$	$X_4(s) = \frac{1}{s+0.5}$	$s = -0.5$	Bounded
$x_5(t) = e^{-0t} = 1$	$X_5(s) = \frac{1}{s}$	$s = 0$	Bounded
$x_6(t) = e^{1.5t}$	$X_6(s) = \frac{1}{s-1.5}$	$s = +1.5$	Unbounded
$x_7(t) = e^{at}$	$X_7(s) = \frac{1}{s-a}$	$s = +a$ $a > 0$	Unbounded
$x_8(t) = e^{2.8t}$	$X_8(s) = \frac{1}{s-2.8}$	$s = +2.8$	Unbounded
$x_9(t) = e^{20t}$	$X_9(s) = \frac{1}{s-20}$	$s = +20$	Unbounded

Although variable s is a complex variable given by $s = \sigma + j\omega$, for these examples the pole values are real, and the values are the same as the rate constant in the exponential function. For example, $x_7(t) = e^{at}$, and the pole is at $s = a$.

Remark **Decaying** exponential signals have Laplace transforms with poles in the LHP. **Growing** or **increasing** exponential signals have Laplace transforms with poles in the RHP. We can generalise this observation in the Key Result that follows.

Key result: Poles in LHP, RHP and on $j\omega$ axis

Signals whose transforms have all the poles in the LHP are bounded.
 Signals whose transforms have any one pole in the RHP are unbounded.

However, we must still consider the poles on the $j\omega$ axis.

Poles on $j\omega$ axis

Signals whose transforms have poles in the LHP and no multiple poles on the $j\omega$ axis are bounded, otherwise they are unbounded.

This is intuitively so; we recall that a step function has the transfer function $R(s) = 1/s$ which is a bounded (and constant!) signal and has one pole at $p = 0$. A ramp function has the transform $R(s) = 1/s^2$ which has two poles at the origin on the $j\omega$ axis. The ramp function is unbounded, since its magnitude increases without bound as time progresses.

10.8 System stability

Stability is one of the most important considerations in any system. It is thus essential to develop simple tools to examine the stability of systems. We term *unstable* systems as those that will, for example, move off position and not return to a stable equilibrium position after some initial excitation. A simple example of this is shown in Figure 10.25 (a) and (b), where we can see a pendulum and an inverted pendulum.

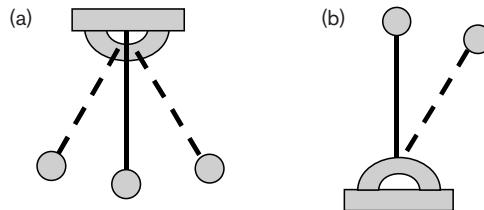


Figure 10.25 Pendulum and inverted pendulum.

The pendulum shown in Figure 10.25(a) is a stable system. If we hold the rod horizontally and release it, it goes to its rest position after some initial oscillation. An inverted pendulum, as shown in Figure 25(b), is an unstable system. If we hold the rod vertically and release it, it stays vertical unless perturbed, and then it falls from its unstable position to the horizontal, stable, position. A real example of a system which is like an inverted pendulum is a mountain bike. This is an unstable system where the rider stabilises the bike using pedal force and rider balance to keep the bike upright and moving (most of the time!).

Another example of an unstable system is an exothermic reactor. In this case, the exothermic reaction process gives out heat which must be removed by a coolant if the process is to remain under control or stable. If the heat removal balance is lost, then a runaway situation can develop, causing a spiralling or unbounded temperature rise.

10.8.1 What is a stable system?

We call a system *stable* if its output signal is bounded for *any* bounded input signal. We call this type of system stability ‘bounded-input bounded-output stability’.

A simple method to check the system stability is to examine the poles of the systems, since there is a relationship between the system poles and the system stability, which we explore now.

Consider a general transfer function of the form:

$$G(s) = \frac{n(s)}{d(s)} = \frac{b_m s^m + b_{m-1} s^{m-1} + \dots + b_1 s^1 + b_0}{a_n s^n + a_{n-1} s^{n-1} + \dots + a_1 s^1 + a_0}$$

Set the denominator polynomial equal to zero to find the poles:

$$a_n s^n + a_{n-1} s^{n-1} + \dots + a_1 s^1 + a_0 = 0$$

We can factorise this polynomial into real (first order) and complex poles (second order).

$$s^r (s + p_1) \dots (s + p_j) \dots (s^2 + 2\zeta\omega_{n1}s + \omega_{n1}^2) \dots (s^2 + 2\zeta\omega_{nj}s + \omega_{nj}^2)$$

The system poles will therefore consist of some real poles and some imaginary poles. From the superposition principle (Chapter 2) the response of the general system is the summation of the responses from all the individual poles. If *only one* of these poles lies in the RHP, the total response will be dominated by this pole and the system becomes unstable. A system is therefore stable if *all* the poles lie in the LHP.

Key result: Stability test

A system is stable if all its poles lie in the Left Half Plane.

What about poles on the $j\omega$ axis?

To answer this question we give two examples: Figures 10.26(a) and (b).

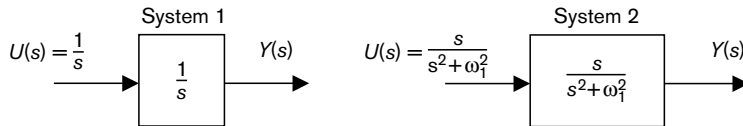


Figure 10.26 Two examples of systems with bounded input signal and unbounded output signal.

In the two examples, we can see that the input signal is a step and a sine wave. Both of these are therefore *bounded* input signals. If we examine the poles of each system we find that:

- (a) For system 1, the pole lies at the origin, on the $j\omega$ axis.
- (b) For system 2, the poles lie at $\pm \omega_1$, both on the $j\omega$ axis.

However, the output signals are given by:

(a) For system 1,

$$Y(s) = \frac{1}{s} \times \frac{1}{s} = \frac{1}{s^2}$$

which represents a ramp signal (an unbounded signal)

(b) For system 2,

$$Y(s) = \frac{s}{s^2 + \omega_1^2} \times \frac{s}{s^2 + \omega_1^2} = \left(\frac{s}{s^2 + \omega_1^2} \right)^2$$

The output is shown in Figure 10.27, where we see increasing unbounded oscillations. Although the input signal to the system is bounded, the output signal is unbounded; hence the system is unstable.

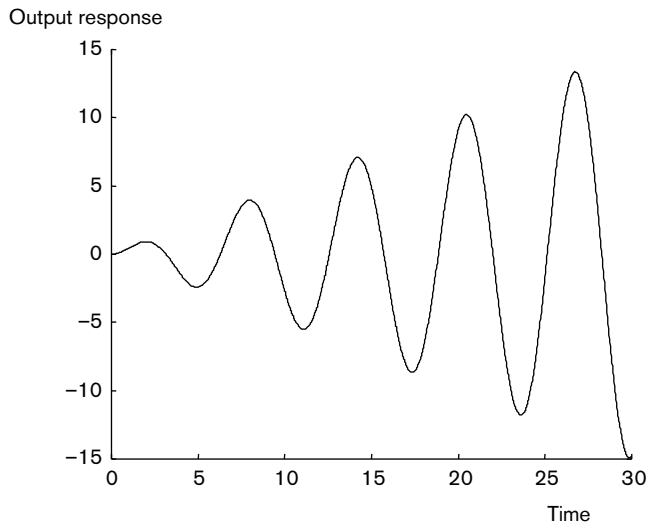


Figure 10.27 Unstable response from system with poles on $j\omega$ axis.

Although we might think that these are specific examples, the statement about stability tells us that for a system to be stable, it must produce a bounded output signal for *any* bounded input signal. In both the above cases, we have chosen a bounded input signal, but the output is unbounded; hence the systems are unstable. What we should remember from this is that poles on the $j\omega$ axis are not in the LHP, and therefore our stability test is correct provided we do not include the $j\omega$ axis in the LHP.

Example Figures 10.28 and 10.29 show the relationship between the stability of a first-order system and its pole position. The first-order system is given by

$$G(s) = \frac{1}{s+a}$$

We can see that the first-order system is *stable* if its pole lies in the LHP. If the pole lies in the RHP including the origin the system is unstable.

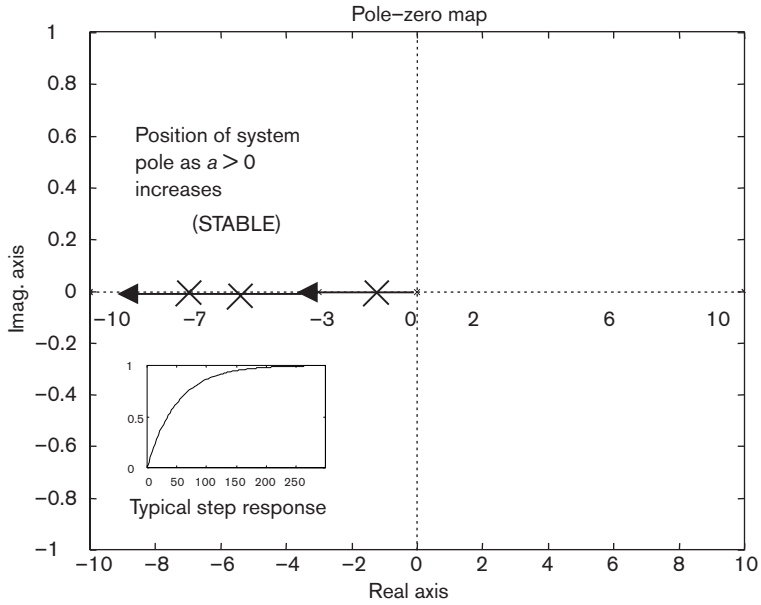


Figure 10.28 The loci of poles for the first-order system $a > 0$.

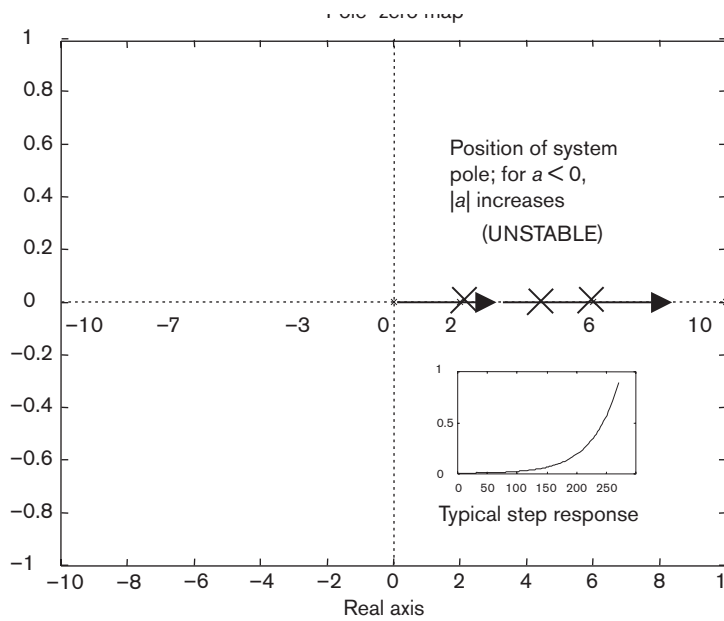


Figure 10.29 The loci of poles for the first-order system $a < 0$.

M Using MATLAB to check the stability of a system

Enter the system transfer function using:

```
s = tf('s');
g = ...
```

then either

(i) run:

```
pzmap(g)
```

If *all* the poles are located in the LHP the system is stable. Otherwise the system is unstable.

or

(ii) use

```
pole(g)
```

If *all* the poles have negative real parts then the system is stable.

To find the damping ratio and natural frequency contours run `sgrid` and read the values from the contours

Problem The following transfer function represents a large sea-going tanker where $U(s)$ is the input signal from the rudder and $Y(s)$ is the heading angle of the ship. Investigate the stability of the system.

$$Y(s) = \frac{0.0192(1+46s)}{(1-123.0s)(1+16.0s)} U(s)$$

Solution Using MATLAB we enter the transfer function as follows

```
s = tf('s');
g = 0.0192*(1+46*s)/((1-123*s)*(1+16*s));
```

The function `pzmap(g)` produces the output shown in Figure 10.30.

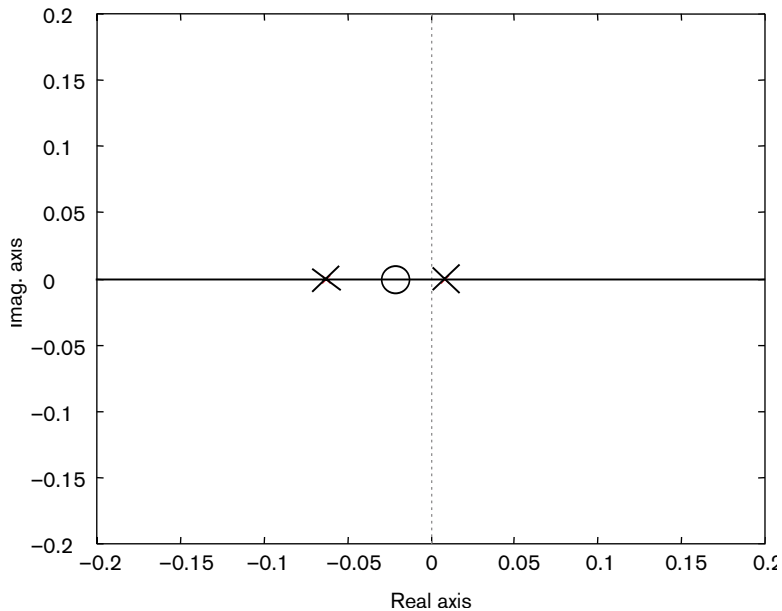


Figure 10.30 Pole-zero map for the tanker system transfer function.

Figure 10.30 shows that both the poles and the zero lie on the real axis, but one pole lies in the RHP. Since one pole is in the RHP the system is therefore unstable.

What we have learnt

- ✓ To find the poles and zeros from the transfer function representation of a system:
 - System poles are the roots of the denominator polynomial equation of the system transfer function
 - System zeros are the roots of the numerator polynomial equation of the system transfer function
- ✓ To recognise the link between the first-order time constant, τ , and the associated pole position:
 - As the time constant increases the pole approaches the origin
- ✓ To understand the relationship between the second-order system parameters (ζ and ω_n), and the associated contours on the pole-zero map:
 - Fixing the damping ratio fixes the angle of the second-order poles from the real axis
 - Fixing the natural frequency ensures the poles lie a fixed distance from the origin
- ✓ To understand how the zeros in a system arise and the blocking effect they have on signals.
- ✓ To understand that by *closing the loop*, the poles of the closed-loop system will be different from the system poles.
- ✓ To identify whether a signal is bounded or unbounded.
- ✓ To establish whether a system is stable or unstable by using the stability test:
 - The system $G(s)$ is stable if all the poles of $G(s)$ lie in the Left Half Plane, otherwise the system is unstable

Multiple choice

- M10.1** What are the pole, p , and zero, z , of the transfer function $G(s) = (s + 2)/(s + 3)$?
- (a) $p = 2, z = 3$
 - (b) $p = 3, z = 2$
 - (c) $p = -2, z = -3$
 - (d) $p = -3, z = -2$
- M10.2** If a first-order system has a pole at $p = -2$, what is the system time constant?
- (a) $\tau = 0.5$
 - (b) $\tau = 2$
 - (c) $\tau = -0.5$
 - (d) $\tau = -2$
- M10.3** Three different first-order systems have the following poles: $p_1 = -0.2, p_2 = -0.4, p_3 = -0.6$. Which system has the fastest step response?
- (a) the system with $p_1 = -0.2$
 - (b) the system with $p_1 = -0.4$
 - (c) the system with $p_1 = -0.6$
 - (d) they all respond the same since they are all first-order systems
- M10.4** Purely oscillatory systems have poles that lie:
- (a) at the same place on the negative real axis
 - (b) at the same place on the positive real axis
 - (c) in the LHP, but not on the real axis
 - (d) on the imaginary axis

M10.5 An example of a bounded signal is:

- (a) e^{-4t}
- (b) e^{2t}
- (c) t
- (d) $e^t \sin t$

M10.6 A bounded signal has poles that lie:

- (a) only in the LHP
- (b) in the LHP and on the imaginary axis
- (c) in the LHP, and single poles on the imaginary axis
- (d) in the RHP

M10.7 A stable system has poles that lie:

- (a) only in the LHP
- (b) in the LHP and on the imaginary axis
- (c) in the LHP, and single poles on the imaginary axis
- (d) in the RHP

M10.8 Are the following systems stable?

(i) $G_1(s) = \frac{2}{s+3}$

(ii) $G_2(s) = \frac{3}{s-3}$

(iii) $G_3(s) = \frac{4}{s(s+2)}$

- (a) yes, no, no
- (b) no, yes, no
- (c) yes, no, yes
- (d) no, no, yes

M10.9 Poles with the same damping ratio lie:

- (a) the same distance from the origin
- (b) on the real axis
- (c) on the same radial line from the origin
- (d) on the imaginary axis

M10.10 If a system has a zero in the RHP, its output to a positive input step will:

- (a) be unstable
- (b) be the same as one having a zero in the LHP
- (c) go negative before going positive
- (d) be negative

Questions: practical skills

Q10.1 Find the poles of the following Laplace transforms and state the general time-domain signal feature to which the poles relate. Select these features from the list:

- Fast exponential decay
- Slow exponential decay
- Constant oscillation
- Slow exponential growth
- Fast exponential growth
- Slow increasing sinusoidal oscillation
- Fast increasing sinusoidal oscillation

(a) $X_1(s) = \frac{4}{(s+3)}$

(b) $X_2(s) = \frac{10s}{(s^2+0.015625)}$

(c) $X_3(s) = \frac{7.4}{(s-0.1)}$

(d) $X_4(s) = \frac{-4}{(s-10.6)}$

(e) $X_5(s) = \frac{6.5}{(4.5s+1)}$

(f) $X_6(s) = \frac{2}{(0.01s+1)}$

(g) $X_7(s) = \frac{10}{(s^2+64)}$

(h) $X_8(s) = \frac{s^2+10s+24}{(s^3-4s^2+64s-256)}$

Q10.2 Using an \times to mark a pole position in the s -domain, make an s -domain sketch and plot the position of the poles found in Laplace transforms in parts (a)–(h) of the previous problem. Label each pole position with the appropriate feature found in the time-domain signal from which the transform was obtained. If there are any zeros, mark their position with a \circ .

Q10.3 In each of the following:

(a) $G(s) = \frac{1}{2s^2 + 3s + 1}$ (b) $G(s) = \frac{25}{16s^2 + 24s + 25}$ (c) $G(s) = \frac{1}{s^2 + 0.8s + 1}$

- (i) Determine ζ , ω_n and K
- (ii) Draw the s -domain plot for the system poles
- (iii) Sketch the appropriate unit step response

Q10.4 What are the poles of the following systems? Comment on the stability of each system.

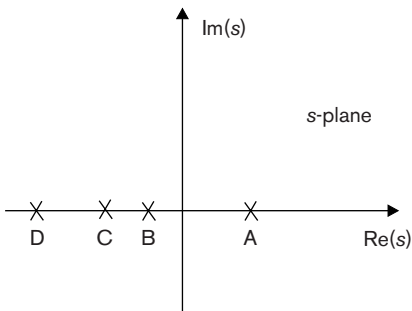
(a) $G(s) = \frac{3}{4s + 1}$ (b) $G(s) = \frac{6}{2s + 3}$
 (c) $G(s) = \frac{10}{s - 4}$ (d) $G(s) = \frac{3}{s^2 + 3s + 9}$
 (e) $G(s) = \frac{6}{s^2 - 3s + 4}$ (f) $G(s) = \frac{10}{s^2 + 2s + 1}$

Q10.5 For the pole map shown:

- (a) comment on the stability of the various first-order systems, A , B , C and D . Assume each system is of the form

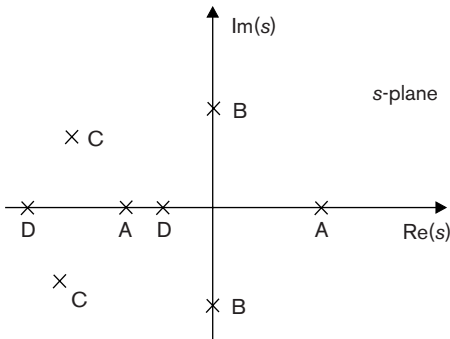
$$G(s) = \frac{1}{s - p}, \quad p = A, B, C \text{ or } D$$

- (b) Which system would have the fastest stable response?



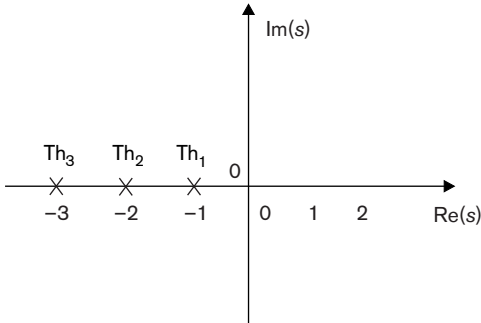
Q10.6 For the pole map shown:

- (a) comment on the stability of systems whose poles are A , B , C and D .
- (b) Comment on the response to a step input for systems A , B , C and D .

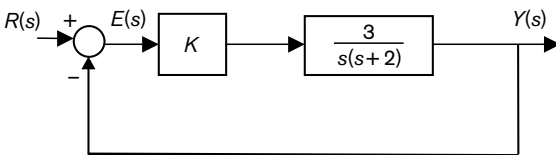


Problems

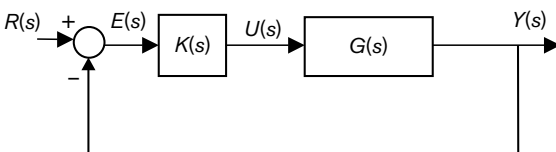
P10.1 A diagram of the pole positions of three thermocouples is given. You have to determine which thermocouple responds fastest to a change in temperature. Determine also the associated time constant for each thermocouple.



P10.2 The system shown represents the position control system for a robot cutting arm within a manufacturing system which produces clothing. The production manager has insisted that in order to save material wastage, there should be no overshoot to any requested change in cutting position. What value of controller K would satisfy this requirement?



P10.3 A typical feedback control system is shown.



The system transfer function has three poles at $s = 0$, $s = -2$, $s = -1$. It has a gain of 5. The controller is a simple gain, where $U(s) = KE(s)$.

- Draw the block diagram, substituting in appropriate transfer functions.
- Is the open-loop system stable?
- Using MATLAB, comment on the stability of the closed-loop system for $K = 0.3, 0.5, 0.7$ and 0.9 .

P10.4 An open-loop system is given by

$$G(s) = \frac{(2s+1)}{(0.5s+1)(12s+1)}$$

The system is placed in a unity feedback loop with a controller $K(s) = (10s+1)/s$ in the forward path.

- What are the system zeros and poles?
- What are the open-loop zeros and poles?
- Using MATLAB, determine the closed-loop zeros and poles. Are any the same as those of the system or open loop?

P10.5 Given a system of the form

$$G(s) = \frac{1 - (s / \alpha)}{(0.333s + 1)(0.25s + 1)}$$

use MATLAB to investigate how the unit step response changes for the zero at $s = \alpha$ taking values of $s = 2, 4, 7, 10$. Comment on your results.

11 Three-term control: PID control



PID or three-term controllers are widely used in industry. For example, in a typical paper mill there may be about 2000 control loops, and over 90% of these would be PID control loops. The PID name comprises the first letters of the three terms which make up this controller:

P stands for the **P**roportional term in the controller

I stands for the **I**ntegral term in the controller

D stands for the **D**erivative term in the controller

In the mid-1900s, several technological and analysis concepts came together to make the PID controller widely accepted as an effective industrial controller.

■ *Technology*

The electronic amplifiers of the time allowed proportional, integrator and differentiator units to be made fairly easily with good reliability. Thus the PID controller could be made using analogue electronic components. This ability to *manufacture* PID controllers made it the first really successful controller product with a high volume market.

■ *Analysis*

The Laplace transform formalism had begun to be understood and used by engineers to study the performance of op-amps and hence the performance of PID control. These simple Laplace transform links, like $[1/s]$ to represent an Integrator and $[s]$ to represent a Differentiator, helped to establish a theoretical basis for analysing a PID control design. Figure 11.1 shows the three links for the PID controller. Figure 11.1 (a) shows the PID blocks, (b) shows the time function forms of the blocks and finally (c) shows the Laplace function forms of the blocks.

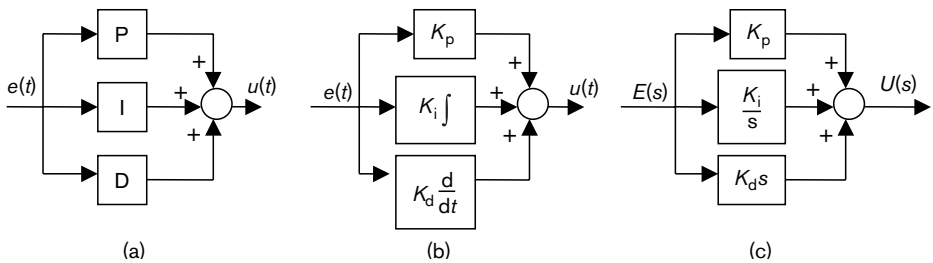


Figure 11.1 PID controller blocks.

In this chapter we are going to learn about the different effects of the three controller terms on the closed-loop system response.

Learning objectives

- To understand the system effects of each of the terms, P, I and D in three-term control.
- To be able to perform simple closed-loop analysis for the different combinations of the terms in PID controllers.
- To investigate the effect of the three terms on disturbance signals.

11.1 Controller assessment framework

To understand and assess the effects of the different terms of the controller, we use a standard transfer function block diagram (Figure 11.2), which has a reference signal, $R(s)$, and a disturbance signal, $D(s)$.

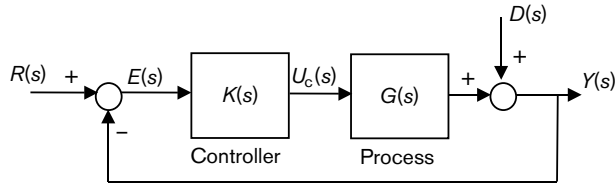


Figure 11.2 Unity feedback closed-loop control.

The controller is represented by $K(s)$ and the plant or process transfer function model by $G(s)$. The output $Y(s)$ is desired to follow the reference signal, $R(s)$. Typically the diagram represents a real system where signal $D(s)$ represents the extra loading demands made on the process. For the purposes of good control, we require the process output, $Y(s)$, to track only the reference signal, $R(s)$, and attenuate or reject the load disturbance signal, $D(s)$. This is why we discuss and assess the performance of the control system in terms of:

(a) reference tracking performance

and

(b) load disturbance rejection performance

Example: An industrial furnace and a load disturbance

A gas burner supplies heat to a furnace which is used for firing ceramic components (Figure 11.3). The reference signal, or set point signal, is the desired value of the furnace temperature needed for firing the ceramics. For example, in the time domain the reference signal is $r(t) = T_{\text{set point}} = 500^\circ\text{C}$, or as a Laplace transform the reference would be $R(s) = 500/s$.

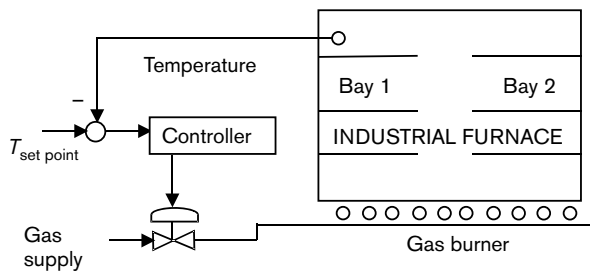


Figure 11.3 Industrial furnace.

Load disturbance signal: Suppose the operator loads the shelves of Bay 1 with ceramics and then switches on the gas burner and sets the reference temperature to 500°C . Provided we have a good control design, the furnace temperature will steadily climb to the reference level. If, one hour later, a second batch of ceramics arrive for firing, the operation of filling the shelves of Bay 2 will create a load disturbance since the furnace door will have to be opened and cold ceramics

will be placed on the shelves. The furnace now has double the loading (quantity) of ceramics to fire and heat up. This is how a typical load disturbance occurs. Note that the temperature reference level of 500 °C has not changed, but the controller has to act to generate more heat to combat the temperature drop caused by the extra demand for heat placed on the furnace. We represent this situation by a load disturbance signal acting on the process output.

11.1.1 Alternative terms

If a control system has to be designed to follow a frequently changing reference signal, this is sometimes referred to as a **servo-control design problem**.

If the reference or set point signal is held at a steady value for long periods, and the control system has to reject disturbance signals, this type of control design problem is sometimes called a **regulator control design problem**.

11.1.2 General block diagram analysis

Starting from the closed-loop system output, $Y(s)$, in Figure 11.2, we work round the loop anticlockwise to obtain

$$\begin{aligned} Y(s) &= D(s) + G(s)K(s)(R(s) - Y(s)) \\ &= G(s)K(s)R(s) + D(s) - G(s)K(s)Y(s) \\ [1 + G(s)K(s)]Y(s) &= G(s)K(s)R(s) + D(s) \end{aligned}$$

giving

$$\begin{aligned} Y(s) &= \frac{G(s)K(s)}{1 + G(s)K(s)} R(s) + \frac{1}{1 + G(s)K(s)} D(s) \\ &= G_{CL}(s)R(s) + S(s)D(s) \end{aligned}$$

The system output comprises two effects and we show this as a reduced system diagram (Figure 11.4). This has two paths: one due to reference tracking and one due to load disturbance rejection.

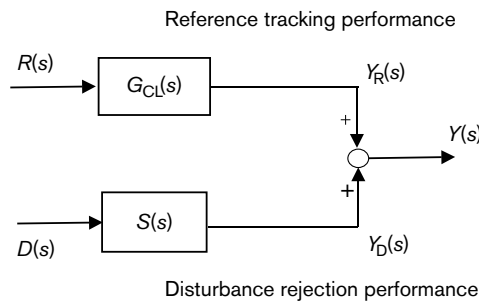


Figure 11.4 Reduced system form of the closed-loop system.

Key result: Reference tracking and disturbance rejection

To assess reference tracking performance we use the transfer function from $R(s)$ to $Y(s)$ with $D(s) = 0$, namely

$$Y_R(s) = \frac{G(s)K(s)}{1 + G(s)K(s)} R(s) \quad \text{Reference tracking performance}$$

For disturbance rejection performance assessment we use the transfer function from $D(s)$ to $Y(s)$ with $R(s) = 0$, namely

$$Y_D(s) = \frac{1}{1+G(s)K(s)} D(s) \quad \text{Disturbance rejection}$$

Using the principle of superposition (Step 4, Chapter 2) we can combine these separate effects to give the overall performance of the control system in the process output as

$$Y(s) = Y_R(s) + Y_D(s)$$

and

$$\begin{aligned} Y(s) &= \frac{G(s)K(s)}{1+G(s)K(s)} R(s) + \frac{1}{1+G(s)K(s)} D(s) \\ &= G_{CL}(s)R(s) + S(s)D(s) \end{aligned}$$

Control design is difficult because we note that

$$G_{CL}(s) + S(s) = \frac{G(s)K(s)}{1+G(s)K(s)} + \frac{1}{1+G(s)K(s)} = 1$$

Thus, we cannot change the reference tracking or disturbance rejection properties independently of one another. This will be highlighted in the examples in this chapter.

Our closed-loop system analysis will look at three properties.

A Closed-loop stability

We perform this analysis only once because we have the same *denominator* in the two transfer functions, $G_{CL}(s)$ and $S(s)$. We can see this from the following simple analysis

$$G_{CL}(s) = \frac{G(s)K(s)}{1+G(s)K(s)} \quad \text{and} \quad S(s) = \frac{1}{1+G(s)K(s)}$$

Set $G(s) = n(s)/d(s)$ and $K(s) = n_K(s)/d_K(s)$; then

$$G_{CL}(s) = \frac{G(s)K(s)}{1+G(s)K(s)} = \frac{n(s)n_K(s)}{d(s)d_K(s) + n(s)n_K(s)} = \frac{n_{CL}(s)}{d_{CL}(s)}$$

$$S(s) = \frac{1}{1+G(s)K(s)} = \frac{d(s)d_K(s)}{d(s)d_K(s) + n(s)n_K(s)} = \frac{n_s(s)}{d_{CL}(s)}$$

The closed-loop stability test is to see whether the pole positions coming from the characteristic polynomial $d_{CL}(s)$ are all in the Left Half Plane of the s -domain. These pole positions will be found as the roots of the equation $d_{CL}(s) = 0$. We also know that the position of the closed-loop poles will give us an intuitive idea of the speed of response in the output signals. We associate poles which have a more negative real part with an increasingly fast speed of response.

B Reference tracking performance

Here we look at the shape of the time response in the output signal. In the transient part of the response, we look to see the speed of response and the size of any overshoot. In the steady state part of the output signal we look to see if any steady offset is present.

C *Disturbance rejection performance*

We look at the shape of the output signal. We look at speed of response, we look at the size of the disturbance peak value, and we look to the steady state region of the output to make sure that no offset exists after the process disturbance transients have died away.

11.2 Proportional control

The desired output of a closed-loop system is known as the reference signal or the set point signal. The controller will use the error, or difference between the reference signal and the measured output signal. We use proportional control when we want the controller action to be proportional to the size of this error signal. Figure 11.5 shows the time and Laplace domain representations for proportional control:

Time domain: $u_c(t) = K_p e(t)$

Laplace domain: $U_c(s) = K_p E(s)$

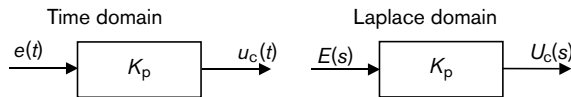


Figure 11.5 Proportional control block diagrams.

Figure 11.6 shows a process with a proportional controller within the closed loop.

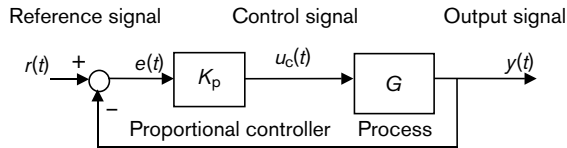


Figure 11.6 Closed-loop proportional control.

Then the following definitions apply:

Error signal: $e(t) = r(t) - y(t)$

Controller signal: $u_c(t) = K_p e(t)$

Consider the situation where $r(t) = 1.0$ and $y(t) = 0.8$; then

$$e(t) = r(t) - y(t) = 1.0 - 0.8 = 0.2$$

and the controller signal is

$$u_c(t) = K_p \times 0.2$$

This is interpreted as 'the controller signal to drive the plant should be positive and proportional to the error of 0.2'. This controller signal will then increase the process output $y(t)$ and reduce the error of $e(t) = r(t) - y(t) = 0.2$. The controller tuning problem is

that of *selecting* a suitable proportional controller gain K_p to meet desired closed-loop stability and output response specifications.

We will use the unity feedback block diagram (Figure 11.7) to help us to assess the reference tracking and disturbance rejection of the different controller types. Here the diagram is used with a proportional controller in place. Note that we have both the reference signal, $R(s)$, and the load disturbance, $D(s)$, present in the analysis.

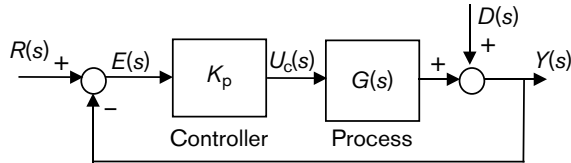


Figure 11.7 Unity feedback block diagram for controller analysis.

11.2.1 First-order system analysis – proportional control

Consider the following transfer function description, which represents a heating system. The input signal is the power in kW from the heater and the output signal $Y(s)$ is the resulting temperature.

$$Y(s) = G(s)U(s)$$

where

$$G(s) = \frac{0.3}{2s+1}$$

The time constant of the process is given as 2 hours. The transducer has a very short time constant in comparison to the process time constant, so we can model the system as a unity feedback system (Figure 11.8). We let the step reference signal be $r(t) = 1$, so that $R(s) = 1/s$ and let the constant load disturbance effect be modelled by $d(t) = 0.5$ so that $D(s) = 0.5/s$.

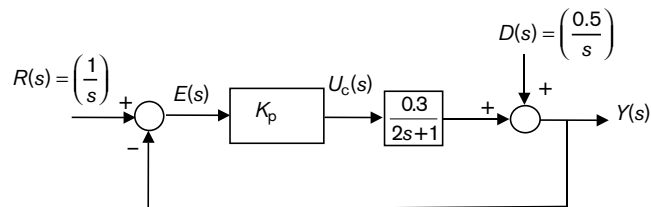


Figure 11.8 Heating control system with proportional control.

A: Closed-loop stability

The open-loop poles are obtained from the denominator of the open-loop transfer function

$$G(s) = \frac{0.3}{2s+1} = \frac{n_{OL}(s)}{d_{OL}(s)}$$

so that $d_{OL}(s) = (2s + 1) = 0$ implies that the single *open-loop pole* lies on the LHP at $s = -0.5$. This LHP location shows that the open-loop system is stable.

The closed-loop poles are obtained from the denominator of the closed-loop transfer function, $G_{CL}(s)$, where in this case:

$$G_{CL}(s) = \frac{0.3K_p}{2s+1+0.3K_p} = \frac{n_{CL}(s)}{d_{CL}(s)}$$

Thus, $d_{CL}(s) = (2s + 1 + 0.3K_p) = 0$ implies that the single *closed-loop pole* lies at the LHP location of $s = -0.5 - 0.15K_p$

Two important observations can be made:

1. If we select the proportional controller gain, K_p to be positive, then for all values $K_p > 0$ the closed-loop pole $s = -0.5 - 0.15K_p$ is in the LHP of the s -plane and the closed-loop system is stable.
2. At $K_p = 0$, the closed-loop pole has a value equal to the open-loop pole. As K_p is increased, the closed-loop pole moves deeper into the LHP. This has the effect of speeding up the closed-loop system response.

Table 11.1 shows the effect of increasing the value of K_p . The calculated locations of the closed-loop pole and the closed-loop system time constant show the closed-loop system pole becoming increasingly fast and hence the corresponding time response will also speed up.

Table 11.1 Comparison of changing proportional gain and closed-loop time constant.

K_p	Closed loop pole	Closed-loop time constant	Comments
0	-0.5	2	For an increasing gain, the closed loop time constant becomes smaller; hence the system response speeds up
1	-0.65	1.54	
4	-1.10	0.91	
10	-2.0	0.5	

We would term this behaviour as ‘an increasing controller gain is causing a faster closed-loop system response’. We can give this a nice interpretation using the s -domain and the system response curves (Figure 11.9).

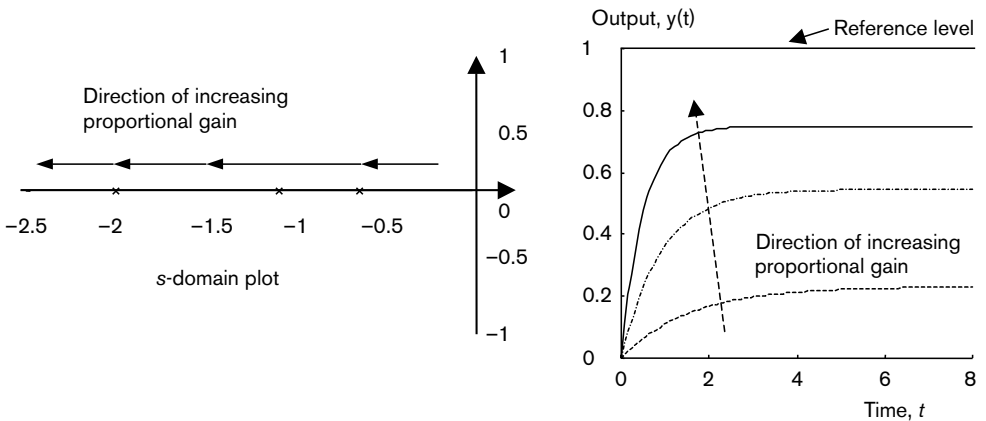


Figure 11.9 Comparison of increasing proportional gain and step responses.

B: Reference tracking performance

For a reference output of $r(t) = 1.0$, we now examine how well the system output, $y(t)$, actually follows this reference signal. We look at the steady state tracking performance using the Final Value Theorem. We can revise this theorem by looking at Chapter 2. The general reference tracking response result for proportional control K_p is given by

$$Y(s) = \frac{G(s)K_p}{1+G(s)K_p} R(s)$$

and the Final Value Theorem gives

$$y_{Rss} = \lim_{t \rightarrow \infty} y_R(t) = \lim_{s \rightarrow 0} sY_R(s) = \lim_{s \rightarrow 0} s \left[\frac{G(s)K_p}{1+G(s)K_p} \right] \frac{R}{s}$$

In this particular example

$$G(s) = \frac{0.3}{2s+1} \text{ and } R(s) = \frac{1}{s}$$

Thus

$$\begin{aligned} y_{Rss} &= \lim_{s \rightarrow 0} s \left[\frac{0.3K_p}{2s+1+0.3K_p} \right] \frac{1}{s} \\ &= \frac{0.3K_p}{1+0.3K_p} = \frac{1}{1+(1/0.3K_p)} \end{aligned}$$

If we complete a table of y_{Rss} values for the same range of proportional control gains that we used to assess the closed-loop stability we find the results shown in Table 11.2.

Table 11.2 Comparison of increasing proportional gain and decreasing e_{ss} .

K_p	y_{Rss}	r	$e_{ss} = r - y_{Rss}$	Comments
0	0	1	1.0	
1	0.23	1	0.65	Offset error decreases
4	0.55	1	0.45	as proportional
10	0.75	1	0.25	controller gain
∞	1.0	1	0.00	increases

This reduction in the steady state offset as the proportional gain is increased can be seen in the closed-loop unit step responses in Figure 11.9. We also plot the control signals for increasing values of K_p (Figure 11.10). The control signal *increases* as proportional gain *increases*.

Two observations can be made:

1. Increasing the size of the proportional gain has two effects on the system output. The speed of the response is increased and the offset from the desired output level is reduced. This offset can only be completely removed by allowing the proportional gain to become infinitely large.

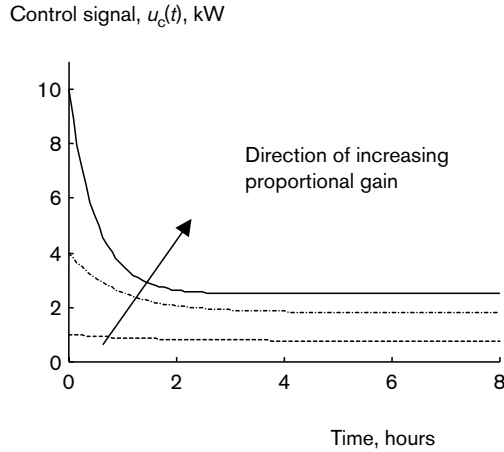


Figure 11.10 Control signal for increasing proportional gain.

- Increasing the proportional term to achieve the desired effects of faster output response and significantly reduced steady state offset causes the controller signal $u_c(t)$ to become increasingly large. This means that the demanded movement of the system actuator could be too large, possibly reaching physical system limits. Thus, too large a proportional gain may not be able to achieve the theoretical improvement because the hardware of the actuator cannot actually deliver the required control signals.

C: Disturbance rejection performance

The principle of superposition allows us to consider the disturbance rejection performance in the closed-loop system output separately from reference tracking performance. As before, the process is modelled by $G(s) = 0.3/(2s + 1)$ and we use the disturbance rejection block diagram of Figure 11.11 for this analysis.

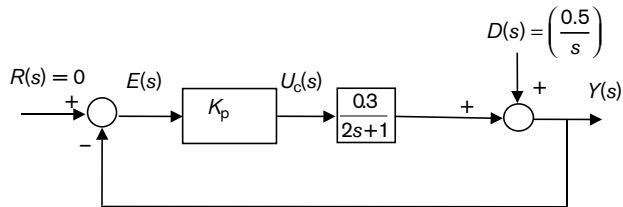


Figure 11.11 Closed-loop system for disturbance rejection analysis.

The general expressions are

$$Y_D(s) = S(s)D(s) = \frac{1}{1 + G(s)K(s)} D(s)$$

Substitute for the system transfer function, and the disturbance $D(s)$, where

$$G(s) = \frac{0.3}{2s+1} \quad \text{and} \quad D(s) = \frac{0.5}{s}$$

to obtain

$$Y_D(s) = S(s)D(s) = \frac{2s+1}{2s+1+0.3K_p} \left(\frac{0.5}{s} \right)$$

Closed-loop stability

Note that the denominator of the disturbance rejection closed-loop transfer function $S(s)$ is identical to that of the reference tracking closed-loop transfer function $G_{CL}(s)$. Hence there is no need to perform a second analysis of the closed-loop system stability because it is the same as that already performed and the same effects will be found:

1. The ability to place the closed-loop pole deeper and deeper into the LHP by increasing the size of the proportional gain.
2. A corresponding increase in the speed of the closed-loop output response $y_D(t)$ as the proportional gain is increased.

The disturbance response, $y_D(t)$

We can investigate the disturbance response and the disturbance rejection performance using simulation or analysis methods.

Method 1: Simulink study of disturbance rejection responses

For the closed-loop system diagram of Figure 11.11:

- (a) Set up the Simulink simulation for this feedback system.
- (b) Plot the disturbance rejection response curves $y_D(t)$ for the proportional gains $K_p = 1, 4, 10$.
- (c) Deduce the link between $y_{D_{SS}}$ and the value of K_p from the graph obtained.

Solution (a) The Simulink diagram is shown in Figure 11.12.

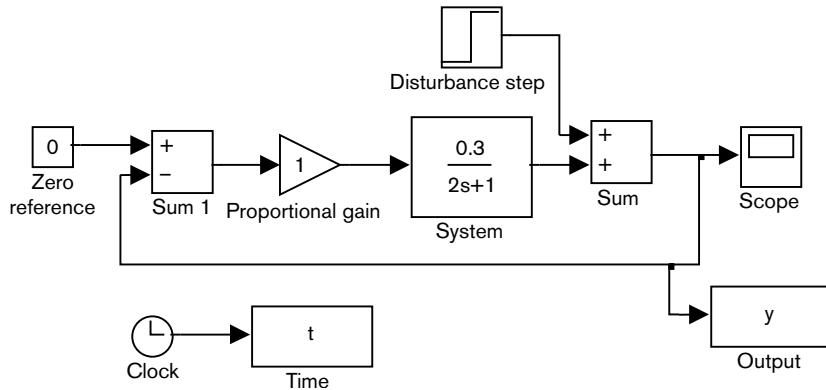


Figure 11.12 Simulink diagram of first-order system with proportional control.

- (b) A plot of $y_D(t)$ for increasing proportional gain is shown in Figure 11.13.
- (c) Table of $y_{D_{SS}}$ versus K_p .
Using MATLAB to determine the end point (steady state value) of the response curves, we find the values given in Table 11.3. We deduce that increasing the proportional gain reduces the steady state offset from zero.

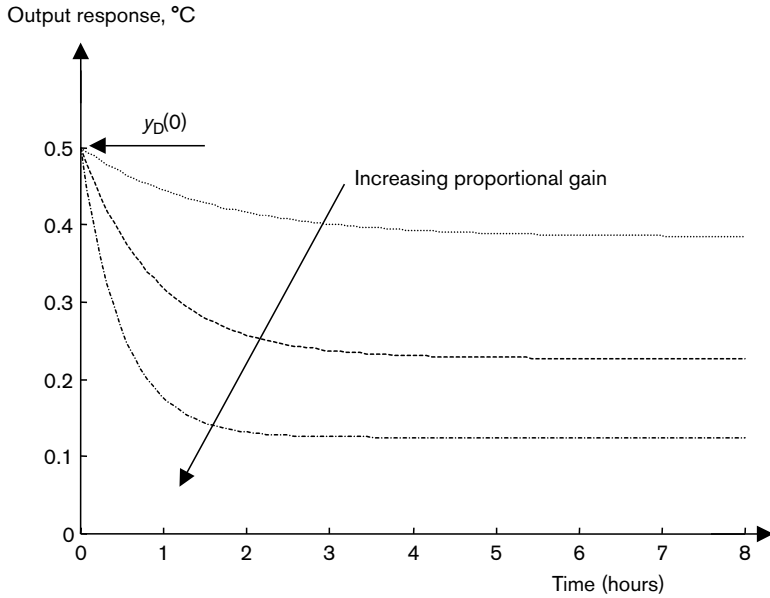


Figure 11.13 Output responses to input disturbance signal for increasing proportional gain.

Table 11.3 Offset values versus proportional gain

K_p	y_{Dss}
1.0	0.3836
4.0	0.2262
10.0	0.1259

Method 2: Disturbance rejection analysis using Laplace transforms

To see the effect of the proportional gain on the speed of response, and steady state value of the disturbance response we firstly calculate the time response $y_D(t)$.

Calculation of $y_D(t)$ from $Y_D(s)$

Using the disturbance rejection block diagram (Figure 11.11), the closed-loop transfer function analysis is:

$$Y_D(s) = S(s)D(s) = \frac{1}{1+G(s)K(s)} D(s)$$

Substituting for

$$G(s) = \frac{0.3}{2s+1}, \quad K(s) = K_p \quad \text{and} \quad D(s) = \frac{0.5}{s}$$

obtains

$$Y_D(s) = \left[\frac{2s+1}{2s+1+0.3K_p} \right] \left(\frac{0.5}{s} \right) = \left[\frac{s+0.5}{s+0.5+0.15K_p} \right] \left(\frac{0.5}{s} \right)$$

From the Laplace transform $Y_D(s)$ we need the time domain response, $y_D(t)$. To compute this we use Laplace Transform tables directly using the identity:

$$\mathcal{L}^{-1} \left\{ \frac{(s+\alpha)}{(s+a)(s+b)} \right\} = \frac{1}{(b-a)} [(\alpha-a)e^{-at} - (\alpha-b)e^{-bt}]$$

The result is obtained as follows:

$$Y_D(s) = \left[\frac{s+0.5}{s+0.5+0.15K_p} \right] \left(\frac{0.5}{s} \right) = 0.5 \left[\frac{s+0.5}{s(s+0.5+0.15K_p)} \right]$$

Identify $\alpha = 0.5$, $a = 0$, and $b = 0.5 + 0.15K_p$, giving

$$\begin{aligned} y_D(t) &= (0.5) \left[\frac{1}{(b-a)} [(\alpha-a)e^{-at} - (\alpha-b)e^{-bt}] \right] \\ &= (0.5) \left[\frac{1}{(0.5+0.15K_p)} [(0.5-0)e^{-0t} - (0.5-0.5-0.15K_p)e^{-(0.5+0.15K_p)t}] \right] \\ &= \frac{0.25}{(0.5+0.15K_p)} [1 + 0.3K_p e^{-(0.5+0.15K_p)t}] \end{aligned}$$

Now that $y_D(t)$ is available as an explicit function of the proportional gain K_p , the shape of the disturbance response can be explained and the dependence of the steady state value y_{Dss} studied.

Disturbance response, $y_D(t)$ – general shape

A look back at Figure 11.13, produced by the simulation study, and using the expression for $y_D(t)$, shows the following disturbance response features:

- (a) Let $t = 0$; then $y_D(0) = 0.5$ and is independent of K_p .
- (b) As K_p becomes larger, the quantity $-(0.5 + 0.15K_p)$ becomes more negative, so that the transient part of the response, which depends on $e^{-(0.5+0.15K_p)t}$, becomes faster.
- (c) The steady state value of y_{Dss} is given by

$$y_{Dss}(K_p) = \frac{0.25}{0.5+0.15K_p}$$

and the steady state value of the disturbance response depends on K_p . Increasing K_p reduces the steady state offset.

Finally, as $K_p \rightarrow \infty$ we see that

$$y_{Dss} = \lim_{K_p \rightarrow \infty} y_{Dss}(K_p) = \lim_{K_p \rightarrow \infty} \left(\frac{0.25}{0.5+0.15K_p} \right) = 0$$

Key result: Proportional control

- The proportional gain K_p is taken to be positive so that $K_p > 0$.
- Increasing the proportional action:
 - means increasing the numerical value of the proportional gain, K_p
 - speeds up the transient portion of the reference tracking response and the transient portion of the load disturbance response
 - decreases but does not entirely remove or eliminate the output offset from the desired reference value
 - decreases but does not entirely eliminate the offset in the output due to the constant load disturbance
 - generally increases the size of the control signal which achieves good reference tracking and disturbance rejection in the system output
 - may cause the controller signal to be too large which may lead to saturation or limiting problems with the system actuators

11.3 Integral control

Integral control is denoted by the I-term in the PID controller and we use an associated integral controller gain, K_i . Integral control is used when we want the controller action to correct for any steady and continuing offset from a desired reference signal level. Integral control overcomes the shortcoming of proportional control by eliminating offset without the use of an excessively large controller gain. The time and Laplace representations are shown in Figure 11.14.

Time domain:
$$u_c(t) = K_i \int^t e(\tau) d\tau$$

Laplace domain:
$$U_c(s) = \left[\frac{K_i}{s} \right] E(s)$$

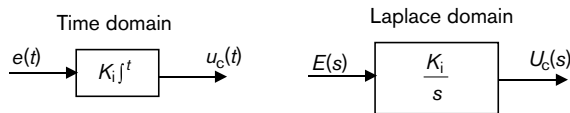


Figure 11.14 Integral control block diagram.

How does integral control achieve what proportional control could not accomplish? Let us look at Figure 11.15.

In Figure 11.15(a) we see the typical unit step response with offset that might be given by proportional control of a first-order system. The error signal $e(t) = r - y(t)$ is shown in Figure 11.15(b). Here the area under the error signal is shaded. Integral control sums up this shaded area. As you can see in Figure 11.15(c), the integral term, $\int^t e(\tau) d\tau$, shows an increasingly large signal in the time region where the steady state offset exists. What integral control manages to do is to increase the size of the control signal whenever a steady offset from a desired reference level starts to occur. This is rather like an automatic procedure to ‘increase’ the controller gain in the presence of steady offsets, and the controller output

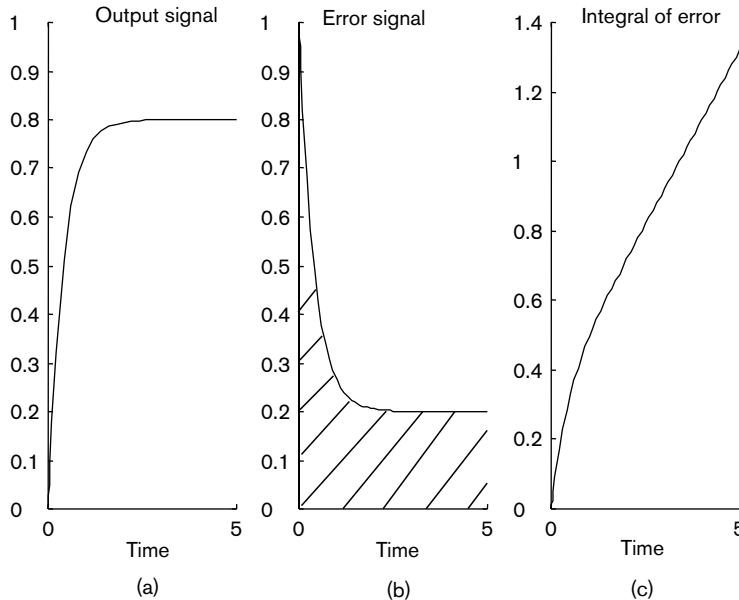


Figure 11.15 (a) Output response; (b) error signal; (c) integral of error.

increases until the offset is eliminated. As you can see, this *guaranteed* property of integral control is going to be very useful indeed in practical industrial control systems.

To give this intuitive insight about integral control a formal basis, we use the familiar analysis framework for reference tracking and disturbance rejection.

11.3.1 First-order system analysis – integral control

The analysis of integral control of a process modelled by a first-order system examines the ability of the system to track the reference signal. This uses the transfer function relationship for $Y_R(s)$. We will also see how integral control performs in rejecting process load disturbance upsets using the transfer function relation for $Y_D(s)$.

We use the heating process that we have previously studied for proportional control.

$$G(s) = \frac{0.3}{2s+1}$$

Let the reference signal be a unit step so that $R(s) = 1/s$ and let the load disturbance effect be modelled by a constant disturbance signal, $D(s) = 0.5/s$. An integral controller is to be used, so we have

$$K(s) = \frac{K_i}{s} \quad K_i > 0$$

and we put all this information on one diagram (Figure 11.16).

Let us first evaluate the two important transfer function relationships, $G_{CL}(s)$ and $S(s)$ for the modelling data in Figure 11.16. Substitute for $G(s)$ and $K(s)$ to find:

$$G_{CL}(s) = \frac{G(s)K(s)}{1+G(s)K(s)} = \frac{0.3K_i}{s(2s+1)+0.3K_i} = \frac{n_{CL}(s)}{d_{CL}}$$

$$S(s) = \frac{1}{1+G(s)K(s)} = \frac{s(2s+1)}{s(2s+1)+0.3K_i} = \frac{n_s(s)}{d_{CL}(s)}$$

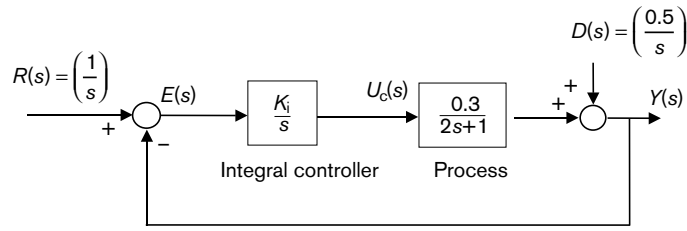


Figure 11.16 Closed-loop system with integral control.

Closed-loop stability

To assess the closed-loop stability we identify the common characteristic equation from the transfer function $G_{CL}(s)$ and $S(s)$ as:

$$d_{CL}(s) = s(2s+1) + 0.3K_i = 2s^2 + s + 0.3K_i = 0$$

Using the usual formula to solve a quadratic, we have

$$s = \frac{-1 \pm \sqrt{1^2 - 4 \times 2 \times 0.3K_i}}{2 \times 2}$$

$$= -0.25 \pm 0.25(1 - 2.4K_i)^{1/2}$$

Firstly we note that if $K_i = 0$ (that is, we have broken the feedback loop), the poles are the open-loop poles given by $s = -0.5$ and $s = 0$.

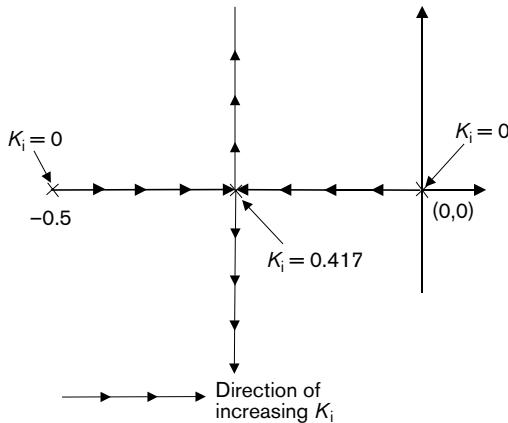
Now we consider the roots of the quadratic for K_i not equal to zero.

Whether $(1 - 2.4K_i)$ is $< 0, = 0, > 0$	Corresponding value of K_i	Roots	Real/complex?
$(1 - 2.4K_i) > 0$	$0 < K_i < 0.417$	$-0.25 \pm 0.25\sqrt{1 - 2.4K_i}$	Real
$(1 - 2.4K_i) = 0$	$K_i = 0.417$	-0.25 and -0.25	Real
$(1 - 2.4K_i) < 0$	$K_i > 0.417$	$-0.25 \pm j 0.25\sqrt{2.4K_i - 1}$	Complex

If the closed-loop system poles are plotted on the s -plane for the different ranges of K_i , we can obtain a graphical s -domain interpretation of the stability of closed-loop integral control for this process (Figure 11.17). We are also able to make some inspired guesses about the type of output responses available from the closed-loop control system. These guesses use our knowledge of second-order systems and will be related to the value of integral control gain K_i chosen.

The analysis for the closed-loop system stability has revealed two important results:

1. The closed-loop system is stable for all the integral controller gain values, $K_i > 0$.
2. The range of responses obtainable covers all the three types of second-order system response shapes. A particular output response shape can be obtained by selecting a particular integral gain value. From the diagram and the analysis the following links can be seen:

**Predictions**

- Closed loop system stable for all values of integral control gain K_i
- $0 < K_i < 0.417$
Poles on real axis
Responses overdamped
- $K_i = 0.417$
Two equal real poles
Response critically damped
- $K_i > 0.417$
Two complex poles
Response underdamped

Figure 11.17 s-domain plots of closed-loop pole position.

- (a) $0 < K_i < 0.417$ Overdamped
 (b) $K_i = 0.417$ Critically damped
 (c) $K_i > 0.417$ Underdamped

Reference tracking performance

We study the reference tracking performance by looking at an exercise.

Problem: Reference tracking responses

For Figure 11.16 with a reference signal of $R(s) = 1/s$ and a disturbance signal set to zero:

- (a) Derive the general closed-loop transfer function and find an expression for the damping ratio, ζ .
- (b) Implement the closed-loop system in Simulink and find the output response for the following different values of K_i .
- (i) $K_i = 0.25$ (ii) $K_i = 0.417$ (iii) $K_i = 1.0$
- (c) Using the expression for damping ratio found in (a), find the damping ratios for the three different closed-loop systems given by the three values of K_i . Correlate your step responses from Simulink to the damping ratios just calculated.
- (d) For $K_i = 1.0$, find also the natural frequency and the damped natural frequency of the system.
- (e) For the three step responses, what is the steady state offset?
- (f) Repeat the three Simulink simulations but use proportional control with gains of
- (i) $K_p = 0.25$ (ii) $K_p = 0.417$ (iii) $K_p = 1.0$

Compare the general shape of the proportional control responses and the steady state offset with those obtained using integral control. Comment on the results found.

Solution The solution to the above problem involves skills of transfer function analysis and simulation studies.

- (a) Evaluate the closed-loop damping ratio for the system with integral control.

$$G_{CL}(s) = \frac{0.3K_i}{s(2s+1)+0.3K_i}$$

We put this in 'unity constant coefficient form':

$$G_{CL}(s) = \frac{1}{(2/0.3K_i)s^2 + (1/0.3K_i)s + 1}$$

If we associate the denominator with the denominator of a standard second-order transfer function,

$$\frac{1}{(1/\omega_n^2)s^2 + (2\zeta/\omega_n)s + 1}$$

and equate coefficients of powers of s, we find

$$\frac{2}{0.3K_i} = \frac{1}{\omega_n^2} \text{ giving } \omega_n = \sqrt{\frac{0.3K_i}{2}}$$

and

$$\frac{2\zeta}{\omega_n} = \frac{1}{0.3K_i} \text{ giving } \zeta = \frac{\omega_n}{0.6K_i} = \frac{1}{2\sqrt{0.6K_i}}$$

- (b) The Simulink model is shown in Figure 11.18.

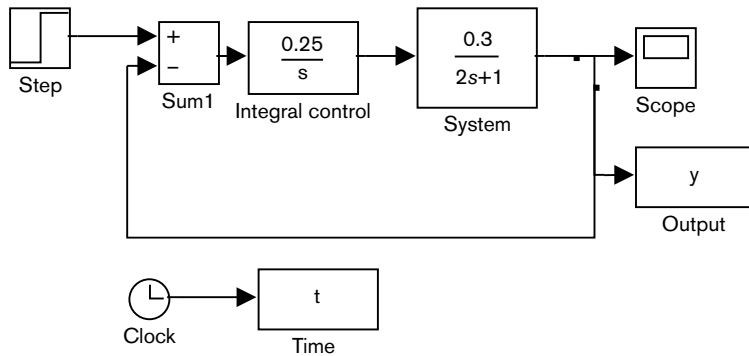


Figure 11.18 Simulink representation of closed-loop system with integral control.

- (c), (d) We need to use knowledge of second-order system transfer functions to find various closed-loop system parameters. We show the working for the case where $K_i = 1.0$ and find:

The natural frequency

Using the formula in (a) above and letting $K_i = 1$,

$$\omega_n = \sqrt{0.3K_i/2} = \sqrt{0.3/2} = 0.39 \text{ rad/s}$$

The damping ratio

The formula in (a) gives $\zeta = 1/(2\sqrt{0.6K_i}) = 1/(2\sqrt{0.6}) = 0.65$.

The damped natural frequency

Using the general formula $\omega_d = \omega_n \sqrt{1 - \zeta^2}$ gives $\omega_d = 0.39 \sqrt{1 - 0.65^2} = 0.30$ rad/s.

- (e) The integral control closed-loop traces can be found from the Simulink model runs. We should find the three traces shown in Figure 11.19. We can find using MATLAB that there is no steady state error.

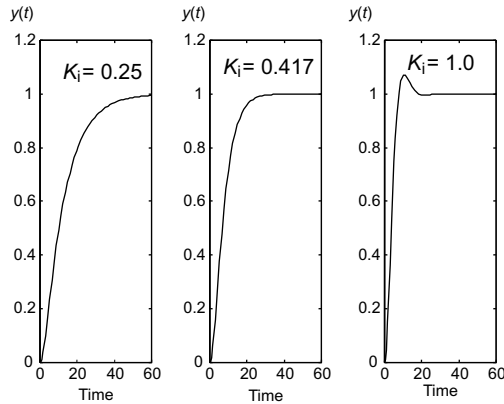


Figure 11.19 Output of closed-loop system for different values of integral gain.

- (f) Repeating the exercise with proportional gains will give us a steady state error in the system.

In the example above, simulation was used to look at the general shape of the transient portion of the reference tracking response. It was clearly shown that a much wider variety of system responses is obtainable. For example, comparing these responses with those produced by the proportional control analysis it could be seen that integral control offers more potential response shapes than proportional control. However, a significant difference occurs when we look at the steady state portion of the output response. In all three examples of the exercise, no offset in steady state was observed under integral control. This means that $y_{ss} = 1$. Proof of this valuable property of integral control uses the Final Value Theorem, and we are going to present two versions for the model data presented above.

Method A: using output response forms

We would like to show that, given $R(s) = 1/s$, $y_{ss} = 1$ and hence $e_{ss} = 0$. We use the output response of the reference tracking system with $D(s) = 0$.

$$\begin{aligned} y_{ss} &= \lim_{t \rightarrow \infty} y(t) = \lim_{s \rightarrow 0} sY(s) = \lim_{s \rightarrow 0} sG_{CL}(s) \left(\frac{1}{s} \right) = \lim_{s \rightarrow 0} G_{CL}(s) \\ &= \lim_{s \rightarrow 0} \frac{0.3K_i}{s(2s+1) + 0.3K_i} = \frac{0.3K_i}{0.3K_i} = 1 \end{aligned}$$

Method B: using offset or error response forms

We use the output response of the reference tracking system response with $D(s) = 0$, and our objective is to show $e_{ss} = 0$.

Recall that $e(t) = r - y(t)$ and hence

$$E(s) = R(s) - Y(s) = \left(\frac{1}{s} \right) - Y(s)$$

Substitute for

$$Y(s) = G_{CL}(s)R(s) = \left[\frac{0.3K_i}{s(2s+1) + 0.3K_i} \right] \left(\frac{1}{s} \right)$$

and

$$\begin{aligned} E(s) &= \left(\frac{1}{s} \right) - \left[\frac{0.3K_i}{s(2s+1) + 0.3K_i} \right] \left(\frac{1}{s} \right) \\ &= \left(\frac{2s+1}{s(2s+1) + 0.3K_i} \right) \end{aligned}$$

Thus

$$e_{ss} = \lim_{t \rightarrow \infty} e(t) = \lim_{s \rightarrow 0} sE(s) = \lim_{s \rightarrow 0} s \left(\frac{2s+1}{s(2s+1) + 0.3K_i} \right) = \lim_{s \rightarrow 0} s \frac{1}{0.3K_i} = 0$$

Remark

We showed that the steady state error was zero for our particular example. The same is true for a general system with an integral controller. The proof of this for a general system as shown in Figure 11.20 is given in the exercises.

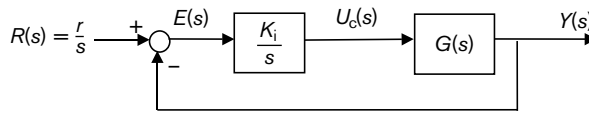


Figure 11.20 Closed-loop system for integral control tracking analysis.

Disturbance rejection performance

The starting point for the analysis of the closed-loop stability was that the characteristic equation for both transfer functions $S(s)$ and $G_{CL}(s)$ were the same and hence that the stability analysis does not have to be repeated. As for the first-order system examples we use:

Reference tracking performance

$$Y_R(s) = G_{CL}(s)R(s) = \frac{0.3K_i}{s(2s+1) + 0.3K_i} R(s)$$

Disturbance rejection performance

$$Y_D(s) = S(s)D(s) = \frac{s(2s+1)}{s(2s+1) + 0.3K_i} D(s)$$

Examine $S(s)$ and $G_{CL}(s)$ closely and we see an important design constraint. If we choose K_i to achieve a good reference tracking response, $Y_R(s)$, then we automatically fix the closed-loop pole positions *and* the related response shape for the disturbance rejection response, $Y_D(s)$. It may be that this *fixed* disturbance rejection response is not a very good one for the real system. If we do the design the other way round and choose K_i to achieve a good disturbance rejection response then we automatically fix the reference tracking response. This fixed tracking response may not be very effective. So we do have to remember this design interplay between the possibly different requirements of the two responses.

Problem: Disturbance rejection responses

Consider the system shown in Figure 11.21. For the heater model of this chapter, $G(s) = 0.3/(2s + 1)$ and for a disturbance input of $D(s) = 0.5/s$:

- (a) Construct a Simulink simulation capable of accepting a range of integral controller gains and recording the disturbance rejection response.
- (b) Run the simulations for the integral gains, $K_i = 1.0, 1.5$ and 3.0 and form a table of largest peak disturbance, the settle time for $|y_D(t)| < 0.01$, and the steady state offset.
- (c) Write an interpretation of the results found.

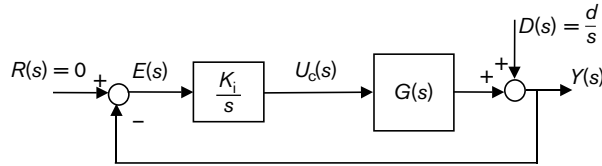


Figure 11.21 Feedback loop for disturbance rejection analysis.

Solution (a) The Simulink model should look like Figure 11.22.

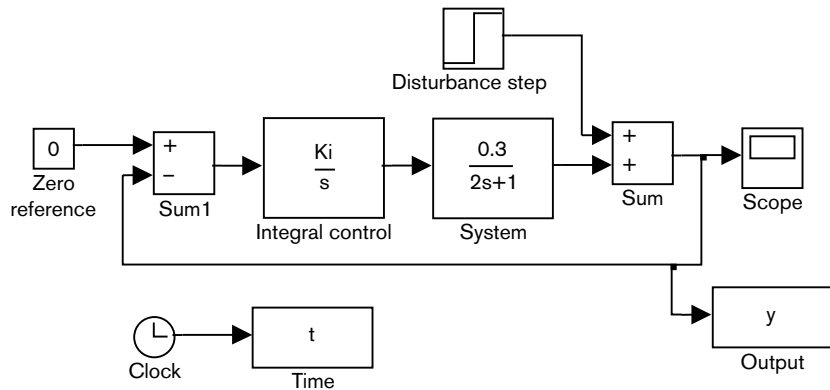


Figure 11.22 Simulink model for disturbance rejection analysis.

(b) The traces will look like those in Figure 11.23.

The table of values to be constructed will contain the following data:

Integral gain, K_i	Absolute magnitude of largest peak in $y_D(t)$	Settle time for $ y_D(t) < 0.01$	Size of offset
1.0	0.035	15.9	0
1.5	0.068	12	0
3.0	0.14	8	0

(c) The example illustrates the link between the type of output response and the closed-loop pole locations. Thus it is possible to see that the largest peak disturbance in the output

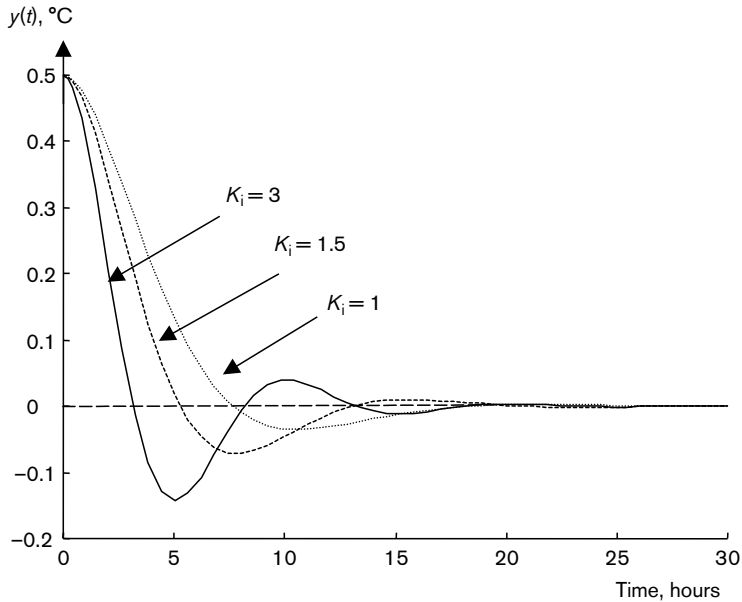


Figure 11.23 System output responses with different values of integral gain for disturbance input signal.

occurs for the case with the most underdamped pole locations. In some real applications a load disturbance peak of this size might not be permissible and a different tuning for K_i must be selected.

The second important property also shown in the exercise is that there is no offset in steady state and the static effect of the load disturbances is fully rejected. We prove this for the first-order system model as follows:

$$\begin{aligned}
 y_{Dss} &= \lim_{t \rightarrow \infty} y_D(t) = \lim_{s \rightarrow 0} sY_D(s) \\
 &= \lim_{s \rightarrow 0} sS(s)D(s) = \lim_{s \rightarrow 0} s \left[\frac{s(2s+1)}{s(2s+1) + 0.3K_i} \right] \left(\frac{0.5}{s} \right) \\
 &= \frac{0}{0.3K_i} = 0
 \end{aligned}$$

General disturbance rejection analysis

For Figure 11.21 and assuming that $G(s)$ has all poles in the LHP, we could form a general transfer function relationship for $Y(s)$ and show that for the general case the following limit holds:

$$y_{ss} = \lim_{t \rightarrow \infty} y(t) = \lim_{s \rightarrow 0} sY(s) = 0$$

Therefore the rejection of constant load disturbance signals is a general guaranteed property for the integral controller. In many practical cases, reference signal tracking performance is not so important, but the ability to keep process outputs at reference levels despite the appearance and disappearance of load disturbance signals is. This makes integral control especially valuable in industry.

Key result: Integral control

- The integral gain K_i is taken to be positive so that $K_i > 0$.
- Introducing integral action into a controller means incorporating an integral term $[K_i/s]$ into the controller.
- Increasing the amount of integral action in a controller means that the integral gain K_i has an increasing numerical value.
- The presence of integral action in a controller usually leads to a wider range of closed-loop system responses (sometimes even unstable ones, although this was not shown in this chapter).
- Selecting a value for integral gain K_i shapes the dynamics of both reference tracking and disturbance rejection responses.
- The presence of integral action in a controller eliminates constant offset signals in steady state for both reference tracking and disturbance rejection responses. This property is guaranteed and does not depend on using prior process gain modelling information.

11.4 Derivative control

If we want the controller to use the rate of change of the error signal in the control action, we use derivative control. This is the D-term in the PID or three-term controller. Derivative control is more specialised than proportional or integral control. For example, in real applications we cannot implement pure derivative action but need to use a modified transfer function representation, or a particular controller feedback structure (Chapter 18). However, derivative control does have some useful design features and is an essential term in some real-world control applications; for example, tachogenerator feedback in d.c. motor control is a form of derivative control. The derivative control block representation is shown in Figure 11.24.

$$\text{Time domain: } u_c(t) = K_d \frac{de}{dt}$$

$$\text{Laplace domain: } U_c(s) = [K_d s] E(s)$$

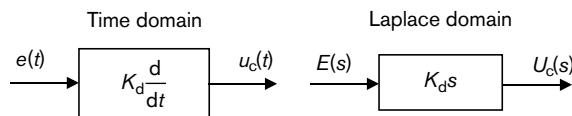


Figure 11.24 Derivative control block diagram.

Firstly we examine how derivative control and proportional control will act differently on the same instantaneous error signal. Figure 11.25 shows the output error signal for two different processes. We see that the error is increasing in both cases, but more rapidly in the second example.

At point X, a proportional controller would respond to the error signal from both processes by producing the same output. The proportional controller acts on the instantaneous error and therefore no account is made for the *rate* at which the process error is increasing. However, a derivative controller acts on the rate of change of the error. It would produce a larger corrective signal for Process 2 than for Process 1, in keeping with

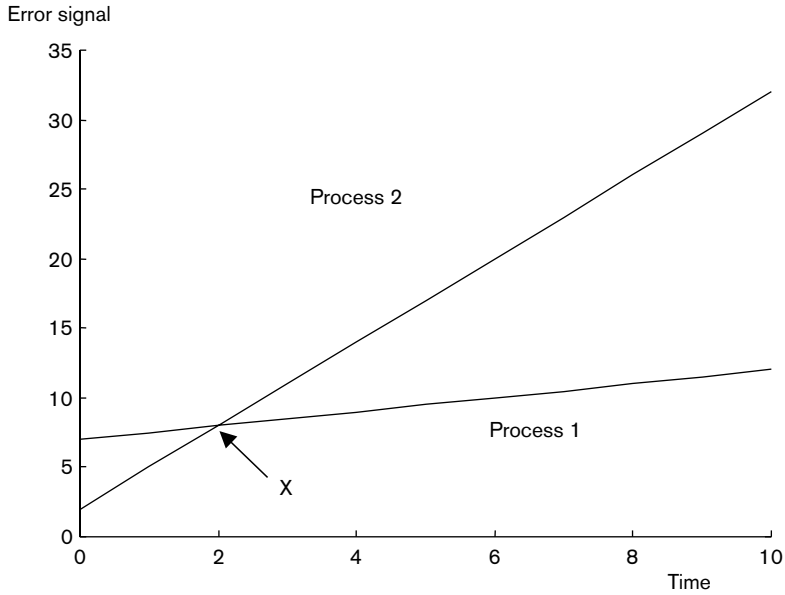


Figure 11.25 Error signal increasing for different processes.

the fact that we may have to act more quickly to ensure that Process 2 does not increase so much that it reaches warning or operational limits.

The second important feature of derivative control is its effect on a constant (steady state) error signal. If an error signal becomes constant (not necessarily zero!), the derivative of the error will be zero. This property, that derivative control has no output when it is acting on constant signals, leads to an important implication for the practical implementation of the derivative control term. If we use pure derivative control along with step reference signals we will frequently find that the derivative controller is producing no control signal and is therefore taking no action. To avoid this happening we always use the derivative control term in combination with a proportional term. This is called proportional and derivative control, or in short, PD control. In this section we investigate the use of this composite controller:

Proportional and derivative control

$$\text{Time domain: } u_c(t) = K_p e(t) + K_d \frac{de}{dt}$$

$$\text{Laplace domain: } U_c(s) = [K_p + K_d s] E(s)$$

11.4.1 Second-order system analysis – proportional plus derivative control

We look at three aspects of derivative control: the first is closed-loop system stability, and this is followed by an assessment of reference tracking and disturbance rejection performance.

To investigate derivative control we perform an assessment using a process whose transfer function is second order. This contrasts with the previous analysis performed for proportional and integral control actions, where the process was assumed to be modelled by a first-order transfer function.

Consider a position control system with low damping where the output signal is given in mm and the input to the process is in volts. Let the process be modelled by the transfer function description

$$G(s) = \frac{2}{s^2 + 0.4s + 1}$$

The reference signal is a unit step, so that $R(s) = 1/s$, and we model the constant load disturbance by $D(s) = (0.5/s)$. A controller with a fixed proportional term of $K_p = 0.2$ is used, along with a derivative controller term, $K_D(s) = K_d s$. The full PD controller is then $K(s) = 0.2 + K_d s$.

Figure 11.26 shows our typical block diagram for tracking and disturbance rejection analysis.

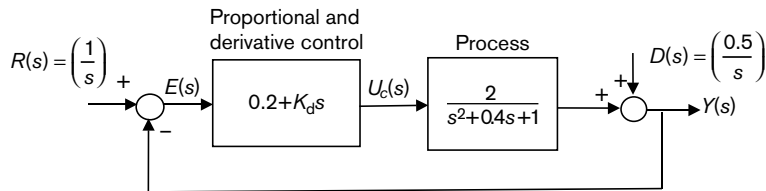


Figure 11.26 Closed-loop system for PD control.

The key transfer functions in the assessment are:

$$G_{CL}(s) = \frac{G(s)K(s)}{1 + G(s)K(s)} = \frac{0.4 + 2K_d s}{s^2 + (0.4 + 2K_d)s + 1.4} = \frac{n_{CL}(s)}{d_{CL}(s)}$$

$$S(s) = \frac{1}{1 + G(s)K(s)} = \frac{s^2 + 0.4s + 1}{s^2 + (0.4 + 2K_d)s + 1.4} = \frac{n_s(s)}{d_{CL}(s)}$$

A: Closed-loop stability

Identify the common characteristic polynomial for $G_{CL}(s)$ and $S(s)$ as the transfer function denominator, $d_{CL}(s)$, to give the closed-loop pole equation as

$$d_{CL}(s) = s^2 + (0.4 + 2K_d)s + 1.4 = 0$$

We establish a link with the standard form for the second-order system.

$$G_{2nd}(s) = \frac{K\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} = \frac{n_{2nd}(s)}{d_{2nd}(s)}$$

Compare the two characteristic equations:

$$d_{2nd}(s) = s^2 + 2\zeta\omega_n s + \omega_n^2 = 0$$

$$d_{CL}(s) = s^2 + (0.4 + 2K_d)s + 1.4 = 0$$

Then clearly,

$$(i) \quad \omega_n^2 = 1.4 \text{ giving } \omega_n = 1.1832$$

$$(ii) \quad 2\zeta\omega_n = (0.4 + 2K_d) \text{ so that } \zeta = (0.4 + 2K_d) / (2 \times 1.1832) = 0.169 + 0.845K_d \text{ where } K_d > 0.$$

Two conclusions follow:

- Derivative control has not changed the dynamical order of the closed-loop system from that of the original system, and the system is closed-loop stable for all the values of $K_d > 0$.
- Derivative control leads to a *direct link* between the value of the derivative gain and the closed-loop system damping ratio, ζ .

It is the link between the selection of K_d and the closed-loop system damping ratio which is of particular value. In fact, we are able to *tune* K_d to achieve a desired value for ζ . From the relationship given for ζ :

$$\zeta = 0.169 + 0.845K_d$$

we can find the values of K_d associated with underdamped ($\zeta < 1$), critically damped ($\zeta = 1$) and overdamped ($\zeta > 1$) systems. We can see this link for the example given in Table 11.3.

Table 11.3 Derivative control and the link to ζ .

Derivative gain value	Damping ratio	Comment
$K_d = 0$	$\zeta = 0.169$	Open-loop system very underdamped
$0 < K_d < 0.983$	$0.169 < \zeta < 1$	Underdamped closed-loop system
$K_d = 0.983$	$\zeta = 1$	Critically damped system response
$K_d > 0.983$	$\zeta > 1$	Overdamped system response

For example, if we wished the design damping ratio to be $\zeta = 0.7$ then we should tune the derivative control so that $K_d = 0.6284$ and we should know that the dynamic response will be underdamped and have the desired design damping ratio. The full range of responses we can obtain can be shown by the usual s -domain plot (Figure 11.27) for the system, given by:

$$K(K_p + K_d s)G(s) = \frac{K(0.2 + 0.6284s)}{s^2 + 0.4s + 1}$$

where K represents the varying gain on the s -domain plot.

B: Reference tracking performance

The closed-loop stability analysis has shown the full range of response dynamics that can be obtained using derivative control. We use a Simulink exercise to examine the actual shape of the reference tracking responses.

Problem: Derivative control – reference tracking response

For the PD control system given in Figure 11.26 and with $D(s) = 0$:

- (a) If $K_d = 0.2$, use a Simulink simulation to find the output response. Find the damping ratio from a theoretical analysis. What steady state offset is found from the simulation?
- (b) If the desired value of damping ratio is $\zeta = 1.2$, what value of derivative gain should be selected? Use the simulation to check that an overdamped response is obtained. What is the value for the steady state offset in this case?
- (c) Use the steady state values for above to demonstrate that changing the derivative gain has not changed the steady state error.

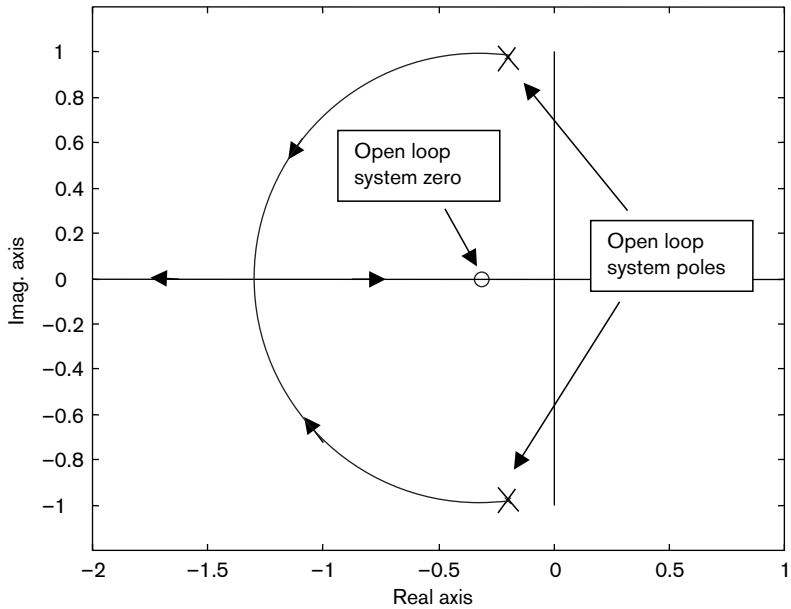


Figure 11.27 Pole-zero map for system with derivative control.

Solution (a) The Simulink model needed is given in Figure 11.28.

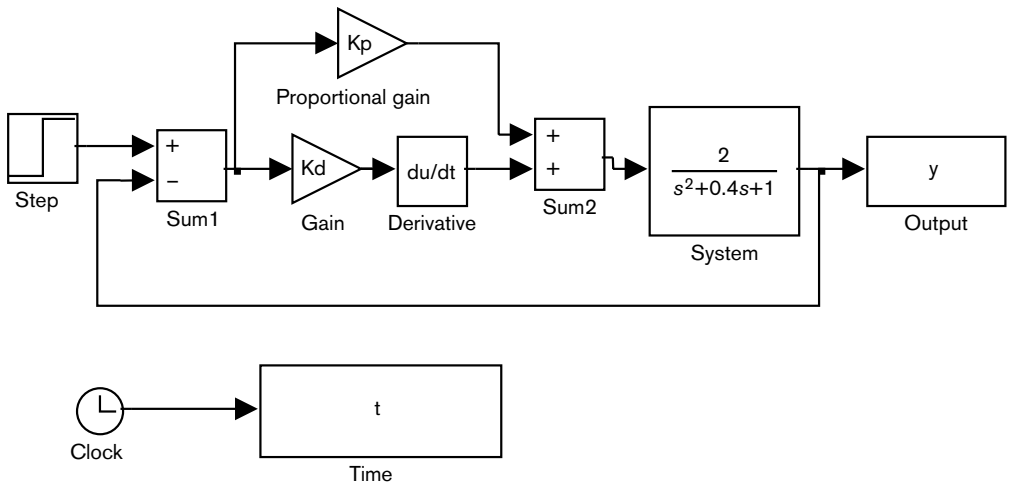


Figure 11.28 Simulink model for PD control tracking analysis.

The damping ratio is found from the formula $\zeta = 0.169 + 0.845K_d$; thus if $K_d = 0.2$, then $\zeta = 0.338$. From the output plot the steady state offset is $e_{R_{ss}} = r - y_{R_{ss}} = 1 - 0.2846 = 0.7154$.

(b) For the desired value of damping ratio, $\zeta = 1.2$, the derivative gain is found from the formula $\zeta = 0.169 + 0.845K_d$; thus $K_d = 1.22$.

(c) The two plots found in the exercise are shown in Figure 11.29.

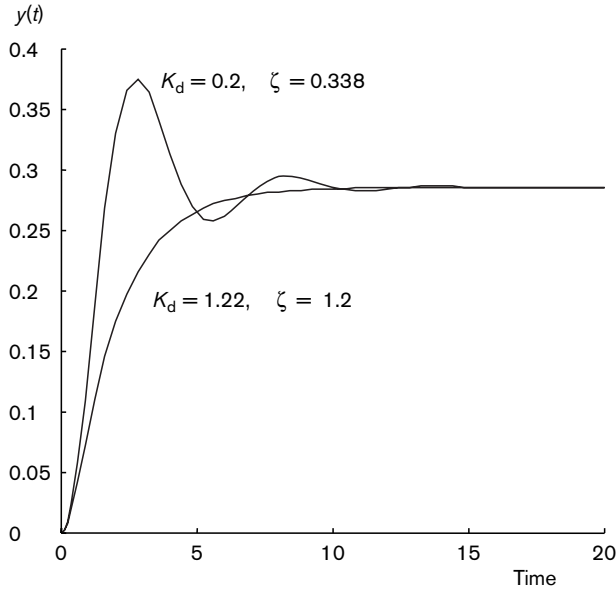


Figure 11.29 Output responses for different values of K_d .

The plots can be used to find the steady state offset as follows:

Derivative gain value K_d	Steady state output y_{Rss}	Steady state offset $e_{Rss} = r - y_{Rss}$
0.2	0.2846	0.7154
1.22	0.2846	0.7154

The steady state offset values obtained in the exercise show that the steady state tracking is independent of the different values of derivative gain.

The results of the exercise demonstrated that the value of the derivative gain appeared to have no effect on the value of the steady state error. In fact, in this example the reference tracking performance of proportional and derivative control depends on the proportional term only. We now intend to support this finding by using some formal analysis. We have from the general block diagram that

$$Y_R(s) = \left[\frac{G(s)K(s)}{1 + G(s)K(s)} \right] R(s) = \left[\frac{2(K_p + K_d s)}{s^2 + (0.4 + 2K_d)s + 1 + 2K_p} \right] \left(\frac{1}{s} \right)$$

and using the Final Value Theorem

$$y_{Rss} = \lim_{t \rightarrow \infty} y_R(t) = \lim_{s \rightarrow 0} s Y_R(s) = \lim_{s \rightarrow 0} s \left[\frac{2(K_p + K_d s)}{s^2 + (0.4 + 2K_d)s + 1 + 2K_p} \right] \left(\frac{1}{s} \right) = \frac{2K_p}{1 + 2K_p}$$

We can see quite clearly that the steady state output value depends only on the value of the proportional gain and is independent of the derivative gain term. If we evaluate this expression for $K_p = 0.2$, we find that

$$y_{Rss} = \frac{2K_p}{1+2K_p} = 0.2858$$

so with $r = 1$, the offset is $e_{Rss} = r - y_{Rss} = 0.7142$. Clearly derivative control is ineffective at ensuring that the steady state reference value is reached. This is the opposite of integral control. The reason for this is that when the error signal is constant, the derivative de/dt is zero and the controller takes no action to correct the offset.

Problem: Disturbance rejection performance

The assessment of disturbance rejection performance for derivative control is illustrated by a similar Simulink exercise. We use Figure 11.26 with $R(s) = 0$, $D(s) = 0.5/s$ and we fix the proportional gain at $K_p = 0.2$.

- If the desired value of damping ratio is $\zeta = 0.65$, what value of derivative gain should be selected? Use a Simulink simulation to obtain the output response. What is the steady state offset?
- If $K_d = 0.983$, use the simulation to find the output response. What type of output response has been obtained: underdamped, critically damped or overdamped? Find the damping ratio to justify the result found. What is the steady state offset?
- Set the derivative gain to zero. What is the steady state output value? Use the results of the simulations to determine whether derivative control provides good disturbance rejection properties.

Solution (a) Figure 11.30 shows the Simulink model that is needed.

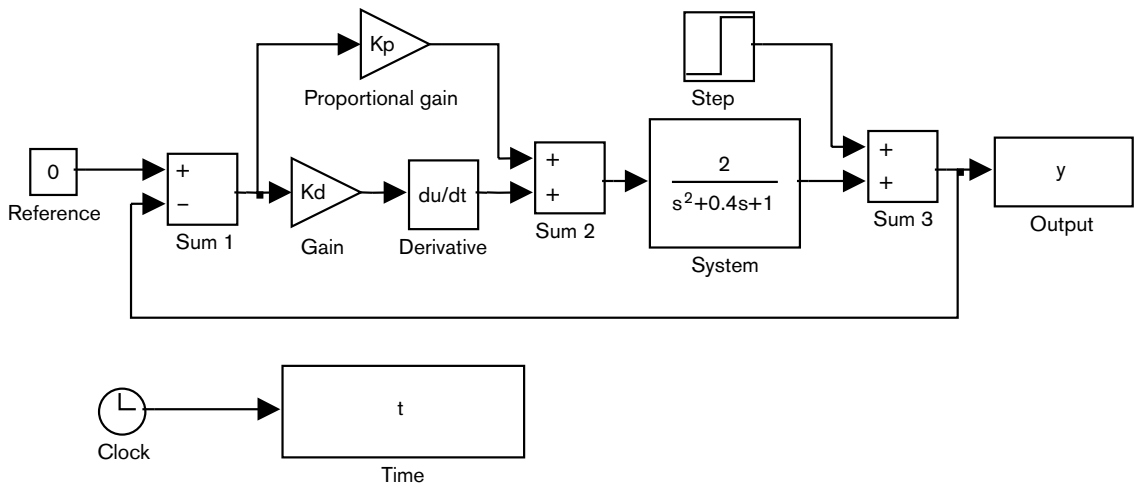


Figure 11.30 Simulink model for PD control and disturbance rejection analysis.

The damping ratio is found from the formula $\zeta = 0.169 + 0.845K_d$. For the desired value of damping ratio, $\zeta = 0.65$, a derivative gain of $K_d = 0.569$ is required. The plot of system response shows the offset to be $e_{Dss} = 0 - 0.3569 = -0.3569$.

- (b) Thus if $K_d = 0.983$, then $\zeta = 0.169 + 0.845K_d = 0.9996$ and a critically damped response results. From the output plot the steady state offset is $e_{Dss} = r - y_{Dss} = 0 - 0.357 = -0.357$.
- (c) The three plots found in the exercise are shown in Figure 11.31.

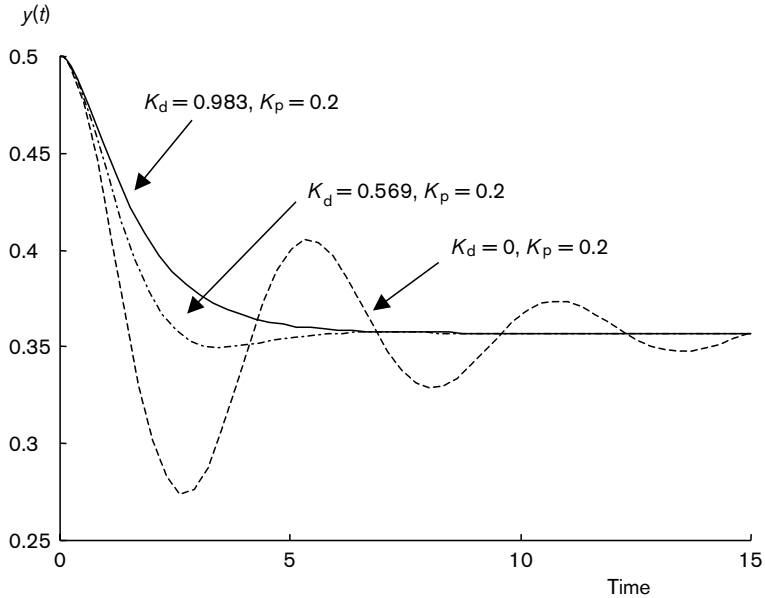


Figure 11.31 Output responses to a disturbance input for different values of K_d .

It is easily seen that the PD control plots have the same steady state value as the plot where the derivative gain has been set to zero.

We now examine the steady state disturbance rejection performance using the Final Value Theorem; recall the response equations as:

$$Y_D(s) = \left[\frac{1}{1 + G(s)K(s)} \right] D(s) = \left[\frac{s^2 + 0.4s + 1}{s^2 + (0.4 + 2K_d)s + 1 + 2K_p} \right] \left(\frac{0.5}{s} \right)$$

and

$$y_{Dss} = \lim_{t \rightarrow \infty} y_D(t) = \lim_{s \rightarrow 0} s Y_D(s) = \lim_{s \rightarrow 0} s \left[\frac{s^2 + 0.4s + 1}{s^2 + (0.4 + 2K_d)s + 1 + 2K_p} \right] \left(\frac{0.5}{s} \right) = \frac{0.5}{1 + 2K_p}$$

Thus $r = 0$ and $y_{Dss} = 0.5/1.4 = 0.357$, and the offset is $e_{Dss} = r - y_{Dss} = -0.357$. Clearly the derivative control term has no effect on the rejection of constant load disturbance signals and again this is the opposite of integral control.

Key result: Derivative control

- The derivative gain K_d is taken to be positive so that $K_d > 0$.
- Introducing derivative action into the controller means incorporating a derivative term into the controller.
- Increasing the amount of derivative action means that the numerical value of the derivative gain K_d is increased.

- Derivative action is usually associated with the controller *anticipating* the future direction of error signals.
- The damping ratio of the closed-loop system response can be tuned by changing the amount of derivative gain in the controller. This is an important practical property of derivative control.
- Derivative control will affect the *shape* of both the reference tracking and disturbance rejection responses.
- Derivative control has *no effect* on constant offsets in either the reference tracking or disturbance rejection responses. This is because the derivative of a constant error is zero and so the controller does not respond to the presence of the constant error. This is quite the opposite effect of the use of integral control.

11.5 PI and PID controller formula

We have studied the individual terms of the PID controller in this chapter. We will find that the P, I and D controllers can be combined to produce more complex controllers such as the PD controller we have just met. We find that as well as the P, I and PD controllers, we will meet the PI and PID controllers in our control studies.

PI controller: $u_c(t) = K_p e(t) + K_i \int_0^t e(\tau) d\tau$ or $U_c(s) = \left[K_p + \frac{K_i}{s} \right] E(s)$

PID controller: $u_c(t) = K_p e(t) + K_i \int_0^t e(\tau) d\tau + K_d \frac{de}{dt}$ or $U_c(s) = \left[K_p + \frac{K_i}{s} + K_d s \right] E(s)$

The choice of which terms (P, I or D) to include for a particular control system is discussed in Chapter 12.

What we have learnt

- ✓ To recognise the different terms in the three-term control law

Proportional $u_p(t) = K_p e(t)$ $U_p(s) = K_p E(s)$

Integral $u_i(t) = K_i \int_0^t e(\tau) d\tau$ $U_i(s) = \frac{K_i}{s}$

Derivative $u_d(t) = K_d \frac{de}{dt}$ $U_d(s) = K_d s$

- ✓ To realise that an increasing proportional gain will cause increasing overshoots but decreasing steady state error in a the system's output response to a step input.
- ✓ To realise that integral action can remove steady state offsets, including those caused by disturbance inputs signals.
- ✓ To realise that derivative action can improve the damping of the closed-loop system but can do nothing to improve steady state performance.

Multiple choice

- M11.1** A regulator problem is where the closed-loop system must:
- (a) try to follow a series of set point changes
 - (b) remove any disturbances acting on the process
 - (c) respond very quickly
 - (d) respond very slowly

- M11.2** A proportional controller, K_p , is used with a first-order system given by $G(s) = K/(\tau s + 1)$. What is the closed-loop transfer function?
- (a) $\frac{KK_p}{\tau s + 1 + KK_p}$
 - (b) $\frac{KK_p}{\tau s + 1}$
 - (c) $\frac{K}{\tau s + 1 + K_p}$
 - (d) $\frac{K_p}{\tau s + 1 + K}$

- M11.3** Increasing a proportional gain will:
- (a) increase the overshoot, decrease the steady state error
 - (b) decrease the overshoot, increase the steady state error
 - (c) increase the overshoot, increase the steady state error
 - (d) decrease the overshoot, decrease the steady state error

- M11.4** The denominator of the sensitivity function $S(s)$:
- (a) is the same as the denominator of the open-loop transfer function $K(s)G(s)$
 - (b) is the same as the denominator of the closed-loop transfer function $G_{CL}(s)$
 - (c) is the same as the numerator of the open-loop transfer function $K(s)G(s)$
 - (d) is the same as the numerator of the closed-loop transfer function $G_{CL}(s)$

- M11.5** Integral control has the advantage of:
- (a) reducing steady state offsets
 - (b) reducing overshoot
 - (c) increasing stability
 - (d) all of the above

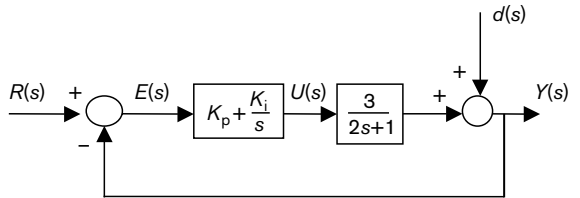
- M11.6** If there is a constant error in the output signal, derivative control:
- (a) will reduce this error to zero
 - (b) will reduce the error but not necessarily to zero
 - (c) will have no effect on the error
 - (d) will increase the error

- M11.7** A PID controller has the form:
- (a) $U(s) = (K_p + K_i + K_d)E(s)$
 - (b) $U(s) = [K_p + K_i s + (K_d/s)]E(s)$
 - (c) $U(s) = [K_p + (K_i/s) + K_d s]E(s)$
 - (d) $U(s) = K_p s E(s) + (K_i/s)E(s) + K_d s E(s)$

- M11.8** Applying which of the following controllers will increase the order of the closed-loop system compared to the open-loop transfer function?
- (a) proportional
 - (b) integral
 - (c) derivative
 - (d) none of the above will increase the order

- M11.9** Which control has a direct effect on the damping ratio in a second-order system?
- (a) proportional
 - (b) integral
 - (c) derivative
 - (d) none of the above will affect the damping ratio

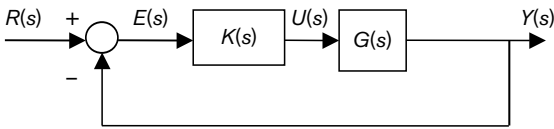
- M11.10** For the following system, what is the sensitivity transfer function?



- (a) $S(s) = \frac{1}{2s^2 + (3K_p + 1)s + 3K_i}$
- (b) $S(s) = \frac{s(2s + 1)}{2s^2 + (K_p + 1)s + 3K_i}$
- (c) $S(s) = \frac{s(2s + 1)}{2s^2 + (K_p + 1)s + K_i}$
- (d) $S(s) = \frac{s(2s + 1)}{2s^2 + (3K_p + 1)s + 3K_i}$

Questions: practical skills

Q11.1 The figure shows a typical closed-loop transfer function with the system transfer function $G(s)$ and the controller transfer function $K(s)$.

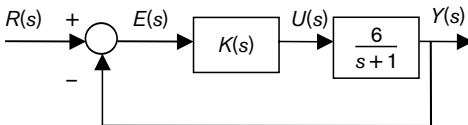


For the following two examples, what are the closed-loop transfer function, $G_{CL}(s)$, and the sensitivity (disturbance rejection) transfer function, $S(s)$?

(a) $K(s) = K_P + K_D s$, $G(s) = \frac{32}{s^2 + 2.4s + 16}$ (b) $K(s) = K_P + \frac{K_I}{s}$, $G(s) = \frac{12}{(s+10)}$

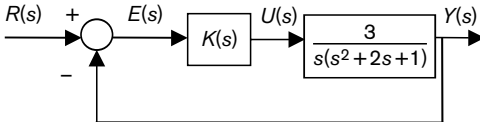
Q11.2 For an input step of magnitude 3, determine the steady state error for the controller $K(s)$ being:

- (a) a proportional controller; (b) a P+D controller; (c) a P+I controller
Comment on the error produced for each controller.

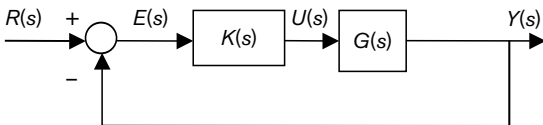


Q11.3 Let the reference be a unit ramp and determine the steady state error for the controller $K(s)$ being:

- (a) a P+D controller; (b) a P+I controller



Q11.4 For the closed-loop system shown and $G(s) = 2/(s + 0.5)$, apply a PI control to give a closed-loop damping of 0.5 and a closed-loop natural frequency of 2 rad/s.



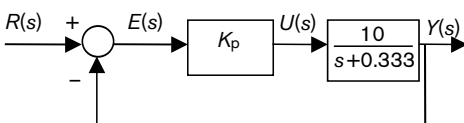
Q11.5 For the general closed-loop system as shown in the previous question and a system given by

$$G(s) = 25 / (s^2 + 0.1s + 0.25)$$

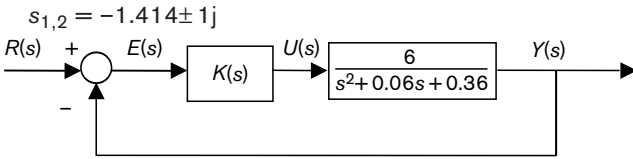
apply a PD controller to give a closed-loop damping of 0.5 and a closed-loop natural frequency of 4 rad/s.

Problems

P11.1 For the system shown, apply a proportional controller to move the time constant to a sixth of its open-loop value.

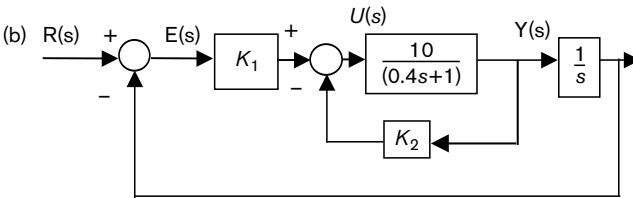
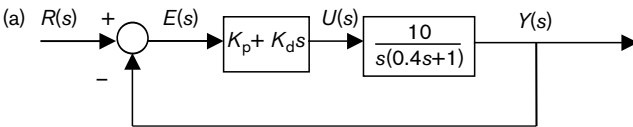


P11.2 For the system shown, apply a PD controller to move the poles to the position.

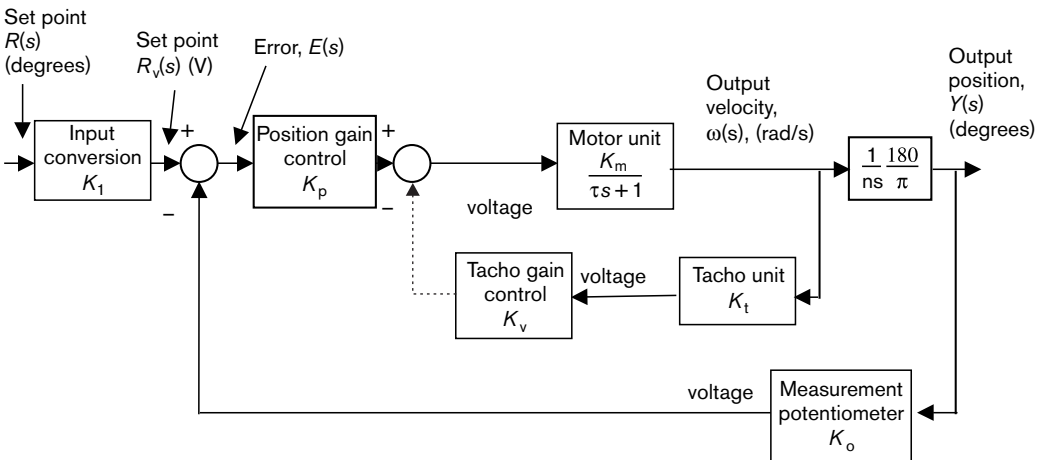


P11.3 A rate-feedback system avoids the problem of the zero on the numerator caused by using a PD controller.

- (i) Calculate the closed-loop transfer functions for the systems below. Note the differences in the structure.
- (ii) Choose the values of K_p and K_d , and K_1 and K_2 to meet the following requirements: damping ratio of $\zeta = 0.7$, $\omega_n = 2$ rad/s.
- (iii) Implement both systems in Simulink and comment on the output response to a unit step input for both systems. Do the results show that the designed system satisfies the design specifications?



P11.4 The figure shows a block diagram of a motor and the measurements of velocity (via the tachometer) and position (via the potentiometer). n represents the gearbox ratio between the rotating shaft and the output shaft. The left-hand side of the diagram represents the controller. A reference set point for the rotating shaft is entered (in degrees) and this is converted to an equivalent voltage. The error is calculated by subtracting the measured position from the desired position. This error is multiplied by a constant gain, K_p , and the resulting voltage used to control the motor.



Values of constants already determined:

$$K_m = 280 \text{ rad/s/V}$$

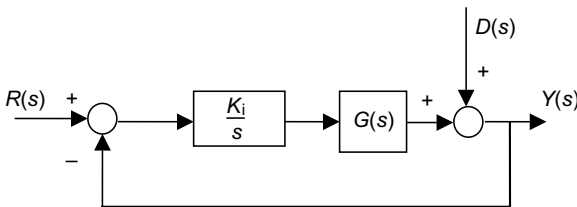
$$K_t = 0.022 \text{ V/rad/s}$$

$$K_o = K_1 = 0.0278 \text{ volts/degree}$$

$$n = 30: \text{ gearbox ratio}$$

$$\tau_m = 0.25 \text{ seconds}$$

- (a) Produce the closed-loop transfer function of the d.c. servo control system using the parameter values listed.
- (b) Set $K_v = 0$ to get the transfer function using only position feedback. What are the closed-loop poles for $K_p = 0.05, 0.1, 0.15, 0.2, 0.25$. Is the system stable for the values? Tabulate your answers and discuss the results.
- (c) Let $K_p = 0.2$ and calculate the closed-loop poles for $K_v = 0.2, 0.4, 0.6, 0.8, 1.0$. Tabulate your answers and find the values of ζ and ω_n . Discuss the results.



P11.5 Consider the use of integral control for reference tracking performance and disturbance rejection. Let the reference be given as $R(s) = r/s$ and the disturbance as $D(s) = d/s$ in the following general block diagram:

- (a) Show $Y(s) = Y_R(s) + Y_D(s)$, where

$$Y_R(s) = \left(\frac{G(s)K_i}{s + G(s)K_i} \right) R(s) \quad \text{and} \quad Y_D(s) = \left(\frac{s}{s + G(s)K_i} \right) D(s)$$

- (b) If $G(0)$ is finite, show that

$$y_{Rss} = \lim_{t \rightarrow \infty} y_R(t) = r$$

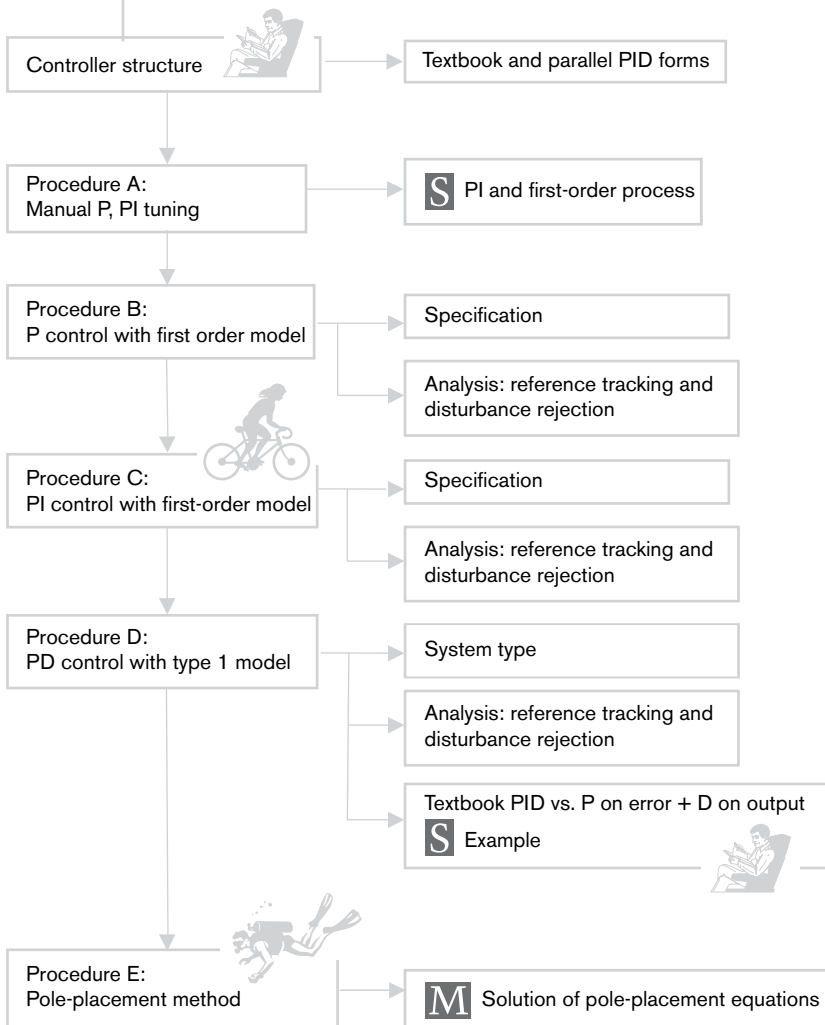
- (c) If $G(0)$ is finite, show that

$$y_{Dss} = \lim_{t \rightarrow \infty} y_D(t) = 0$$

- (d) Write short notes on the value and use of these two theoretical results.

12

PID control: the background to simple tuning methods



Gaining confidence



Help?



Going deeper



Skill section



Time to read

We need to find some methods for choosing the value of the PID gains in the PID controller algorithm. Although the tuning of the controller can be done in a heuristic manner, it is very time-consuming to find, if at all, appropriate gains which would satisfy the performance specifications on the system. We therefore need a procedure for choosing which terms to include in our controller and how to find the actual values of the controller gains. This chapter provides an introduction to the problem of *tuning* the PID controller. We take a look at PI, PD and finally PID design problems. We meet manual tuning and simple pole placement methods.

Learning objectives

- To appreciate the effects of the different terms of the PID controller.
- To be able to select an appropriate controller structure.
- To appreciate elementary manual tuning of PID controller.
- To look at more precise methods of tuning PID controllers.

12.1 Choice of controller structure

When we are faced with the problem of designing a PID controller for a particular system application, we often think that the task looks very difficult. There seems to be so many options and constraints to consider. But in fact the situation is not quite so daunting if we begin from four simple steps:

Step 1: Understand fully the capability of the PID controller.

Step 2: Discover the strengths and weaknesses of the system's existing response performance.

Step 3: Decide what type of closed-loop system performance is required.

Step 4: Choose a suitable method for selecting the PID controller coefficients or tuning the PID controller.

Step 1: The capabilities of PID control. In Chapter 11 we looked at the effects of the separate P, I and D terms of the controller. We saw that each term had specific effects in the transient and steady state portion of the closed-loop system responses. We can summarise these effects usefully in Table 12.1. The table is very similar to a toolbox where each tool (term) can be used to do a particular job. For example, if we need to remove steady state offsets from a closed-loop reference response, the table says that a D-term will not do this, that a P-term will reduce the offset if we increase the proportional K_p but that an I-term will eliminate the offset completely. Therefore we would choose to have integral action in our controller. Take a second example: suppose now after choosing to use integral action, we wish to speed up the closed-loop system response. A look at the table shows that increasing the proportional gain K_p will have just this effect, so we elect to have both proportional (P) action and integral (I) action in our controller. It is not difficult to extend this idea so that we can select the controller structure or architecture to match the effects we wish the controller to achieve. In this way PID control is really a family of controllers, and we use the shorthand labels P, PI, PD and PID to denote the particular controller structures. A full list of controller formulas for the members of the PID family that we will learn about in this chapter is given in Table 12.2.

Table 12.1 Table of effects of individual controller terms: P, I, D.

Control term	Reference tracking Step reference		Disturbance rejection Constant load disturbance	
	Transient	Steady state	Transient	Steady state
P	Increasing $K_p > 0$ speeds up the response	Increasing $K_p > 0$ reduces but does not remove steady state offset	Increasing $K_p > 0$ speeds up the response	Increasing $K_p > 0$ reduces but does not remove steady state offset
I	Introducing integral action, $K_i > 0$, gives a wide range of response types	Introducing $K_i > 0$ eliminates offset in the reference response	Introducing $K_i > 0$ gives a wide range of response types and speeds	Introducing integral action, $K_i > 0$ eliminates steady state offsets
D	Derivative action $K_d > 0$ gives a wide range of responses and can be used to tune response damping	Derivative action has no effect on steady state offset	Derivative action $K_d > 0$ gives a wide range of responses and can be used to tune response damping	Derivative action has no effect on steady state offset

Table 12.2 PID control family.

Label	Time domain form	Laplace domain form
P	$u_c(t) = K_p e(t)$	$U_c(s) = K_p E(s)$
I	$u_c(t) = K_i \int^t e(\tau) d\tau$	$U_c(s) = \left[\frac{K_i}{s} \right] E(s)$
D	$u_c(t) = K_d \frac{de}{dt}$	$U_c(s) = [K_d s] E(s)$
PI	$u_c(t) = K_p e(t) + K_i \int^t e(\tau) d\tau$	$U_c(s) = \left[K_p + \frac{K_i}{s} \right] E(s)$
PD	$u_c(t) = K_p e(t) + K_d \frac{de}{dt}$	$U_c(s) = [K_p + K_d s] E(s)$
PID	$u_c(t) = K_p e(t) + K_i \int^t e(\tau) d\tau + K_d \frac{de}{dt}$	$U_c(s) = \left[K_p + \frac{K_i}{s} + K_d s \right] E(s)$

The formulas in the table are given in *textbook* form with the PI, PD and PID formulas also referred to as *parallel* form. It is useful to explain these terms at this point in the chapter:

- (a) PID in *textbook* form. The PID controller formulas are capable of being written and implemented in several different ways. In fact industrial versions of PID exhibit some interesting but very different forms. These are introduced in Chapter 18 on process control. Thus the formulas of the PID table are often referred to as *textbook* PID controller forms.

- (b) PID in *parallel* or *decoupled* form. The textbook PID form has three decoupled terms, so that a numerical change in any individual coefficient from K_p , K_i or K_d changes only that term. For example if we change the value of K_i , then only the size of the effect of integral action changes, and this change is *decoupled* from the proportional and derivative terms. This decoupling of the three terms is also reflected in the parallel architecture of the PID controller.

The decoupled aspect of the PID controller is best seen in a diagram (Figure 12.1).

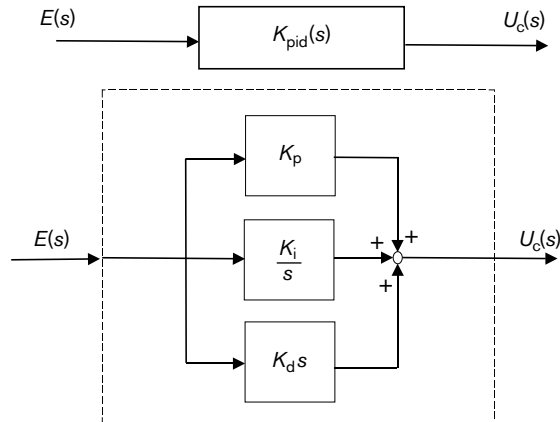


Figure 12.1 Parallel architecture of a PID controller.

Thus the parallel or decoupled form of the PID controller has three *independent parallel* paths. Each path is associated with only one of the PID terms. For example, the *middle path* is only concerned with *integral action*, and changing the value of K_i changes the size of the effect of the integral action.

Step 2: Understanding strengths and weaknesses of the system's existing performance.

In practice, this knowledge is usually obtained by conducting some simple time-domain experiments, often a step response test. We have already learnt about the main features of first-order and second-order system step responses and what we have learnt can be applied in a general way to categorise individual system performance.

Example An engineer conducts a series of step response tests on Process Unit No. CCR72 in a chemical works (Figure 12.2). The system is operating in steady state with the actuator signal, $u(t)$, at its operating reference level. Two different step tests are performed by changing the actuator input signal value from its steady state (or zero) level. One test is a +5% step input change and the second is a -10% step input change from the steady state level. The output temperature responses are plotted (Figure 12.3) and the chemical company's assessment form completed.

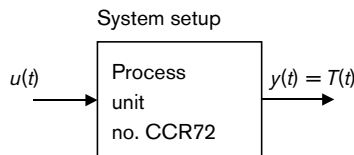


Figure 12.2 Process unit CCR72.

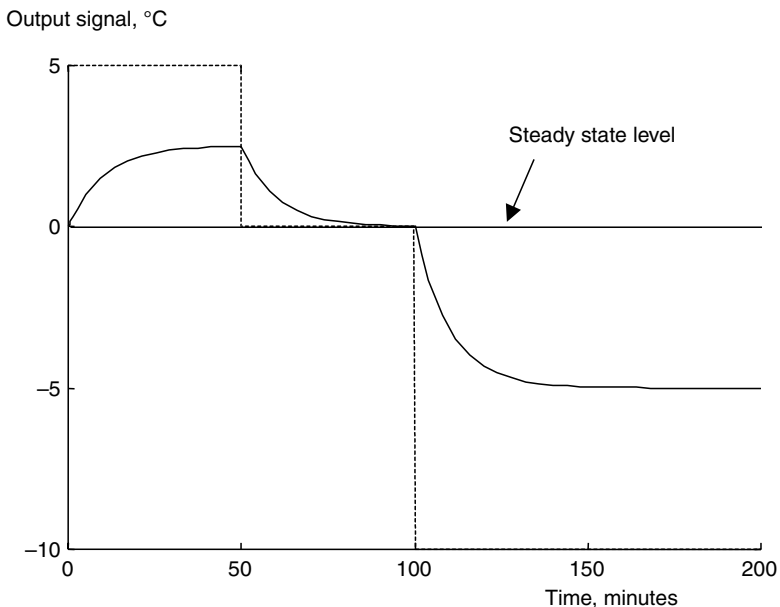


Figure 12.3 Open-loop step test traces from plant.

WJK Chemical Works					
Date 16.04.1998			Location: Shed 4		
			Unit No: CCR 72		
Tests Made: +5% Input change. -10% Input change					
Step size	Delay time	Response type	Time constant	SS value	Gain
+5%	0	First order	10 min	2.5	0.5
-10%	0	First order	10 min	-5.0	0.5
Test comments					
1. Fairly linear process because \pm steps give similar response shapes.					
2. First-order process type – time constant shows response is much too slow. Time constant is about 10 minutes.					
3. Steady offset observed – needs to track reference better.					
4. Proportional control used and does not appear to be very effective.					
5. Operators report process load disturbances frequent and control system cannot cope.					

The example shows how an understanding of the existing control system performance begins to lead to an idea of what sort of control system performance is actually needed. In the example, the test engineer thought the process too sluggish or slow, that the offset needed to be removed and had learnt from the process operators that load disturbances were a problem.

Step 3: Desired closed-loop system performance. From a better understanding of the system and its performance we move on to Step 3, which is concerned with deciding what type of closed-loop system performance has to be delivered by the PID control system. This really comprises two parts:

1. the general considerations which will dictate the structure of the appropriate controller, and
2. the design specifications that the controller has to try to achieve.

Take the general considerations first, since this is really a question about the type of steady state accuracy required to step reference and load disturbance signals. This is important because a requirement for steady state accuracy means that integral action is needed in the controller. We may divide the family of PID controllers as follows:

Steady state accuracy is required

Then choose from PI or PID control

Steady state accuracy is not so important

Then choose from P or PD

The second part of Step 3 concerns the design specifications that the control system should achieve. A typical list of specifications based on Chapter 9 is given by

Time constant, τ_{CL}

Rise time, $t_r(10\%, 90\%)$

Percentage overshoot, OS(%)

Damping ratio, ζ

Natural frequency, ω_n

Closed-loop pole positions, $s_i, i = 1, \dots, n$

Settling time, $t_s(2\%)$ or $t_s(5\%)$

Steady state offset, e_{ss}, e_{RSS} or e_{Dss}

Disturbance peak value, y_{Dpeak}

Disturbance settling time, $D_{ts}(2\%)$ or $D_{ts}(5\%)$

Some of these specifications can be given typical values; for example, closed-loop system damping is often specified to lie in the range 0.6 to 0.75. Other control system performance specifications will depend on the particular type of system being controlled, and engineering experience. Sometimes one of the specifications is especially important and is given critical values which must be achieved.

Example: Control system specifications

Different industrial control systems often have particular controller requirements.

Ship autopilot control	Specification	Design solutions
No overshoot to a step demand in heading, but good speed of response	OS(%) = 0 $e_{ss} = 0$	Design for critical damping $\zeta = 1$ Use proportional and integral control
Liquid tank level system	Specification	Design solution
Good speed of response, but steady state accuracy not required	$\tau_{CL} \ll \tau_{OL}$ e_{ss} small	Design for first-order response type, use proportional control

Gas turbine temperature control	Specification	Design solution
Load changes frequent. Steady state accuracy essential. Fast disturbance rejection needed within 5 minutes	Disturbance rejection design $e_{DSS} = 0$ $D_{ts} < 5$ minutes	Underdamped response design for speed of response. Use proportional and integral control

A further skill often needed in Step 3 is to be able to manipulate a given control specification into a form which can then be used in the PID tuning method. There are a number of useful relationships from first and second-order systems and these will be used later in the chapter.

Step 4: Selecting the PID controller coefficients. This is the problem of choosing the PID controller coefficients to achieve the desired closed-loop control performance. In this chapter we start by introducing a manual tuning method. We then develop procedures for tuning systems when we have some knowledge of system models. This gives a more scientific approach to PID controller tuning. More techniques are given in Chapter 19.

- Procedure A: Manual P, PI controller tuning
- Procedure B: Proportional control of a process with first-order system model
- Procedure C: Proportional and integral control of a process with a first-order model
- Procedure D: Proportional and derivative control of a process with a simple Type 1 model
- Procedure E: PID controller design by pole placement

12.2 Manual tuning method

The manual tuning of PID control is surprisingly common. As we have already learnt and tabulated, the three terms are each associated with particular closed-loop response effects. Manual tuning relies on these intuitive links. We restrict our discussion to proportional and proportional-plus-integral control because derivative control is a little specialised. A systematic design procedure is as follows.

Procedure A: Manual P, PI controller tuning

- Step 1* Determine whether the priority for the closed-loop system is reference tracking, load disturbance rejection or a trade-off between the two.
- Step 2* Determine whether steady state accuracy is essential to the control system performance.
- Step 3* *Proportional control tuning.* Introduce proportional action by increasing the value of proportional gain, K_p , until the speed of response is acceptable.
- Step 4* *Integral control tuning.* If steady state accuracy is considered important (see Step 2) then introduce integral action into the controller by increasing the size of K_i . The integral gain K_i should be increased so that an acceptable settle time is achieved.

Step 5 Balancing the controller terms. Increasing K_i may increase the overshoot; to compensate, decrease K_p . A little fine tuning between K_i and K_p will be necessary to achieve acceptable time responses.

The manual procedure is a *trial and error* process. The control engineer hopes that Steps 4 and 5 will eventually converge to an acceptable solution. Basically, while K_p changes the speed of response, changing K_i alters the settling time, with a tendency to introduce overshoot. Excessive overshoot usually needs to be avoided, but the settle time must be reasonably short so that the desired output level is reached. The procedure is further complicated if it is necessary to obtain satisfactory responses for both reference tracking and load disturbance rejection.

Example In the WJK Chemical Works, an engineer decides to tune up Process Unit No. UV46. An open-loop step response shows the process to give a 5°C output temperature change to a 10% change in actuator input signal level and the open-loop time constant is 10 minutes. The design specifications are:

1. an effective closed-loop time constant of about 3 minutes
2. to have no steady state error in reference tracking mode
3. a $\pm 5\%$ settle time of better than 20 minutes

The engineer follows the Manual Tuning Procedure.

Step 1: The performance is required for reference tracking mode only, so the engineer uses step response tests with a reference step change of 10°C . The settle constraint means that the system must settle to within $\pm 0.5^\circ\text{C}$ of the 10°C reference signal before 20 minutes has elapsed.

Step 2: Steady state accuracy is seen to be important because steady state errors have to be eliminated. This requirement coupled with a desire to speed up the process response leads to the engineer to select proportional and integral control, namely a PI controller.

Step 3: Tuning the proportional term. The experiment involves repeated step response tests, recording the process output on, for example, a chart recorder, and determining when K_p is satisfactory (Figure 12.4).

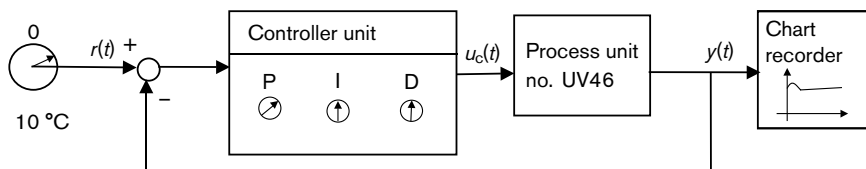


Figure 12.4 Step response test setup on site.

The following table of experimental data was recorded.

K_p	Δy_{ss}	$0.632 \times \Delta y_{ss}$	τ_{CL} (mins)	Experimental time
0.5	0.2	0.126	7.95	32 min
1.0	0.33	0.210	6.60	29 min
2.0	0.5	0.315	4.85	27 min
3.0	0.6	0.378	4.00	25 min
3.5	0.64	0.400	3.60	19 min
3.75	0.65	0.41	3.47	17 min
4.25	0.68	0.43	3.2	16 min
4.75	0.7	0.44	2.9	15 min

To tune up the proportional term K_p involves eight experiments with production disruption of 10 °C step reference changes and experimental run times of three hours. The tuning experiments still have one stage to go.

Step 4: Tuning the integral action. Introducing integral action is designed to provide the steady state accuracy and a $\pm 5\%$ settle time of better than 20 minutes. The following experimental data was recorded.

K_p	K_i	t_s (5%)	Overshoot	Experimental time
4.75	0.1	> 100 min	0	> 100 min
4.75	0.5	12 min	0	20 min
4.75	0.75	9 min	5%	15 min

The engineer resolved to use the settings $K_p = 4.75$ and $K_i = 0.5$. Note that over another two hours of experimental time was required giving a total of five hours of experimental time and production disruption. Further, note that the manual tuning procedure has yet to begin considering the disturbance rejection performance of the PI controller designed!

The example shows quite clearly the significant overhead in person-power costs, and potential production disruption involved in manual tuning of a PID controller. Also it should be noted that assessing the process response relied on the engineer's experience and acumen.

Problem Process Unit No. UV 46 has open-loop data such that there is a 5 °C offset to a 10% input change and a measured time constant of 10 minutes.

- Derive a first-order process model.
- Devise a Simulink model to follow the steps of the Manual Tuning Process.
- Follow the steps of the Manual Tuning Process to find the proportional gain K_p of 4.75 and also the integral gain K_i of 0.5.
- Is the final tuning acceptable?

Solution (a) The process is in open loop. The steady output changed by 5 °C when the input signal undergoes a 10% change; thus,

$$\Delta y_{ss} = 5 \text{ }^\circ\text{C}, \Delta u = 10\% \quad K = \frac{\Delta y_{ss}}{\Delta u_{ss}} = \frac{5 \text{ }^\circ\text{C}}{10\%} = 0.5(\text{ }^\circ\text{C} / \%)$$

τ = Time constant = 10 minutes

Unit UV 46 has a model $G(s) = \frac{0.5}{10s+1}$

(b) A Simulink model which can be used to follow the steps of the tuning procedure is given in Figure 12.5.

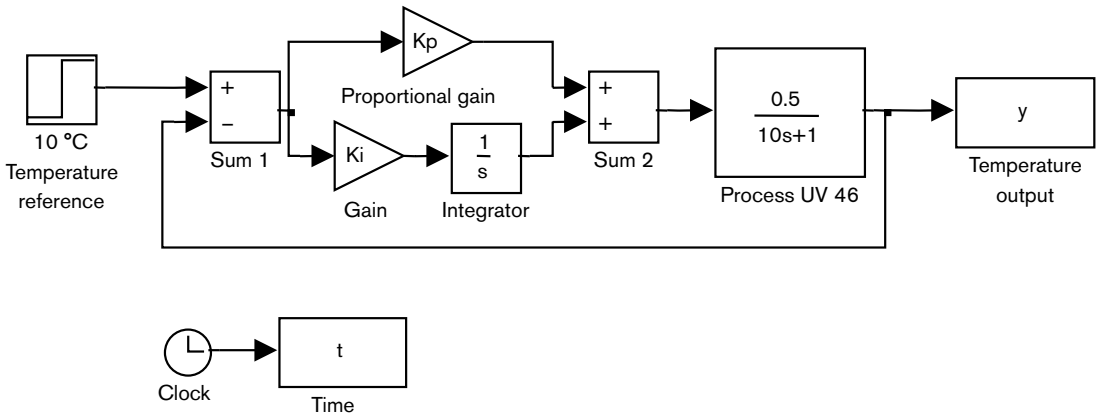


Figure 12.5 Simulink model for manual tuning.

(c) The solution to part (c) follows closely the tables of results given above.

(d) At the end of the tuning stages the step responses should look like those in Figures 12.6(a) and (b).

PID controller tuning using simple process models

We have already demonstrated that the manual tuning method is somewhat laborious, time-consuming and an ineffective use of resources. We must do better, and to do this we use system knowledge. We know that many processes can be given a simple transfer function representation, and we know the formulas and characteristics of the time responses that can be obtained by simple PID closed-loop control. Putting these facts to work enables us to find the proportional, integral and derivative gains necessary to achieve the desired closed-loop performance. We first look at proportional and proportional-plus-integral control of a process which can be represented by a simple first-order system model, and devise procedures for these cases.

12.3 Proportional control of a system with a first-order model

Consider the general reference tracking and disturbance rejection framework (Figure 12.7) first introduced in Chapter 11. The controller $K(s)$ is a proportional controller, $K(s) = K_p$.

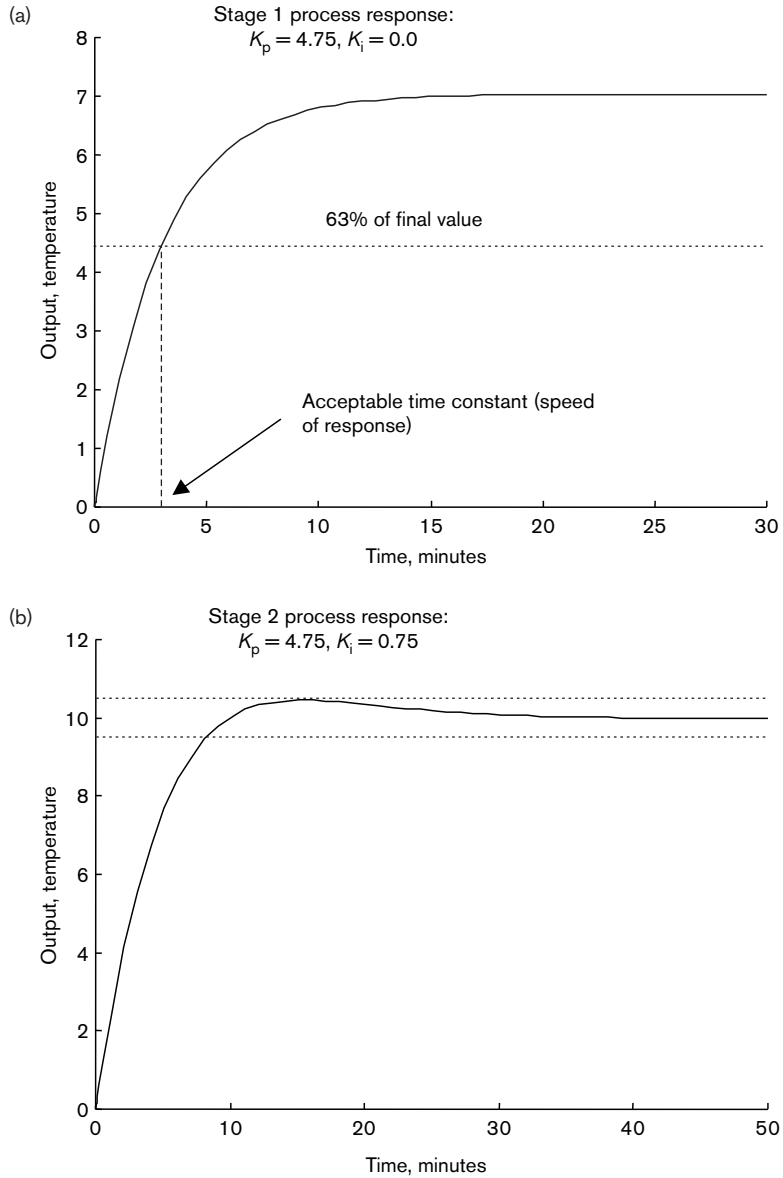


Figure 12.6 (a) Step response for stage 1 of the manual tuning process; (b) step response for stage 2 of the manual tuning process.

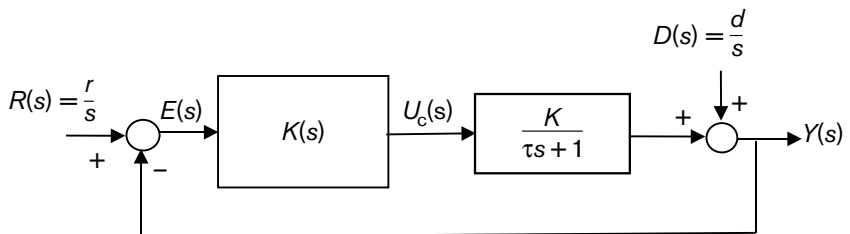


Figure 12.7 Closed-loop control system.

The closed-loop expression is therefore

$$Y(s) = \left(\frac{KK_p}{\tau s + 1 + KK_p} \right) \frac{r}{s} + \left(\frac{\tau s + 1}{\tau s + 1 + KK_p} \right) \frac{d}{s}$$

within which we recognise the separate reference tracking and disturbance rejection responses. Both these transfer functions can be put in time constant form to give:

The reference tracking term is:

$$Y_R(s) = \frac{\frac{KK_p}{1 + KK_p}}{\left(\frac{\tau}{1 + KK_p} \right) s + 1} \left(\frac{r}{s} \right)$$

or

$$Y_R(s) = \left(\frac{K_{CL}}{\tau_{CL} s + 1} \right) \frac{r}{s} \quad \text{with} \quad K_{CL} = \frac{KK_p}{1 + KK_p} \quad \text{and} \quad \tau_{CL} = \frac{\tau}{1 + KK_p}$$

The disturbance rejection term is:

$$Y_D(s) = \left(\frac{\tau s + 1}{\tau s + KK_p} \right) \frac{d}{s} = \left(\frac{K_s(\tau s + 1)}{\tau_{CL} s + 1} \right) \frac{d}{s}$$

where

$$K_s = \frac{1}{1 + KK_p} \quad \text{and} \quad \tau_{CL} = \frac{\tau}{1 + KK_p}$$

From the standard forms above the common closed-loop time constant can be identified as

$$\tau_{CL} = \frac{\tau}{1 + KK_p}$$

and, by applying the Final Value Theorem to $Y_R(s)$ and $Y_D(s)$, we have the following steady state outputs:

$$y_{Rss} = K_{CL} r = \left(\frac{KK_p}{1 + KK_p} \right) r \quad \text{and} \quad y_{Dss} = K_s d = \left(\frac{1}{1 + KK_p} \right) d$$

with steady state errors:

$$e_{Rss} = r - y_{Rss} \quad \text{giving} \quad e_{Rss} = \left(\frac{1}{1 + KK_p} \right) r$$

and

$$e_{Dss} = 0 - y_{Dss} \quad \text{giving} \quad e_{Dss} = - \left(\frac{1}{1 + KK_p} \right) d$$

Closed-loop time constant specification

To achieve a given closed-loop time constant of τ_{CL} , we use the model information K , τ and the formula $\tau_{CL} = \tau / (1 + KK_p)$ to obtain the necessary proportional gain as:

$$K_p = \frac{1}{K} \left(\frac{\tau}{\tau_{CL}} - 1 \right)$$

The consequence of using this value of K_p is to *fix* the closed-loop steady state offsets as:

$$e_{Rss} = \left(\frac{\tau_{CL}}{\tau} \right) r \quad \text{and} \quad e_{Dss} = - \left(\frac{\tau_{CL}}{\tau} \right) d$$

These formulas show that specifying a faster and faster closed-loop response by decreasing τ_{CL} will have the effect of increasing the required proportional gain and reducing the steady state offsets. But we must also remember that an increasing proportional gain will increase the size of the control signal and possibly cause real-world actuators to saturate at the limits of their operational range.

Steady state offset specification for reference tracking

The steady state error relation for reference tracking can be written as

$$|e_{Rss}| = \frac{1}{1 + KK_p} |r|$$

The required steady state error can be written as a percentage of the reference input as

$$e_{ss}(\%) = \frac{e_{Rss}}{r} \times 100$$

Combining these two expressions for e_{Rss} gives

$$\frac{e_{ss}(\%)}{100} |r| = \frac{1}{1 + KK_p} |r|$$

Therefore we can find an expression for K_p as

$$K_p = \frac{1}{K} \left(\frac{100}{e_{ss}(\%)} - 1 \right)$$

Selecting a proportional gain following this rule would lead to the following closed-loop time constant

$$\tau_{CL} = \frac{\tau}{1 + KK_p} = \tau \frac{e_{Rss}}{|r|} = \left(\frac{e_{ss}(\%)}{100} \right) \tau$$

with the associated disturbance rejection error

$$|e_{Dss}| = \frac{1}{1 + KK_p} |d|$$

Steady state offset specification for disturbance rejection

For disturbance rejection specification, we need to know at what reference level the process is operating as previously described in Chapter 9. We denote this output level as

y_{ref} . If the required steady state error percentage specification for disturbance rejection is $e_{\text{Dss}}(\%)$ of the output reference level, y_{ref} , we have

$$|e_{\text{Dss}}| = \left(\frac{e_{\text{Dss}}(\%)}{100} \right) |y_{\text{ref}}| = \frac{|d|}{1 + KK_p}$$

which yields

$$K_p = \left(\frac{1}{K} \right) \left(\frac{100|d|}{e_{\text{Dss}}(\%)y_{\text{ref}}} - 1 \right)$$

Selecting a proportional gain following this rule would lead to the following closed-loop time constant:

$$\tau_{\text{CL}} = \left(\frac{e_{\text{Dss}}(\%)y_{\text{ref}}}{100|d|} \right) \tau$$

with the associated reference tracking error

$$|e_{\text{Rss}}| = \frac{1}{1 + KK_p} |r|$$

We can see from these simple design equations that if we satisfy one design specification, this also fixes the other performance values of the control system.

For example, selecting K_p to achieve a desired closed-loop time constant automatically fixes the steady state offsets achievable. Similarly, a value of K_p which achieves a particular steady state specification has implications for the closed-loop time constant. Ultimately it must be noted that the analysis is linear and success in implementation will rely on the capabilities of the system actuators and sensors. The linear analysis is only a local analysis with limited validity. So with all these provisos we are able to give our second design procedure.

Procedure B: Proportional control of a process with first-order system model

Step 1: Use a process identification method to find a first-order model for the system responses of the form: $G(s) = K/(\tau s + 1)$.

Step 2: If disturbance rejection control design is of interest then it is necessary to estimate the size of the constant disturbances, and also record the output level at which the process operates. This we denote as y_{Ref} .

Step 3: Controller target specification. Select from:

- (a) Target closed-loop time constant, τ_{CL}
- (b) Reference tracking error percentage $e_{\text{ss}}(\%)$
- (c) Disturbance rejection error percentage of reference output level, y_{Ref} , given by $e_{\text{Dss}}(\%)$

Step 4: Compute the proportional gain, K_p , choosing from one of the following:

Controller specification	Calculation for K_p	Associated performance values (fixed by value of K_p)
Design closed-loop time constant, τ_{CL}	$K_p = \frac{1}{K} \left(\frac{\tau}{\tau_{CL}} - 1 \right)$	$e_{Rss} = \left(\frac{\tau_{CL}}{\tau} \right) r$ $e_{Dss} = - \left(\frac{\tau_{CL}}{\tau} \right) d$
Reference tracking error percentage, $e_{ss}(\%)$	$K_p = \frac{1}{K} \left(\frac{100}{e_{ss}(\%)} - 1 \right)$	$\tau_{CL} = \left(\frac{e_{ss}(\%)}{100} \right) \tau$ $e_{Dss} = \left(\frac{e_{Dss}(\%)}{100} \right) d $
Disturbance rejection error percentage, $e_{Dss}(\%)$ of y_{ref}	$K_p = \frac{1}{K} \left(\frac{100 d }{e_{Dss}(\%) y_{ref}} - 1 \right)$	$\tau_{CL} = \left(\frac{e_{Dss}(\%) y_{ref}}{100 d } \right) \tau$ $ e_{Rss} = \frac{1}{1 + K K_p} r $

Step 5: Perform any simulation studies necessary to investigate performance issues.

Step 6: Implement K_p

Problem In the WJK Chemical Works, Process Unit No. UV46 gave a 5% offset to a 10% input change, and the open-loop time constant was found to be 10 minutes. It is desired to find a proportional gain to achieve a closed-loop time constant of 3 minutes.

Solution We use Procedure B for proportional control of processes and assume a first-order model for the process.

Step 1: First-order model. The data yields $\tau = 10$ minutes and

$$K = \frac{\Delta y_{ss}}{\Delta u_{ss}} = \frac{5 \text{ }^\circ\text{C}}{10\%} = 0.5 (\text{ }^\circ\text{C per } \%) \text{. Model is } G(s) = \frac{0.5}{10s + 1}$$

Step 3: Controller target specification

Data: $\tau = 10$, $K = 0.5$

Desired closed-loop time constant, $\tau_{CL} = 3$

$$\text{Therefore } K_p = \frac{1}{K} \left(\frac{\tau}{\tau_{CL}} - 1 \right) = \left(\frac{1}{0.5} \right) \left(\frac{10}{3} - 1 \right) = 4.67$$

Associated performance: $e_{Rss} = 0.3r$ and $e_{Dss} = -0.3d$

There is a significant steady state offset for both reference tracking and disturbance rejection performance.

Step 4: Performance issues are covered by the predictions above.

Step 5: Implement $K_p = 4.67$. For this process, *manual* tuning took eight experiments and three hours of experimental run times. Clearly the use of process knowledge has saved time and resource costs.

12.4 Proportional and integral control of a system with a first-order model

We consider the general reference tracking and disturbance rejection framework of Figure 12.7, where the controller is of proportional and integral type (PI) type:

$$K(s) = K_p + \frac{K_i}{s}$$

The closed-loop expression is

$$Y(s) = \left(\frac{K(K_p s + K_i)}{\tau s^2 + (1 + KK_p)s + KK_i} \right) \frac{r}{s} + \left(\frac{(\tau s + 1)s}{\tau s^2 + (1 + KK_p)s + KK_i} \right) \frac{d}{s}$$

and the reference tracking and disturbance rejection terms can be identified as:

$$Y(s) = Y_R(s) + Y_D(s)$$

with

$$Y_R(s) = G_{CL}(s)R(s) = \frac{n_{CL}(s)}{d_{CL}(s)}R(s) = \frac{K(K_p s + K_i)}{\tau s^2 + (1 + KK_p)s + KK_i} \left(\frac{r}{s} \right)$$

and

$$Y_D(s) = S(s)D(s) = \frac{n_S(s)}{d_{CL}(s)}D(s) = \frac{(\tau s + 1)s}{\tau s^2 + (1 + KK_p)s + KK_i} \left(\frac{d}{s} \right)$$

We consider the steady state specification and the transient specifications on overshoot and settling time.

Steady state offset specification for reference tracking and disturbance rejection

We assume that we can find a PI controller that will stabilise the closed loop. This means that we are able to find controller coefficients K_p , K_i so that the closed-loop characteristic equation has roots in the Left Half Plane:

$$d_{CL}(s) = \tau s^2 + (1 + KK_p)s + KK_i = 0$$

With this assumption of closed-loop stability we can use the Final Value Theorem to investigate the steady state properties of the closed-loop system. The following steady state outputs can be found:

$$Y_{Rss} = \lim_{t \rightarrow \infty} y_{Rss}(t) = \lim_{s \rightarrow 0} s \left(\frac{K(K_p s + K_i)}{\tau s^2 + (1 + KK_p)s + KK_i} \right) \frac{r}{s} = r$$

with steady state error $e_{Rss} = r - Y_{Rss} = 0$.

Also, for the disturbance rejection:

$$Y_{Dss} = \lim_{t \rightarrow \infty} y_{Dss}(t) = \lim_{s \rightarrow 0} s \left(\frac{(\tau s + 1)s}{\tau s^2 + (1 + KK_p)s + KK_i} \right) \frac{d}{s} = 0$$

with steady state error $e_{Dss} = 0 - Y_{Dss} = 0$. The above analysis is only true provided we can design the PI controller to give a closed-loop stable system, but if this is achieved, the analysis shows perfect steady state tracking to step reference signals, where $e_{Rss} = 0$ and perfect steady state rejection to step load disturbance signals, with $e_{Dss} = 0$.

Overshoot and settling time specification for transient response

In Chapter 9, standard second-order systems theory was used to link some simple design performance values to the parameters ζ and ω_n . We recall these results as:

Overshoot

The link between overshoot and damping ratio was given by

$$OS(\%) = 100 \exp\left(\frac{-\zeta\pi}{(1-\zeta^2)^{1/2}}\right) \quad 0 < \zeta < 1$$

If $OS(\%) > 0\%$ then set

$$L_{os} = \ln\left(\frac{OS(\%)}{100}\right)$$

and the required damping ratio is

$$\zeta = \frac{L_{os}}{\sqrt{\pi^2 + L_{os}^2}}$$

If $OS(\%) = 0\%$ then set $\zeta = 1$.

Settle time

The link between settle time, damping ratio and natural frequency was given by

$$t_s(5\%) = \frac{3}{\zeta\omega_n} \quad \text{and this rearranges to give} \quad \zeta\omega_n = \frac{3}{t_s(5\%)}$$

$$t_s(2\%) = \frac{4}{\zeta\omega_n} \quad \text{and this rearranges to give} \quad \zeta\omega_n = \frac{4}{t_s(2\%)}$$

The above formulas can be used to translate overshoot and settle time specifications into damping ratio and natural frequency values. The link to the control design is obtained using the closed-loop characteristic equation, $d_{CL}(s)$ which is a function of the first-order system model parameters (K and τ) and the coefficients of the PI control law, K_p , K_i .

The standard second-order system characteristic equation is

$$d_{2nd}(s) = s^2 + 2\zeta\omega_n s + \omega_n^2$$

and we compare this to the scaled closed-loop characteristic equation,

$$\frac{d_{CL}(s)}{\tau} = s^2 + \left(\frac{1+KK_p}{\tau}\right)s + \left(\frac{KK_i}{\tau}\right) = 0$$

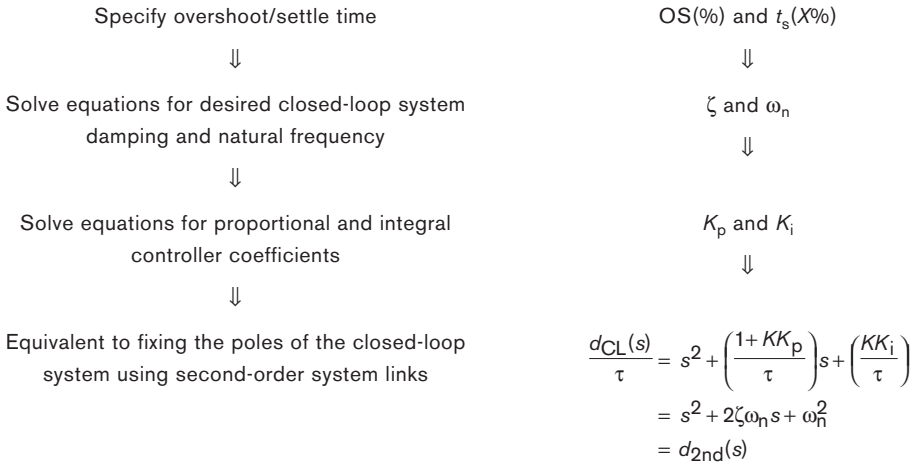
Hence, if design values for the closed-loop system damping ratio, ζ , and natural frequency ω_n are given, along with the model parameters K and τ we can equate coefficients to obtain the required PI controller coefficients. The equations which can be rearranged to give the controller parameters are

$$\frac{1+KK_p}{\tau} = 2\zeta\omega_n \quad \text{and} \quad \frac{KK_i}{\tau} = \omega_n^2$$

and these give

$$K_p = \frac{2\zeta\omega_n\tau - 1}{K} \quad K_i = \frac{\omega_n^2\tau}{K}$$

We are able to collect these links together as tools useful to design a PI controller to meet the transient response specifications.



To formalise this we create the following design procedure.

Procedure C: Proportional and integral control of a process with a first-order model

Step 1: Use process identification methods to find a first-order model for the system response. The model transfer function is

$$G(s) = \left[\frac{K}{\tau s + 1} \right]$$

Step 2: Controller target specification

(a) PI control guarantees zero steady state offset for both step reference signals tracking and step reference disturbance signals.

(b) Specify overshoot OS(%):

$$\text{If OS\%} > 0 \text{ then } \zeta = \frac{|L_{os}|}{\sqrt{\pi^2 + L_{os}^2}} \text{ and } L_{os} = \ln \left(\frac{\text{OS\%}}{100} \right)$$

$$\text{If OS\%} = 0 \text{ then } \zeta = 1$$

(c) Specify settle time requirement: $t_s(5\%)$ or $t_s(2\%)$:

$$\text{If } t_s(5\%) \text{ then } \zeta\omega_n = 3/t_s(5\%)$$

$$\text{If } t_s(2\%) \text{ then } \zeta\omega_n = 4/t_s(2\%)$$

Step 3: Compute controller coefficients, K_p , K_i .

$$\text{Solve } \frac{1+KK_p}{\tau} = 2\zeta\omega_n \text{ to give } K_p = \frac{2\zeta\omega_n\tau - 1}{K}$$

$$\text{Solve } \frac{KK_i}{\tau} = \omega_n^2 \text{ to give } K_i = \frac{\omega_n^2 \tau}{K}$$

Step 4: Use simulation tools to assess the performance and examine the disturbance rejection performance.

Step 5: Implement PI controller

$$K(s) = \left[K_p + \frac{K_i}{s} \right]$$

Problem In the WJK Chemical Works, Process Unit No. UV46 gave a 5 °C offset to a 10% actuator input change, and the open-loop time constant was found to be 10 minutes. It is desired to find a three-term controller to ensure no steady state offset to step reference signals, to achieve a 5% settle time of less than 20 minutes at most and to have only a little overshoot.

Solution We use the steps of Procedure C, and assume a first-order model for the process.

Step 1: First-order model identification. The data yields an open-loop time constant $\tau = 10$ min and the system gain parameter is found from:

$$K = \frac{\Delta y_{ss}}{\Delta u_{ss}} = \frac{5 \text{ }^\circ\text{C}}{10\%} = 0.5 \text{ (}^\circ\text{C per \%)} \quad \text{Model is } G(s) = \frac{0.5}{10s + 1}$$

Step 2: Controller target specification

Zero steady state offsets require PI control; therefore use

$$K(s) = K_p + \frac{K_i}{s}$$

$$t_s(5\%) = 20 \text{ min} \quad \text{hence } \zeta\omega_n = 3/t_s(5\%) = 0.15$$

Little overshoot implies that the closed-loop damping ratio should be close to unity, so choose $\zeta = 0.9$ and hence $\omega_n = 0.15/\zeta = 0.167$.

Step 3: Compute controller coefficients

We need to solve:

$$\frac{1 + 0.5K_p}{10} = 2\zeta\omega_n \quad \text{and} \quad \frac{0.5K_i}{10} = \omega_n^2$$

With $\zeta = 0.90$, $\zeta\omega_n = 0.15$ and $\omega_n = 0.167$ we obtain

$$K_p = 4.0 \quad \text{and} \quad K_i = 0.56$$

Step 4: Use simulation tools to assess the performance and examine the disturbance rejection issues. We give the result of a reference tracking simulation (Figure 12.8) to see if we have met the desired specification on settling time, overshoot and steady state error. We see that the value for $t_s(5\%)$ is just over 10 minutes, but well under the desired specification of 20 minutes. The overshoot is less than 5% and the steady state error is zero.

Step 5: Implement PI controller

$$K(s) = 4 + \frac{0.56}{s}$$

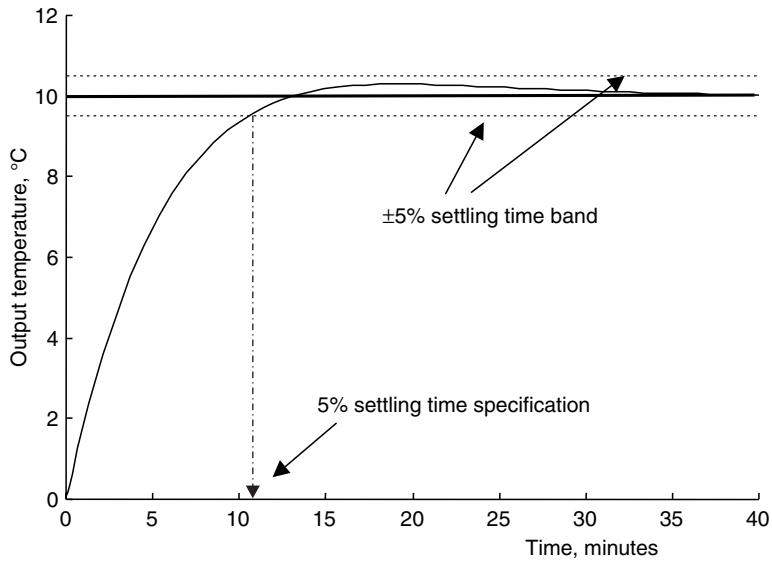


Figure 12.8 PI design with a model-based approach.

Remark

When this PI tuning problem was solved by manual tuning, a total of five hours of experiment time and production disruption occurred. The manual tuning method eventually determined suitable PI settings as $K_p = 4.75$ and $K_i = 0.5$. By way of contrast, the model based procedure above quickly gave very similar PI settings. Again the use of an engineering procedure led to a significant saving in person power costs and significantly reduced lost production.

12.5 Proportional and derivative control procedures

In most applications of three-term control, the structure of the controller will use the P and I terms. The reason is, as we have seen, that PI control will eliminate steady offsets in reference tracking and remove similar offsets caused by constant load disturbances. Recently this guaranteed property was referred to as *the magic of integral control*; indeed it is a very powerful practical property widely exploited in industrial applications. By way of contrast, the derivative term is a rather specialist tool in the PID toolbox, nonetheless it has its uses.

Proportional-derivative control is often found in electric motor-driven position systems (Figure 12.9). The mathematical relationship between angular position, θ and angular velocity ω of a rotating shaft is given by

$$\omega = \frac{d\theta}{dt} \quad \text{or} \quad \theta(t) = \int^t \omega(\tau) d\tau$$

and this leads to a pure integrator in the system transfer function model. Consider the following sequence of modelling diagrams, as given in Figure 12.10.

Thus it can be seen that the motor–shaft system has a pure integrator (or a pole at the position of $s = 0$) in the transfer function model. Proceeding now to the control design stage, the first step in the general procedure for designing a control system was to *know*

the plant. Let us consider an analysis for this general form of transfer function plant model, which is known as a Type 1 system.

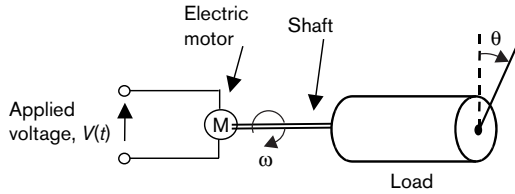


Figure 12.9 Schematic of d.c. motor position control.

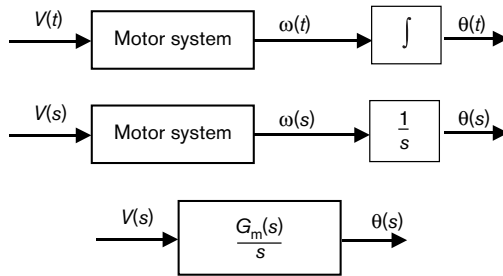


Figure 12.10 Block diagram for position control system showing integrator.

Remark

Definition of system type

One form of system classification is concerned with the number of pure integrators existing in a system model. Write

$$G(s) = \frac{n(s)}{s^\ell d(s)}$$

where $n(s) = b_m s^m + \dots + b_0$ and $d(s) = s^n + a_{n-1} s^{n-1} + \dots + a_0$; then the index ℓ is known as the system type index.

- A Type 0 plant has no pure integrators
- A Type 1 plant has one pure integrator
- A Type 2 plant has two pure integrators

Example: Reference tracking and output/load disturbance rejection for a Type 1 system

For the analysis we use the usual reference tracking and load disturbance rejection block diagram (Figure 12.11) and consider the proportional control of a Type 1 plant.

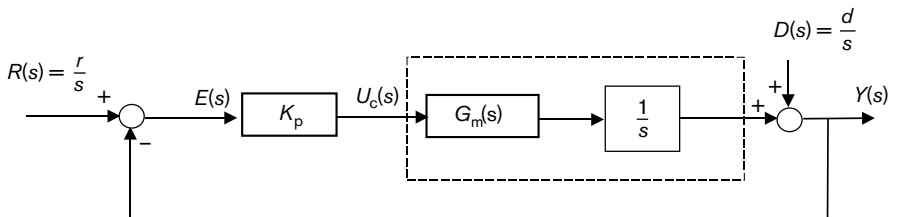


Figure 12.11 Proportional control of Type 1 system.

The closed-loop expressions are

$$\begin{aligned} Y(s) &= G_{CL}(s)R(s) + S(s)D(s) \\ &= \frac{G_m(s)K_p}{s + G_m(s)K_p} R(s) + \frac{s}{s + G_m(s)K_p} D(s) \end{aligned}$$

from which it follows that

$$\text{Reference tracking performance: } Y_R(s) = \frac{G_m(s)K_p}{s + G_m(s)K_p} R(s)$$

$$\text{Disturbance rejection performance: } Y_D(s) = \frac{s}{s + G_m(s)K_p} D(s)$$

We make the assumption that $G_m(s)$ is stable with finite system gain and we also assume that K_p has been selected to stabilise the closed-loop system. This allows us to investigate the steady state properties of the closed-loop system for a step reference signal $R(s) = (r/s)$ and a constant load disturbance of the form $D(s) = (d/s)$, and we find

$$y_{Rss} = \lim_{t \rightarrow \infty} y_R(t) = \lim_{s \rightarrow 0} sY_R(s) = \lim_{s \rightarrow 0} s \left(\frac{G_m(s)K_p}{s + G_m(s)K_p} \right) \frac{r}{s} = r$$

and

$$y_{Dss} = \lim_{t \rightarrow \infty} y_D(t) = \lim_{s \rightarrow 0} sY_D(s) = \lim_{s \rightarrow 0} s \left(\frac{s}{s + G_m(s)K_p} \right) \frac{d}{s} = 0$$

Thus the outcome of this analysis is that proportional control of a Type 1 plant will have perfect steady state reference tracking, and perfect steady state disturbance rejection. We have demonstrated these properties with only some mild assumptions on the $G_m(s)$ transfer function, and we have shown that for the Type 1 plant we do not need the integral term in the three-term controller. The question now is: what advantage does derivative control bring to the design?

To gain insight into the value of derivative control we introduce a first-order model for the process $G_m(s)$ and look at some alternative structures of the PD controller.

Textbook PD control

We use the full closed-loop assessment diagram with a PD controller in the forward path of the closed-loop (Figure 12.12).

The usual block diagram analysis yields:

$$Y(s) = Y_R(s) + Y_D(s)$$

with

$$Y_R(s) = G_{CL}(s)R(s) = \frac{K(K_p + K_d s)}{\tau s^2 + (1 + KK_d)s + KK_p} R(s)$$

and

$$Y_D(s) = S(s)D(s) = \frac{s(\tau s + 1)}{\tau s^2 + (1 + KK_d)s + KK_p} D(s)$$

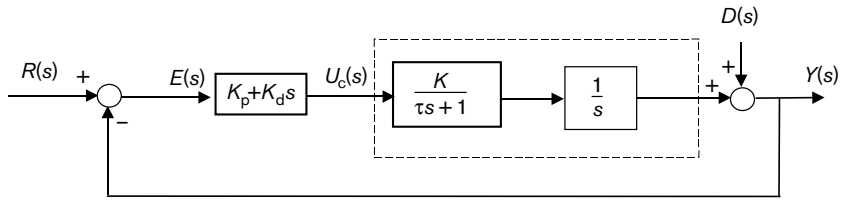


Figure 12.12 PD control of system.

The closed-loop stability of the system will be given by the denominator of $G_{CL}(s)$ and $S(s)$, which is seen to be

$$d_{CL}(s) = \tau s^2 + (1 + KK_d)s + KK_p = 0$$

If we divide through by τ then we can make the important identification with the standard second-order transfer function form involving closed-loop design parameters ζ and ω_n :

$$\frac{d_{CL}(s)}{\tau} = s^2 + \left(\frac{1 + KK_d}{\tau}\right)s + \frac{KK_p}{\tau} = s^2 + 2\zeta\omega_n s + \omega_n^2 = d_{2nd}(s)$$

From this come the design equations:

$$\frac{KK_p}{\tau} = \omega_n^2 \quad \text{and} \quad \frac{1 + KK_d}{\tau} = 2\zeta\omega_n$$

where K_p can be used to set ω_n and K_d used to set the damping ratio ζ , or vice versa. Thus the design utility of PD control is that it can be used to meet the closed-loop stability requirements and shape the transient response of the closed-loop. However, there are some remarks to be made before we adopt a design procedure.

Remark

1. If we look again at the form of $G_{CL}(s)$, this is as follows:

$$G_{CL}(s) = \frac{K(K_p + K_d s)}{\tau s^2 + (1 + KK_d)s + KK_p}$$

We see that $G_{CL}(0)$ is unity, and this guarantees good steady state reference tracking, but we also note that the numerator $n_{CL}(s) = K(K_p + K_d s)$ has a zero and so $G_{CL}(s)$ is not quite the usual second-order system form, which does not have the zero. In some cases this zero will restrict the speed of the response obtainable by the controller or cause excessive overshoot.

2. The second problem arising is one from the practical implementation of PD control. For this, consider the control law where we have

$$\begin{aligned} U_c(s) &= [K_p + K_d(s)]E(s) \\ &= \underbrace{K_p E(s)}_{\text{Proportional action}} + \underbrace{K_d s [R(s) - Y(s)]}_{\text{Derivative action}} \end{aligned}$$

Looking at the derivative term only, we know that the reference signal will be a sequence of step change signals, while the measured output signal will be slower and smoother in changing. Both of these signal types will be differentiated by the derivative control term.

When the reference signal is constant, then its derivative is zero, but at the point where the step changes value the derivative action will produce an impulse-like feature or spike on the control signal. This is known as *derivative kick*. This effect could lead to, say, a voltage spike entering the actuator device circuitry, which is not likely to be beneficial. To accommodate this real-world practical effect, it is common to use a slightly different structure for PD control. We use this new structure as the basis for a PD design procedure in this chapter. We should note that there are other important practical aspects to be considered for derivative control, but these must wait until Chapter 18 for a more detailed presentation.

12.5.1 Proportional control on error, derivative control on measured variable

To overcome the problems of derivative kick, and to obtain a closer link to the standard second-order transfer function form, we apply PD control in the structure of Figure 12.13.

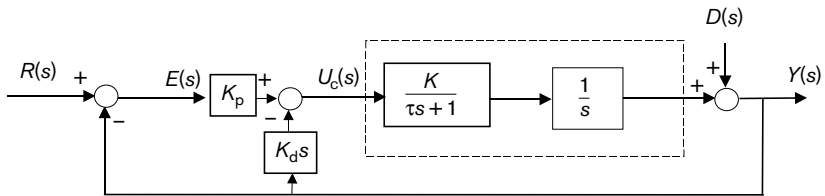


Figure 12.13 Derivative term acting on the error signal.

The block diagram analysis follows as:

$$Y(s) = D(s) + \frac{K}{s(\tau s + 1)} [K_p(R(s) - Y(s)) - K_d s Y(s)]$$

giving

$$Y(s) = \frac{KK_p}{\tau s^2 + (1 + KK_d)s + KK_p} R(s) + \frac{s(\tau s + 1)}{\tau s^2 + (1 + KK_d)s + KK_p} D(s)$$

and

$$Y_R(s) = G_{CL}(s)R(s) = \frac{KK_p}{\tau s^2 + (1 + KK_d)s + KK_p} R(s)$$

$$Y_D(s) = S(s)D(s) = \frac{s(\tau s + 1)}{\tau s^2 + (1 + KK_d)s + KK_p} D(s)$$

where we see that the transfer function $G_{CL}(s)$ resembles our standard second-order transfer function with no zero in the numerator. We have previously used standard second-order systems theory to link simple design performance values to the parameters ζ and ω_n . We recall these results in Table 12.3.

We use the above formulas to translate overshoot and settle time specifications into closed-loop damping ratio and natural frequency values. The link to the control design then uses the closed-loop characteristic equation, $d_{CL}(s)$, which is a function of the Type 1 system model parameters K and τ and the coefficients of the PD controller, K_p and K_d . The standard second-order system characteristic equation is given by

$$d_{2nd}(s) = s^2 + 2\zeta\omega_n s + \omega_n^2 = 0$$

Table 12.3 Second-order system design links.

Design specification	Closed-loop plant parameters
Overshoot OS(%) > 0%	$L_{os} = \ln\left(\frac{OS(\%)}{100}\right)$ $\zeta = \frac{ L_{os} }{\sqrt{\pi^2 + L_{os}^2}}$
Overshoot OS(%) = 0%	$\zeta = 1$
Settle time, $t_s(5\%)$	$\zeta\omega_n = \frac{3}{t_s(5\%)}$
Settle time, $t_s(2\%)$	$\zeta\omega_n = \frac{4}{t_s(2\%)}$
Damping ratio, ζ and natural frequency, ω_n , or damped natural frequency, ω_d specified	Target $d_{CL}(s) = s^2 + 2\zeta\omega_n s + \omega_n^2$ or $d_{CL}(s) = s^2 + a_1 s + a_0$

We compare this to the expression

$$d_{CL}(s) = \tau s^2 + (1 + KK_d)s + KK_p = 0 \text{ or equivalently } s^2 + \frac{(1 + KK_d)}{\tau} s + \frac{KK_p}{\tau} = 0$$

Equating the coefficients of $d_{2nd}(s)$ and $d_{CL}(s)$ gives the required PD controller coefficients:

$$\frac{1 + KK_d}{\tau} = 2\zeta\omega_n \text{ and } \frac{KK_p}{\tau} = \omega_n^2$$

and these give

$$K_d = \frac{2\zeta\omega_n\tau - 1}{K} \quad K_p = \frac{\omega_n^2\tau}{K}$$

We are able to collect these links together to design a PD controller to meet transient response specifications. This procedure is formalised as follows.

Procedure D: Proportional and derivative control of a process with a simple Type 1 model

Step 1: Use the process identification method to find a first-order component model for the system response. The model transfer function is assumed to have the form:

$$G(s) = \frac{K}{s(\tau s + 1)}$$

Step 2: Controller target specification

- (a) The Type 1 model assumption ensures zero steady state offset for both step reference signals tracking and constant output disturbance signals.
- (b) Specify overshoot OS(%):

$$\text{If OS}(\%) > 0 \text{ then } \zeta = \frac{|L_{os}|}{\sqrt{\pi^2 + L_{os}^2}} \text{ with } L_{os} = \ln\left(\frac{\text{OS}(\%)}{100}\right)$$

$$\text{If OS}(\%) = 0 \text{ then } \zeta = 1$$

(c) Specify settle time requirement $t_s(5\%)$ or $t_s(2\%)$

$$\text{If } t_s(5\%) \text{ then } \zeta\omega_n = 3/t_s(5\%)$$

$$\text{If } t_s(2\%) \text{ then } \zeta\omega_n = 4/t_s(2\%)$$

(d) Specify damping ratio, ζ , and damped natural frequency, ω_d .

(e) The above specification can be cast as a target closed-loop characteristic equation of either of the forms

$$d_{CL}(s) = s^2 + 2\zeta\omega_n s + \omega_n^2$$

or

$$d_{CL}(s) = s^2 + a_1 s + a_0$$

Step 3: Compute controller coefficients, K_p , K_d .

Solve

$$\frac{1 + KK_d}{\tau} = a_1 \text{ which gives } K_d = \frac{a_1\tau - 1}{K}$$

Solve

$$\frac{KK_p}{\tau} = a_0 \text{ which gives } K_p = \frac{a_0\tau}{K}$$

Step 4: Use simulation tools to assess the performance and examine the disturbance rejection performance.

Step 5: Implement PD control law as

$$U(s) = K_p E(s) + K_d s Y(s)$$

Problem Figure 12.14 shows the Type 1 model of an experiment where a d.c. motor is being used for position control.

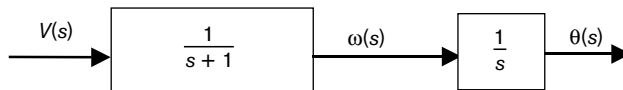


Figure 12.14 d.c. motor example.

A design for proportional control on error and derivative control on measured variable, $\theta(t)$, is required which satisfies the following design specification:

Specification: the closed loop is to have damping ratio of $\zeta = 0.6$ and a damped natural frequency of 10 rad/s.

(a) Calculate the appropriate proportional and derivative gains.

(b) Investigate the system response to a reference step of $r(t) = 0.1$.

- (c) Examine the control signal being applied to the system input. Comment on the practical feasibility of such a control input.

Solution (a) We use the steps of Procedure D, but since a Type 1 model has been given for the process we go directly to Step 2.

Step 2: Controller target specification

The closed loop is to have damping ratio of $\zeta = 0.6$ and a damped natural frequency of 10 rad/s. We use the relation between ω_d , ω_n and ζ to find a value for ω_n .

$$\omega_d = \omega_n \sqrt{1 - \zeta^2}$$

Hence

$$10 = \omega_n \sqrt{1 - 0.6^2} \text{ yields } \omega_n = \frac{10}{\sqrt{0.64}} = 12.5$$

The Type 1 model information is $K = 1$ and $\tau = 1$. The design specification is $\zeta = 0.6$ and $\omega_n = 12.5$, which gives a target closed-loop characteristic equation of

$$a_1 = 2\zeta\omega_n = 15.0$$

$$a_0 = \omega_n^2 = 156.25$$

with $d_{CL}(s) = s^2 + a_1s + a_0$.

Step 3: Compute controller coefficients, K_p , K_d .

Solve

$$\frac{1 + KK_d}{\tau} = a_1 \text{ which gives } K_d = \frac{a_1\tau - 1}{K} = 14.0$$

Solve

$$\frac{KK_p}{\tau} = a_0 \text{ which gives } K_p = \frac{a_0\tau}{K} = 156.25$$

Step 4: We use the PD controller in the form: $U(s) = 156.25E(s) + 14.0Y(s)$.

- (b) Investigate the system response to a reference step of $r(t) = 0.1$

We use a Simulink simulation for the step reference investigation with a step occurring at $t = 0.5$ seconds. The model used is given in Figure 12.15.

The reference step response trace is given in Figure 12.16 and shows a response with about 6% overshoot.

- (c) Using the same Simulink simulation the control signal is given in Figure 12.17.

Clearly, even with the more practical proportional control on error and derivative control on measured variable structure, we have a control signal input which is not very desirable. It has a spike, and it has large magnitude despite the fact that the reference change is only a small one. This is an example of proportional kick, which can be another implementational problem. The reference change is $r = 0.1$, the proportional gain is 156.2 and the control signal is approximately $15.6 = 0.1 \times 156.2$. A real PD controller like this could not be used, and as we see later in Chapter 18, various modifications are made to make PID control practical.

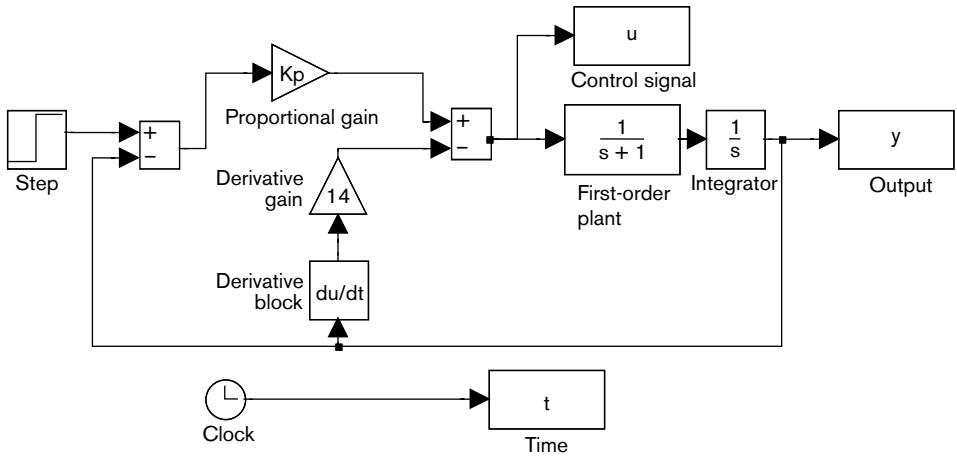


Figure 12.15 Simulink model of d.c. motor.

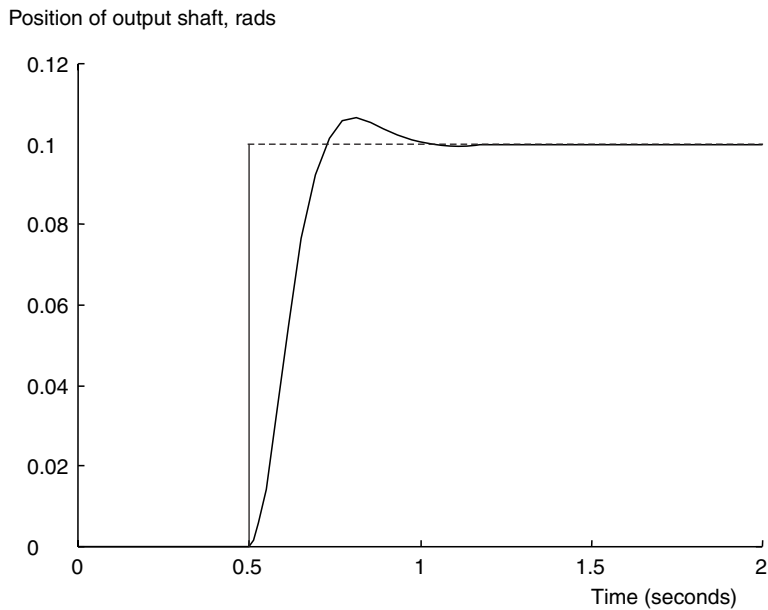


Figure 12.16 Step response of d.c. motor.

As a closing remark on this PD control section, it is worth mentioning that the motivation was the classical d.c. motor position and tachogenerator control loop applications. We have used a load or output disturbance in Figure 12.12 and Figure 12.13 because PD control, Type 1 plant and these disturbances have some good analytical properties. However, in any practical situation, the analysis and the control design must use the disturbances as they affect the plant. For example, if the application were subject to torque disturbances at the motor input, then we would find that not all the good properties found here for PD control would apply and we would advise some other PID controller solution.

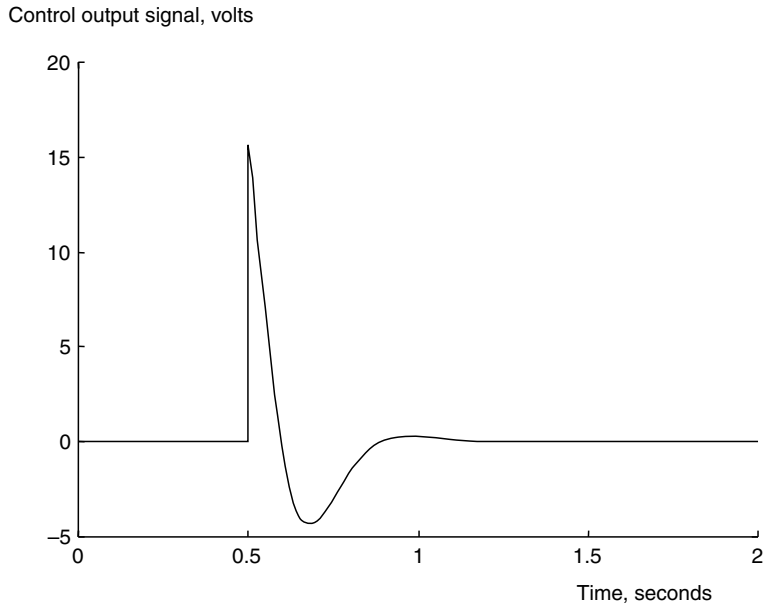


Figure 12.17 Control signal for d.c. motor.

12.6 PID controller design by pole placement

In devising methods to achieve a three-term control design we have introduced a simple transfer function model for the system and used closed-loop requirements to help us derive the controller coefficients. This approach exploits any knowledge we have on the process or requires us to find a simple model using system modelling or identification methods. As the design became more involved we learnt how we often have to resort to simulation to see how the design is meeting other design requirements. For example, we might design for good reference tracking, but need reasonable disturbance rejection performance too. We also learnt something of the intuitive and guaranteed properties of the terms in the PID controller.

One of the ideas which has been emerging from the work is that of turning the specification into a target closed-loop characteristic polynomial and then selecting the controller coefficients to match this polynomial. We conclude this chapter by looking at a simple pole placement method for PID control.

Pole placement process model

We assume a second-order model for the process given by:

$$G(s) = \frac{b_1s + b_0}{s^2 + a_1s + a_0}$$

The value of the pole placement method will rely on the versatility of this model to represent a wide range of typical processes. We note only two useful ones here:

1. *Second-order oscillatory system*

Set $b_1 = 0$; then

$$G(s) = \frac{b_0}{s^2 + a_1s + a_0} = \frac{b_0/a_0}{(1/a_0)s^2 + (a_1/a_0)s + 1} = \frac{K}{(1/\omega_n^2)s^2 + (2\zeta/\omega_n)s + 1}$$

Thus underdamped oscillatory systems can be represented by the model.

2. *First-order system with time delay*

Let

$$G(s) = \frac{Ke^{-sT_d}}{\tau s + 1}$$

and replace the time delay by a first-order Padé approximation, namely,

$$e^{-sT_d} = \frac{(1 - 0.5T_d s)}{(1 + 0.5T_d s)}$$

and hence

$$\begin{aligned} G(s) &\equiv \frac{K(1 - 0.5T_d s)}{(\tau s + 1)(1 + 0.5T_d s)} = \frac{-0.5KT_d s + K}{0.5T_d \tau s^2 + (\tau + 0.5T_d) s + 1} \\ &= \frac{\left(-\frac{K}{\tau}\right)s + \left(\frac{K}{0.5T_d}\right)\tau}{s^2 + \left(\frac{\tau + 0.5T_d}{0.5T_d \tau}\right)s + \left(\frac{1}{0.5T_d \tau}\right)} = \frac{b_1 s + b_0}{s^2 + a_1 s + a_0} \end{aligned}$$

PID control block diagram

We use the textbook form of the PID controller given by

$$G_c(s) = K_p + \frac{K_i}{s} + K_d s$$

We introduce the closed-loop controller assessment block diagram as in Figure 12.18.

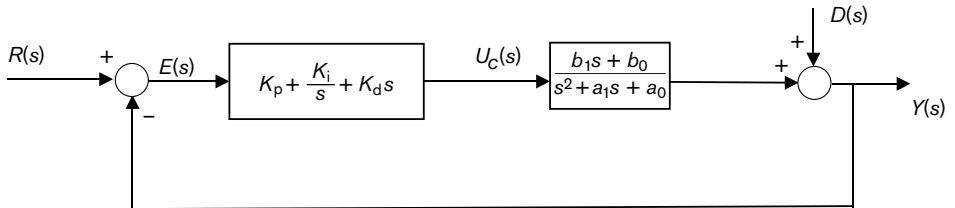


Figure 12.18 Block diagram for general pole placement problem.

Analysis of the block diagram yields

$$Y(s) = Y_R(s) + Y_D(s)$$

with

$$Y_R(s) = G_{CL}(s)R(s) = \frac{n_{CL}(s)}{d_{CL}(s)}R(s) \quad \text{and} \quad Y_D(s) = S(s)D(s) = \frac{n_S(s)}{d_{CL}(s)}D(s)$$

where the common denominator polynomial is given by

$$d_{CL}(s) = s(s^2 + a_1s + a_0) + (K_d s^2 + K_p s + K_i)(b_1s + b_0)$$

and the numerator polynomials of $G_{CL}(s)$ and $S(s)$ are

$$n_{CL}(s) = (K_d s^2 + K_p s + K_i)(b_1s + b_0)$$

$$n_s(s) = s(s^2 + a_1s + a_0)$$

Design specification

The pole placement method requires that the engineer specifies a design closed-loop pole polynomial, and then the controller coefficients are chosen to match the design polynomial. In this case, the assumed four-parameter second-order model structure and the use of a full PID controller lead to a third order closed-loop polynomial $d_{CL}(s)$. An appropriate choice of a target closed-loop polynomial is to use one real pole (at $s = -\alpha \omega_n$) and a quadratic pair as

$$d_{design}(s) = (s + \alpha \omega_n)(s^2 + 2\zeta\omega_n s + \omega_n^2)$$

The two components of the specification are completed as follows:

1. Second-order pole pair in the quadratic factor:

$$d_{2nd}(s) = (s^2 + 2\zeta\omega_n s + \omega_n^2)$$

As the notation suggests, this pole pair can be chosen as an underdamped pole specification or as desired LHP closed-loop pole positions selected by the control engineer. Overshoot, settle time, damping ratio or other relationships can be used in this step.

2. Real pole in the factor:

$$d_{1st}(s) = (s + \alpha \omega_n)$$

We choose this pole to be faster than the quadratic pair so that the response is dominated by the second-order response characteristic.

Design equations

$$d_{design}(s) = (s + \alpha \omega_n)(s^2 + 2\zeta\omega_n s + \omega_n^2) = s^3 + (\alpha + 2\zeta)\omega_n s^2 + (1 + 2\alpha\zeta)\omega_n^2 s + \alpha \omega_n^3$$

The closed-loop polynomial expands as

$$d_{CL}(s) = (1 + K_d b_1)s^3 + (a_1 + K_d b_0 + K_p b_1)s^2 + (a_0 + K_p b_0 + K_i b_1)s + K_i b_0$$

To equate powers use:

$$\frac{d_{CL}(s)}{(1 + K_d b_1)} = s^3 + \left(\frac{a_1 + K_d b_0 + K_p b_1}{1 + K_d b_1}\right)s^2 + \left(\frac{a_0 + K_p b_0 + K_i b_1}{1 + K_d b_1}\right)s + \left(\frac{K_i b_0}{1 + K_d b_1}\right)$$

Equating coefficients of like powers yields:

For s^2 : $a_1 + K_d b_0 + K_p b_1 = (1 + K_d b_1)(\alpha + 2\zeta)\omega_n$

For s : $a_0 + K_p b_0 + K_i b_1 = (1 + K_d b_1)(1 + 2\alpha\zeta)\omega_n^2$

For s^0 : $K_i b_0 = (1 + K_d b_1)\alpha \omega_n^3$

This leads to a simple linear system of equations where α , ζ and ω_n are known from the design specification and a_1 , b_1 and a_0 , b_0 are known from the process model. The equations can then be solved for the PID coefficients K_p , K_i , K_d :

$$\begin{bmatrix} b_1 & 0 & b_0 - b_1(\alpha + 2\zeta)\omega_n \\ b_0 & b_1 & -b_1(1 + 2\alpha\zeta)\omega_n^2 \\ 0 & b_0 & -b_1\alpha\omega_n^3 \end{bmatrix} \begin{pmatrix} K_p \\ K_i \\ K_d \end{pmatrix} = \begin{bmatrix} -a_1 + (\alpha + 2\zeta)\omega_n \\ -a_0 + (1 + 2\zeta)\omega_n^2 \\ \alpha\omega_n^3 \end{bmatrix}$$

It is worth remarking that the above pole placement equations do not guarantee a feasible controller because the constraint that the PID gains must be positive has not been built into the formulas. Sometimes a PID gain will be computed as a negative value and it is necessary to modify the specification to obtain a sensible answer. When this happens, the method is really indicating that the control problem is difficult to solve and deserves closer investigation.

Procedure E: PID controller design by pole placement

Step 1: Use process identification or modelling methods to find a transfer function representation of the form:

$$G(s) = \frac{b_1s + b_0}{s^2 + a_1s + a_0}$$

Identify the values of the parameters, b_1 , b_0 , a_1 , a_0 .

Step 2: Controller design closed-loop characteristic polynomial specification

(a) For the quadratic factor, $d_{2nd}(s) = (s^2 + 2\zeta\omega_n s + \omega_n^2)$, use

Design specification	Closed-loop plant parameters
Overshoot OS(%) > 0%	$L_{os} = \ln\left(\frac{OS(\%)}{100}\right)$ $\zeta = \frac{ L_{os} }{\sqrt{\pi^2 + L_{os}^2}}$
Overshoot OS(%) = 0%	$\zeta = 1$
Settling time, $t_s(5\%)$	$\zeta\omega_n = \frac{3}{t_s(5\%)}$
Settling time, $t_s(2\%)$	$\zeta\omega_n = \frac{4}{t_s(2\%)}$
Damping ratio, ζ , and natural frequency, ω_n or damped natural frequency, ω_d , specified	Target $d_{CL}(s) = s^2 + 2\zeta\omega_n s + \omega_n^2$ or $d_{CL}(s) = s^2 + a_1s + a_0$

(b) Real pole in the factor $d_{1st}(s) = (s + \alpha\omega_n)$, select the parameter, α , to give a faster pole than the quadratic pair.

(c) Tabulate α , ζ , ω_n .

Step 3: Compute controller coefficients, K_p , K_i , K_d . Use the matrix equations

$$\begin{bmatrix} b_1 & 0 & b_0 - b_1(\alpha + 2\zeta)\omega_n \\ b_0 & b_1 & -b_1(1 + 2\alpha\zeta)\omega_n^2 \\ 0 & b_0 & -b_1\alpha\omega_n^3 \end{bmatrix} \begin{pmatrix} K_p \\ K_i \\ K_d \end{pmatrix} = \begin{bmatrix} -a_1 + (\alpha + 2\zeta)\omega_n \\ -a_0 + (1 + 2\zeta)\omega_n^2 \\ \alpha\omega_n^3 \end{bmatrix}$$

Step 4: Use simulation tools to assess the performance and examine the disturbance rejection performance.

Step 5: Implement PD control law as

$$U(s) = \left(K_p + \frac{K_i}{s} + K_d s \right) E(s)$$

Problem A simple boiler system is modelled by a second-order transfer function with d.c. gain of unity, damping ratio of 0.2 and natural frequency of 0.5 rad/s. A PID design is required for plant implementation.

- (a) Use the pole placement method such that the dominant second-order closed-loop has poles at $s = -0.7 \pm j0.7$.
- (b) Use a Simulink simulation to examine the unit step response.

Solution (a) We follow the Procedure E.

Step 1: Use process identification or modelling methods to find a transfer representation of the form:

$$G(s) = \frac{b_1 s + b_0}{s^2 + a_1 s + a_0}$$

Identify the values of the parameters, b_1, b_0, a_1, a_0 .

The data gives the process model as

$$G(s) = \left[\frac{K}{(1/\omega_n^2)s^2 + (2\zeta/\omega_n)s + 1} \right] \\ = \frac{1}{(1/0.5^2)s^2 + (2 \times 0.2/0.5)s + 1} = \frac{0.25}{s^2 + 0.2s + 0.25}$$

hence $b_1 = 0, b_0 = 0.25, a_1 = 0.2, a_0 = 0.25$.

Step 2: Controller design closed-loop characteristic polynomial specification

For the quadratic factor, $d_{2nd}(s) = (s^2 + 2\zeta\omega_n s + \omega_n^2)$, use the data as $s = -0.7 \pm j0.7$. This gives the parameter values as $\zeta = 0.71$ and $\omega_n = 0.99$.

For the real pole in the factor $d_{1st}(s) = (s + \alpha\omega_n)$, select the parameter, α , to give a faster pole than the quadratic pair; chose $s = -1$, giving $\alpha = 1.01$.

Step 3: Compute the controller coefficients, K_p, K_i, K_d , using the matrix equation. Enter the parameters $a_0, a_1, b_0, b_1, \omega_n, \zeta$ and α in MATLAB as a0, a1, b0, b1, om, z and a1pha. Then introduce MATLAB code as follows:

```
% Model Parameters
b1=0; b0=0.25; a1=0.2; a0=0.25;

%Design Parameters
z=0.71; om=0.99; alpha=1.01;
```

```

%Matrix Equation Entries
M(1,1)=b1; M(1,2)=0; M(1,3)=b0-b1*(alpha+2*z)*om;
M(2,1)=b0; M(2,2)=b1; M(2,3)=-b1*(1+2*alpha*z)*om*om;
M(3,1)=0; M(3,2)=b0; M(3,3)=-b1*alpha*om*om*om;
x(1,1)=-a1+(alpha+2*z)*om;
x(2,1)=-a0+(1+2*z)*om*om;
x(3,1)=alpha*om*om*om;
%Solution for PID
pid=inv(M)*x;

```

A run of the code gives the PID coefficients as: $K_p = 8.4874$, $K_i = 3.9200$, $K_d = 8.8228$.

(b) Use a Simulink simulation to examine the unit step response

This part covers Steps 4 and Step 5 of Procedure E. The simulation is given in Figure 12.19 and The unit step response plot is shown in Figure 12.20. The response shows a high amount of overshoot (approaching 30%) and if this is unacceptable, we would have to go back and reselect the parameters of our pole placement design polynomial.

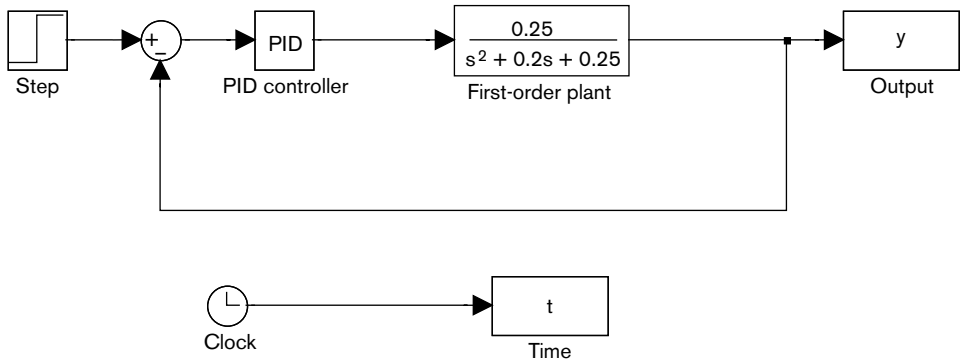


Figure 12.19 Simulink model for pole placement exercise.

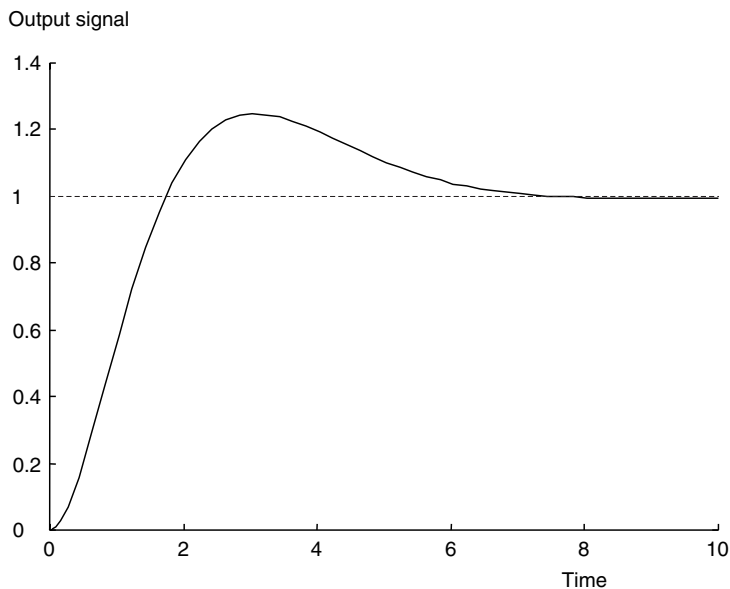


Figure 12.20 Output signal resulting from pole placement design.

What we have learnt

- ✓ Five procedures for tuning systems:
 - Procedure A: manual P, PI controller tuning
 - Procedure B: proportional control of a process with first-order system model
 - Procedure C: proportional and integral control of a process with a first-order model
 - Procedure D: proportional and derivative control of a process with a simple Type 1 model
 - Procedure E: PID controller design by pole placement
- ✓ Tuning PID controllers manually is a very time-consuming and inefficient procedure.
- ✓ Derivative control is especially suited in the PD control of systems represented by Type 1 system models.
- ✓ Using the knowledge gained on first and second-order models enables us to develop PID tuning procedures which are effective and efficient.

Multiple choice

- M12.1** The PID controller given by $U(s) = [K_p + (K_i/s) + K_d s]E(s)$ is referred to as:
- (a) a cascade form
 - (b) a decoupled form
 - (c) an industrial form
 - (d) a series form
- M12.2** If steady state accuracy is required, then:
- (a) choose a P controller
 - (b) choose a PD controller
 - (c) choose a PI controller
 - (d) choose any of the above
- M12.3** Manual PI controller tuning involves:
- (a) tuning the proportional gain, K_p , until an acceptable response occurs
 - (b) tuning K_i after tuning K_p
 - (c) finding a balance between the values for K_p and K_i
 - (d) all of the above
- M12.4** A disadvantage with manual tuning is:
- (a) the time it takes to do all the tuning runs
 - (b) the wrong values of K_d often results
 - (c) only PI controllers can be manually tuned
 - (d) only unit step responses can be used with the process
- M12.5** For simple first-order systems, if we fix the value of K_p to give a required closed-loop time constant:
- (a) the required steady state value will also be achieved
 - (b) the required steady state value will not be achieved
 - (c) the required steady state value might be achieved
 - (d) the value of K_p has no effect on e_{ss} .
- M12.6** Choosing a PI controller to stabilise a control system means choosing the values of K_p and K_i such that:
- (a) there is no overshoot
 - (b) the steady state error is zero
 - (c) the closed-loop characteristic equation has roots on the real axis
 - (d) the closed-loop characteristic equation has roots in the LHP
- M12.7** A PD controller:
- (a) introduces a pole at the origin in the open-loop transfer function
 - (b) introduces a poles at the origin in the closed-loop transfer function
 - (c) introduces a RHP zero in the closed-loop transfer function
 - (d) introduces a LHP zero in the closed-loop transfer function

M12.8 Applying the D-term to the process output:

- (a) removes integral action
- (b) avoids derivative kick
- (c) produces proportional kick
- (d) gives reduced system damping

M12.9 A PID controllers:

- (a) cannot provide zero steady state error in the output
- (b) cannot be tuned to provide reasonable disturbance rejection
- (c) cannot be tuned to provide a fast system response
- (d) cannot guarantee disturbance rejection *and* reference tracking for all systems

M12.10 Pole placement design:

- (a) involves equating terms in the design polynomial with the open-loop polynomial
- (b) involves equating terms in the closed-loop characteristic equation with the design polynomial
- (c) involves producing a set of parameters to give no overshoot
- (d) involves producing a set of parameters to give a damping ratio of 0.707.

Questions: practical skills

Q12.1 It is common to find that step response tests are used to give a preliminary idea of the type of PID control needed in particular application. List at least four features that might be found in a step response test. Explain why these features are important for decisions about a possible PID design.

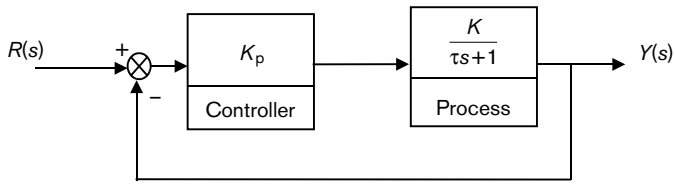
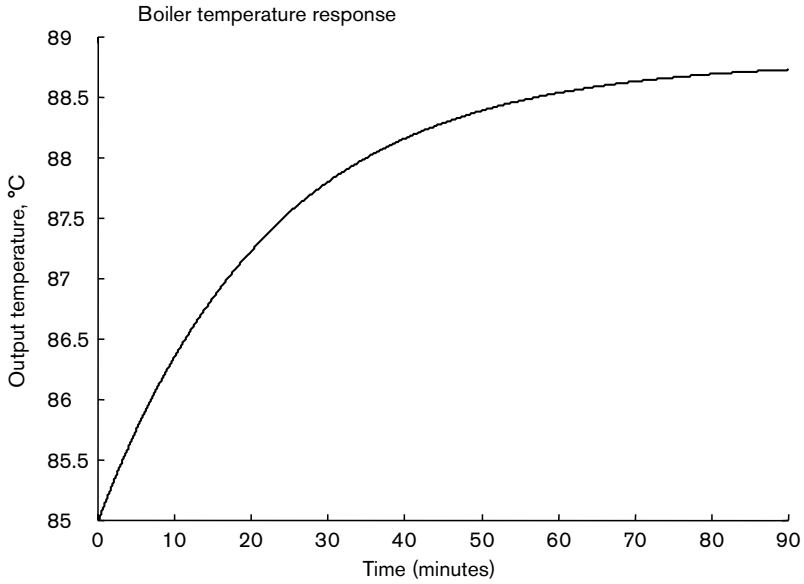
Q12.2 First-order system descriptions are commonly used as simple models. Give the transfer function for a first-order system. Identify and describe the d.c. gain and the time constant. Why is the d.c. gain so called? What multiple of the time constant corresponds to the 95% point? Prove your answer.

Q12.3 Knowledge of the system *type* will give immediate information on the system response. Complete the following table and answer the control questions given.

System	System type	Control question
$G(s) = \frac{k}{\tau s + 1}$?	What is the outcome of unity feedback with proportional control and a step reference input?
$G(s) = \frac{k}{s(\tau s + 1)}$?	What is the outcome of unity feedback with proportional control and (a) a step reference input and (b) a ramp input?
$G(s) = \frac{k}{s^2(\tau s + 1)}$?	What is the outcome of unity feedback with proportional control and (a) a step reference input, (b) a ramp input and (c) an acceleration input?

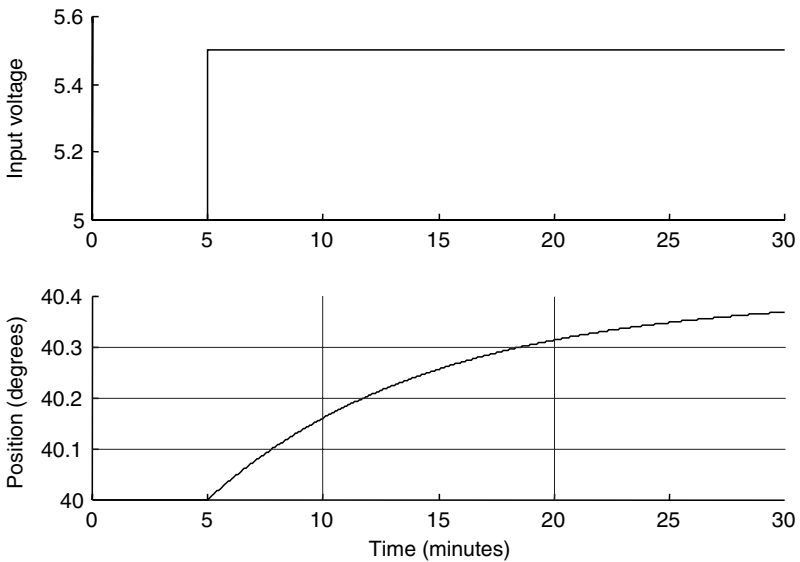
Q12.4 A boiler output step response is shown below, where the boiler input signal is changed from 4 m³ per minute to 4.5 m³ per minute. The initial steady output water temperature was 85 °C.

- (a) Using the step response plot, determine the parameters of a first-order transfer function model, K , and τ for the transfer function, $G(s) = K/(\tau s + 1)$.
- (b) It is proposed to use proportional control in a unity feedback control loop as shown. If the proportional gain is $K_p = 4$, use the transfer function you have found to show that the steady state offset to a reference change of 10 °C is 0.3 °C.
- (c) What structural change to the controller would you recommend to completely eliminate the offset?



Key: Reference input, $R(s) = 10/s$
 Temperature output, $Y(s)$

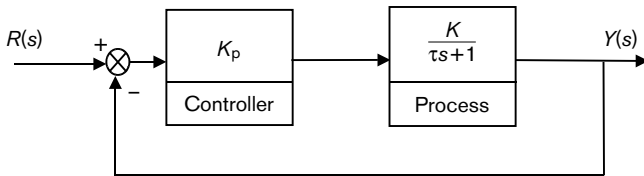
Q12.5 An engineer is testing a motor drive position system before designing a PID control system. The test procedure involves the usual plus- and minus-going step responses and the positive step response is shown in the figure.



- (a) Examine the plot carefully and list at least two particular aspects of the traces that might be important for the PID control design.
- (b) Use the trace to find a suitable Simulink simulation based on first-order system models. Verify your simulation against the trace.
- (c) Itemise the decisions that should be taken about the structure of the PID controller to be designed.
- (d) Use the manual test procedure to design a suitable PID controller which reaches 40.3 degrees in less than 5 minutes and exhibits no steady state error.

Problems

P12.1 The figure below shows a process with proportional control:



The closed-loop transfer function for the system is given by

$$Y(s) = \frac{K_{CL}}{\tau_{CL}s + 1} R(s)$$

Investigate the following problems:

- (a) Prove that the closed-loop gain may be expressed as

$$K_{CL} = \frac{K_p K}{(1 + K_p K)}$$

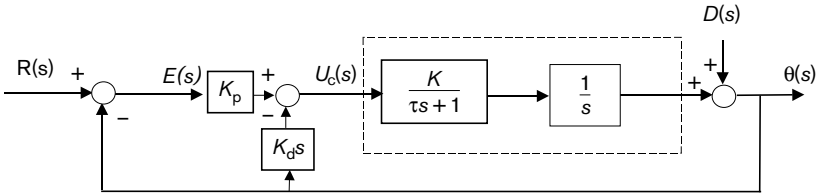
- (b) Prove that the closed-loop time constant is given by

$$\tau_{CL} = \frac{\tau}{(1 + K_p K)}$$

- (c) If $R(s) = r/s$, where this is a step of magnitude r , show that as $K_p \rightarrow \infty$ then $K_{CL} \rightarrow 1$. What does this mean for $e_{ss} = r - y_{ss}$?
- (d) Show that as $K_p \rightarrow \infty$ then $\tau_{CL} \rightarrow 0$. What does this mean?
- (e) How are the above theoretical results used in the tuning of proportional control of a first-order system?

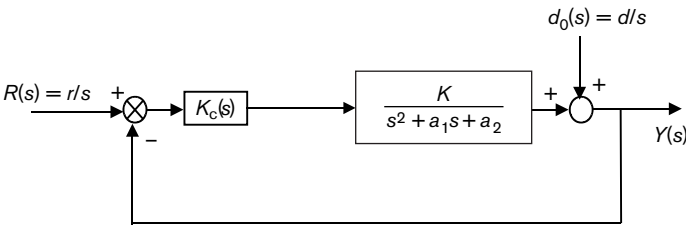
P12.2 A speed control loop in a manufacturing plant gave a 0.5 m s^{-1} change in speed when a 2% reference change was made. The speed measurement was found to be subject to a significant amount of measurement noise. The speed loop time constant was found to be 5 minutes. It is desired to find a three-term controller to ensure no steady state offset to step reference signals, to achieve a 5% settle time of less than 10 minutes at most and to have as little overshoot as possible. Use the steps of Procedure C, assume a first-order model for the speed loop and design a suitable three-term controller.

P12.3 The position control loop based on a d.c. motor actuator is a very common student laboratory experiment. The loop uses a Type 1 model and demonstrates how differential control can be used to tune system damping. A design for proportional control on error and derivative control on measured variable, $\theta(s)$, is required, as shown in the figure. The system time constant is given as 0.5 s and control specification is for critical damping with a natural frequency of 5 rad s^{-1} .

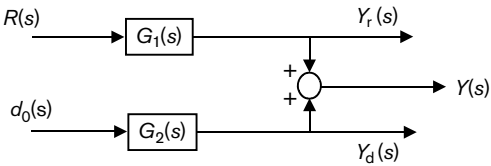


- (a) Calculate the appropriate proportional and derivative gains.
- (b) Investigate the system response to a reference step of $r(t) = 0.2$.
- (c) Comment on the practical feasibility of the control input produced.

P12.4 Knowledge of the expected behaviour of the common PID control configurations is valuable. In the following the investigation centres on PID with a second-order plant model.



(a) Find $G_1(s)$, $Y_r(s)$, $G_2(s)$ and $Y_d(s)$, according to the diagram below.



- (b) If the controller is set to $K_c(s) = K_p + (K_i/s)$, for which $K_p > 0$ and $K_i > 0$ and the closed loop is stable, then prove that there is no tracking reference offset, and that there is complete rejection of $d_0(s)$ in steady state.
- (c) The controller is set to $K_c(s) = K_p + K_d s$, for which $K_p > 0$ and $K_d > 0$ and the closed loop is stable. Prove that, if the closed-loop system is represented in standard second-order transfer function form, then ω_n depends on K_p , ζ can be fixed by K_d and the selection of K_d has no effect on the steady state performance of $Y_r(s)$, $Y_d(s)$.

13 Root locus for analysis and design



In Chapter 10, we learnt the benefits of knowing where the poles and the zeros lie in the s -plane. We emphasised the useful information on the system stability and performance which may be extracted from the knowledge of the position of the poles. We saw that for a stable closed-loop system all the closed-loop poles must strictly lie in the LHP. Since stability is the prime objective of a control system design, we always ensure that our controller stabilises the system by relocating the poles such that they all lie in the LHP.

Once we have ensured closed-loop stability, we then refine the controller to achieve the desired performance. The performance is often specified in terms of time domain indices (time constant, rise time, damping ratio, ...) which we have already met, and frequency domain indices (bandwidth, phase margin, ...), which we meet later in this book. Knowing the location of the poles can help us to design controllers which meet these specifications. For example, the time domain performance indices such as time constants for first-order systems and damping ratio and natural frequency of second-order system are related to the distance of the poles from $j\omega$ axis.

To achieve pole placement or relocation we can manipulate the position of the closed-loop poles by varying the controller gain K . A graphical aid for drawing the locus of closed-loop poles was introduced in 1948 by W. R. Evans. This method still remains a very useful control system design and analysis tool.

When the method was introduced, computers were not available. Hence many rules were developed for sketching the root locus diagram. It is now very easy to plot a root locus diagram using control engineering packages such as MATLAB. Thus we will not discuss in detail how to draw or sketch the root locus. We should, however, be able to understand how a root locus changes when we vary the controller gain from zero to infinity.

Learning objectives

- To understand the relationship between the system dynamics and the root locus.
- To become familiar with the basic principles of the root locus technique.
- To develop some skills in drawing and analysing the root locus diagram.

13.1 The relationship between the poles and system dynamic response: a summary

Figure 13.1 summarises what we have learnt in previous chapters about the relationship between system stability and the closed-loop poles (roots of the characteristic equation). Our main conclusion was that for a system to be stable the system poles should lie in the LHP; otherwise the system is unstable.

Our discussion on the relationship between the location of poles and the type of responses that are expected is summarised in Figure 13.2. For a stable system, the *dominant* poles are usually those with the smallest negative real parts. The system step responses become faster as the *dominant poles* move away from the $j\omega$ axis and deeper into the LHP. However, the unstable poles are dominant in any system.

In designing a controller using feedback, we attempt to move the poles of the system to desired locations in the LHP. Then the resulting closed-loop system may have better stability properties and, hopefully, improved performance. We firstly study how the system poles change using a proportional controller of gain K .

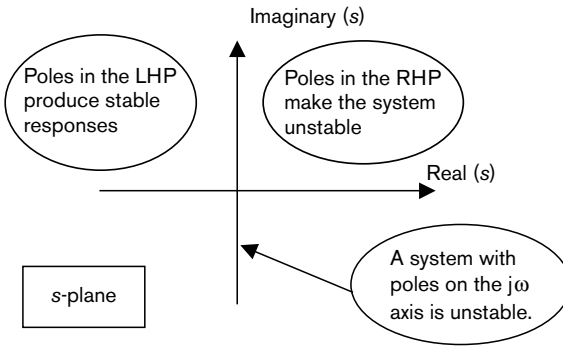


Figure 13.1 System stability and location of roots.

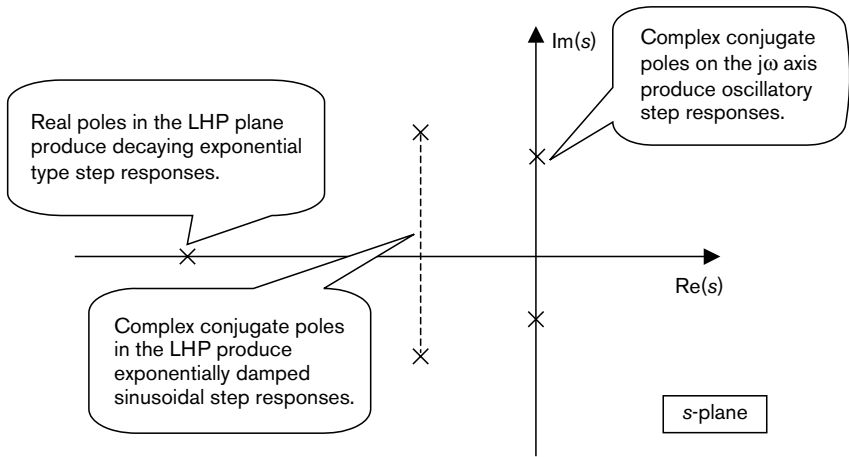


Figure 13.2 The relationship between system step responses and location of the poles.

13.2 Introducing the root locus

Consider the general unity feedback closed-loop system shown in Figure 13.3.

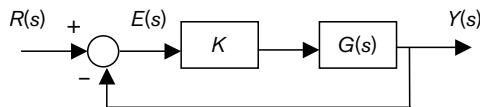


Figure 13.3 A unity feedback system.

If we let $G(s) = n(s)/d(s)$, where $n(s)$ and $d(s)$ are the numerator and denominator polynomials of the system transfer function $G(s)$, the open-loop poles are the roots of $d(s) = 0$. The closed-loop system transfer function is defined as:

$$Y(s) = \frac{KG(s)}{1+KG(s)}R(s) = G_{CL}(s)R(s)$$

which can be rewritten using our numerator and denominator polynomials as

$$G_{CL}(s) = \frac{K \frac{n(s)}{d(s)}}{1 + K \frac{n(s)}{d(s)}} = \frac{Kn(s)}{d(s) + Kn(s)}$$

The closed-loop poles can be determined from the roots of the denominator polynomial, $C(s)$, which we refer to as the closed-loop pole polynomial:

$$C(s) = d(s) + Kn(s) = 0$$

This equation shows that the controller K will directly change the polynomial and hence the system poles. By changing K , we change the position of the system poles on the s -plane. If we change K from 0 to infinity, we can calculate the position of the poles for all values of K . The plot of these poles on the s -plane is the locus of the closed-loop system poles and is called the *root locus*. It is a graphical representation of the closed-loop pole positions for increasing values of gain K .

We can see, very simply, that if we set $K = 0$, then $C(s) = d(s) = 0$, and we find the open-loop poles of the system.

We demonstrate the root locus principle by calculating manually what the root locus sketch will be. We then find that there are several useful rules which aid our sketching. And, of course, MATLAB will produce a root locus plot painlessly, but if we wish to understand and interpret the information we need to have some understanding of how the locus is formed.

We will need knowledge of the poles and zeros of the system transfer function to understand the root locus plot. We calculated the system zeros for a transfer function $G(s) = n(s)/d(s)$ by solving $n(s) = 0$. These zeros turn out to be the *finite* zeros of the system. In root locus applications we must be aware of the **pole-zero conservation principle**. This can be stated quite simply:

Key result: Pole-zero conservation principle

Given a transfer function $G(s) = n(s)/d(s)$, with n_p poles and n_z finite system zeros, we find

$$n_p = n_z + n_{z\infty}$$

where n_z is the number of the finite (or system) zeros and $n_{z\infty}$ is the number of *zeros at infinity*.

In essence, we must have the same number of zeros as poles, whether the zeros are finite or at infinity.

Skill section

Determining poles and finite or infinite zeros

Problem For the following transfer functions, find the poles and zeros and apply the pole-zero conservation principle to find the number of zeros at infinity.

(a) $G_1(s) = \frac{1}{(s+3)(s+5)}$

(b) $G_2(s) = \frac{s+3}{(s^2+3s+5)}$

$$(c) G_3(s) = \frac{s-2}{s(s+4)}$$

Solution

(a) Poles: solve $d(s) = 0$. $(s+3)(s+5) = 0$. $p_1 = -3$, $p_2 = -5$. $n_p = 2$

Zeros: solve $n(s) = 0$. No finite zeros. $n_z = 0$.

Pole-zero conservation principle: $n_p = n_z + n_{z\infty}$

Using $n_p = 2$ and $n_z = 0$, there are two zeros at infinity.

(b) Poles: solve $d(s) = 0$. $(s^2 + 3s + 5)(s + 6) = 0$. $p_{1,2} = -1.5 \pm \sqrt{11}/2$, $p_3 = -6$. $n_p = 3$.

Zeros: solve $n(s) = 0$. $(s + 3) = 0$. $n_z = 1$.

Pole-zero conservation principle: $n_p = n_z + n_{z\infty}$

Using $n_p = 3$ and $n_z = 1$, there are two zeros at infinity.

(c) Poles: solve $d(s) = 0$. $s(s+4) = 0$. $p_1 = 0$, $p_2 = -4$. $n_p = 2$.

Zeros: solve $n(s) = 0$. $s - 2 = 0$. $n_z = 1$.

Pole-zero conservation principle: $n_p = n_z + n_{z\infty}$

Using $n_p = 2$ and $n_z = 1$, there is one zero at infinity.

Problem

An engineer has modelled a servo system and found the open-loop transfer function representation to be:

$$G(s) = \frac{s+2}{s^2+4s+3}$$

(a) What are the zeros of the system? What are the open-loop poles?

(b) Identify the closed-loop pole polynomial equation, $C(s) = d(s) + Kn(s) = 0$, needed for a root locus plot.

(c) Solve the equation for $K = 0$.

(d) Show that as $K \rightarrow \infty$ the closed-loop poles approach the system zero positions.

(e) Tabulate the closed-loop pole positions for a range of proportional gains

$$K = 0.5, 1, 1.5, 2, 2.5, \dots$$

and draw a sketch of the closed-loop poles.

(f) Run the MATLAB `rlocus` command and verify the results.

Solution

(a) The system zeros are found by solving

$$n(s) = 0 \Rightarrow (s+2) = 0$$

This gives $s = -2$. Therefore there is a zero at $z = -2$.

The open-loop poles are found by solving

$$d(s) = 0 \Rightarrow (s^2 + 4s + 3) = 0$$

This results in $s = -3$ and $s = -1$. There are two poles at $p_1 = -3$ and $p_2 = -1$. By the pole-zero conservation principle, we must have the same number of zeros as poles: in this

case we have two poles and one system zero. We must therefore also have one zero at infinity.

- (b) The closed-loop pole polynomial equation is given by $C(s) = s^2 + 4s + 3 + K(s + 2) = 0$:

$$C(s) = s^2 + (4 + K)s + (3 + 2K) = 0$$

- (c) For $K = 0$, we obtain

$$C(s) = s^2 + (4 + K)s + (3 + 2K) = s^2 + 4s + 3 = (s + 1)(s + 3) = 0$$

giving $s = -1$ and $s = -3$. We can see from the answer to (a) that these are the open-loop pole positions.

- (d) To investigate what happens as $K \rightarrow \infty$, we solve the closed loop polynomial equation as follows:

$$C(s) = s^2 + 4s + 3 + K(s+2) = s^2 + (4 + K)s + (3 + 2K) = 0$$

Applying the quadratic root formula gives

$$s = \frac{-(4+K) \pm [(4+K)^2 - 4(3+2K)]^{0.5}}{2} = \frac{-(4+K) \pm (K^2 + 4)^{0.5}}{2}$$

Rearranging gives

$$s = \frac{-(4+K)}{2} \pm \frac{K}{2} \left(1 + \frac{4}{K^2}\right)^{0.5}$$

For very large K , the two closed loop poles become

$$s = \frac{-(4+K)}{2} \pm \frac{K}{2}$$

and $s = -2$ and $s \rightarrow -\infty$. Therefore as $K \rightarrow \infty$, one closed loop pole goes to the finite zero location at $s = -2$, and the other closed loop pole goes to $s = -\infty$, the infinite zero location.

- (e) To tabulate the closed-loop pole positions, we have to solve the second-order equation for $C(s)$. For example, if $K = 0.5$ then we solve

$$C(s) = s^2 + (4 + K)s + (3 + 2K) = s^2 + (4 + 0.5)s + (3 + 1) = s^2 + 4.5s + 4 = 0$$

We then find that $C(s)$ has two real roots $s = -3.28$ and $s = -1.219$ and these are the closed-loop poles of the system for the gain $K = 0.5$. Repeating this process for $K = 1, 1.5, \dots$, we find the results in Table 13.1. We notice that one pole ends at the value $s = -2$, the open-loop finite zero, and the other pole tends to infinity (the zero at infinity), as we predicted in part (d). The closed-loop poles are shown on the s -plane (Figure 13.4).

Table 13.1 Table of closed loop poles for different values of K .

K	0.0	0.5	1.0	1.5	2.0	2.5	∞
p_1	-3.0	-3.28	-3.62	-4.00	-4.41	-4.85	$-\infty$
p_2	-1.0	-1.22	-1.38	-1.50	-1.59	-1.65	-2

- (g) We can use MATLAB to help us plot the root locus. We need only enter the open-loop transfer function and MATLAB calculates the polynomial $C(s)$ and solves the equation $C(s) = 0$ for us. The following MATLAB code produces the plot shown in Figure 13.5.

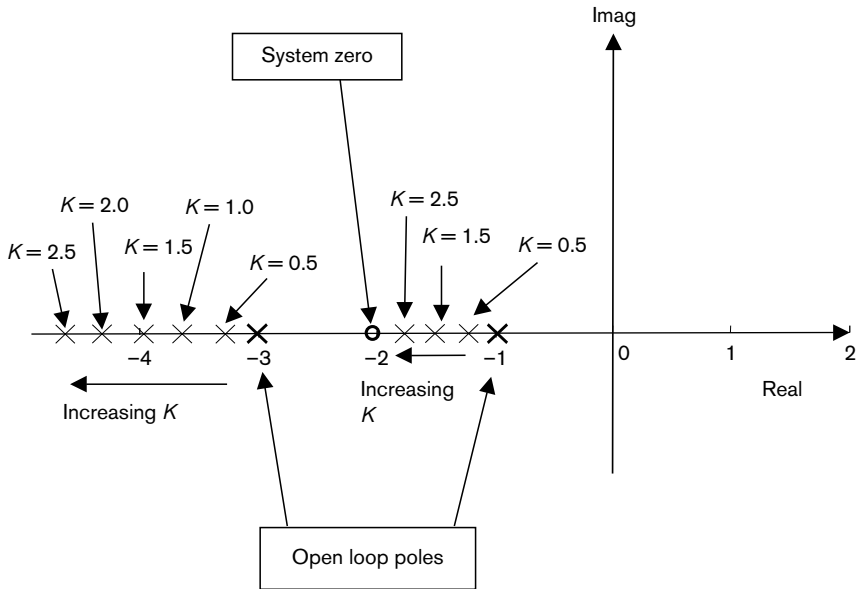


Figure 13.4 Sketch of closed-loop pole positions.

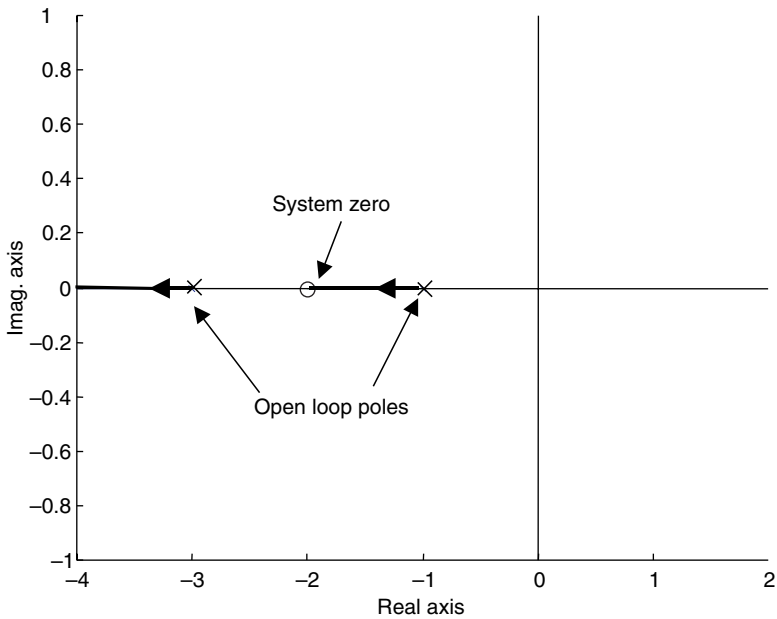


Figure 13.5 MATLAB root locus plot of $(s + 2)/(s^2 + 4s + 3)$.

```
s=tf('s');
g=(s+2)/(s^2+4*s+3);
rlocus(g);
```

13.3 Preliminary MATLAB root locus investigations

In the example above we saw how the manual construction of a root locus is time-consuming and calculation-heavy. MATLAB will enable us to generate root locus plots very easily, so we are going to examine the main features of a plot using MATLAB.

Example 1

We consider the system

$$G(s) = \frac{K}{s(s+2)}$$

There are no system zeros for this system, and two open-loop poles, $p_1 = 0$ and $p_2 = -2$. By the pole-zero conservation principle, we have the same number of zeros (finite or at infinity) as poles; therefore we will have two zeros at infinity. We use the following MATLAB code to plot the root locus.

```
s=tf('s');
g = 1/(s*(s+2));
rlocus(g);
```

This produces the plot shown in Figure 13.6.

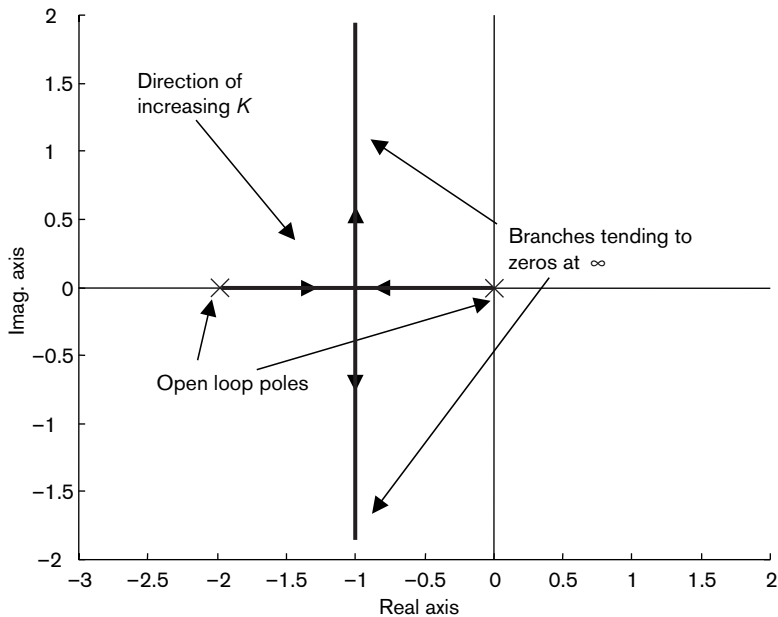


Figure 13.6 Root locus plot of $K/[s(s+2)]$.

The open-loop poles are shown on the root locus diagram; there are no finite system zeros, but we have two zeros at infinity. The root locus plot shows the locus for increasing values of K . These root locus plots start at the open-loop poles and end at the zeros at infinity. In this example, there

were two poles and no system zeros, so we have two zeros at infinity to balance the poles and zeros of the transfer function.

We have two open-loop poles, each with its own locus which ended at a zero; we call each locus a *branch*.

In this example, the branches meet at a point and break away along two asymptotes. The asymptotes are symmetrical with respect to the real axis.

We consider another example to see if the same features occur.

Example 2

We consider the system

$$G(s) = \frac{s-4}{(s+1)(s+2)(s+3)}$$

There is one zero in the RHP at $z = 4$. There are three poles: $p_1 = -1$ and $p_2 = -2$ and $p_3 = -3$. We use the following MATLAB code to plot the root locus.

```
s=tf('s');
g = (s-4)/(s+1)/(s+2)/(s+3);
rlocus(g);
```

This produces the plot shown in Figure 13.7.

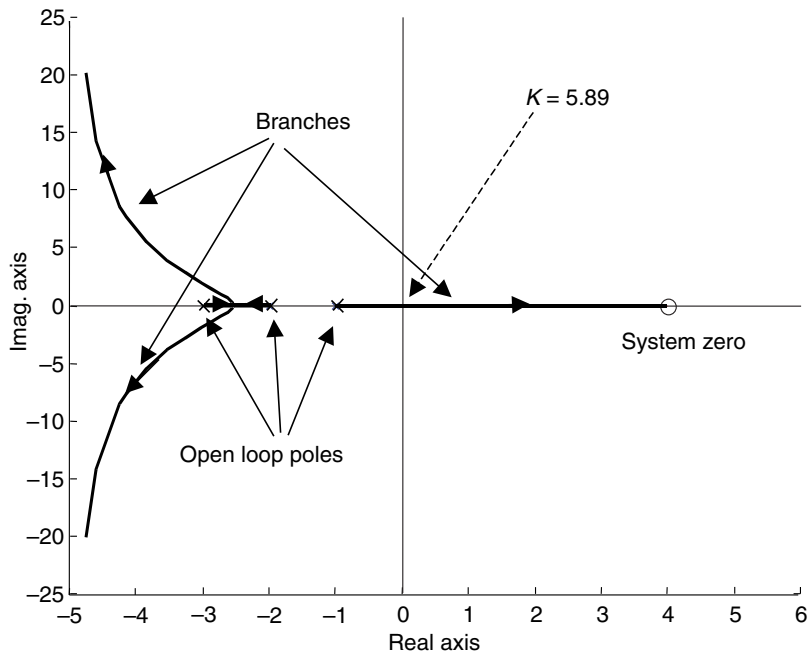


Figure 13.7 Root locus plot of $(s - 4)/[(s + 1)(s + 2)(s + 3)]$.

We note that there is one zero in the RHP. The three open-loop poles give rise to three branches: one branch ends at the zero in the RHP, while the other two meet at a breakaway point and tend to two zeros 'at infinity'. The two branches which are contained wholly in the LHP provide stable

closed-loop poles for all gain K . However, the root locus that ends at the RHP zero tells us that for values of K above a specific value ($K=5.89$), the closed-loop system will have a pole outside the LHP, and therefore the closed-loop system will be unstable. Therefore by plotting the root locus we can determine how large we can make the gain K before we incur instability in the closed-loop system.

Once again we note that the root locus is symmetrical with respect to the real axis.

13.4 Some useful root locus rules

From the previous examples we can formulate a number of useful rules. We do this without going through the proofs, which are not necessary for learning how to use the root locus technique.

Key result: Root locus rules

1. *Number of branches*

The number of branches is equal to the number of closed-loop poles

2. *Symmetry of branches*

The branches are symmetric with respect to the real axis.

3. *Poles of the open-loop system*

The open-loop poles lie on the root locus at $K=0$

4. *Zeros of the open-loop system*

If the system has a number n_z of finite zeros, then the same number of closed-loop poles will go to these zero positions as $K \rightarrow \infty$. The remaining poles will go to the zero positions at infinity.

5. *Locus on real axis*

A segment of the root locus will lie on the real axis only if the number of poles and finite zeros to the right of this segment is an odd number.

6. *Start and end points*

The branches start at the open-loop poles for $K=0$ and end at the open-loop zeros.

7. *Breakaway points*

Breakaway points on the real axis are found by solving the following equation for values of s .

$$\frac{dG_{OL}(s)}{ds} = 0$$

Complex breakaway points must also satisfy this equation, but, in addition, the pole positions must satisfy the closed-loop pole polynomial equation for a real value of gain K .

$$d(s) + Kn(s) = 0$$

8. *Asymptotes of the locus*

If $G(s)$ has n_p poles and n_z system zeros, the root locus are asymptotic to $(n_p - n_z)$ straight lines making angles with the real axis of:

$$\phi = \frac{(1+2k) \times 180^\circ}{n_p - n_z}, \quad k = 0, 1, 2, \dots, (n_p - n_z) - 1$$

Example We consider the servo system shown in Figure 13.8.

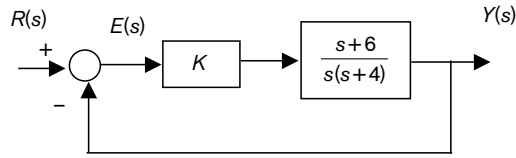


Figure 13.8 Servo system.

The system has an open-loop zero at $s = -6$ and two poles at $s = 0$ and $s = -4$. We use MATLAB to plot the root locus diagram as seen in Figure 13.9, and study the rules as described above.

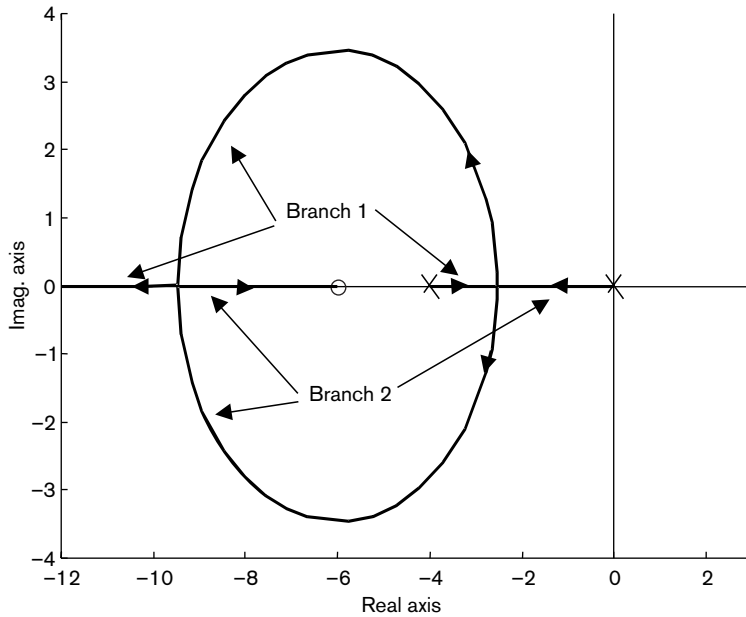


Figure 13.9 Root locus of $K(s + 6)/[s(s + 4)]$.

1. Number of branches

The closed-loop system is given by

$$G_{CL}(s) = G_{CL}(s) = \frac{s+6}{s^2 + 4s + Ks + 6K}$$

The closed-loop poles are found by solving the closed-loop pole polynomial equation

$$C(s) = s^2 + 4s + K(s + 6) = 0$$

This is of second-order and it has two poles. The number of branches is therefore equal to two.

2. Symmetry of branches

The branches are symmetric with respect to the real axis. This symmetry occurs since the poles are either real (on the Real axis) or complex conjugate pairs – one on each side of the real axis.

3. Poles of the open-loop system

The open-loop poles are at $p_1 = 0$ and $p_2 = -4$.

4. Zeros of the open-loop system

There is one system zero at $z_1 = -6$. The pole-zero conservation principle tells us that

$$n_p = n_z + n_{z\infty}$$

Therefore, since $n_p = 2$ and $n_z = 1$, we will have one zero at infinity, $z_2 = \infty$, and only one branch will go to infinity.

5. Locus on real axis

The locus exists only to the right of $z_1 = -6$ and to the right of $p_2 = -4$ since the total number of poles and system zeros is an odd number: $n_p + n_z = 2 + 1 = 3$. The locus does not lie to the right of p_1 , since the number of poles and finite system zeros is an even number.

6. Start and end points

Branch 1 starts at $p_2 = -4$ and ends at $z_2 = \infty$. Branch 2 starts at $p_1 = 0$ and ends at $z_1 = -6$.

7. Breakaway points

We solve the following equation and determine whether we have real or complex breakaway points.

$$\frac{dG_{OL}(s)}{ds} = 0$$

$$\frac{d}{ds} \left(\frac{s+6}{s^2+4s} \right) = \frac{-(s^2+12s+24)}{(s^2+4s)^2} = 0$$

By setting the numerator to zero ($s^2 + 12s + 24 = 0$) we find two real breakaway points at

$$s = -9.5 \text{ and } s = -2.5$$

8. Asymptotes of the locus

$G(s)$ has $n_p = 2$ poles and $n_z = 1$ system zeros. The asymptotes are given by

$$\phi = \frac{(1+2k) \times 180^\circ}{n_p - n_z}, \quad k = 0, 1, 2, \dots, (n_p - n_z) - 1$$

Therefore the root locus are asymptotic to one straight line, making an angle of $\phi = 180^\circ$ for $k = 0$.

13.5 Second-order system performance: root locus contours

We would like to consider what we can find out about the closed-loop system performance from the root locus.

For second-order systems, we have found in Chapter 10 that systems with the same damping ratio have poles that lie on the same radial line from the origin. Likewise, systems with the same value of natural frequency, ω_n , have poles that lie on the same circle centred on the origin. These contours of damping and natural frequency can be overlaid on any s -domain plot. The MATLAB command for this is `sgrid`. Consider the transfer function $G(s)$ given by:

$$G(s) = \frac{1}{(s^2 + 0.5s + 1)(s + 1)}$$

For this example, the MATLAB code would be

```
s = tf('s');
g = 1/((s+1)*(s^2+0.5*s+1));
rlocus(g)
axis equal
sgrid
```

The root locus plot is shown in Figure 13.10.

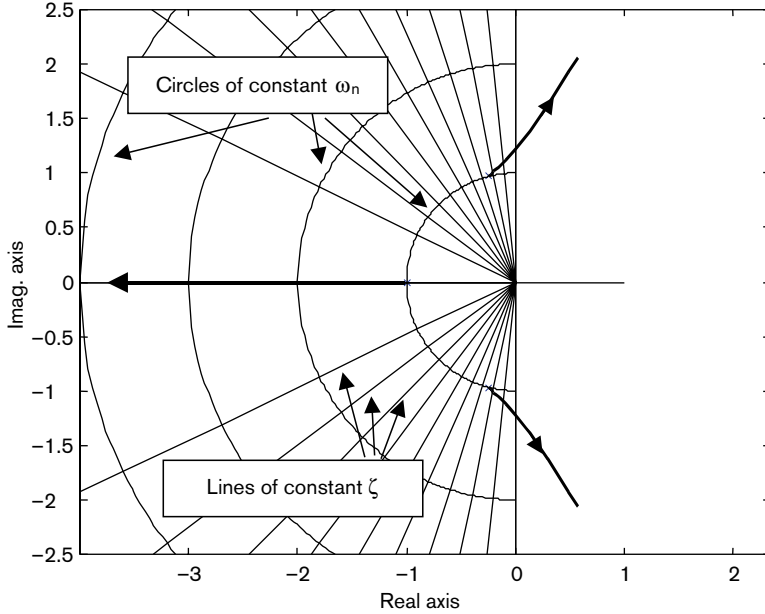


Figure 13.10 Contours of damping and natural frequency.

Example We consider the example of a d.c. motor: Figure 13.11. The transfer function of the motor between the controller input signal $U(s)$ and the output shaft velocity, $\omega(s)$ is given by a simple first-order lag where $\tau = 0.5$ seconds. The position of the shaft is represented by $\theta(s)$. The controller, K , is a proportional controller.

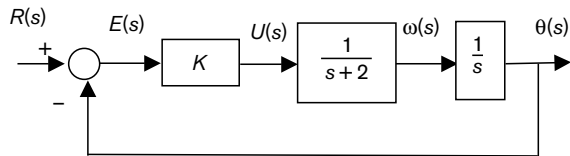


Figure 13.11 D.c. motor.

The open-loop transfer function is given by $G(s) = K/[s(s + 2)]$. The system has two poles at $p = 0$ and $p = -2$ and no open-loop zeros. The root locus plot in Figure 13.12 shows the position of the closed-loop poles as we change K from 0 to ∞ .

The MATLAB Control Toolbox command `rlocus` is used to generate the root locus diagram. By moving the small square cursors along the loci, we are effectively changing the position of the closed-loop poles. The associated change in gain K is shown. The damping and natural frequency are also shown for values of $\zeta < 1$, that is, when the loci do not lie on the real axis. In this example, we note that for $K \leq 1$ the closed-loop poles are real and for $K > 1$ the poles are complex. The

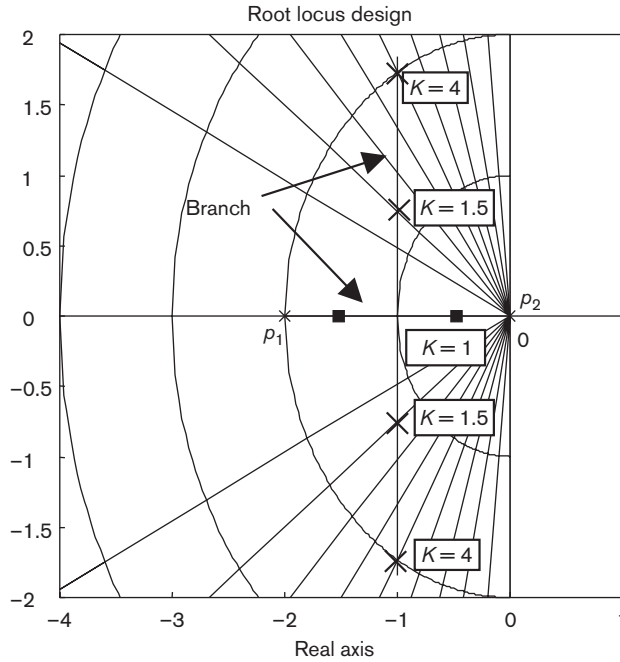


Figure 13.12 Root locus plot for d.c. motor.

values of gain K are also shown at some points on the diagram; we find that we can complete the following table:

Gain	Damping ratio, ζ	Natural frequency, ω_n	Response type
$0 < K < 1$	Not shown but $\zeta > 1$ since both poles lie on real axis		Overdamped
$K = 1$	$\zeta = 1$	$\omega_n = 1$	Critically damped
$K = 1.55$	$\zeta = 0.8$	$\omega_n = 1.25$	Underdamped
$K = 4$	$\zeta = 0.5$	$\omega_n = 2$	Underdamped

In this example, the closed-loop poles approach each other as we increase the gain K . Note that the two poles are real for $0 < K < 1$, and hence the system is over damped. At $K = 1$, the two poles have the same location and the system becomes critically damped. For $K > 1$, the two poles moves toward the two zeros at infinity and the system becomes under damped.

13.6 Effects of adding a pole or a zero to the root locus of a second-order system

We discussed how we could change the value of gain K to change the position of the closed-loop poles. This corresponds to placing a proportional gain, K , in cascade with the system $G(s)$ and finding the closed-loop poles for different values of gain, K . However, proportional control is a simple form of control; it does not provide us with zero steady

state error and it will limit the type of time response we can produce from a system. For example, in some control design problems, to produce the performance required in the design specifications we need to move the poles to some positions on the s -plane, which may not lie on a root locus defined by the simple proportional gain K . To be able to move the poles to any position on the s -plane, we need to use a more complicated controller. For example, we may need to add a zero or a pole to the controller and see how this will affect the root locus and hence the position of the closed-loop poles. Examples of controllers with poles or zeros are:

$$\text{PI control:} \quad K(s) = K_p + \frac{K_i}{s} = \frac{K_p s + K_i}{s}$$

$$\text{Lag controller:} \quad K(s) = \frac{s\tau + 1}{\alpha s\tau + 1} \quad (\tau, \alpha \text{ are controller parameters})$$

Thus, we need to know how the root locus will change if we add a pole or a zero. To investigate this, we will use a simple example.

13.6.1 Effects of adding a zero on the root locus for a second-order system

Consider the second-order system given by

$$G(s) = \frac{1}{(s+p_1)(s+p_2)} \quad p_1 > 0, \quad p_2 > 0$$

The poles are given by $s = -p_1$ and $s = -p_2$ and the simple root locus plot for this system is shown in Figure 13.13(a). When we add a zero at $s = -z_1$ to the controller, the open-loop transfer function will change to:

$$G_1(s) = \frac{K(s+z_1)}{(s+p_1)(s+p_2)}, \quad z_1 > 0$$

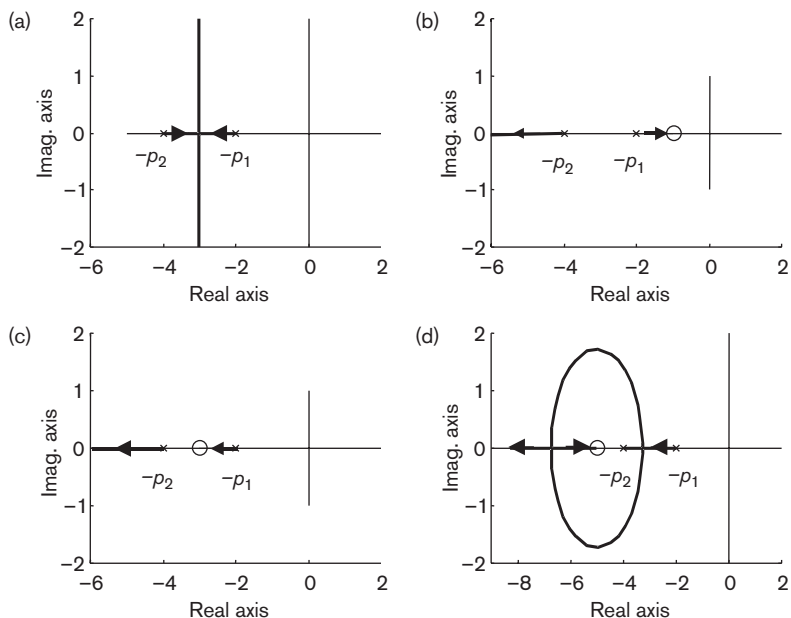


Figure 13.13 Effect of adding a zero to a second-order system root locus.

We can put the zero at three different positions with respect to the poles:

1. To the right of $s = -p_1$ Figure 13.13(b)
2. Between $s = -p_2$ and $s = -p_1$ Figure 13.13(c)
3. To the left of $s = -p_2$ Figure 13.13(d)

We now discuss the effect of changing the gain K on the position of closed-loop poles and type of responses.

- (a) The zero $s = -z_1$ is not present.
For different values of K , the system can have two real poles or a pair of complex conjugate poles. This means that we can choose K for the system to be overdamped, critically damped or underdamped.
- (b) The zero $s = -z_1$ is located to the right of both poles, $s = -p_2$ and $s = -p_1$.
In this case, the system can have only real poles and hence we can only find a value for K to make the system overdamped. Thus the pole-zero configuration is even more restricted than in case (a). Therefore this may not be a good location for our zero, since the time response will become slower.
- (c) The zero $s = -z_1$ is located between $s = -p_2$ and $s = -p_1$.
This case provides a root locus on the real axis. The responses are therefore limited to overdamped responses. It is a slightly better location than (b), since faster responses are possible due to the dominant pole (pole nearest to $j\omega$ axis) lying further from the $j\omega$ axis than the dominant pole in (b).
- (d) The zero $s = -z_1$ is located to the left of $s = -p_2$.
This is the most interesting case. Note that by placing the zero to the left of both poles, the vertical branches of case (a) are bent backward and one end approaches the zero and the other moves to infinity on the real axis. With this configuration, we can now change the damping ratio and the natural frequency (to some extent). The closed-loop pole locations can lie further to the left than $s = -p_2$, which will provide faster time responses. This structure therefore gives a more flexible configuration for control design.

We can see that the resulting closed-loop pole positions are considerably influenced by the position of this zero. Since there is a relationship between the position of closed-loop poles and the system time domain performance, we can therefore modify the behaviour of closed-loop system by introducing appropriate zeros in the controller.

13.6.2 Effects of adding a pole on the root locus for a second-order system

We demonstrate the effect of adding a pole by using the second-order system of the previous example. We add a pole at $s = -p_3$ to the controller. The open-loop transfer function will change to:

$$G_2(s) = \frac{K}{(s+p_1)(s+p_2)(s+p_3)} \quad p_1 > 0, p_2 > 0, p_3 > 0$$

Once again, the root locus of the original system $G(s)$ is shown in Figure 13.14(a). We can put the pole at three different positions with respect to p_1 and p_2 :

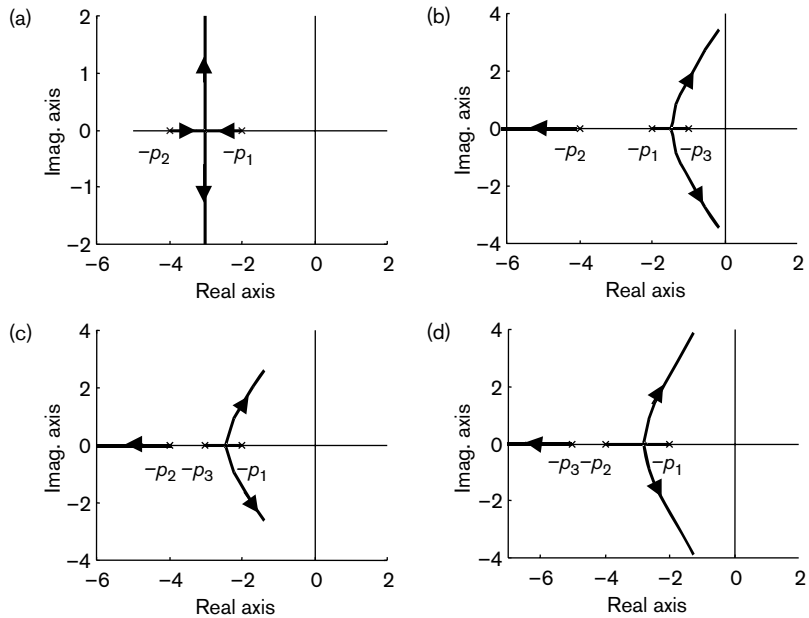


Figure 13.14 The effect on the root locus of adding a pole to a second-order system.

1. To the right of $s = -p_1$, Figure 13.14(b)
2. Between $s = -p_2$ and $s = -p_1$, Figure 13.14(c)
3. To the left of $s = -p_2$ Figure 13.14(d)

We discuss each of the three cases of different pole locations.

- (a) The pole $s = -p_3$ is not present.
For different values of K , the system can have two real poles or a pair of complex conjugate poles. However, for any value of K , the closed-loop pole will remain in the LHP and the closed-loop system will be stable. We can choose K for the system to be overdamped, critically damped or underdamped.
- (b) The pole $s = -p_3$ is located to the right of $s = -p_1$.
Adding a pole to the right of $s = -p_1$ changes the position of the root locus on the horizontal axis. Two branches of the root locus lie on the real axis between the two poles closest to the $j\omega$ axis, and the branches meet at a breakaway point ($s = -1.45$ found from MATLAB) and move towards zeros at infinity. The rule on asymptotes causes the root locus to bend towards the RHP. This has the consequence that at some value of K , the closed-loop system will become unstable. The third branch lies to the left of the leftmost pole and lies on the real axis; the locus moves again towards a zero at infinity. We note that as we increase the gain K , two of the poles become complex, and hence the closed-loop system becomes underdamped.
- (c) The pole $s = -p_3$ is located between $s = -p_2$ and $s = -p_1$.
This case is similar to (b), with the exception that the breakaway point is at $s = -2.4$. The closed-loop complex conjugate poles move towards the RHP as the gain increases; at high gain we will have instability.

(d) The pole $s = -p_3$ is located to the left of $s = -p_2$.

This case is also similar to (b), but the breakaway point has moved slightly to $s = -2.7$.

We can see that adding a pole will slow the closed-loop response, since the root locus has branches that move towards the $j\omega$ axis. This will also reduce the system stability margin.

13.7 Time delays and inverse response systems

Many industrial systems contain time delay effects. For example, we often find that long pipework leads to transport delays in the supply of liquid feedstock. Another example is the computational delay associated with the time taken to perform a control calculation and output the control signal value. In these cases the input $u(t)$ and the output $y(t)$ of a time-delayed system are related by:

$$y(t) = u(t - T_d)$$

where T_d is the dead time or transport delay time. The transfer function of such systems can be determined as:

$$Y(s) = e^{-sT_d} U(s)$$

A time-delayed process may have the transfer function

$$G(s) = e^{-sT_d} G_p(s)$$

Another type of industrial system is the system which exhibits 'inverse response'. This was discussed in Chapter 10 on poles and zeros, where we found that this system is characterised by having a zero in the RHP. Typically this effect is due to the process having competing effects which produce the type of response shown in Figure 13.15.

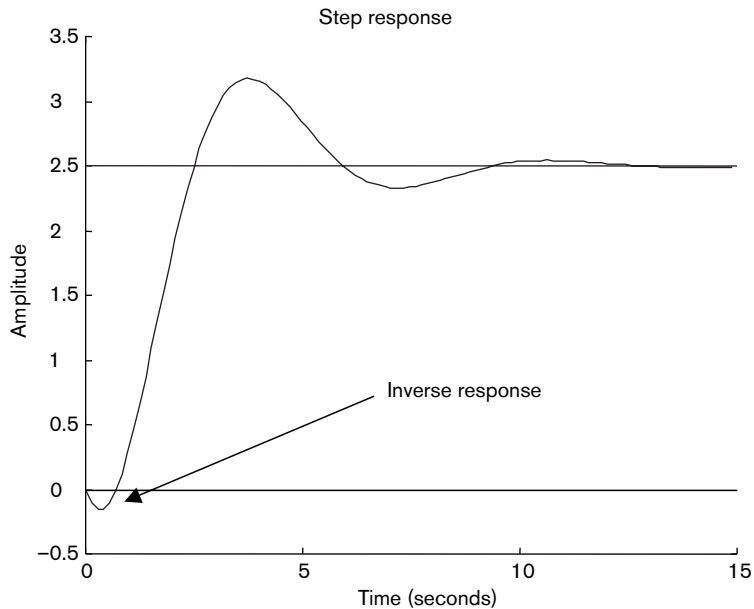


Figure 13.15 Inverse response from system.

13.7.1 Why systems with RHP zeros are called non-minimum phase

Consider a system with a left half plane zero:

$$G_L(s) = \frac{s+1}{(s+2)(s+3)}$$

The system will have its phase given by

$$\angle G_L(j\omega) = \phi(\omega)$$

Now consider a modification of $G_L(s)$ which has its zero in the right half plane:

$$G_R(s) = \frac{s-1}{(s+2)(s+3)}$$

We can write the RHP system as

$$G_R(s) = \left(\frac{s-1}{s+1}\right) \left(\frac{s+1}{(s+2)(s+3)}\right) = G_m(s)G_L(s)$$

where $G_L(s)$ has all its zeros in the LHP and $G_m(s) = (s-1)/(s+1)$ contains the zero in the RHP. If we look at the frequency properties by setting $s = j\omega$, we find:

$$G_R(j\omega) = G_m(j\omega) G_L(j\omega)$$

where

$$G_m(j\omega) = \frac{j\omega-1}{j\omega+1}$$

We find the two properties, magnitude and phase, for $G_R(j\omega)$:

$$|G_R(j\omega)| = |G_m(j\omega)G_L(j\omega)| = |G_m(j\omega)| |G_L(j\omega)| = |G_L(j\omega)|$$

since the magnitude of $|G_m(j\omega)| = 1$,

$$\begin{aligned} \angle G_R(j\omega) &= \angle G_m(j\omega)G_L(j\omega) = \angle G_m(j\omega) + \angle G_L(j\omega) \\ &= -2\tan^{-1}\omega + \phi(\omega) \end{aligned}$$

because $\angle G_m(j\omega) = -\tan^{-1}\omega - \tan^{-1}\omega = -2\tan^{-1}\omega$.

Clearly the gain and phase properties of the system with the RHP zero, $G_R(s)$, can be related to those of the system with the LHP zero. At each frequency, ω ,

$$|G_R(j\omega)| = |G_L(j\omega)|$$

and

$$\angle G_R(j\omega) = -2\tan^{-1}\omega + \phi(\omega)$$

Therefore it is only the phase properties which differ. The minimum phase lag $\phi(\omega)$ occurs for the LHP zero transfer function, which is termed minimum phase. Relative to this, the transfer function with the RHP zero has extra phase lag and is therefore termed *non*-minimum phase.

13.7.2 Transfer functions for time delay and inverse response systems

We found that the inverse response system always had a zero in the RHP:

$$G(s) = (s - z)G_p(s)$$

The time delay system can be modelled by the transfer function

$$G(s) = e^{-sT_d} G_p(s)$$

where we can replace the complex exponential by a Padé approximation:

$$e^{-sT_d} \cong \frac{1 - s(T_d/2)}{1 + s(T_d/2)}$$

However, this gives $G(s)$ a zero in the RHP. We find that due to the presence of this RHP zero, the root locus plots for the two types of industrial system (time delay and inverse response) look similar.

Example Consider a simple unity feedback system which has a time delay, T_d , of 0.3 seconds:

$$G(s) = \frac{Ke^{-0.3s}}{s+1}$$

We replace $e^{-T_d s}$ by its Padé approximation. Note that we can use the following MATLAB commands to produce a first-order approximation

```
order=1;
delay = 0.3;
[num,den] = pade(delay,order);
```

For $T_d = 0.3$, this results in

$$e^{-T_d s} = \frac{-s+6.67}{s+6.67}$$

and we can approximate $G(s)$ by:

$$G(s) \cong \frac{K(-s+6.67)}{(s+6.67)(s+1)}$$

We can see that this time-delay system has a zero at $z_1 = 6.67$ which is on the RHP and thus the system is non-minimum phase. The system has the root locus diagram shown in Figure 13.16. The root locus shows that time-delay systems can be easily made unstable by increasing the value of gain K , since at least one branch of the root locus (in this case both branches) will end at the zero in the RHP.

We have shown that time-delay systems and non-minimum phase systems will often present problems in control design due to the RHP zero that appears in the numerator polynomial of the transfer function. We must be careful in those circumstances not to increase the gain to cause the closed-loop poles to move to the RHP.

13.8 Parameter root locus

When we use a root locus plot, we take an open-loop transfer function $G(s)$ given by

$$G(s) = K \frac{n(s)}{d(s)}$$

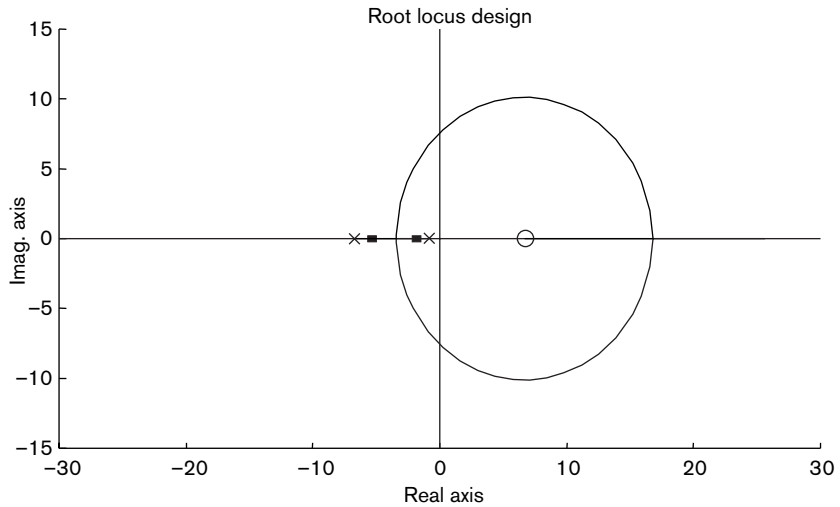


Figure 13.16 Root locus of a time-delay system.

and examine how the poles of the closed-loop system change for variations in parameter K . We are effectively examining the roots of the following equation for different values of K :

$$d(s) + Kn(s) = 0$$

If we can pose a problem in the same format, that is, 'Given two transfer functions $n(s)$ and $d(s)$, how do the roots alter for changes in parameter K ?', we can exploit the root locus theory to our benefit. One clever use of this is illustrated as follows. Let $G(s)$ be given by

$$G(s) = \frac{s+6}{s^3+4s^2+2\alpha s+1}$$

where α may be a system component or parameter. The closed-loop poles are given by the roots of

$$s^3 + 4s^2 + 2\alpha s + 1 = 0$$

Suppose now that we would like to find the closed-loop poles for a range of values of α . To do this, we can rewrite the polynomial in the form:

$$(s^3 + 4s^2 + 1) + \alpha(2s) = 0$$

which is now in the same form as $d(s) + Kn(s) = 0$, where

$$d(s) = s^3 + 4s^2 + 1$$

$$n(s) = 2s$$

and α takes the role of K .

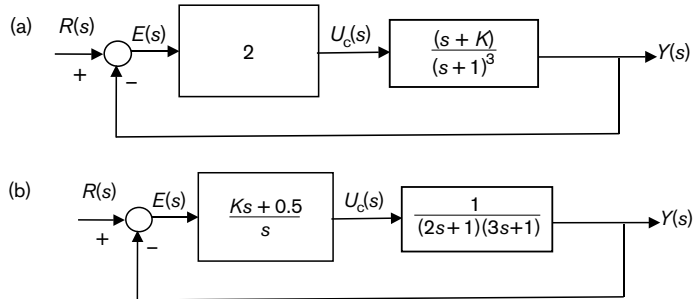
If we enter the fictitious transfer function into MATLAB:

$$G_{\text{fict}}(s) = \frac{2s}{s^3+4s^2+1} = \frac{n(s)}{d(s)}$$

we can use the root locus plot to find how the closed-loop poles vary for different values of $\alpha (= K)$. We do not use the fictitious transfer function for our closed-loop responses; it is merely used to formulate the problem of examining how the closed-loop poles vary for changes in α .

Skill section Parameter root locus: what transfer function should we enter?

Problem For the following systems, what transfer function should you enter to investigate changes in parameter K ?



Solution (a) Find the closed-loop transfer function, $G_{CL}(s)$.

$$Y(s) = \frac{2(s+K)}{(s+1)^3 + 2(s+K)} R(s) = \frac{2(s+K)}{s^3 + 3s^2 + 5s + 1 + 2K} R(s)$$

Closed-loop pole polynomial given by:

$$C(s) = s^3 + 3s^2 + 5s + 1 + 2K = 0 = d(s) + Kn(s)$$

Assign denominator and numerator of fictitious transfer function:

$$d(s) = s^3 + 3s^2 + 5s + 1$$

$$n(s) = 2$$

$$K = K$$

Fictitious transfer function to be entered is:

$$G_{fict}(s) = \frac{2}{s^3 + 3s^2 + 5s + 1}$$

(b) Find the closed-loop transfer function, $G_{CL}(s)$.

$$Y(s) = \frac{Ks+0.5}{s(2s+1)(3s+1) + Ks + 0.5} R(s) = \frac{Ks+0.5}{6s^3 + 5s^2 + s + 0.5 + Ks} R(s)$$

Closed-loop pole polynomial given by:

$$C(s) = 6s^3 + 5s^2 + s + 0.5 + Ks = 0 = d(s) + Kn(s)$$

Assign denominator and numerator of fictitious transfer function:

$$d(s) = 6s^3 + 5s^2 + s + 0.5$$

$$n(s) = s$$

$$K = K$$

Fictitious transfer function to be entered is:

$$G_{\text{fict}}(s) = \frac{s}{6s^3 + 5s^2 + s + 0.5}$$

Problem An engineer is designing a d.c. servo system and the motors available cover a range of time constants, τ . The open-loop transfer function description is given by

$$G(s) = \frac{10}{s(\tau s + 1)}$$

The engineer would like to find the range of values of the time constant τ for which the closed-loop system is stable and the response is underdamped.

Solution Assuming $G(s)$ is under unity feedback with the gain $K = 1$, we find the closed-loop transfer function:

$$G_{\text{CL}}(s) = \frac{\frac{1}{s(\tau s + 1)}}{1 + \frac{1}{s(\tau s + 1)}} = \frac{1}{\tau s^2 + s + 1}$$

The closed-loop poles are given by the roots of the denominator of $G_{\text{CL}}(s)$:

$$\tau s^2 + s + 1 = 0 = s^2 + \frac{1}{\tau}(s + 1) = 0$$

If we write this as $d(s) + Kn(s) = 0$, we would find

$$d(s) = s^2$$

$$n(s) = s + 1$$

and

$$K = 1/\tau$$

This gives $G_{\text{fict}}(s) = (s + 1)/s^2$.

We can now plot the root locus for this system to study the effect of $K = 1/\tau$ on the closed-loop performance. The root locus of the system is shown in Figure 13.17.

We can see that as we increase the value of K the closed-loop poles move from the poles at the origin to a breakaway point to the left of the zero. Thereafter, one branch approaches the zero at $s = -1$ and the other approaches the zero at infinity. We examine the system behaviour for different ranges of K .

Range of K	Range of τ	Response type
$0 < K < 4$	$\infty > \tau > 0.25$	underdamped
$K = 4$	$\tau = 0.25$	critically damped
$4 < K < \infty$	$0.25 > \tau > 0$	overdamped

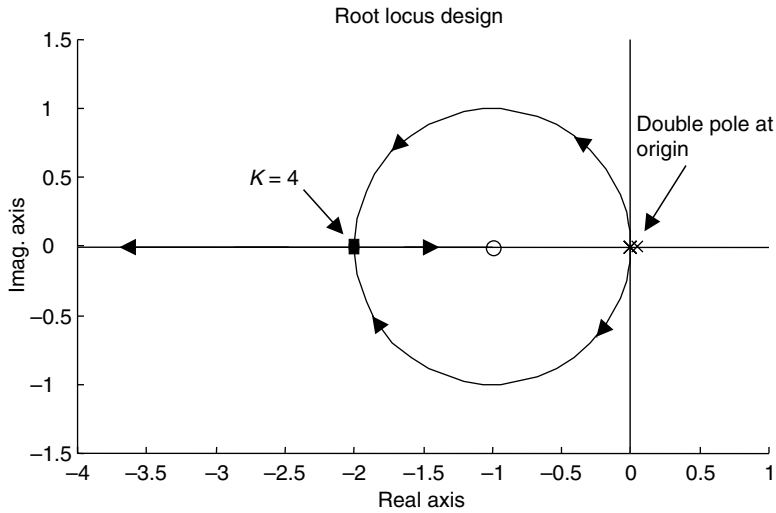


Figure 13.17 Parameter root locus.

13.8.1 How a parameter root locus might arise in a control example

In designing the d.c. motor system, as shown in Figure 13.18, we might be very interested in how the performance depends on motor selection (choice of K_m , τ_m) or the tachogenerator feedback gain (choice of K_t). The parameter root locus can help us investigate these problems.

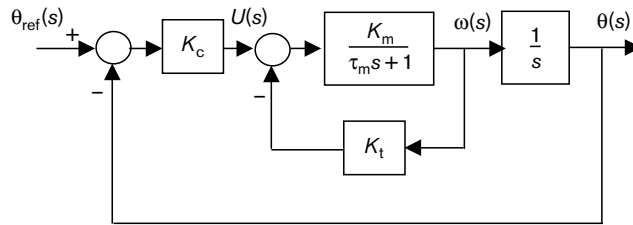


Figure 13.18 Block diagram for d.c. motor.

We first need to find the closed-loop transfer function in terms of the design parameters K_c , K_m , τ_m and K_t . We do this in two steps:

Inner loop transfer function $\omega(s) = \frac{K_m}{\tau_m s + 1 + K_m K_t} U(s)$

Open-loop transfer function $G_{OL}(s) = \frac{K_c K_m}{s(\tau_m s + 1 + K_m K_t)}$

Closed-loop transfer function $\theta(s) = \frac{G_{OL}(s)}{1 + G_{OL}(s)} \theta_{ref}(s)$
 $= \frac{K_c K_m}{\tau_m s^2 + (1 + K_m K_t)s + K_c K_m} \theta_{ref}(s)$

The closed-loop poles of the d.c. motor system are found by solving

$$C(s) = \tau_m s^2 + (1 + K_m K_t)s + K_c K_m = 0$$

Now the parameter root locus can be used. For example, suppose we had $K_c = 0.95$ and motor parameters $K_m = 10$ with $\tau_m = 0.25$. If we wanted to investigate closed-loop performance for a range of tachogenerator gains $K_t > 0$, we use the following:

$$\begin{aligned} C(s) &= 0.25s^2 + (1 + 10K_t)s + 9.5 = 0 \\ &= (0.25s^2 + s + 9.5) + K_t(10s) = 0 \end{aligned}$$

For the parameter root locus we identify

$$d(s) = (0.25s^2 + s + 9.5)$$

$$n(s) = 10s$$

and K_t takes the role of K .

The fictitious plant needed is

$$G_{\text{fict}}(s) = \frac{10s}{0.25s^2 + s + 9.5}$$

and a root locus plot will tell us about the expected closed-loop and dynamic performance that we can expect as we change the value of $K (= K_t)$.

13.9 Using MATLAB `r1tool` and `rlocus` routines

M `rlocus`

`rlocus(g)` computes and plots the root locus of the MATLAB system, $g (= G(s))$, for $K = 0$ to ∞ . We note that the information given by the `help` function will show a different feedback loop, where gain K is in the feedback path. Although the closed-loop transfer functions for the two systems, Figure 13.19(a) and (b) will be different:

(a) $G_{\text{CL}}(s) = \frac{KG(s)}{1+KG(s)}$

(b) $G_{\text{CL}}(s) = \frac{G(s)}{1+KG(s)}$

the closed-loop denominator polynomial will be the same. In both cases, we enter both K and $G(s)$ in MATLAB to form the open-loop transfer function, $G_{\text{OL}}(s) = KG(s)$. The root locus is then given by

`rlocus(gol)`

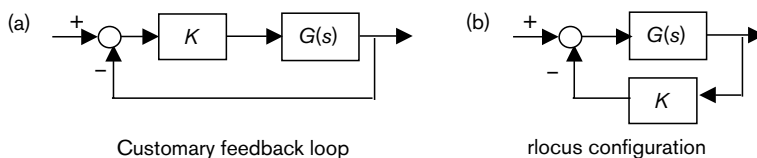


Figure 13.19 `rlocus` configuration.

Mrtool

`rltool` is an interactive MATLAB programme, which we can use to design a controller (compensator as it is called in MATLAB) using root locus techniques.

If we run `rltool`, the window in Figure 13.20 pops up.

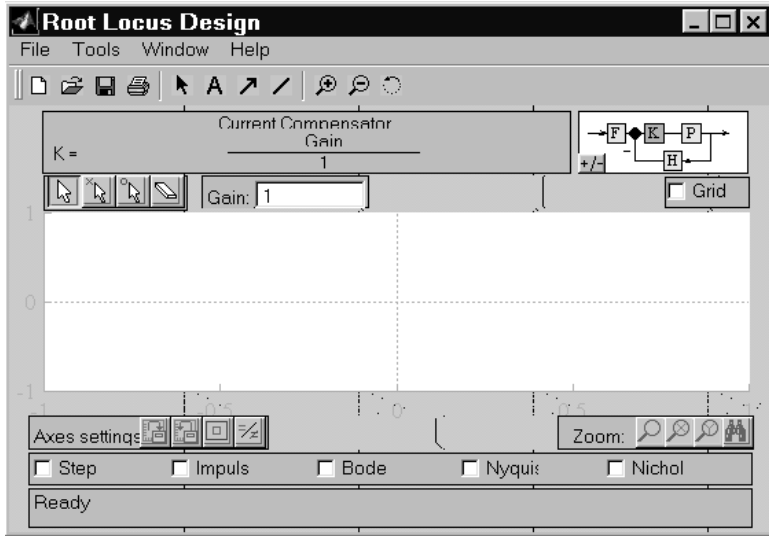


Figure 13.20 `rltool` window.

Note that the feedback system is more general than the one we discussed. It refers to $P(s)$ as the process and $K(s)$ as the controller. It has the system $H(s)$ in the feedback loop and $F(s)$ in the forward path. $H(s)$ and $F(s)$ are initially set to 1 by default and there is no need to change these values if we use the standard system we discussed.

The plant model $G(s)$ can be first entered into MATLAB workspace in the main MATLAB window. This model can then be imported into the programme by selecting the Import Model item from the File menu.

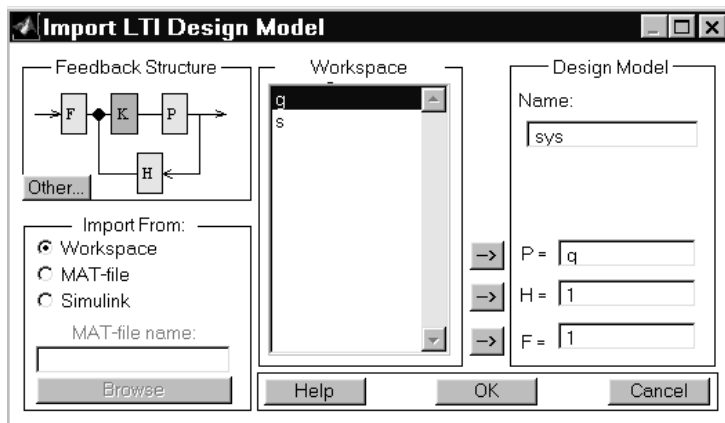


Figure 13.21 `rltool`: import compensator window.

By highlighting the plant model, g , and clicking on the arrow on the left-hand side of P (Figure 13.21) we can enter the plant model. Press OK to complete model import. The root locus diagram will then be automatically plotted.

Example Enter the following MATLAB commands:

```
s=tf('s');
g=(s+4)/(s*(s+1)*(s+6));
rltool
```

and import the model to obtain the root locus diagram (Figure 13.22).

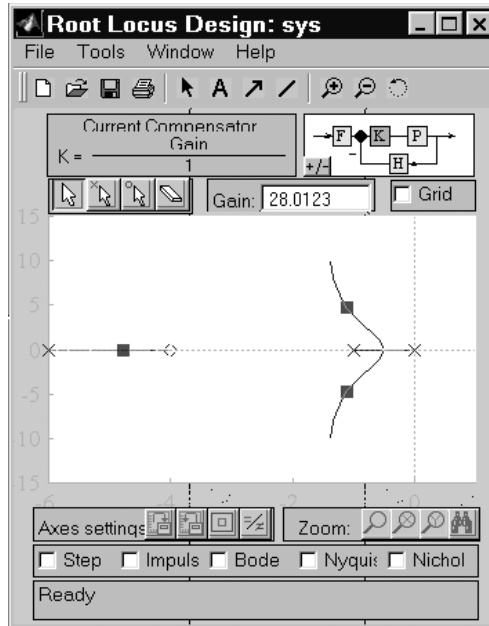


Figure 13.22 Root locus plot for $G(s) = (s + 4)/s(s + 1)(s + 6)$.

The open-loop poles are shown by crosses and the zeros by circles. The squares show the positions of the closed-loop poles, and the corresponding gain $K = 28.01$ can be read in the gain window. By clicking and holding on the squares we can move them along the root locus to follow the positions of the closed-loop poles. The gain K changes as the pole positions change and can be read in the gain window.

The Step, Impulse, Bode, Nyquist or Nichols plots can be plotted by clicking in the square boxes on their left-hand side.

Poles and/or zeros can be added by double-clicking on the Compensator window to bring up the data entry window:



We can then add poles or zeros by clicking on the appropriate button and entering the values of poles or zeros. The poles and zeros can also be entered directly onto the root locus plot: clicking on the small white square with a cross for zero or with a circle for a pole, and then click on the desired position on the s -plane. For example, we may wish to modify the structure of the controller to include additional poles and zeros. We could then use the `r1tool` design tool to explore the effect on the root locus diagram, and hence on the closed-loop system performance and stability, of adding these extra poles and zeros.

What we have learnt

- ✓ To plot the root locus for simple systems.
- ✓ To understand simple features of a root locus plot, start and end points, the branches and asymptotes.
- ✓ To relate systems and controllers to stability and performance properties using a root locus plot.
- ✓ To understand the special effects of RHP zeros on a root locus plot.
- ✓ That adding poles and zeros to the compensator changes the root locus plot and can cause the closed-loop poles to move into the RHP.
- ✓ That adding poles and zeros can move the closed-loop poles to positions on the s -domain with desirable closed-loop performance.
- ✓ To link the effect of time delays and inverse response systems on the root locus plot behaviour.
- ✓ To use the root locus plots for other system parameters, a method known as the parameter root locus.
- ✓ To use the MATLAB `r1tool` for root locus investigations.

Multiple choice

M13.1 A root locus is used for:

- (a) control design
- (b) simulation
- (c) modelling
- (d) (b) and (c)

M13.2 Which statement is correct?

- (a) a branch starts at a pole and ends at a zero
- (b) a branch starts at a zero and ends at a pole
- (c) a branch starts at the origin and ends at a pole
- (d) a branch starts at a zero and ends at the origin

M13.3 A system of order 3 has a root locus with:

- (a) 1 branch
- (b) 2 branches
- (c) 3 branches
- (d) 4 branches

M13.4 A transfer function has a second-order denominator and constant gain as the numerator.

- (a) the system has one zero at infinity
- (b) the system has two finite zeros
- (c) the system has two zeros at the origin
- (d) the system has two zeros at infinity

M13.5 For a system with three poles and two finite zeros:

- (a) 1 branch goes to infinity
- (b) 2 branches go to infinity
- (c) 3 branches go to infinity
- (d) 4 branches go to infinity

M13.6 To start the root locus design tool in MATLAB, we use:

- (a) rtttool
- (b) rlttool
- (c) rstool
- (d) rtool

M13.7 A feedback system has $G(s)$ in the forward path and $H(s)$ in the feedback path. To draw the root locus, we use:

- (a) rlocus(G)
- (b) rlocus(H)
- (c) rlocus(G*H)
- (d) rlocus(G/H)

M13.8 The root locus for the unity feedback system $G(s) = 5$ is:

- (a) a horizontal straight line
- (b) a vertical straight line
- (c) a line passing through the origin
- (d) does not have a root locus

M13.9 The root locus for $G(s)=1/s^2$ is:

- (a) a horizontal straight line on the real axis
- (b) a vertical straight line passing through $(-1,0)$
- (c) a vertical straight line passing through $(1,0)$
- (d) a vertical straight line passing through $(0,0)$

M13.10

A system has two real poles and one real zero where $p_1 < z_1 < p_2 < 0$.

- (a) a branch of the root locus lies on the real axis between z_1 and p_2
- (b) a branch of the root locus lies on the real axis between z_1 and p_1
- (c) a branch of the root locus lies on the real axis between p_2 and ∞
- (d) has no branch on the real axis

Questions: practical skills

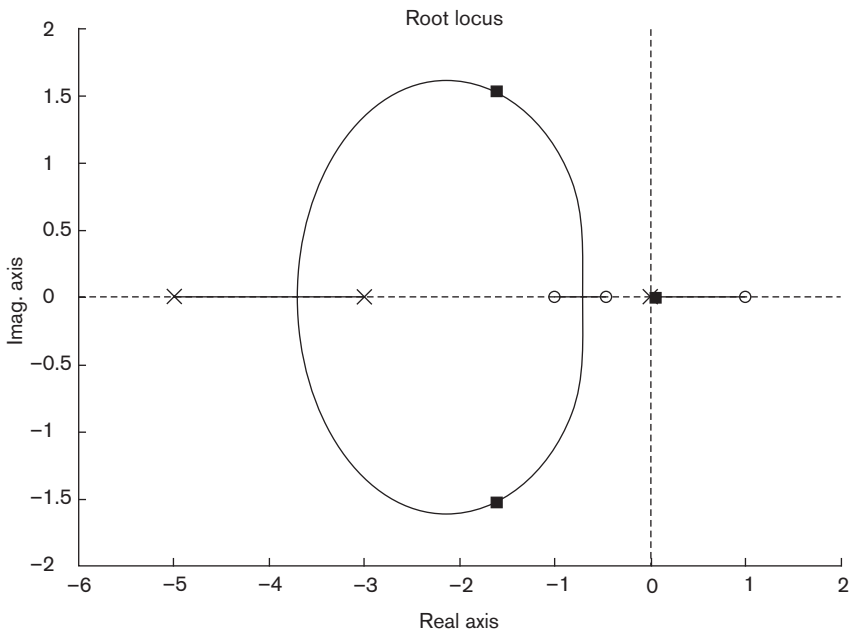
Q13.1 What are the poles and zeros (both finite and infinite) of the following systems?

(a) $G_1(s) = \frac{s+3}{s^2+4s+2}$

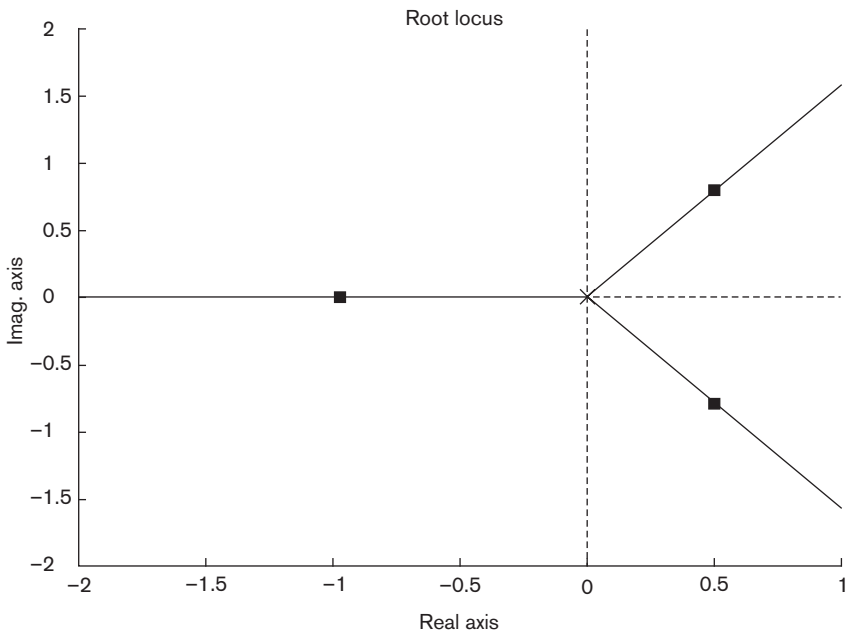
(b) $G_2(s) = \frac{20}{(s+4)(s+5)}$

(c) $G_3(s) = \frac{2}{s(3s+1)(5s+1)}$

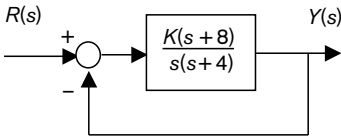
Q13.2 Find the open-loop transfer function for the root locus given below.



Q13.3 Find the system which has the following root locus:



Q13.4 Consider the system shown:



- (a) How many branches does the root locus of the open-loop system have?
 (b) Is there any symmetry in the plot?
 (c) Where are the open-loop poles and zeros on the root locus?
 (d) Where does the root locus start and where does it end?
 (e) Where is the locus on the real axis?
- Q13.5** Sketch the root locus for the unity feedback system, $G(s) = K/s$. For what value of K does the closed-loop system have a time constant of 10 seconds.
- Q13.6** What transfer function do we have to use to plot the parameter (α) root locus for the following open loop transfer functions in a unity feedback system configuration.
- (a) $G_1(s) = \frac{20}{4s^2 + 3\alpha s + 1}$
 (b) $G_2(s) = \frac{\alpha^2}{s^2 + s + \alpha^2}$

Problems

P13.1 Consider the unity feedback system where $G_{OL}(s) = K/s^2$.

- (a) Plot the root locus.
 (b) Add a zero to the system to make it stable.
 (c) Find the value of K for the system to be critically damped.

P13.2 Use MATLAB to draw the root locus of the unity feedback system with the open-loop transfer function

$$G(s) = \frac{K}{s(s+2)(s+4)}$$

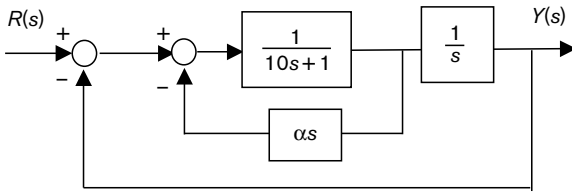
- (a) What is the greatest value of K before the system becomes purely oscillatory?
 (b) Determine the frequency of oscillation in (a).
 (c) Determine the value of K such that the dominant pair of complex poles of the system has a damping ratio of 0.5.

P13.3 A unity feedback system has an open-loop transfer function

$$G(s) = \frac{K}{s^2(s+4)}$$

- (a) Using the root locus, show that the system is unstable for all values of K .
 (b) Add a zero at $s = -a$, $0 < a < 2$, and show that this stabilises the system.
 (c) Investigate the effect of changing K on the closed-loop damping for the dominant pole pair.

P13.4 The block diagram of a control system is given:



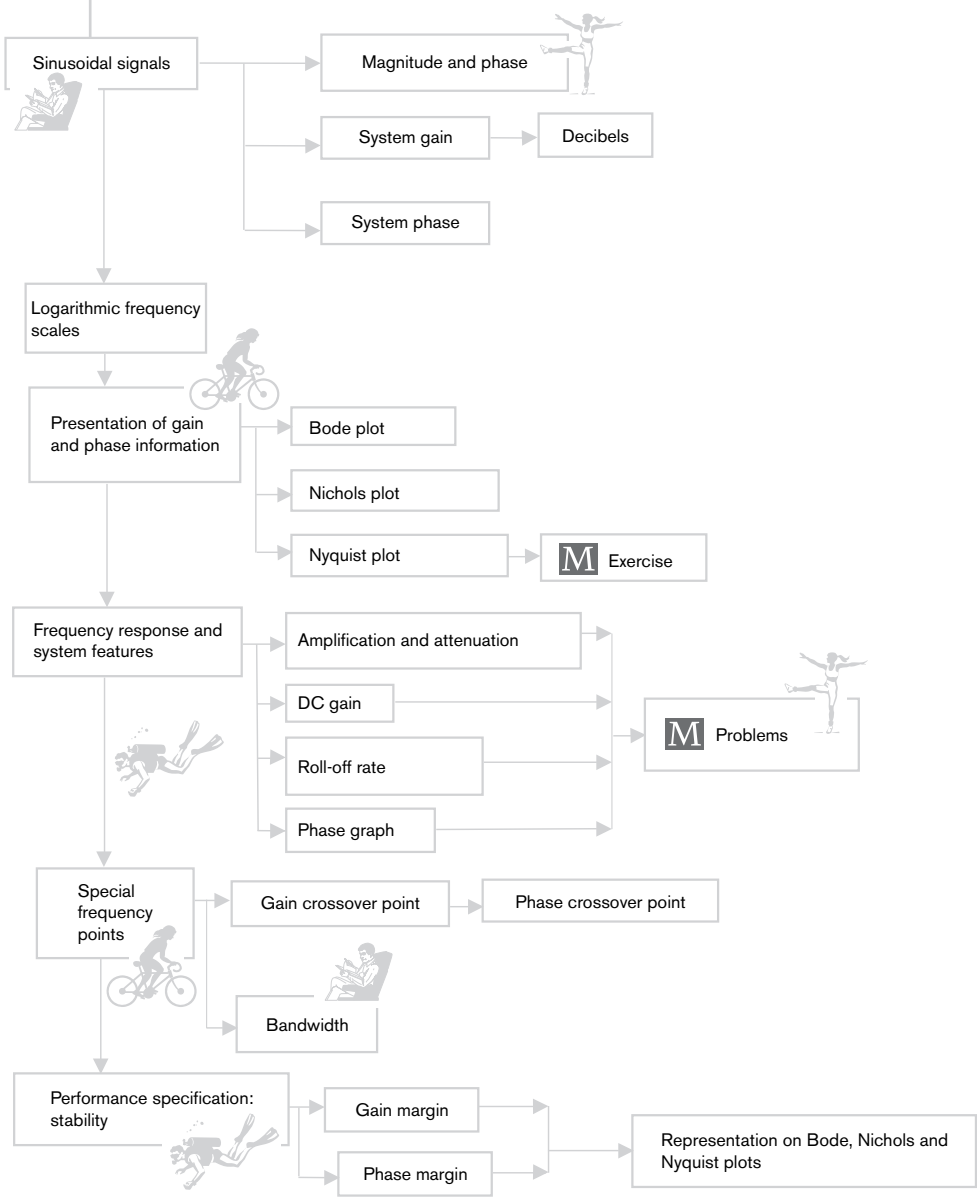
- (a) Determine the closed-loop transfer function from $R(s)$ to $Y(s)$ by working out the inner loop transfer function first.
- (b) Draw the root locus of the system with α as a varying parameter.
- (c) Discuss the effect of the derivative feedback on the closed-loop damping.
- (d) Determine the value of α for the system to be critically damped.

P13.5 Consider a unity feedback system which has the open-loop transfer function

$$G(s) = \frac{s+6}{s(s+5)(s+10)(s+15)}$$

Design a feedback controller using the root locus method for the closed-loop system to have a damping greater than 0.5 and a damped natural frequency of greater than 5.00 rad s^{-1} .

14 The frequency domain



The link between time functions and frequency functions

We are familiar with systems that are dynamic, that is, systems whose output signals vary with time. Examples are shown in Figure 14.1.

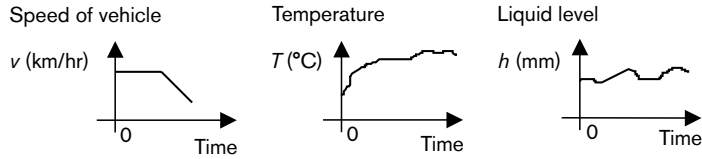


Figure 14.1 System output signals varying with time.

However, difficulties arise in interpreting a frequency response plot because graphs of functions in the *frequency* domain are not common in everyday life or in non-engineering situations. We are therefore not practised at understanding the link between *time* domain systems, such as heating systems which alter room temperature on a daily (time) basis, and their equivalent frequency response plot. A key point underlying the link between time and frequency is the idea that a time-varying signal may be resolved into a sum of sinusoidal functions (cosine or sine waves) of different frequencies (Fourier analysis).

Example: Link between time domain signal and sinusoids of differing frequencies

We can illustrate the link between time domain signals and frequencies by examining how a square wave can be approximated by the sinusoidal functions of Figure 14.2

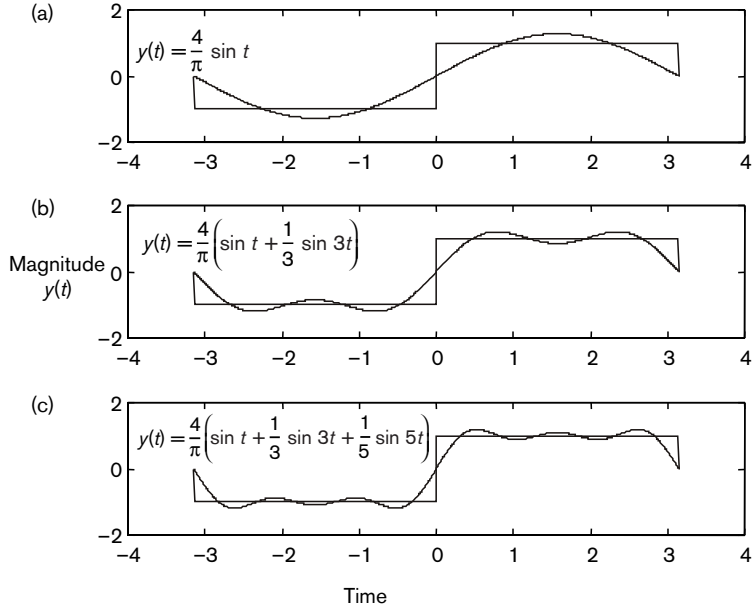


Figure 14.2 Approximating a square wave.

We can see that in plot (a) we have used one sinusoid to approximate the square wave. As we include a combination of sinusoids of differing magnitudes (plots (b) and (c)) we form a closer approximation to the square wave. Hence we can consider a square wave as being composed of

an infinite number of sinusoids. The question that arises now is how this relates to signals from everyday systems, such as house heating systems.

Example: Heating system

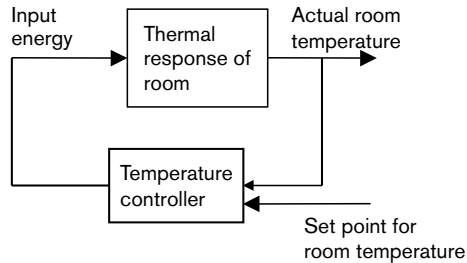


Figure 14.3 Heating system for room.

Consider a heating system (Figure 14.3) where the desired room temperature is 20 °C. The temperature controller alters the input energy to the system to try to maintain the temperature at the desired temperature. The input energy signal will vary depending on the actual temperature measured, and the room temperature will change depending on, for example, the outside temperature or the number of times the doors and windows are opened. Graphs of the variation in both the input energy signals and in the room temperature are shown in Figure 14.4.

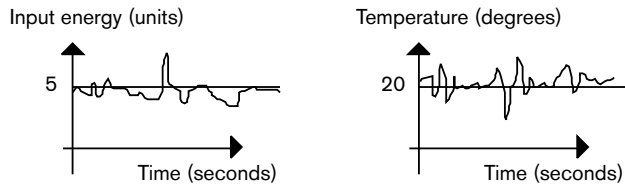


Figure 14.4 Heating system input and output signals.

It can be shown that these signals (which are functions of time) can be decomposed into a number of sinusoidal signals of differing frequencies with varying magnitudes and phases. (This can be done using *Fourier* analysis.) The varying temperature or energy signals (function of time) can be represented by graphs showing the magnitudes or *strengths* of the different frequency components. Hence, both the input energy signals and the output temperature signals can be expressed as functions of frequency, with varying magnitudes and phases.

By applying sinusoidal signals of different frequencies to a system and monitoring the change in amplitude and phase at the system output, we can determine the frequency domain information about the system.

Learning objectives

- To understand the meaning of gain and phase applied to a system.
- To read a frequency scale.
- To recognise that there are different ways of presenting gain and phase information.
- To recognise features on a frequency response plot.

- To find special frequency points on a frequency response plot.
- To interpret a frequency response plot.

We will take the following route through the basic analysis of frequency response plots.

Step 1: Identify magnitude and phase values of a sinusoid

Step 2: Appreciate gain and phase information of a system

Step 3: Familiarise ourselves with the frequency axis

Step 4: Present gain/phase/frequency information on frequency response plots: Bode plots, Nichols plot and Nyquist plots

Step 5: Identify features on the frequency response plot: amplification, attenuation, roll-off rate, shape of phase plots

Step 6: Identify points of special significance: gain crossover, phase crossover, bandwidth

Step 7: Evaluate some frequency domain specifications that can be used for control design

14.1 Identification of magnitude and phase values from a sinusoidal signal

Skill section

The standard form of a sinusoidal signal is

$$u(t) = A \sin(\omega t + \phi)$$

where ω represents the frequency in rad/s, A the amplitude (or strength) of the signal and ϕ the phase shift.

Example Let the following three signals be the main components of a time varying signal.

(a) $u_1(t) = 6 \sin(4t)$

(b) $u_2(t) = 2 \sin(6t - \pi/4)$

(c) $u_3(t) = 1.2 \sin(10t - \pi/2)$

What are the frequencies, magnitudes and phases of the signals? Assume the units of frequency are rad/s, A is in amplitude units and the phase is in radians.

Solution (a) $\omega = 4$ rad/s, $A = 6$ (units), $\phi = 0$ rad

(b) $\omega = 6$ rad/s, $A = 2$ (units), $\phi = -\pi/4$ rad

(c) $\omega = 10$ rad/s, $A = 1.2$ (units), $\phi = -\pi/2$ rad

14.1.1 Magnitude and phase at different frequencies leads to the frequency response

Consider the signals and system shown in Figure 14.5. Known sinusoidal signals are injected at the input to the process. By varying the input signal frequency ω over a range of frequencies, the magnitude and phase effect of the system on the input signals can be determined for the frequency range tested. This is termed the **frequency response**.

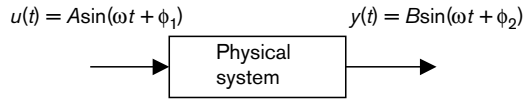


Figure 14.5 Response to a sinusoidal input signal.

By examining the input and output signals and their magnitudes and phases, the effect introduced by the linear system can be inferred. The characteristics of the input and output sinusoids are tabulated:

	Input: $u(t)$	Output: $y(t)$
Frequency	ω rad/s	ω rad/s
Magnitude	A	B
Phase	ϕ_1 rad	ϕ_2 rad

The first important point to note is that the frequency of the input and output sinusoid remains the same. This will be true for *linear* systems. The frequency is usually expressed in rad/s, though sometimes you will see it in Hz.

14.1.2 System gain

The system gain is calculated from the ratio of the amplitudes of the input and output sinusoids.

$$\text{Amplitude of } y(t) = \text{system GAIN} \times \text{amplitude of } u(t)$$

$$\text{System gain} = \frac{\text{Amplitude of } y(t)}{\text{Amplitude of } u(t)} = \frac{B}{A}$$

The physical units of the gain are denoted by :

$$[\text{Gain}] = \frac{[\text{Physical units of output}]}{[\text{Physical units of input}]}$$

For example, if the input signal represented a temperature in $^{\circ}\text{C}$ and the output signal was in volts, the gain would be given in volts/ $^{\circ}\text{C}$.

The gain can also be expressed in terms of **dB (decibels)**:

$$\text{Gain}_{\text{dB}} = 20 \log_{10} \text{Gain}$$

14.1.3 Little gems of information: where do decibels come from?

The units of bels and decibels are not often explained in control engineering textbooks, so that we often use these units without a full appreciation of their origin. Consider the open-loop situation of a system for which we measure the system gain as a *power ratio* of output power, P_o , to input power, P_i :

$$\text{Gain} = \frac{\text{output power}}{\text{input power}} = \frac{P_o}{P_i}$$

To manipulate frequency response plots easily, we measure gain using the logarithmic quantity:

$$\text{Gain} = \log_{10} \left(\frac{P_o}{P_i} \right) \text{Bels}$$

where the power ratio has been evaluated as a logarithm to the base 10 and the units are taken as bels. Introduce now a subdivision of the bel unit such that:

$$1 \text{ Bel} = 10 \text{ decibel} = 10 \text{ dB}$$

and where **dB** is shorthand for **decibel**. This enables the expression for system gain to be written as

$$\text{Gain} = 10 \log_{10} \left(\frac{P_o}{P_i} \right) \text{dB}$$

In practice, the power ratio would not be measured but instead (motivated by an electrical engineering background), we would probably measure the ratio of output voltage to input voltage. Consider now, the power relationships for P_o and P_i across an identical resistance, R , so that

$$P_o = \frac{V_o^2}{R} \text{ and } P_i = \frac{V_i^2}{R}$$

Hence

$$\text{Gain}_{\text{dB}} = 10 \log_{10} \left(\frac{P_o}{P_i} \right) \text{dB}$$

$$= 10 \log_{10} \left(\frac{V_o^2}{V_i^2} \right) \text{dB}$$

$$\text{Gain}_{\text{dB}} = 20 \log_{10} \left(\frac{V_o}{V_i} \right) \text{dB}$$

This analysis explains where the mysterious 20 comes from in dB formulas. Thus to convert a set of output/input readings to dB values we take logarithms to base 10 of the {output/input} ratio and multiply by 20. To reverse this procedure, if G_{dB} is given and we wish to return to absolute gain values, we would use the formula:

$$\text{Gain} = 10^{\text{Gain}(\text{dB})/20}$$

Example A gain of 107 expressed in dB would be

$$\text{Gain}_{\text{dB}} = 20 \log_{10} 107 = 40.59 \text{ dB}$$

A value of 15 dB converts to an absolute gain of

$$\text{Gain} = 10^{15/20} = 5.62$$

It is useful to be conversant with the scale of dB values, for example:

Gain	0.0001 = 10 ⁻⁴	0.1 = 10 ⁻¹	1 = 10 ⁰	10 = 10 ¹	100 = 10 ²	10000 = 10 ⁴
(dB)	-80	-20	0	20	40	80

Very small output/input ratios, *less than 1*, correspond to fairly large *negative* dB values, while a ratio of 1 corresponds to 0 dB and output/input ratios *greater than 1* correspond to *positive* dB values.

Two very common values which should be noted are:

1. If Gain = 1, then because $1 = 10^0$

$$\text{Gain}_{\text{dB}} = 20 \log_{10} 10^0 = 0 \text{ dB}$$

2. If Gain = 0.707, then

$$\text{Gain}_{\text{dB}} = 20 \log_{10} 10^{0.707} = -3 \text{ dB}$$

14.1.4 System phase shift

The phase of the output signal, ϕ_2 , is effectively the phase of the input signal, ϕ_1 , plus any phase effect the system has on the input signal:

$$\phi_2 = \phi_1 + \text{Phase shift}$$

Hence the system **phase shift** is calculated as the phase difference between the output and the input sinusoids.

$$\text{Phase shift} = \phi_2 - \phi_1$$

If the frequency is measured in rad/s, then it is customary to express the phase in appropriate units to this; that is, in radians. (We often then convert this to degrees by multiplying by $180/\pi$.) The phase shift is often negative, which indicates a *phase lag*; that is, the output is lagging behind the input signal. The input sinusoid often has unity magnitude with 0° phase ($u(t) = \cos \omega t$). In this case, the phase of the output signal with respect to the input signal will give the system phase shift directly.

Problem A current-to-pressure transmitter system produces a pressure output signal of $y(t) = 10 \sin(8t - \pi)$ bar when an input signal of $u(t) = 5 \sin(8t - \pi/3)$ amps is injected. Determine the gain and phase shift of the system.

Solution Form a table detailing the important characteristics:

	Input: $u(t)$	Output: $y(t)$
Frequency	8 rad/s	8 rad/s
Magnitude	5 amps	10 bar
Phase	$-\pi/3$ rad	$-\pi$ rad

Gain: The gain can be calculated easily:

$$\text{Gain} = \frac{10 \text{ bar}}{5 \text{ amps}} = 2 \text{ bar/amp}$$

Phase shift: The phase difference is calculated as:

$$\text{Phase shift} = -\pi - (-\pi/3) = -2\pi/3 \text{ rads}$$

This implies that there is a lag in the output signal of $2\pi/3$ rads at this particular frequency.

If we continued to inject, systematically, a range of different frequencies into the system we would be able to analyse the effect the system produced on the input signals and produce a *frequency response* of the system.

14.2 Frequency and logarithmic frequency scales

We will often work in the frequency domain. For control system studies this usually requires some reorientation for two reasons. Firstly, for other engineering applications we may have worked in hertz (Hz); that is, the number of cycles or revolutions/second. Control studies suddenly require you to work in angular frequency, namely rad/s on the frequency axis. Secondly, we may have met different frequency ranges in other engineering areas, for example ultrasonics (~20 kHz), radio frequencies (10 kHz to 1 GHz), radar frequencies (~1 GHz) and microwaves. But in most control studies we will often work in the 0 to 100 Hz frequency range. This is often quite a surprise to some students. We show a scale in Figure 14.6 giving the frequency location of the different activities.

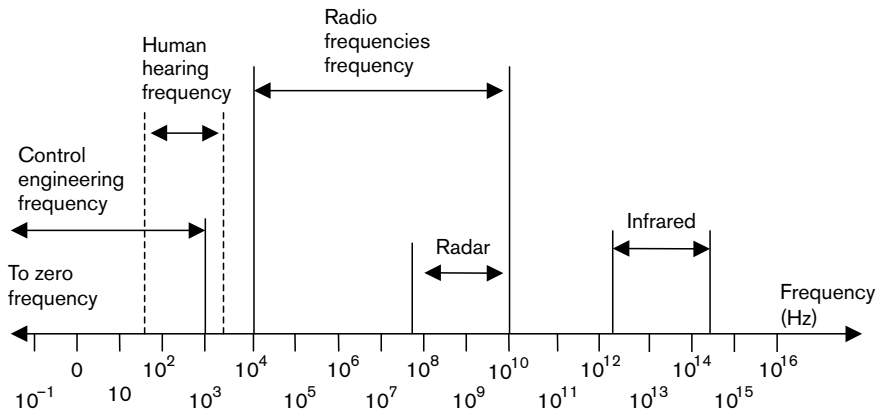


Figure 14.6 Location of control engineering frequencies in the electromagnetic spectrum.

We use angular frequency measured in rad/s for frequency plots and bandwidths and the link between hertz (Hz or cycles/second) and angular frequency (radians/second) is given by:

$$\omega \text{ rad/s} = 2\pi f \text{ where } f \text{ is in Hz.}$$

Consequently our frequency axis may be drawn as Figure 14.7. We note that we do not start the axis at $\omega = 0$ rad/s, but at a low frequency value dependent on the system we are studying.

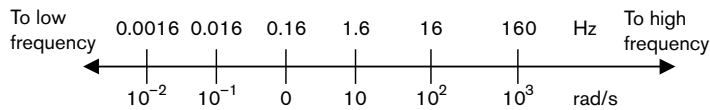


Figure 14.7 The frequency axis in Hz and rad/s.

While we use the frequency units rad/s, we also often find that our graphs have a log frequency scale. Hence in using software packages to set up frequency response plots and other graphs we must be careful not to request $\log_{10}0$, since this would produce an error. For this reason, when using control design packages we would probably use a frequency range of $10^{-3} < \omega < 10^3$ rad/s. This brings us to the logarithmic scale.

Frequency is often plotted on the x-axis as $x = \log_{10}\omega$.

To be able to use this construction, a table of log frequency values is drawn up: Table 14.1.

Table 14.1 Table of $\log_{10}\omega$.

ω	$x = \log_{10}\omega$	ω	$x = \log_{10}\omega$
1	0	10	1.000
2	0.301	20	1.301
3	0.477	30	1.477
4	0.602	40	1.604
5	0.699	50	1.699
6	0.778	60	1.778
7	0.845	70	1.845
8	0.903	80	1.903
9	0.954	90	1.954

This table enables us to position the \log_{10} scale intervals as shown in Figure 14.8(a). Finally, the \log_{10} scale will be seen on plots as Figure 14.8(b).

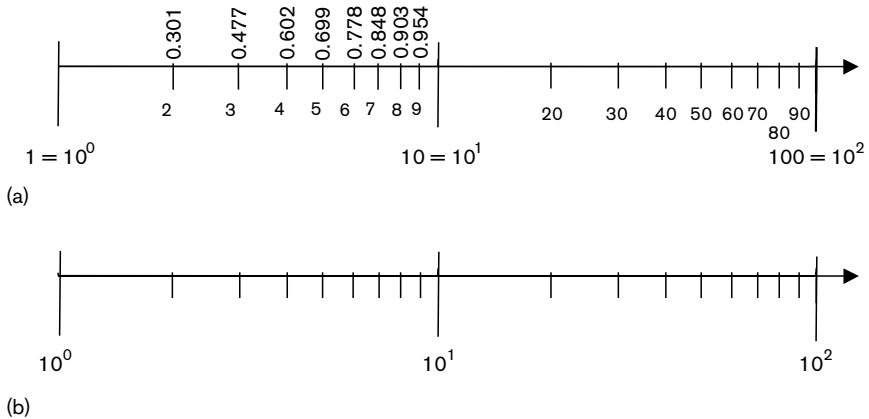


Figure 14.8 (a) The logarithmic scale $x = \log_{10}\omega$; (b) the logarithmic scale.

The user has to be able to read correctly from this scale; for example, can you find 5, 11, and 50 on this scale?

Finally, we call each change of frequency occurring by a multiple of 10 a frequency change of one decade. For example, if we were given the frequency $\omega_0 = 3$ and wished to know the frequency that was one decade higher, we would look for $(10 \times \omega_0) = 30$; two decades higher and we would seek $(10^2 \times \omega_0) = 300$; three decades higher and we would seek $(10^3 \times \omega_0) = 3000$. The frequency that is one decade below $\omega_0 = 3$ is $(10^{-1} \times \omega_0) = \omega_0/10 = 3/10 = 0.3$. Thus on a logarithmic scale this would be shown as in Figure 14.9.

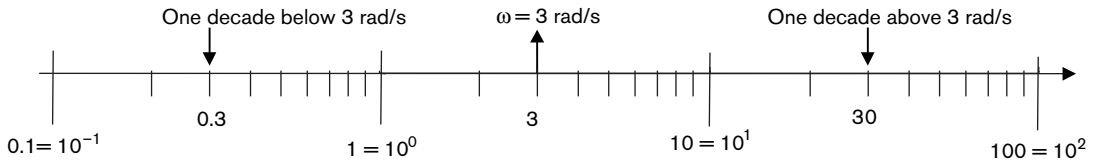


Figure 14.9 Log scales and decade intervals.

In general, if ω_o is the given frequency then we have

$$\omega_A = \omega_o \times 10^n, \text{ where } \omega_A \text{ is } n \text{ decades above } \omega_o$$

and

$$\omega_B = \omega_o \times 10^{-n}, \text{ where } \omega_B \text{ is } n \text{ decades below } \omega_o$$

We will find that we never have to construct a logarithmic scale from first principles because in our laboratories and exercises we will use semi-log paper or computer programs. If you are using control systems design software the flexibility in selecting scales is sometimes bewildering, but the common route is to plot Gain versus \log_{10} frequency or Gain dB versus \log_{10} frequency. It is essential that you know how to read and interpret these scales correctly.

14.3 Presentation of gain and phase information

From our system analysis, we can provide triplets (frequency, gain, phase) of information which indicate to us how a system affects the gain and phase of an input sinusoid at a particular frequency. In engineering terms, the system will process the information carried in the input signals at different frequencies. We often represent this information in one of three different graphical forms:

- a Bode plot
- a Nichols plot
- a Nyquist plot

We will show each of the different forms of frequency, gain and phase information for the following system, which represents a slightly underdamped second-order system:

$$G(s) = \frac{1}{s^2 + 0.6s + 1}$$

In particular, to show the connection between the plots we have selected three frequency points (Table 14.2) and shown them on each presentation format.

14.3.1 General system remarks

A frequency response can usually be divided into a low frequency range, mid-frequency range, and a high frequency range. The phase will alter and, in very general terms, will show some decrease in a particular frequency range as frequency increases. It is also common for the magnitude of the frequency response to decrease at high frequencies; the system will *attenuate* high-frequency signals. This effectively tells us that systems in general only respond weakly to high frequency signals.

Table 14.2 Gain and phase for three particular frequency points for $G(s) = 1/(s^2 + 0.6s + 1)$.

		$\omega_1 = 0.4 \text{ rad/s}$	$\omega_2 = 1.0 \text{ rad/s}$	$\omega_3 = 2.0 \text{ rad/s}$
Polar (angle and radius) measure	Gain (absolute)	1.15	1.67	0.31
	Gain (dB)	1.20	4.45	-10.2
	Phase (deg)	-16°	-91°	-158°
Cartesian (rectangular) measure	x -coordinate	$1.15\cos(-16^\circ)$ = -1.106	$1.67\cos(-91^\circ)$ = -0.029	$0.31\cos(-158^\circ)$ = -0.287
	y -coordinate	$1.15\sin(-16^\circ)$ = -0.317	$1.67\sin(-91^\circ)$ = -1.67	$0.31\sin(-158^\circ)$ = -0.116

14.3.2 The Bode plot

The Bode plot is a graphical representation of the frequency response in the semilog Cartesian coordinate system. It comprises two plots:

1. a **Magnitude** plot: The **gain** of the system versus frequency
2. a **Phase** plot: The **phase** shift induced by the system versus frequency

Frequency axis

The semilog plot has a log scale only on the x (frequency) axis with a linear scale on the y -axis; the magnitude and the phase are plotted against $\log_{10}\omega$; the 'log₁₀' is often omitted when labelling the frequency axis, even though the scale on the frequency axis is $\log_{10}\omega$; the \log_{10} axis does not start at zero.

Magnitude axis

The gain can be plotted in terms of its actual magnitude, but is commonly plotted in units of dB, that is, $20 \log_{10}$ gain versus $\log_{10} \omega$.

Phase axis

The phase axis is linear and usually has units of degrees.

Comment on the system response

There are two graphs produced for a Bode plot: one that describes how the system gain changes over frequency and one that shows the change in phase. In this example (Figure 14.10), we see that the gain is constant at low frequency and has a value of 0 dB. This corresponds to a gain of 1. The gain rises in the mid-frequency range and decreases in the high frequency range. The phase plot starts at 0° and decreases in the mid-frequency range and finally tends to -180°. Finally we note that given three frequency points, $\omega_1, \omega_2, \omega_3$, we can complete the following table. This can be done using MATLAB by plotting the Bode plot using the command `bode(g)`. Using the `ginput(3)` MATLAB function will produce a crosshair cursor which can be used to click on three points on the graph to obtain the coordinates of those points.

	ω_1	ω_2	ω_3
Gain (dB)	1.191	4.396	-10.26
Phase (deg)	-16.10	-90.88	-158.3

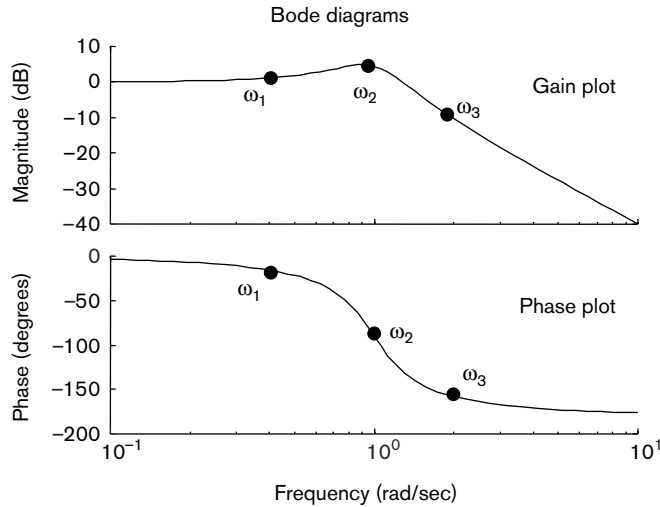


Figure 14.10 Bode plot showing frequencies $\omega_1 < \omega_2 < \omega_3$.

14.3.3 Nichols plots

The Nichols plot uses two axes representing the gain and phase. Frequency is an implicit parameter on the frequency response line.

Horizontal (phase) axis

The horizontal axis is used to represent the phase and usually has units of degrees.

Vertical (magnitude) axis

The vertical axis represents the gain and is commonly divided into units of dB, that is, $20 \log_{10}$ gain.

Comment on system response

We note that the frequency is not represented on an axis in this format (Figure 14.11). Any point on the frequency response has an associated frequency value. We can see that the magnitude diminishes as the frequency increases; that is, the frequency response line takes values lower down the vertical, or magnitude, axis. The phase also decreases as frequency increases; however, we note that decreasing phase is indicated by a line moving to the left.

As we did with the Bode plot we can produce a table of values from the plot. However, we note that we are not able to find a frequency value directly from this plot and we would have to find the value of frequency from other sources.

	ω_1	ω_2	ω_3
Gain (dB)	1.191	4.396	-10.26
Phase (deg)	-16.10	-90.88	-158.3

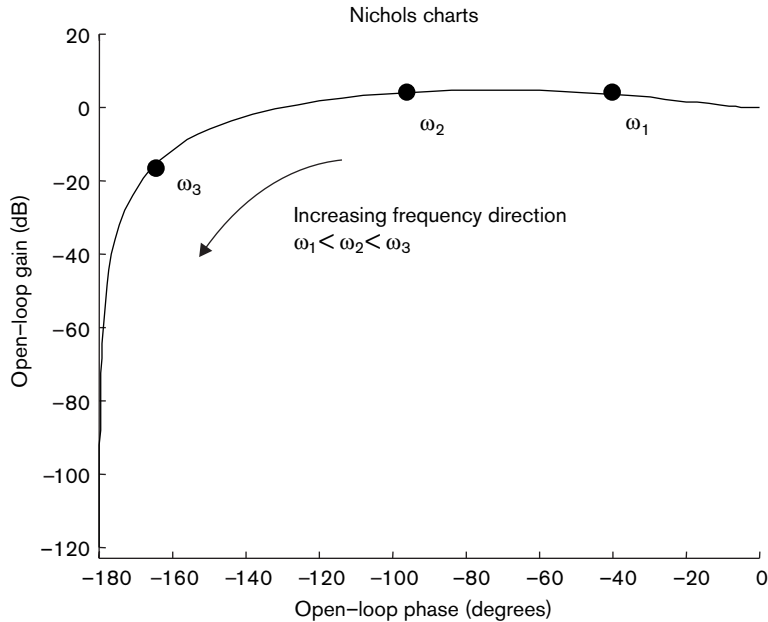


Figure 14.11 Nichols plot showing frequencies $\omega_1 < \omega_2 < \omega_3$.

One of the other uses of Nichols plots is to overlay lines of *closed-loop* information. When this is done it is referred to as a Nichols **chart** (Chapter 17), and it is this *chart* that is often used for control design.

14.3.4 Nyquist plots

This plot has a Cartesian grid (Figure 14.12); there is no frequency axis, nor any gain or phase axis. The gain and phase are plotted as polar coordinates (angle and magnitude from the origin). Therefore lines of constant gain are circles from the origin. Each point on the response curve corresponds to a specific frequency point.

M Problem

Use MATLAB to find the real and imaginary parts from the Nyquist plot and the associated gain and phase calculations.

Solution MATLAB information: finding points in a file of data

```
s = tf('s');
g = 1/(s^2+0.6*s+1)
w = logspace(-1,1,500); % Create frequency variable, w, of 500 points
% from 0.1 (10-1) to 10 (101) rad/s.
[Re,Im]=nyquist(g,w); % Calculates the values required for a Nyquist
% plot using the frequency variable, w. Stores the
% results in the variables 'Im' and 'Real'.
i = find(w>0.4); % Creates a variable i which contains the
% indices of all variables satisfying the test
% criteria.
```

```

i(1) % provides the first index number, that is the
      % index number for which w > 0.4.

ans =
    152
w(152);
ans =
    0.4029 % using this index will provide the index of
           % real and imaginary parts at that frequency

Re(152), Im(152)
ans =
    1.1020
ans =
   -0.3180
Mag = sqrt(Re(152)^2+Im(152)^2)
ans =
    1.1470
Phase = atan(Im(152)/Re(152)) *180/pi
ans =
   -16.10

```

Table 14.3 tabulates the results from the MATLAB calculations.

Table 14.3 MATLAB results for gain and phase calculations.

	ω_1	ω_2	ω_3
Real	1.1020	-0.0255	-0.2851
Imag	-0.3180	-1.6586	-0.1133
Gain = $\sqrt{\text{Re}^2 + \text{Im}^2}$	1.1470	1.6588	0.3068
Phase = $\tan^{-1}(\text{Im}/\text{Re})$	-16.10°	-90.88°	-158.3°

Remark

One point to remember when using a ' \tan^{-1} ' function is that the tangent function repeats itself and therefore the inverse is given either between 0 and 180°, or between -90° and +90°. We must use our *inside knowledge* to interpret the result. In the above example, the phase values were given as 89.12 and 21.68 degrees. From experience with the Bode and Nichols plots we found that the phase decreased towards higher frequencies and we would not expect it to rise. Bearing in mind that the answer could be 180° shifted, we subtract 180° from these positive results to give the correct results in the table.

Comments on system response

If we look at Figure 14.12 we see that as frequency increases and the gain and phase decrease, the response line spirals in towards the origin (zero gain). For example, a gain of 0 dB corresponds to a gain of 1 which can be represented by a circle of radius 1 centred on the origin. We note that the unit 'circle' appears squashed if the axes used by MATLAB are not square.

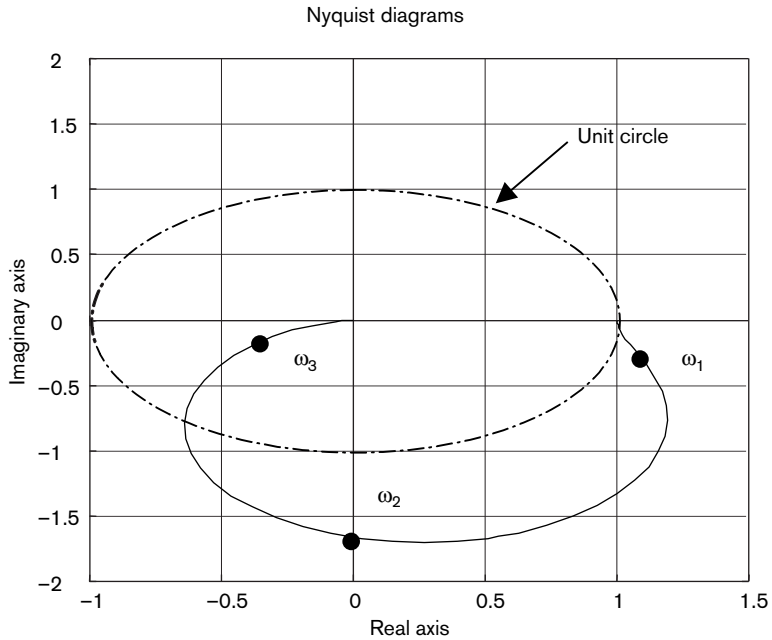


Figure 14.12 Nyquist plot showing frequencies $\omega_1 < \omega_2 < \omega_3$.

Each of these forms of presenting information (Bode, Nichols, Nyquist) has advantages when using different control design methods. In this chapter we look at recognising the information from the graphs and finding some important frequency points. These frequency points can then be used to help us to determine the stability of the closed-loop system.

14.4 The frequency response and system features

We can use the system’s frequency plot to predict the behaviour of the system (Figure 14.13). If we know the frequency of the input signal, we can use the frequency response to show us how the system, or process, $G(s)$ affects the input signal.

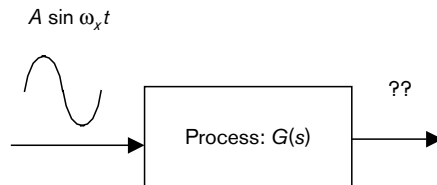


Figure 14.13 Using the frequency response to predict the output.

For the following explanations we shall refer to the following system model:

$$G(s) = \frac{400}{(s+10)(s^2+0.4s+4)}$$

which represents a third-order system with low damping. We will illustrate each feature on the frequency response plot using the Bode plot, Nichols plot and Nyquist plot.

14.4.1 Amplification and attenuation

- (a) An input signal component at frequency ω is **amplified** if the gain on the frequency response plot is *greater* than 1 (or greater than 0 dB) at that frequency.
- (b) An input signal component at frequency ω is **attenuated** if the gain on the frequency response plot is *less* than 1 (or less than 0 dB) at that frequency.

If the gain on the frequency response plot is exactly 1 (or 0 dB), the input and output signal component will have the same magnitude and the input component is therefore neither amplified nor attenuated.

If we examine our three graphical representations of information we find the ranges of amplification and attenuation illustrated (Figure 14.14).

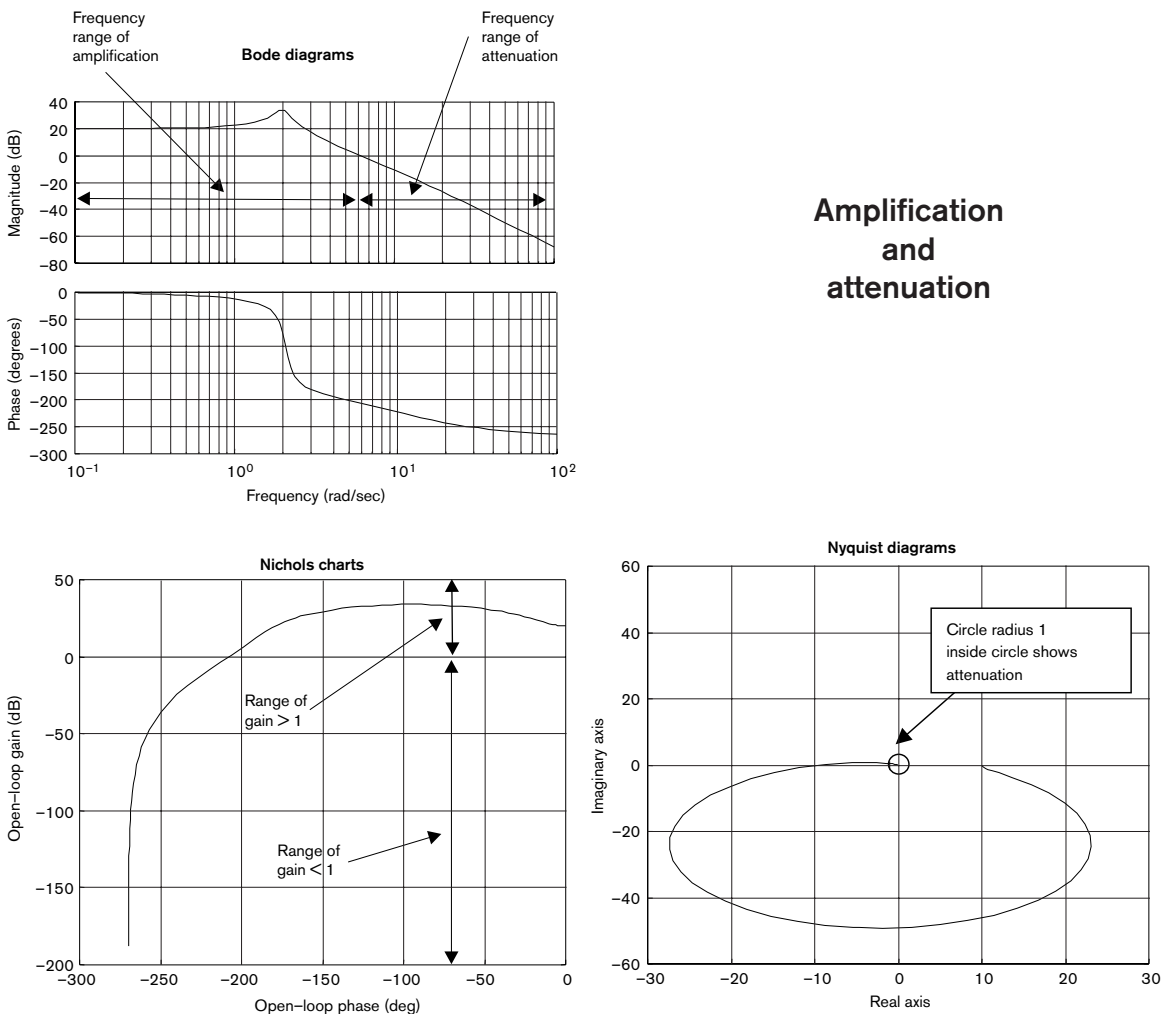


Figure 14.14 Amplification and attenuation.

Bode plot

The amplification and attenuation are shown on the magnitude plot. The magnitude scale is given in dB; a value of 0 dB corresponds to a magnitude of 1. The graphs show that low-frequency input signals will be amplified (or magnified) while the effect of high-frequency signals will be attenuated (or lessened). In the low-frequency range the gain level is 20 dB, which corresponds to a gain of 10. Therefore any input sinusoidal signal in this frequency range will be amplified by a factor of 10.

In the high-frequency range the gain is much lower than 0 dB and decreasing. Therefore in the high-frequency range we say that the input sinusoidal signal magnitudes are attenuated.

Nichols plot

The vertical axis on the Nichols plot is the magnitude axis. A system with a gain greater than 1 (or 0 dB) will provide amplification of input signals.

Nyquist plot

On the Nyquist plot, the frequency response can be represented by polar coordinates. Therefore the gain magnitude is the distance from the origin. For a signal to be amplified, the gain magnitude must be greater than 1. By drawing a circle of unity radius, we can find the points where the input signals would be attenuated (those that lie inside this circle). The points on the frequency response outside the circle will be magnified.

14.4.2 D.C. gain

We are often interested in an engineering system's effect on a constant or *d.c.* signal: effectively a signal with 'zero' frequency. On a frequency response plot, we therefore look at the low-frequency section. If the low-frequency amplification or attenuation level is constant, we refer to this as the **d.c. gain**. We use Figure 14.15 for finding the d.c. gain.

Bode plot

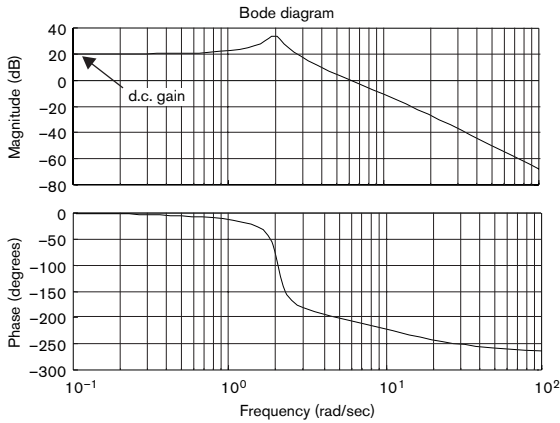
On a Bode plot, the d.c. gain is found on the magnitude plot. Since this is a semilog plot where the frequency scale is in $\log_{10}\omega$, the value of '0' is not marked (since in \log_{10} the frequencies decrease as 0.1, 0.01, 0.001, 0.0001, ...). It is sufficient to take the value in the low-frequency range. The system response shows the d.c. gain to be 20 dB and therefore the engineering system would amplify any slowly varying (low-frequency) signal by a gain of 10.

Nichols plot

On the Nichols plot we can find the value of the d.c. gain from the vertical (magnitude) axis. The low-frequency section on a Nichols plot often starts at the top right-hand side (near 0° phase value).

Nyquist plot

On a Nyquist plot the d.c. gain is a magnitude and hence will be the distance from the origin. Since the low-frequency section often starts at 0° phase, this is represented by the real axis and the d.c. gain will be the point on the real axis at which the plot starts.



d.c. gain

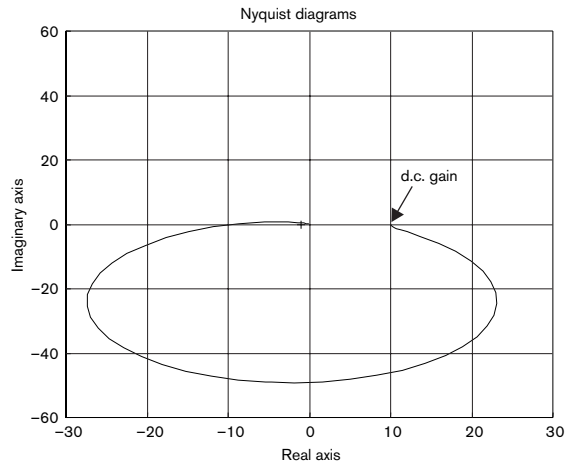
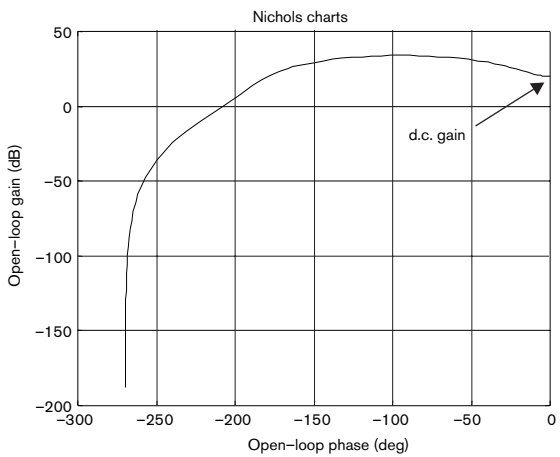


Figure 14.15 Finding the d.c. gain.

14.4.3 Roll-off rate

The roll-off rate is often referred to when using Bode plots. In the high-frequency range of the frequency response plot the gain is decreasing, the level is less than 0 dB and the magnitude of the input signal component will be attenuated. Since the gradient is constant and the graph is decreasing, as we increase the frequency of the input signal, the magnitude of the output signal becomes less and less, giving large attenuation to the high-frequency signals. For example, noise on electrical signals often occurs at high frequency and this will be attenuated, or lessened, by a system which has low gain at high frequencies. The steepness of this line is important for systems that are designed to attenuate certain disturbances or noise. Hence the gradient of the slope is often referred to as the *high-frequency roll-off*, the *high-frequency asymptote* or the *cut-off rate* of the system.

The gradient is calculated by determining the change in magnitude over one decade of frequency. The roll-off rate is then stated, for example, as -20 dB/decade or -40 dB/decade.

In the system response shown in Figure 14.16, the system is third order and this is associated with a roll-off rate of -60 dB/decade.

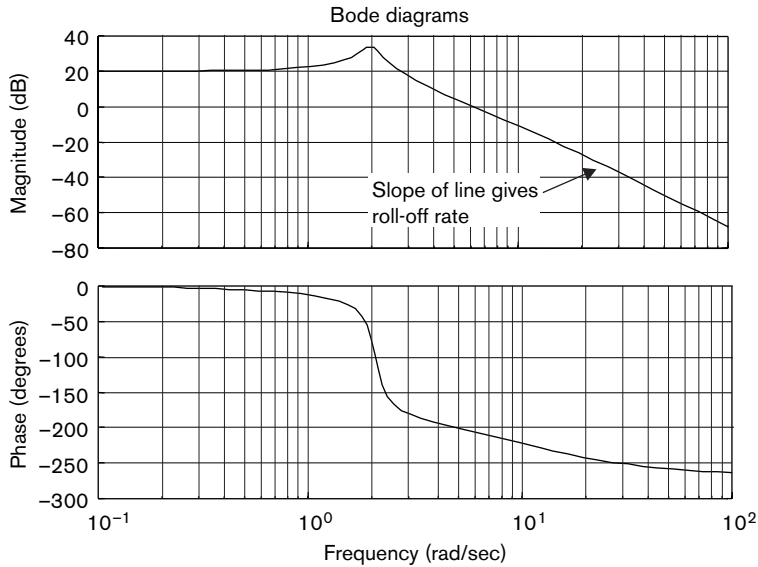


Figure 14.16 Finding the roll-off rate.

14.4.4 Phase graphs

The shape of a phase curve is important when we discuss the phase lag introduced by a system. If there is almost no phase lag, that is, for slowly varying (or low-frequency) signals, the output signal follows the input signal with little perceptible lag. As the frequency of the input signal increases the output signal starts to lag behind the input signal. The combination of amplification and phase lags of -180° can cause closed-loop system instability. This is discussed later as part of the design specifications for frequency domain design.

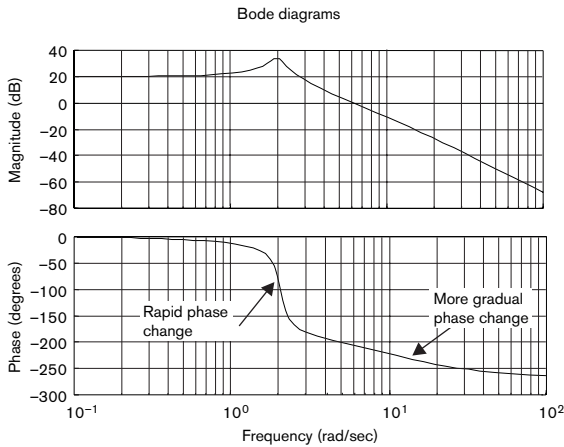
Example: Phase lag

Consider a heating system. We can inject a slowly varying energy signal into the heating system and watch the corresponding temperature change; slow changes to demanded temperature are followed fairly easily. If you provide a quickly varying input energy signal (higher frequency), the room or building that you are heating cannot change temperature fast enough (due to the thermal inertia of the room). Therefore any fast changes do not appear in the output signal. The point at which the input signal alters from being a 'slowly changing' signal to a 'rapidly changing' signal usually occurs in the middle frequency range and depends on the actual system.

14.4.5 Comment on phase change (Figure 14.17)

Bode plot

From the phase graph in the Bode plot, we can see a rapid change in phase between 1 and 3 rad/s. Hence any design which depended on frequencies in this range would have to allow for the fact that the phase changes dramatically over this short frequency range. At higher frequencies the phase changes more gradually.



Phase change

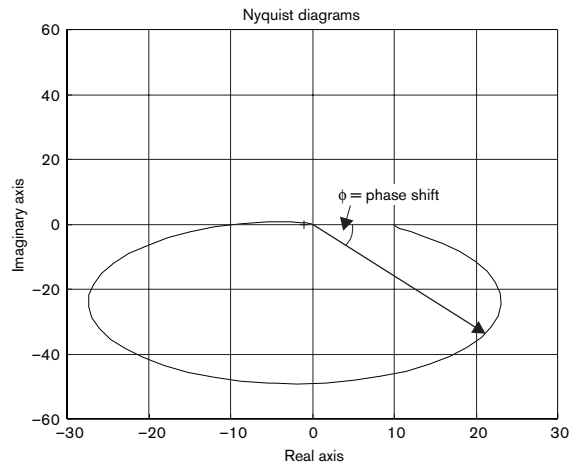
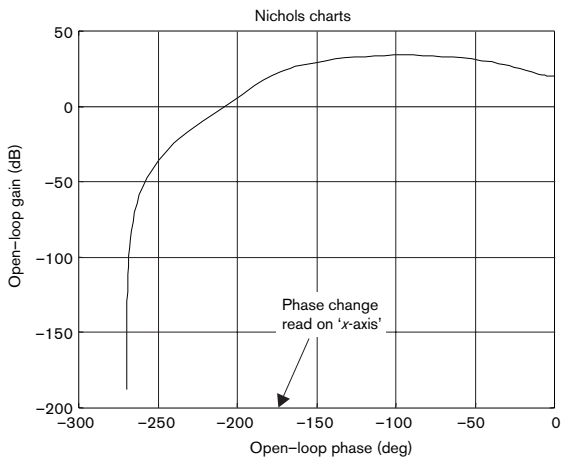


Figure 14.17 Phase change on Bode, Nichols and Nyquist plots.

Nichols plot

The Nichols plot gives us the opportunity to examine how the gain and phase changes occur together. We see that the gain starts off with a value greater than 0 dB at low phase values. Since there is no frequency axis, we cannot immediately see the rate of change of ϕ with frequency. However, we find that the phase changes from 0° to -270° .

Nyquist plot

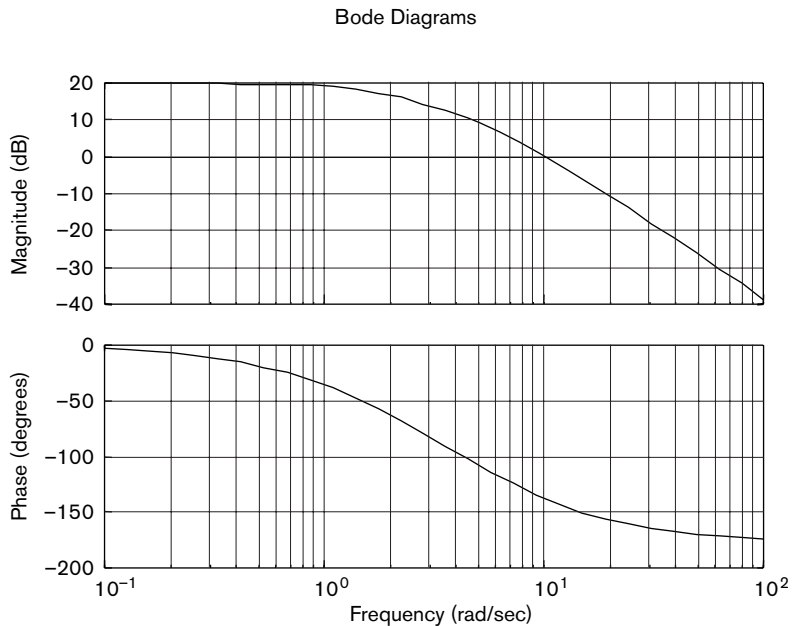
The phase on the Nyquist plot is given by the angle between the line drawn from the origin to the frequency response line and the real axis. At low frequency, the Nyquist plot will usually start on the real axis (0° phase). In this example, the frequency response line spirals into the origin. Therefore the gain is eventually decreasing (to effectively zero) and the phase alters from 0° to a maximum of -270° .

Skill section

Problems Consider the following plots. Answer the following questions, indicating whether it is an inappropriate question for the plot and information given.

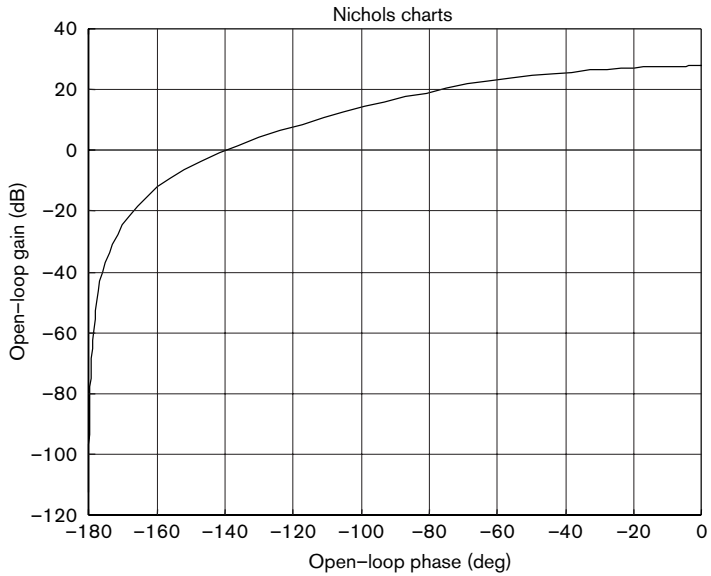
- What is the d.c. gain?
- State the frequency range where the amplification is above 10 dB.
- At what frequency do we find 10 dB of attenuation?
- What is the roll-off rate?
- What is the gain when the phase reaches -135° ?

Solution: Bode plots



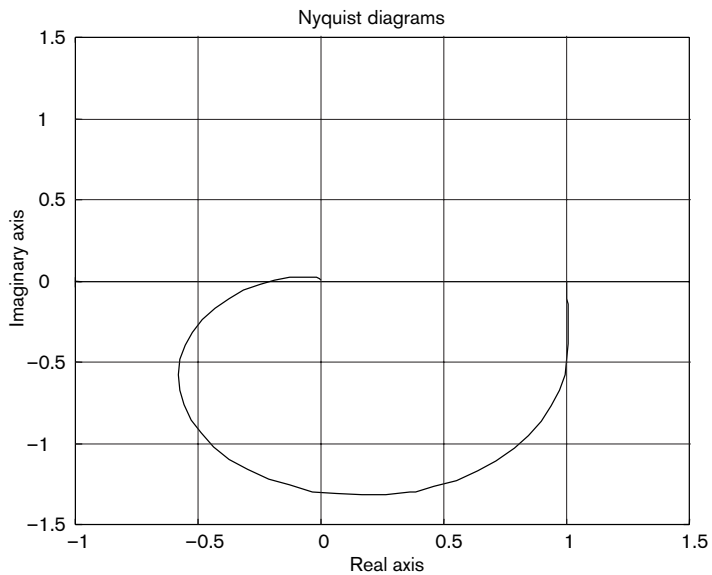
- D.c. gain = 20 dB: absolute gain of 10
- Amplification above 10 dB: low frequency to 5 rad/s
- Attenuation of 10 dB occurs at 20 rad/s
- Roll-off rate is -40 dB/decade
- Gain value when phase = -135° : about 3 dB

Solution: Nichols plot



- (a) D.c. gain approx 28 dB
- (b) Amplification above 10 dB: no frequency range given on graph
- (c) Attenuation of 10 dB = gain of 0.316: no frequency range given on graph, but point on frequency response can be plotted
- (d) Roll-off rate: not appropriate for Nichols plot
- (e) Gain value when phase = -135° : about 5 dB

Solution: Nyquist plot



- (a) D.c. gain = 1. In this case, the gain on the frequency response curve is 1 when the frequency response curve lies on the real axis (0° phase)
- (b) Amplification above 10 dB: no frequency range given on graph, although we can see that there is amplification since the distance from the origin (gain) is greater than unity in a region of the response
- (c) Attenuation of 10 dB = gain of 0.316: no frequency range given on graph, but point on frequency response can be plotted
- (d) Roll-off rate: not appropriate for Nyquist plot
- (e) Gain value when phase = -135° : magnitude from origin = (approx) $\sqrt{0.6^2 + 0.7^2} = 0.922$

14.5 Special frequency points

There are two special frequency points which we use later when we look at the stability of closed-loop systems. In this section we introduce these points and examine where they occur on our three graphical representations.

Key result: Crossover points

- *Gain crossover frequency* = ω_{gco} = frequency at which the gain crosses 0 dB
- *Phase crossover frequency* = ω_{pco} = frequency at which the phase crosses -180°

Gain crossover: in simple systems where we assume that there is only one gain crossover, the system will amplify signals before the gain crossover and attenuate signals after this frequency. Therefore as the crossover frequency of a system increases, it will be able to respond with reasonable magnitude over an increasing frequency range of input signals. If the gain crossover is at high frequencies, this corresponds to being able to react faster to certain input signals. A low crossover frequency could indicate a slowly responding system. The determination of 'low' or 'high' is system-dependent.

Phase crossover: the main point to note is that not all systems will have a phase crossover. In a first-order system, the phase change is only $0^\circ \rightarrow -90^\circ$ and the phase does not cross -180° . Even for a pure second-order system, the phase change only reaches -180° and does not actually *cross* this point. The importance of this point is related to the fact that if a sinusoid is injected into a system and the system inverts sinusoids at this frequency, then the output signal is exactly 180° out-of-phase with its input. This has consequences in control systems where we use negative feedback to compare the output signal with the desired set point.

The gain and phase crossover points for a third-order system are shown for the three presentation plots, namely Figures 14.18(a), (b) and (c).

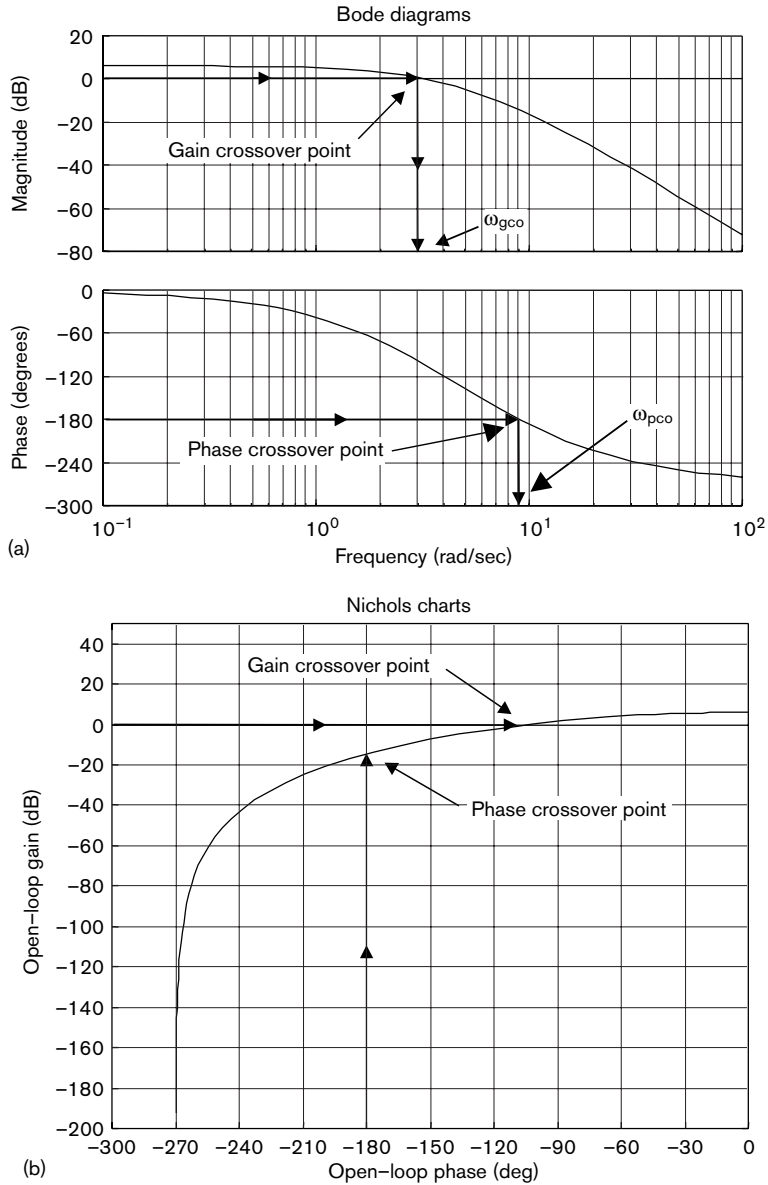


Figure 14.18 (a) Crossover points on Bode diagrams. (b) Crossover points on Nichols plot.

14.5.1 Bandwidth

Bandwidth is a parameter that can be applied to both signals and systems; it indicates the range of frequencies of significant signal power or the range of frequencies over which the system will have a significant magnitude of output signal. There are slightly different, but important, interpretations of this parameter when it is related to either signals or open-loop/closed-loop systems. We state firstly the use of the term bandwidth for signals and open-loop systems before discussing the importance of the closed-loop bandwidth for closed-loop system performance.

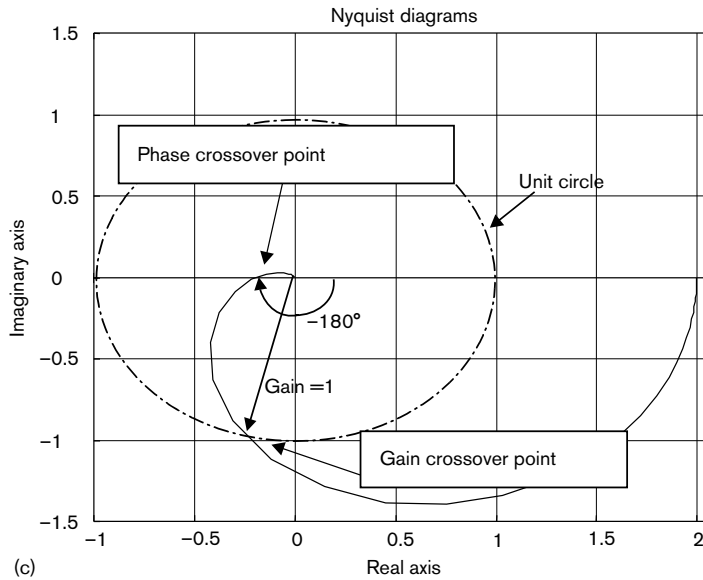


Figure 14.18 (c) Crossover points on Nyquist diagram.

Signal bandwidth

The signal bandwidth is defined for

1. those signals which have a midband range or peak (Figure 14.19(a))
2. those signals with a constant d.c. gain (Figure 14.19(b))

It is defined as the range of frequencies over which the gain is greater than $G/\sqrt{2} = 0.707G$. Since we use often use log scales for gain, we note that the value of $0.707 = -3$ dB.

In Figure 14.19(a), the bandwidth is given by

$$\omega_{bw}: \omega_L \text{ to } \omega_H$$

These two frequencies, ω_L and ω_H , are referred to as the lower and upper half-power points, since the signal power will be reduced to one half of the signal power in the bandwidth range outside these frequency points. (At the point where the power falls by a half, the gain ($=\sqrt{\text{power}}$) falls by a factor of $1/\sqrt{2} = -3$ dB.) However, in Figure 14.19(b), where the d.c. gain is constant, there is no lower half-power point and the bandwidth is simply referred to by the frequency at the upper half-power point, ω_{bw} .

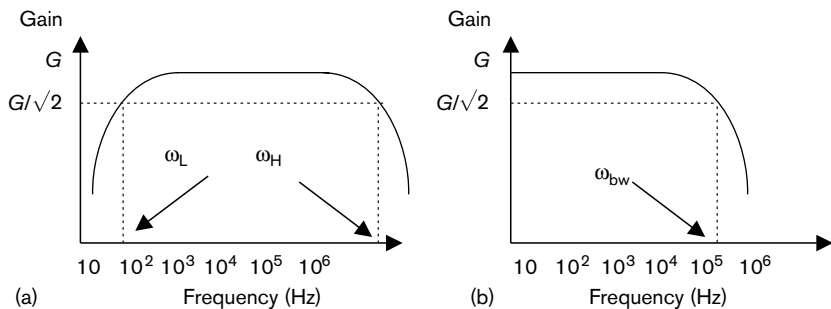


Figure 14.19 Signal bandwidth.

Open-loop system bandwidth

The system bandwidth is only defined for those systems with a *constant* d.c. gain.

Open-loop system bandwidth = ω_{bw} = the frequency range over which the gain is 3 dB down on the d.c. gain

In Figure 14.20(a) the bandwidth is given by ω_{bw} and is the frequency that corresponds to a magnitude which is 3 dB lower than the d.c. level. In control systems we usually refer to the bandwidth occurring at the -3 dB point. Another example of the frequency response of an open-loop system is shown in Figure 14.20(b). However, we cannot state the open-loop system bandwidth for this system since the requirement for a constant d.c. gain has not been satisfied.

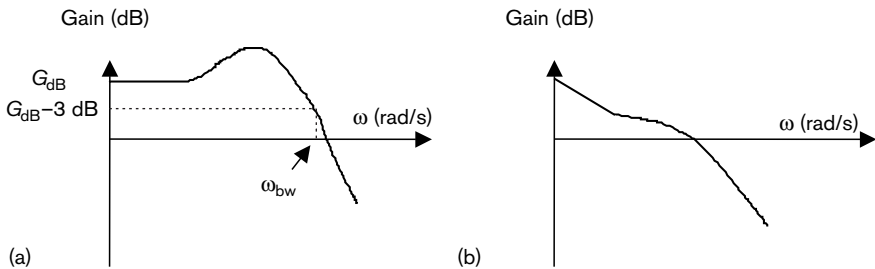


Figure 14.20 Bandwidth interpretations for open-loop systems.

Closed-loop system bandwidth

Closed-loop system bandwidth = ω_{bw} = the frequency range over which the gain is 3 dB down on the zero frequency gain

The application of feedback to open-loop systems such as those shown in Figures 14.20(a) and (b) produces a system whose frequency response has a finite d.c. gain. An example is shown in Figure 14.21, where the gain is unity (or 0 dB) at low frequencies. This is desirable since it implies that all low-frequency signals (for example slowly changing control demand signals) will be passed through by the system and not attenuated. The shape of the gain in the mid-range frequency will change depending on the actual system, while the high-frequency range shows a roll-off which will reduce the effect of high frequencies (noise) within the control system. Since it is desirable for the low-frequency signals to pass through the system unchanged, the 0 dB level at zero frequency is quite common. The bandwidth in this case is actually at the -3 dB level, and indicates the speed of response of the control system and the frequency above which the system may reduce the effects of noise.

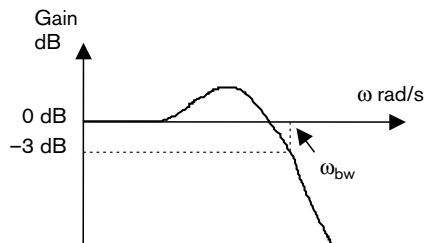


Figure 14.21 Bandwidth interpretation for closed-loop frequency response.

The closed-loop system bandwidth can, for simple systems, be closely related to the open-loop system gain crossover. This is important for control systems design, which is often performed by shaping the open-loop frequency response plot. The design specification on system bandwidth is therefore translated to an equivalent gain crossover specification which can be applied to design on the open-loop frequency response.

14.6 Interpretation of frequency response plot

Consider the mechanical example shown in Figure 14.22. It is a simple version of many mechanical oscillating systems, such as suspension systems. The values of the mass, spring and damper are chosen such that the system oscillates when an input force is applied at certain frequencies. For these values, the output signal (the position of the mass) is 1 metre when the input force signal is given as 1 newton. The corresponding frequency response plot is shown in Figure 14.23.

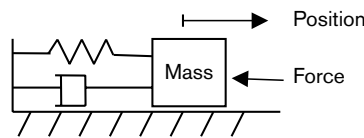


Figure 14.22 Simple mechanical system.

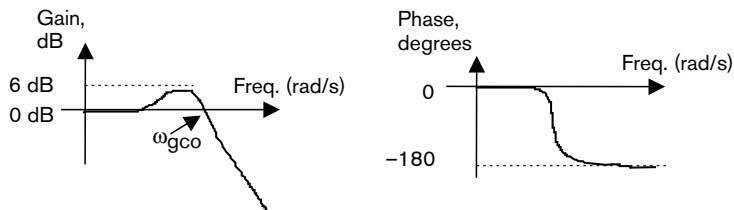


Figure 14.23 Frequency response for simple mechanical system.

Examine the response in the frequency ranges we have discussed.

- *Low-frequency range:* here we can see that the magnitude is 0 dB (gain of 1). Hence every low-frequency input signal injected to the system will give a system output which is neither amplified nor attenuated but is of the same magnitude. The phase shift in this region is 0° , which implies that the output signal will not lag behind the input. For the mechanical system it implies that the mass will follow the direction and magnitude of the input force.
- *Middle frequency range:* the system amplifies the input signals in part of this range, with a maximum amplification of 6 dB (= a gain of 2). The gain crossover frequency is in this range and is indicated by ω_{gco} . After ω_{gco} the system attenuates all input signals. If we consider the phase response, we can see that the phase plot shows a rapid decrease in phase for this system. This indicates that over this middle frequency range the output signals will lag further and further behind the input signals. Physically, the mechanical system will have a resonant frequency in this middle range and near this frequency small magnitude input forces will result in large changes in the mass's posi-

tion. As the frequency of the force input signal increases, the mass–spring–damper system cannot follow the changes of the input signal and begins to lag behind.

- *High frequency range*: the magnitude plot shows a clear *increasing* attenuation of the input signals. At very high frequency, the amplitude of the output signals will be very small. The phase response for this example shows the phase lag levelling off to -180° . This indicates that at high frequencies the output signals will be fully ‘out of phase’ with the input signals. This becomes important later when we discuss stability. If we apply very fast-changing force signals to the mechanical system, the system is not capable of responding – the higher the frequency the smaller the change in the output position becomes.

Problem: Rotating disk system

The storage disk assembly of a computer system is shown in Figure 14.24(a). The input signal is a torque on the drive system and the output signal is the position of the disk. The corresponding frequency response plot is shown in Figure 14.24(b). Describe the features on the frequency response plot, specifically referring back to the physical rotating disk example.

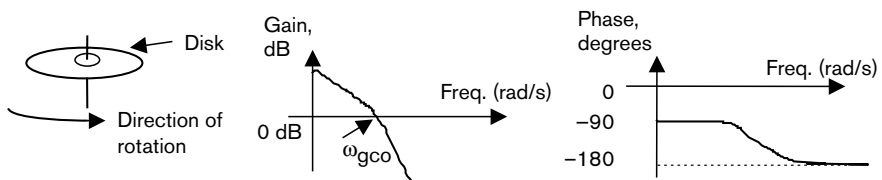


Figure 14.24 Frequency response for rotating system.

Solution Examine the frequency plots in the low, middle and high frequency ranges.

- *Low frequency range*: the input signals are amplified in this range and there is no constant d.c. gain. In fact, as the frequency decreases towards zero, the gain keeps increasing, giving an *infinite* d.c. gain. This is also known as ‘high gain at low frequencies’. In physical terms, if you apply a constant input signal to the system, the disk will keep rotating, therefore the position of the disk from its initial position will continue to increase. By applying a low-frequency torque, you are effectively applying a very slowly changing input signal which will have a similar interpretation to the constant torque input. The output position will lag 90° behind the input torque in this range.
- *Middle frequency range*: the gain crossover frequency lies in this range. The output signal changes from being amplified by the system at frequencies above the gain crossover frequency, to being attenuated below this frequency. The phase lag increases in this region. This corresponds to the disk system not responding fast enough to the changes in the input signal.
- *High frequency range*: in this range we see that the input signal is severely attenuated; the disk cannot physically follow a high-frequency sinusoidal input. The output signal is 180° out of phase with the input.

14.7 Performance specification: gain and phase margins

We have looked at how to read two important frequency points from a frequency response plot. These points were the gain crossover point, where the magnitude of the system crosses a value of 1, and the phase crossover point, where the phase of the system crosses -180° . We now examine why these points are important.

Consider an open-loop system as shown in Figure 14.25. We see that at ω_x rad/s, the process gain is given by K and the phase shift is -180° . An output signal which is 180° (π radians) out of phase with the input will be the inversion of the sinusoidal signal, and hence we can apply a negative sign to the signal. For example, if we let the amplitude of the input signal, A , be 2 and the process gain and phase at 5 rad/s be 3 and -180° respectively, we find that:

$$\text{input sinusoid} = 2 \sin 5t$$

$$\text{output sinusoid} = -6 \sin 5t$$

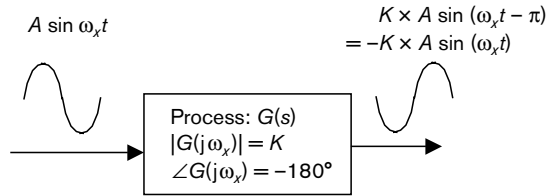


Figure 14.25 Open-loop frequency response.

We now examine how a sinusoidal signal would be passed round a feedback loop applied to this process (Figure 14.26). In this example we have let the frequency, $\omega_x = 5$ rad/s.

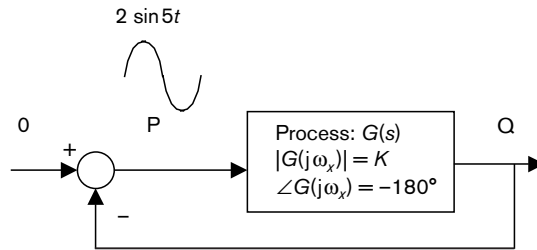


Figure 14.26 Stability of systems.

We will examine what would happen for three values of gain: $K = 0.5$, $K = 1$ and $K = 2$.

Example 1: Gain $K = 0.5$

At point P:	At point Q:
$2 \sin 5t$	$-0.5 \times 2 \sin 5t = -1 \sin 5t$
$0 - (-1 \sin 5t) = 1 \sin 5t$	$-0.5 \times 1 \sin 5t = -0.5 \sin 5t$
$0 - (-0.5 \sin 5t) = 0.5 \sin 5t$	$-0.5 \times 0.5 \sin 5t = -0.25 \sin 5t$
$0 - (-0.25 \sin 5t) = 0.25 \sin 5t$	$-0.5 \times 0.25 \sin 5t = -0.125 \sin 5t$
$0 - (-0.125 \sin 5t) = 0.125 \sin 5t$	$-0.5 \times 0.125 \sin 5t = -0.0625 \sin 5t$
$0 - (-0.0625 \sin 5t) = 0.0625 \sin 5t$	$-0.5 \times 0.0625 \sin 5t = -0.03125 \sin 5t$

We can see that the magnitude of the sine wave is decaying and is not becoming an unbounded signal. We shall now repeat the example, but let the value of K be 1.

Example 2: Gain $K = 1$

At point P:	At point Q:
$2 \sin 5t$	$-1 \times 2 \sin 5t = -2 \sin 5t$
$0 - (-2 \sin 5t) = 2 \sin 5t$	$-1 \times 2 \sin 5t = -2 \sin 5t$
$0 - (-2 \sin 5t) = 2 \sin 5t$	$-1 \times 2 \sin 5t = -2 \sin 5t$

In this example, we find that the system output does not increase or decrease, but continually oscillates. This is because the total phase shift round the loop is -360° ; -180° from the process *plus* -180° from the negative feedback summation, and the gain is unity. Any signal passed round the loop remains in phase with itself and does not alter in magnitude.

The last example looks at a value of K greater than 1: $K = 3$.

Example 3: Gain $K = 3$

At point P:	At point Q:
$2 \sin 5t = 2 \sin 5t$	$-3 \times 2 \sin 5t = -6 \sin 5t$
$0 - (-6 \sin 5t) = 6 \sin 5t$	$-3 \times 6 \sin 5t = -18 \sin 5t$
$0 - (-18 \sin 5t) = 18 \sin 5t$	$-3 \times 18 \sin 5t = -54 \sin 5t$
$0 - (-54 \sin 5t) = 54 \sin 5t$	$-3 \times 54 \sin 5t = -162 \sin 5t$

We can see that, very quickly in this case, the output signal is becoming unbounded. The phase shift of -180° combines with the negative feedback sign to provide a signal in phase with itself but since the gain is greater than one, the magnitude of the signal is increased with each pass round the loop, giving an unbounded signal at the output.

We can intuitively deduce from the above, that if, when the open-loop phase of the system is -180° , the stability of the output will depend on the magnitude of the open-loop gain:

Key result: Test for closed loop instability

- If the open-loop gain is greater than unity when the phase crosses -180° , the closed-loop system will be unstable.

We can also write that:

- If the phase is greater than -180° when the open-loop gain is 1, the closed-loop system will be unstable.

The above is important, since it shows that we can deduce the stability of the closed-loop system by knowing the *open-loop* gain and phase.

14.7.1 Stability determination

We can use the gain and phase indicators of stability to provide a specification for the open-loop system. For example, we can plot the open-loop gain and phase and check to ensure that the closed-loop system is stable. If we consider a general feedback system (Figure 14.27), the open-loop transfer function would be given by

$$G_{OL}(s) = G(s)K(s)H(s)$$

though often in our simple examples we assume that $H(s) = 1$. We also assume in this textbook that we do not have system with multiple gain crossover or phase crossover frequency points. These can lead to *conditionally* stable systems which are not dealt with at this level.

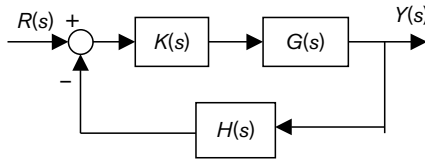


Figure 14.27 General feedback system.

The following is the procedure for checking if the closed-loop system is stable:

Test 1

From the Bode plot of the open-loop transfer function:

1. Find the gain crossover, ω_{gco} .
2. Determine the phase at that point, $\phi(\omega_{gco})$.
3. Is the phase below -180° ? If so, then the closed-loop system is unstable.

Test 2

From the Bode plot of the open-loop transfer function:

1. Find the phase crossover, ω_{pco} .
2. Determine the gain at that point, $|G(\omega_{pco})|$.
3. Is the gain greater than 1? If so, the closed-loop system is unstable.

Remark

We note from the above that although either test will prove that the closed-loop system is unstable, we need to perform *both* tests to determine whether the system is *stable*.

Example Determine from the Bode plot in Figure 14.28 of an open-loop system whether the closed-loop system would be stable.

From Test 1, we find that the gain crossover point, ω_{gco} , is at 3 rad/s. The phase value at this point is almost -210° , that is, below -180° , and the closed-loop system is unstable. We could stop here now, since this is sufficient to prove closed-loop instability, but for practice we shall also do Test 2.

We find that the phase crossover point is at 2 rad/s. The value of gain at this frequency is around 15 dB, well over 0 dB, and the system is unstable.

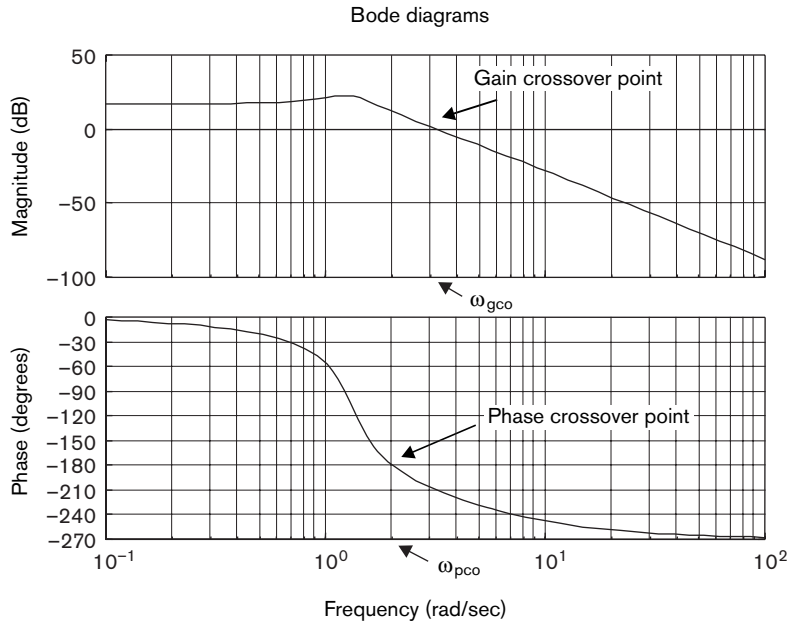


Figure 14.28 Bode plot of open-loop process.

14.7.2 Frequency domain specifications

We would not expect our system models (often given as transfer functions) to be absolutely accurate; therefore we would like to include a margin of error in our design specifications. We would like to have a *gain margin* and a *phase margin* so that the resulting systems are not performing near the bounds of their stability.

In practice, at the gain crossover (when the gain crosses 1 (or 0 dB)), we would like the phase not to be 'just above' -180° but to take values of at least -135° to -120° ; that is, for the phase plot to have a phase margin of 45° to 60° .

Likewise, at the phase crossover (when the phase crosses -180°), we would like the gain not to be 'just below' the 0 dB line, but to take values of -6 dB to -8 dB, giving gain margins of 6 dB to 8 dB.

In other words, the gain margin represents how much gain you can add (in dB) to the gain plot before the closed-loop system would be unstable. The phase margin represents how much phase you can lose before instability occurs.

We can adapt Test 1 and Test 2 above to form a procedure for finding the gain and phase margins.

Phase Margin (PM) procedure

1. Find the gain crossover, ω_{gco} .
2. Determine the phase at that point, $\phi(\omega_{gco})$.
3. Calculate the phase margin: $PM = \phi(\omega_{gco}) - (-180^\circ)$.

If the PM is negative the system is unstable. If the PM is less than 45° , then it may be appropriate to consider redesigning the controller, $K(s)$, to improve the PM of the open-loop system.

Gain Margin (GM) procedure

1. Find the phase crossover, ω_{pco} .
2. Determine the gain at that point, $G(\omega_{\text{pco}})$.
3. Calculate the gain margin: $\text{GM} = 0 \text{ dB} - G(\omega_{\text{pco}})$.

If the GM is negative the system is unstable. If the GM is less than 6 to 8 dB, then it may be appropriate to consider redesigning the controller, $K(s)$, to improve the GM of the system.

Remark

We note that positive gain *and* phase margins indicate a stable closed-loop system. If *either* the gain or phase margin is negative, the closed-loop system would be unstable.

Example: Bode plot gain and phase margins

In Figure 14.29, determine the gain and phase margins.

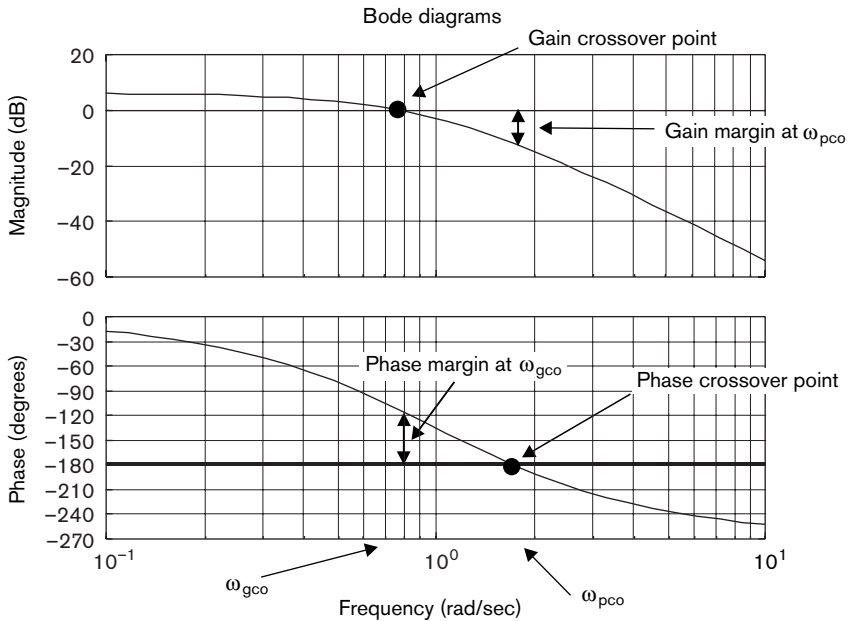


Figure 14.29 Example of gain and phase margins.

Phase margin procedure

1. ω_{gco} is at 0.8 rad/s
2. $\phi(\omega_{\text{gco}})$ is about -116°
3. $\text{PM} = -116 - (-180^\circ) = 64^\circ$

The PM is positive and therefore does not indicate an unstable closed-loop system.

Gain margin procedure

1. ω_{pco} is at 1.8 rad/s
2. $G(\omega_{pco})$ is about -12 dB
3. $GM = 0 \text{ dB} - (-12 \text{ dB}) = 12 \text{ dB}$

The GM is positive and therefore does not indicate an unstable closed-loop system.

Since *both* the gain and phase margins are positive, the closed-loop system is stable.

We can show likewise the gain and phase margins and the tests for stability using the Nyquist plots and the Nichols plot. These are illustrated below by example.

Example: Nichols plot gain and phase margins

For the Nichols plot in Figure 14.30, determine the gain and phase margins and indicate these clearly on the plot.

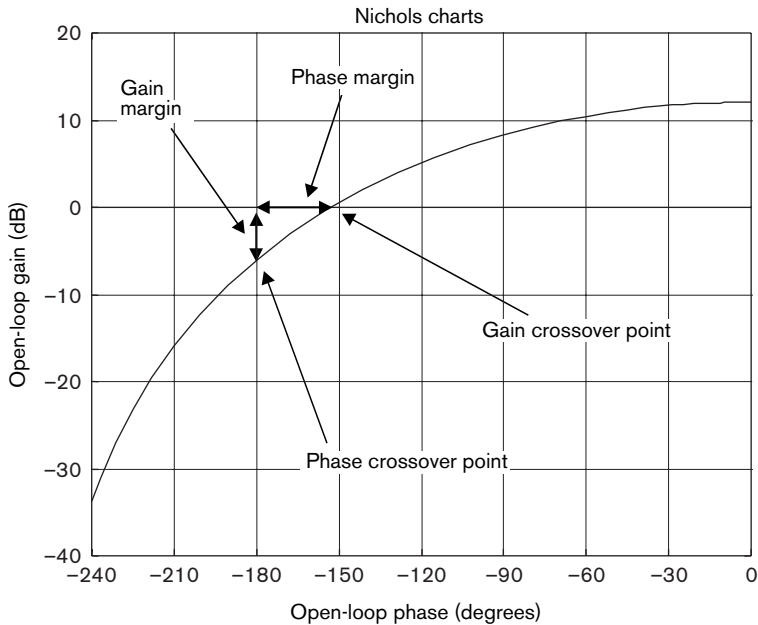


Figure 14.30 Gain and phase margin on Nichols plot.

We follow the procedure for determining the gain and phase margins. We remember three points about Nichols plots:

- there is no frequency axis
- the open-loop gain is on the vertical axis (and hence the gain margin will be read from this axis)
- the open-loop phase is on the horizontal axis (and therefore the phase margin will be read from this axis)

Phase margin procedure

1. Find the gain crossover, ω_{gco} , that is, where the magnitude response line crosses $0 \text{ dB} (= 1)$

2. Find the phase value at this point: $\phi(\omega_{gco}) = -153^\circ$
3. Calculate the phase margin: $PM = -153^\circ - (-180^\circ) = 27^\circ$

This is positive, but not a large value for a phase margin.

We can indicate the phase margin on the plot by drawing a line from the gain crossover point to the -180° phase line.

Gain margin procedure

1. Find the phase crossover, ω_{pco}
2. Determine the gain at this point: $G(\omega_{pco}) = -6$ dB
3. Calculate the gain margin: $GM = 0$ dB $- (-6$ dB) = 6 dB

This is positive, along with the PM, so we can say that the closed-loop system is stable.

We can show the gain margin on the plot by drawing a line from the phase crossover point to the 0 dB line.

Example: Nyquist plot gain and phase margins

From the Nyquist plot in Figure 14.31, determine the gain and phase margins for the system and show these clearly on the polar plot.

We remember five points about the Nyquist plot:

- there is no frequency axis: the axes are the real and imaginary parts of the complex number representation
- the phase is measured as an angle from the positive real axis
- the gain is measured as the radius from the origin
- the value of 0 dB is given by a magnitude of 1
- the value of -180° phase represents the negative real axis

Phase margin procedure

1. Find the gain crossover, ω_{gco} (this is where the frequency response line crosses the circle of radius 1)
2. Find the phase value at the gco: $\phi(\omega_{gco}) = -195^\circ$ (approx.)
3. Calculate the PM: $PM = -195^\circ - (-180^\circ) = -15^\circ$

This phase margin is negative, so the closed-loop system is unstable.

The phase margin is indicated as the angle between the negative real axis (-180°) and the line to the gco point.

Gain margin procedure:

1. Find the phase crossover, ω_{pco} (this is the point where the frequency response crosses the negative real axis)
2. Determine the gain at this point: $G(\omega_{pco}) = 1.2$ (approx.) (note that the gain is the radius from the origin, so there is no negative sign here)

3. Calculate the gain margin:

$$GM_{dB} = 0 \text{ dB} - G(\omega_{pc})_{dB} = 0 - 20 \times \log_{10}(1.2) = -1.58 \text{ dB}$$

This is negative, so the closed-loop system will be unstable.

However, in the Nyquist plot we can calculate the gain margin from the plot. Firstly, we remember from log rules that

$$20 \log_{10} a - 20 \log_{10} b = 20 \log_{10} \frac{a}{b}$$

Therefore if we use this rule we find

$$GM_{dB} = 0 \text{ dB} - G(\omega_{pc})_{dB} = 20 \log_{10} a - 20 \log_{10} b$$

where

$$20 \log_{10} a = 0 \text{ dB} \Rightarrow a = 1$$

$$20 \log_{10} b = 1.58 \Rightarrow b = 1.2$$

$$GM_{dB} = 20 \log_{10} \frac{1}{1.2}$$

$$GM \text{ (not in dB)} = \frac{a}{b} = \frac{1}{1.2} = 0.83$$

On the Nyquist plot we can show the line from the origin to the phase crossover point. The value of the gain margin is therefore 1 over this magnitude.

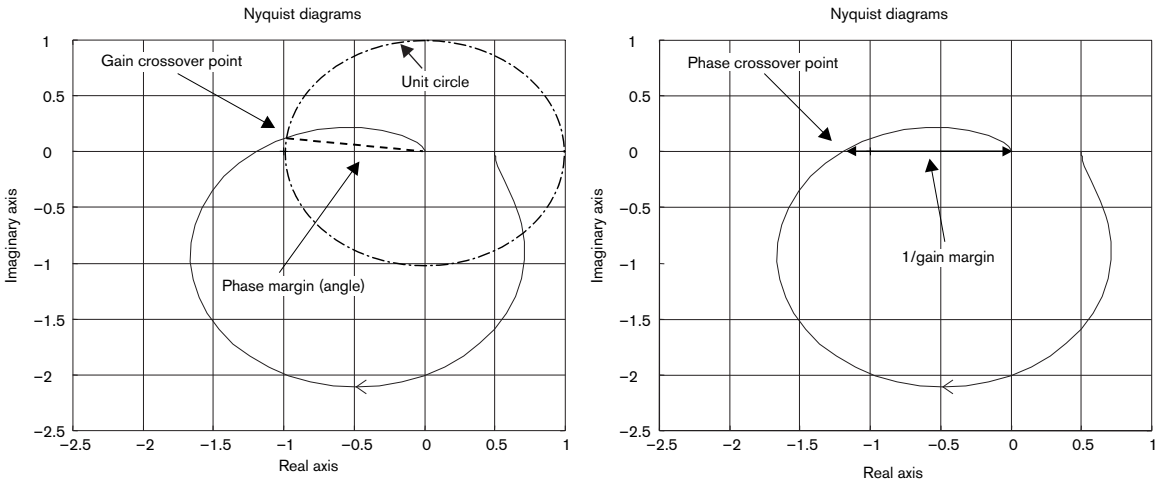


Figure 14.31 Nyquist gain and phase margins.

14.7.3 Link to the time domain

A general rule of thumb which links the phase margin specification to a time domain specification for systems with dominant second-order characteristics, is that:

$$\text{the damping ratio } \zeta = PM_{deg}/100$$

where PM_{deg} represents the phase margin in degrees. We illustrate this for the following system model, using the Bode plot.

Problem A system model has the bode plot shown in Figure 14.32.

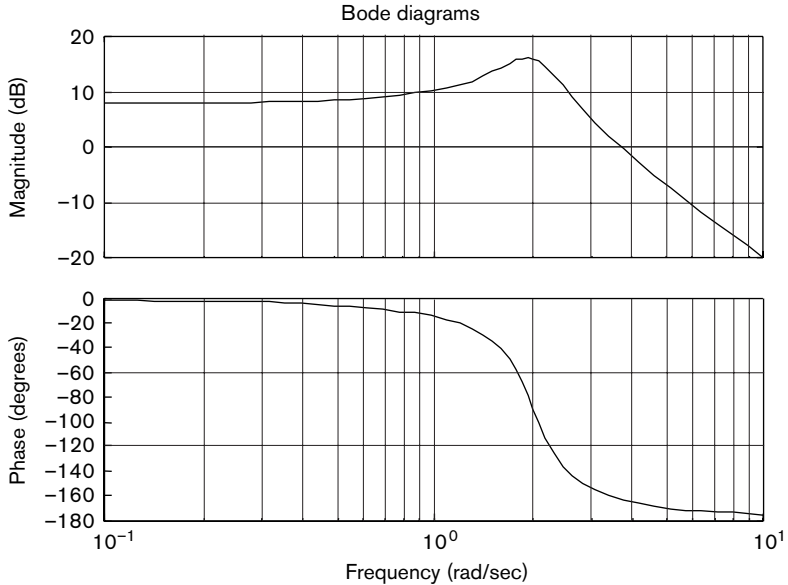


Figure 14.32 Bode plot for damping/phase margin example.

- Calculate the phase margin from the Bode plot.
- Using the rule of thumb for the damping ratio, predict what the damping ratio for the system will be.
- Verify your answer, given that the system model has the transfer function:

$$G(s) = \frac{10}{s^2 + 0.8s + 4}$$

Solution (a) The gain crossover frequency is around 3.7 rad/s. The phase at this point is approximately -163° , giving a phase margin of 17° .

- (b) The rule of thumb for the damping ratio provides:

$$\zeta = PM_{\text{deg}}/100 = 17/100 = 0.17$$

- (c) The second-order system has the standard form:

$$G(s) = \frac{K\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} = \frac{2.4 \times 4}{s^2 + 0.8s + 4}$$

Equating coefficients of powers of s on the denominator gives

$$\omega_n^2 = 4 \text{ and } 2\zeta\omega_n = 0.8$$

This gives us a value for ζ of 0.2, similar to the result from the Bode plot.

What we have learnt

- ✓ Time-varying signals were related to a frequency domain decomposition using the amplitude, phase and frequency of component sinusoids.
- ✓ Physical systems can alter the magnitude and phase of the sinusoidal components of an input signal.
- ✓ Frequency response plots can be generated from gain, phase and frequency information.
- ✓ Frequency response plots can be interpreted in terms of the low, middle and high frequency ranges.
- ✓ Some important features of a frequency response plot are: ranges of amplification and attenuation, d.c. gain, high frequency roll-off.
- ✓ The main frequency points are gain crossover, phase crossover and bandwidth.
- ✓ The frequency response plots can be related to the behaviour of the physical system.
- ✓ How the magnitude and phase of the open-loop process can determine the stability of the closed-loop system.
- ✓ Test procedures which determine, from the values of gain and phase at the phase and gain crossovers, whether the closed-loop system is stable
- ✓ Procedures for determining the **gain margin** and **phase margin** of a system. These provide an indication of *how* stable the closed-loop system is and are used as frequency domain specifications on the *open-loop system*.

Multiple choice

M14.1 What is the phase of the following signals?

- (i) $u_1(t) = 6 \sin(t + \pi/2)$
 (ii) $u_2(t) = -1.5 \sin(t - \pi/2)$
 (a) $\pi/2, -3\pi/2$
 (b) $\pi/2, -\pi/2$
 (c) $-\pi/2, -\pi/2$
 (d) $+\pi, -\pi$

M14.2 What is the equivalent gain in dB of the following magnitudes?

- (i) $|G_1| = 0.1$
 (ii) $|G_2| = +10$
 (a) $-20\text{dB}, +10\text{ dB}$
 (b) $+1\text{ dB}, +20\text{ dB}$
 (c) $-10\text{ dB}, +10\text{ dB}$
 (d) $-20\text{ dB}, +20\text{ dB}$

M14.3 A system has a gain of -20 dB . What is the equivalent magnitude value?

- (a) -0.1
 (b) 0.1
 (c) 1
 (d) -1

M14.4 A system has a phase shift of -30° at $\omega = 2\text{ rad/s}$. A sine wave at 2 rad/s and of magnitude 5 with a phase shift of $+30^\circ$ is applied to the input. What is the phase of the output sine wave?

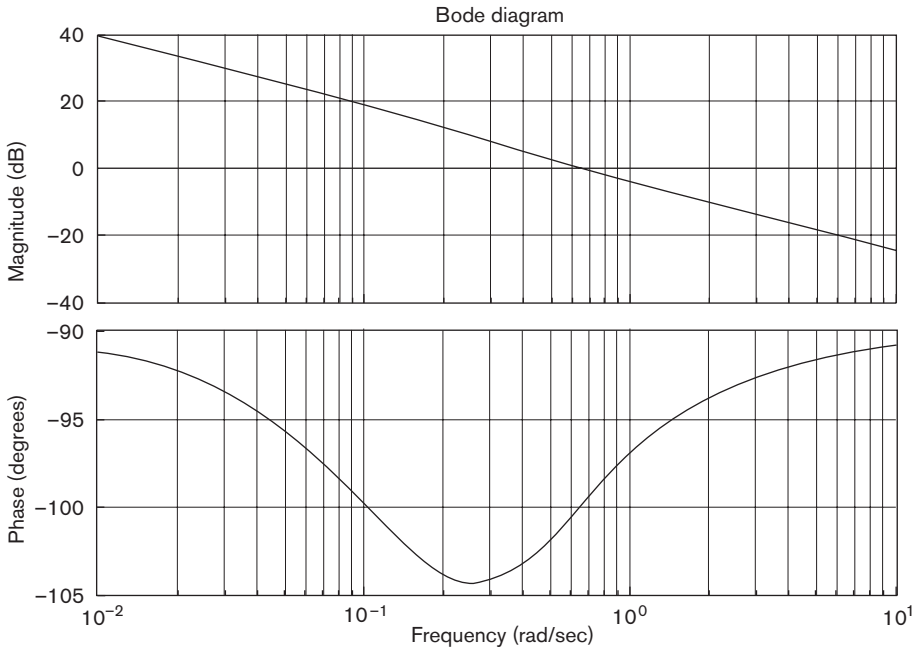
- (a) 30
 (b) -30
 (c) 60
 (d) 0

- M14.5** A Nichols plot is a plot of:
- gain vs. frequency
 - phase vs. gain
 - gain vs. phase
 - phase vs. frequency
- M14.6** Two cascaded systems have phase shifts at a frequency ω_0 of 2° and 10° respectively. What is the overall phase shift?
- 20
 - 5
 - 8
 - 12
- M14.7** If a system's gain is given as $|G(\omega_0)| = 0$ dB, what will happen to an input signal at the frequency ω_0 ?
- it will be amplified
 - it will be attenuated
 - there will be no output signal
 - the output signal will be the same magnitude as the input signal
- M14.8** The phase crossover frequency represents:
- a phase of -180°
 - the frequency when the phase crosses 0°
 - the frequency when the phase crosses -90°
 - the frequency when the phase crosses -180°
- M14.9** The roll-off rate for a system indicates:
- the spread of response
 - the steady state output
 - the high-frequency attenuation
 - the low-frequency gain
- M14.10** An open-loop transfer function $G(s)$ has a constant phase response of -180° . The system has a transfer function of the form:
- K/s
 - K/s^2
 - K/s^3
 - K/s^4

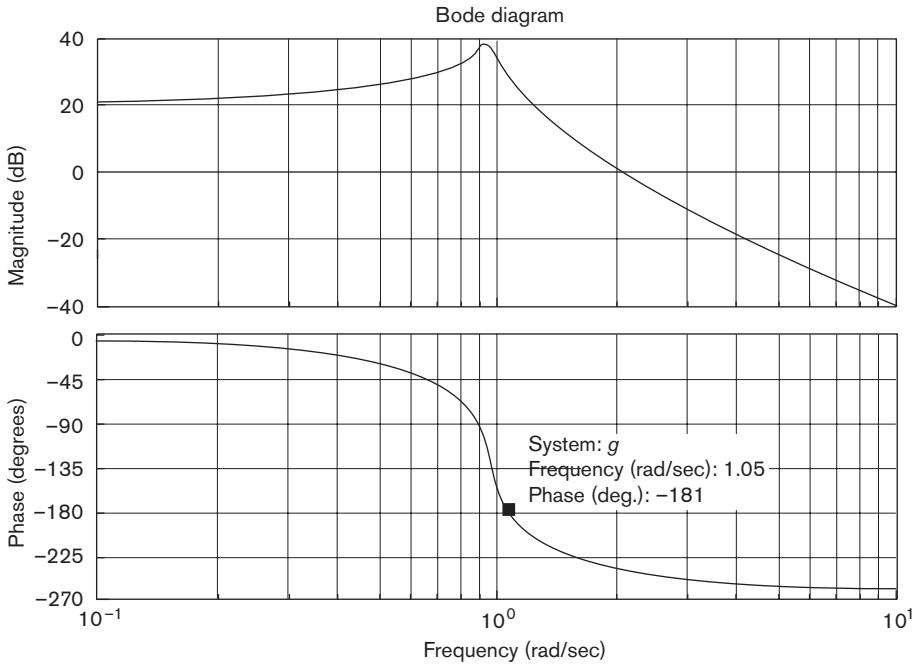
Questions: practical skills

- Q14.1** What are the frequencies, magnitudes and phases of the following signals?
- $u_1(t) = 5 \cos(3t + 0.2)$
 - $u_2(t) = 3 \cos(t - 0.3)$
 - $u_3(t) = 10 \cos(6t)$
 - $u_4(t) = 20 \cos(t - 1.2)$
- Q14.2** What are the equivalent gain values in dB for the following:
- $|G_1| = 30$
 - $|G_2| = 3$
 - $|G_3| = 0.4$
- Q14.3** The following gain values are given in dB. What are the equivalent magnitude values?
- $G_{1 \text{ dB}} = 0.707$ dB
 - $G_{2 \text{ dB}} = 20$ dB
 - $G_{3 \text{ dB}} = 6$ dB
- Q14.4** Consider the Bode plot overleaf.
- What is the d.c. gain?
 - State the frequency range where the amplification is above 10 dB.
 - At what frequency do we find 10 dB of attenuation?
 - What is the roll-off rate?
 - What is the gain when the phase reaches -100° ?

422 The frequency domain



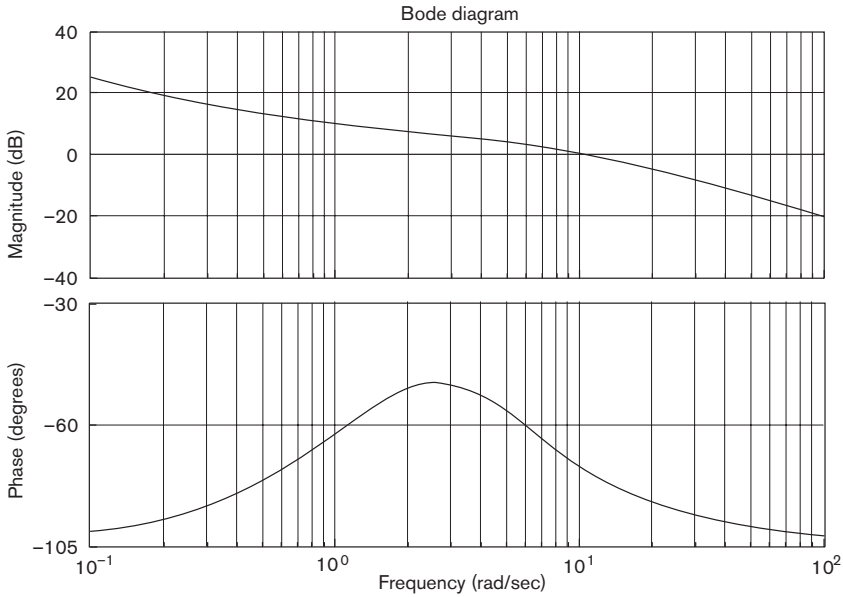
Q14.5 The Bode plot of a system is shown below.



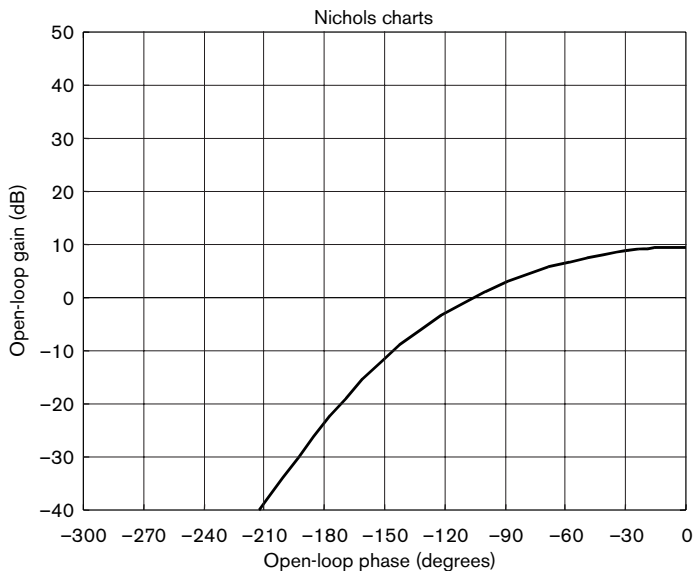
- Find the gain crossover frequency.
- Find the phase crossover frequency.
- Determine the gain and phase margins for the system and state whether the closed-loop system will be stable or unstable.

Problems

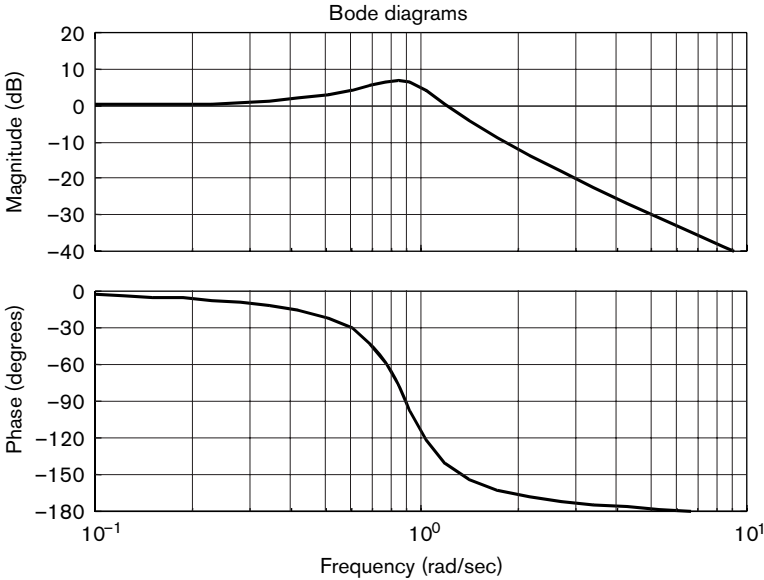
- P14.1** An engineer applies the input $u(t) = 2 \sin(t)$ to a chemical process and measures the output as $y(t) = 0.4 \sin(t - 1.55)$. What are the gain and phase of the system?
- P14.2** A current-to-pressure transmitter system has a gain of -10 dB and phase of -60° at $\omega = 10$ rad/s. An input signal of $u(t) = 5 \cos(10t - \pi/2)$ is injected. What is the output signal?
- P14.3** The Bode plot of an open-loop system is given below. Determine the stability of the system.



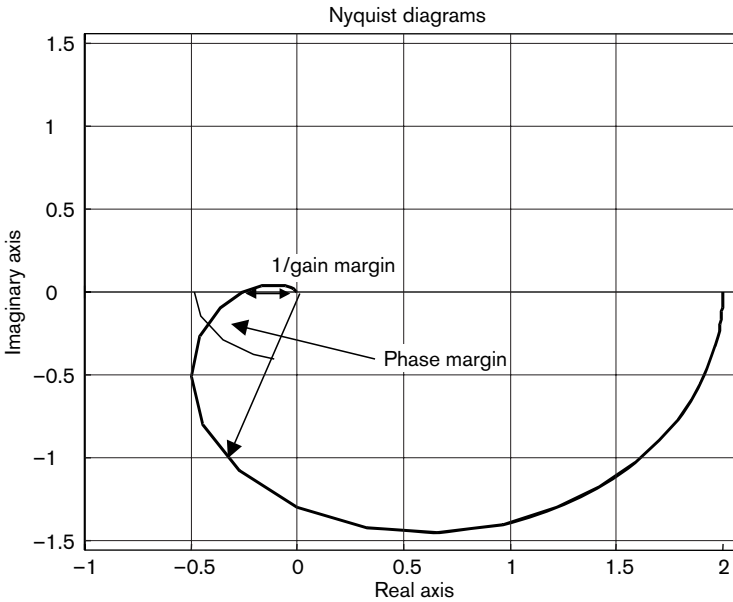
- P14.4** The Nichols plot shows the frequency response of an open-loop system. Determine the stability margins from the Nichols plot. State whether the closed loop will be stable or unstable.



P14.5 A general rule of thumb gives the damping ratio as $\zeta = PM_{deg}/100$. From the following Bode plot of a second-order system, determine the approximate damping ratio.



P14.6 Determine from the following Nyquist plot of an open-loop system whether the closed-loop system will be stable.



P14.7 Many industrial systems have transport or deadtime associated with them. The frequency responses of time-delayed systems are different from the ones we have studied so far. Consider the time-delayed system:

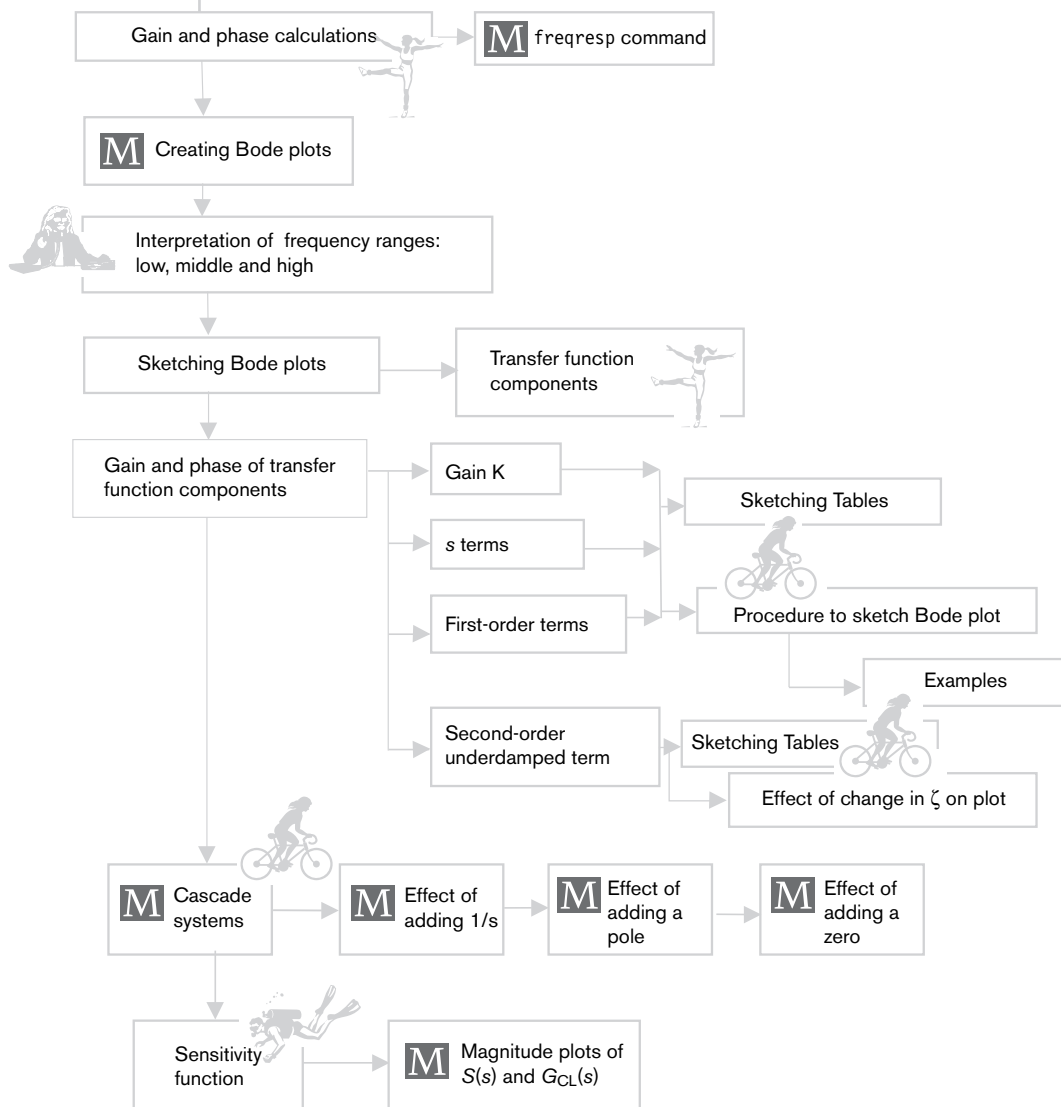
$$G(s) = \frac{e^{-5s}}{10s + 1}$$

Use the following MATLAB commands to plot the Bode plot. Notice the use of the set command to add a transport delay to the system.

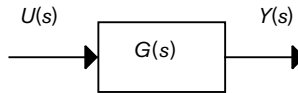
```
s=tf('s');  
g=1/(10*s+1);  
set(g,'InputDelay',5);  
bode(g)
```

Use the graph to explain how the time delay can cause instability in a system.

15 Frequency response using Bode plot presentation



The process of examining the input and output signals of a system can be performed in the *frequency domain* with the help of our Laplace transform representations of signals and systems. The system block diagram is shown below:



The use of Laplace transforms allows us to calculate $Y(s)$ by algebra:

$$Y(s) = G(s) \times U(s)$$

The frequency response of $G(s)$ shows the effect that the system, $G(s)$, has on input signals over a range of frequencies. In this chapter we look at how we represent this frequency response on a Bode plot. There are two key elements:

- (a) *To understand how a Bode plot is constructed and to practise sketching simple Bode plots by hand*

Although computers make the production of Bode plots as easy as typing 'bode', we cannot use Bode plots for design without having a clear understanding of how the different components within the transfer function affect the shape of the magnitude and phase plots. We find that if we understand the gain and phase contributions from four types of transfer function components (gain terms, s -terms, and first- and second-order components), we can easily sketch the Bode plot for many systems. To help us understand how we *add* the logged magnitudes and phases to form a Bode plot, we introduce a Sketching Table. This method formalises our approach for providing sketches of the Bode plot.

- (b) *To understand the effect on a Bode plot of the addition of other simple transfer functions; for this we will use computer-based packages*

We now assume that we have gained some familiarity with Bode plot sketching and revert to using MATLAB representations of Bode plots. In particular, we look at how adding simple lead or lag terms to a system changes the shape of the Bode plot.

Learning objectives

- To understand how the magnitude and phase points are calculated from a transfer function representation.
- To be able to enter appropriate data in order to produce a computer-based Bode plot.
- To divide the frequency range into low, middle and high frequency ranges.
- To find out the basic components that make up any simple transfer function.
- To be able to sketch a Bode plot of a simple transfer function.
- To be able to sketch a Bode plot of a cascaded system.
- To understand how adding simple components changes the shape of the Bode plot.

15.1 The Bode plot

The **Bode plot** is a graphical representation of the frequency response. It is plotted on the semilog Cartesian coordinate system; that is, a two-axis system where the horizontal axis represents a log scale ($\log_{10} \omega$) and the vertical scale represents a linear magnitude scale.

The Bode plot comprises two plots (Figure 15.1); that is:

1. a **Magnitude** plot: The **gain** of the system versus frequency
2. a **Phase** plot: The **phase** shift induced by the system versus frequency

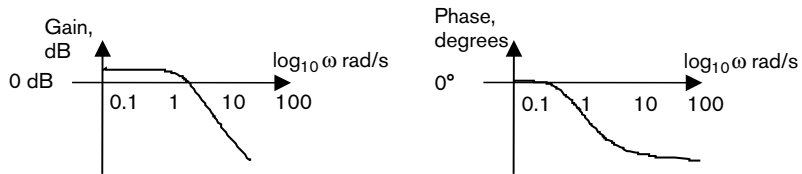


Figure 15.1 Example of a Bode plot (magnitude and phase).

Frequency axis

This is a semilog scale: the semilog plot has a log scale only on the x -axis, with a linear scale on the y -axis. The magnitude and the phase are plotted against $\log_{10} \omega$. We note that the 'log₁₀' is often omitted when labelling the frequency axis and the \log_{10} axis does not start at zero.

Magnitude axis

The gain can be plotted in terms of its actual magnitude, but is commonly plotted in units of dB, that is, $20 \log_{10}(\text{gain})$.

Phase axis

The phase axis usually has units of degrees.

15.2 Gain and phase calculation without using a computer

Given a model of the process, $G(s)$, the frequency response can be calculated by setting $s = 0 + j\omega$ and evaluating the magnitude, $|G(j\omega)|$, and the phase, $\angle G(j\omega)$, of $G(j\omega)$ over the required frequency range. Therefore the frequency response of a scalar system $G(j\omega)$ is a complex number which changes as ω varies over the range $\omega = 0$ to $\omega \rightarrow \infty$. For each value of ω in that range, $|G(j\omega)|$ and $\angle G(j\omega)$ give a gain factor and a phase shift relative to some input signal.

Problem Determine the gain and phase from the Laplace transform of the system given by:

$$G(s) = \frac{0.5}{s+0.5}$$

Solution By substituting $s = 0 + j\omega$, we find

$$G(j\omega) = \frac{0.5}{j\omega + 0.5}$$

Remove the complex number from the denominator by multiplying the numerator and denominator of $G(j\omega)$ by the complex conjugate $(0.5 - j\omega)$ of the denominator. This gives the real (Real) and imaginary (Imag) part of the complex number:

$$G(j\omega) = \frac{0.5}{j\omega + 0.5} = \frac{0.5}{j\omega + 0.5} \left(\frac{0.5 - j\omega}{0.5 - j\omega} \right) = \frac{0.5(0.5 - j\omega)}{(\omega^2 + 0.5^2)} = \frac{0.5^2}{(\omega^2 + 0.5^2)} - j \frac{0.5\omega}{(\omega^2 + 0.5^2)}$$

Now the magnitude and phase can be calculated from

$$|G(j\omega)| = \sqrt{\text{Real}^2 + \text{Imag}^2}, \quad \angle G(j\omega) = \tan^{-1} (\text{Imag}/\text{Real})$$

with $\angle G(j\omega)$ given in radians. In our example, this gives

$$|G(j\omega)| = \frac{0.5}{\sqrt{\omega^2 + 0.5^2}} \quad \text{and} \quad \angle G(j\omega) = \tan^{-1}(-\omega/0.5)$$

Skill section

Using the relations

$$|G(j\omega)| = \sqrt{\text{Real}^2 + \text{Imag}^2} \quad \text{and} \quad \angle G(j\omega) = \tan^{-1} \left(\frac{\text{Imag}}{\text{Real}} \right)$$

solve the following problem.

Problem Calculate the magnitude and phase of the system with Laplace transform

$$G(s) = \frac{0.5}{s + 0.5}$$

for frequency values of 0.2 rad/s and 1.8 rad/s.

Solution Use the expression for the magnitude and phase of $G(j\omega)$ given above.

$$|G(j0.2)| = \frac{0.5}{\sqrt{0.2^2 + 0.5^2}} = 0.928 \quad \text{and} \quad \angle G(j\omega) = \tan^{-1} \left(\frac{-0.2}{0.5} \right) = -0.385 \text{ rad} = -21.8^\circ$$

$$|G(j1.8)| = \frac{0.5}{\sqrt{1.8^2 + 0.5^2}} = 0.268 \quad \text{and} \quad \angle G(j\omega) = \tan^{-1} \left(\frac{-1.8}{0.5} \right) = -1.300 \text{ rad} = -74.5^\circ$$

Physical interpretation

Let us analyse the answers from the above example. If we inject a sinusoid with a frequency of 0.2 rad/s the system will apply a gain of 0.928 to the input signal and introduce a phase lag of -21.8° from the original signal (the negative sign indicating the phase lag). At the higher frequency, 1.8 rad/s, the system reduces the magnitude of the signal even more (a gain factor of 0.268) and the phase lag is increased to -74.5° .

We can summarise the above algebraic manipulations for calculating the gain and phase in Table 15.1.

Table 15.1 Gain and phase calculations.

Transfer function in Laplace variable s	$G(s)$
Set $s = j\omega$	$G(j\omega)$
Find real and imaginary parts of complex function $G(j\omega)$	$G(j\omega) = \text{Real}(G(j\omega)) + j \text{Imag}(G(j\omega))$
Calculate magnitude	$ G(j\omega) = \sqrt{\text{Real}^2 + \text{Imag}^2}$
Calculate phase	$\angle G(j\omega) = \tan^{-1}(\text{Imag}/\text{Real})$

M We can also use the MATLAB command, `freqresp (g, w)`. This calculates both the real and imaginary parts of the transfer function $g(s)$ at frequency w .

Example Find the magnitude and phase of the transfer function $G(s) = 0.5/(s + 0.5)$ at $\omega = 0.2$ rad/s.

```
Solution      s=tf('s');
              g=0.5/(s+0.5);
              p=freqresp(g,0.2)
              p=
                0.8621 - 0.3448 i
              mag = sqrt( real(p)^2 + imag(p)^2)
              mag =
                0.9285
              phase = atan(imag(p)/real(p))
              phase =
                -0.3805
```

Alternatively, we can use the commands: `mag = abs(p)` and `phase = angle(p)`.

15.3 Using computers to form a frequency response plot

We would like to look at the behaviour of the system over a range of frequencies. To do this we need to produce a *frequency response* plot. Although this can be performed by repeating the above calculation for the range of frequencies required, it is more commonly automated by the use of computer programs. The information you may require to pass to the computer program is:

1. a range of frequencies (usually two to four decades of frequency, such as 10^{-2} to 10^2)
2. a Laplace transform representation of the system transfer function, $G(s)$, or open-loop transfer function $G_{OL}(s)$.

15.3.1 Computer calculation of Bode plots

We will introduce four ways of producing a Bode plot in MATLAB – each of which requires a slightly different form of input data. We note that the MATLAB font does not permit the frequency symbol ω , which will appear as w in any of our MATLAB code. We also note that the default units for frequency are radians per second: rad/s. This can be altered if required to, for example, hertz, by editing the plot axes, but we shall continue to use rad/s.

Basic Bode plot

We enter a transfer function

$$G(s) = \frac{6}{(4s+1)(s+1)} = \frac{6}{4s^2 + 5s + 1}$$

either as a function of s

```
s=tf('s');
g=6/(4*s^2+5*s+1)
```

or as vectors of numerator and denominator coefficients

```
num = [6], den=[4 5 1];
g=tf([num],[den]);
```

By simply using the command

```
bode(g)
```

we will find a bode plot of $G(s)$ in the current window for a default frequency range set by the MATLAB package.

Basic Bode plot with magnitude and phase information

If we wish to find out further detailed information, such as specific magnitude and phase values, we can modify the above basic Bode command.

1. Produce a Bode plot over a specific frequency range w_{min} to w_{max}

```
bode(g, {wmin, wmax})
```

For example

```
bode(g, {0.1, 10})
```

will produce a bode plot in the range 0.1 to 10 rad/s.

2. Finding specific magnitude and phase values

```
[mag, phase, w] = bode(g)
```

This command will return the magnitude, phase and frequency values in the MATLAB *matrices* mag and $phase$, and the vector w .

The results are matrices and not vectors, since MATLAB 5.3 now does not assume that the system is a single-input, single-output system but allows greater flexibility for dealing with more complex systems. This does have an effect for us on how we address the results. If we type mag , we find a screenful of values of the form

```
mag(:, :, 28) = 0.0384
mag(:, :, 29) = 0.0239
mag(:, :, 30) = 0.0149
```

For our SISO system, this could equally well be

```
mag(1, 1, 28) = 0.0384
mag(1, 1, 29) = 0.0239
mag(1, 1, 30) = 0.0149
```

since we are dealing with the first (and only) columns of matrix mag .

We can then find particular triplets of magnitude, phase and frequency by using the same index:

```
w(10),mag(1,1,10),phase(1,1,10) or w(10),mag(10),phase(10)
```

will give the 10th frequency value in the file and the corresponding magnitude and phase values.

3. Fixing the frequency points

The default frequency points are distributed over a frequency range 0.1 to 10 rad/s. We find that the closest mid-point frequency is not 1 rad/s but 0.97 rad/s or 1.04 rad/s. By increasing the number of points, or supplying a frequency vector that includes only the specific frequency points we are interested in, we can calculate the gain and phase at a particular frequency. For example, if we supply the vector

```
w=[0.1;0.3;0.5;0.7;0.9;1.0;3.0;5.0;7.0;9.0;10];
```

and type

```
bode(g,w)
```

or

```
[mag,phase]=bode(g,w)
```

MATLAB will use the frequency vector supplied and compute the magnitude and phase at those frequency points.

4. rltool

This is really a design tool and incorporates root locus, Bode, Nyquist and Nichols plots. Hence we can use it for our Bode plot. Firstly, we type

```
rlttool
```

in the MATLAB command window.

By using the 'Import Compensator' window under the File menu we can import an existing transfer function and click on the Bode check box to produce the plot. More information is given in Chapter 13.

All four of these methods will produce the Bode plot for our system. We may find that one may be more suitable for a 'quick look', while we may use another form for further design, or another for finding details on magnitude and phase.

15.4 What are low, middle and high frequencies?

The performance of control systems depends on the behaviour and the shape of the frequency response over different frequency ranges. We usually divide the frequency axis into three ranges (Figure 15.2).

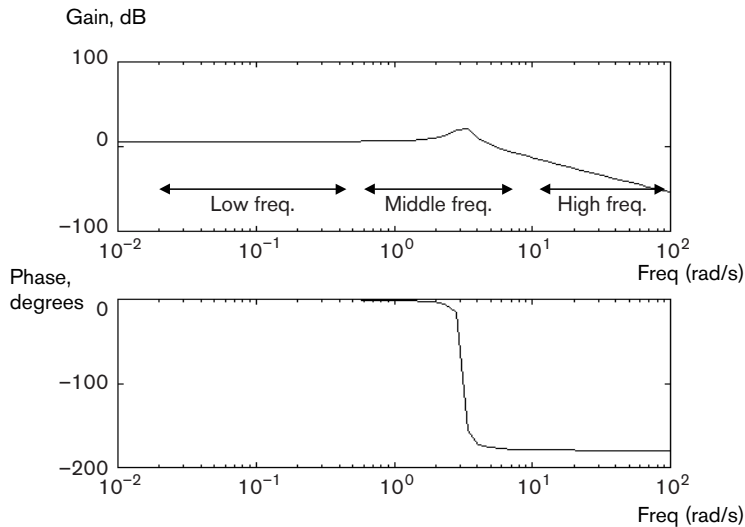


Figure 15.2 Low, middle and high frequency ranges.

We relate these ranges to the indicators of system stability and performance in later sections. These ranges are:

- *The low frequency range:* this range covers the frequencies near zero. For many systems, the gain is often 'constant' over this frequency range, or constantly decreasing.
- *The middle frequency range:* the gain and phase of the system significantly change over this frequency range. This range determines the closed-loop system stability.
- *The high frequency range:* in dB magnitude plots, the *slope* of the system gain is constant over this frequency range. This range determines the performance of the closed-loop system to high-frequency disturbances such as noise.

Problem Consider the plots shown in Figure 15.3. State the approximate frequency ranges that correspond to 'low', 'middle' and 'high' frequency ranges.

- Solution**
- *Low frequency range:* this covers the range where the amplitude and phase are fairly constant and therefore corresponds to a frequency range up until approximately 1.0 rad/s.
 - *Middle frequency range:* this range is from 1.0 rad/s to approximately 4 rad/s (where the magnitude plot slopes off at a constant rate and above the point where the gain crosses 0 dB).
 - *High frequency range:* this range corresponds to the frequencies above 4 rad/s. The magnitude plot has a constant slope and the phase is nearly steady at -180° .

Motivation for sketching Bode plots

Although we mainly use computers for plotting Bode plots and for analysing a process control system, one of the main reasons that Bode plots were used extensively for system analysis and design was the fact that, given a few rules, they were fairly easy to sketch 'on the back of an envelope'. Complex systems could be represented easily. We still use Bode plots for control system design, since they provide a representation of the frequency response plot which we can interpret relatively easily. Although we tend to use computers to plot the points on the Bode plot, practising sketching the following simple

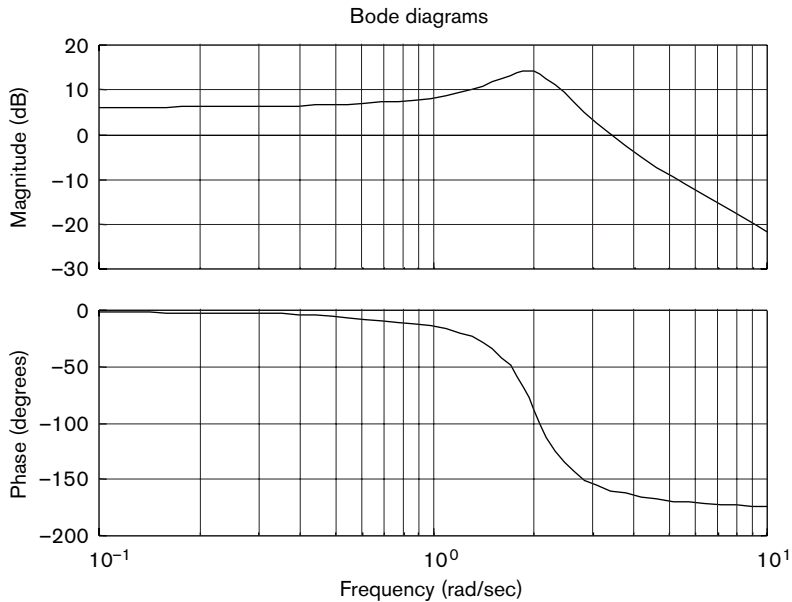


Figure 15.3 Example for different frequency ranges.

first- and second-order plots will give you a deeper understanding the features of the frequency response and how to retrieve information from the plots.

15.5 Transfer function components

We now introduce a skill section where we practise the identification of a number of simple transfer function components present in our transfer functions. These components combine together to make more complex systems.

Once these components have been identified we can form a **Sketching Table**. We find that a Bode plot is formed by the *addition of logged magnitudes* of all the components and the *addition of the phase* of all the components. We show the addition of the gain and phase in the Sketching Table.

Skill section

Transfer function components

We can write any transfer function as a product of a number of components: a number of first-order terms, second-order underdamped expressions etc. One key point to sketching a Bode plot is to recognise which terms we have and the influence they will have on the shape of the frequency response plot. Consider the following two transfer functions:

$$G_1(s) = \frac{20s(3s+1)}{(10s+1)(s^2+0.2s+1)} \quad \text{and} \quad G_2(s) = \frac{6(s^2+10s+100)}{s^2(50s^2+15s+1)}$$

We can see that each transfer function is comprised of a number of components: gain, first-order terms and second-order terms.

We look for the following four terms (Table 15.2) in the transfer function and note whether they appear in the numerator or denominator.

Table 15.2 Transfer function components.

System gain K

Terms in s^p with $p > 0$ (numerator term) or $p < 0$ (denominator term)

First-order terms of form $(\tau s + 1)$

Second-order terms: $\left(\frac{1}{\omega_n^2} s^2 + \frac{2\zeta}{\omega_n} s + 1 \right)$

Sometimes we see terms in the transfer functions such as s^2 or we find that the *basic* component has been raised to a power, p . In these examples we would have $s^2 = s^p$ ($p = 2$) or $1/(s + 1)^3 = (s + 1)^p$ ($p = -3$). It is useful for us to determine the different components, since when we come to examining the Bode plots we will find that each term ‘adds’ its effect to the existing terms.

Finding the Bode plot components

1. Firstly ensure that the transfer function is in ‘unity constant coefficient form’ (Chapter 7). This enables us to read off the system gain K from the transfer function.
2. Take any second-order component $(as^2 + bs + 1)$ and by finding out whether $b^2 - 4ac$ (with $c = 1$ in this form) is positive or negative, determine whether the roots are complex or real.
3. If the roots are real, determine the values and rewrite the second-order expression as two first-order terms, else leave the term as a second-order term.
4. List all the first- and second-order components.

Problem List the transfer function components for $G_1(s)$:

$$G_1(s) = \frac{20s(3s + 1)}{(10s + 1)(s^2 + 0.2s + 1)}$$

- Solution**
1. The transfer function is already in unity constant coefficient form. We can read off the gain K as 20.
 2. From the denominator term $s^2 + 0.2s + 1$, the value of $b^2 - 4ac = 0.04 - 4 = -3.96 < 0$. The roots are complex.
 3. No real roots of second-order components.
 4. The resulting list is:

$$20, s^1, (10s + 1)^{-1}, (3s + 1)^1, (s^2 + 0.2s + 1)^{-1}$$

Ordering the transfer function components

We find that it is convenient to have the components in the order of gain terms, s^p terms, and then first- and second-order terms in order of their *corner frequency*, where we define the corner frequency for first- and second-order terms as follows:

First-order term: $\tau s + 1$: Corner frequency $\omega_c = 1/\tau$

Second-order term: $s^2 + 2\zeta\omega_n s + \omega_n^2$: Corner frequency $\omega_c = \omega_n$

The corner frequency is the point at which the low- and high-frequency approximations to the gain of first (or second) order terms meet. It is discussed more fully in Section 15.7.1. The reason for the ordering of components will become apparent when we combine components together.

Problem For $G_2(s)$, list the transfer function components in order of increasing corner frequency.

$$G_2(s) = \frac{6(s^2 + 10s + 100)}{s^2(50s^2 + 15s + 1)}$$

Solution 1. The transfer function is not in unity constant coefficient form. By altering the second-order term on the numerator we find:

$$G_2(s) = \frac{600(0.01s^2 + 0.1s + 1)}{s^2(50s^2 + 15s + 1)}$$

Now we can read off the gain K as 600.

2. There are two second-order terms:

For the numerator term $(0.01s^2 + 0.1s + 1)$:

the value of $b^2 - 4ac = 0.01 - 0.04 < 0$. The roots are complex.

For the denominator term $(50s^2 + 15s + 1)$:

the value of $b^2 - 4ac = 225 - 200 = 25 > 0$. The roots are real.

3. The second-order component with real roots can be written as

$$(50s^2 + 15s + 1) = (5s + 1)(10s + 1)$$

4. The resulting list is

$$600, (0.01s^2 + 0.1s + 1)^1, (5s + 1)^{-1}, (10s + 1)^{-1}$$

We now have to order the components. We list their corner frequencies:

$$(0.01s^2 + 0.1s + 1): \quad \omega_c = \omega_n = 1 \text{ rad/s}$$

$$\frac{1}{5s + 1}: \quad \omega_c = 1/\tau = 1/5 = 0.2 \text{ rad/s}$$

$$\frac{1}{10s + 1}: \quad \omega_c = 1/\tau = 1/10 = 0.1 \text{ rad/s}$$

In order of increasing corner frequency, with the gain term first, we have

$$600, (10s + 1)^{-1}, (5s + 1)^{-1}, (0.01s^2 + 0.1s + 1)^1$$

Now that we have identified and ordered the different terms that might appear in a system transfer function, we would like to find a general method of evaluating the magnitude and phase change that each simple component introduces into the Bode plot.

15.6 Magnitude and phase of transfer function components

We start with a reminder of the rules of logarithms and calculation of phase which apply when we multiply or divide transfer function components.

Logarithm rules

Multiplication of a and b : $\log ab = \log a + \log b$

Division of a and b : $\log a/b = \log a - \log b$

Powers of a : $\log a^b = b \log a$

The reason we need these rules is that the magnitude on a Bode plot is written in terms of $20 \log_{10} |G(j\omega)|$. Since we are effectively multiplying the terms together in the transfer function, it turns out that we will be adding them as logarithmic values.

Calculation of phase from complex variables

We remember three points for use in the calculation of phase from complex variables.

1. To calculate the phase of a complex variable $x = a + bj$, we form $\angle x = \tan^{-1} b/a$, that is \tan^{-1} (Imaginary/Real).

2. To calculate the phase of a variable with numerator and denominator, $x = (a + bj)/(c + dj)$, we form:

$$\angle x = \angle \text{numerator term} - \angle \text{denominator term}$$

$$\angle x = \tan^{-1}(b/a) - \tan^{-1}(d/c)$$

3. To calculate the phase of a product, $x = (a + bj) \times (c + dj)$, we form:

$$\angle x = \angle \text{first term} + \angle \text{second term}$$

$$\angle x = \tan^{-1}(b/a) + \tan^{-1}(d/c)$$

We now examine each of the four basic system components to find out their magnitude and phase response.

15.6.1 Magnitude and phase of constant gain K **System gain K (magnitude)**

The system gain is given by the real constant K . The gain on the Bode plot is given by $20 \log_{10} K$. The value does not change with frequency, and hence this is represented by a straight horizontal line over all frequencies.

System gain K (phase)

The phase of any real positive constant is 0° . This again does not change with frequency, so the multiplication of the transfer function by a positive constant will not alter the phase.

15.6.2 Magnitude and phase of s -terms**Magnitude of s -term**

Let $G(s) = 1/s$, where $s = \sigma + j\omega$. This is an example of $G(s) = s^p$ where $p = -1$. To find the frequency response, we set $s = j\omega$. This gives:

$$G(j\omega) = \frac{1}{j\omega} \quad |G(j\omega)| = \frac{1}{|j\omega|} = \frac{1}{\omega}$$

The magnitude of $|G(j\omega)|_{\text{dB}} = 20 \log_{10}(1/\omega) = 20 \log_{10}(1) - 20 \log_{10}(\omega) = -20 \log_{10}(\omega)$. We can calculate the value of gain for different frequency points over several decades (Table 15.3).

Table 15.3 Magnitude value for $G(s) = 1/s$.

ω	$\log_{10}(\omega)$	$ G(j\omega) _{dB} = -20 \log_{10}(\omega)$
0.01 rad/s	-2	40 dB
0.1 rad/s	-1	20 dB
1 rad/s	0	0 dB
10 rad/s	1	-20 dB
100 rad/s	2	-40 dB

The graph of $|G(j\omega)|_{dB}$ plotted against $\log_{10}(\omega)$ is given in Figure 15.4 as the line for $1/s$. It shows that the magnitude decreases with a constant rate given by -20 dB/decade. We can find the magnitude of the general term s^p by following the same route.

Magnitude of s^p -term

Let $G(j\omega) = (j\omega)^p$. The magnitude of $G(j\omega)$ is given by $|G(j\omega)| = \omega^p$. If we express this in dB, we find

$$|G(j\omega)|_{dB} = 20 \log_{10}\omega^p = p \times 20 \log_{10}(\omega)$$

In the previous example, where the s -term was $1/s$, the value of p is equal to -1 .

By plotting $|G(j\omega)|_{dB}$ against frequency we find that the magnitude plot of s^p is given by a line passing through the value 0 dB at 1 rad/s with a slope of $20 \times p$ dB/decade.

If p is positive, the s -term is in the numerator and the slope is positive.

If p is negative, the s -term is in the denominator and the slope is negative.

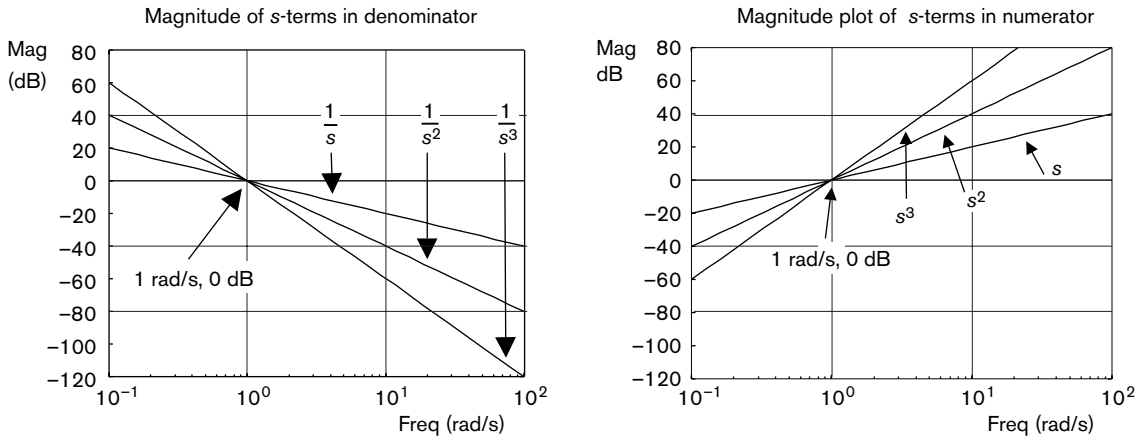


Figure 15.4 Magnitude plot of s -terms in denominator and numerator.

Phase of s -term

Let us consider the phase of $G(s) = s$. Replacing $s = j\omega$ in $G(s)$ gives $\angle G(j\omega) = \angle j\omega$.

1. s -term on numerator: $\angle G(j\omega) = \angle (j\omega) = 90^\circ$, since the phase of a positive imaginary number is 90° .
2. s -term on denominator = $\angle G(j\omega) = \angle (1/j\omega) = \angle (1) - \angle (j\omega) = 0 - 90^\circ = -90^\circ$

Phase of s^p -term

We now consider the general case of s^p .

$$s^p\text{-term: } \angle G(j\omega) = \angle (j\omega)^p = \angle (j\omega) + \angle (j\omega) + \angle (j\omega) + \dots (p \text{ times}) = p \times 90^\circ$$

If the s -term were in the numerator then p would be positive:

$$s^2 \text{ would provide } 2 \times 90^\circ = 180^\circ \text{ phase shift}$$

If the s -term were in the denominator then p would be negative:

$$1/s^3: p = -3 \text{ and the transfer function would provide } -3 \times 90^\circ = -270^\circ \text{ phase change}$$

We find that the phase of the s -term is equal to 90° times p . On a phase plot this is drawn as a horizontal straight line over all frequencies.

15.7 Introducing a sketching table

We now combine the first two terms we have learnt about: the system gain K and the s -terms. When we combine terms, we can form a quick sketch by considering their low-, mid- and high-frequency values. We are particularly interested in the straight line approximation at low frequency, called the low-frequency asymptote, the shape of the plot in the mid-range and the roll-off rate at high frequency. We can add the phase components together, and, because we use magnitude in dB, that is \log_{10} , we can add the magnitude components as well. We can complete a *sketching table*, as shown in the following example.

Problem Draw the Bode plot for the following transfer function

$$G(s) = 6/s$$

Solution Firstly we list the component terms in the transfer function:

$$6, 1/s$$

We consider the following table:

Sketching table (magnitude)			
Component	Low-frequency asymptote	Mid-frequency values	High-frequency asymptote
$K = 6$	$20 \log_{10} 6 = 15.56 \text{ dB}$	$20 \log_{10} 6$	0 dB/decade (horizontal line: no roll-off)
$1/s$	line at -20 dB/decade passing through 1 rad/s at 0 dB	0 dB at 1 rad/s	-20 dB/decade
Total	Low-frequency asymptote: line at -20 dB/decade passing 1 rad/s at $(0 + 20 \log_{10} 6 \text{ dB})$	Mid-frequency at 1 rad/s , $ G = 20 \log_{10} 6 = 15.56 \text{ dB}$	High-frequency roll-off: sum of all roll-off rates: $0 - 20 \text{ dB/decade} = -20 \text{ dB/decade}$

This plot is shown in Figure 15.5. The line for $1/s$ is drawn and the constant system gain merely raises the whole line up by 15.56 dB.

Phase calculation

Sketching table (phase)			
Component	Low-frequency asymptote	Mid-frequency values	High-frequency asymptote
$K = 6$	0°	0°	0°
$1/s$	-90°	-90°	-90°
Total	-90°	-90°	-90°

The phase contributed by the $1/s$ term is $p \times 90^\circ$. In this case, $p = -1$, so the phase is -90° over all frequencies. The constant gain K has 0° phase over all frequencies and therefore has no effect on the phase plot. The result is a straight line at -90° , as shown in Figure 15.5.

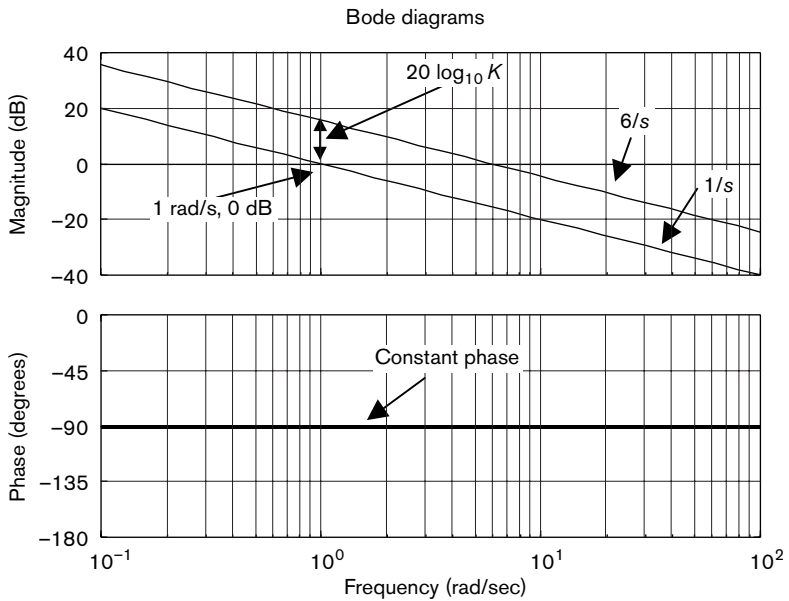


Figure 15.5 Bode plot of $6/s$.

15.7.1 Magnitude and phase of first-order terms

Magnitude of first-order terms

We now turn our attention to the first-order terms of the form:

$$G_1(s) = \frac{1}{(\tau s + 1)} \quad \text{and} \quad G_2(s) = (\tau s + 1)$$

We note the following result for $G_1(j\omega)$ and $G_2(j\omega)$:

Key result: Magnitude and phase of first-order terms

The logged magnitude of $G_1(j\omega) = \frac{1}{(\tau j\omega + 1)}$ is the negative of $G_2(j\omega) = (\tau j\omega + 1)$

The phase of $\angle G_1(j\omega) = \angle \frac{1}{(\tau j\omega + 1)}$ is the negative of $\angle G_2(j\omega) = \angle (\tau j\omega + 1)$

These key results are easy to prove:

Magnitude:

We note that $|G_1(j\omega)| = 1/|G_2(j\omega)| = |G_2(j\omega)|^{-1}$. In logs, we find:

$$|G_1(j\omega)|_{\text{dB}} = 20 \log_{10}(|G_2(j\omega)|^{-1}) = -20 \log_{10}(|G_2(j\omega)|) = -|G_2(j\omega)|_{\text{dB}}$$

Phase:

We note that

$$\angle G_1(j\omega) = \angle (1/(1 + j\tau\omega)) = \angle 1 - \angle (1 + j\tau\omega) = 0 - \angle (1 + j\tau\omega) = -\angle G_2(j\omega)$$

This will simplify matters in our analysis. We will examine only $G_1(s) = 1/(\tau s + 1)$, and apply the negation to the results to find the corresponding results for $G_2(s)$.

The following table shows the magnitude and magnitude(dB) calculations for $G_1(s)$:

Transfer function	$G_1(s) = \frac{1}{\tau s + 1}$
Set $s = j\omega$	$G_1(j\omega) = \frac{1}{j\omega\tau + 1}$
Magnitude of $G(j\omega)$	$ G_1(j\omega) = \frac{1}{ j\tau\omega + 1 } = \frac{1}{\sqrt{\tau^2\omega^2 + 1}}$
Gain in dB: $ G(j\omega) _{\text{dB}}$	$ G_1(j\omega) _{\text{dB}} = 20 \log_{10} (\tau^2\omega^2 + 1)^{-0.5}$

To make it easier to analyse the system gain, we examine the gain response over the different frequency ranges (low, middle and high).

Low-frequency and high-frequency asymptotes

We refer to the straight line at low frequency as the *low-frequency asymptote* (Figures 15.6 and 15.7) of the system. Likewise the line with constant slope at high frequency is referred to as the *high-frequency asymptote*. The high-frequency asymptote will define the *roll-off rate* (Chapter 14) of the system. Thus for $G_1(s)$ this is -20 dB/decade, and for $G_2(s)$ it is $+20$ dB/decade. The low-frequency asymptote and the high-frequency asymptote intersect at $\omega_c = 1/\tau$. We call this frequency *the corner frequency* of the system (Figures 15.6 and 15.7).

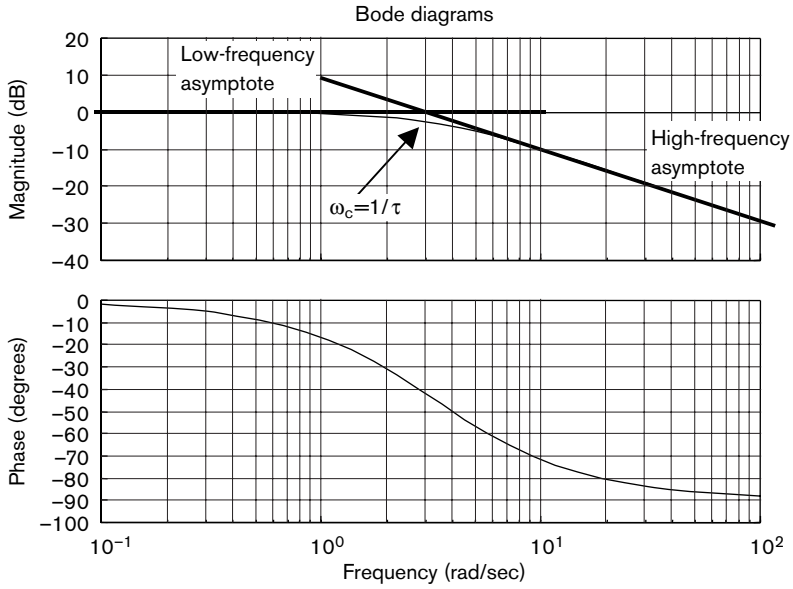


Figure 15.6 Bode plot of $1/(\tau s + 1)$.

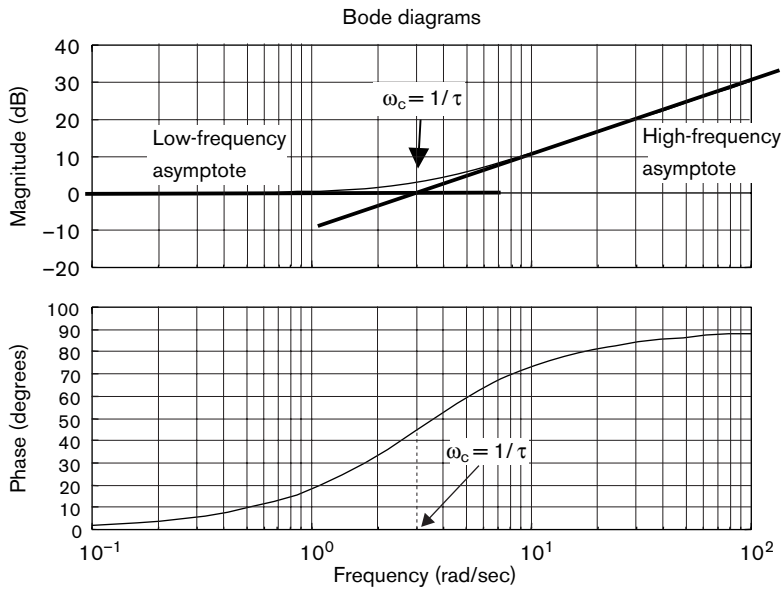


Figure 15.7 Bode plot of $(\tau s + 1)$.

Table 15.4 gives a detailed calculation of the magnitude of $G_1(s)$ in the three frequency ranges.

Table 15.4 First-order system gain calculation.

Gain in dB: $ G(j\omega) _{dB}$	$ G_1(j\omega) _{dB} = 20 \log_{10}(\tau^2\omega^2 + 1)^{-0.5}$
Low-frequency range $\omega \approx 0$	$ G_1(j\omega) _{dB} \rightarrow 20 \log_{10}(1) = 0 \text{ dB}$ horizontal line at 0 dB
Mid-frequency range $\omega = 1/\tau$	$ G_1(j\omega) _{dB} \rightarrow 20 \log_{10}(1 + 1)^{-0.5}$ $= -20 \log_{10}(2)^{0.5} = -10 \log_{10}(2)$ $= -3 \text{ dB}$ point 3 dB below 0 dB line
High-frequency range $\omega \gg 1$	$ G_1(j\omega) _{dB} \approx 20 \log_{10}(\tau^2\omega^2)^{-0.5}$ $= -20 \log_{10}\sqrt{\tau^2\omega^2}$ $= -20 \log_{10}\tau\omega$ $= -20 \log_{10}\tau - 20 \log_{10}\omega$ Line with slope of: -20 dB/decade (found by differentiating above with respect to $\log_{10}\omega$) Line intersects the horizontal 0 dB line at $\omega_c = 1/\tau$

First-order system phase plot

We can calculate the system phase shift for $G_1(s)$.

Transfer function	$G_1(s) = \frac{1}{\tau s + 1}$
Set $s = j\omega$	$G_1(j\omega) = \frac{1}{j\omega\tau + 1}$
$\angle G(j\omega)$	$\angle G_1(j\omega) = \angle (1/(1 + j\tau\omega))$ $= \angle 1 - \angle (1 + j\tau\omega)$ $= 0 - \tan^{-1}(\tau\omega)$ $= -\tan^{-1}(\tau\omega)$

Once again, we examine the results over the different frequency ranges (low, middle and high) – Table 15.5.

Table 15.5 First-order system phase calculation.

$\angle G(j\omega)$	$\angle G_1(j\omega) = -\tan^{-1}(\tau\omega)$
Low-frequency range $\omega \approx 0$	$\angle G_1(j\omega) \approx -\tan^{-1}(0) \approx 0$ The phase at low frequency is approximately zero
Mid-frequency range $\omega = 1/\tau$	$\angle G_1(j\omega) = -\tan^{-1}(1) = -45^\circ$ The phase at the corner frequency $\omega_c = 1/\tau$, is -45°
High-frequency range ($\omega \gg 1$)	$\angle G_1(j\omega) = -\tan^{-1}(\infty) = -90^\circ$ The phase at high frequency equals -90°

Note that the phase at the corner frequency is exactly half the phase at high frequencies, as shown in Figure 15.6 and 15.7.

Remark

In practice it is often taken that the phase changes over two decades, from one decade below the corner frequency, $\omega_c/10$ to one decade above, $10 \times \omega_c$.

Lead and lag terms

The terms $1/(\tau s + 1)$ and $(\tau s + 1)$ commonly appear in transfer functions. We have seen that the term $1/(\tau s + 1)$ introduces -90° of phase lag. This transfer function is therefore referred to as a first-order lag: 'first-order' since it represents a first-order system or component, from the power of s in the denominator, and 'lag' due to the 90° of phase lag that it introduces. Similarly, since the term $(\tau s + 1)$ introduces $+90^\circ$ of phase change, it is referred to as a first-order lead term. These properties will be used in Chapter 16, when we design lag and lead compensators to provide a required amount of phase change.

Procedure to sketch Bode plot

- Step 1:* Identify all the transfer function components and list them in terms of gain, s -terms, and then first- and second-order terms in order of increasing corner frequencies.
- Step 2:* Choose the frequency range of interest, often three (or four) decades wide, (0.1 to 100 rad/s), centred around the region of interest: for first-order systems, centred around the corner frequency. If there is more than one corner frequency then the frequency range should enclose all corner frequencies.
- Step 3:* Complete a sketching table for the range of frequencies chosen.
- Step 4:* Find the low-frequency asymptote from the sketching table. The value of the asymptote will depend on whether there are any s -terms in the transfer function.
- (i) If there are no s -terms, then the low-frequency asymptote will be a horizontal line with d.c. gain of $20 \log_{10} K$, where K is the transfer function gain.
 - (ii) If there are s -terms, then the low-frequency asymptote is formed by the logged sum of the slope of these lines, which pass nominally through 0 dB at 1 rad/s and are then raised by $20 \log_{10} K$.
- Step 5:* If there is more than one corner frequency, put them in order of numerically increasing frequency values. Find the (first) corner frequency, $\omega_c = 1/\tau$. Find the corresponding point on the low-frequency asymptote.
- Step 6:* Draw the high-frequency asymptote. This is a straight line of slope -20 dB/decade for lag terms starting from the point $(\omega_c, 20 \log_{10} K)$, and a slope of $+20$ dB/decade for lead terms.
- Step 7:* Find the next corner frequency and the plot will then decrease (or increase) at an extra -20 dB/decade or $+20$ dB/decade for lag and lead terms respectively.
- Step 8:* Plot the phase response using the phase shift values at different frequency ranges.

15.8 Elementary examples

We illustrate the use of sketching tables by examining several examples which are very common in process control systems, all of which involve a combination of the three basic terms: gain K , s -terms and first-order terms.

Example 1: Bode plot of gain and first-order lag

Problem Sketch the Bode plot for the following transfer function:

$$G(s) = \frac{20}{4s+1}$$

Solution First we list the transfer function component terms:

$$20, \frac{1}{4s+1}$$

We identify the corner frequencies, as these will usually occur in our mid-frequency range. In the above transfer function, there is only one corner frequency and it is at $\omega_c = 1/4 = 0.25$ rad/s. We can then complete the sketching table.

Sketching table (magnitude)

Component	Low-frequency asymptote	Mid-frequency value: 0.25 rad/s	High-frequency roll-off
$K = 20$	$20 \log_{10}(20) = 26.02$ dB	26.02 dB	0 dB/decade
$\frac{1}{4s+1}$	0 dB	-3 dB	-20 dB/decade
Total	26.02 dB	23.02 dB	-20 dB/decade

Sketching table (phase)

Component	Low-frequency values $\omega < 0.025$ rad/s	Mid-frequency value: 0.25 rad/s	High-frequency values $\omega > 2.5$ rad/s
$K = 20$	0°	0°	0°
$\frac{1}{4s+1}$	0°	$\tan^{-1}4\omega = -45^\circ$	-90°
Total	0°	-45°	-90°

We make two remarks about the phase table. Firstly, we note again that the constant gain K has no effect on the phase at all, and secondly, that we consider 'low frequency' to be a decade below the (lowest) corner frequency and 'high frequency' to be a decade above the (highest) corner frequency.

We can then sketch the Bode plot using the information in the tables.

Figure 15.8 shows the low- and high-frequency asymptotes as well as the magnitude and phase at the corner frequency. We have used MATLAB to plot the Bode plot in dashed lines to

illustrate how good the sketching approximation is. (In MATLAB we used the command: `bode(g, {0.01,10})`).

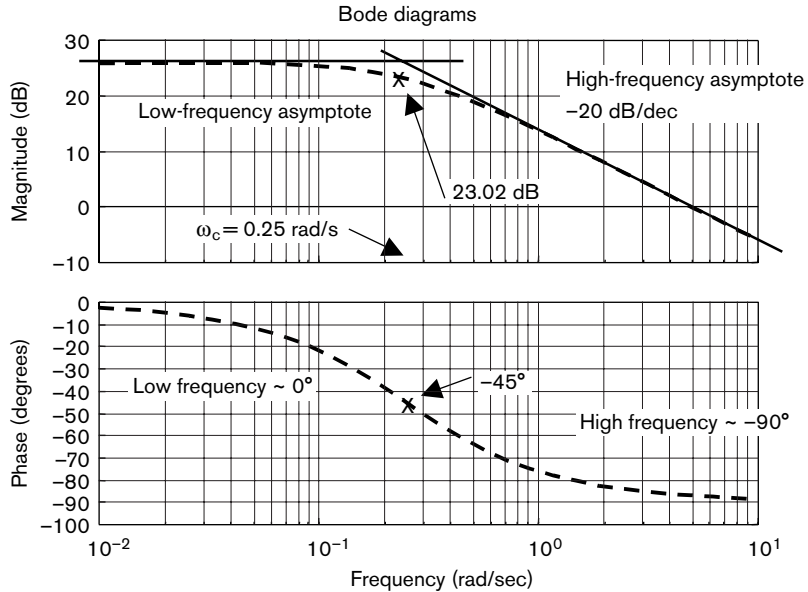


Figure 15.8 Bode plot of $20/(4s + 1)$.

Example 2: Finding a simple transfer function from a Bode plot

Just as we have used a transfer function description to sketch a Bode plot, we can use a Bode plot to help us find out the system transfer function. This is particularly useful when we have some experimental results from a process that we have taken by injecting sinusoids at different frequencies, and we wish to find a transfer function for the system.

Problem An engineer tries to find the time constant of an RC network by applying unit magnitude sine waves of different frequencies and zero phase at the input of the RC circuit. The engineer measures the magnitude in dB and phase of the output voltage, $y(t)$, across the capacitor and sets up the table shown below:

The output data from the RC network									
ω (rad/s)	0.01	0.05	0.1	0.5	1.0	5.0	10.0	60.0	100.0
$ Y(j\omega) _{dB}$	0.00	-0.3	-0.97	-8.6	-14	-28	-34	-49	-54
$\angle Y(j\omega)$	-3	-14	-26	-68	-79	-87	-88	-89	-90

- (a) Draw the Bode plot for the system.
- (b) Draw the low-frequency and high-frequency asymptotes.
- (c) Find the time constant of the system.

Solution (a) $|Y(j\omega)| = |G(j\omega)| \times |U(j\omega)|$

$$\angle Y(j\omega) = \angle G(j\omega) + \angle U(j\omega)$$

Since $|U(j\omega)| = 1$ and $\angle U(j\omega) = 0$, $|Y(j\omega)| = |G(j\omega)|$ and $\angle Y(j\omega) = \angle G(j\omega)$

Thus, the system has a frequency response $G(j\omega)$ similar to the voltage signal across the capacitor $Y(j\omega)$. To draw the Bode plot, transfer the data at each frequency onto the semilog graph paper (Figure 15.9).

MATLAB commands:

```
w=[0.01 0.05 0.1 0.5 1.0 5.0 10.0 60.0 100.0];
mag=[0 -0.3 -0.97 -8.6 -14 -28 -34 -49 -54];
ph=[-3 -14 -26 -68 -79 -87 -88 -89 -90];
subplot(2,1,1);semilogx(w,mag,+,)
subplot(2,1,2);semilogx(w,ph,+,)
```

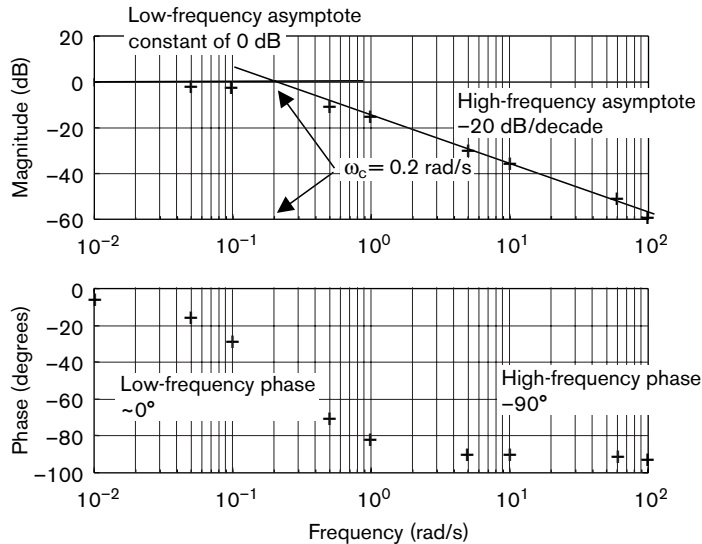


Figure 15.9 Data plotted on semilog paper.

(b) The RC network is a first-order system. The features on the Bode plot which verify this are:

(i) Roll-off rate

Find the slope of the gain response at high frequencies by selecting two frequencies which are one decade apart, for example, $\omega = 1$ and $\omega = 10$. Subtracting the gain at $\omega = 1$ (-14 dB) from the gain at $\omega = 10$ (-34 dB), gives the slope as -20 dB/decade as the roll-off rate. This is the roll-off rate for a first-order system.

(ii) Phase change

The phase change is from 0° to -90° , implying a possible first-order lag term.

(iii) The low-frequency asymptote is a constant horizontal line which implies there are no s-terms in the transfer function, and which therefore gives us the gain value.

The horizontal gain asymptote is at 0 dB = $20 \log_{10} K$ dB (equivalent to a gain $K = 1$).

(c) The corner frequency is at 0.2 rad/s from the Bode plot. The time constant is given by:

$$\tau = 1/0.2 = 5.0 \text{ seconds}$$

The transfer function of the RC circuit is therefore

$$G(s) = \frac{1}{5s+1}$$

Example 3: Combination of integrator and first-order lag

Problem Sketch the Bode plot for the following transfer function:

$$G(s) = \frac{10}{s(s+3)}$$

Solution To list the components, we need the transfer function in 'unity constant coefficient form'. Therefore we rewrite the transfer function by dividing the numerator and denominator by 3.

$$G(s) = \frac{3.33}{s(0.33s+1)}$$

The components are now:

$$3.33, \frac{1}{s}, \frac{1}{0.33s+1}$$

The corner frequency of the first-order lag is $\omega_c = 3 \text{ rad/s}$. We can now complete the sketching table.

Sketching table (magnitude)			
Component	Low-frequency asymptote	Mid-frequency values: 3 rad/s	High-frequency roll-off
$K = 3.33$	$20 \log_{10}(3.33) = 10.45 \text{ dB}$	10.45 dB	0 dB/decade
$\frac{1}{s}$	Line decreasing at -20 dB/decade Passes 1 rad/s at 0 dB	Line decreasing at -20 dB/decade	-20 dB/decade
$\frac{1}{0.33s+1}$	0 dB	-3 dB	-20 dB/decade
Total	Low-frequency asymptote line at -20 dB/decade Passes 1 rad/s at 10.45 dB	7.45 dB	-40 dB/decade

Sketching table (phase)

Component	Low-frequency values $\omega < 0.3 \text{ rad/s}$	Mid-frequency values 3 rad/s	High-frequency values $\omega > 30 \text{ rad/s}$
$K = 3.33$	0°	0°	0°
$\frac{1}{s}$	-90°	-90°	-90°
$\frac{1}{0.33s + 1}$	0°	$-\tan^{-1}0.33\omega = -45^\circ$	-90°
Total	-90°	-135°	-180°

We can then sketch the Bode plot using the information in the tables. We can use MATLAB to check how close our sketching approximation is to the actual frequency response plot. We have used the MATLAB command `bode(g, {0.1, 100})` (Figure 15.10).

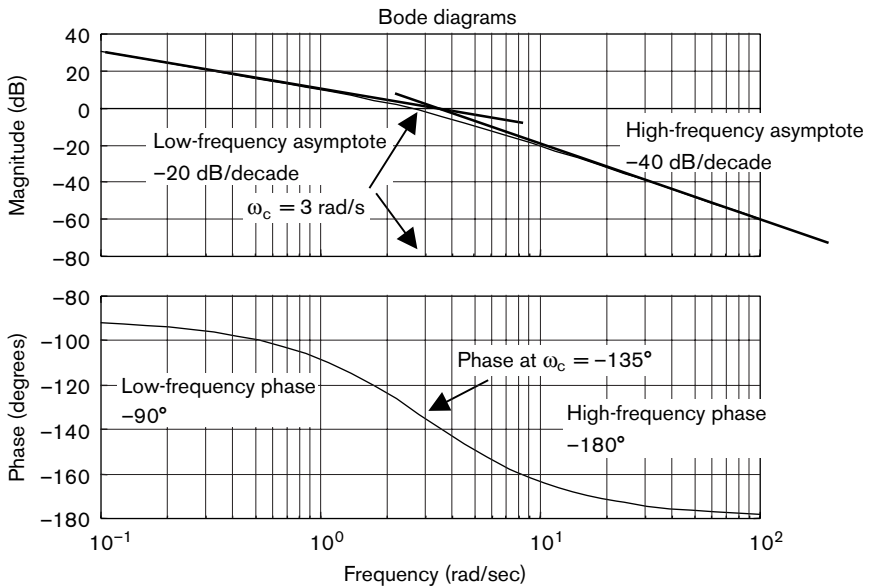


Figure 15.10 Bode plot of $G(s) = 10/[s(s + 3)]$.

Example 4: Two first-order lag terms

Problem Consider the following second-order transfer function. Choose an appropriate frequency range and sketch the Bode plot.

$$G(s) = \frac{10}{0.66s^2 + 2.33s + 1}$$

450 Frequency response using Bode plot presentation

Solution We can check to find out whether the system is under- or overdamped. Recall that for an overdamped system the second-order transfer function has two real poles, that is, the transfer function can be split into

$$G(s) = \frac{K}{(\tau_1 s + 1)(\tau_2 s + 1)}$$

To verify that our transfer function represents an overdamped system, we check the value of $b^2 - 4ac$ in the original: $2.33^2 - 4 \times 0.66 = 2.7889 > 0$. The transfer function represents a second-order overdamped system and can be written as

$$G(s) = \frac{10}{(2s + 1)(0.33s + 1)}$$

The corner frequencies of the two first-order components are:

$$\frac{1}{2s + 1}: \quad \text{corner frequency: } \omega_{c1} = 1/\tau = 0.5 \text{ rad/s}$$

$$\frac{1}{0.33s + 1}: \quad \text{corner frequency: } \omega_{c2} = 1/\tau = 3.0 \text{ rad/s}$$

The transfer function components are then given as:

$$10, \frac{1}{2s + 1}, \frac{1}{0.33s + 1}$$

We can now complete the sketching table.

Sketching table (magnitude)				
Component	Low-frequency asymptote	Mid-frequency values: (i) 0.5 rad/s	Mid-frequency values: (ii) 3 rad/s	High-frequency roll-off
$K = 10$	$20 \log_{10}(10) = 20.0 \text{ dB}$	20.0 dB	20.0 dB	0 dB/decade
$\frac{1}{2s + 1}$	0 dB	-3 dB	Line decreases at -20 dB/decade from intersection with low-frequency asymptote	-20 dB/decade
$\frac{1}{0.33s + 1}$	0 dB	0 dB	-3 dB	-20 dB/decade
Total	Low-frequency asymptote: horizontal line at 20 dB	$20 - 3 = 17 \text{ dB}$	Line decreases at -20 dB/decade from intersection with low-frequency asymptote, then decreases at -40 dB/decade	-40 dB/decade

Sketching table (phase)

Component	Low-frequency value $\omega < 0.05 \text{ rad/s}$	Mid-frequency value: 0.5 rad/s	Mid-frequency value: 3 rad/s	High-frequency value: $\omega > 30 \text{ rad/s}$
$K = 10$	0°	0°	0°	0°
$\frac{1}{2s+1}$	0°	-45°	$-\tan^{-1}(2\omega) = -81^\circ$	-90°
$\frac{1}{0.33s+1}$	0°	$-\tan^{-1}(0.33\omega) = -9^\circ$	-45°	-90°
Total	0°	-54°	-126°	-180°

The Bode plot is shown in Figure 15.11. We notice the high-frequency asymptote in the magnitude plot is -40 dB/decade consistent with our second-order system. Likewise, the phase decreases to -180° .

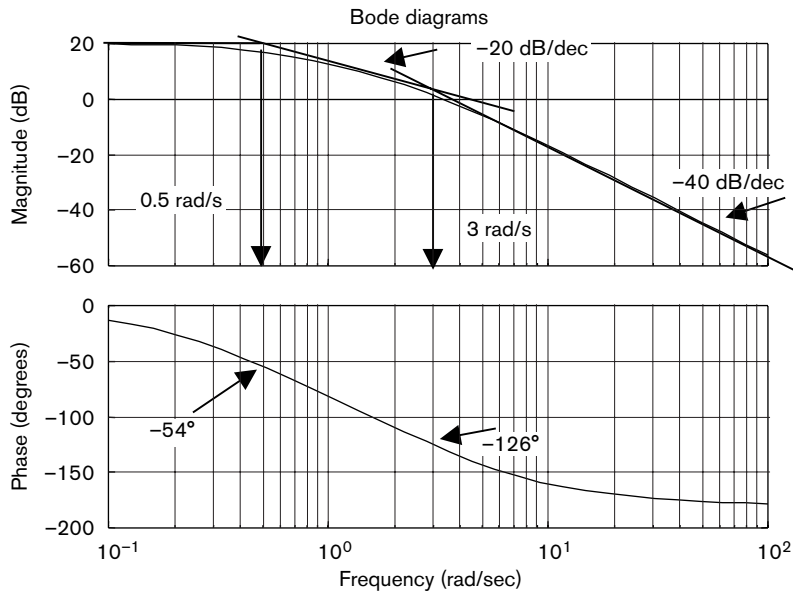


Figure 15.11 Bode plot $10/[(2s+1)(0.33s+1)]$.

Summary: By using a combination of all constant K , s -terms, and first-order lead and lag terms, we can sketch all but second-order underdamped features.

15.9 Second-order underdamped system

A second-order underdamped system has a transfer function of the form:

$$G(s) = \frac{1}{(1/\omega_n^2)s^2 + (2\zeta/\omega_n)s + 1}$$

where ω_n is the natural frequency and ζ is the damping ratio of the system. We remember that

1. The damping ratio, ζ , will be less than 1.
2. The roots of the denominator (poles of the transfer function) will be complex.
3. The step response of the system will show an overshoot depending on the value of the damping.

The magnitude and phase plots of a second-order underdamped system are shown in Figure 15.12. We note the following features.

- The underdamped second-order frequency response has a low-frequency asymptote at 0 dB.
- The gain plot has a peak in the mid-frequency range. The peak does not occur at the natural frequency, but at the damped natural frequency, ω_d . This property can be found by differentiating $|G(j\omega)|$ with respect to ω . If ζ is small (< 0.4), then ω_n and ω_d are near in value.
- The value of gain at the natural frequency ω_n is $1/2\zeta$ or $20 \log_{10}(1/2\zeta)$.
- The peak is dependent on the damping ratio of the system: as the damping decreases, the peak value of the magnitude plot is increased.
- The high-frequency asymptote meets the low-frequency asymptote on the 0 dB axis at the natural frequency.
- The natural frequency represents the corner frequency for the underdamped second-order system.
- The phase changes over approximately two decades, from one decade below the corner frequency to one decade above the corner frequency.

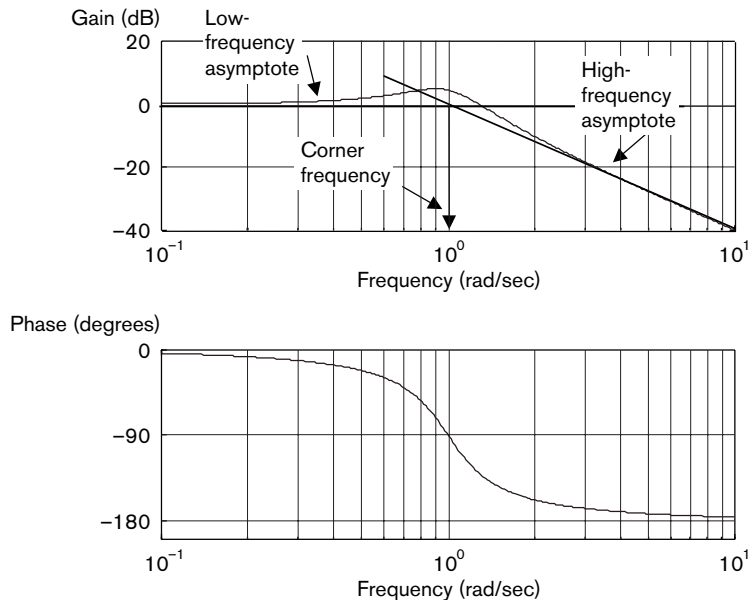


Figure 15.12 Bode plot of underdamped second-order system.

We now look at the transfer function analysis to justify our remarks.

15.9.1 Analysis: magnitude response of second-order underdamped systems

$$G(j\omega) = \frac{1}{(1/\omega_n^2)(j\omega)^2 + (2\zeta/\omega_n)(j\omega) + 1} = \frac{1}{(\omega/\omega_n)^2 j^2 + 2\zeta(\omega/\omega_n)j + 1} = \frac{1}{[1 - (\omega/\omega_n)^2] + j2\zeta(\omega/\omega_n)}$$

The gain of the complex function $G(j\omega)$ is the gain of its numerator divided by the gain of its denominator:

$$|G(j\omega)| = \frac{1}{|[1 - (\omega/\omega_n)^2] + j2\zeta(\omega/\omega_n)|} = \frac{1}{\sqrt{[1 - (\omega/\omega_n)^2]^2 + 4\zeta^2(\omega/\omega_n)^2}}$$

Convert the gain into decibels:

$$\begin{aligned} |G(j\omega)|_{\text{dB}} &= 20 \log_{10}(1) - 20 \log_{10} \left\{ \left[1 - \left(\frac{\omega}{\omega_n} \right)^2 \right]^2 + 4\zeta^2 \left(\frac{\omega}{\omega_n} \right)^2 \right\}^{0.5} \\ &= -20 \log_{10} \left\{ \left[1 - \left(\frac{\omega}{\omega_n} \right)^2 \right]^2 + 4\zeta^2 \left(\frac{\omega}{\omega_n} \right)^2 \right\}^{0.5} \end{aligned}$$

Examine the general feature of the gain response over the different frequency ranges:

Gain in dB: $ G(j\omega) _{\text{dB}}$	$ G(j\omega) _{\text{dB}} = -20 \log_{10} \left\{ \left[1 - \left(\frac{\omega}{\omega_n} \right)^2 \right]^2 + 4\zeta^2 \left(\frac{\omega}{\omega_n} \right)^2 \right\}^{0.5}$
Low-frequency range $\omega \approx 0$	$ G(j\omega) _{\text{dB}} = 20 \log_{10}(1) = 0 \text{ dB}$ horizontal line at 0 dB
Mid-frequency range $\omega = \omega_n$	$ G(j\omega) _{\text{dB}} \rightarrow -20 \log_{10}(1 - 1 + 4\zeta^2)^{0.5}$ $= -20 \log_{10}(2\zeta) = 20 \log_{10}(1/2\zeta)$ If $\zeta < 0.5$, $\log_{10}(1/2\zeta) > 0$ and there will be a peak in the magnitude above the 0 dB line
High-frequency range ($\omega \gg 1$)	$ G(j\omega) _{\text{dB}} \approx -20 \log_{10}[(\omega/\omega_n)^4 + 4\zeta^2(\omega/\omega_n)^2]^{0.5}$ since $\omega^4 \gg \omega^2$ $ G(j\omega) _{\text{dB}} \approx -20 \log_{10}[(\omega/\omega_n)^2]$ $= -40 \log_{10}(\omega) - \log_{10}\omega_n$ Asymptote has slope of: -40 dB/decade (found by differentiating above with respect to $\log_{10} \omega$) Line intersects the horizontal 0 dB line at $\omega = \omega_n$

15.9.2 Analysis: phase response of second-order underdamped systems

We can similarly find the phase of the transfer function, $G(j\omega)$, as a function of frequency over different frequency ranges. Using the transfer function $G(j\omega)$:

$$\begin{aligned} \angle G(j\omega) &= \angle \frac{1}{(1/\omega_n^2)(j\omega)^2 + (2\zeta/\omega_n)(j\omega) + 1} = \angle \frac{1}{(\omega/\omega_n)^2 j^2 + 2\zeta(\omega/\omega_n)j + 1} \\ &= \angle \frac{1}{[1 - (\omega/\omega_n)^2] + j2\zeta(\omega/\omega_n)} = \angle 1 - \angle [1 - (\omega/\omega_n)^2 + 2\zeta(\omega/\omega_n)j] \\ &= 0 - \tan^{-1} \left(\frac{2\zeta(\omega/\omega_n)}{1 - (\omega/\omega_n)^2} \right) = -\tan^{-1} \left(\frac{2\zeta(\omega/\omega_n)}{1 - (\omega/\omega_n)^2} \right) \end{aligned}$$

Phase $\angle G(j\omega)$	$\angle G(j\omega) = -\tan^{-1} \left(\frac{2\zeta(\omega/\omega_n)}{1 - (\omega/\omega_n)^2} \right)$
Low-frequency range $\omega \approx 0$	$\angle G(j\omega) \rightarrow -\tan^{-1}(0) = 0^\circ$ Asymptote at 0°
Mid-frequency value $\omega = \omega_n$	$\angle G(j\omega) \rightarrow -\tan^{-1}(\infty) = -90^\circ$ Mid-frequency phase value is -90°
High-frequency range ($\omega \gg 1$)	$\angle G(j\omega) \approx -\tan^{-1} \left(\frac{2\zeta(\omega/\omega_n)}{1 - (\omega/\omega_n)^2} \right)$ since $\omega^2 \gg \omega$ $\angle G(j\omega) \approx -\tan^{-1}(0)$ $= -180^\circ$ Phase is decreasing towards high frequency. Asymptote at $\angle G(j\omega) = -180^\circ$

15.10 Effect on gain and phase plots of increasing the damping ratio

Figure 15.13 shows how the gain and the phase plots of the underdamped second-order system change for different values of damping. The gain plot shows a large peak for low damping; this corresponds to a higher overshoot on a step response plot.

15.10.1 Sketching procedure for simple second-order underdamped system

- Step 1: Select the desired frequency range and mark it on the log scale of the semilog graph paper.
- Step 2: Select the desired range of the gain in dB and mark it on the linear scale of the log paper.
- Step 3: The low-frequency asymptote is given by the 0 dB line (unless combined with other gains or s -terms).
- Step 4: Find the corner frequency as $\omega_c = \omega_n$. Draw a line of slope -40 dB/decade from the point $(\omega_n, 0 \text{ dB})$. This line is the high-frequency asymptote.
- Step 5: Find the gain at the corner frequency, ω_n , using $|G(j\omega_n)| = 20 \log_{10}[1/(2\zeta)]$ dB.
- Step 6: Starting at the low-frequency asymptote, draw a curve passing through the gain at ω_n and finishing at the high-frequency asymptote.

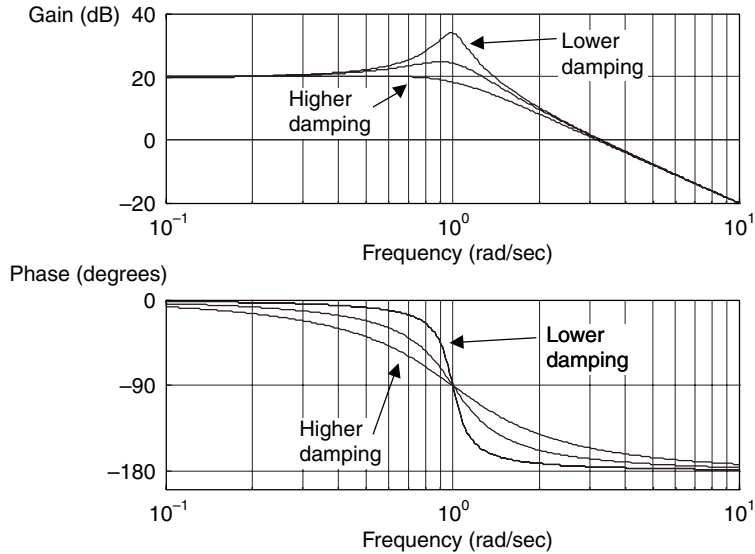


Figure 15.13 Changes in Bode plot for different damping factors.

Step 7: Plot the phase response using the phase values of approximately 0° a decade down from the corner frequency, -180° a decade up from ω_c , and -90° at ω_c . If it is necessary to have a more accurate approximation, use the expression for the phase, $\angle G(j\omega)$, to fill in extra points.

Problem: identification of second-order transfer function from Bode plot

An engineer has measured the frequency response of a position d.c. servo system by applying a sine wave of amplitude of 2 volts and zero phase to the position reference point of the system. The amplitude and the phase of the position output sine wave has been measured and is given in Table 15.6.

Table 15.6 Gain and phase information for identification problem.

The output data from the d.c. servo system							
ω (rad/s)	0.1	0.4	0.8	1	3	4	10
Magnitude (volts)	20	24.2	45.0	50.2	1.01	0.63	0.10
Output phase (degree)	-7	-12.2	-49.3	-89.0	-172.3	-173.1	-177.7

- Draw the Bode plot of the system.
- Draw the low-frequency and high-frequency asymptotes.
- Assuming the system is second-order, find the d.c. gain, natural frequency and the damping ratio of the system.

Solution (a) The gain of the system is:

$$\text{System gain} = \frac{\text{output amplitude at } \omega}{\text{input amplitude at } \omega}$$

We should note that the input is not, in this case, a signal of magnitude 1, but a signal of magnitude 2. The system gain can be calculated using the equation and then converted to dB. The results are shown in the following table and plotted in Figure 15.14, along with the phase response.

$\omega(\text{rad/s})$	0.1	0.4	0.8	1	3	4	10
System gain, $ G(j\omega) $	10	12.1	22.5	25.1	0.50	0.31	0.05
$ G(j\omega) _{\text{dB}}$	20	21.6	27.0	28.0	0.086	-4.01	-19.9

- (b) The low-frequency asymptote in Figure 15.14 is the horizontal line of height 20 dB. The high-frequency asymptote is drawn at a slope of -40 dB/decade and meets the low-frequency asymptote at 1 rad/s . This is the corner frequency of the second-order system.

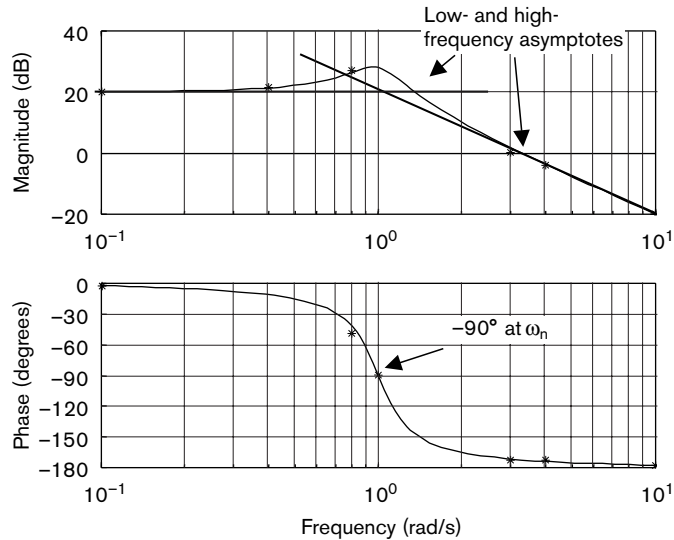


Figure 15.14 Magnitude and phase points of second-order system plotted.

- (c) System gain: the system gain at low frequency is 20 dB. The gain $K = 10$. Natural frequency ω_n : the corner frequency is at $\omega_n = 1 \text{ rad/s}$. For a second-order system, the phase at ω_n is -90° .

Damping ratio, ζ : calculate the damping ratio, ζ , using the value of the gain at ω_n . (Hint: $|G(j\omega_n)| = K/2\zeta$.)

$$\text{Gain dB at } \omega_n = \text{sum of all logged gain components} = 20 \log_{10}(K) + 20 \log_{10}(1/2\zeta)$$

$$28 \text{ dB} = 20 \log_{10} K - 20 \log_{10}(2\zeta)$$

$$20 \log_{10}(2\zeta) = 20 \text{ dB} - 28 \text{ dB} = -8 \text{ dB}$$

$$2\zeta = 10^{-8/20} = 0.3981$$

$$\zeta = 0.199 \approx 0.2$$

Therefore the second-order system is given by

$$G(s) = \frac{10}{s^2 + 0.4s + 1}$$

15.11 Further examples using MATLAB plots

Complex systems often comprise a number of subsystems in cascade. An example of this is shown in Figure 15.15. The input signal is $U(s)$, the output is $Y(s)$ and $Q(s)$ is an intermediate variable. For example, a power system may have a steam turbine (subsystem 1) driving an electric generator (subsystem 2). A second example is a d.c. servo system which has an amplifier in cascade with a motor. Each subsystem may be represented by a first-, second- or higher-order system. We have approached the technique of plotting frequency responses by considering that all systems are combinations of a basic few subsystems. Therefore when it comes to plotting cascade systems, we really only need to extend the number of the smaller subsystems to account for all the factors in the cascaded system.

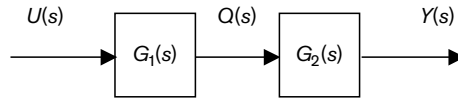


Figure 15.15 Cascade system.

Consider the two cascade subsystems $G_1(s)$ and $G_2(s)$. Define $G_p(s)$ as the transfer function of $U(s)$ to $Y(s)$:

$$Y(s) = G_p(s)U(s) = G_2(s)G_1(s)U(s)$$

$$G_p(s) = G_2(s)G_1(s)$$

Magnitude:

$$|G_p(j\omega)| = |G_2(j\omega)| |G_1(j\omega)|$$

$$|G_p(j\omega)|_{\text{dB}} = 20 \log_{10}(|G_2(j\omega)| |G_1(j\omega)|) = 20 \log_{10} |G_2(j\omega)| + 20 \log_{10} |G_1(j\omega)|$$

Phase relationship:

$$\angle G_p(j\omega) = \angle G_2(j\omega) + \angle G_1(j\omega)$$

This shows us that the magnitude of the combined transfer function is just the sum of logarithmic magnitudes of the individual ones. Similarly, the combined phase is the sum of the individual phases. We now just need to consider all the transfer function components in $G_2(s)$ and $G_1(s)$ separately and combine them using our sketching table for adding logarithmic magnitude and adding phase.

We will look at three examples of combinations of our transfer function components. We will plot the responses using MATLAB and will not produce sketching tables for the following, although it would be easy to use the tables to combine the different components.

Problem: The effect of a adding a pole at the origin

Consider the following transfer functions, $G_1(s)$. Determine the effect on the Bode plot of adding the term $G_2(s) = 1/s$ in cascade with $G_1(s)$.

$$G_1(s) = \frac{4}{s^2 + 1.6s + 4}$$

Solution The combined system is given by

$$G_p(s) = \frac{4}{s(s^2 + 1.6s + 4)}$$

We will use our method for examining the two plots. Firstly we note that the second-order term is not in 'unity constant coefficient form'. By dividing the numerator and denominator by 4 we find:

$$G_p(s) = \frac{1}{s(0.25s^2 + 0.4s + 1)}$$

We now check to see if the second-order term can be formed from two first-order components: check $b^2 - 4ac$: this value is negative, which implies that the second-order term would give us complex roots and represents an underdamped system. We note that the corner frequency $\omega_c = \omega_n$ is given from the second-order component as 2 rad/s.

We can see from the magnitude responses in Figure 15.16 that the second-order plot has a constant d.c. gain, while the integrator adds a high gain at low frequency. The integrator also increases the order of the system and therefore the roll-off rate is increased at high frequency. In the mid-frequency range, for this example, the integrator causes the gain to decrease slightly at the natural frequency. The gain crossover becomes lower, moving from above 2 rad/s to between 1 and 2 rad/s.

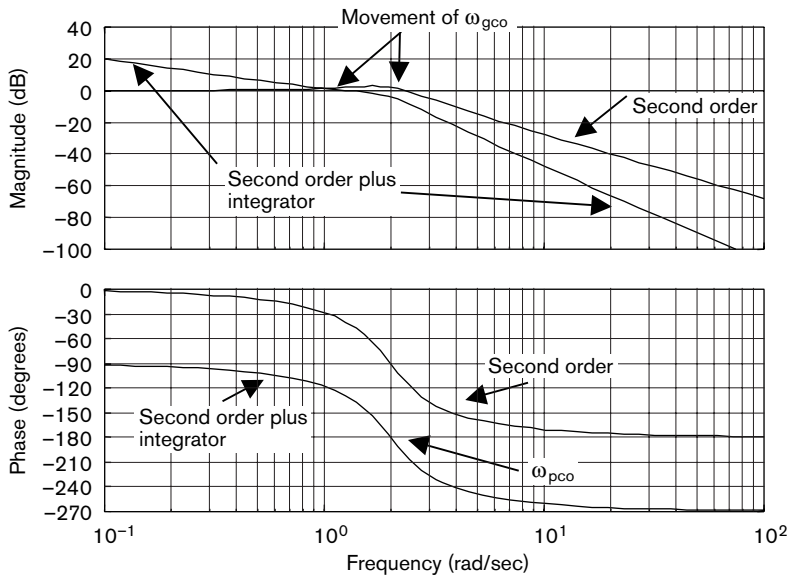


Figure 15.16 Effect of adding an integrator to the system.

For the phase response, the integrator term adds -90° of phase at all frequencies, hence lowering the second-order plot entirely. This causes the phase to cross -180° and the phase crossover frequency becomes $\omega_{pco} = 2$ rad/s. We can write down the the gain and phase margins of the original and cascaded systems.

$G_1(s)$: GM none

PM $> 60^\circ$

$G_p(s)$: GM about 5 dB

PM about 45°

Problem: the effect of adding a real pole

Let $G_1(s)$ be the following transfer function:

$$G_1(s) = \frac{1}{0.25s^2 + 0.4s + 1}$$

Consider the effect on the Bode plot of $G_1(s)$ by adding a first-order lag, $G_2(s)$:

$$G_2(s) = \frac{10}{2s + 1}$$

Solution We note that the second-order term is the same as in the previous example and therefore represents a complex underdamped system. The combined system is given by

$$G_p(s) = \frac{10}{(2s + 1)(0.25s^2 + 0.4s + 1)}$$

where we note that we have added a gain of 10 and a lag term $1/(2s + 1)$.

We can see from Figure 15.17 that the gain has raised the magnitude plot by 20 dB at low frequencies. The mid-frequency values are 17 dB at 0.5 rad/s and 9.9 dB at 2 rad/s. The roll-off has been increased from -40 dB/decade to -60 dB/decade by the addition of the lag term. The lag term would have reduced the gain crossover, but the addition of the gain term has masked this effect and we see an increase in the gain crossover frequency from 2.5 to 3 rad/s due to the higher d.c. gain.

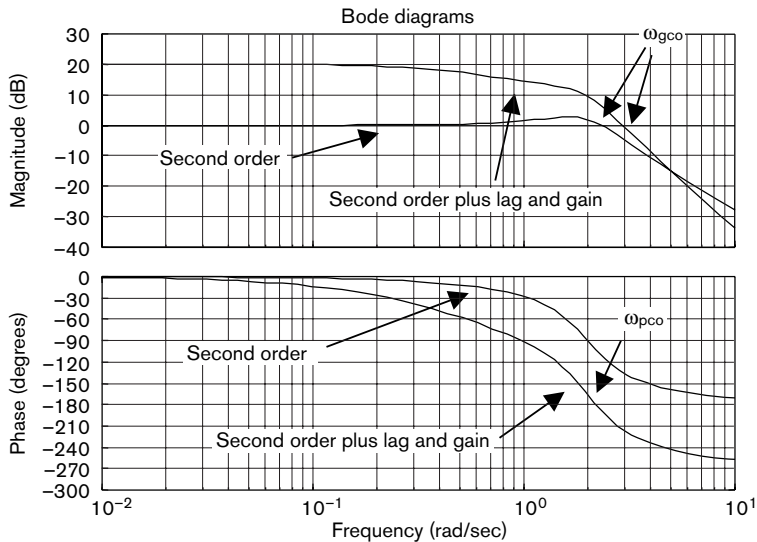


Figure 15.17 Bode plot of $10/[(2s + 1)(0.25s^2 + 0.4s + 1)]$.

For the phase plot, the gain term makes no change to the phase plot. The first-order lag term has its corner frequency before the corner frequency of the second order term. Therefore we see that the phase starts to decrease according to the first-order term before the second-order term has an effect. The overall phase lag has now become -270° with a phase crossover frequency at 2.2 rad/s.

The gain and phase margins of the original and altered systems are

$$\begin{array}{ll} G_1(s): & \text{GM none} \quad \text{PM} > 60^\circ \\ G_p(s): & \text{GM about } -9 \text{ dB} \quad \text{PM about } -33^\circ \end{array}$$

Since the GM and PM are negative, this tells us that the closed-loop system would be unstable.

Problem: The effect of adding a real zero

Let $G_1(s)$ be the following transfer function:

$$G_1(s) = \frac{1}{0.25s^2 + 0.4s + 1}$$

Consider the effect on the Bode plot of $G_1(s)$ by adding a first-order lead term, $G_2(s)$:

$$G_2(s) = (2s + 1)$$

We make a comment here that, in practice, pure lead terms are not implemented since they form an improper transfer function. However, they do occur in combination with other components.

Solution

We note that ω_n is given from the second-order component as 2 rad/s and the corner frequency from the first-order lead is 0.5 rad/s.

Looking at Figure 5.18, we see that although the low-frequency asymptote has remained the same for both the cases with and without the lead term, the lead term has introduced increased gain in the mid-frequency range. This has caused the gain crossover frequency to increase from 2.5 to 9 rad/s. The roll-off rate is not as high due to the introduction of the lead term.

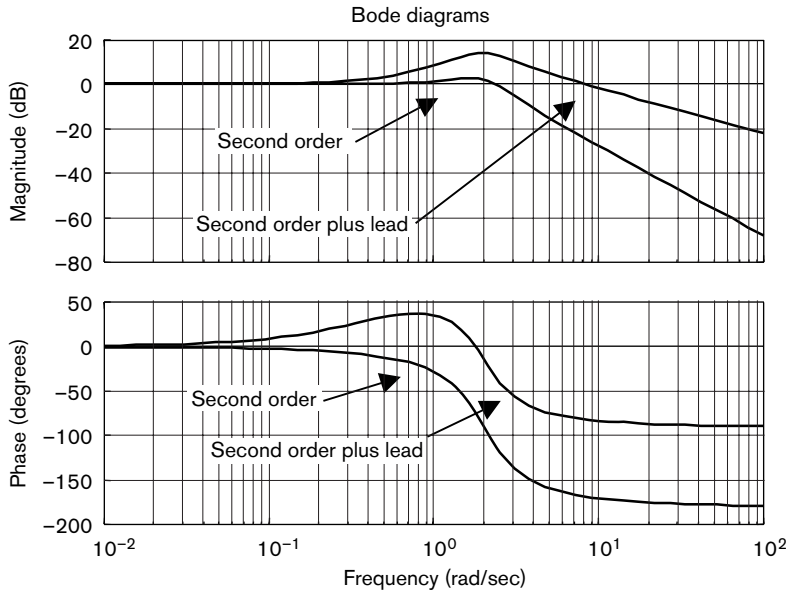


Figure 15.18 Bode plot of $10(2s + 1)/(0.25s^2 + 0.4s + 1)$.

The lead term has its corner frequency before the corner frequency of the second-order term. Therefore we see that the phase starts to increase before the decreasing phase effect of the underdamped second-order system takes effect. The overall phase lag has now become -90° . In some design cases we wish less phase lag in the system and we introduce phase lead through phase lead compensators (Chapter 16).

Since the phase response does not cross -180° for both plots, there is no gain margin before or after the lead term is added. However, the increase in ω_{gco} causes the PM to decrease.

15.12 Magnitude plots of closed-loop and sensitivity transfer functions

From Chapter 8, we remember that the closed-loop transfer function $G_{CL}(s)$ and the sensitivity transfer function $S(s)$ are closely linked. We examine this in more detail and then look at the magnitude plots for the two transfer functions.

15.12.1 The sensitivity function

We have produced designs for our system which satisfy specifications related to $G_{CL}(s)$. However, we have not considered how the disturbance rejection problem fits into the frequency domain. We will find out in the following that the disturbance rejection depends on the **sensitivity function**, which is the transfer function from the disturbance to the output. Consider the unity feedback system in Figure 15.19.

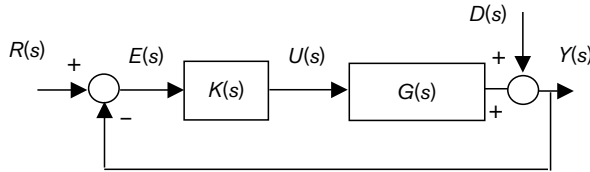


Figure 15.19 The unity feedback system.

We can write the output $Y(s)$ in terms of the reference signal $R(s)$ and the disturbance $D(s)$ as follows:

$$\begin{aligned} Y(s) &= \frac{G_{OL}(s)}{1+G_{OL}(s)} R(s) + \frac{1}{1+G_{OL}(s)} D(s) \\ &= \frac{K(s)G(s)}{1+K(s)G(s)} R(s) + \frac{1}{1+K(s)G(s)} D(s) \\ &= G_{CL}(s)R(s) + S(s)D(s) \end{aligned}$$

We can see that the closed-loop response is a linear combination of the responses due to the reference $R(s)$ and the disturbance $D(s)$. Because of the importance of the two transfer functions, $S(s)$ and $G_{CL}(s)$, we give them the special names of **sensitivity** and **complementary sensitivity** functions.

Key result

Sensitivity function:
$$S(s) = \frac{1}{1+K(s)G(s)}$$

Complementary sensitivity function:
$$T(s) = \frac{K(s)G(s)}{1+K(s)G(s)}$$

We note that

$$T(s) + S(s) = 1$$

Thus $T(s)$ complements $S(s)$; hence the name complementary sensitivity function. This relationship shows an important constraint in any control design problem. For the standard unity feedback system that we use in this book, the complementary sensitivity function, $T(s)$, and the closed-loop transfer function, $G_{CL}(s)$, are the same transfer function. We will continue to use the closed loop transfer function notation, $G_{CL}(s)$.

The relationship between the reference and the output can be written as

$$Y(j\omega) = G_{CL}(j\omega)R(j\omega)$$

where we would like the output Y to behave like the reference R . To achieve this, we should design the controller so that $G_{CL}(j\omega) = 1$ over all frequencies. The complementarity between G_{CL} and S would then ensure that $S(j\omega) = 0$, and perfect disturbance rejection would follow. Unfortunately this perfection is not possible for real processes and controllers. What can be achieved is that $G_{CL}(j\omega) \sim 1$ and $S(j\omega) \sim 0$ in the low-frequency region, and in the high-frequency region we obtain $G_{CL}(j\omega) \sim 0$ and $S(j\omega) \sim 1$. Such designs will be satisfactory for low-frequency reference and disturbance signals, but we will have to use the controller to shape the mid-frequency ranges for the stability margins and use controller roll-off to deal with the high-frequency measurement noise going round the loop. This becomes clearer by examining the magnitude plots for the two transfer functions, G_{CL} and S .

Example We use an open-loop transfer function of plant and controller given by

$$G_{OL}(s) = \frac{6}{s(s+1)(s+3)}$$

1. Closed-loop magnitude plot for a unity feedback configuration

$$Y_R(s) = G_{CL}(s)R(s)$$

The following MATLAB code will produce the magnitude plot of $G_{CL}(s)$.

```

gol = 6/(s*(s+1)*(s+3);      % open loop transfer function
gcl=gol/(1+gol);             % closed loop transfer function
w = logspace(-1,2,300);      % frequency range 0.1 to 100 rad/s
[mag,ph] = bode(gcl,w);      % bode calculation
magdb = 20*log10(mag);       % convert mag to dB
semilogx(w,magdb(:))         % plot on semilog axes

```

Figure 15.20 shows the magnitude plot. We can examine the plot in the low-, mid- and high-frequency ranges:

- *Low-frequency range*: we note that we have a gain of 1 at low frequencies. Since $Y_R(s) = G_{CL}(s)R(s)$, a gain of unity at low frequencies indicates that the closed loop system will track low-frequency reference signals accurately.
- *Mid-frequency range*: this section of the plot is usually shaped by the controller to give the required performance in terms of, for example, acceptable bandwidth (speed of response) and overshoot. We note that the peak on the closed-loop magnitude plot is usually given the notation M_p and occurs at the frequency ω_p . The value of M_p can be evaluated for second-order systems to be

$$M_p = \frac{1}{2\zeta(\sqrt{1-\zeta^2})}$$

$$\omega_p = \omega_n\sqrt{1-2\zeta^2}$$

and can be used as a design specification for systems which exhibit primarily second-order characteristics.

- *High-frequency range*: usually a degree of roll-off is required to remove noise or high-frequency disturbances which may affect the system.

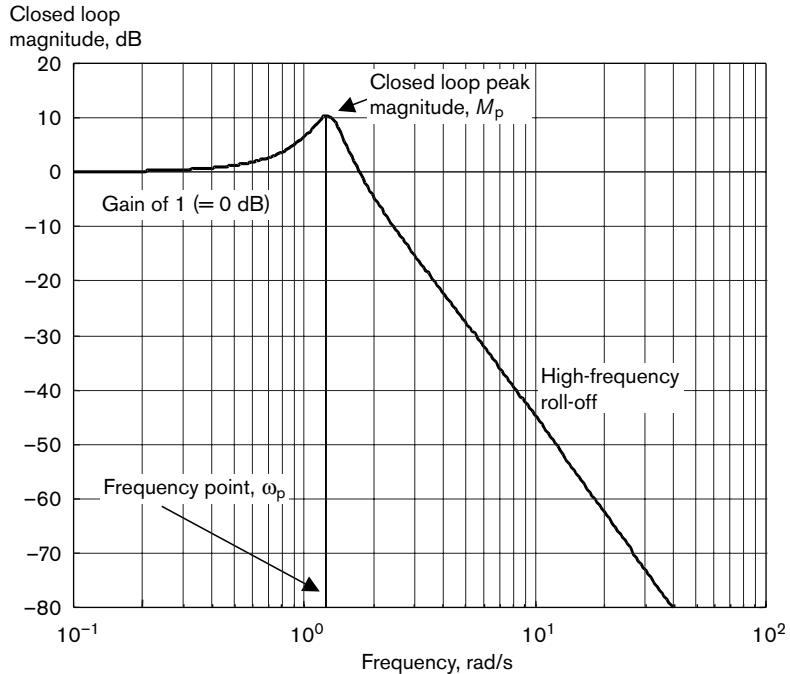


Figure 15.20 Magnitude plot of closed-loop transfer function.

2. Sensitivity magnitude plot

$$Y_d(s) = S(s)d(s)$$

We recall that the sensitivity plot gives us the information on how the disturbances affect the output of the system. The following MATLAB code will produce the magnitude plot of $S(s)$.

```

gol = 6/(s*(s+1)*(s+3);           % open loop transfer function
sens=1/(1+gol);                   % sensitivity TF
w = logspace(-1,2,300);           % frequency range 0.1 to 100 rad/s
[mag2,ph2] = bode(sens,w);        % calculate frequency response of 'sens'
magdb2 = 20*log10(mag2);          % convert mag to dB
semilogx(w,magdb2(:))             % plot on semilog axes

```

Figure 15.21 shows the magnitude plot of the sensitivity function. Once again, we can examine the plot in the low-, mid- and high-frequency ranges:

- *Low-frequency range*: there is very low gain at low frequency. Hence any low-frequency disturbances (usually undesirable) will be attenuated by the system.
- *Mid-frequency range*: the mid-frequency range shows a peak. The magnitude and position of this peak on the frequency axis can be determined by tuning the controller. For example, it may be important not to amplify signals above 1 rad/s. In our plot, we would have to redesign the controller (which would change the open-loop transfer function) to meet this specification.

- *High-frequency range*: we find that the sensitivity function is usually unity at high frequencies (and this corresponds to the transfer function $G_{CL}(s)$ taking near-zero values at high frequency).

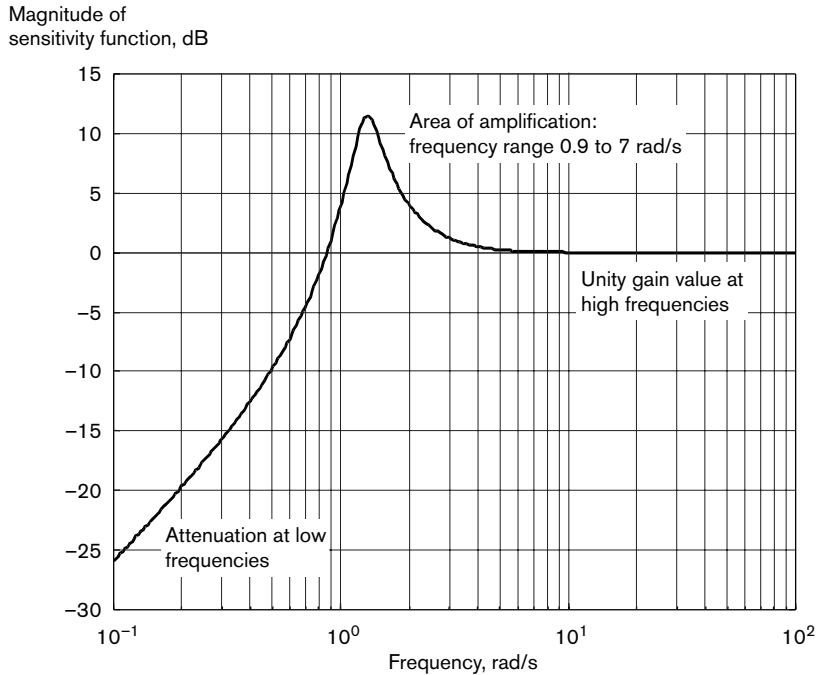


Figure 15.21 Magnitude plot of sensitivity transfer function.

What we have learnt

- ✓ How to use MATLAB to plot a Bode plot.
- ✓ How to enter different frequency ranges into the MATLAB bode command.
- ✓ That there are four basic components which can be combined together to form a Bode plot:
 - system gain K
 - s-terms (s^p)
 - first-order lag/lead terms $(\tau s + 1)^p$
 - second-order underdamped terms $\frac{1}{(1/\omega_n^2)s^2 + (2\zeta/\omega_n)s + 1}$
- ✓ How to form a sketching table which identifies key values of combinations of transfer function components.
- ✓ How to recognise low- and high-frequency asymptotes and find the corner frequencies for systems.
- ✓ To identify the alterations to Bode plots that some simple cascade transfer functions introduce.

Multiple choice

M15.1 A Bode plot represents:

- (a) the gain vs. frequency
- (b) the phase vs. frequency
- (c) the gain vs. phase
- (d) the gain vs. frequency and the phase vs. frequency

M15.2 The phase of the system $G(s) = 1/(s + 1)$ is:

- (a) $\tan^{-1} \omega$
- (b) $-\tan(\omega)$
- (c) $\tan(\omega)$
- (d) $-\tan^{-1} \omega$

M15.3 The magnitude and phase of the transfer function $G(s) = 1/(s + 1)$ at $\omega = 1$ is:

- (a) 0.707 and 45°
- (b) -3 dB and 0.78 rad
- (c) 0.707 and -45°
- (d) 3 dB and -90°

M15.4 The high-frequency range indicates:

- (a) the noise attenuation
- (b) the stability
- (c) the midrange amplification
- (d) the steady state error

M15.5 The transfer function $G(s) = 10/(3s + 1)$ has a corner frequency at:

- (a) 3 rad/s
- (b) 0.33 rad/s
- (c) 1 rad/s
- (d) 30 rad/s

M15.6 The transfer function $G(s) = 1/(0.01s + 1)$ is:

- (a) a lag term
- (b) a lead term
- (c) a term providing high-frequency amplification
- (d) none of the above

M15.7 The closed-loop gain of a feedback system can be calculated from the open-loop gain ($|G(j\omega)|$) using the relationship:

- (a) $\frac{|G(j\omega)|}{|1 + G(j\omega)|}$
- (b) $\frac{1}{1 + |G(j\omega)|}$
- (c) $\frac{|G(j\omega)|}{|1 - G(j\omega)|}$
- (d) $|G(j\omega)|^2$

M15.8 Two cascade systems have transfer functions $G(j\omega)$ and $H(j\omega)$. The overall frequency response can be represented by:

- (a) $G(j\omega) + H(j\omega)$
- (b) $20 \log_{10}(G(j\omega)/H(j\omega))$
- (c) $20 \log_{10}(G(j\omega)) + 20 \log_{10}(H(j\omega))$
- (d) $G(j\omega)/H(j\omega)$

M15.9 Using an integrator ($1/s$) to control a system leads to:

- (a) more phase lag
- (b) less phase lag
- (c) more phase lead
- (d) less phase lead

M15.10 Phase lead terms

- (a) improve stability
- (b) decrease phase margin
- (c) decrease gain margin
- (d) increase overshoot

Questions: Practical skills

Q15.1 Find the d.c. gains and the corner frequencies for the following systems:

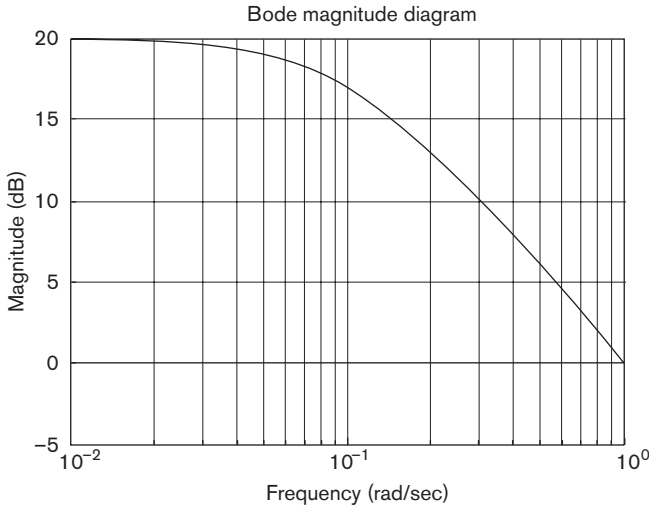
(a) $G_1(s) = \frac{s+2}{(s+0.3)(s+0.5)}$

(b) $G_2(s) = \frac{s+4}{s(s+0.1)(s^2+s+1)}$

Q15.2 Find the low-frequency gain, the high-frequency roll-off and the low- and high-frequency phases for each term in the following transfer function:

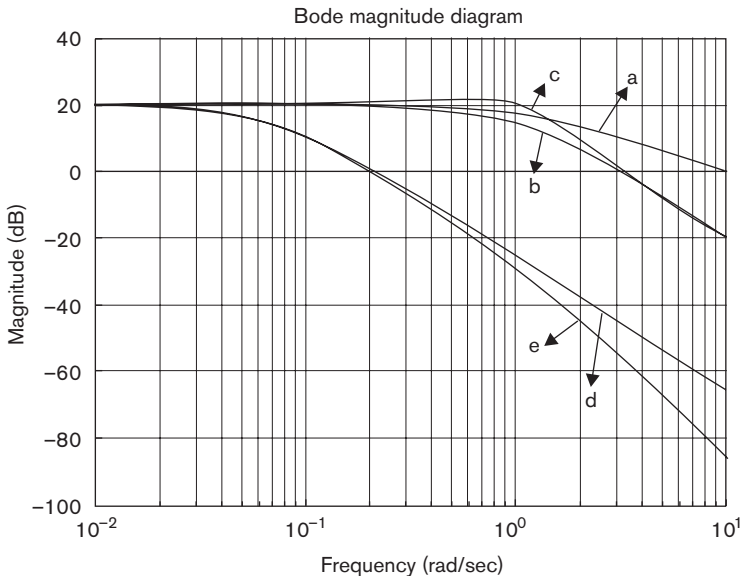
$$G(s) = \frac{20(0.3s + 1)}{s(4s + 1)(s^2 + 2s + 8)}$$

Q15.3 The gain response of a first-order system is given below:



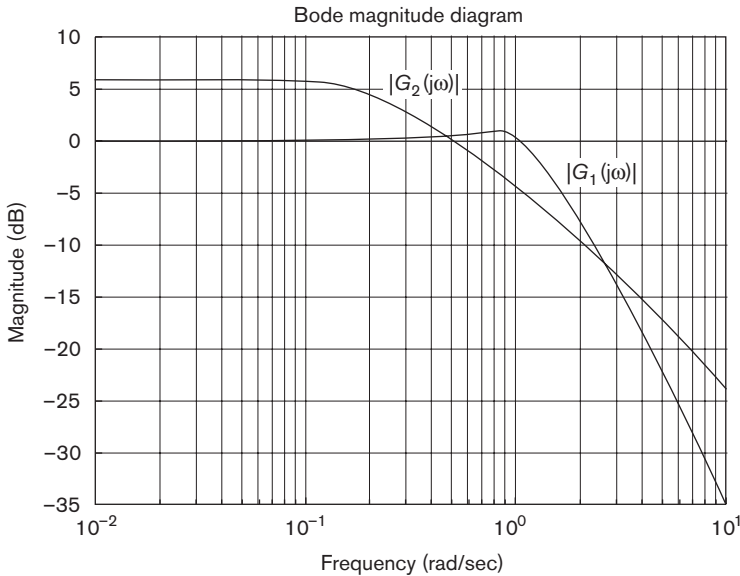
Find the gain and the time constant of the system.

Q15.4 The gain responses of the systems $ga(s)$, $gb(s)$, $gc(s)$, $gd(s)$ and $ge(s)$ are given below:

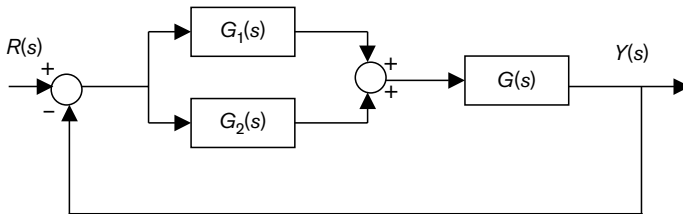


Determine the order of each system, and for any second-order system state whether the system is under-, over- or critically damped.

Q15.5 A cascaded system $G(s)$ is given by $G(s) = G_1(s)G_2(s)$, where $G_1(s)$ and $G_2(s)$ have the gain responses shown below. Find the gain response of $G(s)$.



Q15.6 If we wanted to investigate the closed loop stability of the following system by means of a Bode plot, what open loop transfer function should we enter in MATLAB?



Problems

P15.1 Consider the second-order system:

$$G(s) = \frac{1}{(s^2 + 2\zeta s + 1)}$$

Write an M-file to plot the gain response of $G(s)$ for $\zeta = 0.2, 0.4, 0.6, 0.8, 1$ and 2 on the same figure window. Use your plots to state a trend between the maximum of the gain response and the damping ratio.

P15.2 For the following system, find the function for $|G(j\omega)|$ and $\angle G(j\omega)$. Hence work out the gain and phase crossover frequencies and the gain margin and phase margin.

$$G(s) = \frac{10}{s+1}$$

Check your analysis using suitable MATLAB commands.

468 Frequency response using Bode plot presentation

P15.3 Consider the unity feedback system

$$G(s) = \frac{K}{s(10s+1)}$$

Use `r1tool` to find the value of gain K for the system to have a phase margin of greater than 40° .

P15.4 A PI controller of the form:

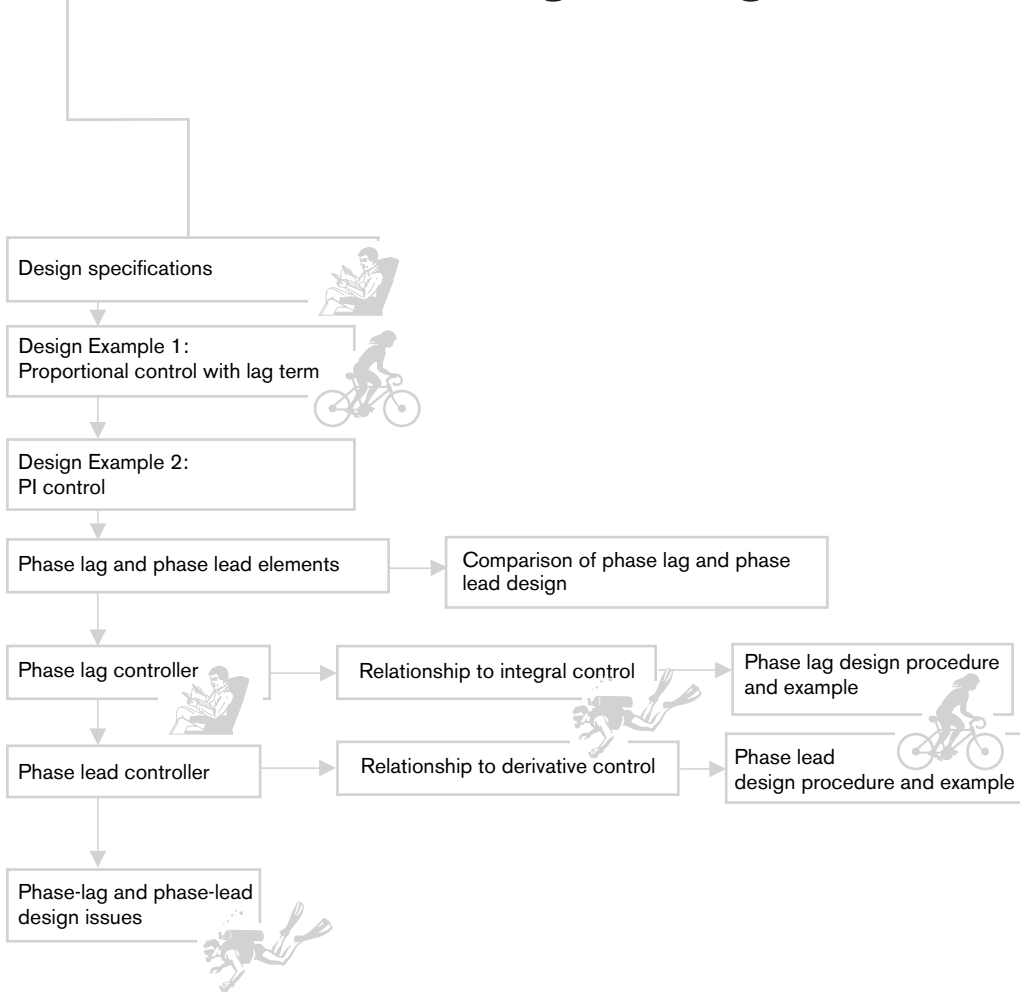
$$C(s) = K \left(1 + \frac{0.1}{s} \right)$$

is used to control the system

$$G(s) = \frac{s+5}{s(10s+1)}$$

Use `r1tool` to find a value for K to give a closed-loop damping ratio of 0.5.

16 Controller design using the Bode plot



We have learnt about the Bode plot representation and how easy it is to cascade simple process systems and modify the frequency response plot accordingly. We use this when we come to design with the Bode plot where we are going to cascade a controller with the system to give a satisfactory overall response. We present two design examples which look at introducing simple terms, such as gains and lag or lead terms, into the controller. Then we look at two controllers, called phase lag and phase lead controllers, which are specifically aimed at providing additional phase to the open-loop phase plot.

Learning objectives

- To understand the effect that cascading simple controller elements has on the Bode plot of the system.
- To design simple controllers using the Bode plot.
- To study the characteristics of phase lag and phase lead controllers.
- To appreciate the effects that phase lag and phase lead elements have on the system open-loop frequency response.
- To be able to analyse the effects of phase lag and phase lead controllers on closed-loop system stability and performance.
- To learn how to design phase lag and phase lead controllers.

16.1 Design specifications

Before we begin the design examples, we provide a summary table of design specifications that we have met, either in the time domain or in the frequency domain. We note that there is a formula or approximation for the design specifications. In some cases these are fairly complex and we did not derive the formulas, but list them to provide a design specification table for use in examples outwith this textbook. Many of the formulas have been derived from second-order systems and are therefore only appropriate for use in second order or systems with dominant second-order poles; therefore the formulas or approximations should not be used blindly with *any* control system.

Design specifications		
Specification	Symbol	Formula or approximation
Overshoot	OS(%)	$\exp\left(\frac{-\pi\zeta}{\sqrt{1-\zeta^2}}\right) \times 100$
Time to overshoot	t_{OS}	$\frac{\pi}{\omega_n \sqrt{1-\zeta^2}}$
Settling time	$t_s(2\%)$	$\frac{4}{\zeta\omega_n}$
	$t_s(5\%)$	$\frac{3}{\zeta\omega_n}$

Design specifications		
Specification	Symbol	Formula or approximation
Rise time	t_r	$\frac{1 + 1.1\zeta + 1.4\zeta^2}{\omega_n}$ or $\frac{1.8}{\omega_n}$
Phase margin	PM	$\tan^{-1}\left(2\zeta \frac{1}{(4\zeta^4 + 1)^{0.5} - 2\zeta^2}\right)^{0.5}$ or 100ζ
Gain crossover frequency	ω_{gco}	$\omega_n[\sqrt{(4\zeta^2 + 1)} - 2\zeta^2]^{0.5}$
Resonant peak	M_p	$\frac{1}{2\zeta\sqrt{1 - \zeta^2}}$
Resonant frequency	ω_p	$\omega_n\sqrt{1 - 2\zeta^2}$
Damped natural frequency	ω_d	$\omega_n\sqrt{1 - \zeta^2}$
Bandwidth	ω_{bw}	$\omega_n[1 - 2\zeta^2 + \sqrt{(2 - 4\zeta^2 + 4\zeta^4)}]^{0.5}$

16.2 Design example 1: proportional control with lag term added

We use the liquid level example of Chapter 5 where a valve controls the flow of liquid into a tank (Figure 16.1).

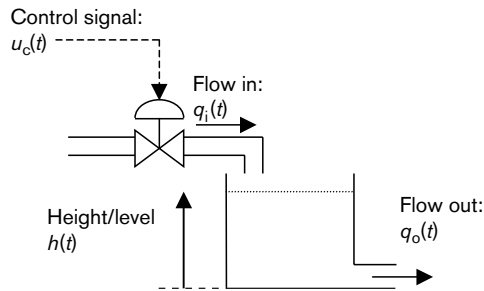


Figure 16.1 Control of liquid level.

The measured level in the tank is given in mA. The modelling in Chapter 5 resulted in a process description shown in Figure 16.2.

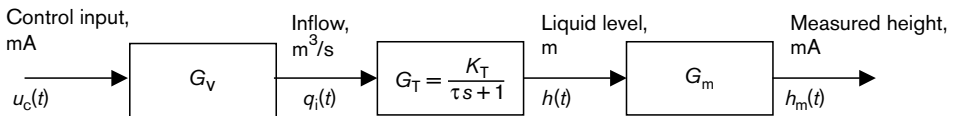


Figure 16.2 Model of liquid level process.

In this control example, the plant engineer has determined that precise level control is not so important, but there is a lot of noise on the measurement which is causing concern. The physical aspects of the tank are also important. The tank is open and 5.5 m high. Filling from empty, the working level in the tank is 5.0 m, and we must avoid at all

costs the tank overflowing. These practical problems result in the following control design specification:

1. Steady state error: e_{ss} (permitted) as a percentage of set point:

$$e_{ss} = \frac{y_{\max} - y_{\text{ref}}}{y_{\text{ref}}} \times 100 = \frac{5.5 - 5}{5} \times 100 = 10\%$$

On a unit step we need $|e_{ss}| < 0.1$ m; on a 5 m step this will translate into a steady state error $|e_{ss}| < 0.5$ m.

2. High roll-off rate to provide sharper cut-off into the high frequency range in order to provide good noise attenuation.
3. Because the tank is open, if the overshoot on a 5 m reference exceeds 0.5 m the tank will overflow, so we must check that the overshoot is within limits.

16.2.1 Solution procedure

- (a) Draw the closed-loop block diagram.
- (b) Decide on the structure of the controller.
- (c) Use the formula for steady state error to determine the value of controller gain K .
- (d) Determine the parameters of any other controller terms to meet other specifications.

(a) Closed-loop block diagram

The closed-loop block diagram is shown in Figure 16.3, where we can see the actuator, process and transducer blocks.

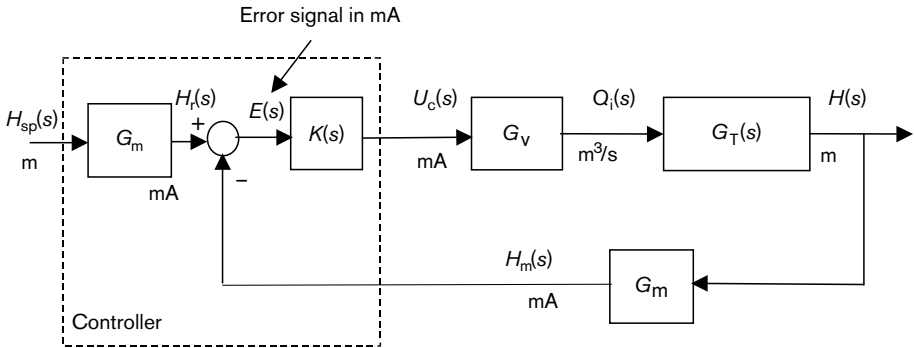


Figure 16.3 Liquid level feedback control system.

We know that the signal, $H_T(s)$, at the comparator must have the same units as the feedback signal, $H_m(s)$. However, we do not usually fix the level set point in mA; we use the units of the actual controlled signal, $H(s)$, so that a set point in metres, for example, would result in an output in the same units. We therefore introduce the input scaling block G_m in the forward path of the block diagram. In practice an electronic controller unit would contain both the input scaling, G_m , the comparator and the controller transfer block $K(s)$.

The *open-loop* transfer function is given by

$$G_{OL}(s) = K(s)G_vG_T(s)G_m = K(s)G_{sys}(s)$$

where $G_{\text{sys}}(s) = G_v G_T(s) G_m$. We use the parameters found in Chapter 5 and we change the time base from seconds to minutes, since the time constant is fairly long (= 1758 seconds).

$$G_v = 0.0034 \text{ m}^3/\text{s}/\text{mA} = 0.204 \text{ m}^3/\text{min}/\text{mA}$$

$$K_T = 140 \text{ m}/\text{m}^3/\text{s} = 2.333 \text{ m}/\text{m}^3/\text{min}$$

$$\tau = 1758 \text{ s} = 29.3 \text{ min}$$

$$G_m = 0.378 \text{ mA}/\text{m}$$

We find that the loop transfer function is given by:

$$G_{\text{OL}}(s) = \frac{0.1799K(s)}{29.3s+1}$$

(b) Controller structure

The specification does not require zero steady state error, therefore there is no immediate requirement for integral action in the controller. The specification on steady state error can be met with a simple proportional gain, K . We must check whether the resulting value of K produces a control signal $u_c(t)$ which is impractical.

The criterion for improved roll-off can be met by shaping the Bode plot to produce a higher roll-off rate. This can be achieved by including a first-order lag term with its corner frequency above the corner frequency of the loop transfer function. However, this may also decrease the gain crossover (thereby reducing the speed of response) and we must therefore take care in the placing of the corner frequency to ensure that the speed of response is not compromised greatly.

The controller could be given by

$$K(s) = \frac{K}{\tau_c s + 1}$$

(c) Steady state error specification

The transfer function from $H_{\text{sp}}(s)$ to $E(s)$ can be calculated using signal equations. First we calculate $H_r(s)$ to $E(s)$, then use $H_r(s) = G_m H_{\text{sp}}(s)$:

$$E(s) = H_r(s) - G_m H(s) = H_r(s) - G_m G_T(s) G_v K(s) E(s)$$

$$E(s)(1 + G_m G_T(s) G_v K(s)) = H_r(s)$$

$$E(s) = \frac{1}{1 + G_m G_T(s) G_v K(s)} H_r(s) = \frac{1}{1 + G_{\text{OL}}(s)} G_m H_{\text{sp}}(s)$$

Using the system and controller transfer functions gives:

$$E(s) = \frac{0.378}{1 + \frac{0.1799K}{(29.3s+1)(\tau_c s+1)}} = \frac{(29.3s+1)(\tau_c s+1)0.378}{(29.3s+1)(\tau_c s+1) + 0.1799K}$$

We note that the specification on e_{ss} is given in metres, yet the error signal here is in mA. We use the scaling value of G_m to convert the 0.1 m specification for a unit step response to a specification on e_{ss} in mA: $e_{\text{ss}} \leq 0.0378 \text{ mA}$.

The steady state error is given by:

$$e_{ss} = \lim_{s \rightarrow 0} sE(s) = \lim_{s \rightarrow 0} s \frac{(29.3s+1)(\tau_c s+1)0.378}{(29.3s+1)(\tau_c s+1)+0.1799K} \frac{1}{s}$$

$$= \frac{0.378}{1+0.1799K} \leq 0.0378$$

Solving this for K gives a requirement for $K \geq 50.03$ in order to meet the specification. A value of $K = 50.03$ will therefore ensure that $|e_{ss}| < 0.1$ to a unit step input. A larger value of K will reduce this offset, but will also produce increased overshoot. We therefore choose $K = 50.03$ as the minimum value of K that satisfies the specification and proceed to check whether this value will also meet the overshoot specification.

(d) Shaping the Bode plot and verifying closed-loop time responses

The loop transfer function with gain K is given by

$$KG_{sys}(s) = \frac{9.0}{(29.3s+1)}$$

The corner frequency of this transfer function is at 0.0341 rad/min.

We would like to introduce a first-order lag term to provide more high-frequency measurement noise filtering. The insertion of the lag term will increase the roll-off rate, but we do not wish this to decrease the gain crossover excessively as this will cause the system to respond more slowly. We therefore consider a lag term with a corner frequency one decade higher than the system value of 0.03 rad/min. Therefore the pole of the lag term at $s = -0.33$ rad/min gives a value for the time constant of 3 min. For comparison we also consider a lag term with its corner frequency at 0.067 rad/min between the system corner frequency and the design one. This gives us three open-loop transfer functions to compare.

Gain compensation only	Lag term 1: Corner frequency 0.33 rad/min, time constant $\tau = 3$ min	Lag term 2: Corner frequency 0.067 rad/min, time constant $\tau = 15$ min
$G_1(s) = \frac{9.0}{(29.3s+1)}$	$G_2(s) = \frac{9.0}{(29.3s+1)(3s+1)}$	$G_3(s) = \frac{9.0}{(29.3s+1)(15s+1)}$

The Bode plots of these open-loop transfer functions are shown in Figure 16.4, where we can see that for $G_1(s)$ the roll-off rate is -20 dB/decade, as expected for a first-order system, and the roll-off rates for the second-order lag-compensated systems are -40 dB/decade.

From the Bode plots we also find the gain crossover and corresponding phase margin for the three systems:

	$G(s)$ Gain compensation only	$G_2(s)$ Lag term 1	$G_3(s)$ Lag term 2
ω_{gco}	0.3 rad/min	0.25 rad/min	0.13 rad/min
PM	100°	60°	40°

Let us look at the closed-loop step responses for the three systems to correlate these values to the speed of response and the closed-loop time domain behaviour of the system. The closed-loop transfer function from $H_1(s)$ to $H(s)$ is given by

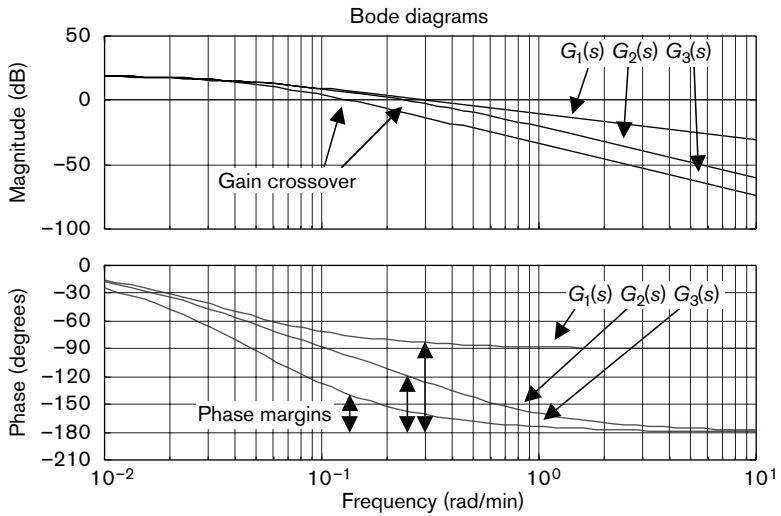


Figure 16.4 Bode plots of the open-loop transfer functions $G_1(s)$, $G_2(s)$ and $G_3(s)$.

$$H(s) = \frac{\text{forward}}{1 + \text{open loop}} = \frac{G_T(s)K(s)G_vG_m}{1 + G_{OL}(s)} H_{sp}(s)$$

We can see from the closed-loop step unit responses in Figure 16.5 that the steady state error requirement has been met. The level rises to a steady value of 0.9 m, giving a 0.1 m offset. We can see that the system becomes slower for Systems 2 and 3, corresponding to the decrease in gain crossover giving a lower bandwidth for the system. We also note that the behaviour becomes more oscillatory, with increasing overshoots as the corner frequency of the lag term approaches that of the system corner frequency. This can occur when the phase margin reduces as it does in this example.

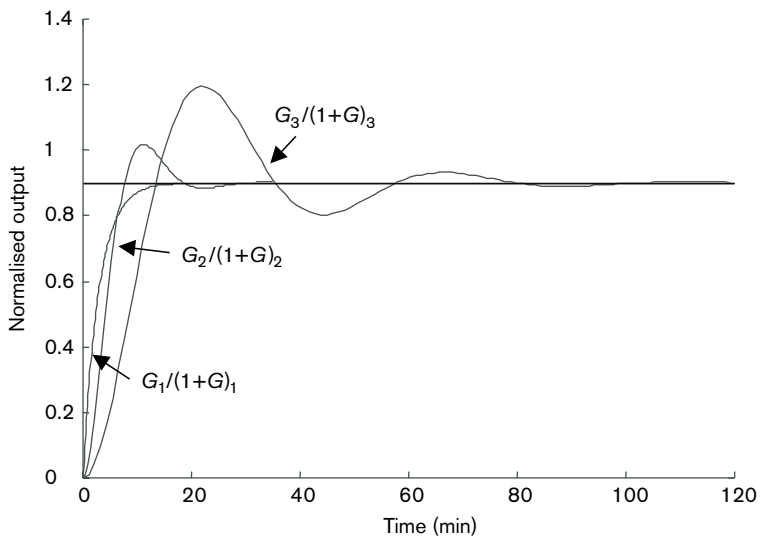


Figure 16.5 Closed-loop step responses for the three systems.

We realise that $G_3(s)$ is an unacceptable design because the overshoot peak of 1.2 (20% of final value) is equivalent to a 6.0 m level on a 5.0 m reference step. Thus the tank would overflow for this design.

Figure 16.6 shows the Bode plots of the closed-loop systems. We would not expect to see unity d.c. gain (this would imply a zero steady state error), but we find that the design has a d.c. gain of $20 \log_{10} 0.9 = -0.91$ dB. From the plots we see that we have a d.c. gain just below 0 dB, as expected. We note that for Systems 2 and 3 we see a peak in the Bode plot; this peak is greater than 0 dB and relates to the overshoot we saw in the step responses of Systems 2 and 3. We notice that the roll-off is greater in both Systems 2 and 3, as specified in the design criterion. For example, at 3 rad/min the attenuation has changed from -20 dB with System 1 to -40 dB with System 2.

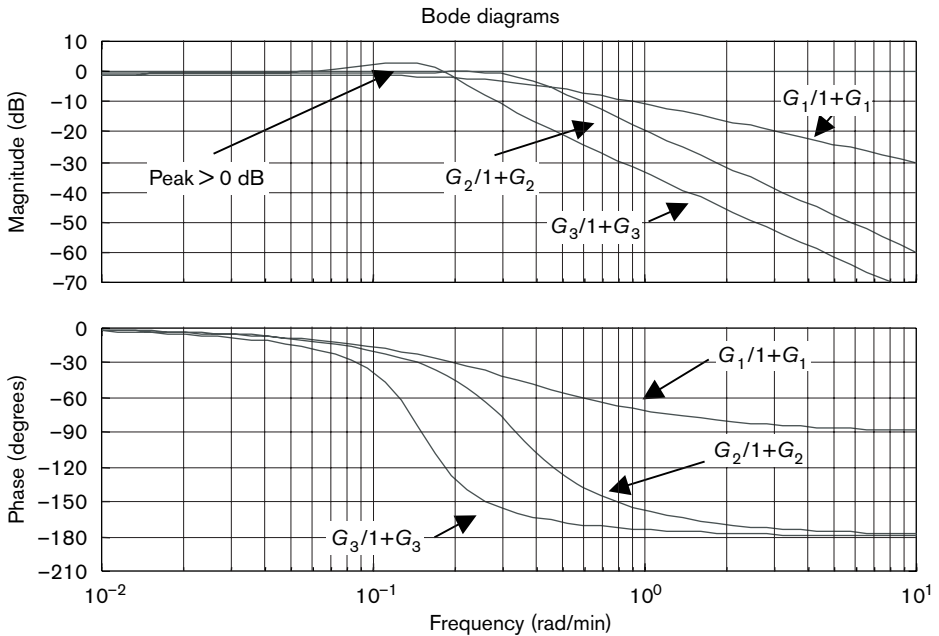


Figure 16.6 Bode plots of closed-loop transfer functions.

It remains to check the control signal $u_c(t)$ to verify that we are not demanding too much from the actuation equipment (Figure 16.7). For System 1, we see that due to the step input and the fact that the controller $K(s)$ is simply a gain, we have a sharp rise in control output followed by a decay to the steady state level of 16 mA. This steady state level is within the equipment range. However, the rate of increase in controller output must also be physically realisable. This sharp rise in the proportional control output signal is a good example of *proportional kick* (Chapter 18). Systems 2 and 3 show more slowly responding control signals which give rise to the slower step responses. The magnitudes and rates of change are much more acceptable for the actuation equipment. As the final controller we would choose design number 2 due to its speed of response and improved phase margin. Design 3 was unacceptable due to the large overshoot and the speed of response for Design 1 was probably too fast for real actuators.

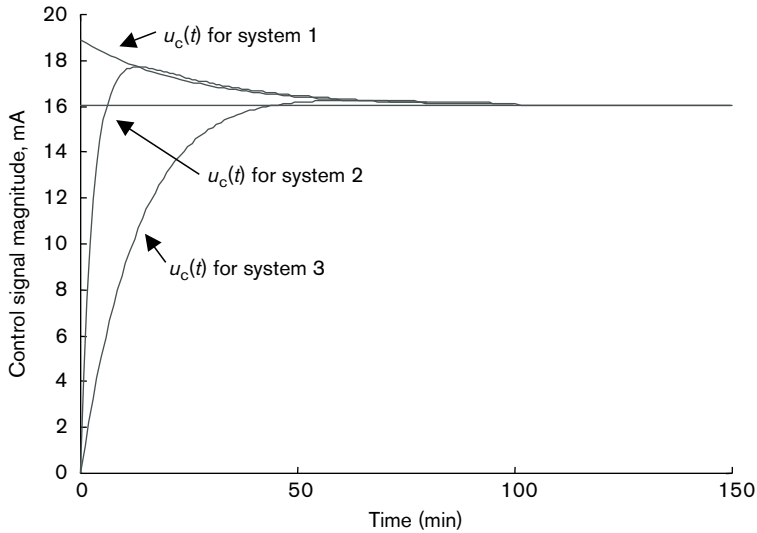


Figure 16.7 Control signals for closed-loop step responses.

16.3 Design example 2: PI control

Design Example 2 uses the model from Chapter 5 for the conveyor belt system in a manufacturing system. The Actuator–Process–Transducer block diagram is shown in Figure 16.8, where

$$G_{dc}(s) = \frac{2}{2+0.5s} \quad G(s) = \frac{1}{0.1s+0.5} \quad G_m = 159.2$$

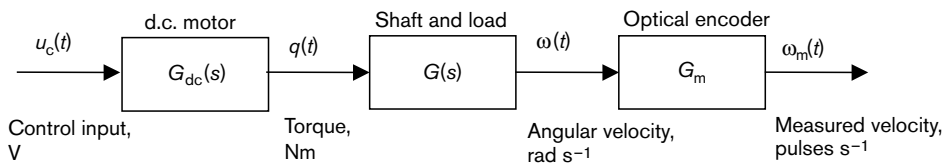


Figure 16.8 Actuator–Process–Transducer model for conveyor belt system.

To meet the manufacturing requirements it is imperative that the conveyor belt operates at the correct speed. It must have acceptable transient performance in moving from one speed setting to the next. In addition, the process engineer who derived the model found that the design manuals for the equipment were not precise on the model parameters. To allow for errors in the parameters, we require a reasonable phase margin which would allow the system to differ slightly from the design model but without compromising the stability of the system. These requirements result in the design specification:

1. Zero steady state error to step changes in input speed
2. Overshoot to be less than or equal to 15% of final value
3. PM of 45° to 60°

16.3.1 Solution procedure

- (a) Draw the closed-loop block diagram.
- (b) Decide on the structure of the controller.
- (c) Plot the Bode plots of the system and determine the parameters of the controller.
- (d) Examine the step responses to find whether the time domain behaviour is acceptable.

(a) Closed-loop block diagram

The closed-loop block diagram (Figure 16.9) is similar in structure to that of Design Example 1. We have introduced the block G_m in the forward path to allow us to compare the actual input and output of the system. In practice, alternative scaling blocks could be introduced to produce the reference and output signals in units such as revolutions/s.

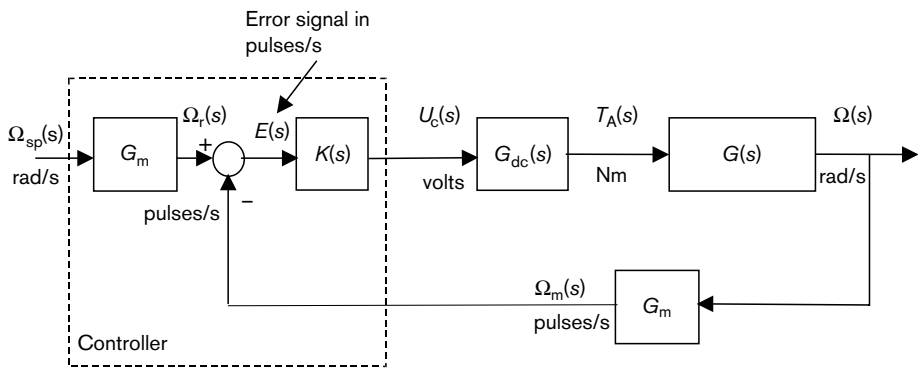


Figure 16.9 Closed-loop block diagram for conveyor system.

The Actuator–Process–Transducer transfer function is given by

$$G_{\text{sys}}(s) = G_m G(s) G_{\text{dc}}(s) = \frac{3184}{(2 + 0.5s)(0.1s + 0.5)}$$

Changing the first-order terms to gain–time constant form gives

$$G_{\text{sys}}(s) = \frac{3184}{(0.25s + 1)(0.2s + 1)}$$

This represents an overdamped system with corner frequencies at $\omega_{c1} = 4$ rad/s and $\omega_{c2} = 5$ rad/s. The open-loop transfer function is given by

$$G_{\text{OL}}(s) = G_{\text{sys}}(s) K(s) = \frac{3184K(s)}{(0.25s + 1)(0.2s + 1)}$$

(b) Controller structure

To meet the criterion for zero steady state error we introduce integral action in the controller. From our knowledge of Bode plots, we know that an integral controller of the form $K(s) = K_i/s$ will produce infinite d.c. gain but will also reduce the phase by 90° over all frequencies. Consider a PI controller of the form

$$U_c(s) = \left(K_p + \frac{K_i}{s} \right) E(s) = \frac{K_p s + K_i}{s} E(s) = K_c \left(\frac{\tau_c s + 1}{s} \right) E(s)$$

where $K_c = K_i$ and $\tau_c = K_p/K_i$. The integrator will still provide the infinite d.c. gain required, but the lead term with corner frequency $1/\tau_c$ will introduce $+90^\circ$ of phase at high frequencies; hence it compensates the lag introduced due to the integral term. Therefore we consider two controller designs, one with pure integral action and the other with the PI form:

$$K_1(s) = \frac{K_c}{s} \quad \text{and} \quad K_2(s) = \frac{K_c}{s}(\tau_c s + 1)$$

(c) Calculation of design parameters from Bode plot analysis

The loop transfer functions to be used for plotting the Bode plot become

$$G_1(s) = \frac{3184K_c}{s(0.25s+1)(0.2s+1)} \quad \text{and} \quad G_2(s) = \frac{3184K_c(\tau_c s+1)}{s(0.25s+1)(0.2s+1)}$$

We plot the Bode plot for $G_1(s)$ with a value of $K_c = 1$ (Figure 16.10). The system with gain of $K_c = 1$ is unstable, since the phase margin is -60° . To achieve a phase margin of $+60^\circ$ by changing the gain alone, we would need to use the gain to move the magnitude plot down so that the gain crossover frequency occurred at the point with $+60^\circ$ phase margin. We see from Figure 16.10 that we need to reduce the gain by about 50 dB = 316. The Bode plot of $G_1(s)$ with $K_c = 1/316 = 0.0032$ is also shown in the figure. A change in gain does not alter the phase plot, but we see that the gain crossover is significantly reduced. This will lead to a slower speed of response.

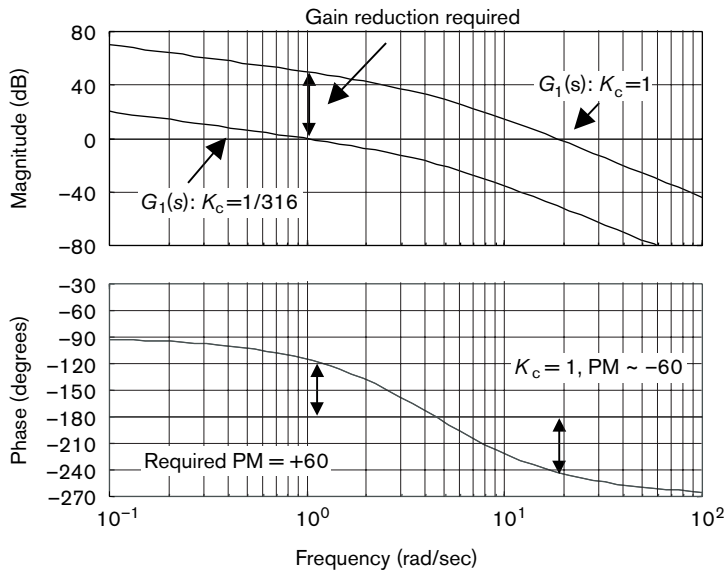


Figure 16.10 Bode plots of $G_1(s)$ with $K_c = 1$ and $K_c = 1/316$.

We can introduce a lead term to increase the gain crossover slightly. This will also change the phase plot for the system. We let the gain K_c be slightly larger than 0.0032 and set the zero of the lead term above the poles of $G_{sys}(s)$.

$$K_c = 0.01, \quad \tau_c = 0.1$$

The Bode plots of $G_1(s)$ with $K_c = 1/316$ and $G_2(s)$ are shown in Figure 16.11.

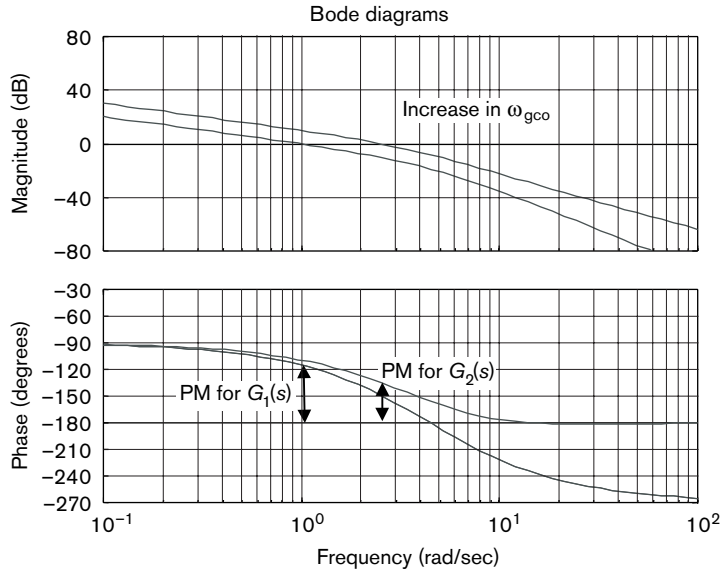


Figure 16.11 Bode plot of $G_1(s)$ with $K_c = 1/316$ and $G_2(s)$.

We can see that the gain crossover has been increased slightly from 1 to 2.5 rad/s. This should produce a faster response. However, we note that the phase margin has now decreased from 60° to 45° , but we are still within the design specifications.

(d) Analysis of time domain behaviour

Let us examine the system step responses to determine the effects of the two controllers on the process output signal and the controller output signal. The closed-loop transfer functions are given by

$$\Omega(s) = \frac{G_m K(s) G_{dc}(s) G(s)}{1 + G_{OL}(s)} \Omega_{sp}(s)$$

In this example, the closed-loop transfer function reduces to

$$\Omega(s) = \frac{G_1(s)}{1 + G_1(s)} \Omega_{sp}(s)$$

The controller output signal, $U_c(s)$, is given by

$$U_c(s) = \frac{G_m K(s)}{1 + G_{OL}(s)} \Omega_{sp}(s)$$

Figure 16.12 shows the closed-loop step responses for system $G_1(s)$ and $G_2(s)$. We see that the second controller does indeed produce a faster response, but due to the corresponding decrease in phase margin there is a large increase ($> 20\%$) in overshoot. Figure 16.13 shows the corresponding controller output signals.

The overshoot of about 22% is outside the design specification. However, control design is an iterative procedure and we would return to part (c) of the design procedure and choose a smaller gain K_c and reassess the performance of the controller.

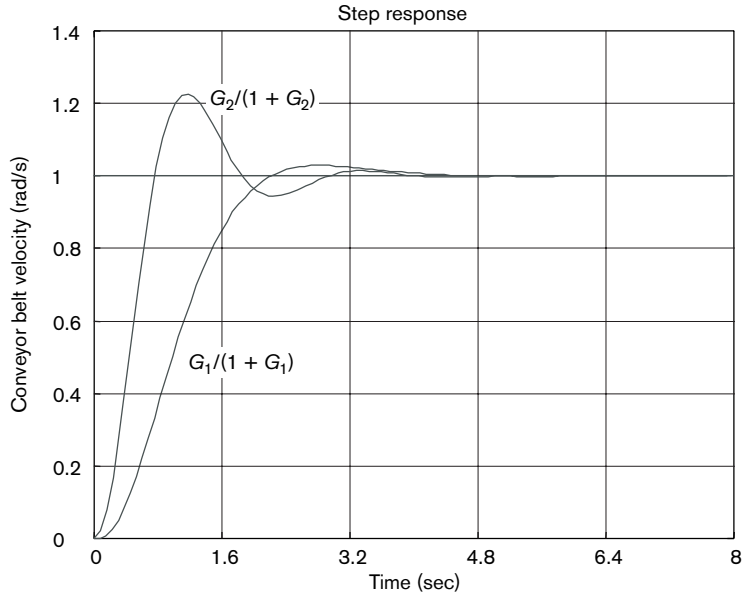


Figure 16.12 Closed-loop step response for $G_1(s)$ and $G_2(s)$.

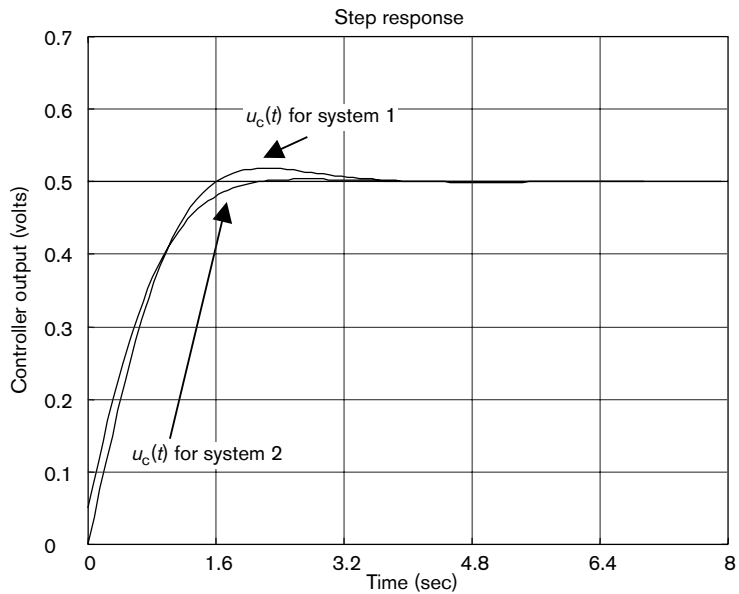


Figure 16.13 Controller output signals for control Systems 1 and 2.

The previous two examples showed the introduction of a phase lag term and a phase lead term, respectively. The combination of both the phase lag and phase lead terms can produce a controller element with a 0° phase change at low and high frequency, since the change in phase caused by one term is cancelled at higher frequencies by the other term. We can use the mid-frequency properties of these new controllers to shape the closed-loop system frequency response to meet gain and phase margin specifications.

16.4 Phase lag and phase lead elements

We define the phase lag and phase lead controllers as:

$$\text{Phase lag controller: } G_{\text{lag}}(s) = \frac{\tau s + 1}{\alpha \tau s + 1} \quad \alpha > 1$$

$$\text{Phase lead controller: } G_{\text{lead}}(s) = \frac{\tau s + 1}{\beta \tau s + 1} \quad \beta < 1$$

We notice that the controllers have both a lag term and a lead term. The difference in the controllers is due to the different values of α and β . In the phase lag controller, the corner frequencies are $1/\tau$ and $1/\alpha\tau$, and can be termed the low and high corner frequencies, $\omega_L = 1/\alpha\tau$ and $\omega_H = 1/\tau$. For the phase lead controller, the low and high carrier frequencies are at $\omega_L = 1/\tau$ and $\omega_H = 1/\beta\tau$, respectively. The Bode plots of the controllers are shown in Figure 16.14.

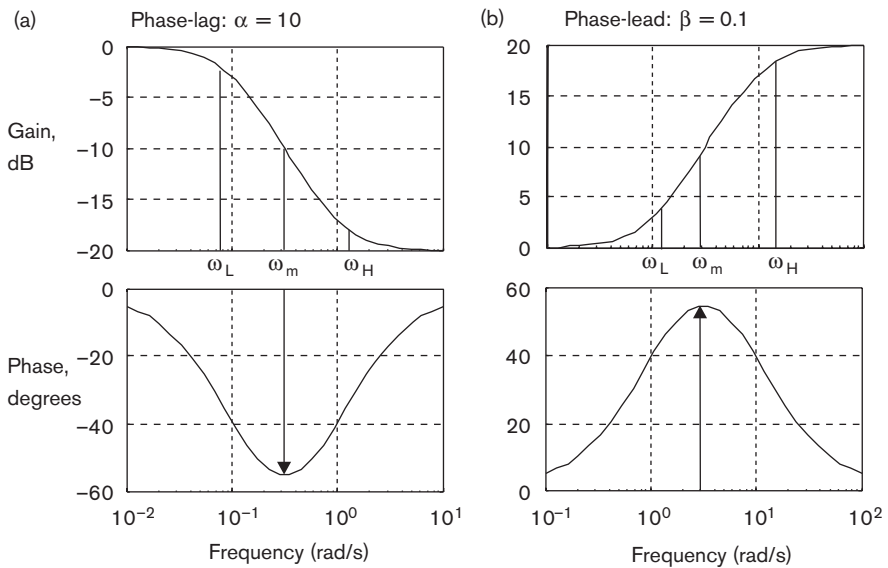


Figure 16.14 Bode plots of phase lag and phase lead elements.

From the individual phase lag and phase lead elements, we would usually see a -90° and a $+90^\circ$ change in phase respectively. However, we can use the combination of lag and lead terms to produce a controller with no phase change at low and high frequencies, but with a specified amount of phase change in the mid-frequency range.

16.4.1 Discussion: phase lag controller properties

We see in Figure 16.14(a), for the phase lag controller, that the phase decreases, but before it reaches -90° the lead term acts to increase the phase. The final change in phase at high frequencies is therefore zero. The maximum change in phase is determined by how closely the two corner frequencies are placed. If they are far apart, then the lag term can cause a large reduction in phase before the lead term acts to increase the phase. If they are

close together, then the lag term can only cause a small change in phase before the lead term brings the phase back to 0° .

Peak change in phase (phase lag controller)

This occurs at the geometric mean of the two corner frequencies on the logarithmic scale and is given by:

$$\omega_m = \frac{1}{\tau\sqrt{\alpha}} \quad \text{where } \omega_L = \frac{1}{\alpha\tau} < \omega_m < \omega_H = \frac{1}{\tau}$$

Magnitude properties

From the magnitude plot, Figure 16.14(a), we find that the gain at low frequencies is unity. The gain at high frequencies approaches $20 \log_{10} 1/\alpha$. We can verify this from the magnitude plot, where the value of α used was 10. The gain starts at 0 dB and decreases to a constant value of $-20 \text{ dB} = 20 \log_{10}(1/10)$. At ω_m , the magnitude has changed by half its total change in dB: $10 \log_{10}(1/\alpha) = -10 \text{ dB}$.

A summary of the phase lag characteristics is given in Table 16.1.

16.4.2 Discussion: phase lead properties

The Bode plot of the phase lead controller given by

$$G_{\text{lead}}(s) = \frac{\tau s + 1}{\beta\tau s + 1} \quad \beta < 1$$

is shown in Figure 16.14(b). The phase increases towards $+90^\circ$, but the lag term affects the phase change and reduces the final phase change to zero.

Peak change in phase (phase lead controller)

As with the phase lag controller, the peak change in phase is determined by how close together the two corner frequencies are. The peak change occurs at

$$\omega_m = \frac{1}{\tau\sqrt{\beta}}, \quad \text{where } \omega_L = \frac{1}{\tau} < \omega_m < \omega_H = \frac{1}{\beta\tau}$$

Magnitude properties

From the magnitude plot, the gain at low frequencies is unity. At high frequencies the gain will tend to $20 \log_{10} 1/\beta$. We can verify this from the magnitude plot, where the value of β used was 0.1. The gain starts at 0 dB and increases to a constant value of $+20 \text{ dB} = 20 \log_{10}(1/0.1)$. At ω_m , the magnitude has changed by half to $+10 \text{ dB}$. A summary of the phase lead characteristics is given in Table 16.1.

16.4.3 Phase lag and phase lead compensation design

The phase lag and phase lead elements have some useful characteristics, which can be employed to modify the gain and phase margins and time domain behaviour of closed-loop systems. For this purpose, we cascade these elements with the processes to be controlled and then choose the parameters of the elements, time constant τ , and either α or β , to meet our control design objectives. The compensated system is schematically shown in Figure 16.15.

Table 16.1 Gain and phase characteristics of phase lag and phase lead elements.

Summary table	Lag term		Lead term	
System	$G_{lag}(j\omega)$		$G_{lead}(j\omega)$	
Transfer function	$\frac{j\tau\omega + 1}{j\alpha\tau\omega + 1}$		$\frac{j\tau\omega + 1}{j\beta\tau\omega + 1}$	
Parameter range	$\alpha > 1$		$\beta < 1$	
Corner frequency	$\omega_L = \frac{1}{\alpha\tau}$	$\omega_H = \frac{1}{\tau}$	$\omega_L = \frac{1}{\tau}$	$\omega_H = \frac{1}{\beta\tau}$
Low- and high-frequency gain	0 dB $20 \log_{10}(1/\alpha) < 0, \alpha > 1$		0 dB $20 \log_{10}(1/\beta) > 0, \beta < 1$	
Low- and high-frequency phase	0° 0°		0° 0°	
Peak phase change	$\phi_m = \sin^{-1}\left(\frac{1-\alpha}{1+\alpha}\right)$		$\phi_m = \sin^{-1}\left(\frac{1-\beta}{1+\beta}\right)$	
Frequency	$\omega_m = \frac{1}{\tau\sqrt{\alpha}}$		$\omega_m = \frac{1}{\tau\sqrt{\beta}}$	
Magnitude $ G _m$	$10 \log_{10}(1/\alpha) < 0, \alpha > 1$		$10 \log_{10}(1/\beta) > 0, \beta < 1$	

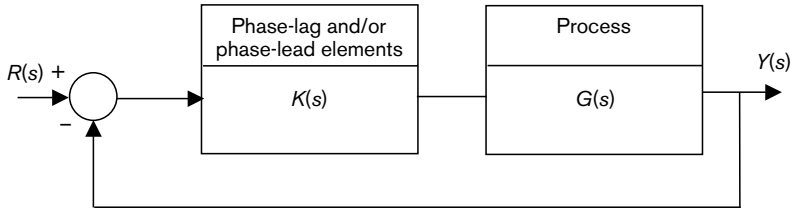


Figure 16.15 Cascade compensator.

When we cascade the lag and/or lead elements, $K(s)$, we change the open-loop system from $G(s)$ to $K(s)G(s)$. We now look at the differences that a phase lag and phase lead compensator would produce when cascaded with a system.

System: $G(s) = \frac{50}{s(0.1s+1)}$ Feedback: Unity gain feedback

Controller (a): phase lag	Controller (b): phase lead
$\alpha = 4.22, \tau = 1.203$	$\beta = 0.455, \tau = 0.0586$

The philosophy behind the two controllers is quite different. For the phase lag design (Figure 16.16(a)), we aim to move the gain crossover to a lower frequency where there is an improved phase margin for the open-loop system. The change in phase occurs at low

frequency and has little effect near the gain crossover where the phase margin is calculated. For the phase lead controller (Figure 16.16(b)), the aim is to use the additional phase from the controller to improve the phase margin of the system by adding phase at the gain crossover frequency, ω_{gco} . In the figure we can see that the phase lead controller has been placed to add extra phase at ω_{gco} . The resulting increase in gain occurs at and after ω_{gco} .

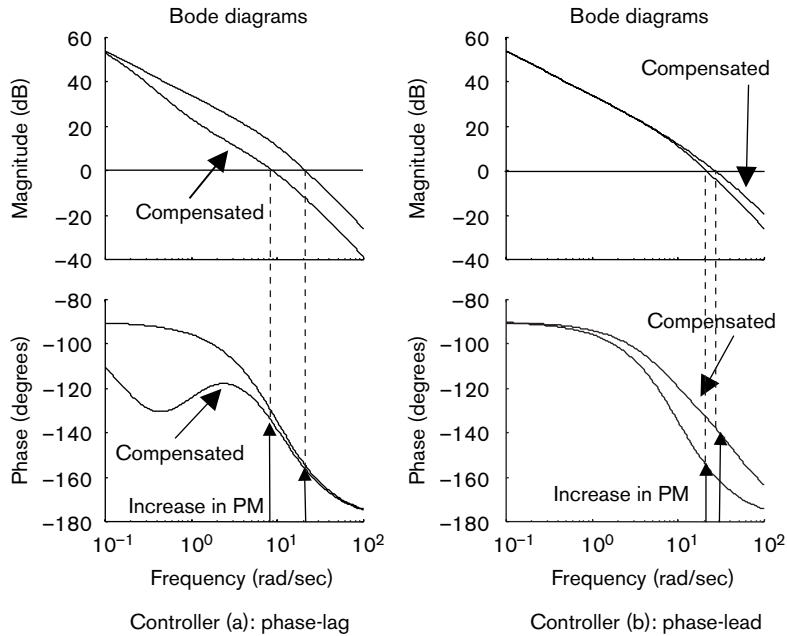


Figure 16.16 The frequency responses of the uncompensated and compensated systems.

The corresponding closed-loop step responses are shown in Figure 16.17. Figure 16.17(a) shows the time responses for the system with and without the phase lag compensator, and Figure 16.17(b) shows the time responses when we use the phase lead compensator. We first note that in both cases the overshoot has decreased. The lag compensated system has a lower gain crossover than the lead compensated system and thus has a slower step response. The gain at low frequencies has remained unchanged in both cases. The gain over the high-frequency region is reduced for the lag-compensated system, while the gain has increased for the lead-compensated system. Thus, the lead-compensated system is more sensitive to noise and high-frequency disturbance, and consequently will produce a more active actuator (input) signal.

In particular, if a higher phase margin is required, we can choose a lead element with a maximum phase lead at around the gain crossover frequency to improve the phase margin. On the other hand, we can choose a lag element to improve the steady state behaviour of the process or reduce the system bandwidth.

We learned in previous chapters how to specify the system performance in terms of the frequency and time domain parameters. One of the important control design parameters is the phase margin, which gives an indication of the relative stability of the closed-loop system. In the following sections, we will discuss how to design phase lag and phase lead controllers to meet a specified phase margin.

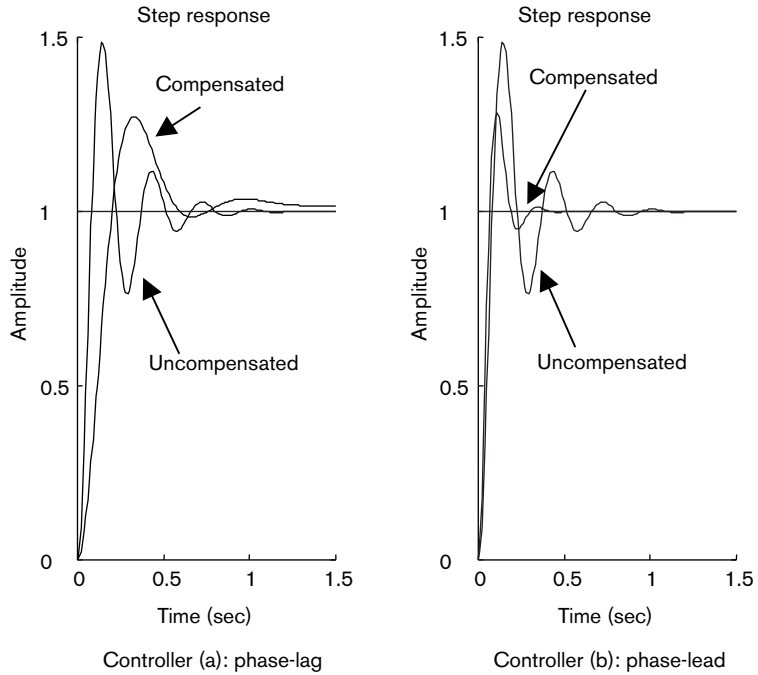


Figure 16.17 The step responses of the compensated and uncompensated systems.

16.5 Phase lag controller

A phase lag controller has the transfer function:

$$G_{lag}(s) = \frac{\tau s + 1}{\alpha \tau s + 1} \quad \alpha > 1$$

The corner frequencies of the controller are at $\omega_L = 1/(\alpha \tau)$ and $\omega_H = 1/\tau$. Typical Bode plots for the phase lag controller are shown in Figure 16.18 for two different values of α . For comparison purposes, we have also inserted the Bode plot of a pure integral term since this gives a good idea of why the phase lag properties are useful.

16.5.1 Relationship with the integral controller

The phase lag controller is used when the compensated system exhibits a satisfactory transient response but a poor steady state performance. This implies that the gain at low frequency should be increased without altering the transient response characteristics. Since the phase lag controller gain at low frequency is unity, the controller does not change the loop gain at *very low* frequencies. However, for low frequencies and large values of α , the magnitude of $|\tau s| = |\tau \omega| = (\tau \omega) \ll 1$ and so $(\tau s + 1) \cong 1$, and for large values of α , the magnitude of $|\alpha \tau s| = |\alpha \tau \omega| = (\alpha \tau \omega) \gg 1$ and so $(\alpha \tau s + 1) \cong \alpha \tau s$. The phase lag controller can then be simplified to:

$$G_{lag}(s) = \frac{1}{\alpha \tau s} = \frac{K_1}{s} \quad \text{where} \quad K_1 = \frac{1}{\alpha \tau}$$

Clearly, since the dynamic characteristic of the phase lag controller is similar to an integral controller in the low-frequency range, we use this controller for systems where the steady state error is not satisfactory. If we used integral control, we would suffer a -90° phase change over all frequencies, but the phase lag controller has the advantage that the phase lag at high frequencies is zero.

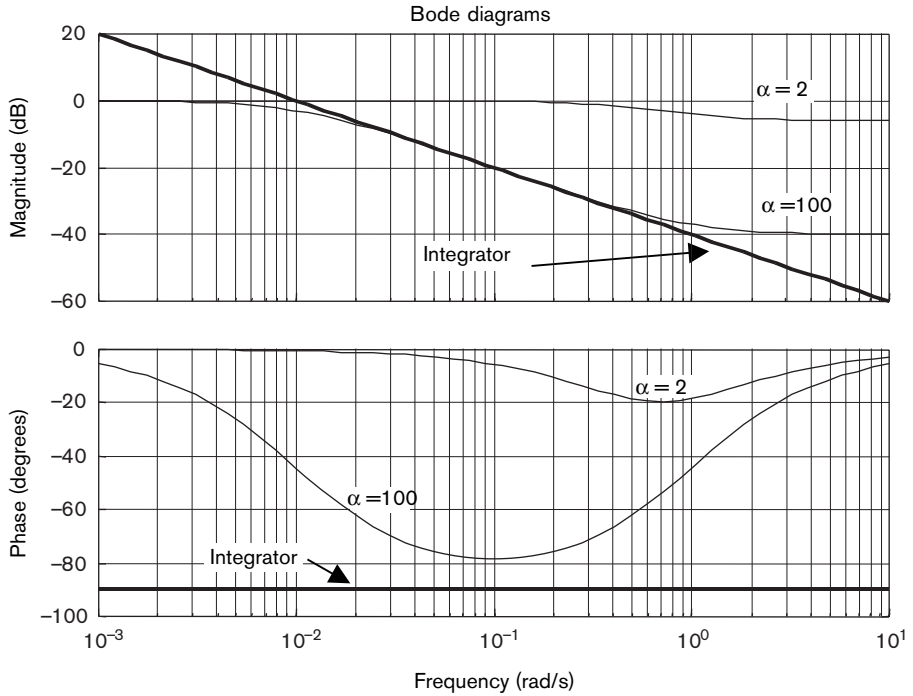


Figure 16.18 Phase lag controllers with different values of α and relationship with integral controller.

16.5.2 Phase lag design

We consider the general feedback system shown in Figure 16.19. We have introduced the controller $K(s) = KG_{\text{lag}}(s)$. The controller comprises two elements: a phase lag controller, $G_{\text{lag}}(s)$, and a gain, K .

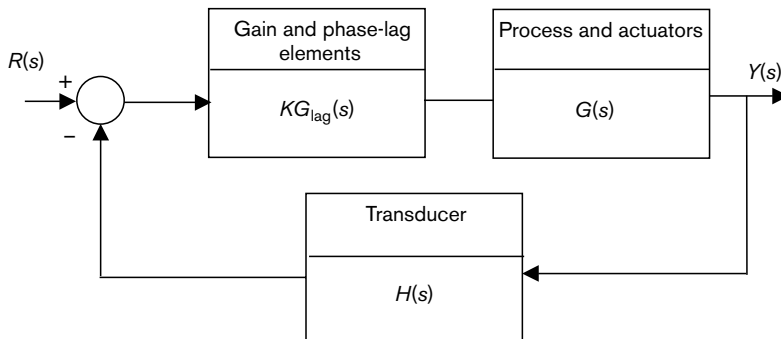


Figure 16.19 Closed-loop system for phase lag design.

16.5.3 Control design specification

The design specification is usually given in terms of constraints on the steady state error and phase margin, for example:

1. $e_{ss} \leq 0.01$ for a particular class of inputs
2. $PM \geq 45^\circ$

If the design requirement is on the damping ratio, then for systems with dominant second-order characteristics, we can translate this into an approximate phase margin requirement using the relationship:

$$\text{Phase margin (degrees)} \cong 100\zeta$$

16.5.4 Phase lag design procedure

A design procedure for a phase lag controller with a design specification on steady state error and phase margin is given here.

1. Assume the d.c. gain of $G_{\text{lag}}(s)$ is unity, $|G_{\text{lag}}(0)| = 1$, and determine the controller gain K in order that the closed-loop system

$$G_{\text{CL}}(s) = \frac{KG(s)}{1 + KG(s)H(s)}$$

satisfies the steady state error requirement. To do this we use the final value theorem to find an appropriate value for the gain K .

$$e_{ss} = \lim_{s \rightarrow 0} sE(s) = \lim_{s \rightarrow 0} \left\{ s \frac{1}{1 + KG(s)H(s)} R(s) \right\}$$

2. Use this gain, K , to draw the Bode plot of the open-loop transfer function with the transfer function, $KG(s)H(s)$, and find the system phase margin PM_{sys} .
3. If the design specification on phase margin is not met, we need to move the gain crossover frequency point to a location on the frequency axis, where we can obtain the required phase margin. We therefore find the frequency point ω_c , where:

$$\angle KG(j\omega_c)H(j\omega_c) = -180^\circ + PM_{\text{spec}} + \phi_c$$

where PM_{spec} is the specification for the phase margin and ϕ_c represents an additional phase of 5° . This additional phase is approximate and its aim is to compensate for the effect of any phase lag introduced by the controller at the frequency point ω_c .

4. Choose ω_c as the new crossover frequency. We select a value for the high corner frequency, ω_H , of the phase lag controller a decade down from this point:

$$\omega_H = \frac{\omega_c}{10}$$

The reasoning behind this is that any significant phase change caused by the phase lag controller should only occur in the range of a decade up or down from the corner frequencies. The controller parameter τ is then given by $\tau = 10/\omega_c$.

5. From the Bode plot of $KG(s)H(s)$, determine the reduction in gain, $K_{\text{red}}(\text{dB})$, necessary to bring the magnitude plot down to 0 dB at the new gain crossover frequency ω_c .

6. Determine α using the relation:

$$20 \log(\alpha) = K_{\text{red}}(\text{dB}) \rightarrow \alpha = 10^{K_{\text{red}}/20}$$

7. Calculate the phase lag controller:

$$G_{\text{lag}}(s) = \frac{\tau s + 1}{\alpha \tau s + 1}$$

8. Draw the Bode plot of the compensated system $KG_{\text{lag}}(s)G(s)H(s)$ and check the value of the phase margin. If the specification is not met, repeat from Step 3 but this time select different value for ϕ_c or decrease the value of ω_c .

9. The frequency domain design is now complete but now the time domain performance must be examined. Step and ramp response tests, disturbance rejection and noise rejection time responses are usual.

This procedure can be summarised as:

Step	Procedure
1.	Find K to satisfy e_{ss}
2.	Draw Bode plot of $KG(s)H(s)$. Find PM_{sys}
3.	From Bode plot find the new ω_{gco} point which meets desired $PM (+\phi_c)$
4.	Set $\tau = 10/\omega_c$
5.	Determine gain required to achieve movement of gain crossover, K_{red}
6.	Calculate $\alpha = 10^{K_{\text{red}}/20}$
7.	Calculate phase lag controller $G_{\text{lag}}(s) = \frac{\tau s + 1}{\alpha \tau s + 1}$
8.	Draw Bode plot of $KG_{\text{lag}}(s)G(s)H(s)$ and check specification met
9.	Examine time domain responses for acceptable performance

Problem The unity feedback system in Figure 16.20 represents a position control system where $Y(s)$ is the output signal in mm and $U(s)$ is the input signal in volts.

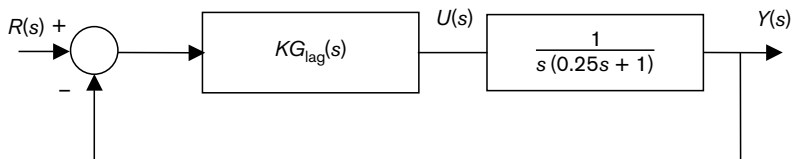


Figure 16.20 Position control system for phase lag design.

Design specifications

1. Steady state error e_{ss} to a ramp input is less than or equal to 0.01
2. $PM > 40^\circ$

Solution We note that the system is a Type 1 system (it has one integrator) and therefore gives zero steady state error to a step input. In this example, the specification is to track a ramp signal with a small but specified steady state error tolerance. There is also a requirement on phase margin to provide reasonable transient performance. We follow the phase lag design procedure with:

$$G(s) = \frac{1}{s(0.25s + 1)}$$

1. For a unit ramp input, $R(s) = 1/s^2$; hence:

$$e_{ss} = \lim_{s \rightarrow 0} sE(s) = \lim_{s \rightarrow 0} s \left(\frac{1}{1+KG(s)} \right) R(s) = \lim_{s \rightarrow 0} \left(s \frac{1}{1+K[1/s(0.25s+1)]} \right) \frac{1}{s^2} = \frac{1}{K}$$

Since $e_{ss} \leq 0.01$ is required, this gives $K \geq 100$.

We choose $K = 100$ since this is the minimum gain that satisfies the steady state error specification. An increase in K will result in higher overshoot.

2. The Bode plot of $KG(s)$ is given in Figure 16.21. The phase margin is approximately 12° , which does not meet the design specification.

M We can use the MATLAB commands

```
s=tf('s'); gsys=1/(s*(0.25*s+1)); K = 100;
margin(K*gsys)
```

to produce a Bode plot which has the gain and phase margin information on the plot.

3. We need to find the frequency where the $PM = PM_{sys} + \phi_c = 45 + 5 = 50^\circ$. From Figure 16.21, we see that this occurs when ω_{gco} is moved to 3.5 rad/s. We let $\omega_c = 3.5$ rad/s.

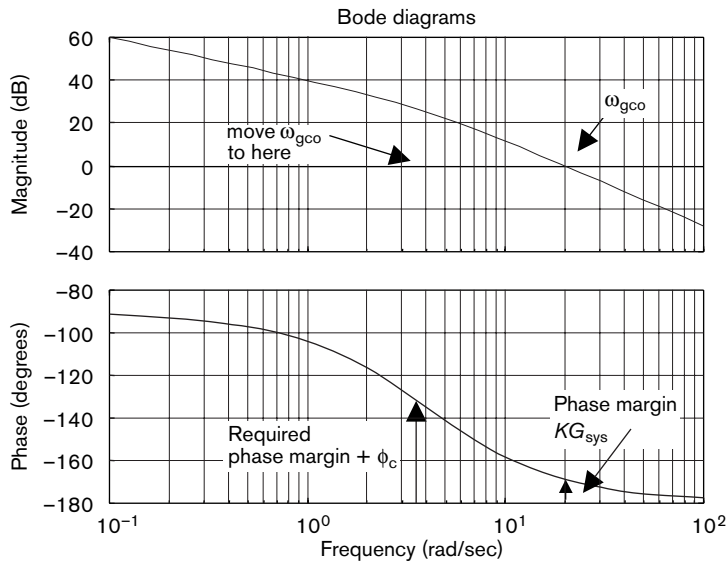


Figure 16.21 Bode plot of $KG_{sys}(s)$.

4. Set the τ value for the compensator at $\tau_c = 10/\omega_c = 2.86$ seconds.

5. To move the frequency response line down so that it passes 0 dB at $\omega_c = 3.5$ rad/s, we need to reduce the gain by approximately 27 dB, so set $K_{red} = 27.0$ dB.
6. Determine α :

$$\alpha = 10^{27/10} = 22.4$$

7. Calculate phase lag controller:

$$G_{lag}(s) = \frac{\tau s + 1}{\alpha \tau s + 1} = \frac{2.86s + 1}{64.1s + 1}$$

8. The Bode plot for the compensated system is given in Figure 16.22. We can see that the phase margin is increased to above 40° . The gain crossover is reduced, so the system will respond more slowly.

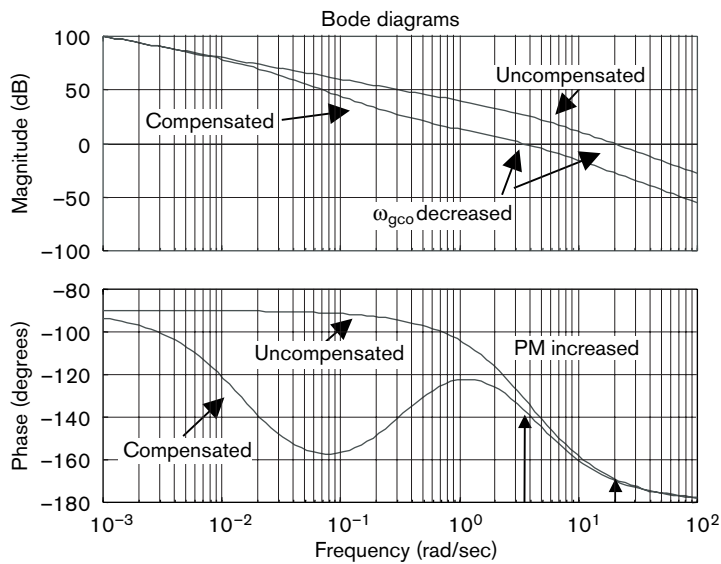


Figure 16.22 Bode plot of original system and compensated system.

9. The step response for the closed-loop system is seen in Figure 16.23. The compensated system responds more slowly with a long settling time, but has less overshoot and is less oscillatory. The phase lag controller reduces the bandwidth of the closed-loop system. This results in slower transient response.

16.6 Phase lead control design

A phase lead controller has the transfer function:

$$G_{lead}(s) = \frac{\tau s + 1}{\beta \tau s + 1} \quad \beta < 1$$

with the low and high corner frequencies given by:

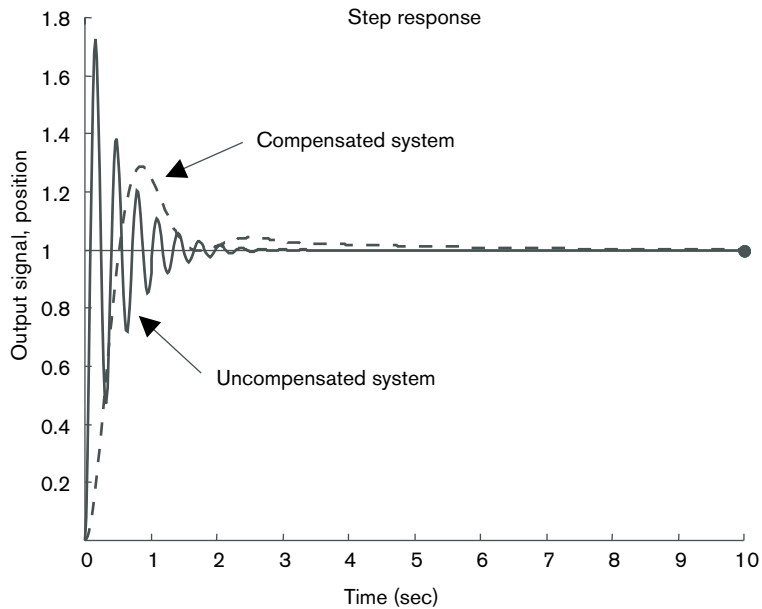


Figure 16.23 Closed-loop step responses for compensated and uncompensated systems.

$$\omega_L = \frac{1}{\tau} \quad \text{and} \quad \omega_H = \frac{1}{\beta\tau}$$

Typical Bode plots for a phase lead controller are shown in Figure 16.24. We see that the low-frequency gain is unity (0 dB) and we can calculate the high-frequency gain from

$$20 \log_{10}(1/\beta)$$

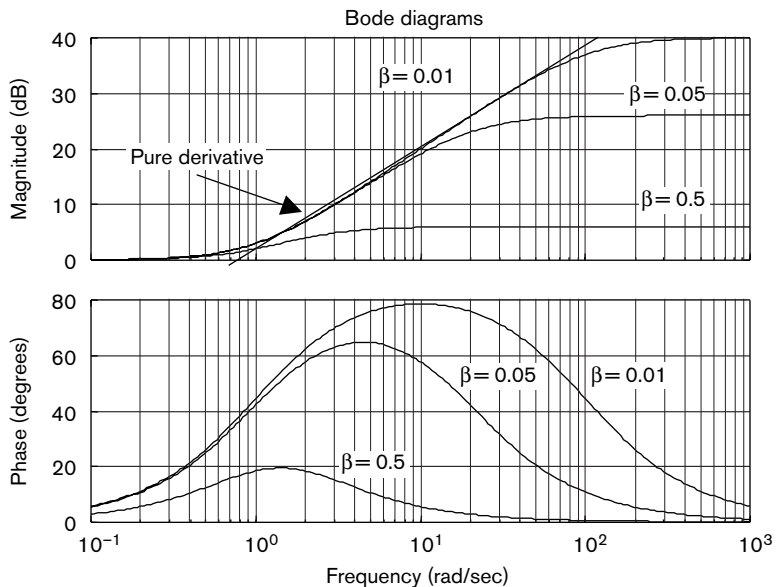


Figure 16.24 The frequency response of the lead controller for decreasing β .

For small values of $\beta \ll 1$, the phase lead controller gain plot approaches the gain plot of a pure derivative term in the mid-frequency range, as shown in Figure 16.24.

The phase plot of the phase lead controller starts and ends at 0° , but the location of the maximum phase peak depends on the value of β . We can calculate the value of this peak phase from the expression for $\angle G_{\text{lead}}(j\omega)$:

$$\phi = \angle G_{\text{lead}}(j\omega) = \tan^{-1}(j\tau\omega + 1) - \tan^{-1}(j\beta\tau\omega + 1)$$

The maximum phase lead, ϕ_m , can be found using calculus and ϕ_m occurs at the frequency, ω_m , where

$$\omega_m = \frac{1}{\tau\sqrt{\beta}}$$

The peak phase value is given by the expression

$$\phi_m = \sin^{-1}\left(\frac{1-\beta}{1+\beta}\right)$$

For a given maximum phase lead, ϕ_m , we can calculate the controller parameter, β , as

$$\beta = \frac{1 - \sin(\phi_m)}{1 + \sin(\phi_m)}$$

We note that the value of β depends solely on ϕ_m . We will use this relationship in the phase lead design procedure.

16.6.1 Relationship with the derivative controller

For small values of $\beta \ll 1$, the phase lead controller transfer function can be approximated by:

$$G_{\text{lead}}(s) = \frac{\tau s + 1}{\beta\tau s + 1} \approx (\tau s + 1) \approx \tau s$$

for $\beta \ll 1$ and values of ω in the mid- to high-frequency range. Hence

$$G_{\text{lead}}(s) \approx K_D s \quad \beta \ll 1$$

where $K_D = \tau$. The phase lead controller will therefore approach a pure derivative controller as the value of β approaches zero. Since the dynamic characteristic of the phase lead controller is similar to a derivative controller in the middle and high-frequency ranges, we use this controller to improve the systems with poor stability characteristics. The phase lead controller has the advantage that the phase lead at low frequencies is zero, whereas a pure derivative controller has 90° phase lead over all frequencies. This implies that we can design a phase lead controller while minimising its effects on the low-frequency response of the system. Therefore the phase lead controller is used when the open-loop system is either unstable or has undesirable transient response. The phase lead controller changes the shape of the frequency response at high frequencies.

The maximum practical phase lead for a one-stage phase lead controller is 60° to 70° . If we require a higher phase lead, we can use a two-stage phase lead controller by cascading two one-stage controllers.

16.6.2 Phase lead design procedure

We describe the design procedure related to a unity feedback system, as shown in Figure 16.25.

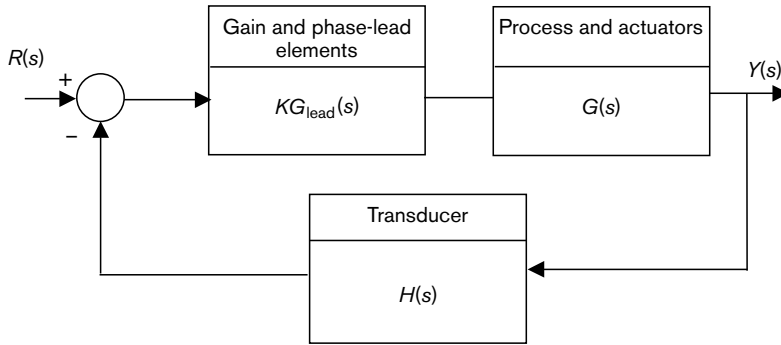


Figure 16.25 The phase lead control system.

Design specification

As for the phase lag design, the design specification is often given in terms of constraints on the steady state error and phase margin, for example:

1. $e_{ss} \leq 0.01$ to a particular class of inputs
2. $PM \geq 45^\circ$

Design procedure

1. The d.c. value of $G_{lead}(s)$ is unity, $|G_{lead}(0)| = 1$, and determine the controller gain K in order that the closed-loop system

$$G_{CL}(s) = G_{CL}(s) = \frac{KG(s)}{1 + KG(s)H(s)}$$

satisfies the steady state error requirement. We recall that we can use the final value theorem to find an appropriate value for the gain K .

$$e_{ss} = \lim_{s \rightarrow 0} sE(s) = \lim_{s \rightarrow 0} \left\{ s \frac{1}{1 + KG(s)H(s)} R(s) \right\}$$

2. Use this gain, K , to draw the Bode plot of the open-loop system $KG(s)H(s)$ and find PM_{sys} . For unstable systems, PM_{sys} will be negative.
3. If the design specification on phase margin is not met, then find the additional phase lead to be added to the system in order to meet the specification

$$\phi_m = (PM_{spec} + \phi_c) - PM_{sys}$$

where an extra phase lead of $\phi_c = 5^\circ$ to 20° is required to compensate for the shift in the crossover frequency. The level of this extra phase lead depends on the cut-off rate of the uncompensated system. For a -60 dB/decade cut-off rate, the extra value is about 12° .

4. Determine the factor β using the relationship:

$$\beta = \frac{1 - \sin(\phi_m)}{1 + \sin(\phi_m)}$$

5. The phase lead controller introduces a maximum phase of ϕ_m dependent on the value of β just calculated. The phase margin of the system is measured at the existing gain crossover point, ω_{gco} , of the system and we would like to place the additional phase close to this gain crossover to satisfy the phase margin specifications. The peak phase of the compensator occurs at a frequency of ω_m , which is unknown at present, but we know that the phase lead magnitude at this mid-frequency point is given by $[20 \log_{10}(1/\beta)]/2 = 10 \log_{10}(1/\beta)$. Thus we have to make sure that the system has a gain crossover frequency at ω_m in order to have maximum phase shift added at this frequency. To make ω_m the new gain crossover frequency, we must select the point where the gain of uncompensated system is $|KG(s)|_{dB} = -10 \log_{10}(1/\beta)$. Then, when we add in the compensator, the controller gain of $10 \log_{10}(1/\beta)$ cancels out with the system gain of $-10 \log_{10}(1/\beta)$ to give the new gain crossover point at $\omega_m = \omega_{gco}$. This point can be read from the Bode plot of the uncompensated system.
6. Calculate the phase lead controller time constant τ using the relationship:

$$\tau = \frac{1}{\omega_m \sqrt{\beta}}$$

7. Calculate the phase lead controller:

$$G_{lead}(s) = \frac{\tau s + 1}{\beta \tau s + 1}$$

8. Draw the Bode plot of the compensated system $KG_{lead}(s)G(s)H(s)$, check the resulting phase margin and repeat the steps if necessary.
9. Time domain responses, steps, ramps, supply and load disturbance responses and noise rejection simulations should be performed and assessed.

Step	Phase lead design procedure
1.	Find K to satisfy e_{ss}
2.	Draw Bode plot of $KG(s)H(s)$. Find PM_{sys}
3.	Find additional phase ϕ_m required to meet desired $PM (+\phi_c)$: $\phi_m = PM_{spec} + \phi_c$
4.	Calculate $\beta = \frac{1 - \sin(\phi_m)}{1 + \sin(\phi_m)}$
5.	Determine the frequency point ω_m at which the gain is $-10 \log_{10}(1/\beta)$
6.	Calculate $\tau = \frac{1}{\omega_m \sqrt{\beta}}$
7.	Calculate phase lead controller $G_{lead}(s) = \frac{\tau s + 1}{\beta \tau s + 1}$
8.	Draw Bode plot of $KG_{lead}(s)G(s)H(s)$ and check specification met
9.	Examine the time domain responses for acceptable performance assessment

Problem We look at the unity feedback system in Figure 16.26.

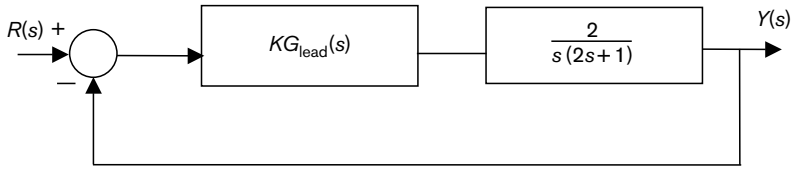


Figure 16.26 Phase lead control system.

Design specifications

1. Steady state error e_{ss} to a unit ramp input is less than or equal to 0.5
2. $PM \geq 45^\circ$

Solution We use the transfer functions

$$G_{sys} = \frac{2}{s(2s+1)} \quad H(s) = 1$$

$$\begin{aligned}
 1. \quad e_{ss} &= \lim_{s \rightarrow 0} sE(s) = \lim_{s \rightarrow 0} s \left(\frac{1}{1 + KG_{sys}(s)} \right) R(s) \\
 &= \lim_{s \rightarrow 0} s \left(\frac{1}{1 + \frac{2K}{s(2s+1)}} \right) \frac{1}{s^2} = \lim_{s \rightarrow 0} s \left(\frac{s(2s+1)}{s(2s+1) + 2K} \right) \frac{1}{s^2} \\
 &= \frac{1}{2K}
 \end{aligned}$$

Since we require $e_{ss} \leq 0.5$, we must select $K \geq 1$.

2. The Bode plot of the transfer function KG_{sys} is given in Figure 16.27. We find that the phase margin is approximately 28° .
3. We calculate the peak phase needed as $\phi_m = (PM_{spec} + \phi_c) - PM_{sys} = (45 + 10) - 28 = 27^\circ$
The extra phase, $\phi_c = 10^\circ$, is added to compensate for the shift in the crossover frequency.
4. Determine the factor β :

$$\beta = \frac{1 - \sin(27^\circ)}{1 + \sin(27^\circ)} = 0.376$$

If we use MATLAB for the calculation, we must remember that the \sin function assumes an input in radians, as do many calculators! Therefore in MATLAB we would use

$$\beta = \frac{1 - \sin(27 \times \pi / 180)}{1 + \sin(27 \times \pi / 180)} = 0.376$$

5. The uncompensated gain that we need to find is given by

$$-10 \log_{10}(1/\beta) = -10 \log_{10}(1/0.376) = -4.25 \text{ dB}$$

Looking at the uncompensated system Bode plot we find this gain occurs at $\omega_m = 1.2 \text{ rad/s}$.

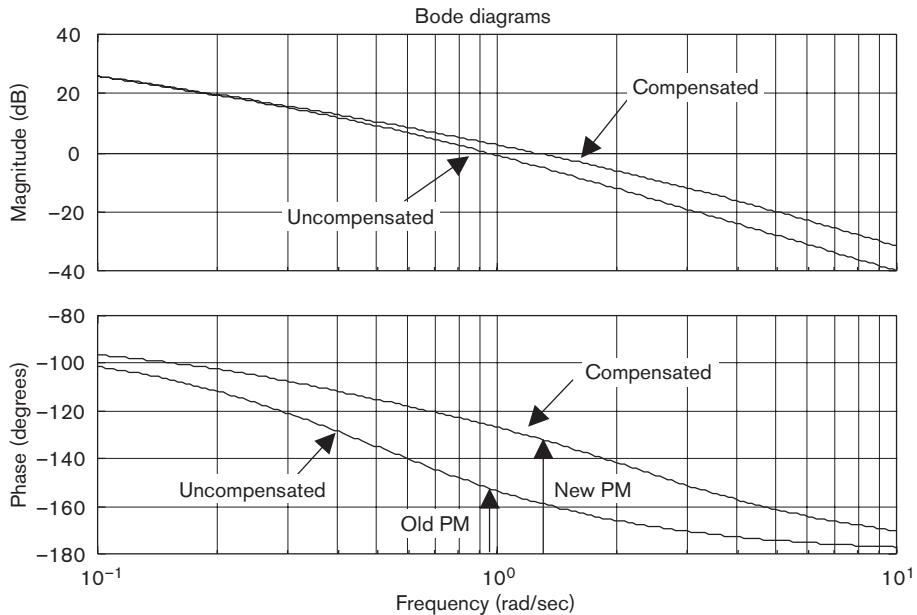


Figure 16.27 Bode plots of phase lead design example.

6. Calculate the time constant τ using the relationship:

$$\tau = \frac{1}{\omega_m \sqrt{\beta}} = \frac{1}{1.2 \sqrt{0.376}} = 1.359$$

7. Calculate the phase lead controller:

$$G_{\text{lead}}(s) = \frac{1.359s + 1}{0.511s + 1}$$

8. We check the phase margin of the compensated system from the plot in Figure 16.27. The new phase margin is about 47° . This meets the required phase margin.
9. The step response of the closed-loop compensated system is given in Figure 16.28. We see that the peak overshoot has been reduced from 1.4 to 1.2. The speed of response of the compensated system has increased slightly due to the higher gain crossover incurred when we included the lead compensator.

Remark

We can also see that the compensated system has a larger bandwidth and hence the step response is faster. This has the disadvantage that noise amplification will increase. Sometimes, to alleviate this increase of gain at high frequency, we find that the lead compensator has been written as

$$G_{\text{lead}}(s) = \beta \frac{\tau s + 1}{\beta \tau s + 1}$$

This provides a compensator which has a lower gain at low frequencies but has a unity gain at high frequencies. However, this change in gain would cause any previous calculation on K to meet a steady state error specification to be in error, and we would need to modify the steady state error calculation accordingly.

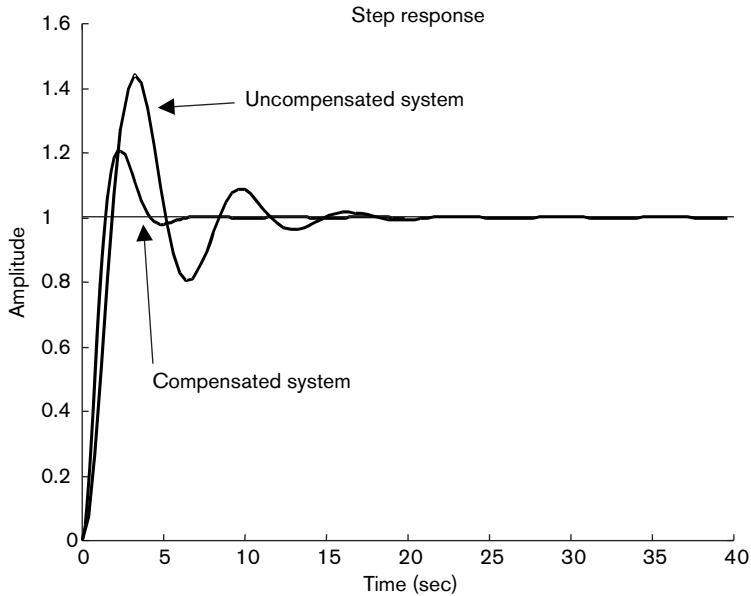


Figure 16.28 Step response of phase lead compensated system.

16.7 Design is iterative: a cautionary tale

We now look at a phase lead design where the procedure is followed but the design specification cannot be met.

Problem We consider designing a phase lead compensator for the system in Figure 16.29.

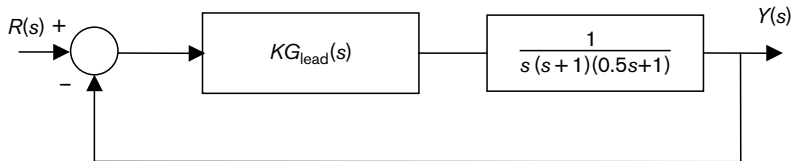


Figure 16.29 System for phase lead design.

Design specifications

1. Steady state error e_{ss} to a unit ramp input is less than 0.05
2. $PM > 45^\circ$

Solution We follow the phase lead design procedure but present the results more briefly.

$$\begin{aligned}
 1. \ e_{ss} &= \lim_{s \rightarrow 0} sE(s) = \lim_{s \rightarrow 0} s \left(\frac{1}{1 + KG_{sys}(s)} \right) R(s) \\
 &= \lim_{s \rightarrow 0} s \left(\frac{1}{1 + K / [s(s+1)(0.5s+1)]} \right) \frac{1}{s^2}
 \end{aligned}$$

$$\begin{aligned}
 &= \lim_{s \rightarrow 0} s \left(\frac{s(s+1)(0.5s+1)}{s(s+1)(0.5s+1)+K} \right) \frac{1}{s^2} \\
 &= \frac{1}{K}
 \end{aligned}$$

Since we require $e_{ss} \leq 0.05$, we select $K \geq 2$.

Design exercise 1

2. The Bode plot of KG_{sys} is shown in Figure 16.30. The phase margin is approximately 18° .
3. $\phi_m = (PM_{spec} + \phi_c) - PM_{sys} = (45 + 5) - 18 = 32^\circ$
4. $\beta = \frac{1 - \sin(32^\circ)}{1 + \sin(32^\circ)} = 0.307$
5. The frequency at which the uncompensated gain is $-10 \log_{10}(1/\beta) = -10 \log_{10}(1/0.307) = -5.13 \text{ dB}$ occurs at $\omega_m = 1.5 \text{ rad/s}$.
6. $\tau \frac{1}{\omega_m \sqrt{\beta}} = \frac{1}{1.5 \sqrt{0.307}} = 1.203$
7. $G_{lead}(s) = \frac{1.203s + 1}{0.369s + 1}$
8. The Bode plot for $KG_{lead}(s)G_{sys}$ is shown as 'Design 1' in Figure 16.30. The phase margin is much the same as before at 18° .

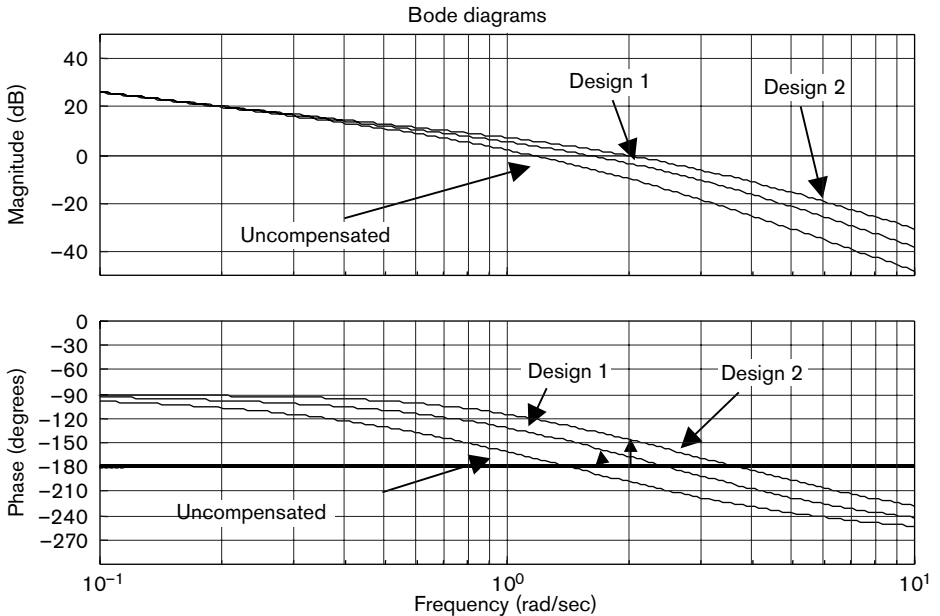


Figure 16.30 Phase lead compensation of $2/[s(s+1)(0.5s+1)]$.

We try again, but this time increase the value of ϕ_c to 25° .

Design exercise 2

2. The phase margin is approximately 18° as in Step 2 above.

$$3. \phi_m = (PM_{\text{spec}} + \phi_c) - PM_{\text{sys}} = (45 + 25) - 18 = 52^\circ$$

$$4. \beta = \frac{1 - \sin(52^\circ)}{1 + \sin(52^\circ)} = 0.119$$

5. The frequency at which the uncompensated gain is

$$-10 \log_{10}(1/\beta) = -10 \log_{10}(1/0.119) = -9.24 \text{ dB}$$

occurs at $\omega_m = 1.9 \text{ rad/s}$.

$$6. \tau = \frac{1}{\omega_m \sqrt{\beta}} = \frac{1}{19 \sqrt{0.119}} = 1.53$$

$$7. G_{\text{lead}}(s) = \frac{1.53s + 1}{0.182s + 1}$$

8. The Bode plot for $KG_{\text{lead}}(s)G_{\text{sys}}$ is shown as 'Design 2' in Figure 16.30. The phase margin has increased to around 30° , but this still does not meet the design specification.

So why does our design procedure fail? If we look closely at the phase of the original system, we see that it is decreasing fairly rapidly around the gain crossover point. Although we add extra phase at this point, the phase lead compensator will move the gain crossover of the compensated system to a higher frequency where the phase has fallen to a lower value. We try to compensate for this by our 'additional phase' of ϕ_c , but in cases where the phase is decreasing rapidly, the movement of the gain crossover frequency to a higher value prevents the compensated system from achieving the desired specification. In such examples we would have to consider modifying the controller, for example by considering a phase lag controller instead.

16.8 Summary of the effects of phase lag and phase lead controllers on system responses

We provide a summary table (Table 16.2) on the particular features of the phase lead and phase lag controllers.

16.8.1 Phase lag-lead controllers

We learnt that the phase lag controller can be used to adjust the steady state error by reducing the loop bandwidth and slowing down the system transient response. On the other hand, we used the phase lead controller to increase the loop bandwidth and make the system faster. For high-order systems or systems with large steady state error, a phase lead controller may result in very large bandwidth, which may not be acceptable in practice due to noise amplification. For these problems we occasionally use a phase lag-lead controller, where the lead section can be used to adjust the loop bandwidth and the lag

Table 16.2 Summary table for phase lag and phase lead controller features.

	Phase lag compensator	Phase lead compensator
Phase and gain at low frequencies	0°, 0 dB	0°, 0 dB
Phase and gain at high frequencies	0° $20 \log_{10}(1/\alpha) < 0 \text{ dB} \quad \alpha > 1$	0° $20 \log_{10}(1/\beta) > 0 \text{ dB} \quad \beta < 1$
Compensator behaviour similar to:	PI at low frequency	PD controller
Stability margins	Usually lower than uncompensated system	Usually better than uncompensated system
Closed-loop bandwidth	Reduced	Increased
Speed of response	Reduced	Increased
Specific uses	Used in systems where steady state error not satisfactory	Used to improve system stability

section can be used to provide additional phase margin. The design of the lag-lead compensator is a combination of the phase lag and phase lead procedures.

A phase lag-lead controller has the following transfer function:

$$G_{\text{lag-lead}}(s) = \left(\frac{\tau_{\text{lag}}s + 1}{\alpha\tau_{\text{lag}}s + 1} \right) \left(\frac{\tau_{\text{lead}}s + 1}{\beta\tau_{\text{lead}}s + 1} \right)$$

We assume $\alpha = 1/\beta$ so that the gain is 0 dB at low frequencies. The design procedure for a lag-lead controller follows the procedure for a lag controller initially; then, once α has been calculated, the value of β in the phase lead section is constrained by the equation $\beta = 1/\alpha$.

We have presented several designs in this chapter which looked at shaping the open-loop frequency response on the Bode plot. The design specifications were often those related to open-loop criteria. In the next chapter we look at design based on the use of closed-loop specifications and how we use a graphical method to relate open- and closed-loop information. We do this on the Nichols chart.

What we have learnt

- ✓ To use our knowledge of Bode plots to design simple controllers.
- ✓ To learn how the open-loop frequency response plots can be shaped by cascading appropriate controllers
- ✓ To design phase lag and phase lead controllers of the form.

$$G_{\text{lag}}(s) = \frac{\tau s + 1}{\alpha\tau s + 1} \quad \alpha > 1$$

$$G_{\text{lead}}(s) = \frac{\tau s + 1}{\beta \tau s + 1} \quad \beta < 1$$

- ✓ To recognise when either a phase lag or phase lead control design would be suitable.

Multiple choice

- M16.1** Adding a lag term to the controller transfer function will:
- increase the roll-off rate
 - decrease the roll-off rate
 - increase the system's speed of response
 - alter the steady state error to a step input
- M16.2** Adding a lead term to the open-loop transfer function will:
- have no effect on the steady state error
 - reduce the phase by -90° in the region of the lead term's breakpoint
 - reduce the phase by -180° in the region of the lead term's breakpoint
 - increase the phase by $+180^\circ$ in the region of the lead term's breakpoint
- M16.3** Increasing the controller gain will cause:
- the gain crossover frequency to reduce
 - the gain crossover frequency to increase
 - the system to respond more slowly
 - none of the above
- M16.4** A controller transfer function is given by $K(s) = (2s + 1)/(0.2s + 1)$. Is $K(s)$ a lag or lead controller and what is the corresponding value of α or β ?
- Lag controller, $\alpha = 10$
 - Lag controller, $\alpha = 2$
 - Lead controller, $\beta = 0.1$
 - Lead controller, $\beta = 0.2$
- M16.5** A ship autopilot has several tuning knobs to change the performance of the controller. The captain finds that the ship's response to heading change is slow.
- the captain should increase the controller lag action
 - the captain should decrease the controller lag action
 - the captain should decrease the controller lead action
 - the captain should decrease the controller gain
- M16.6** A lead-compensated controller is used to control the angular velocity of a d.c. servo system. The response has an unacceptable level of overshoot. To reduce the overshoot:
- the controller should be replaced by a lag controller
 - the gain of the lead controller should be increased
 - the gain of the lead controller should be decreased
 - the time constant of the lead controller should be decreased
- M16.7** Which statement is correct?
- a PI controller is a lag controller and a PD controller is a lead controller
 - PI and PD controllers are lag controllers
 - a PI controller is a lead controller and a PD controller is a lag controller
 - PI and PD controllers are lead controllers
- M16.8** A lift control system exhibits a large steady state error. To improve the performance, we should:
- reduce the controller gain
 - increase the controller gain
 - add a lead controller
 - reduce the controller gain at high frequencies
- M16.9** The closed dynamics of a car suspension system can be represented by a second-order system. The controller used is a lead controller. To improve the damping ratio, we should:
- decrease the phase margin
 - decrease the gain margin
 - decrease the phase and gain margin
 - increase the phase margin

M16.10 You are asked to design a controller for a water tank system which has sluggish dynamics. What controller do you choose?

- (a) a constant gain controller
- (b) a lag controller
- (c) a PI controller
- (d) a lead controller

Questions: practical skills

Q16.1 Consider a unity feedback system where the open-loop transfer function is given by

$$G(s) = \frac{K}{s(s+1)^2}$$

- (a) Find the phase margin of the system for the values of gains $K = 0.1, 1, 2$ and 10 .
- (b) Comment on the relationship between K and the phase margin.

Q16.2 A unity feedback positioning servo system has the open-loop transfer function given by

$$G(s) = \frac{K}{s(s+1)}$$

- (a) For the following values of gains, $K = 0.1, 0.3, 1$ and 2 , construct a table containing
 - (i) PM
 - (ii) % overshoot for closed-loop step response
- (b) What is the relationship between overshoot and phase margin for this system?

Q16.3 The model of a spacecraft attitude controller is given by $G(s) = 1/s^2$. An engineer has designed a lead controller with the transfer function

$$H(s) = \frac{0.0043(0.3s+0.004)}{(0.04s+0.03)}$$

Plot the frequency response of $G(s)$ and $G(s)H(s)$ and calculate the GM and PM of both systems to see the improvement achieved due to the lead controller.

Q16.4 A lag compensator with the transfer function $H(s) = (s+1)/(6s+1)$ is designed to control a unity feedback system with the plant transfer function given by

$$G(s) = \frac{200000}{(s+10)(s+10)(s+100)}$$

Find:

- (a) the steady state error to a unit step input
- (b) the overshoot of the closed-loop system

Q16.5 Consider the following two controllers:

$$K_1(s) = \frac{10(s+0.5)}{(s+5)} \quad K_2(s) = \frac{0.1(s+1)}{(s+0.05)}$$

- (a) Find the lead (α, τ) or lag (β, τ) controller parameters.
- (b) Is $K_1(s)$ or $K_2(s)$ a lead or a lag controller?

Problems

P16.1 Design a lead compensator for the unity feedback system

$$G(s) = \frac{200}{s(s+1)}$$

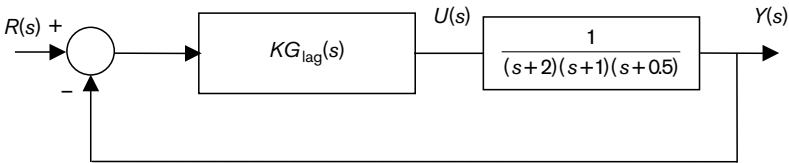
to give a damping ratio of 0.55. Comment on your results

P16.2 For the unity feedback system below, design a lag controller such that

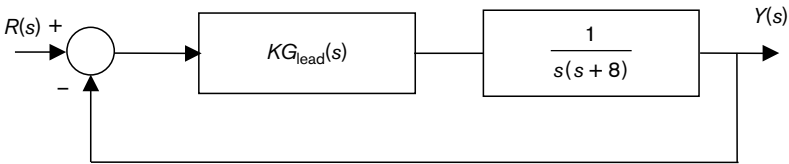
(i) $e_{ss} \leq 5\%$

and

(ii) $PM \geq 30^\circ$



P16.3 Design a lead compensator for the following d.c. servo system such that the steady state error to a unity rate ramp is less than 0.1 and the damping ratio is greater than 0.5.



P16.4 The open-loop transfer function of a mechanical system is represented by:

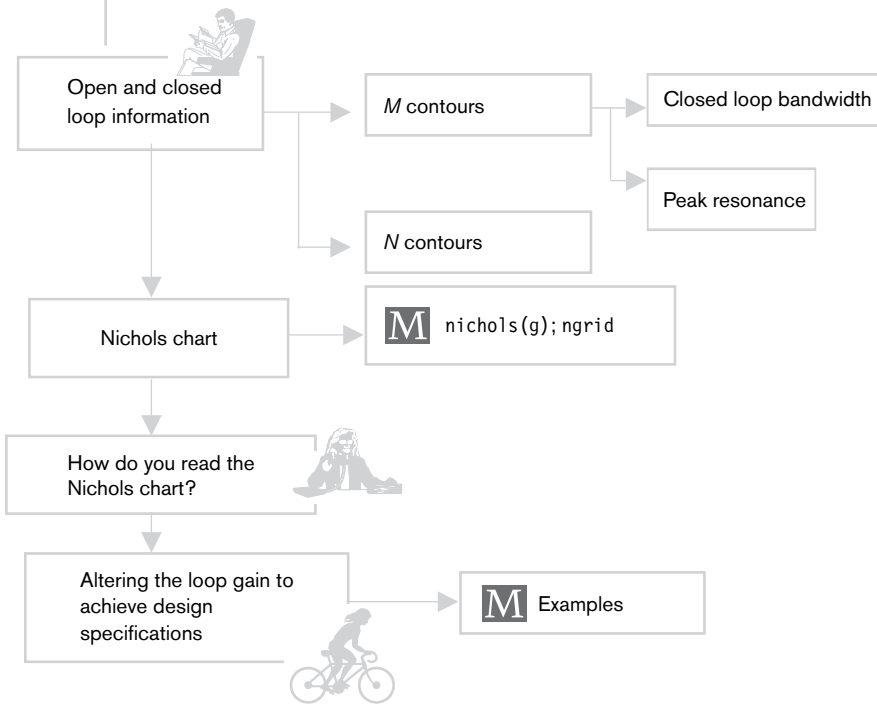
$$G(s) = \frac{4s+2}{(s+1)^3}$$

(a) Use `r1tool` to study the stability of the closed-loop system.

(b) Define a lead compensator in such a way that the dominant root locus moves further away from the imaginary axis.

17

Analysis and simple design using the Nichols chart



Gaining confidence



Help?



Going deeper



Skill section



Time to read

In previous chapters on Bode plot design, we met a class of control design problems which were based on manipulating the open-loop transfer function characteristics to achieve some desired closed-loop performance. For example, we adjusted the open-loop phase to improve the closed-loop system stability margins. We then have to form the closed-loop transfer function and check whether the open-loop control design meets the design specification. Hence there is a need to establish a relationship between the open-loop characteristics, $G_{OL}(j\omega)$, and the closed-loop characteristics, $G_{CL}(j\omega)$, in our control design problems. In many cases we have complicated models or sets of frequency response data and the problems of manipulating unwieldy transfer functions or frequency response data files are not easy. However, control design packages like MATLAB make the calculation of response information as simple as typing

$$gc1 = go1/(1+go1);$$

where $go1$ and $gc1$ are the open and closed-loop transfer functions respectively. However, traditionally the problem of gaining closed-loop information easily was solved by using a Nichols chart. The Nichols chart is still part of control design packages, since it gives valuable information by graphically relating open-loop and closed-loop information on one plot. We find that Nichols charts are less commonly used than the Bode plot techniques we have been dealing with, but particular industries, such as the aerospace industry, often use Nichols charts to a greater degree. We present here an introduction to the Nichols chart and give several examples of simple design.

We use the feedback control system of Figure 17.1, which has a controller $K(s)$, an actuator and process transfer function given by $G(s)$ and a measurement block given by $H(s)$. We would commonly have the reference $R(s)$ and the output signal $Y(s)$ in the same units and therefore we often introduce a scaling block, $H(s)$, which is similar to the measurement gains, on the reference input.

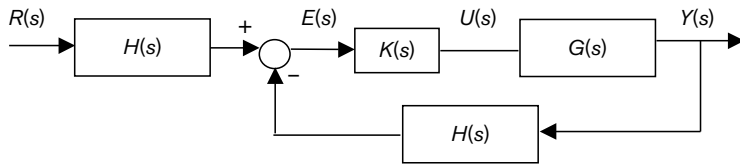


Figure 17.1 The general feedback system.

The open loop transfer function, $G_{OL}(s)$, is

$$G_{OL}(s) = K(s)G(s)H(s)$$

We can write the closed-loop transfer function, from $R(s)$ to $Y(s)$, as:

$$G_{CL}(s) = \frac{\text{forward}}{1 + \text{open loop}} = \frac{K(s)G(s)H(s)}{1 + G_{OL}(s)}$$

It can be rather difficult and cumbersome to find an explicit relationship between these two transfer functions for a general system. Hence we use a graphical technique known as the Nichols chart and show how it can be used for analysis and simple design. In essence we follow the procedure shown in Figure 17.2.

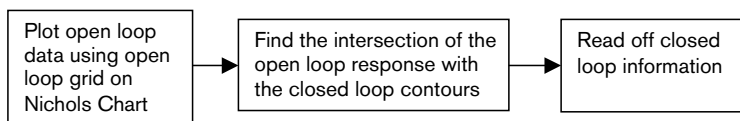


Figure 17.2 Nichols chart method.

This is very much like using an explorer's chart or map; the contours will be a guide to the landscape or features of the closed-loop system. For example, in Figure 17.3 we can find a grid-reference for the point marked X. Using the gridlines at the side of the map, X is at 48.6 and 78.8. By using the contours, which give the height above sea level, we can find that the height at point X is 310 m.

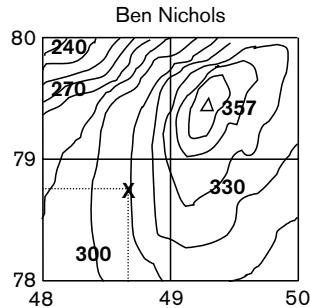


Figure 17.3 Contours on explorer's map.

Learning objectives

- To establish a relationship between the open-loop and closed-loop frequency responses.
- To understand the Nichols chart and learn how to extract open-loop and closed-loop control design parameter values.
- To be able to use the Nichols chart to make simple adjustments to the gain, phase margin and gain margin.
- To learn how to generate the Nichols chart using MATLAB.

17.1 Adding closed loop information to a Nichols plot

Firstly, we revise our knowledge of the Nichols plot by using Figure 17.4.

From the Nichols plot we see that the x-axis represents the open-loop phase and the y-axis shows the open-loop gain. For example, we can find that at the gain crossover (0 dB) the phase is approximately -145° . There is no frequency axis on the plot; frequency is an implicit parameter on the graph, every point on the actual response line representing a different frequency point. In the example shown in the plot, we have marked in some frequency points and we can see that as frequency increases, ω_1 to ω_3 , the open-loop phase decreases from -120° to -180° to -210° . The open-loop magnitude initially rises from 2 dB to a peak of about 7 dB; thereafter it falls to a level of -40 dB on this graph. All the information we find from the plot is related to the open-loop gain and phase. However, by placing a set of contours of closed-loop information on top of this plot we can read both open and closed-loop information from the one frequency response map, which we call a *Nichols chart*.

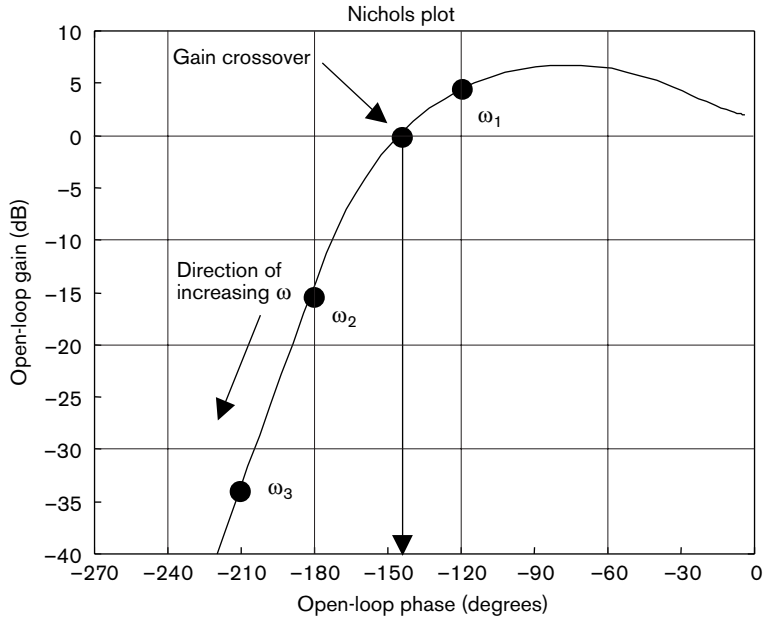


Figure 17.4 Nichols plot.

To calculate the contour lines for the closed-loop information, we consider the cases for the gain and phase separately:

$$\text{Gain: } |G_{CL}(j\omega)| = \left| \frac{G_{OL}(j\omega)}{1+G_{OL}(j\omega)} \right|$$

$$\text{Phase: } \angle G_{CL}(j\omega) = \angle \frac{G_{OL}(j\omega)}{1+G_{OL}(j\omega)}$$

If we let the open-loop frequency response, $G_{OL}(j\omega)$, have the real and imaginary parts given by

$$G_{OL}(j\omega) = X(\omega) + jY(\omega)$$

then the magnitude and phase of $G_{CL}(j\omega)$ can be calculated in terms of $X(\omega)$ and $Y(\omega)$.

17.1.1 Closed-loop gain – M contours

If we consider $|G_{CL}(j\omega)| = \text{constant} = M$, then the contour lines of constant closed-loop gain turn out to be circles in the real/imaginary axes which can then become the magnitude contours shown on the Nichols plot (Figure 17.5). We do not show the formal analysis for this as we always use a computer or pre-printed Nichols chart paper.

The **M -contours** are often referred to as M -circles, although they often lose their 'shape' unless imposed on a square grid in the Nyquist plane.

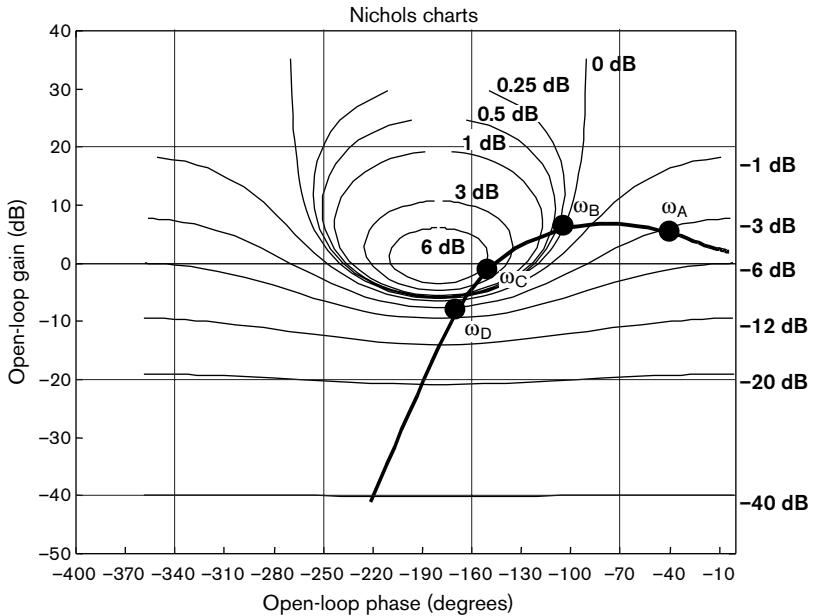


Figure 17.5 M -contours on Nichols plot.

Reading closed-loop bandwidth

From Figure 17.5 we see that the system frequency response line crosses the -3 dB contour (closed-loop value) at the frequency point ω_D when the open-loop gain and phase would be about -7 dB and -170° , respectively.

Reading closed-loop peak magnitude

The response line never quite reaches the 6 dB closed-loop magnitude and therefore the closed-loop peak magnitude would be around 5 dB at frequency point ω_C . We can verify this by looking at the closed-loop magnitude Bode plot (Figure 17.6) to check the closed-loop values. We see that the response line starts at -5 dB, crosses the 0 dB line and is almost tangential to the 5 dB line. This is also true of the frequency response line in Figure 17.5, where the lines of constant magnitude are now M -contours. The points ω_A , ω_B , ω_C and ω_D are marked on both plots and correspond to closed-loop magnitudes of -3 dB, 0 dB, 5 dB and -3 dB respectively; note also that the frequencies at these points satisfy $\omega_A < \omega_B < \omega_C < \omega_D$.

17.1.2 Close-loop phase: N -contours

If we set $\angle G_{CL}(j\omega) = \text{constant } \theta$ and $\tan(\theta) = N$, then lines of constant closed-loop phase also turn out to be contours with different centres and radii in the real/imaginary axes. These contours can also be superimposed on the Nichols plot and are called **N -contours** (Figure 17.7).

We have marked on the plot some new points ω_E , ω_F , ω_G and ω_H , where the closed-loop phase is given as -4° , -28° , -128° and -183° respectively. This can be verified from the closed-loop phase response shown in Figure 17.8. We note that the closed loop phase values are not shown explicitly; we must use the mouse to click on the appropriate points to find these values.

When we combine both the M - and N -contours together on the Nichols plot, we form the Nichols chart (Figure 17.9).

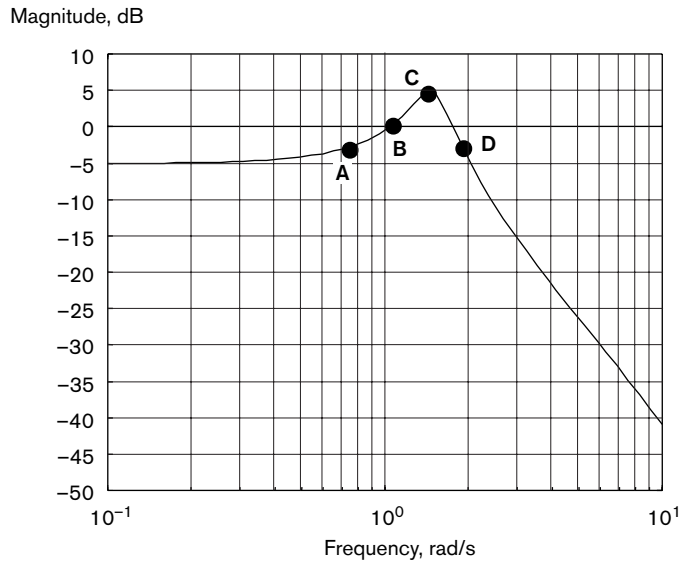


Figure 17.6 Closed-loop frequency response plot.

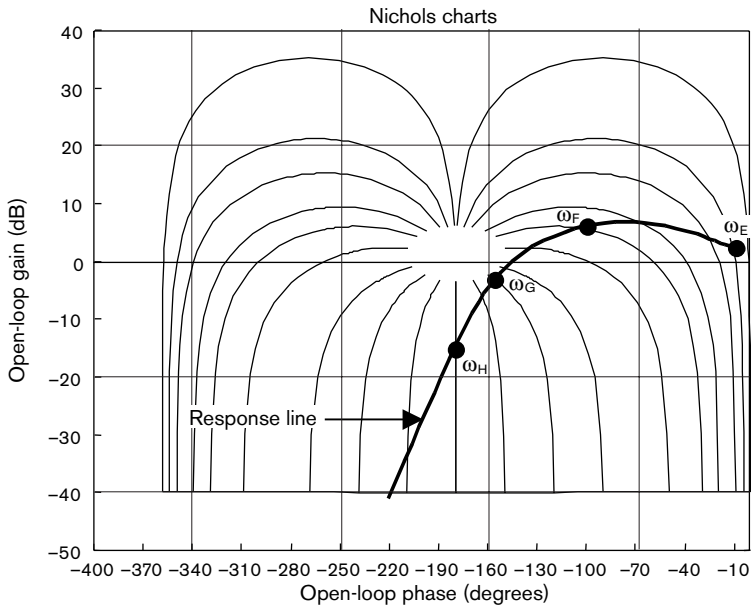


Figure 17.7 N -contours on the Nichols plot.

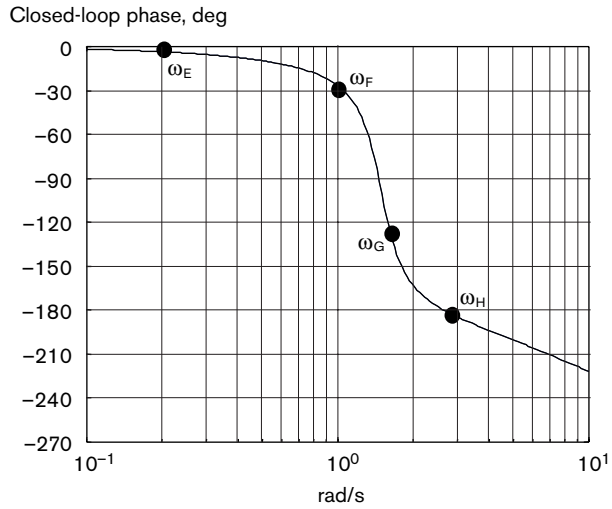


Figure 17.8 Closed-loop phase response.

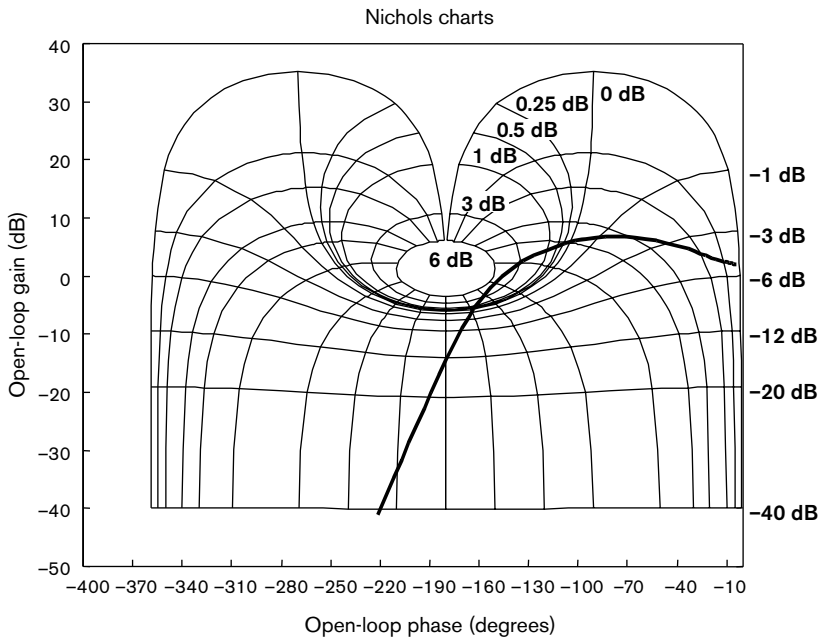


Figure 17.9 Nichols chart.

M 17.1.3 Using MATLAB to produce a Nichols chart

To produce a Nichols chart for the MATLAB transfer function, *g*, with the closed-loop dB magnitudes superimposed, use the commands


```

figure(1)           % introduce a new figure
ngrid              % set up the Nichols chart
nichols(g,{0.1,10}) % plot a Nichols chart for the frequency range
                  % given by values enclosed in { }; in this case
                  % 0.1 < ω < 10.

```

Problem For the transfer function $G(s) = 10/[s(s + 5)]$:

- choose an appropriate frequency range for the gain and phase calculation
- produce a Nichols chart for this range
- find the maximum peak magnitude for the closed-loop system

Solution (a) A suitable frequency range for a transfer function usually includes a range of frequencies centred on the corner frequencies of the system. This is often a decade above and below the corner frequency points, and then, since we plot frequency responses in decades, we find the nearest decade points which include this range. In this case, the corner frequency of the lag term is at 5 rad/s. We would like to include the range of 0.5 to 50 rad/s and therefore choose the values of 0.1 to 100 rad/s which will include these points.

- (b) We enter the following MATLAB commands to produce the Nichols chart in Figure 17.10.

```

s=tf('s'); g=10/(s*(s+5));
figure(1); ngrid;
nichols(g,{0.1,100});

```

- (c) The response line is tangential to the 0 dB line and does not cross this. Therefore the closed-loop response has no peak value above 0 dB.

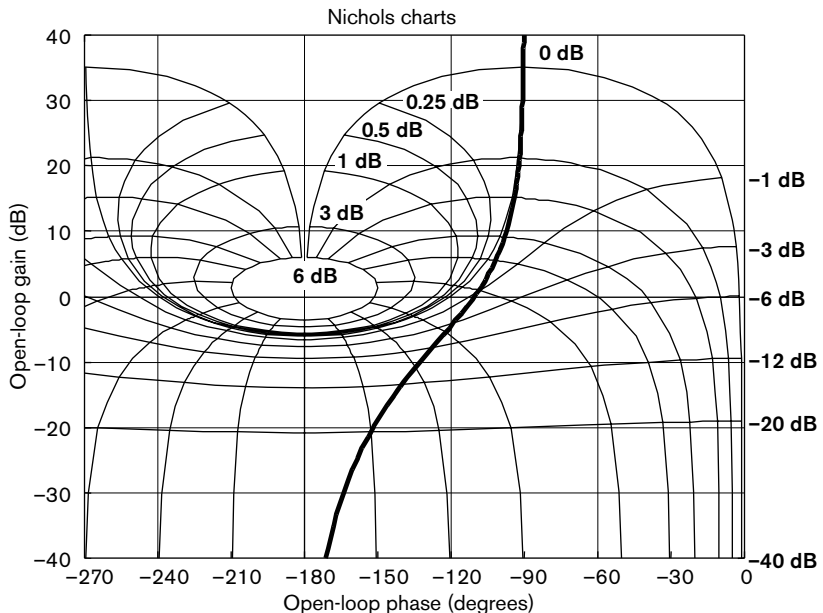


Figure 17.10 Nichols chart of $10/[s(s + 2)]$.

17.2 The Nichols chart

We summarise what we have learnt so far about the coordinate system used on the Nichols chart.

Key result: Open and closed loop data on the Nichols chart

The Nichols chart has essentially two coordinate systems:

1. Open-loop coordinates
 - The horizontal axis represents the phase of the open-loop system in degrees.
 - The vertical axis represents the gain of the open-loop system in dB.
2. Closed-loop coordinates
 - The intersection points of the open-loop response line with the M -contours give the values of the closed-loop gain in dB.
 - The intersection points of the open-loop response line with the N -contour gives the values of the closed-loop phase in degrees.

17.2.1 How to read the Nichols chart and find the closed-loop frequency response

As we have seen, the Nichols chart relates the open-loop frequency response of the system, $G_{OL}(j\omega)$ to the unity feedback closed-loop system:

$$G_{CL}(j\omega) = \frac{G_{OL}(j\omega)}{1 + G_{OL}(j\omega)}$$

Thus, we can read off the closed-loop frequency response, $G_{CL}(j\omega)$, from the Nichols plot of the open-loop system $G_{OL}(j\omega)$. This was one of the useful applications of the Nichols chart before the advent of digital computers and the development of computer-aided control design packages such as MATLAB Control Toolbox. The procedure is still useful for control design problems where a model of the process is not available in transfer function form but the plant frequency response is known as a table of data points, possibly obtained experimentally. The procedure for calculating the closed-loop frequency response is as follows:

1. Plot the open-loop frequency response on the Nichols chart using frequency information obtained from Bode plots or experimental data.
2. Find the intersection of the open-loop frequency response with the M -contours and N -contours.
3. Read the closed-loop gain and closed-loop phase from the M -contours and N -contours.
4. Calculate the corresponding frequencies for these gains by either extrapolating or interpolating the frequencies of the open-loop gain response.
5. Plot the frequency response of the closed-loop system.

We notice from Step 4 that the frequencies cannot be directly read from the Nichols chart. However, they can be found

- (a) by using the frequency information in an experimental table, which contains open-loop gain and phase information

- (b) by using MATLAB: the frequencies can be easily read by double-clicking the mouse on any point on the response line

Problem An engineer performs some experiments on an industrial process. It is known that the control system has the structure shown in Figure 17.11 and that the controller is a proportional controller.

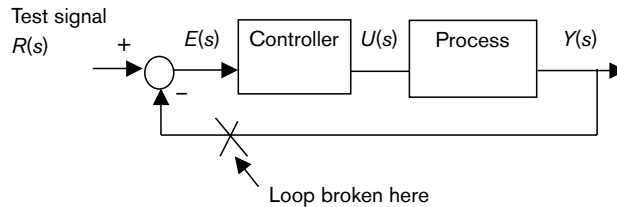


Figure 17.11 Model for control experiments.

The engineer performs a frequency response test of the system and finds the data shown in Table 17.1. From the open-loop experimental information, produce the Bode plot of the closed-loop magnitude frequency response and find the maximum peak on this closed-loop response plot.

Table 17.1 Experimental data from control system.

Frequency, rad/s	Open-loop gain, dB	Phase, degrees
0.1	18	-93
0.3	9	-97
0.5	6	-102
0.7	3	-108
1.0	0	-121
1.5	-2	-143
1.7	-3	-158
2.0	-5	-182
2.2	-7	-193
2.5	-9	-205
2.8	-13	-220

Solution The plot of the frequency response data on the Nichols chart is shown in Figure 17.12. We can use the M - and N -contours to form a table of approximate frequency and closed-loop magnitude, as given in Table 17.2. From this we can produce the Bode plot of the closed-loop magnitude response plot (Figure 17.13). We see that the peak magnitude appears at approximately 1.7 rad/s and is equal to 4 dB.

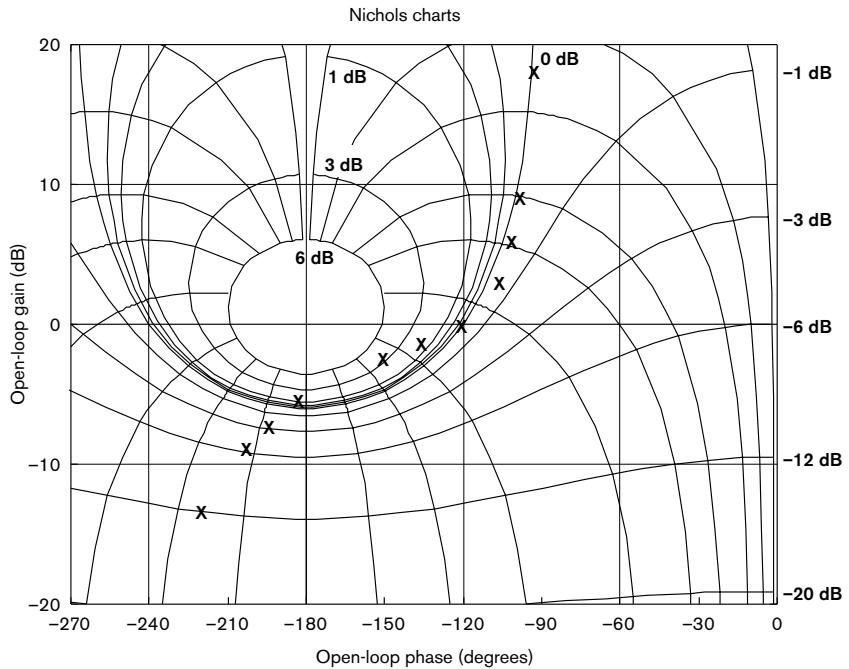


Figure 17.12 Nichols chart of experimental data.

Table 17.2 Closed-loop information from Nichols chart.

Frequency, rad/s	Closed-loop magnitude, dB
0.1	0
0.3	-0.1
0.5	-0.1
0.7	-0.3
1.0	0
1.5	2
1.7	4
2.0	2
2.2	-2
2.5	-7
2.8	-12

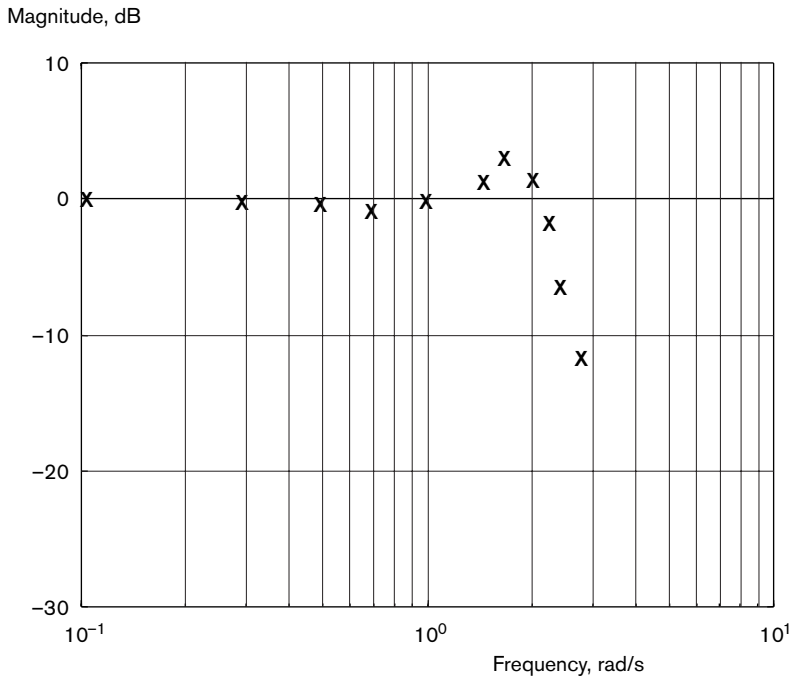


Figure 17.13 Closed-loop magnitude frequency response plot.

17.3 Design specifications on the Nichols chart

Control design specifications are often given in terms of performance indices such as bandwidth and peak overshoot, M_p , of the *closed-loop* response. Since the Nichols chart represents the relationship between closed-loop and open-loop parameters, it can help us to modify the transfer function of the open-loop system in order to achieve our specified closed-loop performance. In other words, on the Nichols chart we have a prescribed magnitude and frequency destination to which we would like the open-loop transfer function to be moved. In Bode plot design methods, we have to rely on methods which relate open and closed-loop information approximately in order to reach the design target.

Figure 17.14 shows a Nichols chart with the open loop frequency response of $G(j\omega)$. When we perform a control design we are looking for a compensated system $K(j\omega)G(j\omega)$ which will pass through specified design point(s). We may wish our frequency response line to pass through any of the four points D_1 to D_4 , where:

- D_1 : specification on phase margin
- D_2 : specification on closed-loop peak magnitude
- D_3 : specification on closed-loop bandwidth
- D_4 : specification on gain margin

The problem for the control engineer is to choose $K(j\omega)$.

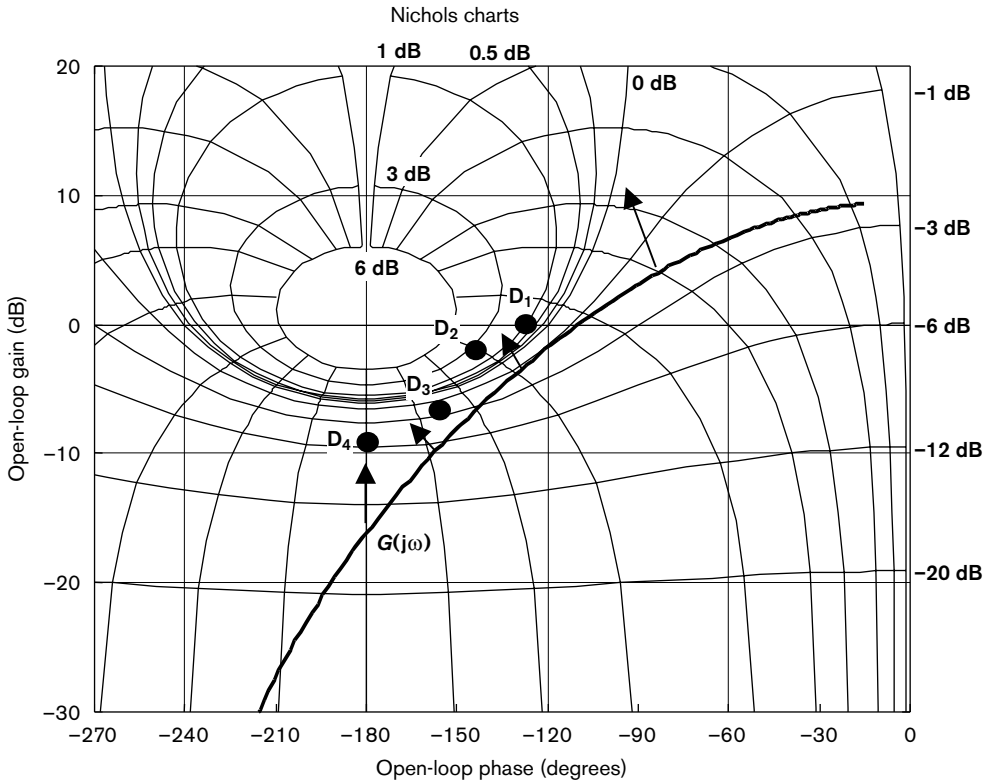


Figure 17.14 Control design specifications on the Nichols chart.

17.4 Reading gain and phase margins from the Nichols plot

To gain further insight and practice into the use of the Nichols chart, we will look at how we can alter the control gain, K , to achieve specifications on the open-loop gain margin, open-loop phase margin, closed-loop bandwidth or the closed-loop peak value M_p at frequency ω_p . We start by explaining how to read the gain and phase margins from the Nichols plot.

17.4.1 Gain margin

The gain margin is defined at the phase crossover frequency, ω_{pco} , as the magnitude in dB from the 0 dB line. We can find this value from the Nichols plot in Figure 17.15 by using the following procedure:

1. On the open-loop phase axis, find the -180° phase point.
2. Follow the -180° phase line up to the open-loop response line. This point is at frequency ω_{pco} .
3. The gain margin is the magnitude in dB between 0 dB and the point ω_{pco} .

$$GM_{\text{dB}} = 0_{\text{dB}} - |G_{\text{OL}}(j\omega_{\text{pco}})|_{\text{dB}}$$

It can be read off the vertical magnitude (dB) axis.

Remark

The closed-loop system is unstable if the gain margin is negative. If the open-loop gain is increased by the gain margin, then the overall gain at 180° phase shift will be unity (0 dB). The closed-loop system will then be bordering on stability for this gain, since there will be some closed-loop poles on the $j\omega$ axis.

We note that *both* the gain and phase margins must be positive for a closed-loop system to be stable, whereas, if one of the GM or the PM is negative, the system will be unstable.

Problem

For the Nichols chart in Figure 17.15, determine the gain margin and state whether the closed-loop system is unstable. Note that having a positive gain margin does not guarantee stability.

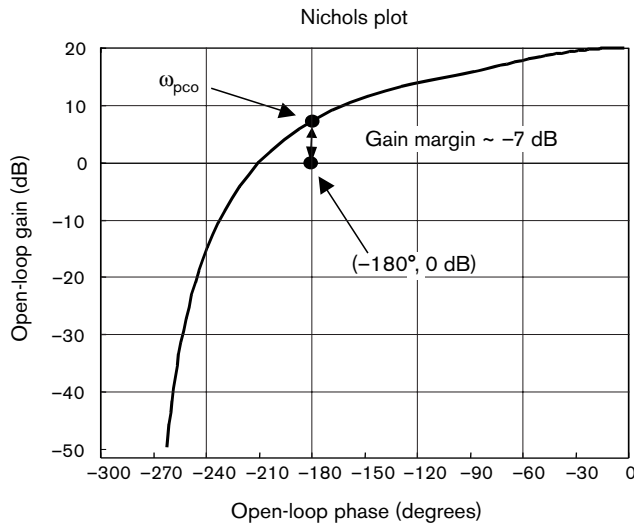


Figure 17.15 The gain margin for an unstable system.

Solution

The gain margin is marked on the plot and is calculated as -7 dB. Since this is negative, the closed-loop system will be unstable. We note that if the gain margin is measured *above* the $(-180^\circ, 0$ dB) point on the Nichols chart, the GM will be negative, giving an unstable system.

17.4.2 The phase margin

The phase margin (PM) is defined at the gain crossover frequency as the phase shift of the open loop transfer function plus 180° . Using the Nichols chart, the phase margin can be calculated by using the following procedure:

1. On the open-loop magnitude axis, find the 0 dB magnitude line.
2. Follow the 0 dB line across to the open-loop response line. The intersection point found is at frequency ω_{gco} .
3. The phase margin is the difference in phase between -180° and the phase at this point:

$$PM = \angle G_{OL}(j\omega_{gco}) - (-180^\circ).$$

It can be read off the horizontal phase axis.

Thus the phase margin is the amount of extra phase lag which the system can tolerate before it becomes unstable.

Problem For the Nichols chart shown in Figure 17.16, determine the phase margin of the system and state whether the closed-loop system is unstable.

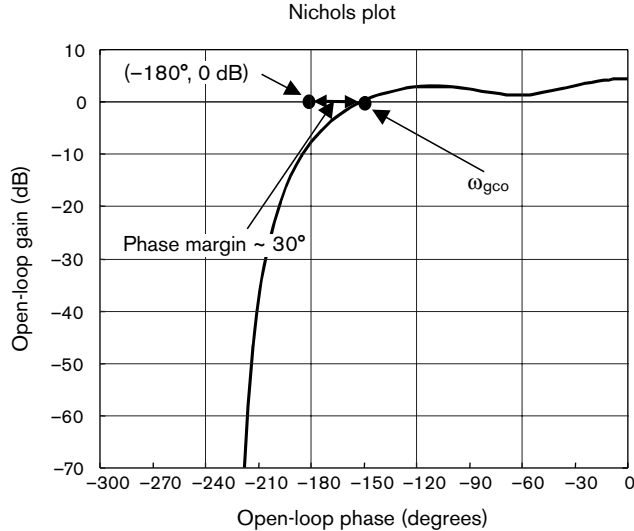


Figure 17.16 The gain margin and phase margin for a stable system.

Solution The phase margin can be read from the Nichols chart as 30° . This is positive and therefore the closed-loop system is not unstable *due to the PM*; we would have to check the gain margin before we could pronounce closed loop stability for the system. If the phase margin lies to the left of the $(-180^\circ, 0 \text{ dB})$ point, the phase margin will be negative and the closed-loop system will be unstable.

17.5 Altering the loop gain to achieve design specifications

We now study how to adjust the system gain in order to meet specifications on:

1. open-loop gain margin
2. open-loop phase margin
3. closed-loop bandwidth
4. closed-loop peak magnitude, M_p

Although we only use open-loop information to calculate the gain and phase margin design specifications, we can also use the Nichols chart to analyse the change on the closed-loop performance, which the alteration in loop gain will cause.

We use the system in Figure 17.17 to illustrate the design process.

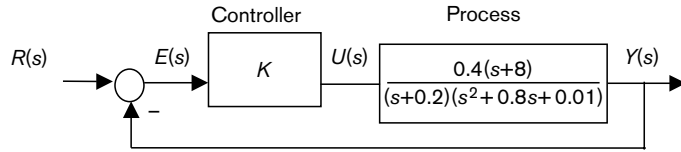


Figure 17.17 Block diagram for closed-loop design specification problems.

Example: Achieving a gain margin specification

We would like to select the proportional gain, K , to achieve a gain margin specification of 10 dB.

We plot the open-loop Nichols plot and examine the existing GM. This turns out to be approximately 3 dB, as shown in Figure 17.18. We would have to lower the plot by 7 dB to meet the specification. Therefore the value of controller gain K becomes

$$K = 10^{-7/20} = 0.4467$$

The Nichols chart of the compensated system is shown in Figure 17.19. The specification on GM has been met but the consequences of the reduction in gain are a reduction in closed-loop bandwidth. The response crosses the -3 dB line at a frequency of 0.389 rad/s compared with the original closed-loop bandwidth of 0.56 rad/s. These frequencies can be found from MATLAB by clicking on the response line. This bandwidth reduction will result in, for example, a slower step response from the system. However, the reduction in gain has also reduced the value of M_p to 6 dB, which will give a lower peak in the step response, though still a relatively large one.

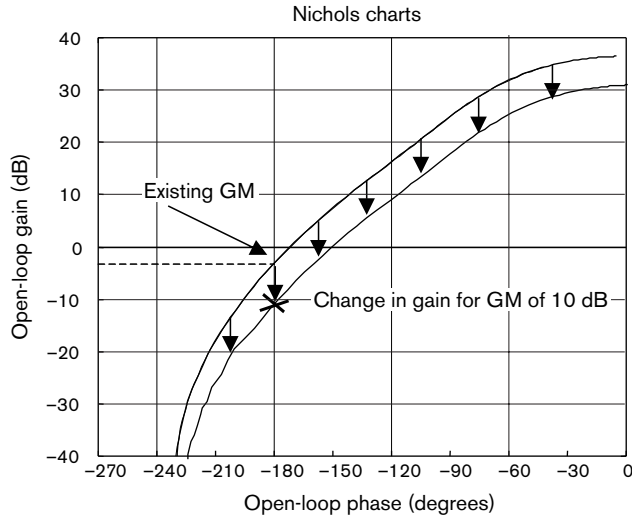


Figure 17.18 Change in gain K to satisfy GM specification.

Specification	Change	Comment
Gain margin	3 dB to 10 dB	Improved stability margins
Phase margin	$\sim 10^\circ$ to $\sim 45^\circ$	Improved stability margins
ω_{bw}	Reduction from 0.56 to 0.389 rad/s	Slower step response
M_p	Reduction from high value to 6 dB	Peak in step response reduced but still large

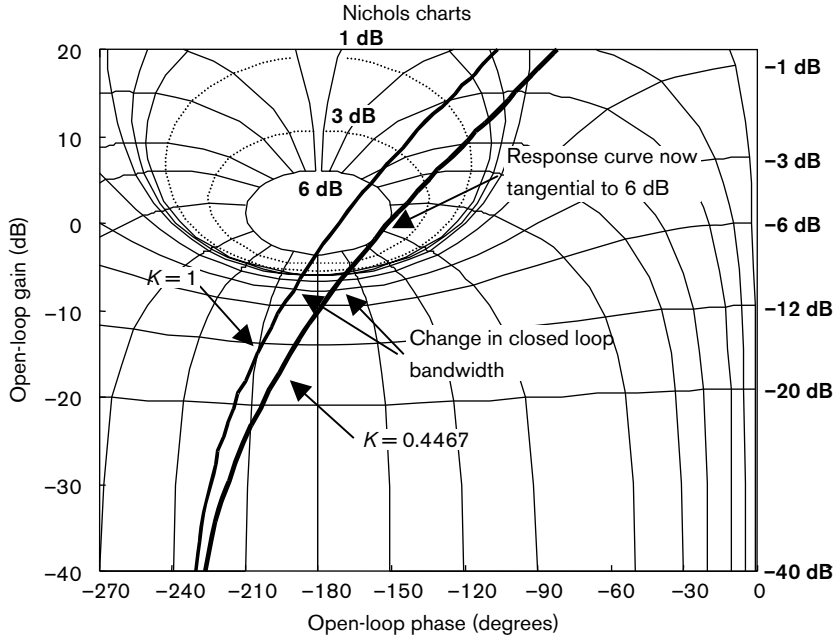


Figure 17.19 Nichols plot showing response meeting GM specification.

Example: Achieving a phase margin specification

In this example, we would like to alter the gain K to meet a specification of 45° phase margin. Once again we plot the open-loop information and find that, for $K = 1$, the PM is about 10° (Figure 17.20). We need to move the response line down to ensure that it crosses the 0 dB line at -135° phase; that is, we need to lower the response line by 11.4 dB. This is equivalent to setting $K = 10^{-11.4/20} = 0.2692$. The compensated plot is shown in Figure 17.21. As for the GM design, the loop gain has been lowered with a corresponding decrease in closed-loop bandwidth from

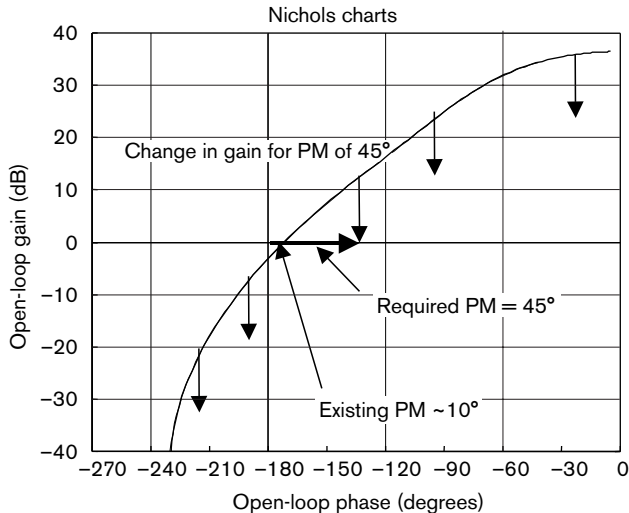


Figure 17.20 Gain alteration required to achieve PM specification.

0.56 rad/s to 0.29 rad/s, which will provide a slower time response. The peak magnitude has now reduced to 3 dB, which will give a much reduced peak overshoot to a step response.

Specification	Change	Comment
Gain margin	3 dB to ~14 dB	Improved stability margins
Phase margin	~10° to 45°	Improved stability margins
ω_{bw}	Reduction from 0.56 to 0.29 rad/s	Slower step response
M_p	Reduction from high value to 3 dB	Peak in step response reduced

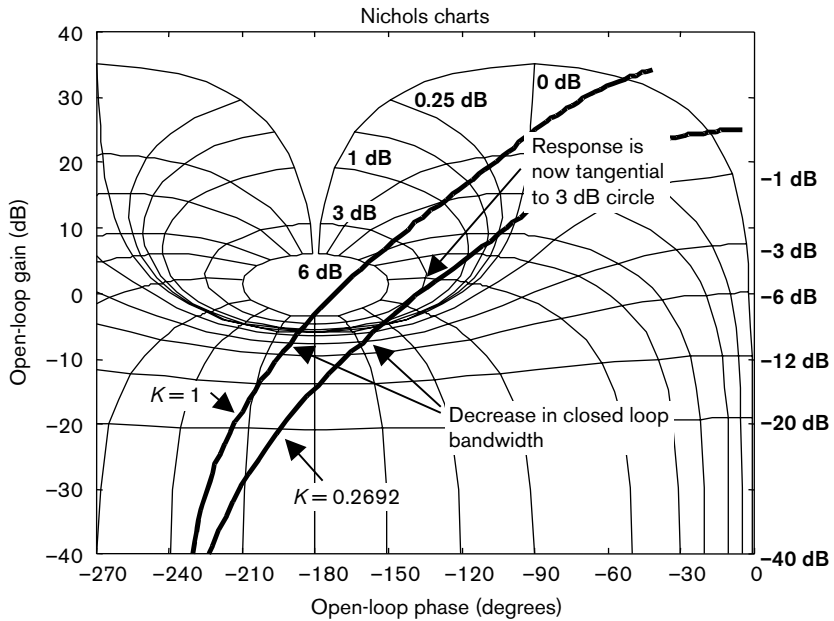


Figure 17.21 Loop gain altered to meet phase margin specification.

Example: Achieving an M_p specification – closed-loop maximum gain peak

The smallest M -contour tangential to the open-loop frequency plot gives the value of the maximum peak of the closed-loop frequency response. We would like to find the gain K such that the peak magnitude of the closed-loop system is less than some specified value, say α dB. This would determine the peak overshoot in a step response in the time domain.

Procedure

Plot the open-loop frequency response on the Nichols chart. Read the maximum peak of $|G_{CL}(j\omega)|$ by finding the smallest M -contour tangential to the open-loop response. If the value of this M -contour is greater than α dB, then move the open-loop response up or down, vertically, until it is tangential to the M -contour of value α dB. The vertical distance moved is the amount by which the gain has to be changed.

For the system used in the previous two examples, we would like to alter K to provide a closed-loop peak magnitude of 1 dB. The response with $K=1$ is shown in Figure 17.22. Finding a

value for K to meet a specification is not as straightforward as satisfying a GM or PM specification. This is because it is not easy by inspection to locate the point on the original curve which would become tangential to the required M -contour. We proceed by trial and error. Luckily, the use of MATLAB makes this a fairly simple procedure and we find that by trying a couple of gain values we can 'home in' on the most appropriate gain setting. We could also have sketched the response curve on a separate piece of paper to a sheet with a Nichols chart and moved one sheet over the other to place the new response line tangential to the required M -contour.

We find the M -contour with magnitude 1 dB. We need to move the $K = 1$ response line so it is tangential to this line. This requires a gain reduction of 14.2 dB, as shown in the figure. Hence the value of K is $K = 10^{-14.2/20} = 0.19$.

This loop gain provides a gain margin of approximately 15 dB and a phase margin of over 60° . The closed-loop bandwidth is reduced from 0.59 rad/s to 0.216 rad/s. These changes are summarised in the following table.

Specification	Change	Comment
Gain margin	3 dB to 15 dB	Improved stability margins
Phase margin	$\sim 10^\circ$ to $> 60^\circ$	Improved stability margins
ω_{bw}	Reduction from 0.56 to 0.216 rad/s	Slower step response
M_p	Reduction from high value to 1 dB	Peak in step response reduced significantly

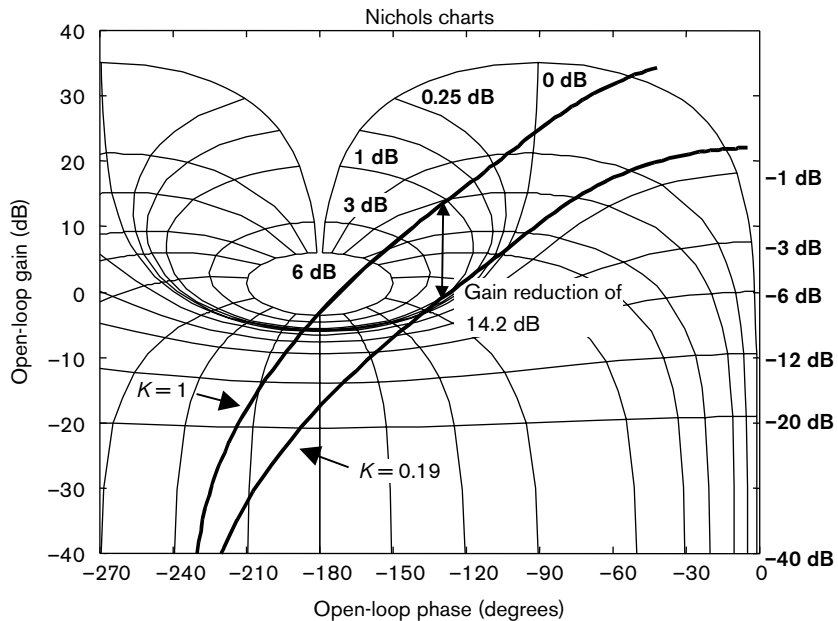


Figure 17.22 Setting gain K to meet a specification on peak magnitude.

Example: Achieving a bandwidth specification

We are often given a specification on closed-loop bandwidth, since this relates to the response time of the system and provides an indication of the frequencies at which noise and disturbances

might be rejected. A higher bandwidth will produce a more quickly responding system, but at the penalty of increased gain at high frequencies, which may amplify undesirable noise or disturbances. Hence the Nichols chart is particularly useful in attempting to analyse systems with a closed-loop bandwidth specification.

The closed-loop bandwidth is often taken to be at the point where the response curve crosses the -3 dB M -contour. We note that although we can find the closed-loop -3 dB point, there is no frequency axis on the chart. We must either refer to the data tables which produced the frequency response information or cross-reference from the closed-loop plot to the open-loop frequency response plots or a table of gain/phase/frequency data. MATLAB helps in this instance, since by using the cursor and clicking on the response line, MATLAB returns the frequency value at that point.

In this example we would like to meet a bandwidth specification of 0.35 rad/s. The original response line for $K=1$ is shown in Figure 17.23. Using MATLAB we can click on the response line to find the frequency value at different points. To help this process we may need to provide a higher intensity of frequency points. We can do this by creating a frequency vector

$$w = \text{logspace}(-2,1,300)$$

which provides 300 points in the frequency range 10^{-2} to 10^1 . Then we can use the `nichols` command, which creates the response line for these frequency points:

$$\text{nichols}(g,w)$$

When we find the frequency closest to 0.35 rad/s, we note the open-loop magnitude and the open-loop phase at the -3 dB M -contour vertically below that point. In this case, the gain reduction required is 8 dB, which gives a gain of $K = 10^{-8/20} = 0.398$. The modified frequency response is also shown as on Figure 17.23.

Specification	Change	Comment
Gain margin	3 dB to 13 dB	Improved stability margins
Phase margin	$\sim 10^\circ$ to $\sim 40^\circ$	Improved stability margins
ω_{bw}	Reduction from 0.56 to 0.35 rad/s	Slower step response
M_p	Reduction from high value to ~ 5.5 dB	Peak in step response reduced but still large

Note that by altering the open-loop gain characteristic, several other control design specifications may not be met or may be violated. Hence more complex modification to the open-loop system is required, which can be done by introducing controllers such as lead or lag compensators in the loop.

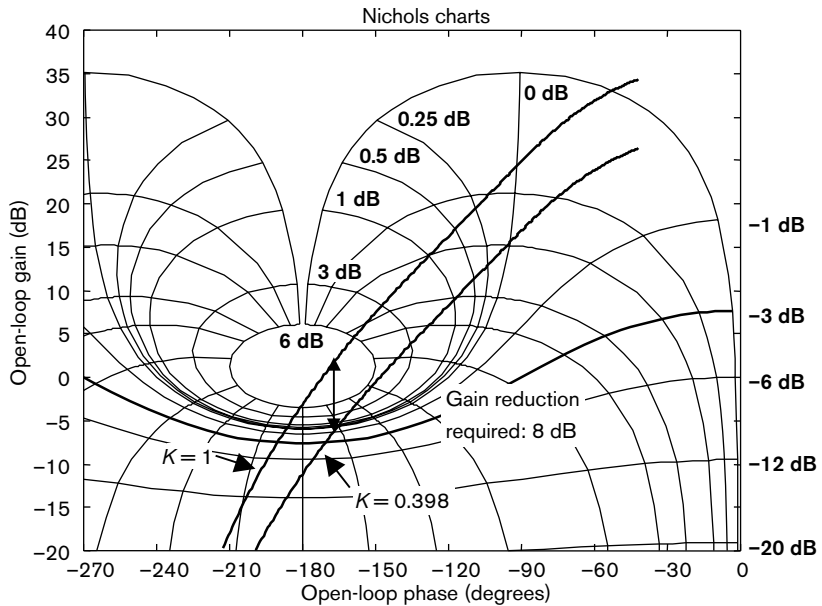


Figure 17.23 Altering gain K to meet bandwidth specification.

What we have learnt

- ✓ To relate open and closed-loop transfer functions.
- ✓ To use MATLAB to plot a Nichols chart.
- ✓ To read closed-loop information from the M - and N -contours on a Nichols chart.
- ✓ To adjust the controller gain K to meet the following design specifications:
 - open-loop gain margin and phase margin
 - closed-loop peak magnitude, M_p
 - closed-loop bandwidth

Multiple choice

M17.1 The vertical axis on the Nichols chart represents:

- (a) open-loop gain
- (b) open-loop phase
- (c) closed-loop gain
- (d) closed-loop phase

M17.2 M -contours show:

- (a) open-loop gain
- (b) open-loop phase
- (c) closed-loop gain
- (d) closed-loop phase

M17.3 The phase margin of a system can be found from the Nichols chart by examining:

- (a) the intersection of M -contours with the phase axis
- (b) the intersection of M -contours with the gain axis
- (c) the intersection of the open-loop plot with the 0 dB magnitude axis
- (d) the intersection of the open-loop plot with the -180° phase axis

M17.4 N -contours show:

- (a) open-loop gain
- (b) open-loop phase
- (c) closed-loop gain
- (d) closed-loop phase

M17.5 To find the peak value of a closed-loop transfer function on the Nichols chart, we look for:

- (a) the M -contour which intersects the open-loop Nichols plot
- (b) the N -contour which intersects the open-loop Nichols plot
- (c) the M -contour which is tangent to the open-loop Nichols plot
- (d) the M -contour which is tangent to the gain axis

M17.6 The peak value of a closed-loop transfer function is related to

- (a) system overshoot
- (b) system bandwidth
- (c) system steady state error
- (d) system poles

M17.7 The closed-loop bandwidth on the Nichols chart:

- (a) is the frequency at which the -3 dB M -contour intersects the open-loop Nichols plot
- (b) is the frequency at which the N -contour intersects the open-loop Nichols plot
- (c) is the frequency at which the M -contour intersects -3 dB on the gain axis
- (d) is the frequency at which the 3 dB M -contour intersects the open-loop Nichols plot

M17.8 The Nichols chart can relate the open-loop frequency response easily to:

- (a) the closed-loop frequency response
- (b) the open-loop peak overshoot
- (c) the steady state error to an input step
- (d) none of the above

M17.9 Multiplying an open loop transfer function by a gain of 0.1 will shift all the points on the response line on the Nichols chart:

- (a) vertically down
- (b) vertically up
- (c) horizontally to the left
- (d) horizontally to the right

M17.10 An open-loop system has a PM of 90° . The Nichols plot crosses the 0 dB magnitude axis at:

- (a) 90°
- (b) -90°
- (c) 180°
- (d) -180°

Questions: practical skills

Q17.1 A system has an open-loop transfer function of $G(s) = (s - 1)/s^2$. Find the GM and PM using a Nichols chart.

Q17.2 A d.c. servo system has the open-loop transfer function $G(s) = K/[s(s + 8)]$. Find the value of K for the closed-loop system to have a peak resonance value of 3 dB.

Q17.3 Plot the Nichols chart for the following phase lead and phase lag controllers and find their maximum phase shifts.

$$G(s) = \frac{10s+1}{0.1s+1} \quad H(s) = \frac{0.1s+1}{10s+1}$$

Q17.4 A unity feedback system has the following transfer functions:

$$\text{plant: } G(s) = \frac{s+5}{s^3+15s^2+50s}$$

$$\text{PI controller: } K(s) = \frac{10s+1}{s}$$

Find the resonant peak of the closed-loop system using a Nichols chart.

Q17.5 For the system with open-loop transfer function given by

$$G(s) = \frac{13}{500s^2+60s+1}$$

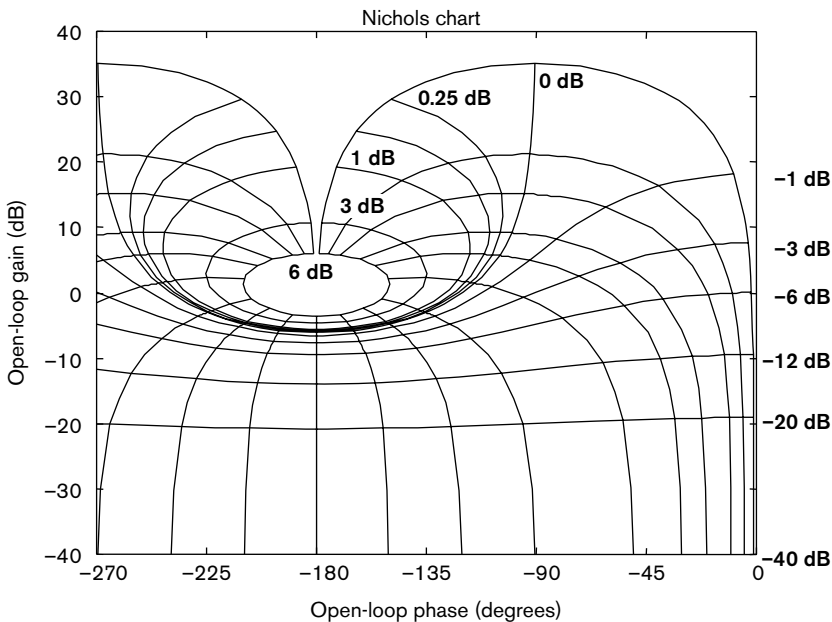
find the closed-loop bandwidth and the resonant peak from a Nichols chart.

Problems

P17.1 An engineer has found the frequency response of the open loop of a system and tabulated it as shown below.

Frequency (rad/s)	0.01	0.02	0.03	0.06	0.11	0.19	0.34	0.61	1.1	2
Gain (dB)	21	9.5	16.2	11.2	4	-3	-13	-23	-33	-43
Phase (°)	-32	-52	-76	-101	-125	-146	-160	-168	-173	-176

(a) Sketch the Nichols plot on the Nichols chart below.



(b) Find the resonant peak and bandwidth of the closed-loop system.

528 Analysis and simple design using the Nichols chart

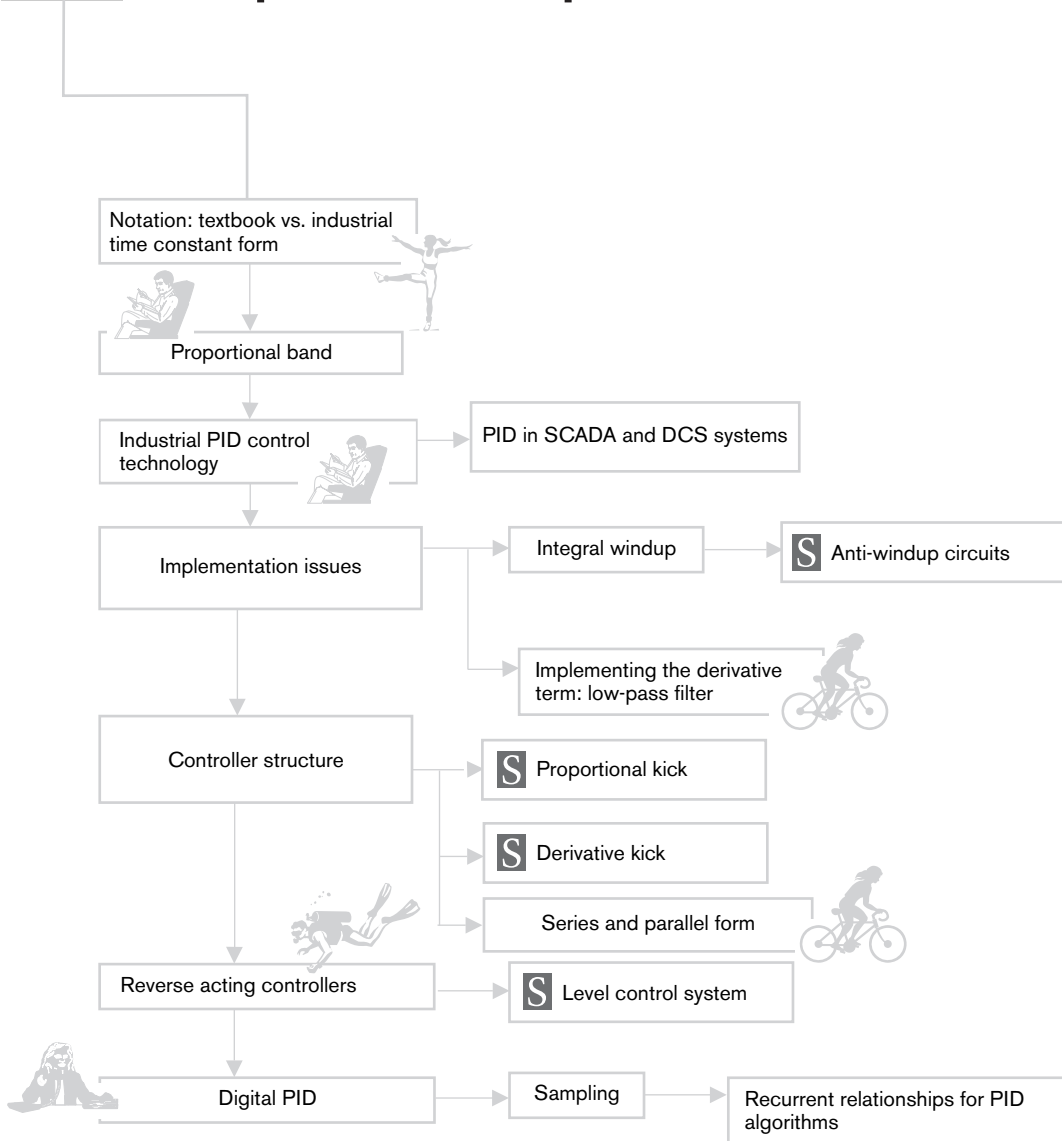
P17.2 An engineer has performed a closed-loop frequency test on a two-tank process control system and found the data shown in the table below. Use a Nichols chart to find the open-loop frequency response and hence the values of M_p , GM and PM of the system.

ω (rad/s)	0.01	0.02	0.03	0.06	0.11	0.19	0.34	0.61	1.11	2
Gain (dB)	-0.6	-0.56	-0.40	0.15	1.81	0.66	-11.5	-22.8	-33.3	-43.6
Phase(°)	-2.4	-4.4	-8.2	-16	-37	-109	-155	-168	-174	-177

P17.3 The sensitivity transfer function, $S(s) = 1/(1 + G(s))$ represents the frequency response from the input disturbance to the output signal.

- Find a relation between the open-loop transfer function $S(s)$ and $G(s)$ and $G_{CL}(s)$, where $G_{CL}(s)$ is the closed-loop transfer function.
- Using the information in the tables of the previous question, and the results from your plot in the previous question, form a table showing the values of $S(j\omega)$ for the frequencies given in the table.
- Use your Nichols chart to plot the sensitivity frequency response on a Bode plot and hence find out where on the frequency range we may have disturbance amplification.

18 The practical aspects of PID control



In the application of industrial control, we usually find three-term or PID controllers being used in process loops. The complexity of these control loops will invariably depend on the particular application. For example, in a paper mill, a liquid level control system may use PI control in a single loop. But in the paper making process itself we will come across the *wet-end* unit, where paper slurry is poured onto a moving mesh conveyor system to produce paper sheet. The control of this unit is more complicated and we find that a solution using several control loops is needed. We often find that each of these loops uses industrial PID controllers, but on occasion, even more complicated controllers may be needed. Thus, in general, industrial control involves a large number of simple control loops and a much smaller number of more complicated control systems. These special control schemes will be on the more complicated and demanding units in the process or manufacturing production line. For the simple control loops, we usually find that PID controllers have been used.

If we look back at the development of PID control, we find that mass-produced electronic PID controllers have been available since about the 1940s. In the first forty years of sales these controllers were analogue, but from the 1980s onward we find microprocessor digital technology beginning to take over. Over the same period, we find the widespread introduction and development of Distributed Computer Systems (DCS systems) and Supervisory Control and Data Acquisition Systems (SCADA systems) to operate large manufacturing and process lines. The PID controller is a standard feature of the control tools of the computer-based industrial systems. Thus in industry, we meet PID controllers in several different formats and implemented by different technologies. In this chapter, we describe the PID controller as it is found in industry today.

Our first task is to consider the basic notation of industrial PID, and we follow this by looking at current PID control technology. After this we examine three different PID controller issues:

- We investigate the practicalities of implementing integral and derivative terms in PID controllers.
- We look at the varieties of industrial PID controller structures that industrial control engineers have developed.
- We make the step to digital PID control and examine how we use sampled process signals in PID control algorithms.

Learning objectives

- To understand common industrial PID controller notation.
- To be able to describe process controller units as might be found in industrial systems.
- To understand the practical methods for implementing the integral and derivative terms in a PID controller.
- To be able to interpret the notation and operation of common forms of industrial PID controllers.
- To understand simple digital PID control methods.

18.1 Understanding common notation for industrial PID controllers

We have previously used a straightforward notation to define a three-term or PID controller. Three simple gains, K_p , K_i and K_d were used in the parallel branches of the PID controller. We used this parallel form of PID in Chapters 11 and 12. We begin our discussion from the parallel representations, one for the time-domain and one for the Laplace s -domain.

Time-domain PID controller formula **Transfer function PID controller formula**

$$u(t) = K_p e(t) + K_i \int_0^t e(\tau) d\tau + K_d \frac{de}{dt} \qquad U(s) = \left[K_p + \frac{K_i}{s} + K_d s \right] E(s)$$

where K_p is the proportional gain, K_i is the integral gain, K_d is the derivative gain, and the controller operates on the reference error signal, $e(t) = r(t) - y(t)$. We call these *textbook* PID formulas, and they correspond to an ideal structure that has three parallel paths corresponding to the *Proportional*, *Integral* and *Derivative* terms (Figure 18.1).

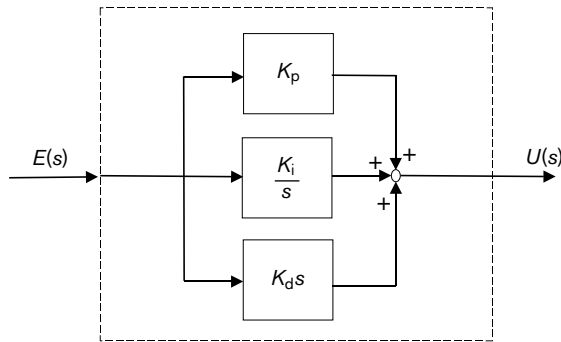


Figure 18.1 Parallel form of PID controller.

If we now look up PID control in a Works Manual we usually find that industrial notation differs from these textbook formulas by defining a **time constant form** for PID. To arrive at this new form in the case of the time domain formula, we use the following steps, starting with the textbook form:

$$u(t) = K_p e(t) + K_i \int_0^t e(\tau) d\tau + K_d \frac{de}{dt}$$

We first factor out the proportional gain, K_p :

$$u(t) = K_p \left[e(t) + \frac{K_i}{K_p} \int_0^t e(\tau) d\tau + \frac{K_d}{K_p} \frac{de}{dt} \right]$$

We then define new time constants τ_i and τ_d , and this gives a new formula:

$$u(t) = K_p \left[e(t) + \frac{1}{\tau_i} \int_0^t e(\tau) d\tau + \tau_d \frac{de}{dt} \right]$$

where K_p is the proportional gain, $\tau_i = K_p/K_i$ is the integral time constant and $\tau_d = K_d/K_p$ is the derivative time constant.

We can also give a similar analysis for the transfer function expressions:

$$U(s) = \left[K_p + \frac{K_i}{s} + K_d s \right] E(s)$$

We first factor out the proportional gain, K_p :

$$U(s) = K_p \left[1 + \frac{K_i}{K_p s} + \frac{K_d}{K_p} s \right] E(s)$$

We then define new time constants τ_i and τ_d to find a new formula:

$$U(s) = K_p \left[1 + \frac{1}{\tau_i s} + \tau_d s \right] E(s)$$

This has the same definitions for K_p , τ_i and τ_d as the new time domain formula above. The textbook and industrial time constant forms are summarised in Table 18.1.

Table 18.1 Textbook and industrial time constant form of PID controller equation.

Domain	Time domain	Laplace s-domain
Textbook	$u(t) = K_p e(t) + K_i \int_0^t e(\tau) d\tau + K_d \frac{de}{dt}$	$U(s) = \left[K_p + \frac{K_i}{s} + K_d s \right] E(s)$
Industrial	$u(t) = K_p \left[e(t) + \frac{1}{\tau_i} \int_0^t e(\tau) d\tau + \tau_d \frac{de}{dt} \right]$	$U(s) = K_p \left[1 + \frac{1}{\tau_i s} + \tau_d s \right] E(s)$

Skill section

Parallel form to time constant form

We need to be able to move between the gain and time constant forms of PID controllers. The works technical sheets appear to show the four different PID controller transfer functions,

1. $G_{c1}(s) = \left[7.2 + \frac{5.26}{s} + 1.07s \right]$
2. $G_{c2}(s) = 4.85 \left[1 + \frac{1}{10.36s} + 2.57s \right]$
3. $G_{c3}(s) = \left[4.85 + \frac{0.468}{s} + 12.46s \right]$
4. $G_{c4}(s) = 7.2 \left[1.0 + \frac{1}{1.369s} + 0.149s \right]$

We practise our skills of putting PID controllers into common standard forms by showing that (1) $G_{c1}(s) = G_{c4}(s)$ and (2) $G_{c2}(s) = G_{c3}(s)$.

1. We start with $G_{c1}(s)$ and we first factor out the proportional gain:

$$G_{c1}(s) = \left(7.2 + \frac{5.26}{s} + 1.07s \right) = 7.2 \left(1 + \frac{5.26}{7.2} \frac{1}{s} + \frac{1.07}{7.2} s \right)$$

We then define new time constants $\tau_i = 7.2/5.26 = 1.369$ and $\tau_d = 1.07/7.2 = 0.149$ to find a new formula:

$$G_{c1}(s) = 7.2 \left(1 + \frac{1}{1.369s} + 0.149s \right)$$

Clearly this shows that $G_{c1}(s) = G_{c4}(s)$.

2. We see that $G_{c3}(s)$ is in the form with three independent gains, so starting from $G_{c2}(s)$, we must multiply through by the proportional gain to find the textbook form:

$$\begin{aligned} G_{c2}(s) &= 4.85 \left[1 + \frac{1}{10.36s} + 2.57s \right] = \left[4.85 + \frac{4.85}{10.36s} + 4.85 \times 2.57s \right] \\ &= \left(4.85 + \frac{0.468}{s} + 12.46s \right) \end{aligned}$$

This is the same as $G_{c3}(s)$.

In other industrial manuals, we are quite likely to find other traditional terms that some manufacturers continue to use. If we know what these terms are then at least we shall have a chance of understanding what the Works' resident expert on PID tuning is talking about when it comes to terms like proportional band and reset time.

18.1.1 Proportional band, denoted PB

Proportional band is a term that is sometimes used in the specification of PID in the process industries. We know that when proportional control is used on its own, it has the form $u(t) = K_p e(t)$, or if we use *changes* in the control signal and the error signal then $\Delta u = K_p \Delta e$, where K_p is the proportional gain. We also know that the controller input and output signals often have a limited range which may be set according to the process actuation or measurement devices. The signal $e(t)$ and the control signal $u(t)$ may then have different units and different ranges of variation. We first introduce definitions for the signal ranges as

$$u_R = u_{\max} - u_{\min} \quad \text{and} \quad e_R = e_{\max} - e_{\min}$$

Proportional band is a PID controller term that is defined in terms of these signal limits. It is the percentage of the full range of $e(t)$ that gives rise to a 100% variation in $u(t)$. This is represented in Figure 18.2, where we see the difference between a 20% and a 60% proportional band. It is easy to see from the plots why the proportional bands are sometimes referred to as *narrow* or *wide*.

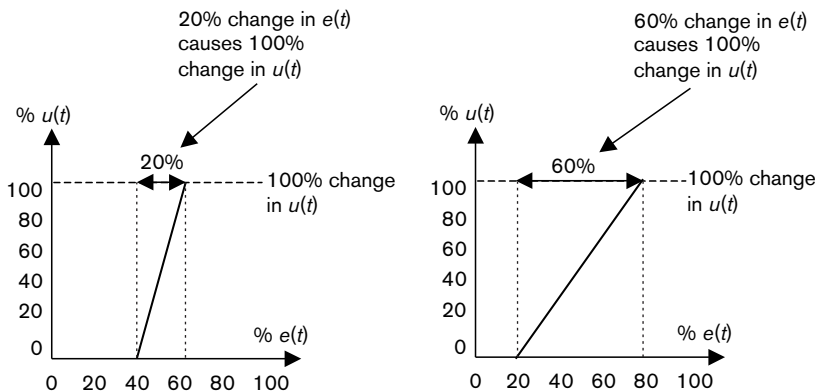


Figure 18.2 20% and 60% proportional bands.

To calculate the proportional band, we first use the signal ranges to introduce definitions for the normalised control and error signals as follows:

$$\Delta u_N = \frac{\Delta u}{u_R} \quad \text{and} \quad \Delta e_N = \frac{\Delta e}{e_R}$$

where Δu_N is the normalised control signal and Δe_N is the normalised error signal. It is important to realise that the above normalised signals are dimensionless, and that Δu_N and Δe_N represent fractional changes on the full signal change possible. The basic controller relationship $\Delta u = K_p \Delta e$ can now be substituted in the expression for the normalised signals:

$$\Delta u_N = \frac{\Delta u}{u_R} = \frac{K_p \Delta e}{u_R} = \frac{K_p \Delta e_N e_R}{u_R} = \frac{K_p e_R}{u_R} \Delta e_N$$

We now determine the change in controller input, Δe_N which is necessary to give a unit change in the controller output, Δu_N . In other words, we simply set $\Delta u_N = 1$ and solve for the Δe_N fraction. If we multiply this Δe_N value by 100%, we get the proportional band of the controller. Using the formula above, we find:

$$\Delta u_N = 1 = \frac{K_p e_R}{u_R} \Delta e_N \quad \text{so that} \quad \Delta e_N = \frac{u_R}{K_p e_R}$$

The proportional band is thus:

Key result: Proportional band

$$PB = \left(\frac{1}{K_p} \right) \left(\frac{u_R}{e_R} \right) \times 100\% = \left(\frac{\Delta e}{\Delta u} \right) \left(\frac{u_R}{e_R} \right) \times 100\%$$

We find that the controller signals and error signals are often scaled to lie between 0 and 100%, in which case the ranges u_R and e_R are equal. If these ranges are equal, then the ratio is unity, and we get the simplified relationship quoted in many control textbooks,

$$PB = \frac{100\%}{K_p}$$

We can also use this formula in reverse, for if the control and error ranges are equal, and the proportional band, PB , is known, then the proportional gain, K_p , can be found as

$$K_p = \frac{100\%}{PB}$$

For the case where the control and error ranges are equal we can look at a range of proportional gains and find the correlation that *small* proportional gain values correspond to *large* proportional band percentage values and *vice versa*. We have used the formula $PB = 100\%/K_p$ to construct a table to show this (Table 18.2).

Table 18.2 Proportional gains and proportional band values.

Proportional gain	0.25	0.5	1	10	50
Proportional band	400%	200%	100%	10%	2%

Problem In a hydraulic process it was found that the controller input variable, $e(t)$, was a mechanical displacement of range $e_R = 1$ cm, and the effective controller output, $u(t)$, was a pressure of range $u_R = 2$ bar. At a given setting of the controller, 0.1 cm of change in $e(t)$ caused 0.5 bar change in control output, $u(t)$. What is the proportional band setting of this controller?

Solution The controller input and output ranges are not equal, so we must use the full formula:

$$PB = \left(\frac{1}{K_p} \right) \left(\frac{u_R}{e_R} \right) \times 100\% \quad \text{or} \quad PB = \left(\frac{\Delta e}{\Delta u} \right) \left(\frac{u_R}{e_R} \right) \times 100\%$$

Since we do not have data on K_p , we must use the second formula as follows:

$$PB = \left(\frac{\Delta e}{\Delta u} \right) \left(\frac{u_R}{e_R} \right) \times 100\% = \frac{0.1 \text{ cm}}{0.5 \text{ bar}} \frac{2 \text{ bar}}{1 \text{ cm}} \times 100\% = 40\%$$

There are other traditional terms that you may find being used: reset time and pre-act time are two of these. *Reset time* is another name for the time constant of the integral term. Some manufacturers refer to the integral time constant τ_i in *time per repeat*. Other manufacturers use the inverse of this and scale the integral term using *reset rate*. The units of reset rate are, not surprisingly, repeats per time. *Pre-act time* is an industrial control name given to the derivative time constant, τ_d . All of these traditional terms have their origin in analogue technology and an industrial need for descriptive labels for the effects of the different terms in PID controllers. We expect these terms to have less importance as the PID tuning process becomes completely automatic.

18.2 Industrial PID control technology

In industrial implementations, PID control is packaged in an easy-to-use format. The days of having to construct a PID controller from analogue or other components is long past. We usually meet PID control in one of two forms: either as a process controller hardware unit or as a PID tuning interface window in an industrial SCADA system.

18.2.1 Process Controller Unit

Most process control companies produce a range of hardware process control units. These units offer a limited but effective range of PID control capabilities. Advanced features will usually include autotuning routines and the ability to operate several control loops. For the enthusiast we will find that manual tuning using the process controller interface is still also available. However, there are good reasons why we must fully understand the background to the tuning procedures and which form of PID controller is being used. For example, we may be called upon to solve PID tuning problems on troublesome loops or be required to manually tune up particular loops in the plant. We would advise a careful study of the company-provided User's Handbook to avoid controller tuning mistakes. A typical interface found on the modern process controller is shown in Figure 18.3.

In the **WJK SelfTune Pioneer** process controller we can find features typical of an industrial product. At the top of the device we see a trend display showing the set point, denoted SP, and the process variable, denoted PV. We would use the set point press-button marked 'SP' to set up the desired value of the set point level. The buttons marked 'Δ' and '∇' would be used to increase or decrease numerical inputs like values for

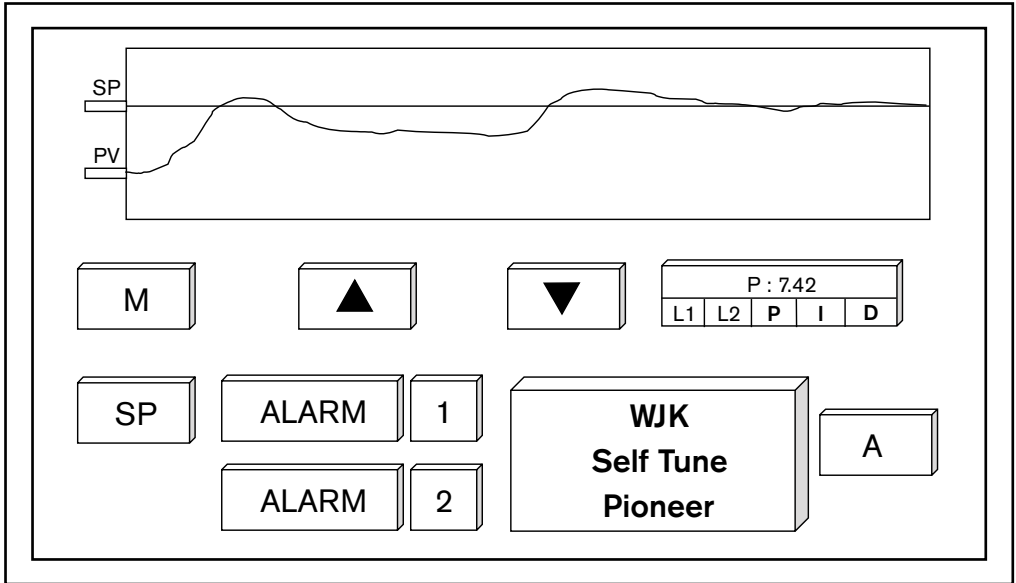


Figure 18.3 The WJK SelfTune Pioneer PID process control unit.

the set point or coefficients for the PID control law used. Manual tuning can be activated by the ‘M’ button, and an autotune or automatic tuning procedure initiated using the ‘A’ button. We can enter PID parameters via the ‘P’, ‘I’ and ‘D’ buttons for loops L1 and L2. Clearly, it is essential to know the form of the PID law to make sure that the correct tuning values are being entered. At present the display shows the value of the ‘P’ term, $P = 7.42$. Finally, there are some process alarm settings which can be made using the ‘ALARM1’ and ‘ALARM2’ entry ports.

The correspondence between the terms used in general control engineering and those often found in the process industries is shown in Table 18.3.

Table 18.3 Summary of technical terms.

General control engineering		Process control industries	
Term	Notation	Term	Notation
Output	$y(t), Y(s)$	Process variable	$PV, PV(s)$
Reference	$r(t), R(s)$	Set point	$SP, SP(s)$
Error	$e(t), E(s)$	Set point error	$E, E(s)$

18.2.2 SCADA system PID tuning interface window

Most complex industrial processes are under computer control. We call a complete installation (hardware and software) a Distributed Computer System (DCS) or a Supervisory Control and Data Acquisition (SCADA) system. In a heavy industry like the steel

industry we would possibly find the Automation Engineer referring to the overall control system as a DCS, while on an offshore oil platform in the North Sea or Gulf of Mexico we would probably hear a Platform Process Engineer call virtually the same system the platform's SCADA system. The technological differences between the two computer control systems are not so great, but as is often the case, different industries have evolved different terminology and jargon.

In the DCS or SCADA system, access to the setup and tuning of the PID control loops will be via a control engineer's interface with window-like functionality. We usually find that access to this window will be accompanied by password protection and security checks. This is to prevent unauthorised alterations to key process operational parameters. Since some of the processes under control may be capable of causing catastrophic damage, it is important to guarantee staff safety from the results of vandalism, sabotage or simple operational errors. We show a typical example of an engineer's interface in Figure 18.4. In the interface, we can see features very similar in notation and presentation to those on the front panel of the process controller unit (Figure 18.3); however, the SCADA system usually has greater functionality than a simple process unit, which might be controlling only one or two loops. The SCADA system may be able to page through different windows which display information on different control loops.

In the SCADA interface window we find common terms such as 'SP' and 'PV', which denote set point and process variable respectively. We see there is an extra variable 'DEV' showing the deviation of the process variable from the set point value. There are alarm limits, graphical setup parameters and manual and automatic tuning mode controls. If we look carefully, we can see the PARAMETERS section. This is where the PID parameters are stored and modified. Cursor editing alters the controller parameters. Looking at the figure, we can see $G = 1.00$; $TI = 30.00$; $TD = 0.00$; $TF = 0.00$; $TS = 10.0$. These are the PID controller parameters, where G represents proportional gain, TI represents a parameter

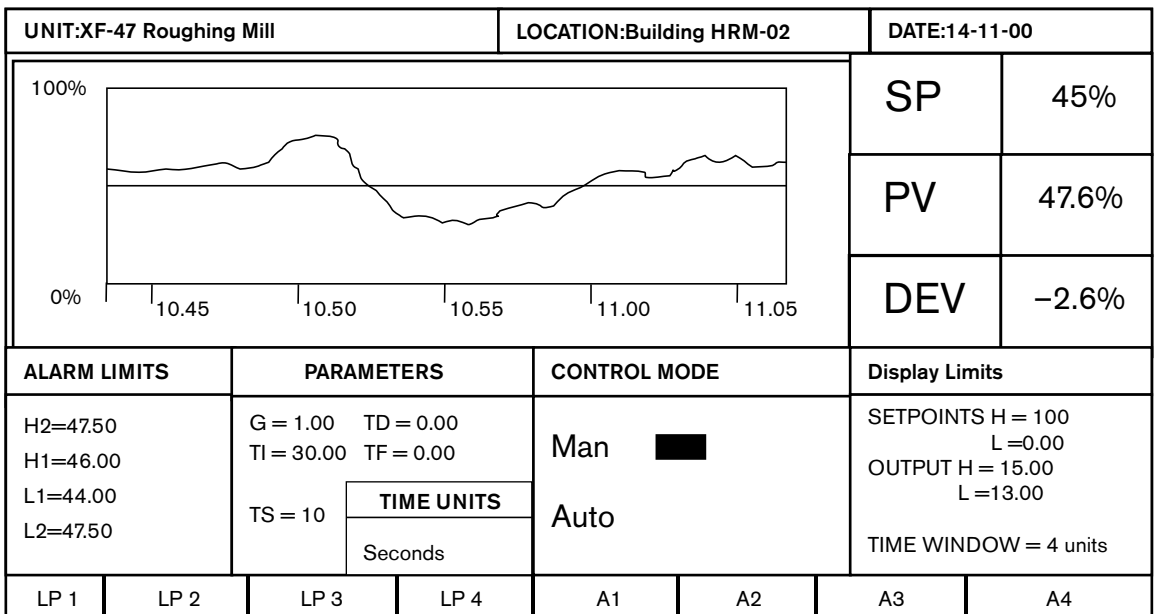


Figure 18.4 SCADA Interface.

related to the integral term, TD represents a parameter related to the derivative term, TF represents a parameter related to the derivative filter and TS is a parameter related to sample interval. To understand fully how the PID terms have been defined and implemented we need to refer to the User's Manual as provided by the DCS/SCADA system manufacturer. For example, in this particular software implementation, the parameter TS (signal sample period) indicates that the PID controller has been given discrete or sampled-data form.

18.3 The issues in implementing an industrial PID controller

The first important lesson we have learnt from these examples of the industrial technology for PID control is that we need to understand fully *how* the PID controller has been implemented before attempting a tuning exercise. Often, this information can only be obtained from the manufacturer's User's Manual or by discussion with the installer's personnel. However, industrial PID control faces several common problems in the implementation of the terms of the controller and it is the solutions to these problems that we shall look at next.

1. *Real systems are not linear, but at best are linear in a small signal range.*

This means that nonlinear systems effects are common in most processes. One nonlinear effect which is known to deteriorate PID control is the presence of saturation in controller and actuator devices (Figure 18.5). This leads to the phenomenon of *integral wind-up*. In this chapter, we will see how this has led to the development of anti-wind-up circuits to mitigate the wind-up effect.

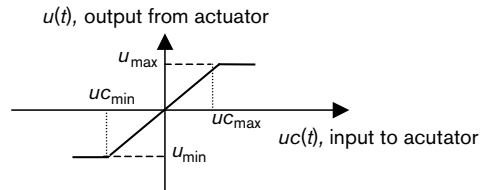


Figure 18.5 Actuator saturation.

2. *Real systems are not disturbance free.*

In previous chapters, we have learnt how PID can reduce the effects of low-frequency supply and load disturbances. However, we have also met high-frequency measurement noise in the descriptions of PID control. In this chapter, we will see how this noise is amplified by the pure derivative of the textbook PID law, and how it is necessary to introduce a filter on the derivative term to reduce noise amplification.

3. *Real actuators may not respond well to rapidly changing input signals.*

For example, if we use a derivative term in the forward path this creates unwanted spikes on the actuator input if the reference signal changes in a stepwise manner. This is called derivative kick. We remove derivative kick by moving the derivative term to a different position in the control loop. Thus the architecture or structure of the PID controller is changed. This is one example of the structural flexibility that we find in PID control. Over the years, this flexibility in PID has been exploited by industrial engineers to create an extensive family of controllers and we need to learn how to recognise and interpret the different structures.

To summarise, we list in Table 18.4 the various PID problems and the solutions proposed which will be detailed in the next sections. These are all part of the art of implementing industrial PID control.

Table 18.4 The issues in implementing an industrial PID controller.

Problem	Solution
Nonlinear effects in real processes, particularly saturation in controller and actuators. This leads to <i>integral wind-up</i> which causes excessive overshoot.	<i>Anti-wind-up circuits</i> to mitigate the wind-up effect.
Measurement noise in the output. These high-frequency signals are amplified by the pure derivative term of the textbook PID.	Use a <i>bandwidth limited derivative</i> term to prevent measurement noise amplification.
Use of proportional and derivative terms in the forward path. This causes rapid changes or spikes in the control signal when the reference signal changes stepwise. These effects called <i>proportional and derivative kick</i> .	Move the proportional and derivative terms to different positions in the controller. This <i>structural flexibility</i> has led to new forms of PID controllers. Other structural formats include reverse acting PID controllers.

18.4 Integral wind-up and anti-wind-up circuits

Integral wind-up is an effect caused by real actuators having an input–output characteristic involving saturation or limiting of the actuation output. We know, for example, that a valve operates between the positions of being fully closed and fully open and in between these two states has an output that we often treat as being approximately linearly related to the input. Similarly, we know that a d.c. motor driving a rotating shaft may have its rotational speed limited to prevent runaway or over-speed situations which might cause equipment damage. In both of these practical examples, the actuator control signal will produce a limited or saturated actuator output. This type of nonlinear characteristic is shown in Figure 18.6, where we see how the input has been capped by the saturation characteristic.

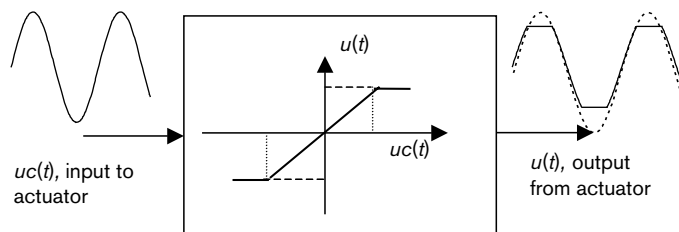


Figure 18.6 The input–output effect of the saturation characteristic.

18.4.1 Understanding the integral wind-up effect

To understand the integral wind up effect, we look at the cascade of a proportional and integral controller, with a saturating actuator as in Figure 18.7.

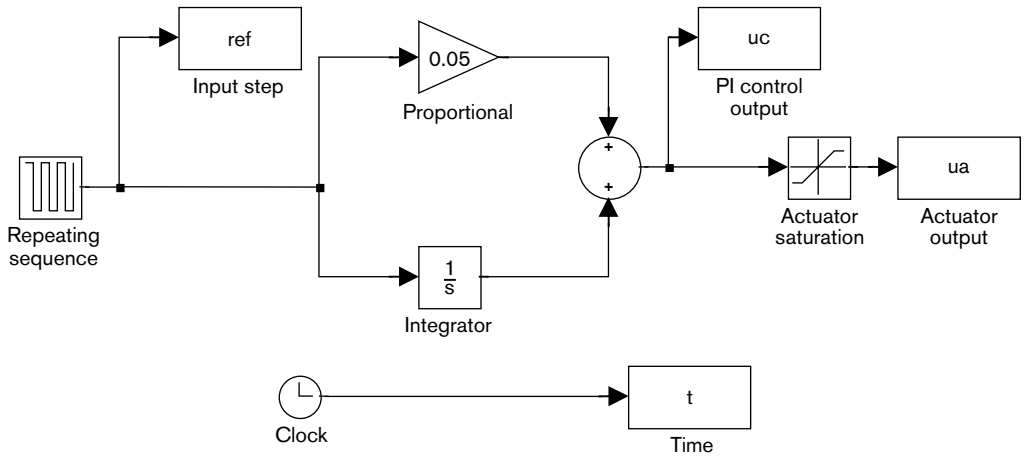


Figure 18.7 Simulink simulation for wind-up effect of the saturation characteristic on PI control.

The input step

We assume that a switched step signal is applied at the input to the proportional and integral controller, as seen in Figure 18.8(a). We can see that we have an input step, set to +1 at $t = 0$ and switching to -1 at $t = 10$. The output from the proportional and integral controller is shown in Figure 18.8(b). Here we can see that the integrator term in the controller integrates up the step so that a peak occurs at $t = 10$ s, when the integrator starts the long descent after the step has changed sign. We call this effect the *wind-up* of the integrator followed by the *unwinding* of the integrator output.

The integral control output $u(t)$ is now used by the saturating actuator to produce the actuator output $u_a(t)$. This is shown in Figure 18.8(c). We can see how the saturating actuator clips or limits the full extent of the control signal. The two implications of this control signal are:

1. If the system is in closed-loop control and the control signal enters a saturation region, then the system reverts to open-loop control. Firstly, we should note that this is potentially dangerous for open-loop unstable systems. Secondly, the application of this constant signal input is similar to a step signal entering the system, and we usually find that this causes an excessive overshoot in the process output. For this example, this constant signal input occurs in the period from $t = 2$ s to $t = 18$ s.
2. In the constant signal period, the controller is really demanding action that the actuator cannot deliver. The integrator is *winding up* in the period $t = 2$ s to $t = 10$ s. At $t = 10$ s, the error $e(t)$ changes from positive to negative and the PI controller output is at $u(t) = 10$. The integrator now begins to *unwind* until $t = 18$ s. At this time, we see that the control signal enters the linear region of the saturation actuator and the controller becomes effective once more. These stages are shown in Figure 18.8(d).

These two reasons lead us to realise that the winding up and unwinding of the integrator delays the effective action of the control signal. What we need to do is switch off the integral term as soon as the control signal enters the saturation regime and prevent the wind-up effect. Thus, an anti-wind-up circuit is simply a scheme for switching on and off the integral term when saturation of the actuator output occurs.

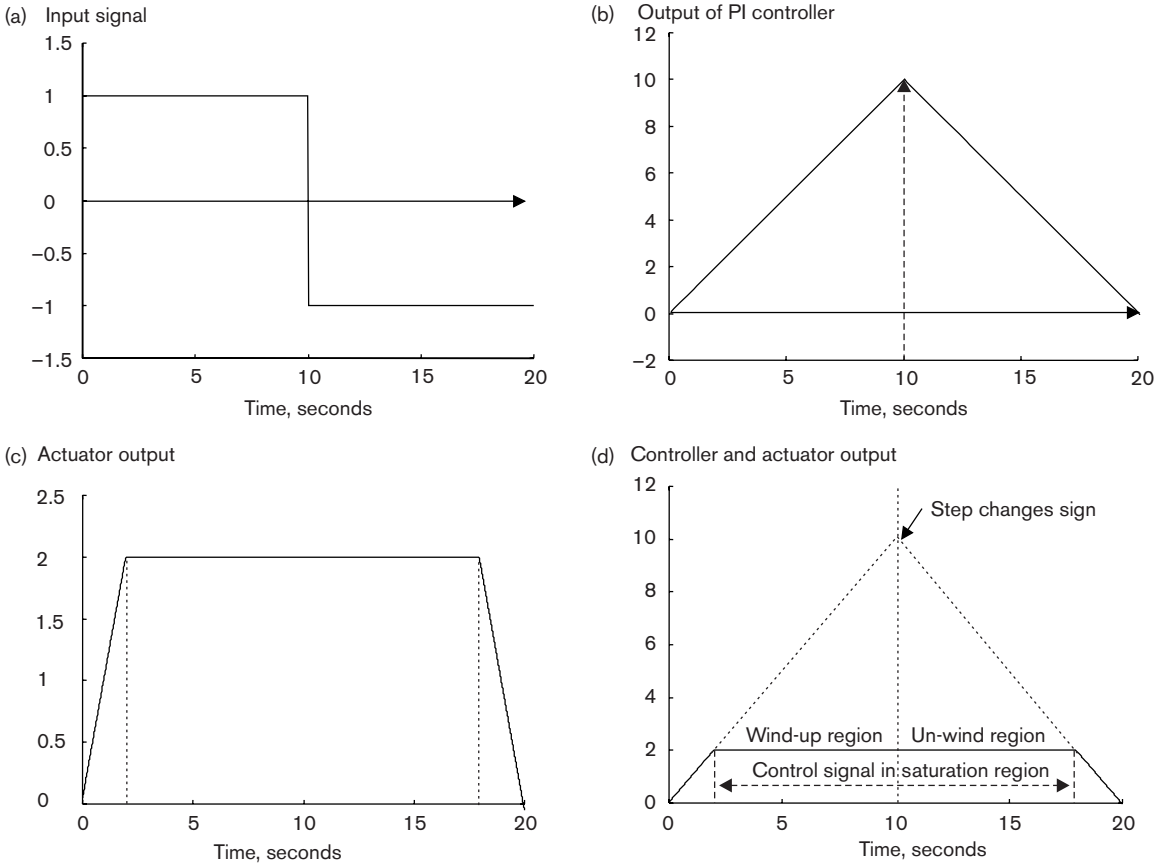


Figure 18.8 The effect of actuator saturation on the PI controller output. (a) The step input signal; (b) the PI controller output; (c) clipped actuator output; (d) comparison of controller and actuator outputs.

18.4.2 A simple anti-wind-up circuit

To correct for integral wind-up, an anti-wind-up circuit is commonly used. As we have already determined, the purpose of this circuit is to detect when actuator saturation is reached and then switch off the integral action. When the control signal returns to the linear region of the saturation characteristic, then the integral action should be resumed once more. Many engineers have used their ingenuity on this problem and found different ways to design an anti-wind-up circuit. Some of these designs are commercially proprietary to particular controllers. In most PID controllers on the market, an anti-wind-up circuit will be present, but the details of the circuit will not usually be released to the end-user. Indeed, it is often sufficient for us to know that one has been used.

A simple anti-wind-up circuit for PI control is shown in Figure 18.9. We have introduced a *multiplier* in this circuit:

$$e_{sw}(t) = S_w \times K_p e(t)$$

If S_w is 0, then the input to the integrator is zero and we have effectively stopped the integral action. The aim of this circuit is to switch off the integrator action whenever $u_{pi}(t)$

drives the actuator into the saturation region and to let $S_w = 1$ for normal operation when the actuator is not saturated. From Figure 18.9, we have:

1. If $u_{pi}(t)$ remains in the linear unsaturated region, $-u_{max} < u_{pi}(t) < u_{max}$, then $e_{pi}(t) = u_{max} - \text{abs}(u_{pi}(t)) > 0$ and the switch signal is $S_w = 1$.
2. If $u_{pi}(t)$ enters the saturation region: either $u_{pi}(t) \geq u_{max}$ or $u_{pi}(t) \leq -u_{max}$ then $e_{pi}(t) = u_{max} - \text{abs}(u_{pi}(t)) < 0$ so that the switch signal is $S_w = 0$. This disengages the integrator action and the integrator output is held at a constant value. The actuator output takes the saturation level at this time. This situation is maintained until $K_p e(t)$ brings $u_{pi}(t)$ back inside the $\pm u_{max}$ linear region. When this happens then case 1 above occurs and the integral action is resumed.

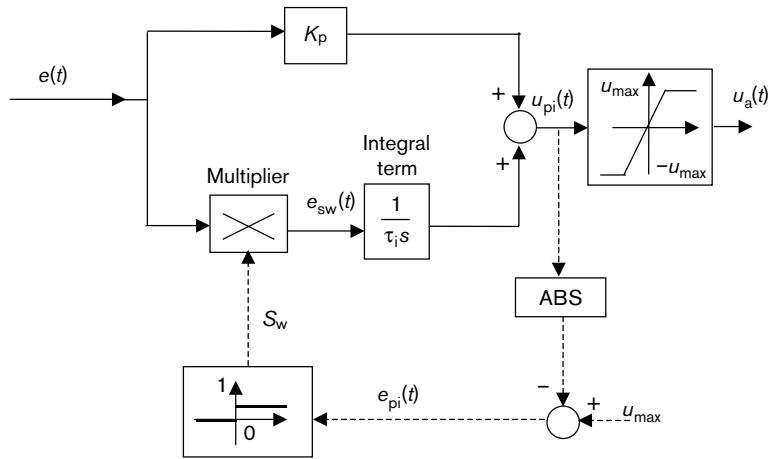


Figure 18.9 A simple anti-wind-up circuit for PI control.

Problem A temperature control loop in an ethylene plant has first-order system dynamics given by

$$G(s) = \frac{8.55}{0.55s + 1}$$

where the time constant is 0.55 min. The process is found to have a measured time delay of 0.15 minutes. The steam supply valve for this loop has a saturation characteristic with unit slope, and a maximum control output range of $-1.7 < u_{output} < 1.7$. Plant operators have reported that a reference change of 10 °C for the loop temperature output produces an excessive temperature transient. This transient was continually activating temperature monitor alarms at the SCADA system console. The works manual shows that a PI controller is in use on the loop and that no anti-wind-up circuit had been incorporated with the design. The PI controller is given as $G_{pi}(s) = 0.09 + (0.95/s)$.

1. Use Simulink to investigate the current performance of the loop.
2. Use the simple anti-wind-up circuit of Figure 18.9 to see if any improvement is possible. Develop the appropriate Simulink simulation and quantify the performance changes in terms of the settling time, percentage overshoot and rise time.

Solution 1. The data of the problem loop can be collected together as

Process transfer function: $G_{\text{plant}}(s) = 8.55/(0.55s + 1)$

Process transport delay: $T_d = 0.15$ minutes

PI controller transfer function: $G_{\text{pi}}(s) = 0.09 + (0.95/s)$

Actuator saturation characteristic, unit slope: $u_{\text{max}} = 1.7$

Temperature reference = 10 °C

A Simulink simulation of the control loop is shown as Figure 18.10.

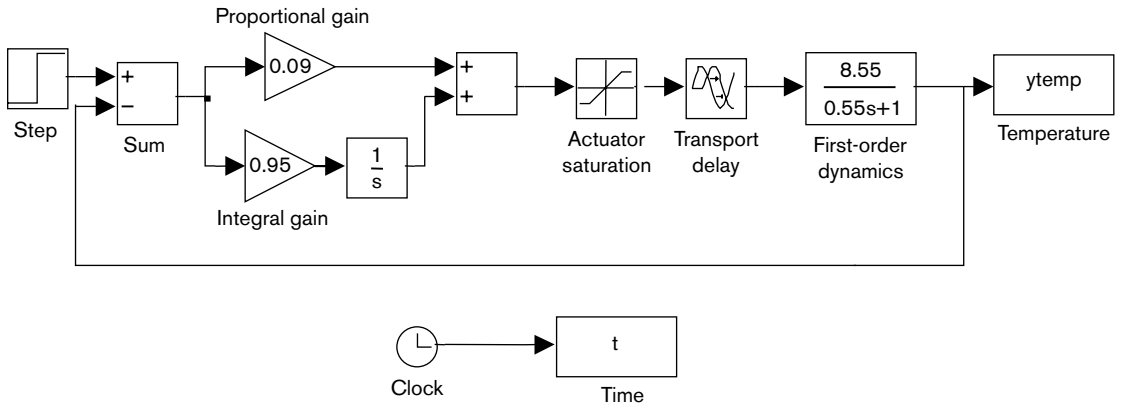


Figure 18.10 A Simulink simulation for the temperature control loop.

The temperature reference change response is shown in Figure 18.11.

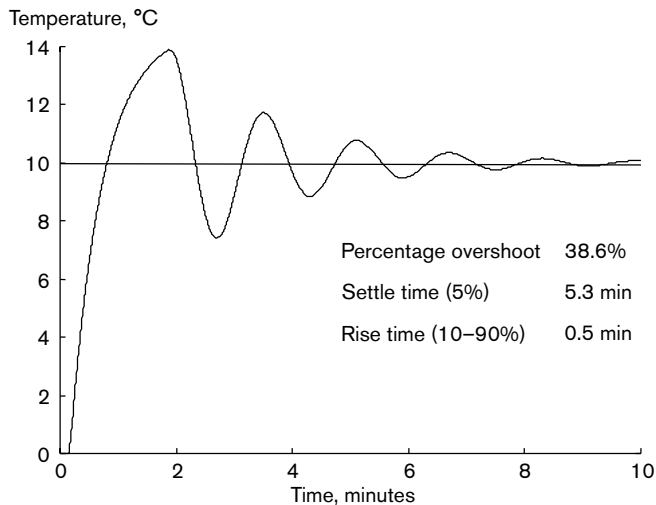


Figure 18.11 Reference response, 10 °C step change.

We can see from the response of Figure 18.11 and the performance figures that there is an excessive overshoot in temperature of nearly 40% and that the response takes just over five minutes to settle. This clearly correlates with the observations of the plant operating staff, since it will be the overshoot in temperature which is setting off the monitoring alarms.

2. A Simulink realisation of the simple anti-wind-up circuit of Figure 18.9 interfaced to the system description of the temperature loop is shown in Figure 18.12.

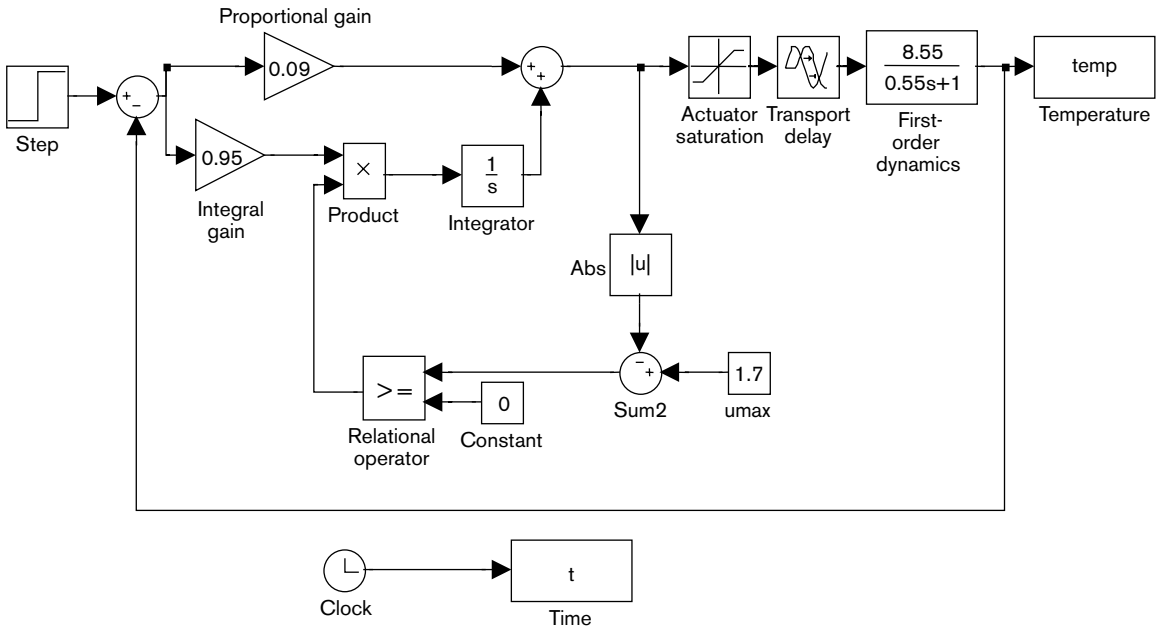


Figure 18.12 A Simulink realisation of the simple anti-wind-up circuit and the PI controlled temperature loop.

The temperature reference change response is shown in Figure 18.13. We see that the introduction of the anti-wind-up circuit has reduced the overshoot to 17% and reduced the settle time to just under three minutes. The reset time remains the same, and so the speed of response is unchanged. Overall, we have a much-improved control loop performance.

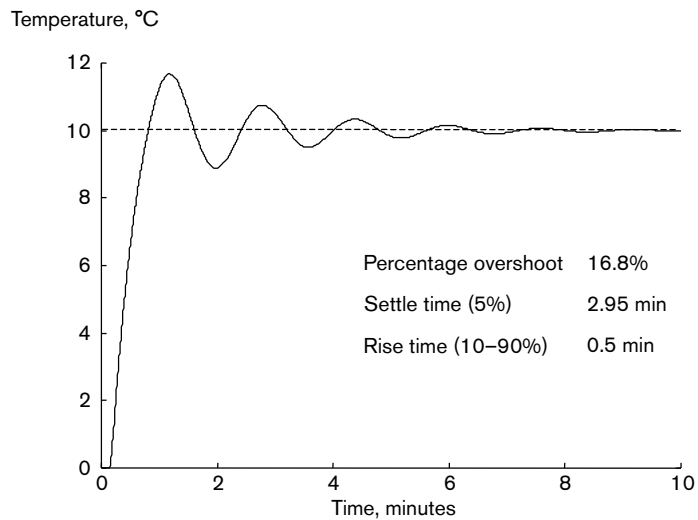


Figure 18.13 Reference response, 10 °C step change; anti-wind-up circuit installed.

18.5 Implementing the derivative term

In the PID controllers that we have looked at so far, we have seen that the controller acts on a reference signal error given by, $e(t) = r(t) - y_m(t)$, where $r(t)$ is the reference input and $y_m(t)$ is the measured output variable.

If we look at real control applications, we find that the feedback signal $y_f(t)$ is the sum of the measured output $y_m(t)$ and a measurement noise component $y_n(t)$, shown as a dashed line in Figure 18.14. The error signal, $e(t)$, can then be written as follows, $e(t) = r(t) - y_f(t) = r(t) - (y_m(t) + y_n(t)) = e_f(t) - y_n(t)$. What we have found from this expression is that the error signal contains a corrupting noise signal. This noise component has important implications for the use of the derivative term in PID controllers.

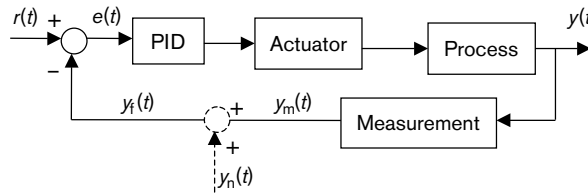


Figure 18.14 Block diagram of PID control applied to system with measurement noise.

We consider a time-constant form of the PID controller:

$$U(s) = K_p[1 + (1/\tau_i s) + \tau_d s] E(s)$$

The differential term (without the gain K_p) is given by

$$U_d(s) = \tau_d s E(s) = G_d(s) E(s)$$

If we consider the Bode magnitude plot of $G_d(s)$ in Figure 18.15, where we have let $\tau_d = 1.0$, we find that the plot shows that the derivative term produces amplification of the

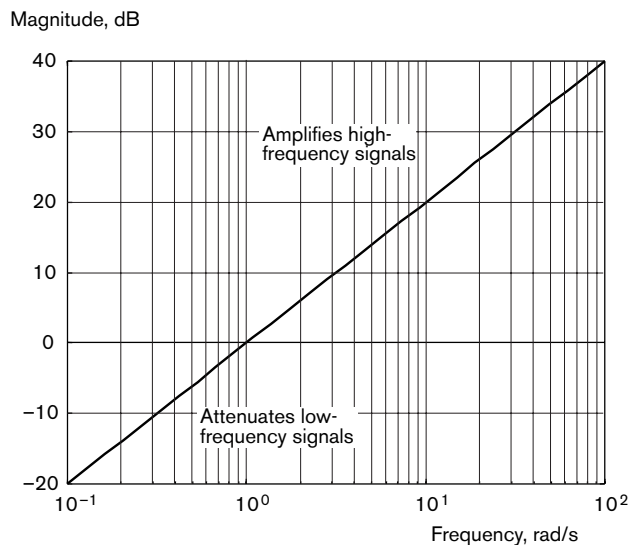


Figure 18.15 Magnitude plot of derivative term.

high-frequency signals. These high-frequency signals often come from the measurement noise within the system. We have also learnt in previous chapters that the differential term has no effect on steady (constant) signals. We can also see this from the Bode plot. In the low-frequency range we find that the differential term has very small gain values. It is this near-zero gain which attenuates low-frequency signals.

The consequence of the possibility of measurement noise being present within the system is that we do not, in practical applications, apply the derivative directly to the measured output of the process. Instead, we introduce a *low-pass* filter in the D-term. A low-pass filter has the effect of attenuating high-frequency signals.

We incorporate a low-pass filter of the form $G_f(s) = 1/(\tau_f s + 1)$ into the derivative term $U_d(s)$ as follows:

$$U_d(s) = G_f(s)(\tau_d s)E(s) = \frac{\tau_d s}{\tau_f s + 1}E(s) = G_{md}(s)E(s)$$

where $G_{md}(s)$ represents the modified derivative transfer function. We note that the high-frequency gain of $G_{md}(s)$ is given by

$$K_{high} = \lim_{\omega \rightarrow \infty} |G_{md}(j\omega)| = \frac{\tau_d}{\tau_f}$$

If we parametrise the filter time constant in terms τ_f in terms of τ_d by the formula $\tau_f = \tau_d/N$, we obtain a high-frequency gain given by $K_{high} = \tau_d/\tau_f = N$. The final modified derivative term is then given as

$$G_{md}(s) = \frac{\tau_d s}{(\tau_d / N)s + 1}$$

In commercial PID controller devices, the value of N is selected to be in the range $5 \leq N \leq 20$ depending on manufacturer. However, the important fact is that we have created a new modified D-term with limited high-frequency amplification of noisy signals.

We can see the effect of the low-pass filter when we look at the Bode magnitude plot of $G_{md}(s)$ in Figure 18.16. This plot shows the magnitude of $G_{md}(s)$ with N set to 5. From our analysis, we know that the high-frequency gain is given by N or, in dB, $20 \log_{10}(N) = 20 \log_{10}(5) = 13.9794 \approx 14$ dB. We can see from the magnitude plot that the high-frequency gain is indeed 14 dB.

We conclude this section by setting the formula for the modified derivative control term in the full PID controller form to obtain:

$$U(s) = K_p \left(1 + \frac{1}{\tau_i s} + \frac{\tau_d s}{(\tau_d / N)s + 1} \right) E(s)$$

where K_p is the proportional gain, τ_i is the integral time constant, τ_d is the derivative time constant and N is a derivative filter parameter satisfying $5 \leq N \leq 20$. We should note that different manufacturers often use different forms for the modified D-term and even different notation for the PID controller itself. As an example, the SCADA interface shown in Figure 18.3 earlier in the chapter used the notation, G, TI, TD, TF for the terms K_p , T_i , τ_d and τ_f , and the PID formula used by this manufacturer was

$$U(s) = [G_{pid}(s)]E(s) = G \left[1 + \frac{1}{TIs} + \frac{TDs}{TFs + 1} \right] E(s)$$

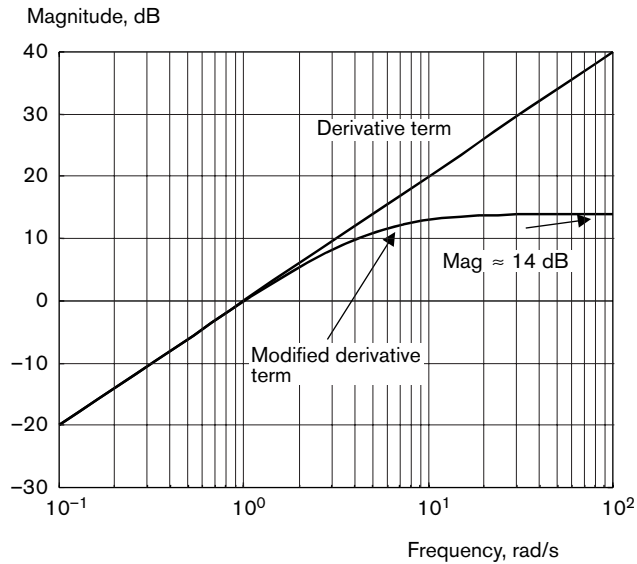


Figure 18.16 Bode plot for the pure differential term, $G_d(s) = s$, and the modified differential term, $G_{md}(s) = s/(0.2s + 1)$.

18.6 Industrial PID controller structures

We have looked at how the individual PID controller terms are implemented for use in industrial applications. We have looked at the specific ways in which the integral and derivative terms are changed to overcome wind-up and noise amplification effects respectively. Again, motivated by the practical problems of industrial control, we now examine the flexibility of the structure of the PID controller itself. This eventually leads to the larger family of PID controllers that are widely available for industrial applications.

18.6.1 Proportional kick

We find that the structural innovations introduced to the PID controller are based on direct experience of using these controllers in industrial applications. In this section, we follow an empirical investigation using the Simulink software to show why these structural changes were made; we start with *proportional kick*. We use a simple Simulink simulation for the textbook PI control of a first-order process model. This is shown in Figure 18.17, where a unit step change occurs in the reference signal at $t = 1$.

If we look at the process output plot as shown in Figure 18.18, we find, as might be expected, a first-order-like step rise to the steady output value of unity. We know that the integral term in the PI controller will ensure no steady state offset, and that is what we find in the output plot. But now if we look at the control signal plot in Figure 18.18, we see a surprising spike in the signal. This is *proportional kick*. The term *kick* refers to the effect the control signal will have at the input to the actuator. For example, this might be

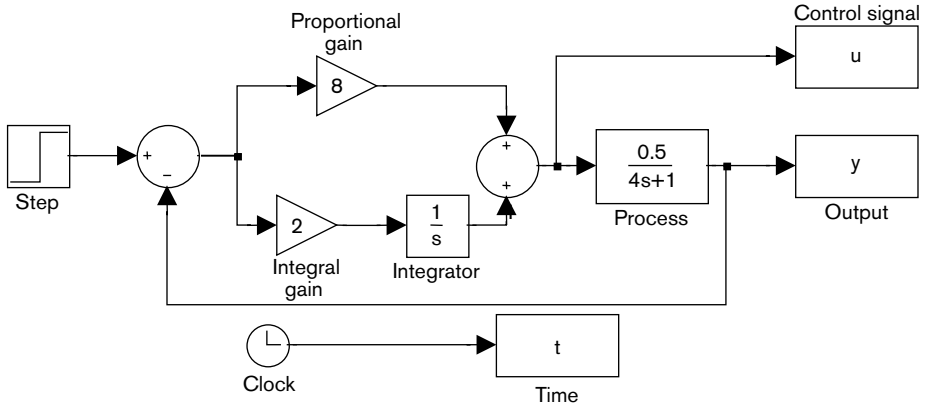


Figure 18.17 Standard textbook PI control of a first-order process model.

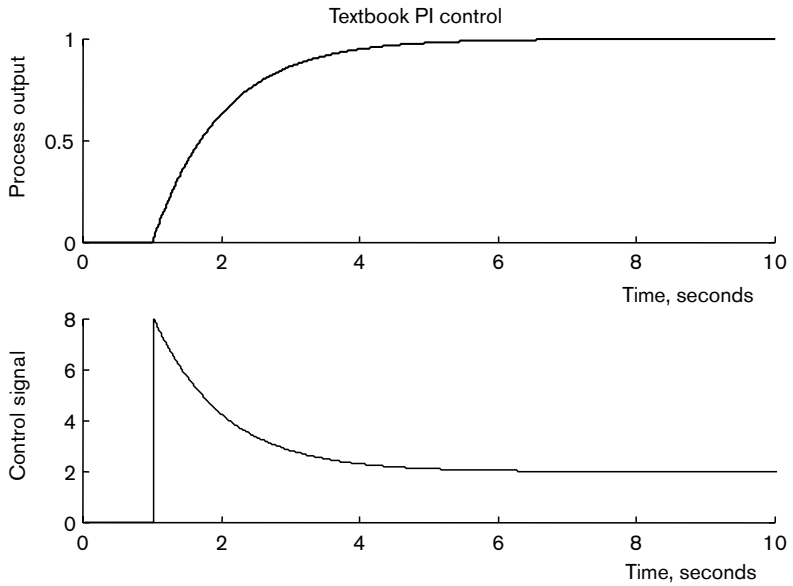


Figure 18.18 Output and control signals in the standard PI control of a first-order process model.

a kick in the current controlling a valve actuator and, as can be imagined, the effect of the sudden increase in current could be quite damaging.

To remove the proportional kick, the remedy is to move the proportional term into the feedback path so that it does not directly operate on reference changes (Figure 18.19).

This move restructures the form of the PI controller. The formula for this modified controller is now

$$U(s) = -K_p Y(s) + \frac{K_i}{s} E(s)$$

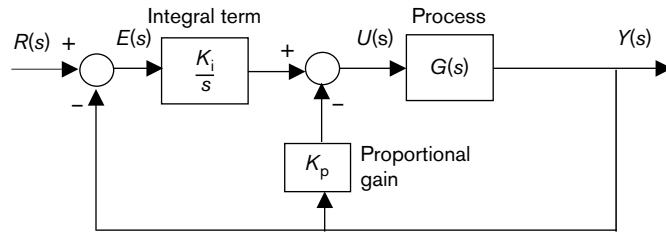


Figure 18.19 Restructuring the PI controller.

Using the terminology of process control, the system output is often called the process variable and denoted 'PV', the reference signal is called the set point, denoted 'SP', and the error, $e(t) = r(t) - y(t)$, is termed set point error and denoted 'E'. The restructured PI controller is then referred to as 'P on process variable and I on set point error'. To see that this new structure does in fact remove the proportional kick, we use the Simulink simulation of Figure 18.20.

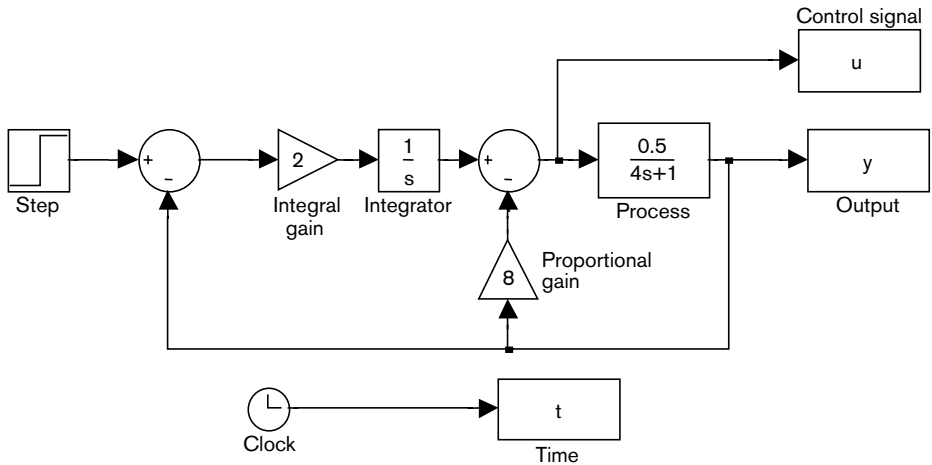


Figure 18.20 The P on process variable, I on set point error controller used on a first-order process model.

The step response signals are shown in Figure 18.21. Comparing Figures 18.18 and 18.21, we immediately see that the spike due to proportional kick no longer appears in the control signal of the 'P on process variable and I on set point error' controller. But if we look at the output plot from the new controller, we see that the speed of response is much slower.

We can experiment with new tuning for the I-term since we want the steady state value to be reached quicker. If we increase the I-gain to 6, the output and control signals of Figure 18.22 are obtained. Here we can see that the output response has a similar performance to the original textbook PI controller, yet the control signal is much smoother and although an overshoot has appeared the signal is devoid of any *kick* effect.

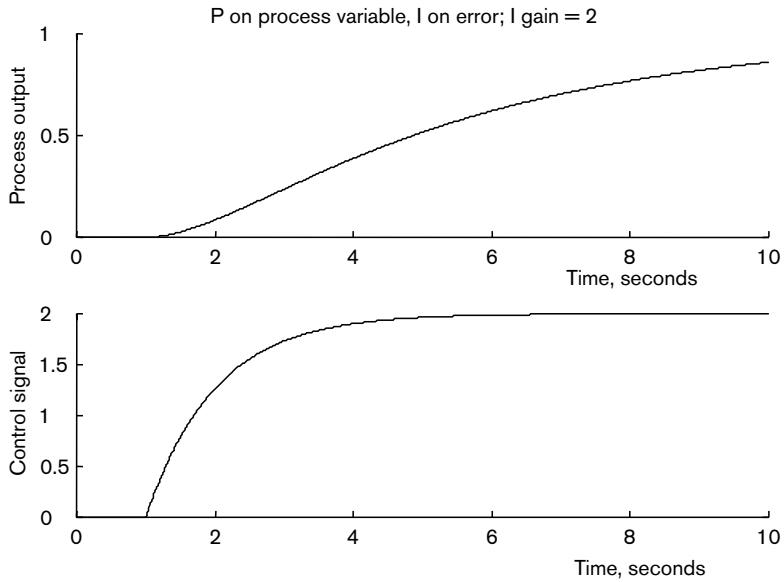


Figure 18.21 Output and control signals in the restructured PI control of a first-order process model. Controller removes proportional kick.

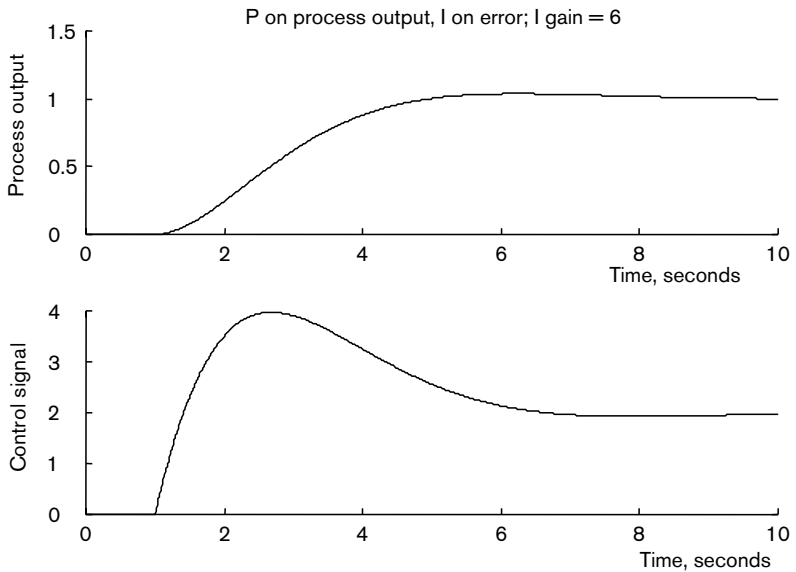


Figure 18.22 Output and control signals from the re-tuned controller which removes proportional kick.

18.6.2 Derivative kick

We use derivative control to enhance the closed-loop stability of a loop and to shape the response by tuning the damping in systems with approximately second-order dynamics. We have already seen how it is necessary to modify the formula of the D-term to avoid problems with measurement noise amplification. Now we look at practical implications

of the position of the D-term in the usual textbook structure of the PID controller. In Figure 18.23 we have a Simulink model of PID control of a simple first-order process where the controller is in the usual forward path position. We have used the modified D-term which incorporates noise filtering. If we write a general transfer function for the PID controller of Figure 18.23, we would have:

$$U(s) = G_{\text{pid}}(s)E(s) = \left(K_p + \frac{K_i}{s} + \frac{K_d s}{\tau_f s + 1} \right) E(s)$$

where $K_p = 2.6$, $K_i = 0.763$, $K_d = 0.224$ and $\tau_f = 0.112$.

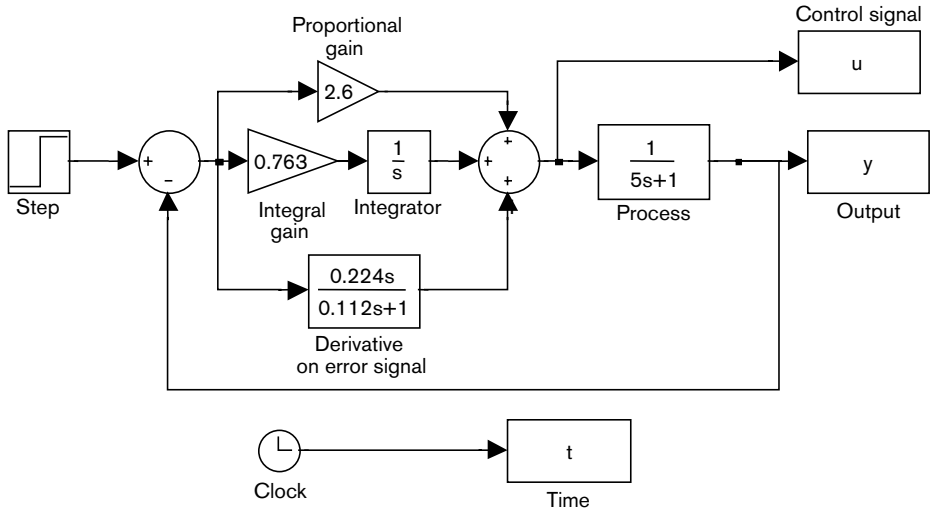


Figure 18.23 A Simulink model of standard PID-on-error applied to a first-order process.

We run the simulation to look at the process variable output signal and the control signal. The signal plots are shown in Figure 18.24, where a step change in reference has occurred at $t = 1$.

The process output response looks satisfactory, the I-term ensures there is no steady state error and the rise time is about 4 time units. A look at the control signal in Figure 18.24 shows a remarkably different interpretation of the apparently benign output response. We see a sharp spike on the control signal at the time of the step reference change. This is a *derivative kick* effect. In practice this control signal could be driving an actuator device like a motor or a valve, and the kick would create serious problems for any electronic circuitry used in the device. Derivative kick is very similar in origin to proportional kick, explained in the previous section. The remedy for derivative kick also follows that used for proportional kick, so that we move the derivative term into the feedback path and restructure the PID controller (Figure 18.25).

Using the process control terminology, we call this new controller a PI-D controller, with 'P and I on set point error and D on process variable'. The formula for the PI-D controller is given as

$$U(s) = \left(K_p + \frac{K_i}{s} \right) E(s) - \frac{K_d s}{\tau_f s + 1} Y(s)$$

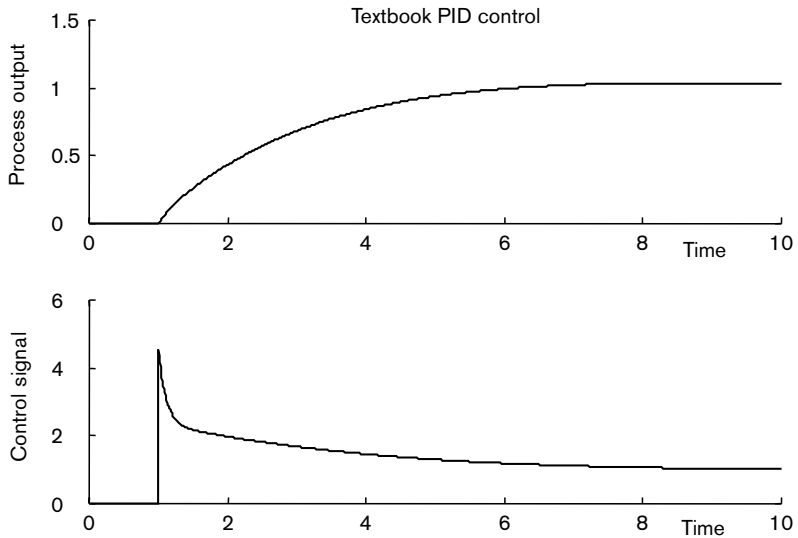


Figure 18.24 Output and control signals in the standard PID control of a first-order process model.

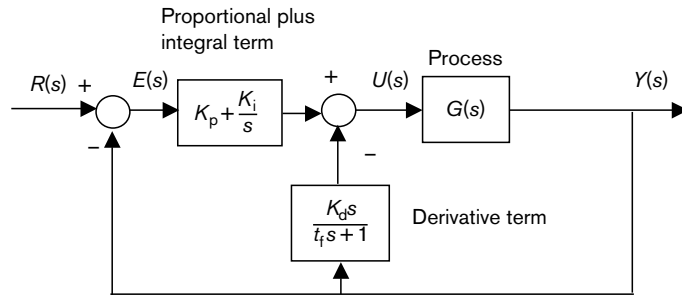


Figure 18.25 PID structure: PI on error, D on process variable.

The modified Simulink simulation is shown in Figure 18.26, and the associated output and control signal traces are shown in Figure 18.27.

Comparing the control signals of Figures 18.24 and 18.27, we immediately see that the spike due to derivative kick is much reduced in the new PI-D controller response. In the standard PID control signal the spike reached a peak of 5 units and was very narrow and sharp, while in the PI-D control signal the peak is 2.7 units and appears as a gentle exponential decay. The difference in the two process output signals is hardly noticeable. The restructuring of the standard PID controller into the PI-D controller to avoid derivative kick is a second example of how industrial engineers have modified the PID control method to solve practical implementation problems. This structural flexibility is discussed in more detail in the next section.

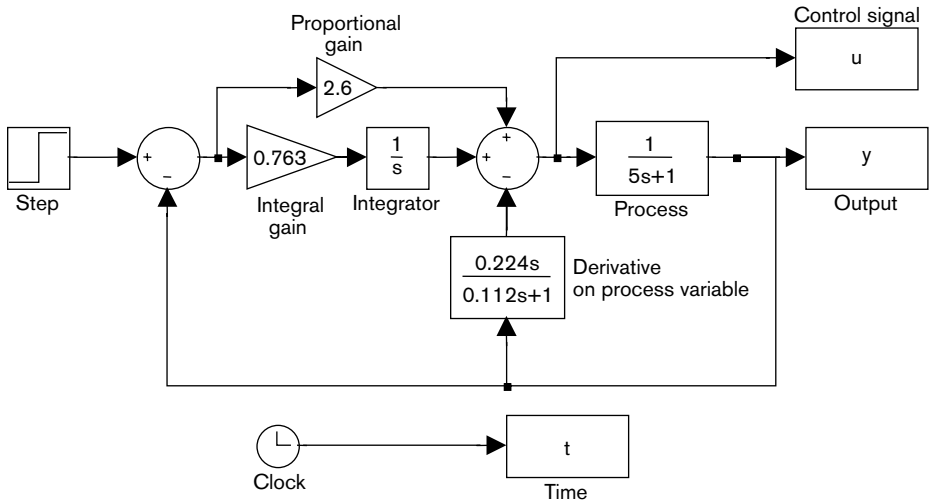


Figure 18.26 Simulink model with PI on error and D on process variable.

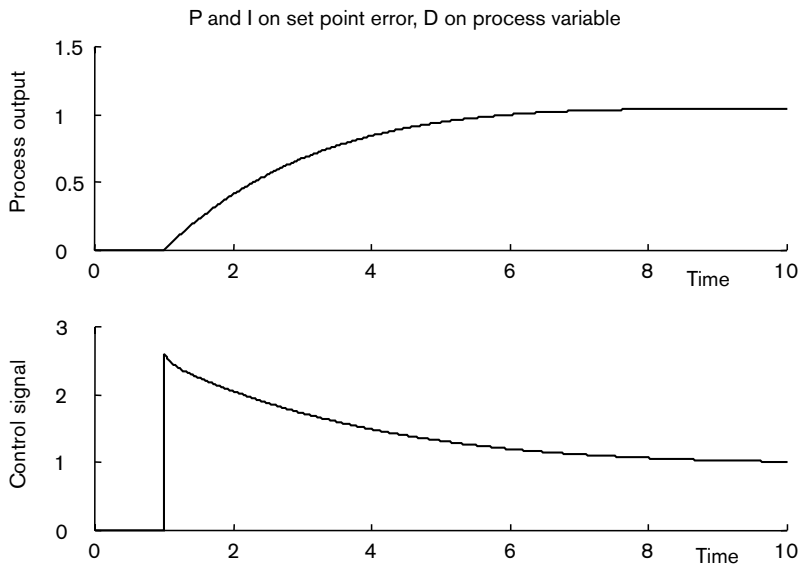


Figure 18.27 Output and control signals in the PI-D control of a first-order process model. Controller removes derivative kick.

18.7 Different forms of industrial PID controllers

We saw in earlier sections how the PID controller was modified to overcome practical implementation problems like integral wind-up and measurement noise amplification. We also saw how industrial engineers have exploited the structural flexibility in the PID framework to solve the problems associated with proportional and derivative kick. In this section we are going to explore the structural flexibility in a more systematic way

and finally define the parallel (non-interacting) PID structure and the series (interacting) PID structure to complete the description of the possibilities in the PID framework.

We learnt in the sections on proportional and derivative kick how the PID controller was restructured to avoid these effects. As an example, to avoid both proportional and derivative kick a PID controller would be restructured as 'I on set point error with P and D on process variable'. The name for this PID controller would be hyphenated and given as the I-PD controller. The letters before the hyphen refer to the terms acting on the set point error, and the letters after the hyphen refer to the terms acting on the process variable. To take a second example, to avoid derivative kick alone, the PID controller could be structured as 'P and I on set point error and D on process variable'. This PID controller would be given the name PI-D controller. In this example, the PI before the hyphen indicates that the PI terms act on the set point error, and the D after the hyphen shows that the D-term acts on the process variable. Clearly, in this way we can conveniently label a family of PID controllers with different structures and different engineering properties. Typically we might find the controllers P, I, PI, PD, PID, PI-D, I-P and I-PD listed in an industrial control manual. The final feature of the PID control structure is whether the controller is in parallel (non-interacting) or series (interacting) form.

18.7.1 A parallel PID controller

The structure of the textbook form of PID controller that we have been using throughout this book has three distinct parallel paths, each path carrying only one term of the controller. It is sometimes given other names, like the **decoupled form** or the **non-interacting form**. These names reflect the independence of the parallel paths and their separate effects. Within this parallel structure only the definition of the coefficients and the implementation devices used for the terms I and D will differ between different commercial products. Table 18.5 lists the typical industrial formulas that might be met in practical applications. We use both the control engineering labels and the process control terms.

18.7.2 A series PID controller

Early technological versions of some PID controllers used pneumatic hardware for which a series transfer function representation was appropriate. In later analogue PID devices some manufacturers retained the series structure. Despite the likelihood that today's PID controller will be implemented digitally and in parallel form, these series PID formulas are often still listed in industrial PID controller manuals. The basic series or interacting PID control law is given in terms of a product of transfer functions as

$$U(s) = G_{\text{series}}(s)E(s) = \left[K_s \left(1 + \frac{1}{T_i s} \right) (1 + T_d s) \right] E(s)$$

We can draw this transfer function relation as the block diagram of Figure 18.28.

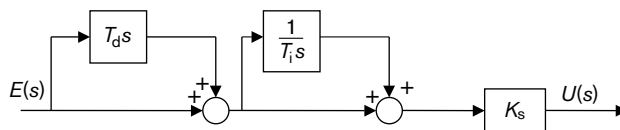


Figure 18.28 Series or interacting PID controller block diagram.

Table 18.5 The parallel PID controller structure: summary of transfer function formulas.

Structure	General notation	Process control industries
P	$U(s) = [K_p]E(s)$	$U = [G]E$
I	$U(s) = \left[\frac{K_i}{s} \right] E(s)$	$U = G \left[\frac{1}{T_i s} \right] E$
PI	$U(s) = \left[K_p + \frac{K_i}{s} \right] E(s)$	$U = G \left[1 + \frac{1}{T_i s} \right] E$
PD	$U(s) = \left[K_p + \frac{K_d s}{\tau_f s + 1} \right] E(s)$	$U = G \left[1 + \frac{TD_s}{TF_s + 1} \right] E$
PID	$U(s) = \left[K_p + \frac{K_i}{s} + \frac{K_d s}{\tau_f s + 1} \right] E(s)$	$U = G \left[1 + \frac{1}{T_i s} + \frac{TD_s}{TF_s + 1} \right] E$
PI-D	$U(s) = \left[K_p + \frac{K_i}{s} \right] E(s) - \left[\frac{K_d s}{\tau_f s + 1} \right] Y(s)$	$U = G \left[\left(1 + \frac{1}{T_i s} \right) E - \left(\frac{TD_s}{TF_s + 1} \right) PV \right]$
I-P	$U(s) = \left[\frac{K_i}{s} \right] E(s) - [K_p] Y(s)$	$U = G \left[\left(\frac{1}{T_i s} \right) E - (1) PV \right]$
I-PD	$U(s) = \left[\frac{K_i}{s} \right] E(s) - \left[K_p + \frac{K_d s}{\tau_f s + 1} \right] Y(s)$	$U = G \left[\left(\frac{1}{T_i s} \right) E - \left(1 + \frac{TD_s}{TF_s + 1} \right) PV \right]$

There are two technical features of the series form of PID controller that we should examine. Firstly, we should look at a time domain formula for the controller, and secondly we should know how to establish the connection with the parallel PID controller.

A time domain formula for the series PID controller

We begin from the transfer function equation of the series controller as

$$U(s) = \left[K_s \left(1 + \frac{1}{T_i s} \right) (1 + T_d s) \right] E(s)$$

If we multiply the expression out, we obtain

$$U(s) = \left[K_s \left(1 + \frac{T_d}{T_i} \right) + \frac{1}{T_i s} + T_d s \right] E(s)$$

We can convert the expanded relation immediately to a time domain form as

$$u(t) = K_s \left[\left(1 + \frac{T_d}{T_i} \right) e(t) + \frac{1}{T_i} \int_0^t e(\tau) d\tau + T_d \frac{de}{dt} \right]$$

From this form we can also see that there is a link with the parallel PID controller formula.

The connection between the series and parallel PID controller formulas

Starting from the expanded form for the series PID controller:

$$U(s) = \left[K_s \left(1 + \frac{T_d}{T_i} \right) + \frac{1}{T_i s} + T_d s \right] E(s)$$

we must factor out the term $(1 + T_d/T_i)$ as follows:

$$U(s) = K_s \left(1 + \frac{T_d}{T_i} \right) \left[1 + \left(1 + \frac{T_d}{T_i} \right)^{-1} \times \frac{1}{T_i s} + \left(1 + \frac{T_d}{T_i} \right)^{-1} \times T_d s \right] E(s)$$

This will simplify to give

$$U(s) = K_s \left(\frac{T_i + T_d}{T_i} \right) \left[1 + \frac{1}{(T_i + T_d)s} + \left(\frac{T_i T_d}{T_i + T_d} \right) s \right] E(s)$$

The usual formula for the parallel PID controller is

$$U(s) = K_p \left(1 + \frac{1}{\tau_i s} + \tau_d s \right) E(s)$$

Comparing the two forms gives the following identities

$$K_p = K_s \left(\frac{T_i + T_d}{T_i} \right)$$

$$\tau_i = (T_i + T_d)$$

$$\tau_d = \left(\frac{T_i T_d}{T_i + T_d} \right)$$

We have discovered that the series interacting PID controller can be unravelled and written in the form of a parallel or non-interacting controller formula. However, past engineering usage means that despite the theoretical simplicity of the parallel form, the seemingly more complicated series form is still used and cited. Some claim that the series form is easier to tune manually, so we have included it here for completeness and possible future reference.

18.8 Reverse acting controllers

Inverse systems and reverse acting controllers are terms that often cause some confusion, so we give a brief reminder of inverse systems and then go on to discuss reverse acting controllers in more detail. Some processes have difficult dynamics, and inverse response systems belong to this class of problems. The most often cited examples of inverse response processes are either the level control of a drum boiler system found in processes requiring a supply of steam like a power station, or an exothermic reactor system from the chemical industry. In such processes, if we make a positive step change at the process input we find that the output step response first goes negative and then recovers to finish up positive. Such a response is shown in Figure 18.29.

We have earlier learnt in Chapter 10 how the inverse system response has a physical origin where two competing effects, a fast dynamic effect and a slow dynamic effect, conspire to produce the negative start to the response before the step recovers to settle at a positive steady state value.

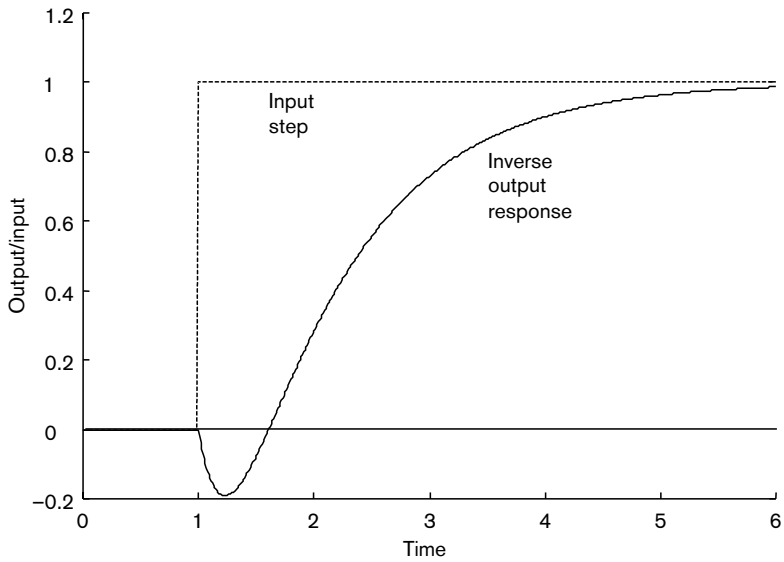


Figure 18.29 Inverse system response.

Reverse acting controllers should not be confused with inverse response processes. Reverse acting controllers come about because the process has a negative gain. In this case, a positive-going step change produces a negative-going step response. We show such a step response in Figure 18.30.

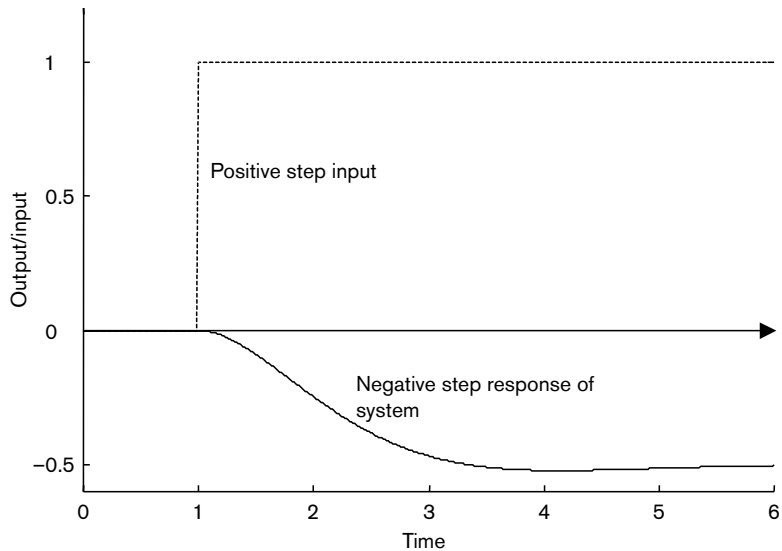


Figure 18.30 Step response of system with negative gain.

If negative feedback is used with a process with negative gain, then positive feedback results and closed-loop stability cannot often be found. To rectify this problem, an additional gain of $[-1]$ is placed at the output of the controller to maintain a negative feedback

loop, as shown in Figure 18.31. This combination of a controller and the $[-1]$ block is called a reverse acting controller.

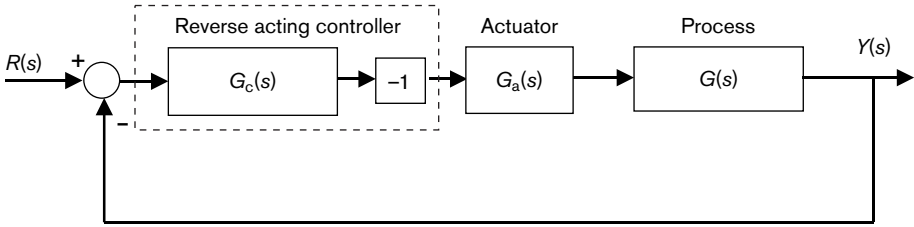


Figure 18.31 Reverse acting controller for process with negative gain.

Problem A control system is required for level control in a valve-controlled tank filling system. The particular system adopted is shown in Figure 18.32.

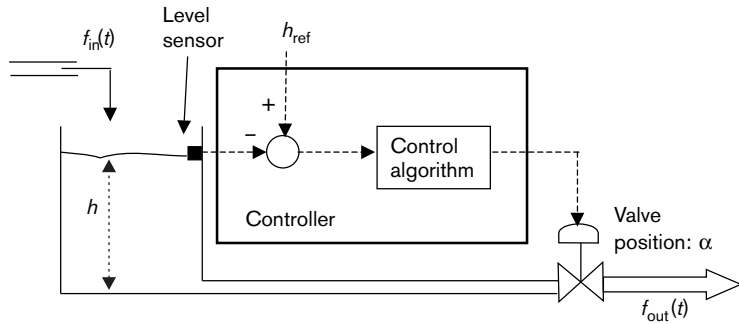


Figure 18.32 Level control system.

The tank is used as a buffer store for the process and has a design operating level of 4 m. From time to time the tank is scoured out and has to be refilled from empty. The supply inflow, $f_{in}(t)$, is subject to small variations about a nominal constant value, f_{IN} . The valve on the outlet pipe is assumed to have a linear saturation characteristic, with the valve setting variable, α , lying between 0 and 1. The rated outflow is given by f_R and is the maximum outflow at the outlet pipe that can be achieved with the valve fully open. The level in the tank is denoted by $h(t)$. The system data are summarised in Table 18.6.

Table 18.6 Data for the level control system.

Variable	Value	Units
Cross-sectional area of tank, A	6	m^2
Nominal constant inflow, f_{IN}	2	m^3/min
Rated outflow, f_R	5	m^3/min
Valve fully closed α	0	dimensionless
Valve fully open α	1	dimensionless

An engineer proposes to design a start-up system and a PI controller for this system.

1. Develop a system model for the tank level control system.
2. Use the data and the model to determine a startup procedure for the system such that the tank level reaches the desired operating level of 4 m.
3. Show that a reverse acting controller is required.
4. Use the data to design a PI controller to achieve closed-loop pole placement at $s = -0.7 \pm j0.7$.
5. Develop a Simulink simulation for assessing the reference tracking and disturbance rejection performance.

Solution

1. *Develop a system model for the tank level control system*

We develop a general system model and then use the data in our calculations. The model we shall derive will be based on physical principles. We begin from a rate of change equation:

Rate of change of liquid volume in the tank = inflow – outflow

The volume of liquid in the tank, $V(t)$, is given by:

$$V(t) = \text{cross-sectional area} \times \text{height of liquid} = A \times h(t)$$

Liquid inflow is given by $f_{in}(t)$. Liquid outflow depends on how far the valve is open at time t and the rated outflow, so that $f_{out} = \alpha(t) \times f_R$.

Substituting into the rate of change equation, we have:

$$\frac{d\{V(t)\}}{dt} = \frac{d\{A \times h(t)\}}{dt} = f_{in}(t) - f_{out} = f_{in}(t) - \alpha(t) \times f_R$$

The cross-sectional area of the tank is constant and therefore we can rearrange this equation as

$$\frac{dh(t)}{dt} = \left[\frac{-f_R}{A} \right] \alpha(t) + \frac{1}{A} f_{in}(t)$$

To complete the model description we need to add the initial condition for the height of the liquid in the tank at time, $t = 0$. This we write as $h(0) = h_0$. The full model description in the time domain is then

$$\frac{dh(t)}{dt} = \left[\frac{-f_R}{A} \right] \alpha(t) + \frac{1}{A} f_{in}(t), \text{ with initial condition } h(0) = h_0.$$

In the Laplace domain, we have directly from the transform of the time equation

$$sH(s) - h_0 = \left[\frac{-f_R}{A} \right] \alpha_v(s) + \left[\frac{1}{A} \right] F_{in}(s)$$

where $H(s)$ is the Laplace transform of $h(t)$ and $\alpha_v(s)$ is the Laplace transform of $\alpha(t)$. This equation rearranges to give the Laplace transfer function model as

$$H(s) = \left[\frac{-K_v}{s} \right] \alpha_v(s) + \left[\frac{K_F}{s} \right] F_{in}(s) + \left[\frac{1}{s} \right] h_0$$

where $K_v = f_R/A$ and $K_F = 1/A$.

2. Use the data and the model to determine a startup procedure

The tank start up assumes that the tank is empty. This means that $h(0) = h_0 = 0$. The operating level of the liquid is to be 4 m. The data table shows that the constant nominal inflow is fixed at $f_{IN} = 2 \text{ m}^3/\text{min}$. We can describe *startup* as filling the tank to a height of 4 m from empty. Since $A = 6 \text{ m}^2$, a level of 4 m corresponds to a volume of 24 m^3 . For the shortest start up time, we simply close the outlet valve by setting $\alpha = 0$ and allow the inflow to run for $24/2 = 12$ minutes. After 12 minutes we must then either:

- (i) alter the valve position manually in *open-loop control* to give the required position such that the level remains at 4 m.
- (ii) switch in the controller which acts on the feedback level position signal. This controller will vary the valve position to maintain the level at 4 m.

To stay in steady conditions at the $t = 12$ min point *without* switching on the controller, we must switch the valve setting to a particular value α_{ss} , which corresponds to a steady state condition where the level, $h(t)$, does not change. We can find the steady state value for α_{ss} from the time domain model by setting $dh/dt = 0$ and substituting in the nominal steady value of the supply input, f_I , as follows:

$$\left. \frac{dh}{dt} \right|_{ss} = 0 = \left(\frac{-f_R}{A} \right) \alpha_{ss} + \frac{1}{A} f_{IN}$$

We see that this rearranges as $\alpha_{ss} = f_{IN}/f_R = 2/5 = 0.4$. Therefore we can list a startup procedure as:

Startup procedure

Turn off the inflow stream

Ensure the tank is empty

Close out flow valve, setting $\alpha = 0$

Turn on the inflow stream and timer simultaneously

Run for 12 minutes

At 12 minutes:

EITHER: Set valve to $\alpha = \alpha_{ss} = 0.4$, giving open-loop control

OR: Switch on PI controller, giving closed-loop control

3. Show that a reverse acting controller is required.

The controller will regulate the level of liquid in the tank. The steady conditions have already been determined so that the steady liquid level is $h_{ss} = h_{ref} = 4$ m. The valve setting to achieve this is $\alpha_{ss} = 0.4$. We have also determined that this steady level is reached at $t_0 = 12$ min and that the starting level is $h(t_0) = h(12) = h_{ref} = 4$. The model equations are:

$$H(s) = \left[\frac{-K_v}{s} \right] \alpha_v(s) + \left[\frac{K_F}{s} \right] F_{in}(s) + \left[\frac{1}{s} \right] h_0$$

where

$$K_v = \left(\frac{f_R}{A} \right) \text{ and } K_F = \left(\frac{1}{A} \right)$$

Now to demonstrate that a reverse acting controller is required. As we can see from the transfer function model, the link from control valve input to change in liquid level contains a minus sign, and the feedback system is going to require a reverse acting controller. Let us assume that we do

not realise this and use the usual unity feedback system with a controller shown as in Figure 18.33. The controller output $\alpha(s)$ combines both the steady state valve position for the operating point of 4 m plus a proportional term which is applied to variations around 4 m:

$$\alpha(s) = \alpha_{ss} + K_p E(s)$$

The control signal contains an offset and a proportional term and is a simple way of having proportional control with the offset α_{ss} acting in place of an integral term. We have used this form here so that we can do some simple analysis of the feedback loop with just proportional control. For the moment we ignore the presence of the valve saturation characteristic in the loop and conduct a purely linear analysis.

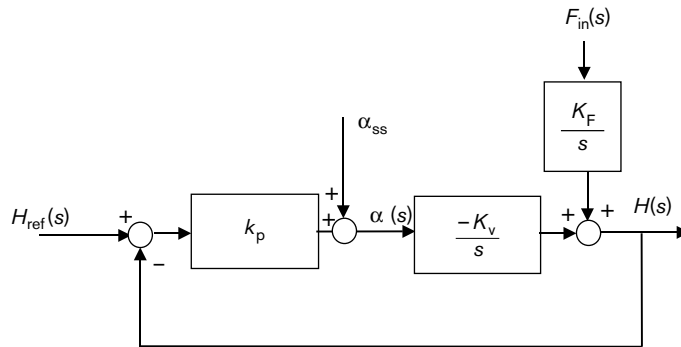


Figure 18.33 Unity gain feedback control for level system.

The closed-loop expression between $H(s)$ and the input signals can be derived from:

$$H(s) = \frac{-K_v}{s} \alpha(s) + \frac{K_F}{s} F_{in}(s)$$

$$\alpha(s) = \alpha_{ss} + k_p (H_{ref}(s) - H(s))$$

Substituting, we have:

$$H(s) = \frac{-K_v}{s} [\alpha_{ss} + k_p (H_{ref}(s) - H(s))] + \frac{K_F}{s} F_{in}(s)$$

$$H(s) = \frac{-K_v}{s} \alpha_{ss} - \frac{-K_v k_p}{s} (H_{ref}(s) - H(s)) + \frac{K_F}{s} F_{in}(s)$$

$$H(s) \left[1 - \frac{K_v k_p}{s} \right] = \frac{-K_v}{s} \alpha_{ss} - \frac{-K_v k_p}{s} H_{ref}(s) + \frac{K_F}{s} F_{in}(s)$$

giving finally

$$H(s) = \frac{-K_v s}{s - K_v k_p} \alpha_{ss} - \frac{K_v s}{s - K_v k_p} H_{ref}(s) + \frac{K_F}{s - K_v k_p} F_{in}(s)$$

The closed-loop poles are therefore given by $s - K_v k_p = 0$ or $s = K_v k_p$. Since both K_v and k_p are both positive this leads to an unstable closed-loop pole. Clearly, if k_p were negative, then the closed-loop would be stable. This we can achieve by using a reverse acting controller and putting an additional gain of $[-1]$ at the output of the controller. In physical terms we are stating that when we have an increasing level over the desired operating level, we need to *open* the valve to provide

corrective action. In many other examples, for example a heating control system, an increasing temperature signal would require us to close a valve to reduce the heating effect.

4. Use the data to design a reverse acting PI controller to achieve closed-loop pole placement at $s = -0.7 \pm j0.7$.

For the desired PI design we alter Figure 18.33 by adding an integral term in the controller (Figure 18.34). This replaces the need for having the steady state offset α_{ss} . We use the following data:

$$K_V = \left(\frac{f_R}{A}\right) = \left(\frac{5}{6}\right) = 0.83 \quad \text{and} \quad K_F = \left(\frac{1}{A}\right) = \left(\frac{1}{6}\right) = 0.17$$

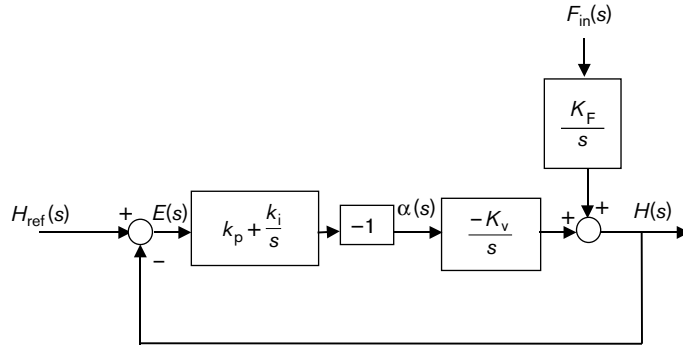


Figure 18.34 Reverse acting PI control system design framework.

The closed-loop expression is obtained from Figure 18.34 as follows:

$$H(s) = \frac{-0.83}{s} \alpha(s) + \frac{0.17}{s} F_{in}(s)$$

$$\alpha(s) = (-1) \left(k_p + \frac{k_i}{s} \right) (H_{ref}(s) - H(s))$$

Substituting, we have:

$$H(s) = \frac{-0.83}{s} (-1) \left(k_p + \frac{k_i}{s} \right) (H_{ref}(s) - H(s)) + \frac{0.17}{s} F_{in}(s)$$

$$H(s) = \frac{0.83(k_p s + k_i)}{s^2} (H_{ref}(s) - H(s)) + \frac{0.17}{s} F_{in}(s)$$

$$H(s) = \left[1 + \frac{0.83(k_p s + k_i)}{s^2} \right] = \frac{0.83(k_p s + k_i)}{s^2} H_{ref}(s) + \frac{0.17}{s} F_{in}(s)$$

giving finally

$$H(s) = \frac{0.83(k_p s + k_i)}{s^2 + 0.83k_p s + 0.83k_i} H_{ref}(s) + \frac{0.17s}{s^2 + 0.83k_p s + 0.83k_i} F_{in}(s)$$

The closed-loop poles are given by $\rho_{CL}(s) = s^2 + 0.83k_p s + 0.83k_i = 0$. The closed-loop pole design equation is derived using $s = -0.7 \pm j0.7$ as $p_{design}(s) = s^2 + 1.4s + 0.98$. Thus we have

$$\rho_{CL}(s) = s^2 + 0.83k_p s + 0.83k_i = p_{design}(s) = s^2 + 1.4s + 0.98$$

and the design equations $0.83k_p = 1.4$ and $0.83k_i = 0.98$ yield $k_p = 1.69$ and $k_i = 1.18$. The reverse acting PI control law which meets the pole placement specification is

$$\alpha(s) = (-1) \left(1.69 + \frac{1.18}{s} \right) (H_{\text{ref}}(s) - H(s))$$

5. Develop a Simulink simulation for assessing the reference tracking and disturbance rejection performance

The Simulink simulation to be developed will be used for assessing the PI controller strategy and will incorporate the saturation characteristic. The simulation is shown in Figure 18.35.

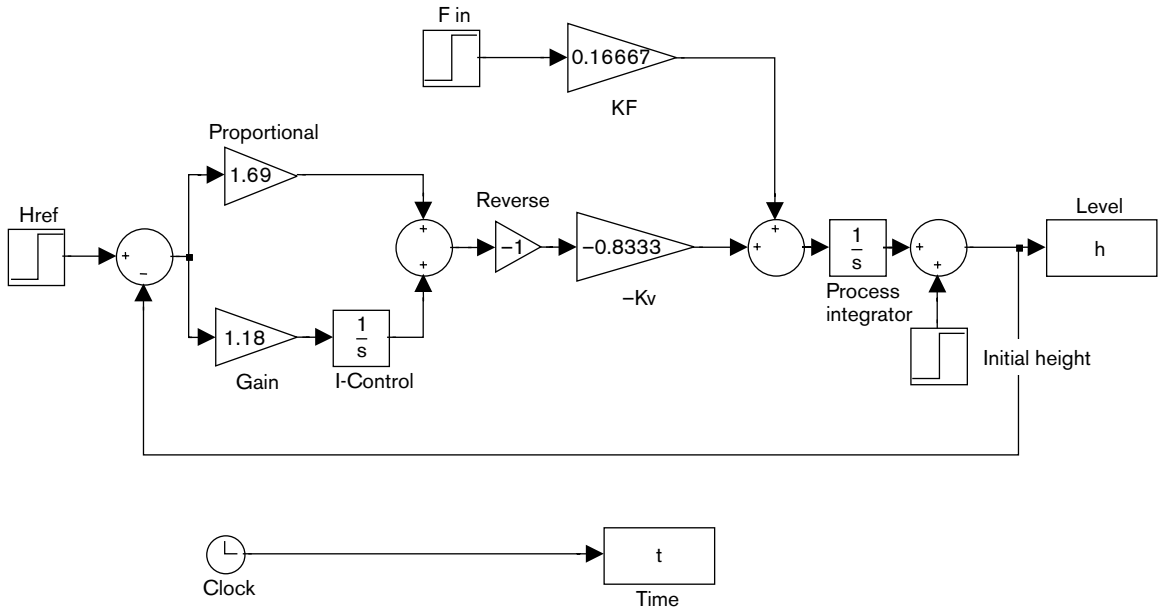


Figure 18.35 Simulink model for reverse acting controller.

There are some particular points that we should note:

- (a) We note that we have combined the integral terms in the level model equation:

$$H(s) = \frac{-0.83}{s} \alpha(s) + \frac{0.17}{s} F_{\text{in}}(s) + \frac{h_0}{s}$$

becomes

$$H(s) = \frac{-0.83\alpha(s) + 0.17F_{\text{in}}(s)}{s} + \frac{h_0}{s}$$

The first equation uses two integrators and relates the level to the integral of the difference in volume of the inflow and the outflow. In the second equation the level, $h(t)$, is given by the integral of the difference between the input and output flows. If the input flow equals the output flow, then there is no change in level.

- (b) Since we are starting at the operating level of 4 m, we must initialise the simulation. To do this we set initial values for the integrators. We are effectively initialising the *state variables* within the system, and we find out more about this in Chapters 20–23.

We provide the following initial conditions:

- (i) $H_{\text{ref}}(s)$ is modelled by a step input with value of 4.
- (ii) The initial condition of the integrator in the integral control is set to -0.4 . (Hence, if the error is zero (as in steady state), the output of the controller will have a contribution of zero from the proportional term and will be given by the current integrator value $= -0.4$.)
- (iii) The initial condition of the integrator within the process is zero, since this relates to the integration of the (input–output) signal, which in steady state will be zero.
- (iv) The value for F_{in} is given as a step at time 0 with value $2 \text{ m}^3/\text{min}$, which is the nominal inflow for a level of 4 m.
- (v) The initial condition for the height is given by h_0/s , which can be represented by a step input signal of height 4 m.

These initial conditions will give rise to a system in steady state; the output level, $h(t)$, will be 4 m and the error signal will be zero, ensuring that the system remains at the operating point.

- (c) We also set up the Simulink simulation to do positive and negative step reference tracking and disturbance performance tests. We do these plus–minus tests whenever we are investigating nonlinear effects in a process description. The saturation characteristic is a nonlinear system component, so the plus–minus tests must be performed.

The plots for a $\pm 10\%$ level reference change (change of $\pm 0.4 \text{ m}$) and a $+10\%$ input flow disturbance (change of $+0.2 \text{ m}^3/\text{min}$) are shown in Figures 18.36(a) and (b) respectively. The plots exhibit the properties we should now expect from PI control. We see that the new reference level of 4.4 m is attained by the control law with no steady state offset, and in the second plot the constant flow disturbance is completely rejected in steady state.

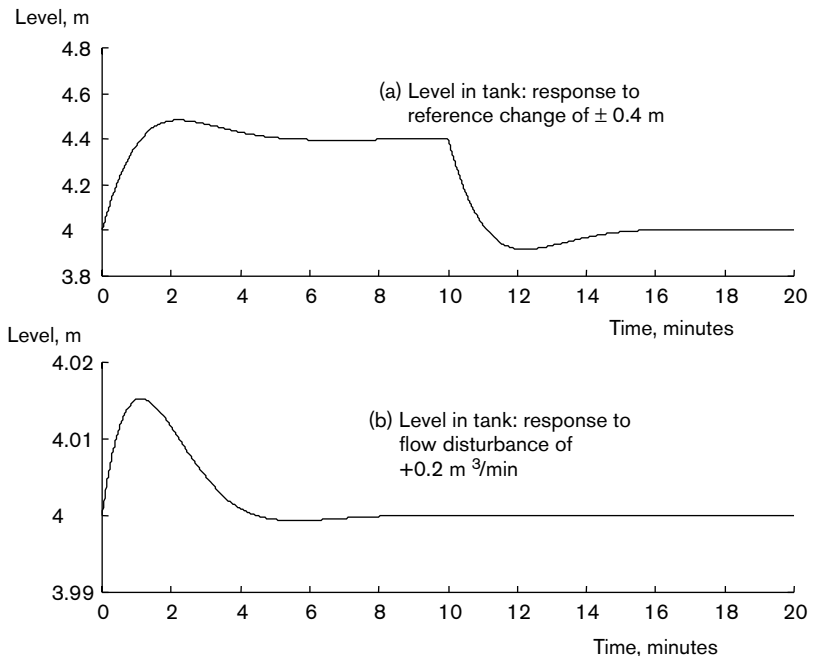


Figure 18.36 Reverse acting PI controller performance plots.

When testing the full controller design for implementation, the valve saturation should be included in the simulation, but the various sets of tests should include some purely linear simulations omitting the saturation, while others should include the saturation. In this way a thorough investigation of the effect of the valve *nonlinearity* can be made. Due to space considerations, we did not present all the necessary tests here, but we should realise that for many real industrial control designs extensive simulations will be performed.

In this section we found that a reverse acting controller is a very simple extension of a controller to accommodate a negative system gain. We need this because some physical systems used in industry have this property. We also saw how not to confuse this with inverse systems. In the case study, where we designed a simple reverse acting PI controller, we also met some of the typical features of industrial control design.

18.9 Digital PID control

Our world is one of continuous variable types. Quantities like flow, temperature and voltage are not discrete signals or variables but continuous ones. However, we find that industrial controllers and SCADA systems will use digital computers and manipulate sampled data values. The control formula implemented will be a digital algorithm for the controller. So, although we find it easier to understand and analyse PID control using the continuous time and Laplace s -domain, real-world implementations will use digital recurrent relationships of sampled data values and the analysis tools of the z -domain. For an in-depth treatment of these topics we would need another book. Instead, we concentrate on one very important practical aspect. Earlier in the chapter we looked at the two forms of industrial implementation of PID control: the process controller unit and the SCADA system PID tuning interface. When faced with these types of device it is very important to know the exact form of the PID controller used. We saw how it was possible to have a decoupled textbook form with coefficients K_p , K_i , K_d or one of the many industrial forms using the PID parameters K_p , τ_i , τ_d . In this section we come to appreciate that it is necessary to further check whether the PID controller is using a continuous time form or a sampled data form. As we will find out, the digital computer form uses a recurrent relationship of sampled data values with new coefficients for the P, I and D effects. We cannot stress too strongly the need to find out what form is being used by the software, for if the control engineer inserts the wrong tuning values in the wrong PID controller formula the results could be very catastrophic. We hope that a brief introduction to the context of digital PID control will prevent this from happening!

18.9.1 Sampling and digital control

Digital control is really sampled data manipulation using a new formula for the PID controller. We often use the technical terms *algorithm* and *recurrent relationship* to describe this formula, but we should also appreciate the general framework for this digital implementation. We find that the process or system is interfaced from the continuous time domain to the digital computing domain by a suitable analogue to digital (A/D) interface device. Similarly, once the computing domain has produced a new controller output value using the controller recurrent relationship, this has to be fed to the continuous-time domain process and actuators by a suitable digital-to-analogue (D/A) interface device. The outcome of this is a digital control framework as shown in Figure 18.37.

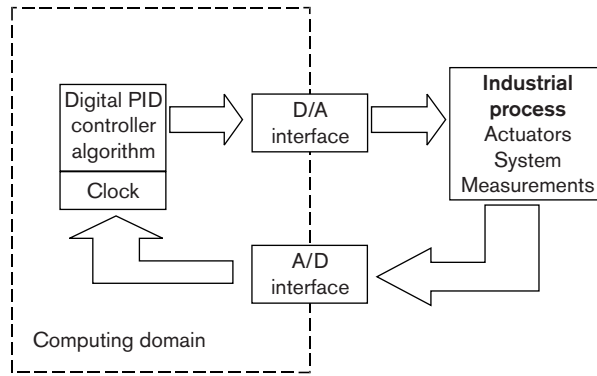


Figure 18.37 Digital control framework.

The PID control algorithm is usually a fairly simple recurrent relationship that uses sampled measurements from the A/D device and outputs a discrete sequence of control command signals. At the A/D interface side there will be some signal processing and filtering of the analogue signal before it is sampled and used by the digital control algorithm, while at the D/A interface the discrete sequence of control values has to be turned into an analogue signal to drive the process actuators. If we were to look at the PID algorithm at the coding level, we should find that the lines of code for PID control loop are surprisingly innocent. But we should remember that the dynamic performance of ship autopilots, boilers, power stations, aircraft and steel mills will depend on our skill in producing a correct and accurate PID control design.

To ensure that we understand the structure of sampling and reconstruction involved in the digital control framework, we use Figure 18.38. Here we can see that digital control involves passing from the continuous time domain to the sampled data domain and then back to the continuous time domain of the industrial system. We note that digital filters can cause *aliasing* to occur; effectively, the sampling in the digital process can alter the frequency of some signals so that high-frequency signals could appear as low-frequency signals within the controller bandwidth. An *analogue* anti-alias filter must be placed before the digital sampling to prevent aliasing occurring. We do not cover this further here, but provide this as a warning to students implementing digital control systems.

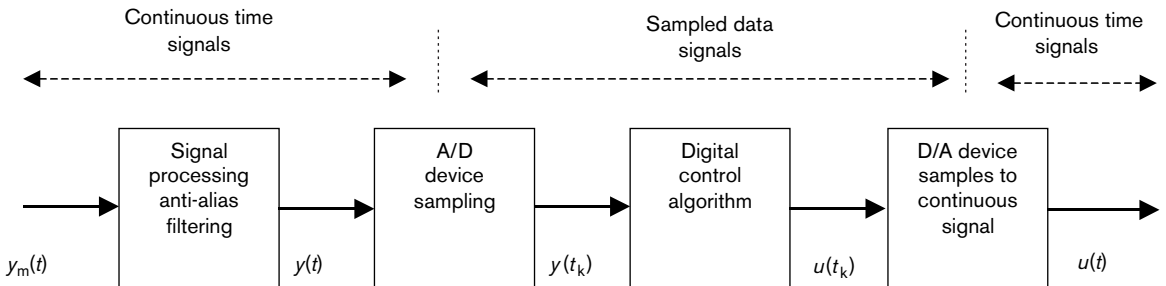


Figure 18.38 The sampling process.

Skill section

Sampling notation

To perform analysis with sampled data signals, we need to introduce some notation. A digital or sampled data signal is a continuous time signal that has been sampled, usually at a fixed interval or fixed time period, T . This period of time, T , is known as the sample interval. This basic operation of sampling creates different types of sample data notation.

Sampling: Set $t_k = kT$, for $k = 0, 1, 2, \dots$. Then the sampled signal is defined by $y(t_k) = y(kT)$.

Example: For $T = 5$ seconds:

$$t_0, t_1, t_2, \dots = t(0T), t(1T), t(2T), \dots = 0, 5, 10, \dots \text{ seconds}$$

$$y(t_0), y(t_1), y(t_2), \dots = y(0), y(5), y(10), \dots$$

Shorthand notation for sampled data

We find that this $y(t_k), y(kT)$ notation is cumbersome, so we use a shortened notation. We assume that the sample period is fixed at T , then at $t_k = kT$ we write $y_k = y(t_k)$, $k = 0, 1, 2, \dots$

Example: $y(t_0), y(t_1), y(t_2), \dots = y_0, y_1, y_2, \dots = y(0), y(5), y(10), \dots$

We can see that we have used a simple subscript notation to indicate the time indexing of the sampling of the continuous time domain signal. We can use this in more advanced ways, for example:

$$\text{if } t_{k+1} = (k+1)T \text{ we can write } y_{k+1} = y(t_{k+1})$$

$$\text{if } t_{k+10} = (k+10)T \text{ we can write } y_{k+10} = y(t_{k+10})$$

$$\text{if } t_{k-10} = (k-10)T \text{ we can write } y_{k-10} = y(t_{k-10})$$

Examples

If $T = 0.25$ s and $t = 0$, then $t = 0T$, giving $k = 0$. We write $y(t_0) = y_0$

If $T = 0.25$ s and $t = 10$, then $t = 40T$, giving $k = 40$. We write $y(t_{40}) = y_{40}$

If $T = 0.05$ s and $t = -10$, then $t = -200T$, giving $k = -200$. We write $y(t_{-200}) = y_{-200}$

18.9.2 A PID digital control algorithm

We are going to write new recurrent relationships for PID control based on using the time domain PID controller form with sampled data signals. The principle is simply to replace the terms of the PID controller algorithm by numerical approximations. As you may already know from studies with numerical methods, it is possible to have different numerical approximations for numerical integration and differentiation, such as the mid-point integration rule, the trapezoidal integration rule, or even Simpson's integration rule. In PID control, different numerical formula will lead to different digital PID algorithms. To avoid developing many different formulas and simply to illustrate the main principles, we are going to use *backward* forms for numerical differentiation and integration. The backward form for differentiation is

$$\left. \frac{de}{dt} \right|_k = \frac{e_k - e_{k-1}}{T}$$

The form for integration can be seen from Figure 18.39, where we approximate the integral of the error between time t_{k-1} and time t_k by the shaded area:

$$\int_{t_{k-1}}^{t_k} e(\tau) d\tau = e(t_k) \times T$$

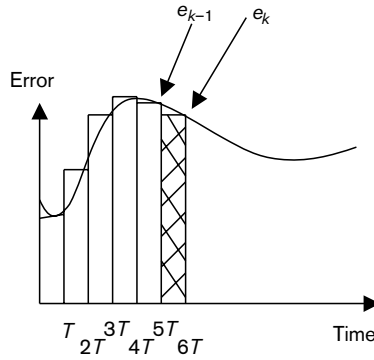


Figure 18.39 Integration approximation.

We base the development on the industrial parametrisation of the PID control law that includes a modified derivative term. In Laplace transforms, we use

$$U(s) = K_p \left(1 + \frac{1}{\tau_i s} + \frac{\tau_d s}{\tau_f s + 1} \right) E(s) = \left(K_p + \frac{K_p}{\tau_i s} + \frac{K_p \tau_d s}{\tau_f s + 1} \right) E(s)$$

We start by separating out the three terms of the PID controller and writing

$$\begin{aligned} U(s) &= K_p E(s) + \left(\frac{K_p}{\tau_i s} \right) E(s) + \left(\frac{K_p \tau_d s}{\tau_f s + 1} \right) E(s) \\ &= U_p(s) + U_i(s) + U_d(s) \end{aligned}$$

From this total expression for the PID controller, we will work with the time domain form:

$$u(t) = u_p(t) + u_i(t) + u_d(t)$$

When we sample this total PID control signal, we get

$$u(t_k) = u_p(t_k) + u_i(t_k) + u_d(t_k)$$

or in shortened form:

$$u_k = u_{pk} + u_{ik} + u_{dk}$$

What we have to find in the derivation of a sampled PID form are recurrent formulas for each of the three terms u_{pk} , u_{ik} , u_{dk} . We do this by examining the three terms in turn.

Proportional term

The Laplace transform for the proportional term is given from the above as

$$U_p(s) = K_p E(s)$$

The time domain form is then

$$u_p(t) = K_p e(t)$$

Sampling at $t = t_k$ gives

$$u_p(t_k) = K_p e(t_k)$$

or using the shortened notation:

$$u_{pk} = K_p e_k$$

Integral term

The Laplace transform for the integral term is given as

$$U_i(s) = \left(\frac{K_p}{\tau_i s} \right) E(s)$$

The time domain form is then

$$u_i(t) = \left(\frac{K_p}{\tau_i} \right) \int_{t_0}^t e(\tau) d\tau$$

and we note that we can also write

$$u_i(t_{k-1}) = \left(\frac{K_p}{\tau_i} \right) \int_{t_0}^{t_{k-1}} e(\tau) d\tau$$

Sampling at $t = t_k$ gives

$$u_i(t_k) = \left(\frac{K_p}{\tau_i} \right) \int_{t_0}^{t_k} e(\tau) d\tau = \left(\frac{K_p}{\tau_i} \right) \int_{t_0}^{t_{k-1}} e(\tau) d\tau + \left(\frac{K_p}{\tau_i} \right) \int_{t_{k-1}}^{t_k} e(\tau) d\tau$$

We identify the first term in this expression as the term $u_i(t_{k-1})$ mentioned above. The second term is approximated using our integral approximation

$$\left(\frac{K_p}{\tau_i} \right) \int_{t_{k-1}}^{t_k} e(\tau) d\tau = \left(\frac{K_p}{\tau_i} \right) (e(t_k) \times T)$$

giving

$$u_i(t_k) = u_i(t_{k-1}) + \left(\frac{K_p}{\tau_i} \right) T e(t_k)$$

If we use the shortened notation, we obtain

$$u_{ik} = u_{ik-1} + \left(\frac{K_p T}{\tau_i} \right) e_k$$

Derivative term

The Laplace transform for the derivative term is given as

$$U_d(s) = \left(\frac{K_p \tau_d s}{\tau_f s + 1} \right) E(s)$$

This may be written as

$$(\tau_f s + 1) U_d(s) = (K_p \tau_d s) E(s)$$

The time domain form is then

$$\tau_f \frac{du_d}{dt} + u_d(t) = (K_p \tau_d) \frac{de}{dt}$$

Sampling at $t = t_k$ gives

$$\tau_f \left. \frac{du_d}{dt} \right|_{t=t_k} + u_d(t_k) = (K_p \tau_d) \left. \frac{de}{dt} \right|_{t=t_k}$$

We use the usual numerical approximation for the differentiation:

$$\left. \frac{de}{dt} \right|_k = \frac{e_k - e_{k-1}}{T}$$

to obtain

$$\tau_f \left(\frac{u_d(t_k) - u_d(t_{k-1})}{T} \right) + u_d(t_k) = (K_p \tau_d) \left(\frac{e(t_k) - e(t_{k-1})}{T} \right)$$

and we rearrange this to give

$$u_d(t_k) = \left(\frac{\tau_f}{\tau_f + T} \right) u_d(t_{k-1}) + \left(\frac{K_p \tau_d}{\tau_f + T} \right) (e(t_k) - e(t_{k-1}))$$

Using the shortened notation, we finally have

$$u_{dk} = \left(\frac{\tau_f}{\tau_f + T} \right) u_{dk-1} + \left(\frac{K_p \tau_d}{\tau_f + T} \right) (e_k - e_{k-1})$$

Thus we have derived three recurrent relationships for the individual proportional, integral and derivative terms in a digital PID control algorithm. We use these results in the composite sampled form, $u_k = u_{pk} + u_{ik} + u_{dk}$, depending on the particular structure of the PID control law being used. We summarise the recurrent relationships in Table 18.7.

Table 18.7 Recurrent relationships for a digital PID control algorithm.

PID controller	$U(s) = K_p \left[1 + \frac{1}{\tau_i s} + \left(\frac{\tau_d}{\tau_f s + 1} \right) \right] E(s)$
$U(s) = U_p(s) + U_i(s) + U_d(s)$	$u_k = u_{pk} + u_{ik} + u_{dk}$
$u(t) = u_p(t) + u_i(t) + u_d(t)$	
Proportional term	
$U_p(s) = K_p E(s)$	$u_{pk} = K_p e_k$
Integral term	
$U_i(s) = \left(\frac{K_p}{\tau_i s} \right) E(s)$	$u_{ik} = u_{ik-1} + \left(\frac{K_p T}{\tau_i} \right) e_k$
Derivative term	
$U_d(s) = \left(\frac{K_p \tau_d s}{\tau_f s + 1} \right) E(s)$	$u_{dk} = \left(\frac{\tau_f}{\tau_f + T} \right) u_{dk-1} + \left(\frac{K_p \tau_d}{\tau_f + T} \right) (e_k - e_{k-1})$

Problem The PID control design for a liquid level system arrives at a solution that requires a reverse acting controller containing only the P and I terms. The controller, which is designed in the s-domain and uses a decoupled form, is given by

$$U(s) = (-1) \left[1.69 + \left(\frac{1.18}{s} \right) \right] E(s)$$

If the sampling process uses a sample period of $T = 0.1$ min, use Table 18.7 to devise an algorithm for the PI controller calculation.

Solution The controller design is given by

$$U(s) = (-1) \left[1.69 + \left(\frac{1.18}{s} \right) \right] E(s)$$

and uses only the P and I terms; thus the full PID control expression reduces to

$$U(s) = -K_p \left(1 + \frac{1}{\tau_i s} \right) E(s) = - \left(K_p + \frac{K_p}{\tau_i s} \right) E(s)$$

We can propose a digital algorithm of the form

$$u_k = [-1](u_{pk} + u_{ik})$$

where we retain the reverse acting controller structure. We identify the parameter equations

$$K_p = 1.69 \quad \text{and} \quad \frac{K_p}{\tau_i} = 1.18; \quad \text{we will also need } T = 0.1$$

From the table, we can construct the digital PI control recurrent relationships. We set

$$u_{pk} = K_p e_k \quad \text{and} \quad u_{ik} = u_{ik-1} + \left(\frac{K_p T}{\tau_i} \right) e_k$$

Using the data of the problem we have

$$u_{pk} = 1.69 e_k, \quad u_{ik} = u_{ik-1} + 0.118 e_k$$

and a suitable controller algorithm may be given as:

Initial step

Set $u_{i-1} = 0$; $k = 0$

Loop

Get e_k

Compute $u_{pk} = 1.69 e_k$; $u_{ik} = u_{ik-1} + 0.118 e_k$

Compute PI control signal, $u_k = [-1](u_{pk} + u_{ik})$

Store u_{ik} to give u_{ik-1} next iteration

Update counter, $k = k + 1$

Repeat loop step

The PID recurrent relationships devised in this section use the actual values of error signals, e_k , e_{k-1} and control signals, u_{ik} , u_{ik-1} and the output of the PID algorithm is the actual controller signal needed, namely u_k . This type of digital PID controller algorithm is known as a positional PID control algorithm. By way of contrast, in the next section we shall consider *velocity* or *incremental* digital PID algorithms that are sometimes used instead.

18.9.3 Velocity or incremental digital PID control algorithms

In some industrial applications, particularly stepper motor control, PID control is used in an incremental form. For this we use

$$\begin{aligned}\Delta u_k &= u_k - u_{k-1} \\ &= u_{pk} + u_{ik} + u_{dk} - (u_{pk-1} + u_{ik-1} + u_{dk-1}) \\ &= (u_{pk} - u_{pk-1}) + (u_{ik} - u_{ik-1}) + (u_{dk} - u_{dk-1}) \\ &= \Delta u_{pk} + \Delta u_{ik} + \Delta u_{dk}\end{aligned}$$

To complete the analysis, we use Table 18.7 to obtain three incremental PID terms as follows.

Proportional increment

$$\Delta u_{pk} = u_{pk} - u_{pk-1} = K_p(e_k - e_{k-1}) = K_p \Delta e_k$$

Integral increment

$$\Delta u_{ik} = u_{ik} - u_{ik-1} = \left[u_{ik-1} + \left(\frac{K_p T}{\tau_i} \right) e_k \right] - \left[u_{ik-2} + \left(\frac{K_p T}{\tau_i} \right) e_{k-1} \right]$$

Hence, writing this in terms of integral and error increments,

$$\Delta u_{ik} = (u_{ik-1} - u_{ik-2}) + \left(\frac{K_p T}{\tau_i} \right) (e_k - e_{k-1}) = \Delta u_{ik-1} + \left(\frac{K_p T}{\tau_i} \right) \Delta e_k$$

Derivative increment

$$\begin{aligned}\Delta u_{dk} &= u_{dk} - u_{dk-1} \\ &= \left[\left(\frac{\tau_f}{\tau_f + T} \right) u_{dk} + \left(\frac{K_p \tau_d}{\tau_f + T} \right) (e_k - e_{k-1}) \right] - \left[\left(\frac{\tau_f}{\tau_f + T} \right) u_{dk-1} + \left(\frac{K_p \tau_d}{\tau_f + T} \right) (e_{k-1} - e_{k-2}) \right] \\ &= \left(\frac{\tau_f}{\tau_f + T} \right) (u_{dk-1} - u_{dk-2}) + \left(\frac{K_p \tau_d}{\tau_f + T} \right) [(e_k - e_{k-1}) - (e_{k-1} - e_{k-2})] \\ &= \left(\frac{\tau_f}{\tau_f + T} \right) (u_{dk-1} - u_{dk-2}) + \left(\frac{K_p \tau_d}{\tau_f + T} \right) (e_k - 2e_{k-1} + e_{k-2})\end{aligned}$$

or

$$\Delta u_{dk} = \left(\frac{\tau_f}{\tau_f + T} \right) \Delta u_{dk-1} + \left(\frac{K_p \tau_d}{\tau_f + T} \right) (e_k - 2e_{k-1} + e_{k-2})$$

This equation is a very common implementation of the derivative term in a PID controller and can also be written as

$$\Delta u_{dk} = \left(\frac{\tau_f}{\tau_f + T} \right) \Delta u_{dk-1} + \frac{K_p \tau_d}{\tau_f + T} (\Delta e_k - \Delta e_{k-1})$$

As with the positional algorithms, we can construct a table of PID control velocity increment terms, as shown in Table 18.8. These can be combined to form different structures for incremental digital PID algorithms. Furthermore this set of recurrent formulas can be used in two ways: (a) the control increment can be applied directly as in stepper motor control, or (b) the increments can be used in place of a positional algorithm with

Table 18.8 Increment relationships for digital velocity PID control algorithms.

PID controller	$U(s) = K_p \left[1 + \frac{1}{\tau_i s} + \left(\frac{\tau_d}{\tau_f s + 1} \right) \right] E(s)$
$u_k = u_{k-1} + \Delta u_k$, with $\Delta u_k = \Delta u_{pk} + \Delta u_{ik} + \Delta u_{dk}$	
Proportional increment	$\Delta u_{pk} = K_p \Delta e_k$
Integral increment	$\Delta u_{ik} = \Delta u_{ik-1} + \left(\frac{K_p T}{\tau_i} \right) \Delta e_k$
Derivative increment	$\Delta u_{dk} = \left(\frac{\tau_f}{\tau_f + T} \right) \Delta u_{dk-1} + \left(\frac{K_p \tau_d}{\tau_f + T} \right) (e_k - 2e_{k-1} + e_{k-2})$ $= \left(\frac{\tau_f}{\tau_f + T} \right) \Delta u_{dk-1} + \left(\frac{K_p \tau_d}{\tau_f + T} \right) (\Delta e_k - \Delta e_{k-1})$

the control update equation, $u_k = u_{k-1} + \Delta u_k$. As with many other aspects of industrial PID control, there exists a veritable storehouse of knowledge residing in industrial manuals, which has been accumulated over many years of industrial applications of these methods. We should always be prepared to learn from this experience and find out as much as possible about any particular problem or application area we might be faced with.

As a last word of warning for this practical chapter, sampling of systems should be done carefully: numerical approximations do not hold if the sample time T is large in comparison with the time constants within the process; a general guide is that the sampling frequency should be greater than 15 times the closed-loop natural frequency of the system, else the performance of the digitally controlled system will not achieve the required specifications.

What we have learnt

- ✓ To recognise how PID control is found in industrial control applications either as a process control unit or in a SCADA system control-tuning interface.
- ✓ To convert between textbook and industrial forms for PID control involving K_p , K_i , K_d and K_p , τ_i , τ_d notation.
- ✓ To understand the problems and the solution for proportional and derivative kick in PID control.
- ✓ To understand the problems caused by integral wind-up and measurement noise in industrial PID control.
- ✓ To construct anti-windup circuits and introduce a modified differential term to mitigate measurement noise effects.
- ✓ To recognise and be able to interpret different industrial PID controller structures including parallel, and series PID controllers.

- ✓ To identify the need for a reverse acting controller and how not to confuse this with inverse response processes.
- ✓ To understand the context of digital PID control and how to construct simple digital PID algorithms.

Multiple choice

M18.1 The parallel presentation of PID controllers is described by:

- (a) $u(t) = K_p e(t) + K_i \int_0^t e(\tau) d\tau + K_d \frac{de}{dt}$
- (b) $u(t) = K_i e(t) + K_p \int_0^t e(\tau) d\tau + K_d \frac{de}{dt}$
- (c) $u(t) = K_d e(t) + K_i e(t) + K_d \frac{de}{dt}$
- (d) $u(t) = K_p e(t) + K_i \int_0^\infty e(\tau) d\tau + K_d \frac{de}{dt}$

M18.2 A PID controller has a proportional band of 50%; the proportional gain is:

- (a) $K_p = 50$
- (b) $K_p = PB/50$
- (c) $K_p = 50PB$
- (d) $K_p = 100/PB$

M18.3 A process is controlled by a PID controller. The sensor has high measurement noise. To reduce the effect of noise, we:

- (a) use *anti-wind-up circuits* to mitigate the wind-up effect
- (b) use a *bandwidth-limited derivative* term to prevent measurement noise amplification
- (c) use proportional and derivative terms in the forward path
- (d) move the proportional and derivative terms to different positions in the controller

M18.4 Anti-integral wind-up removes the effect of:

- (a) input saturation
- (b) saturation due to large process disturbances
- (c) derivative kick
- (d) (a) and (b)

M18.5 To implement the derivative term, we usually use a low-pass filter. The time constant of a low-pass filter should be:

- (a) much smaller than the derivative time constant
- (b) much smaller than the integral time constant
- (c) much smaller than the system time constant
- (d) much larger than the derivative time constant

M18.6 We may observe derivative kick in systems where:

- (a) the integral term operates on the error
- (b) the derivative term operates on the output
- (c) the integral term operates on the output
- (d) the derivative term operates on the error

M18.7 A PID controller is represented by the transfer function:

$$K(s) = \left[K_s \left(1 + \frac{1}{T_i s} \right) (1 + T_d s) \right]$$

The three terms are connected in:

- (a) parallel
- (b) random
- (c) series
- (d) parallel and series

M18.8 A reverse acting controller response is needed when the system has:

- (a) controller poles on the RHP
- (b) system poles on the RHP
- (c) system zeros on the RHP
- (d) a negative system gain

M18.9 Which statement is correct?

- (a) steady state error can be reduced by increasing integral gain
- (b) overshoot can be reduced by increasing derivative gain
- (c) steady state error can be increased by increasing integral gain
- (d) (a) and (b)

M18.10 A temperature control system uses only the integral term of a PID controller such that the controller is $K(s) = 1/s$. To implement the integral controller on a computer with an update rate of 1 second, we use the equivalent discrete form:

- (a) $u_t = u_{t-1} + e_t$
- (b) $u_t = u_{t-1} - e_t$
- (c) $u_t = e_t$
- (d) $u_t = -e_t$

Questions: practical skills

Q18.1 What are the full expressions for the textbook decoupled and the industrial gain–time constant forms for the PID controller?

- (a) Define the precise relationship between the coefficients of the decoupled textbook form and the interactive industrial form for a PID controller.
 (b) Use the formulas to determine if the following PID controllers are identical:

$$G_P(s) = 4.56 \left[1 + \frac{1}{2.35s} + 1.60s \right] \text{ and } G_Q(s) = \left[4.56 + \frac{1.94}{s} + 6.92s \right]$$

Q18.2 A steel rolling stand has a hydraulic loop which is considered to be a high gain feedback loop. A technician is trying to decide whether this is the case. The data from the technical manual is given as the following table. What is the proportional controller gain being used in this hydraulic loop?

Controller input variable, $e(t)$	Range value: $e_R = 12 \text{ V}$
Controller output variable, $u(t)$	Range value: $u_R = 5000 \text{ N}$
Controller proportional band, PB%	2

Q18.3 PID controllers are widespread in industrial power generation plant applications and their associated engineering issues are very well developed, and accepted. Sometimes special structures are given for particular loops. One industrial SCADA manual gave a PID formula as:

$$\text{output} = K_c \left[-PV + \frac{1}{T_i s} E - \left(\frac{T_d s}{a T_d s + 1} \right) PV \right]$$

with $0.1 \leq a \leq 0.25$ and $E = SP - PV$

- (a) What do the variables PV , E , and SP mean?
 (b) Draw a block diagram for this controller.
 (c) What is the purpose of the term

$$\left(\frac{T_d s}{a T_d s + 1} \right)$$

in the controller?

Q18.4 Interacting PID controller forms can be difficult to visualise and understand. One obvious solution is to resolve the specific controller formula into one of the standard forms.

- (a) Write the given interacting PID controller in standard decoupled textbook form.

$$U(s) = \left[4.8 \left(1 + \frac{1}{10.56s} \right) (1 + 0.56s) \right] E(s)$$

- (b) Write the given interacting PID controller in industrial time constant–gain form.

$$U(s) = \left[11.23 \left(1 + \frac{1}{8.39s} \right) (1 + 1.54s) \right] E(s)$$

Problems

P18.1 Industrial process controller hardware units are continually evolving. For example, changes are continually being made to the operator interface technology, to the hardware technology for control

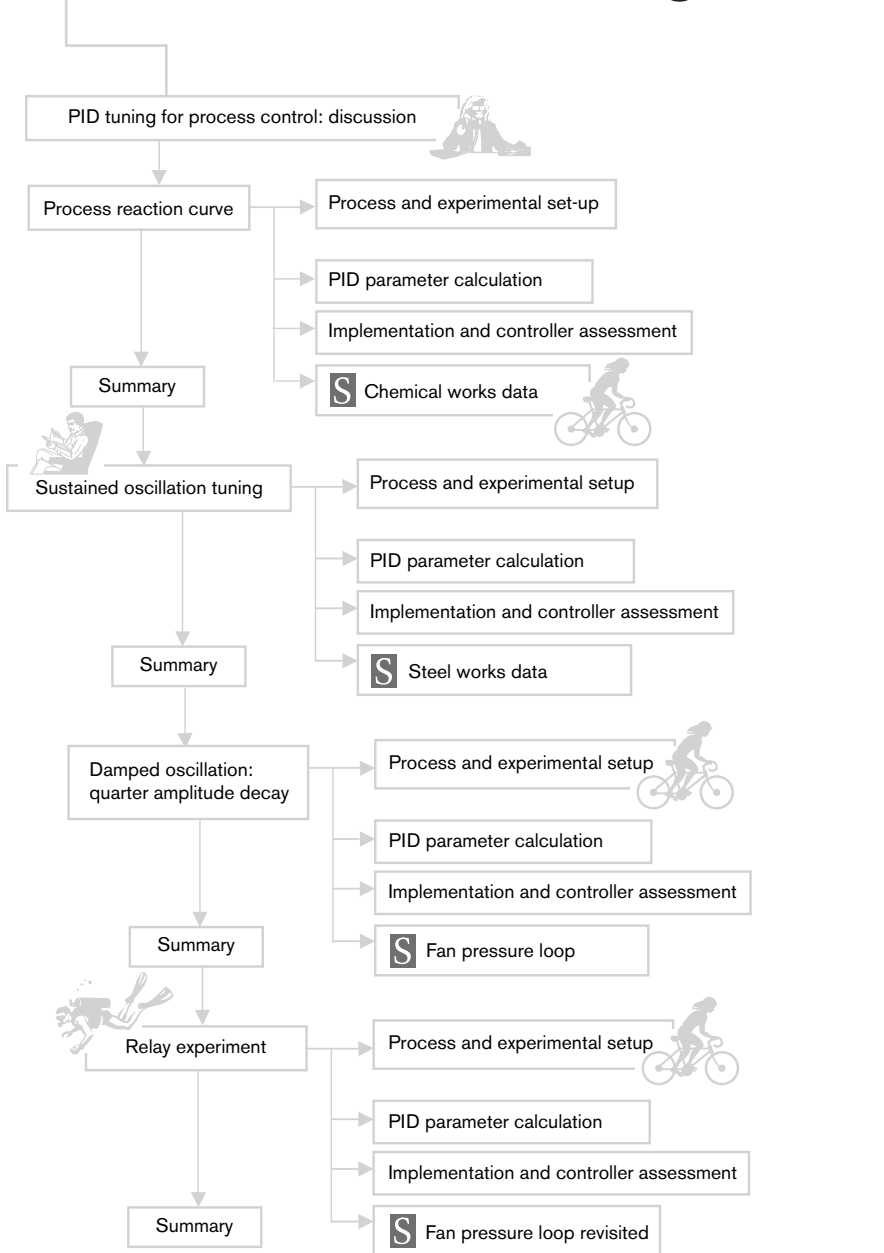
method implementation, to the range of features that really aid the end-user, and to those features that are considered to enhance marketability. Use Internet searches and industrial magazines to compile an up-to-date survey of current products, the technology that the devices use, and market trends. The above headings should give an indication of the issues to consider in the survey.

P18.2 The marketability of industrial process controller hardware units is not completely dependent on the type of control methods used by the device. However, the control success will depend on the control method used and the range of control features offered. Use Internet searches and industrial magazines to compile an up-to-date survey of the control methods used by current controller hardware units in the market-place.

P18.3 Proportional, Integral and derivative controller units are widespread in process and industrial control. The current technology is based on microprocessor devices with digital interfaces being common. Many extra features like manual tuning, autotune and multiple loops are offered. This project looks at the industrial take-up of these advanced features by practising engineers. What do engineers think of PID control? What do engineers find difficult about PID control? A careful questionnaire should be constructed, and visits to a local company arranged to obtain first-hand experience of the equipment involved. An assessment of the successes and problems of current PID controller technology should be constructed.

P18.4 SCADA system technology is much more difficult to categorise and study. Yet SCADA system technology keeps much of industry coordinated, operational and efficient. Find a local company, industry or process plant and arrange to visit the plant control engineer to learn about the structure, function and operation of a SCADA system. Try to talk to system operators to learn something of the nature of the machine–person interfaces used. Look for general features, like scale and level of complexity. Use the Internet to discover the names of the main SCADA system installers and collect company literature. Arrange the information by useful and intelligent headings. An assessment of the successes and problems of current SCADA system technology should be made and written as a report or dissertation.

19 PID controller tuning methods



In the process industries, systems may be classified according to the type and level of hardware and software technology used. Typically, we find that a large-scale installation can be broken down into a number of smaller process units. For example, in a hot strip mill in the steel industry we have furnaces, followed by conveyors with heat shields with several types of rolling stands along the process path and finishing with a coiling unit. Similarly, in a brewery, we start from units concerned with preparing the *mash* and *wort* from which beer is made, through to the fermentation vats, followed by the bottling plant. We rank these smaller system units and the level of associated control that they require into three levels: simple, intermediate and complex process units.

Controllers for these three classes of process units will depend on the level of complexity of the system and its operation. In a manufacturing plant we might use a conveyor belt system to transport goods from one part of the factory to another, and for this, simple ON-OFF controllers will do. At the other extreme, complex systems with highly connected physical and chemical processes and demanding performance objectives will need carefully designed controllers. Such controllers will often depend on the loop structure as much as the individual controller transfer function design for achieving successful control. Good examples are a multi-stand steel rolling mill or the high-speed bottling unit at the end of the brewing process. Clearly a simple ON-OFF controller is not going to work here since there has to be coordination between 'conveyor belt' speed and the temperature of the steel product; in fact, we are going to need quite a clever control system.

The much more numerous intermediate class of systems will have fairly simple dynamics and single loop control structures. Sometimes the control loops may be cascaded, but they are rarely more complicated than this. For these processes three-term control or PID control is usually adequate. But, we have a different type of problem with this class of systems which we illustrate with a local example. Here in Scotland we have one of the largest Vitamin C manufacturing plants in Europe, and the plant is getting bigger. When we talked to the local engineers about this development, we were told that there will be around 1500 new intermediate technology loops to be commissioned, tuned up and then maintained. In another example we read a report that a typical Canadian paper mill has about 2000 control loops, and that around 98% of the loops will be for intermediate technological systems. This means some 1960 PID control loops to commission, tune and maintain!

Tuning over 1500 PID loops in a Vitamin C plant or over 1960 PID loops in single paper mill will be costly in terms of personnel time and possible lost production time and output during the test stages. We also have the problem that it is necessary to go back and check the tuning of the loops from time to time. Good plant management would have a rigorous timetable of controller tuning and instrumentation inspection to keep the plant availability at high levels. We can easily see that the cost of this will rise if highly qualified technical personnel have to be used to complete this task. This situation of tuning a large number of simple control loops is quite common in the process and primary industrial sectors. We usually find that the response of technical engineering management to this is to establish common plant standards based on one or two reliable simple PID tuning procedures. There is usually a company manual available which sets out the routines to be followed.

Manufacturers of controllers have taken this one step further, since in most process controllers on the market today the whole PID tuning procedure has been partially automated. Commercial controller units have an *autotune* button which allows PID controller tuning to be accomplished at the touch of a button by technician staff. Some of the procedures automated in today's commercial PID process controllers were first proposed in the 1940s by J. G. Ziegler and N. B. Nichols. Of course, the actual implemented routines have changed to allow the use of digital technology, but the basic principle of an online plant experiment followed by a PID calculation remains unchanged. In this chapter the basic principles for the following routines are described:

- Process reaction curve
- Sustained oscillation PID tuning
- Damped oscillation PID tuning (quarter amplitude decay)
- The relay method

Since these procedures have been automated in commercial equipment we might wonder why it is useful to learn about them! Unfortunately, these PID tuning procedures have limitations and it is as important to know when *not* to use a method as it is to know *how* to use a method. An inappropriate application of one of these techniques easily leads to very poor loop control.

Learning objectives

- To understand the basic structure of rule-based PID controller methods.
- To be able to use the process reaction curve method for PID tuning.
- To understand the family of PID tuning routines based on the principle of a sustained oscillation.
- To appreciate the practical and theoretical limitations inherent in this group of widely applied PID tuning methods.

19.1 Understanding PID tuning procedures

We have seen previously how PID controllers devised to solve particular engineering problems, or constructed from hardware components, have led to a number of different formulas for industrial PID laws. In this chapter we use the following PID controller formula:

$$u(t) = K_p \left[e(t) + \frac{1}{\tau_i} \int_0^t e(\tau) d\tau + \tau_d \frac{de}{dt} \right]$$

where K_p is the proportional gain
 τ_i is the integral time constant
 and τ_d is the derivative time constant.

The rule-based PID tuning methods described in this chapter each follow an explicit sequence of steps which make up the tuning procedure or algorithm. However, each of these steps corresponds to hidden or inherent calculations which we do not realise have been performed. Thus, we could follow the steps of the procedure without understanding the principles of the method at all. But it is only by understanding the principles that we can decide whether the method is an appropriate solution to the PID tuning problem being tackled. So we need to understand the *explicit* calculations that we have to perform in these routines and in parallel we need to understand the *implicit* reasons as to why these calculations are being made. We find that the easiest way to do this is to follow the general steps as they are laid out in Table 19.1. Each of the procedures to be described in the following sections has individual steps which fit the general procedural steps of the

figure. However, we generally find that there is no indication of the theoretical steps which have been used to devise the PID tuning procedure. Instead, we often find qualitative instructions are given, for example, we might be told that the rule for calculating the PID control parameters produces *little overshoot*. What this actually means is that if the system under test really did have the features of the model assumed for the theoretical control law derivation, then the PID law calculated we would indeed give *little overshoot* in the system response. Thus, the agreement between intended and actual control performance will depend on the degree of agreement between actual system features and the model assumed to describe these features.

Table 19.1 Explicit and implicit steps in PID rule-based tuning.

Explicit steps in the PID tuning procedure	Implicit steps in the PID Tuning procedure
↓	↓
<p>1. Make a system experiment with parameter measurements</p> <p>The system can be in open loop or closed loop depending on the method. A standard test signal like a step is usually used or the system is put into an extreme condition like the verge of instability.</p>	<p>1. A system model identification experiment is being performed</p> <p>The system identification will be performed as a parametric or a non-parametric identification. For example, a step response may be used to fit a simple parametric model:</p> $G(s) = \frac{Ke^{-sT}}{\tau s + 1} = \text{(first-order plus delay)}$
↓	↓
<p>2. PID control law parameters K_p, τ_i, τ_d are calculated by formulas</p> <p>Notes:</p> <p>(a) The PID controller is usually in the form:</p> $u(t) = K_p \left[e(t) + \frac{1}{\tau_i} \int_0^t e(\tau) d\tau + \tau_d \frac{de}{dt} \right]$ <p>(b) Depending on the structure P, PI, PD, PID a set of rules will be used for computing K_p, τ_i, τ_d.</p> <p>(c) The rules will use parameters calculated from experiments for example, process delay time, or system gain at the phase crossover point.</p>	<p>2. A PID control design has been completed to achieve a specific closed-loop performance</p> <p>Notes:</p> <p>(a) Each rule base has been devised assuming a particular model form, for example, first-order plus delay.</p> <p>(b) The formula used to compute K_p, τ_i, τ_d achieves different closed-loop performance depending on the particular model form.</p> <p>(c) The engineer selects the required PID controller structure. For example, including the I-term will ensure no steady state offsets.</p>
↓	↓
<p>3. Implement the control law</p> $u(t) = K_p \left[e(t) + \frac{1}{\tau_i} \int_0^t e(\tau) d\tau + \tau_d \frac{de}{dt} \right]$	<p>3. The implementation is performed by the hardware unit and is automatic once new parameters are available</p>

So although the PID tuning procedures to be described are simple to follow and apply to real plant and processes it is useful to remember where possible limitations might degrade the expected control law performance.

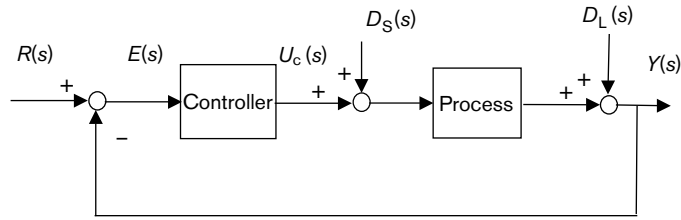


Figure 19.1 Closed-loop system diagram.

When we are specifying the desired control loop performance for our standard system (Figure 19.1), we should be taking a global view and considering performance against the usual list: closed-loop stability, disturbance and noise rejection and finally, robustness. We should then convert the performance list into a set of acceptable numerical values for the time and frequency domain specification variables. An example of the performance requirements is given in Table 19.2.

Table 19.2 Performance requirements.

Performance requirement	Comments
Closed-loop stability	An essential closed-loop control system requirement
Reference tracking Supply disturbance rejection Load disturbance rejection	These are generally time-domain performance specifications and requirements. Usually specified through design parameters like rise time (t_r), percentage overshoot (OS(%)), steady state error (e_{ss}), settle time (t_s)
Measurement noise rejection	This is a frequency domain performance specification for controller roll-off
Robustness of the control system performance to model uncertainty	These are frequency domain performance specifications and define the gain margin, phase margin and the maximum sensitivity

The PID control tuning procedures of this chapter usually only satisfy two or three of these design requirements together in a perfectly matched design. Yet in many industrial control designs we may be seeking to achieve six or more of these closed-loop system requirements in the same design. So we must not be too surprised if these techniques do not deliver the expected performance. For this reason we must be very sure about the actual level of performance we are seeking from a particular loop and use these requirements to assess the performance achieved from using a particular PID controller tuning procedure.

Before we describe the individual procedures of the chapter it is useful to introduce and revise some common concepts that are frequently referred to. We can discuss these using the two topics listed under the implicit principles of Table 19.1, namely, system identification and PID control design.

19.1.1 System identification aspects

The most commonly made assumption for the system identification step is that the system can be represented by a first-order lag-plus-deadtime model. The model transfer function is given by

$$G(s) = \frac{Ke^{-sT_d}}{\tau s + 1}$$

for which K is the d.c. gain, τ is the time constant and T_d is the delay time of the process. The response to a step of height h is shown in Figure 19.2.

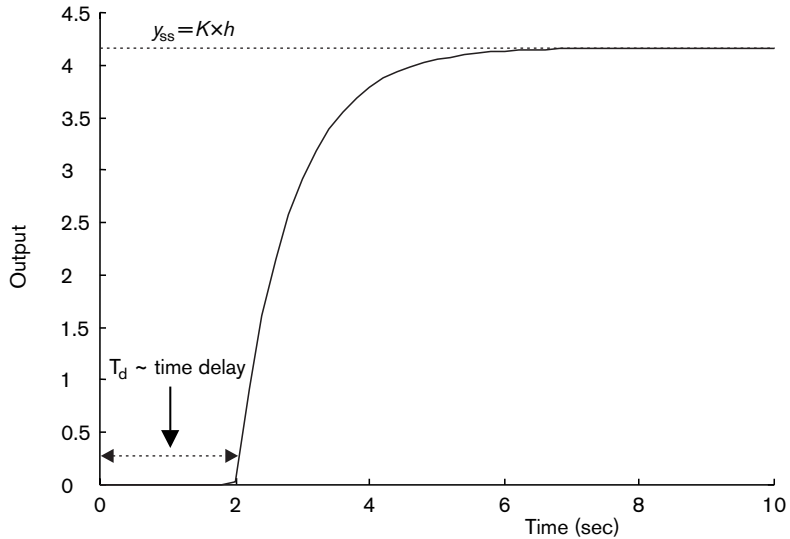


Figure 19.2 Step response of 'first-order lag plus deadtime' system.

There are several reasons why this model is commonly used in the PID tuning procedures to be described. One reason is that the model has only a few unknown parameters to be found; in fact, just three parameters, K , τ and T_d . Another reason is the practical one; we often find that a cascade of process units in a production facility or a process line produces a system step response resembling a time-delay followed by a first-order rise to a steady state output level. We use this real-world observation to make the engineering assumption that the process dynamics can be represented quite well by the first-order plus time delay transfer function model.

19.1.2 PID control design

The online experimental phase of the PID tuning procedures generates data for use in the evaluation of a formula taken from a rule table. We find that there are two issues to consider in the PID controller part of these procedures:

- A:** the *hidden* controller PID design specification that has been used to generate the set of formulas.
- B:** the choice of the terms that are to be used in the PID controller.

A: PID controller performance specification issues

Process control engineers and production managers are concerned with maintaining good quality process output values. To achieve this we usually find that the quality emphasis is on time-domain trend plots of physical variables like temperature, pressure, material thickness and liquid flow. This has affected the specifications of controller performance such that these specifications are mainly given in the time-domain based on the step response using qualitative labels like no overshoot (Figure 19.3(a)), a little overshoot (Figure 19.3(b)) or a quarter amplitude decay (Figure 19.3(c)). As control engineering technology progressed, more refined performance specifications were developed and we have learnt about many of these, for example, gain margin and phase margin, found in Chapter 14. However, we still find that the time domain specification of, for example, 'quarter amplitude decay' is often used today in industry. As we can see in Figure 19.3(c), a quarter amplitude decay means that the response decay is such that the second peak of the response is one quarter of the first peak. The PID controller parameters will be selected to achieve this response. Whether this type of closed-loop performance is desirable depends on the application, but it is useful to know that this corresponds to a damping ratio of 0.225 with 50% overshoot on step reference changes and a phase margin of 30° . Over the years we have seen that PID controller design rule bases have been formulated for various types of closed-loop performance behaviour. However, we have usually found it very difficult to find out exactly what controller design specification has been implemented by a particular controller manufacturer. Such information is usually commercially sensitive and often the only way to find out is to arrange for a plant test period and assess the performance from actual process data.

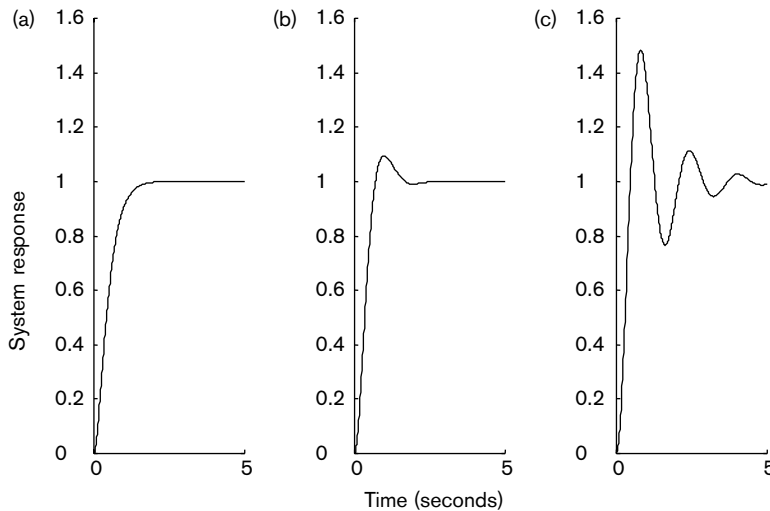


Figure 19.3 Qualitative performance: no overshoot, little overshoot, quarter amplitude decay.

B: Selecting terms in the PID controller

In the rule-based PID tuning procedures, the selection of the combination of PID terms is possibly the only place where we can exercise some of the knowledge that we have learnt about PID control. We gave a full discussion of the effects of the individual P, I, and D terms in the controller in Chapter 11. Now we use this knowledge to select the terms

appropriate for a practical control application. We have already learnt that the PID controller structures on offer are P, PI and PID, and we have to decide which one will achieve the performance we are looking for. We capture this in the question and answer flowchart of Figure 19.4.

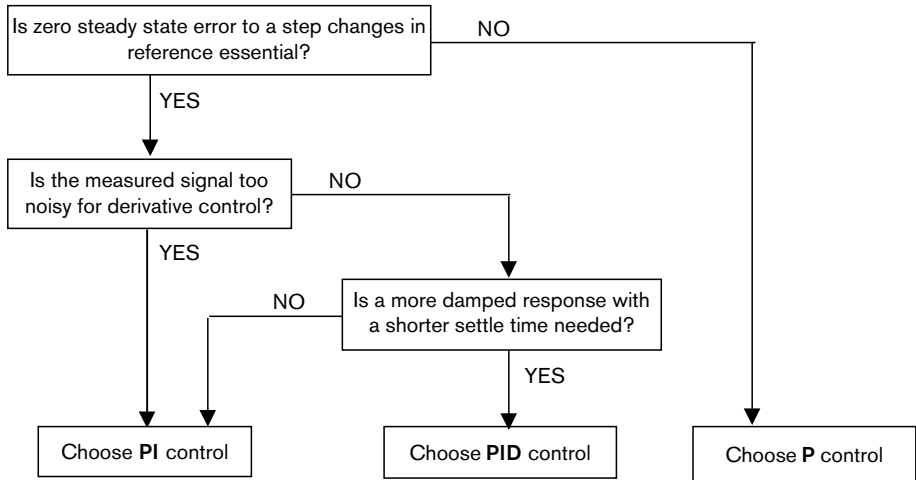


Figure 19.4 PID control – term selection chart.

19.2 Process reaction curve PID tuning method

The first method that we look at is one originally devised by Ziegler and Nichols. This is a manual procedure which is based on measuring the step response of the system. Thus, the underlying principle is that we can represent the system by a first-order lag-plus-deadtime model. It is also an open-loop method and therefore we can only use it with an open-loop *stable* process. The full procedure is given next.

Step 1: The process setup

The process is assumed to be in open loop and with steady conditions applying. The control input variable is denoted u_0 , and this is holding the output variable at the level y_0 . At this time and during the course of the experiment no process disturbances must be seen to occur, otherwise the test must be re-run.

Step 2: Applying an input step change

The input variable u is changed from u_0 to a new constant level, u_1 . This is an input step change of $\Delta u = u_1 - u_0$. The output $y(t)$ is monitored carefully and the step response recorded on a chart recorder, or as a digital file for later processing. We show the system setup and response in Figure 19.5.

To be able to apply this method, the step response must be capable of being reasonably well approximated by a first-order lag-plus-deadtime type of step response.

Step 3: Processing the data

The aim of this step is to use the recorded data to determine the parameters for use in the PID tuning rule-base. Consider the recorded output of the process variable (Figure 19.6).

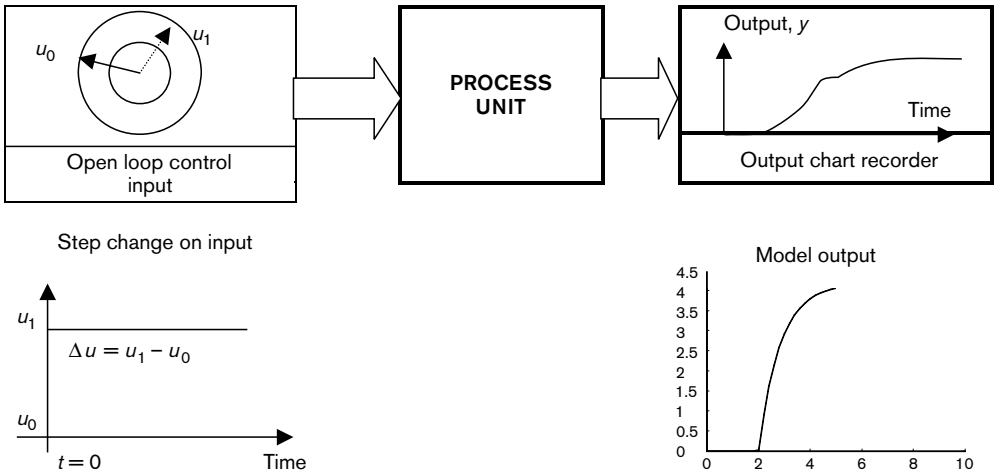


Figure 19.5 Process reaction curve method: plant experimental setup.

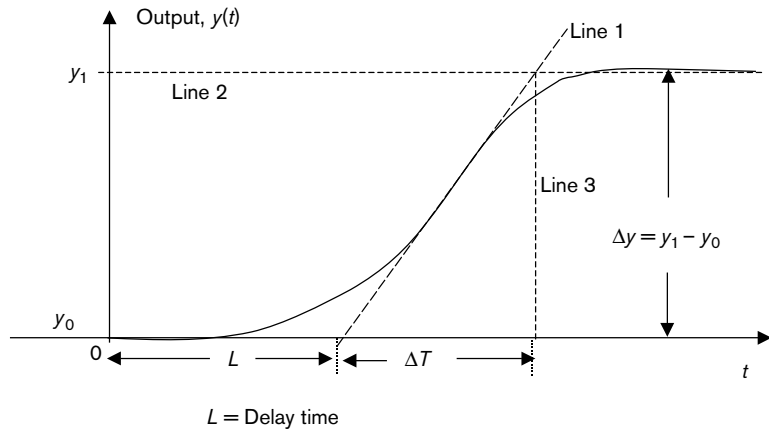


Figure 19.6 Recorded output trace.

In real plant tests the recorded measurement trace could be a little noisy, so care has to be taken that the level of noise does not invalidate the test results. Sometimes it is possible to see several data spikes on the measured trace. We call them *data outliers*. They are usually produced by the measurement process and are ignored in the data processing. On the recorded trace in Figure 19.6 we first draw *Line 1*, which is a tangent to the steepest part of the transient change. We then draw *Line 2*. This is drawn level with the final value attained by the output. Finally, the dotted *Line 3* is drawn to enable the following measurements and calculations to be made:

1. Measurement L ; this is the effective time delay in the system response.
2. Recorded output change, $\Delta y = y_1 - y_0$; this is the change in the level of the process output.
3. Measurement of ΔT and computation of R :

$$R = \left(\frac{\Delta y}{\Delta T} \right) = \frac{(y_1 - y_0)}{\Delta T}$$

This is the reaction rate for the system response.

4. Computation $R_N = (R/\Delta u)$ where $\Delta u = u_1 - u_0$; this is the reaction rate R normalised with respect to the effective input changes, Δu .

Step 4: Calculating the PID parameters

The PID controller has the common industrial form:

$$u(t) = K_p \left[e(t) + \frac{1}{\tau_i} \int_0^t e(\tau) d\tau + \tau_d \frac{de}{dt} \right]$$

The PID parameters K_p , τ_i and τ_d are required. The data from the experiment are the estimated process time delay, L , and the estimated normalised process reaction rate, R_N . The PID controller settings recommended by Ziegler and Nichols are presented as a table of rules or computation formula in Table 19.3.

Table 19.3 Ziegler–Nichols PID settings – reaction curve method.

Controller structure	Proportional gain, K_p	Integral time constant, τ_i	Derivative time constant, τ_d
Case (i) P	$\frac{1}{R_N L}$	–	–
Case (ii) PI	$\frac{0.9}{R_N L}$	$3L$	–
Case (iii) PID	$\frac{1.2}{R_N L}$	$2L$	$0.5L$

The *hidden* controller performance specification for this rule base is a closed-loop response which exhibits a quarter amplitude decay. To use the table, we first have to decide on the structure of the PID controller to be used and then compute the values of the parameters, K_p , τ_i , and τ_d in the PID structure. For example, if we decided that the controller must have integral action but that the process variable measurement was too noisy for D-Action, then we would select PI control. We would read the formula on the line Case (ii) in the table, which gives the computations $K_p = 0.9/(R_N L)$ and $\tau_i = 3L$.

Step 5: Implement and assess the controller

The controller parameter values calculated in Step 4 are now ready for implementation. For this we would probably use the data input facilities of the process controller unit (usually a keypad) or the control engineer's interface window if a SCADA system is being used. We must remember that the parameters calculated using the rules do not *guarantee* closed-loop stability for all systems. Systems which do not look similar to the first-order lag-plus-deadtime model will probably not be stabilised by the method. To complete the tuning exercise we would recommend to the plant management that a thorough assessment of closed-loop performance should be made. This could use an agreed set of explicit

test experiments, or we would suggest subjective assessment by process operator observations and records of closed-loop process performance.

Problem A technician produces the Process Reaction Curve data shown in the works *pro forma* table which has been designed to smooth the plant's PID tuning work.

WJK Chemical Works – Controls Department, Grangemouth			
Date of Test	23 March, 2000	Process Unit Code	U47-898
Input Level – Start	100	Output Level – Start	300 °C
Input Level – End	110 (10% step)	Output Level – End	330 °C
Comments Furnace temperature sensor noisy, no data outliers, no load disturbances. Data file sent to Controls on 24 March, 2000			
Authorised by	J. Arturson	Date	22/03/00

PID control design

- (a) Use the reaction curve method to process the data file and produce the tuning parameters for a PI controller.

Controller assessment

- (b) Fit an appropriate first-order plus time delay model transfer function, given by

$$G(s) = \frac{Ke^{-sT_d}}{\tau s + 1}$$

for which K is the d.c. gain, τ is the time constant and T_d is the delay time of the process.

(c) Use a Simulink simulation to obtain some idea of the likely success of the PI tuning.

Solution (a) We follow the steps of the reaction curve method.

Step 1: the process setup

The control input variable is denoted u_0 , and this is holding the output variable at the level y_0 . In this example we find that $u_0 = 100$ and $y_0 = 300$. We note from the comments on the *pro forma* sheet that there were no process disturbances during the course of the experiment, so we can assume that the data is valid for the reaction curve method.

Step 2: applying an input step change

The new constant level, $u_1 = 110$; hence the input change is $\Delta u = u_1 - u_0 = 110 - 100 = 10$.

The output data looks like a first-order plus delay type of step response, so we expect the design method to work well.

Step 3: processing the data

We see that the data is noisy but that there are no data outliers. On the recorded trace (Figure 19.7) we first draw *Line 1*, which is a tangent to the steepest part of the transient change, *Line 2*, which is level with the final value attained by the output, and finally *Line 3* is drawn in vertically to allow the following measurements to be made:

1. $L = 0.4$
2. $\Delta y = y_1 - y_0 = 330 - 300 = 30$
3. $R = \left(\frac{\Delta y}{\Delta T}\right) = \frac{(y_1 - y_0)}{\Delta T} = \left(\frac{30}{4.5}\right) = 6.67$
4. Computation $R_N = \left(\frac{R}{\Delta u}\right) = \left(\frac{6.67}{10}\right) = 0.667$

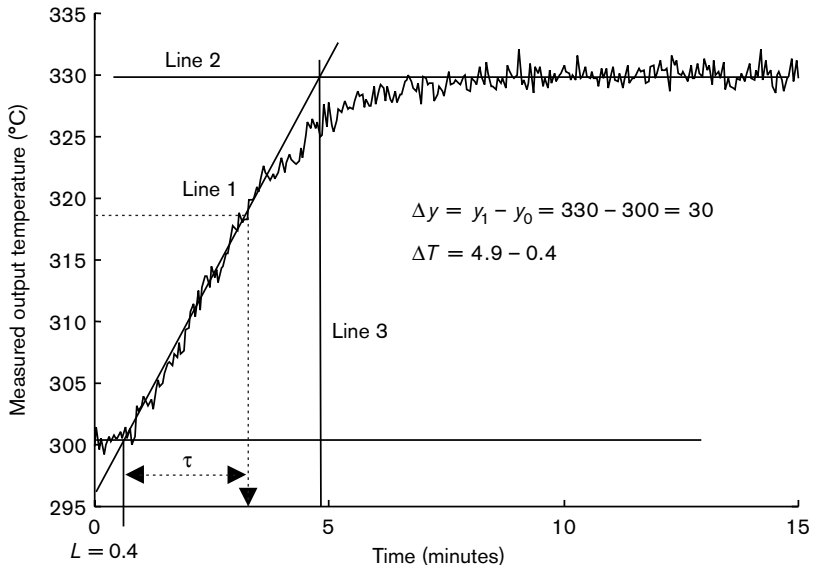


Figure 19.7 Process reaction curve: measurements on recorded trace.

Step 4: calculating the PID parameters

The PID controller is assumed to have the form

$$u(t) = K_p \left[e(t) + \frac{1}{\tau_i} \int_0^t e(\tau) d\tau + \tau_d \frac{de}{dt} \right]$$

The PID parameters K_p , τ_i and τ_d are required. We are to find a PI controller, so we set the derivative time constant to zero, $\tau_d = 0$. We use the following line from the Ziegler–Nichols table:

Controller structure	Proportional gain, K_p	Integral time constant, τ_i	Derivative time constant, τ_d
Case (ii) PI	$\frac{0.9}{R_N L}$	$3L$	–

We compute

$$K_p = \left(\frac{0.9}{R_N L} \right) = \left(\frac{0.9}{0.667 \times 0.4} \right) = 3.37$$

and $\tau_i = 3L = 3 \times 0.4 = 1.2$ with $\tau_d = 0$.

The final step of the procedure would be to perform the controller assessment.

Step 5: controller assessment

(b) Fit an appropriate ‘first-order lag-plus-deadtime’ model transfer function given by

$$G(s) = \frac{Ke^{-sT_d}}{\tau s + 1}$$

From the data analysis step we have that $L = T = 0.4$; it is not difficult to determine the model time constant, τ . We first need to find the value y_τ at which $y(t)$ reaches 63% of the rise in value.

$$y_\tau = 300 + 0.632 \times (330 - 300) \approx 319$$

Secondly, we need to find the time it takes for $y(t)$ to reach this value. Thus, from Figure 19.7, the time it takes is 3.5 seconds. However, we remember that there is a delay of 0.4 seconds, which gives

$$\tau = 3.5 - 0.4 = 3.1 \text{ seconds}$$

We need to find the d.c. gain of the model now; for this we use

$$K = \left(\frac{\Delta y}{\Delta u} \right) = \left(\frac{(y_1 - y_0)}{(u_1 - u_0)} \right) = \left(\frac{330 - 300}{110 - 100} \right) = \left(\frac{30}{10} \right) = 3$$

The model may now be given as

$$G(s) = \left[\frac{3e^{-s0.4}}{(3.1s + 1)} \right]$$

(c) A Simulink model is shown in Figure 19.8. This requires that we use the model

$$G(s) = \left[\frac{3e^{-s0.4}}{(3.1s + 1)} \right]$$

and the PI control

$$u(t) = 3.37 \left[e(t) + \frac{1}{1.2} \int_0^t e(\tau) d\tau \right] = 3.37e(t) + 2.81 \int_0^t e(\tau) d\tau$$

We have added measurement noise to the output. The parameters we used were: mean of zero, variance of 0.5, initial seed 61233 and sample time of 0.5.

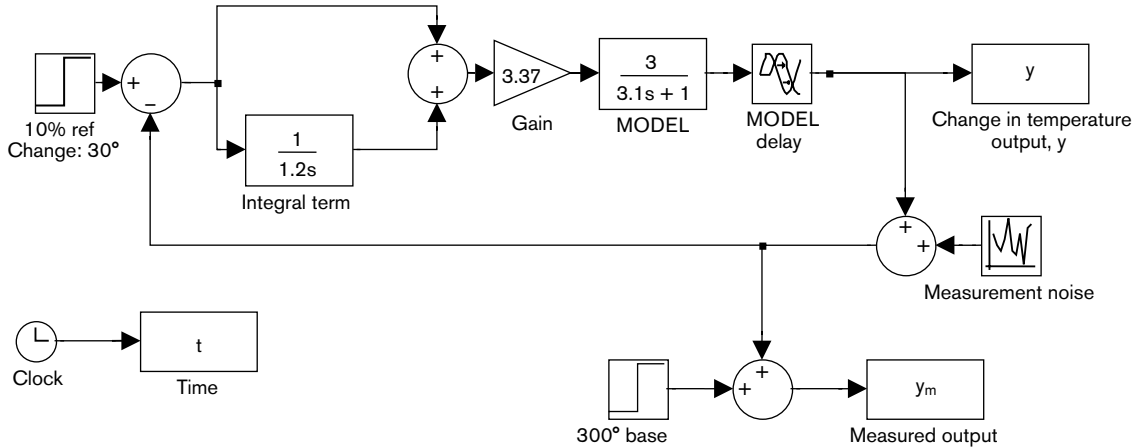


Figure 19.8 Simulink model for PI control.

The Simulink model inputs a step of 30° (10% of the 300° reference temperature) to the model. The change in temperature is given by the variable y . We have added the base operating temperature of 300 °C to our measured output in Figure 19.9.

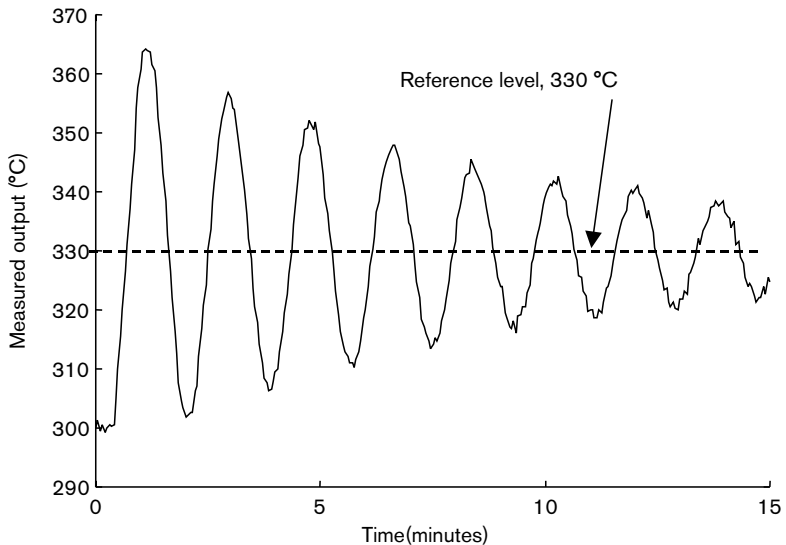


Figure 19.9 PI Control system output.

We see quite clearly from Figure 19.9 that the design is closed-loop stable, but that it is very oscillatory. We also notice that there has been a noise filtering effect in the process. The design is only useful for a starting design, and some more tuning is going to be necessary to dampen out these oscillations. In particular, we introduce some derivative control, as shown in the simulation model: Figure 19.10.

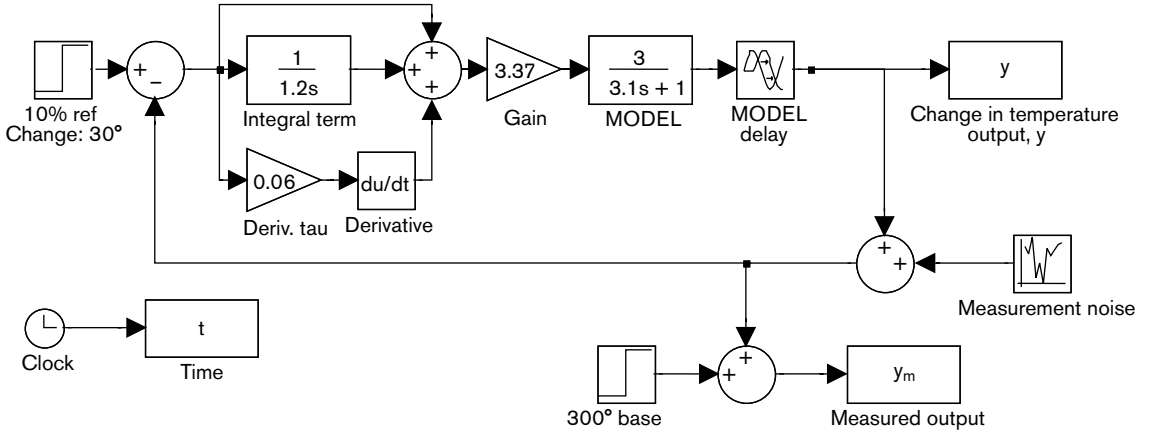


Figure 19.10 Process reaction curve: PI control + D added.

The fascinating effect displayed here is that just a little bit of derivative control, where the derivative time constant is set to $\tau_d = 0.06$, has a really soothing effect on the oscillatory nature of the step response. But note also that the output trace is just a little bit noisier than the PI system output trace. This can be seen in Figure 19.11.

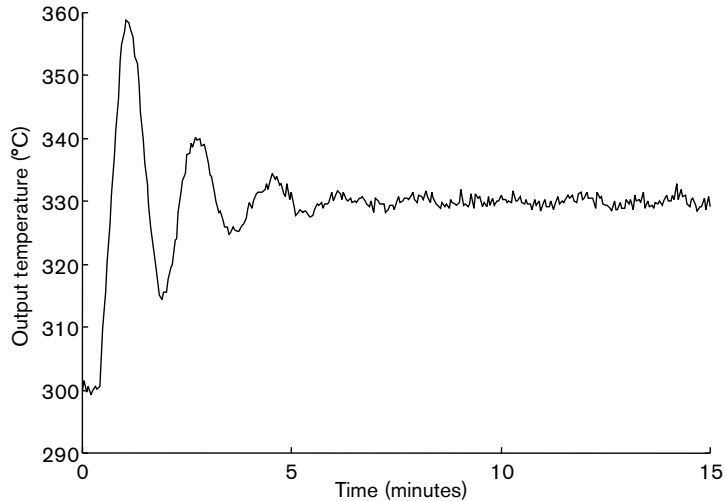


Figure 19.11 Process reaction curve method: output response for PI + D control.

A summary of the requirements and advantages and disadvantages of the process reaction curve method is given in Table 19.4.

Table 19.4 Summary for the process reaction curve method.

System setup	Open loop
System restrictions	Open-loop stable For reasonable success, first-order-like step responses
Test signal	Manually generated Step signal One test signal only
Test signal issues	No disturbances during test Minimal noise permitted Test signal moves process away from operating point and nonlinear process effects possible Test signal may have to be large to give good system identification; could cause production disruption
PID design rules	Quarter amplitude decay design
Advantages	Simple procedure and computations Good method to find initial PID settings Low level of process/control knowledge needed
Disadvantages	Response too oscillatory Overshoot too large Design specification too limited

19.3 Sustained oscillation PID tuning

In practical situations in industry, we often find that process operators are very reluctant to (1) open up closed-loop systems for step response tests and (2) allow significant sized test signals to be injected into process units. At the same time production management will be energetically vigorous in their quest to improve quality at all stages of the production process. If we wish to tune and re-tune PID control loops quickly, we soon find that an alternative to the process reaction curve tuning method is needed. The main requirement of the new method is that we should use the system in closed loop. At the same time as Ziegler and Nichols formulated the open-loop process reaction curve method, they also introduced a new closed-loop PID tuning method. We know this as the Ziegler–Nichols method of sustained oscillations. It is a simple idea and it has provided the practical and theoretical framework for a long-lasting and highly successful PID tuning technology. The basic principle at work in this method begins from the industrial system being in closed loop, but only under proportional control. In the real world we find that, for a large class of industrial systems, the proportional gain can be increased until a sustained oscillation or continuous cycling of the process output variable is observed. What has happened is that the sustained oscillation in the output indicates that the proportional gain value has brought the system to the verge of closed-loop instability. For this reason, we call this critical proportional gain value the *ultimate* gain, denoted K_u . We can use the control systems theory of earlier chapters to establish a link between K_u and the gain margin of the system, GM .

19.3.1 Basic principles of the theory behind the sustained oscillations

To follow the basic principles of the theory we use Figure 19.12. The Bode plot shows the system frequency response. To be at the verge of instability and to show sustained oscillations we would have to have a gain margin of 0 dB. To achieve this we would have to raise the magnitude plot by $K_x \text{ dB} = 20 \log_{10} K_u \text{ dB}$. Alternatively, we see on the Nyquist plot that we can multiply the response by the ultimate gain, K_u , and have the frequency point, ω_{pc0} , move out to the '-1' point. From these ideas we can easily find a link between the existing open-loop gain plot, $G(j\omega)$, the ultimate gain, K_u , the gain margin, GM , and a desired gain margin, GM^D .

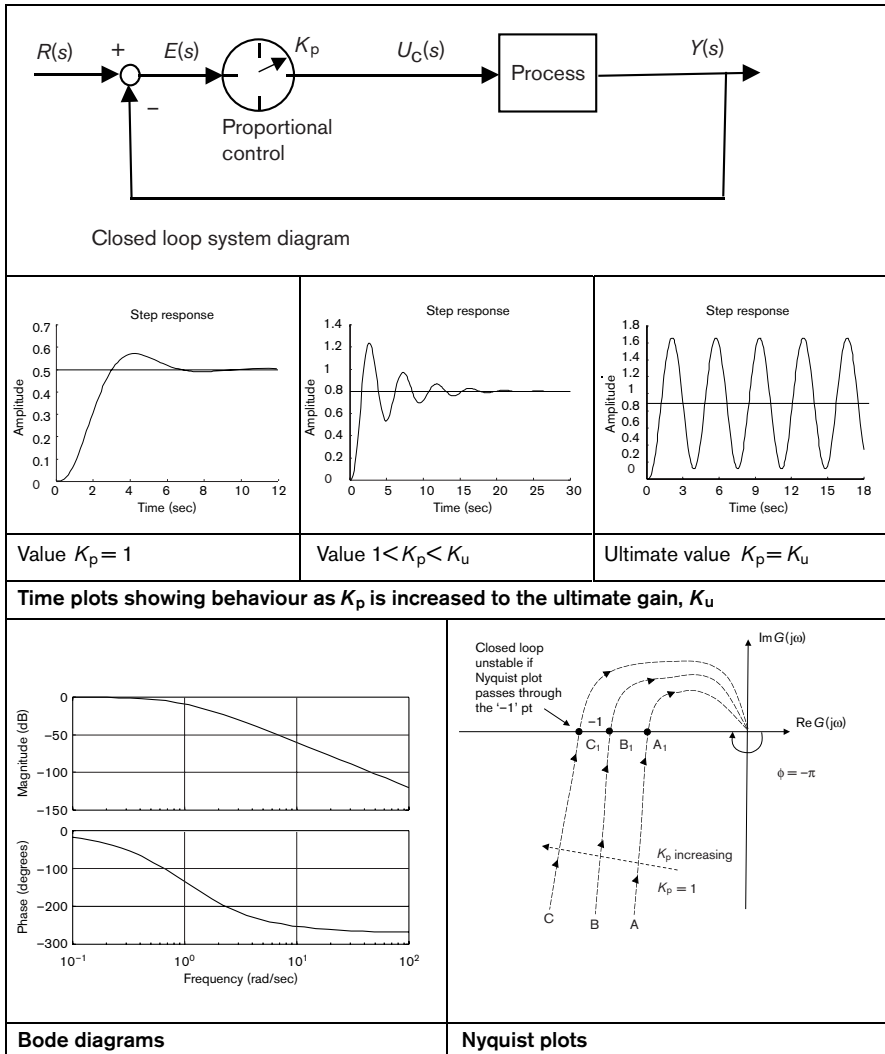


Figure 19.12 Sustained oscillations – time and frequency domain plots.

In Figure 19.12 we can see that the effect of increasing the system gain is to achieve a sustained oscillation at the system output. This is a time domain trace which we can record and use for measurements.

When we have turned up the gain to reach the point of sustained oscillation, we can record the value of the proportional gain as the ultimate gain, K_u . We know that the system is at the verge of instability and the output is oscillating at the frequency ω_{pco} . We have the relation

$$K_u \times |G(j\omega_{\text{pco}})| = 1$$

and the unknown system information $|G(j\omega_{\text{pco}})|$ is then given by

$$|G(j\omega_{\text{pco}})| = 1/K_u$$

To find a proportional gain, K_p , to achieve a desired gain margin, GM^D we use the same type of calculation. We need to find a proportional gain, K_p , so that $GM^D \times K_p |G(j\omega_{\text{pco}})| = 1$, which rearranges to give

$$K_p = 1/(GM^D \times |G(j\omega_{\text{pco}})|)$$

From the sustained oscillation experiment, we have a measurement for $|G(j\omega_{\text{pco}})| = 1/K_u$, which we use. Clearly we find

$$K_p = K_u/GM^D$$

Since K_u is measured from the sustained oscillation experiment, and GM^D is a design specification, computation of the appropriate proportional gain K_p is straightforward. The full Ziegler–Nichols sustained oscillation tuning procedure builds on these fundamentals to yield a PID controller parameter-setting routine. The rule base also uses the phase crossover frequency information in the form of an ultimate period P_u , where $P_u = 2\pi/\omega_{\text{pco}}$. We measure this from the oscillating $y(t)$ or $e(t)$ signals using a peak-to-peak measurement, a zero crossing measurement, a peak-to-trough measurement or a trough-to-trough measurement. These are simple procedures to automate and help to make an automatic version of this procedure feasible, as we will see later in the chapter, but for now we follow the individual steps of the sustained oscillation procedure.

19.3.2 Sustained oscillation PID tuning procedure

Step 1: The process setup

We start by assuming that the process is in steady state and under closed-loop control. We try to make sure that measurement noise is minimal and that no process disturbances occur during the experiment. We disengage the integral and derivative terms of the controller. This may require the use of a large integral time constant, τ_i , and a zero value for the derivative time constant, τ_d .

Step 2: The experimental stage

The proportional gain K_p is carefully incremented and closed-loop pulse response tests are performed until a sustained oscillation is observed in the output variable. This will not be a perfect oscillation, but a very slowly decaying one. Why? Because it is necessary to strictly maintain closed-loop stability, otherwise an unstable process loop may occur. The final oscillatory trace found should be recorded.

Step 3: Processing the data

We should make sure we have a note of the ultimate gain K_u value. From the sustained oscillation trace it is necessary to measure the ultimate period, P_u . If there is noise on real process data then we might need to find the mean of several P_u readings. Data outliers on

the trace should be ignored. A typical trace is shown in Figure 19.13, where both measurement noise and data outliers are present.

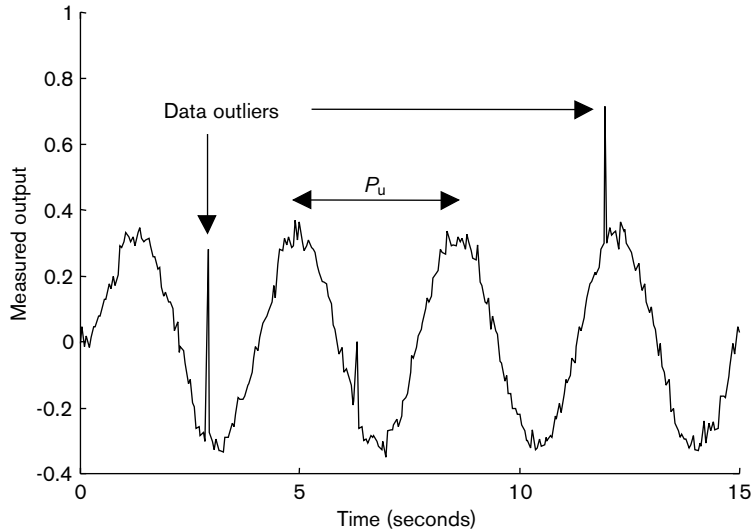


Figure 19.13 Measurements for sustained oscillation tuning method.

Step 4: Calculating the PID parameters

The industrial PID controller form is assumed:

$$u(t) = K_p \left[e(t) + \frac{1}{\tau_i} \int_0^t e(\tau) d\tau + \tau_d \frac{de}{dt} \right]$$

The data from the experiment are:

Estimated ultimate gain, K_u

Estimated ultimate period, P_u

and a table from the *Chemical Engineers' Handbook* (eds. R. H. Perry and C. H. Chilton, 1973; published by McGraw-Hill, New York) is used (Table 19.5).

Table 19.5 Sustained oscillation PID tuning rules.

Controller structure	Performance criterion	Proportional gain, K_p	Integral time constant, τ_i	Derivative time constant τ_d
Case (i) P	1/4 decay	$0.5K_u$	–	–
Case (ii) PI	1/4 decay	$0.45K_u$	$0.833P_u$	–
Case (iii) PID	1/4 decay	$0.6K_u$	$0.5P_u$	$0.125P_u$
Case (iv) PID	Some overshoot	$0.33K_u$	$0.5P_u$	$0.33P_u$
Case (v) PID	No overshoot	$0.2K_u$	$0.33P_u$	$0.5P_u$

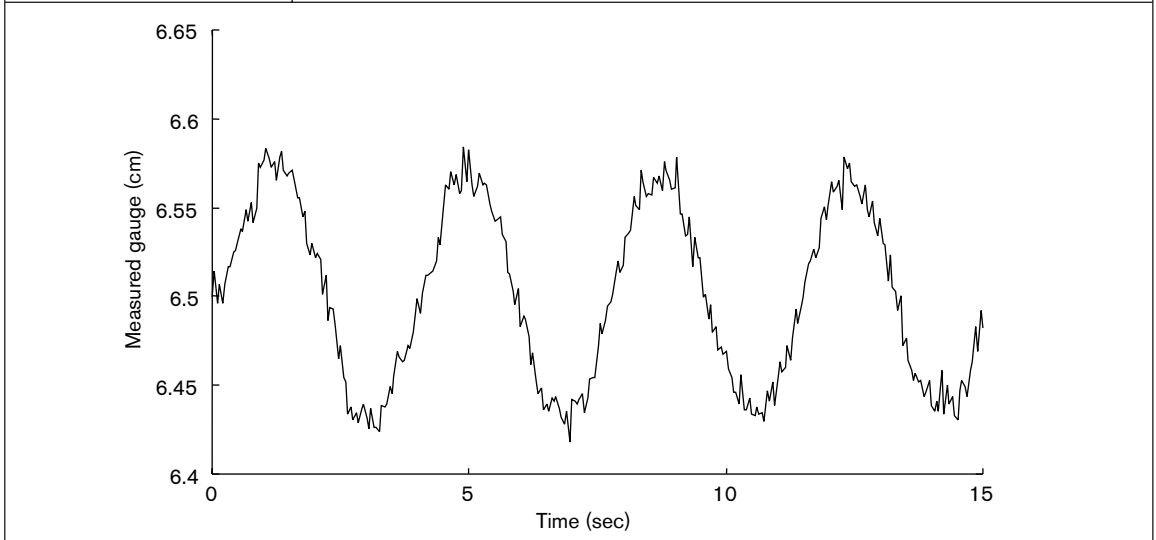
In this rule base the controller design philosophy has been made explicit, and we have a wider choice of performance outcomes to choose from.

Step 5: Implement and assess performance of the controller

After implementation of the controller parameters, assessment of performance would be from real-time and historical plant data analysis.

Problem In a steel works a sustained oscillation test was performed and the System Engineering Department form filled out by the operand. This data form has been constructed to enable simple rule-based tuning to be performed. On completion of the data analysis the PID coefficients were to be implemented by the Maintenance Department.

JMR Steel plc, Rotherham, England, UK		We provide the world's best steel!	
Process Unit	Stand 4E, McCance	Test Date	28 June 1998
Input reference	6.5 cm	Output level – start	6.5 cm
Input pulse	+0.65 cm (10% step)	Oscillating gain value	2.5
Pulse time	0.1 s	Noise variance	0.00005
Test approval	John Michaelson	Position	Section Leader
Comments	Data collection went well. Quite noisy.		



- (a) The data correspond to a sustained oscillation test. Process the data and produce the tuning parameters for a PID controller using the rules to give *some overshoot*.
- (b) The works design manual gives a transfer function model for the stand unit as $G(s) = 3/(s + 1)^3$. Use a Simulink simulation to obtain some idea of the likely success of the PID tuning.

Solution (a) We follow the steps of the sustained oscillation PID tuning procedure.

Step 1: the process setup

We assume that the test conditions satisfied those required for the test to be a success. In particular, we assume that the process was in steady state; the data sheet says that the steady level at the start of the test was 6.5 cm and that a +10% pulse was applied for 0.1 s.

Step 2: the experimental stage

The proportional gain K_p was incremented in closed loop until a sustained oscillation was observed. The data sheet shows that this occurred at $K_p = 2.5$. This is the value of the ultimate gain, K_u . The final oscillatory trace found was recorded.

Step 3: processing the data

We need to measure the ultimate period, P_u . If there is noise or data outliers on the traces we have to do the best we can, since this procedure is only approximate. The trace is shown in Figure 19.14 where the measurement of P_u is shown.

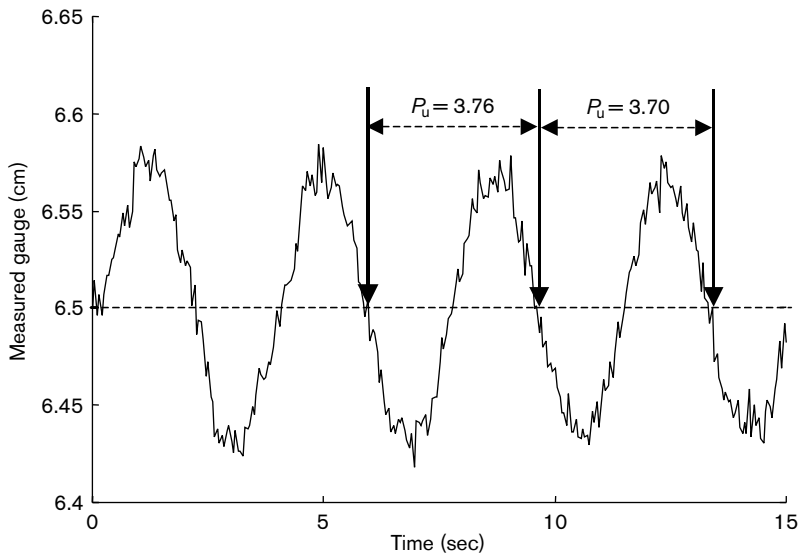


Figure 19.14 Sustained oscillation test recording.

We should ensure that the trace has settled, and if the data is noisy we should take the average of several readings. In this case we take P_u to be the average of 3.76 and 3.70: $P_u = 3.73$.

Step 4: calculating the PID parameters

The industrial PID controller form is assumed:

$$u(t) = K_p \left[e(t) + \frac{1}{\tau_i} \int_0^t e(\tau) d\tau + \tau_d \frac{de}{dt} \right]$$

The data from the experiment yields the estimated ultimate gain, $K_u = 2.5$, and the estimated ultimate period, $P_u = 3.73$. From the rule base we select the formula line marked some overshoot.

Controller structure	Performance criterion	Proportional gain, K_p	Integral time constant, τ_i	Derivative time Constant τ_d
Case (iv) PID	Some overshoot	$0.33K_u$	$0.5P_u$	$0.33P_u$

We calculate the PID parameters as follows:

$$K_p = 0.33K_u = 0.33 \times 2.5 = 0.825$$

$$\tau_i = 0.5 P_u = 0.5 \times 3.73 = 1.865$$

$$\tau_d = 0.33 \times 3.73 = 1.23$$

In a real tuning exercise we are now ready to complete the procedure.

Step 5: implement and assess performance of the controller

(b) We are going to use the transfer function model $G(s) = 3/(s + 1)^3$ and Simulink to obtain some idea of the performance of the PID tuning.

We write

$$u(t) = K_p \left[e(t) + \frac{1}{\tau_i} \int_0^t e(\tau) d\tau + \tau_d \frac{de}{dt} \right] = 0.825 \left[e(t) + \frac{1}{1.865} \int_0^t e(\tau) d\tau + 1.23 \frac{de}{dt} \right]$$

The Simulink simulation is given in Figure 19.15 with the measurement block parameters of mean 0, variance 0.00005, seed 61233 and sample time 0.05. The 10% (0.65 cm) step response plot in Figure 19.16 shows an overshoot of 23% in the measured output, which is a little high for most design purposes, but the method has produced quite good results for the very little process information which was supplied to the design.

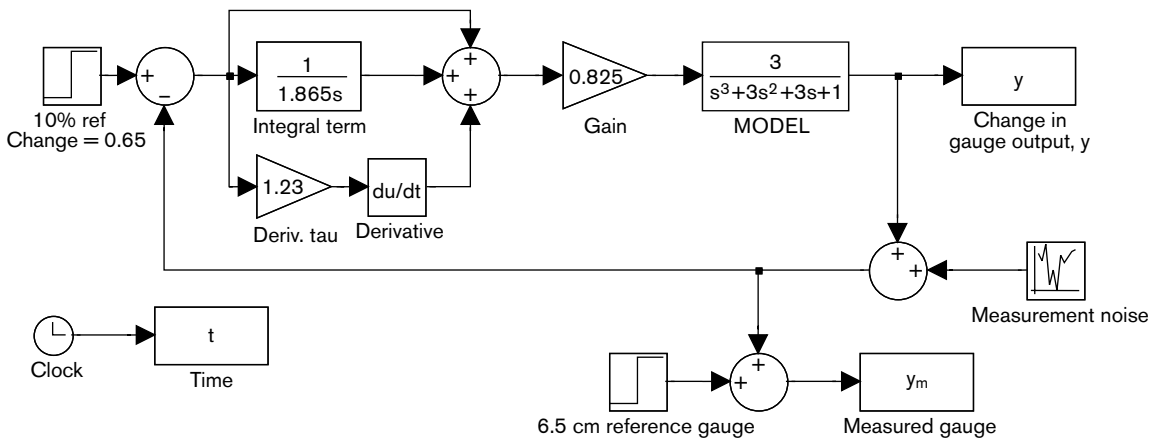


Figure 19.15 Simulink model for sustained oscillation assessment.

A summary of the sustained oscillation procedure and the advantages and disadvantages of the method is given in Table 19.6.

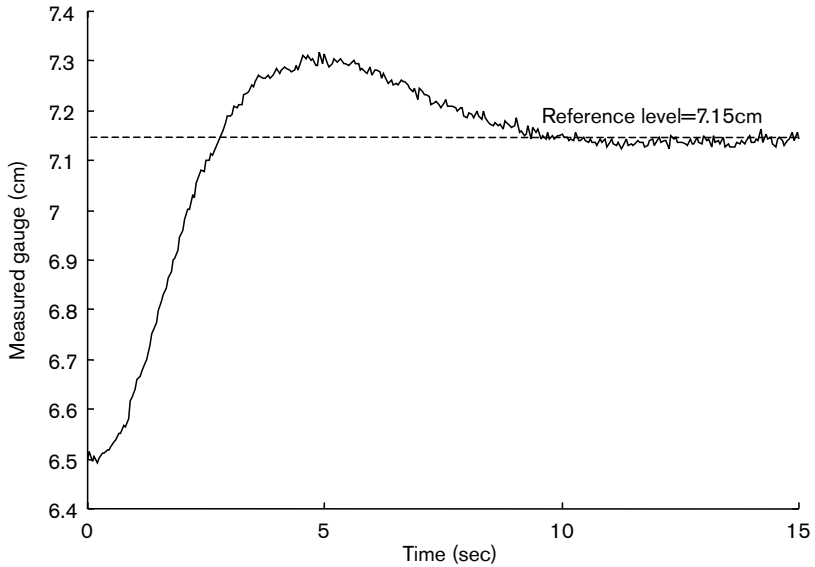


Figure 19.16 Step response for sustained oscillation test model.

Table 19.6 Summary: sustained oscillation method.

System setup	Closed-loop Proportional control only
System restrictions	System must have phase lag in excess of -180° otherwise sustained oscillation not possible. Most industrial systems have delays and/or nonlinearity present and these usually ensure a sustained oscillation will occur as the proportional gain is increased
Test signal	Manually generated Short pulse to initiate the oscillating response Sequence of tests needed to find K_u
Test signal issues	No disturbances during test Minimal noise permitted Oscillation may have to be large to ensure sufficient measurement resolution A sequence of tests needed; may be time-consuming
PID design rules	Quarter amplitude decay design Other qualitative rules available
Advantages	Simple non-parametric model assumed Procedure has good theoretical basis relating it to more advanced controller specifications Simple procedure and computations Process test performed in closed loop
Disadvantages	Verge of closed-loop instability reached Procedure requires multiple tests and can be disruptive and lengthy

19.4 Damped oscillation or quarter amplitude decay PID tuning procedure

Despite the simplicity of the sustained oscillation procedure, we do not think that the idea of operating a closed-loop system close to the verge of instability is very practical or desirable. Consequently, we use a variant of the sustained oscillation procedure known as the damped oscillation or quarter amplitude decay method instead. In this procedure, the proportional gain alone is used to try and achieve a quarter amplitude decay response and the tuning rules use data extracted from this.

Step 1: The process setup

As with the sustained oscillation method, we make the tuning experiment with the system in closed loop. We also disengage the integral and derivative actions. This we achieve by making the integral time constant, τ_i , as large as possible, and setting the derivative time constant, τ_d , to zero. During the experiment we will need to ensure that no process disturbances occur and that measurement noise is minimised.

Step 2: The experimental stage

The objective of this step is to adjust the proportional gain, K_p , until we obtain the quarter amplitude decay step response shape. Thus, experimentally a sequence of step response tests will be performed. However, we should remember that real practical systems do not always have nice perfect second-order responses. So we must exercise some good judgement to obtain a reasonable approximation to the quarter amplitude decay response and terminate the procedure sensibly. If we fail to do so we will make an excessive number of step responses tests in search of an accuracy which is simply unattainable from the system! We decompose this step into a number of sub-steps as follows.

Step 2.1: initialise

Set the loop counter to zero: $k = 0$.

Choose the initial proportional gain K_p so that the closed-loop system is stable.

Step 2.2: step response test

Run an appropriate step response for the system, the step change being applied to the closed-loop system reference input. We record the step response trace.

Step 2.3: measurement step

We use Figure 19.17 for the details of this step.

Measure the amplitudes A_k and B_k .

Compute ratio $R_k = B_k/A_k$; if $R_k \approx 0.25$ then measure the period, P , between the first and second peaks and go to Step 3.

If $R_k \ll 0.25$ then increase K_p

If $R_k \gg 0.25$ then decrease K_p

Set $k = k + 1$ and repeat Step 2.2.

As we can see, this is an engineering procedure which needs some trial and error investigation to decide on the appropriate gain size for *increase* K_p or *decrease* K_p . Also, the accuracy with which the process can deliver a quarter amplitude decay response will only be discovered by careful observation of the actual system responses.

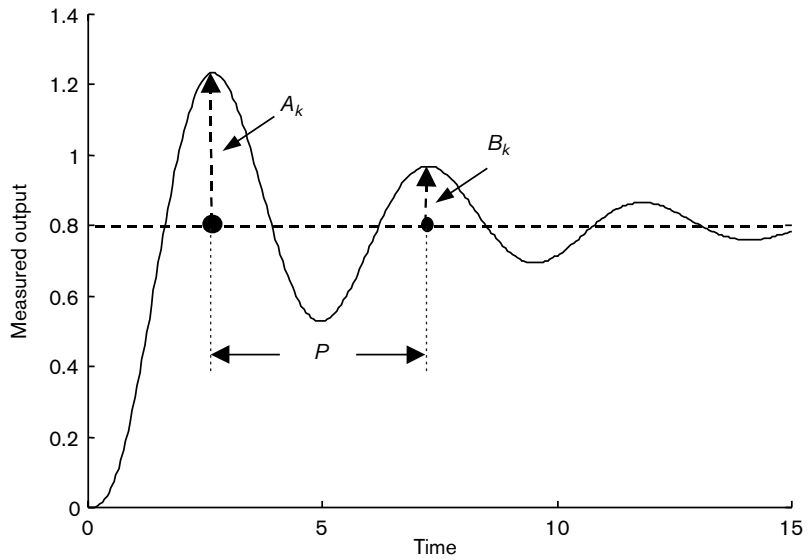


Figure 19.17 Damped oscillation method: measurements to be made.

Step 3: Calculating the PID parameters

The data obtained from the experimental stage are:

1. K_p , the proportional gain yielding an approximate quarter amplitude decay response.
2. P , a measurement of the period between first and second peaks of the quarter amplitude decay response.

The damped oscillation method provides only a rule base for a full PID controller, as given in Table 19.7.

Table 19.7 PID tuning for damped oscillation data.

Controller structure	Performance criterion	Proportional gain, K_p	Integral time constant, τ_i	Derivative time constant, τ_d
PID	1/4 decay	K_p	$P/1.5$	$P/6$

Step 4: implement and assess controller performance

The calculated controller parameters are ready for use. We would complete the tuning exercise with a thorough assessment of closed-loop performance. This would use a set of explicit test experiments or we would suggest subjective assessment by process operator observations and records of closed-loop process performance over a period of time.

Problem At the Clyde Power company, the engineers favour the quarter amplitude decay PID tuning procedure. An engineer has been running the procedure for a fan pressure loop in the power station and has completed the company data sheet for processing by the Instrumentation and Controls Department. The data collection sheet is given on p. 603.

- (a) Examine the trace obtained and comment on the possible type of process under control. Will the process be easy to control?
- (b) Process the data and produce the tuning parameters for the quarter amplitude decay PID tuning method.
- (c) It is found from the design reports that the transfer function model for the process is given as

$$G(s) = \frac{s^2 - 2s + 2}{(s + 3)^3(s^2 + s + 4.25)}$$

Use a MATLAB root locus plot to analyse the possible control difficulties. Use a Simulink simulation to obtain more insight into the success of the PID tuning.

- Solution
- (a) The data trace is obtained from a positive-going step, yet what we see is a small transient before the main transient gets under way. We associate this behaviour with an inverse response type and anticipate a Right Half Plane zero effect. We know the test was performed under proportional control only. We see that the process has not achieved a steady state value anywhere near 21 bar, so the integral control is going to have a significant job to do. The sizes of the response values are also very small, hardly moving away from the 20 bar level. We therefore think that the open-loop process gain is very small.
 - (b) To produce the PID tuning parameters for the quarter amplitude decay PID tuning method, we need the final K_p value, and the period, P , between first and second peak values. From the data sheet we have $K_p = 0.2$, and the trace yields a period of 3 seconds. We complete the tuning table in Table 19.8.

Table 19.8 Tuning for damped oscillation data for fan pressure loop.

Controller structure	Performance criterion	Proportional gain, K_p	Integral time constant, τ_i	Derivative time constant, τ_d
PID	1/4 decay	K_p	$P/1.5$	$P/6$
Numerical values		0.2	$\tau_i = 3/1.5 = 2$	$\tau_d = 3/6 = 0.5$

- (c) We are given the transfer function model for the process as

$$G(s) = \frac{s^2 - 2s + 2}{(s + 3)^3(s^2 + s + 4.25)}$$

Using the MATLAB function conv we find tha the numerator and denominator can be written as

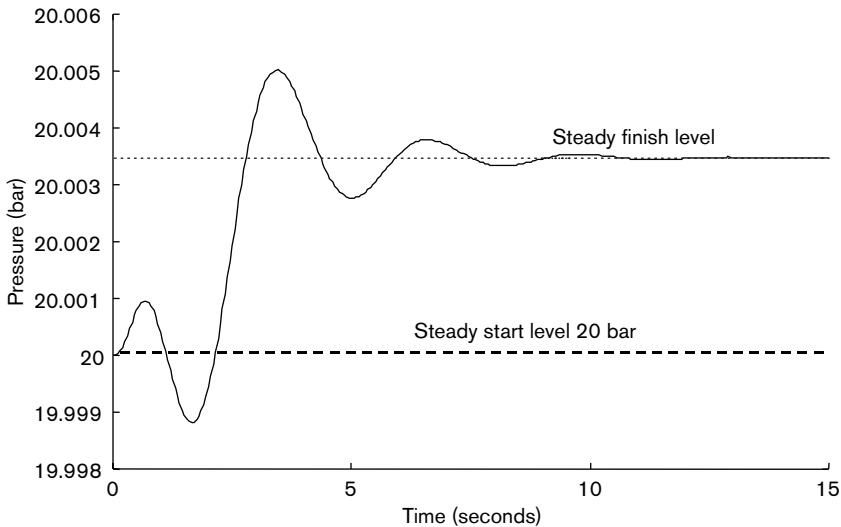
```
gdenom conv([ 1 9 27 27], [1 1 4.25])
gnum = [1 -2 2]
```

This will enable us to have a look at the possible control difficulties and the time domain performance of the PID design we have performed. From the transfer function we can find the poles and zeros.

The open-loop poles: solve $(s + 3)^3(s^2 + s + 4.25)$ and we find $s = -3, -3, -3$, and $s = -0.5 \pm 2j$ as the open-loop poles. This complex conjugate pair of poles is close to the imaginary axis and will give rise to a very underdamped type of response.

Test sheet	Generating power for Scotland			Clyde Power plc Glasgow, Scotland
Position	Section Leader	Authorised by		Wilkie MacJack
Comments	Loop free from measurement noise. Procedure took 20 mins			
Process unit	Unit 445-E, Colville	Test date	25 March 1999	
Loop details	Fan pressure loop, slow responding			
Input reference	5% of nominal (= 1 bar)	Start value	20 bar	
Data taken				
Gain	First peak	Second peak	Steady value	Ratio calculation
1	20.0249	20.0187	20.0170	$R_k = \left(\frac{20.0187 - 20.0170}{20.0249 - 20.0170} \right) = \left(\frac{0.0017}{0.0079} \right) = 0.22$
0.5	20.0125	20.0094	20.0087	$R_k = \left(\frac{20.0094 - 20.0087}{20.0125 - 20.0087} \right) = \left(\frac{0.0007}{0.0038} \right) = 0.18$
0.2	20.0050	20.0038	20.0034	$R_k = \left(\frac{20.0038 - 20.0034}{20.0050 - 20.0034} \right) = \left(\frac{0.0004}{0.0016} \right) = 0.25$

FINAL TRACE (Gain 0.2)



The zeros: solve $s^2 - 2s + 2 = 0$ and we find $s = +1.0 \pm 1j$ as the process zeros. This complex conjugate pair of zeros is in the RHP and close to the imaginary axis. This will make the control of the process very difficult by restricting the proportional gain that can be applied.

The process gain: we find $G(0) = 2/(3^3 \times 4.25) = 0.0174$. This is why the proportional gain is having little effect in achieving good steady state values as seen in the data provided in the test.

A MATLAB root locus plot is easily generated as in Figure 19.18. Clearly the dominant poles lie even closer to the imaginary axis and good time performance is going to be hard to obtain.

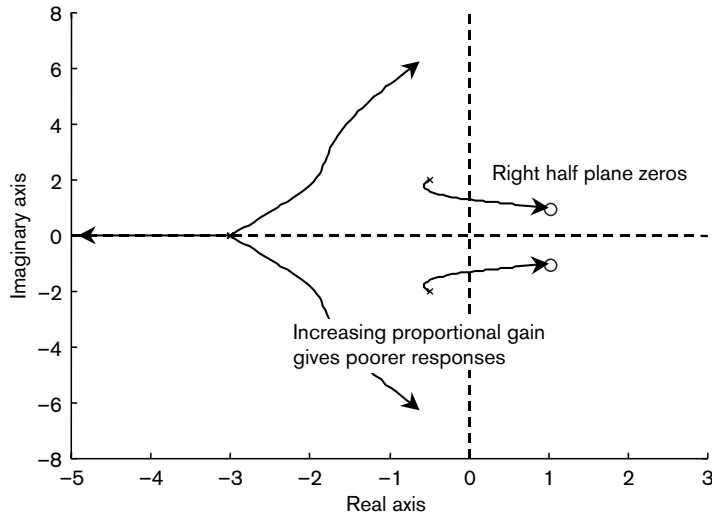


Figure 19.18 Root locus plot of system for damped oscillation tuning method.

A Simulink simulation to obtain more insight into the success of the PID tuning is given in Figure 19.19. The results of a 5% step response (Figure 19.20) show a very slow response, taking over 40 minutes to reach the desired steady state level. Notice that the final response is not quarter amplitude decay. However, we are at least pleased that the quarter amplitude decay method has

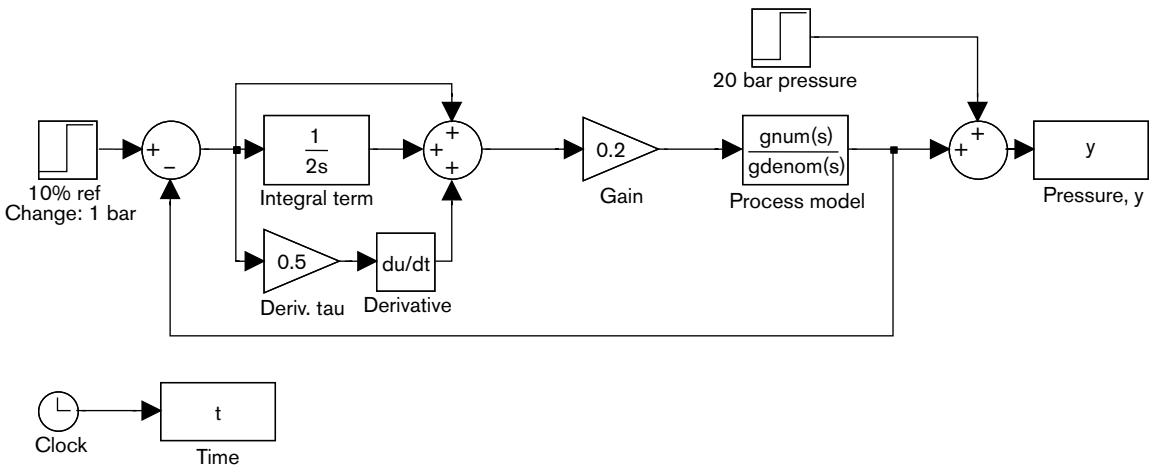


Figure 19.19 Simulation model for damped oscillation example.

produced closed-loop stable PID parameters. But we do know from the root locus plot that the performance is going to be poor. We might therefore advise a closer look at the process itself to see if the fan actuators are powerful enough for this process. Alternatively, we might look at what is causing the RHP zero to be present in this process.

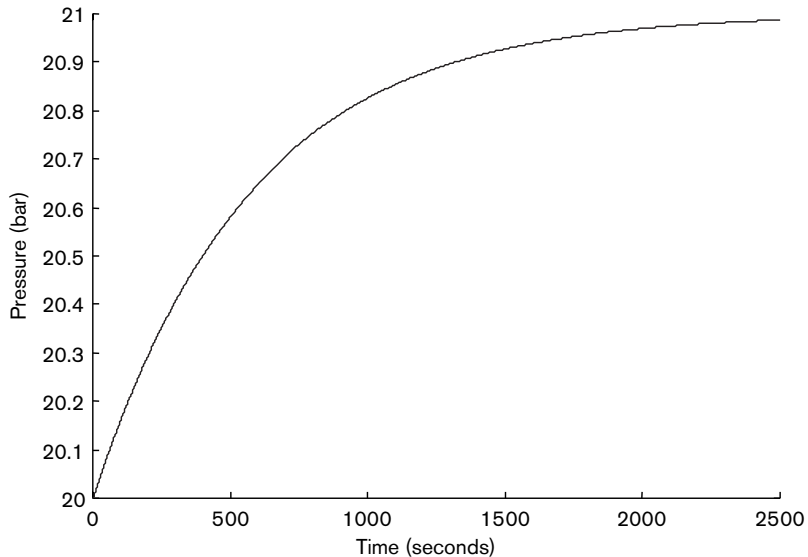


Figure 19.20 Step response (5%) for damped oscillation example.

A summary of the procedure and comments on the performance of the damped oscillation method are shown in Table 19.9.

19.5 The relay experiment

We have seen that the key motivation for the development of process control PID test routines is to have a tuning procedure which is simple, quick, reliable, repeatable and economical. We need these properties to deal with the tuning of the large number of simple PID loops present in many industrial process plants. In the 1980s, process controller technology changed from analogue to digital and the controller-to-engineer interface became more sophisticated. Electronic displays enabled small but effective trending graphs to be devised and keypads enabled the digital input of controller parameters. At that time control engineers were challenged to develop *automated* PID tuning procedures. Several technologies were pursued, expert systems and pattern recognition methods, for example. But it was a very simple and elegant solution which leads the way to today's *autotune* technology. This was the relay experiment, proposed as a solution to the automated PID problem by Karl Astrom of Sweden. The idea is shown in Figure 19.21.

Table 19.9 Summary: Damped oscillation method.

System setup	Closed-loop Proportional control only
System restrictions	Closed-loop stabilised by proportional control only
Test signal	Manually generated Step test signal Sequence of tests needed to find K_p and period, P
Test signal issues	No process disturbances during test Minimal measurement noise preferred Careful selection of increment size for the proportional gain to minimise the number of experiments and production process disruption Important to appreciate that the system may not be able to deliver perfect quarter-amplitude decay response
PID design rules	Quarter amplitude decay Only rules for PID structure available
Advantages	Simple procedure and computations Process test performance in closed loop Decaying process response obtained; avoids the verge of instability associated with sustained oscillation
Disadvantages	Procedure requires multiple tests Excessive test time possible unless careful and sensible judgement exercised Production disruptions possible

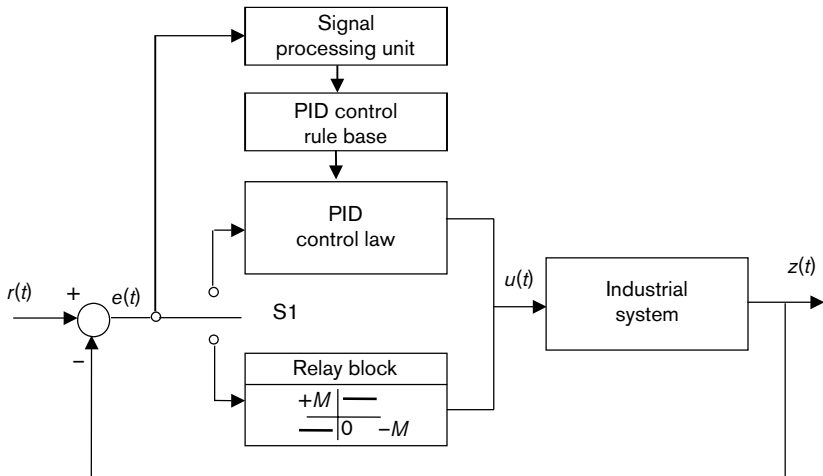


Figure 19.21 The relay setup experiment for PID tuning.

The relay is an ON–OFF control device, and this is the experimental core of the new controller unit. The characteristic of the relay is shown in Figure 19.22.

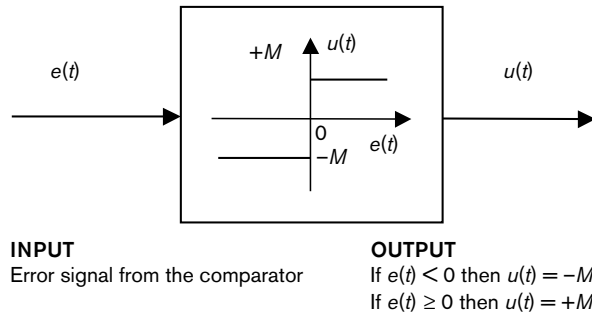


Figure 19.22 The ON–OFF relay characteristic.

These new process controller devices typically have an autotune button marked A or T (A for Autotune or T for Tune). When pressed, an automatic procedure starts whereby the closed-loop controller is switched to the ON–OFF relay and system oscillation is set up. This is monitored at the error signal $e(t)$ and used by a signal processing unit to find the ultimate gain K_u and ultimate period, P_u . This data is then used in a rule base to compute PID parameters. Finally, the parameters are passed to the actual PID controller algorithm, and control switched from the ON–OFF relay back to the PID controller. This completes the process of automated PID controller tuning.

This procedure sounds remarkably like the sustained oscillation method in that the relay experiment finds the process's ultimate data, K_u , and P_u through the use of a oscillation. Indeed we find this to be the case, since the relay sets up a stable oscillation at the phase crossover frequency, ω_{pco} . Technically we call this oscillation a limit cycle. The significant and important difference from the sustained oscillation experiment is that the relay experiment produces a *self-generated* and stable oscillation, so that the closed-loop system does *not* have to be driven to the verge of instability. The role of the signal processing component of the controller unit is to find the ultimate period, P_u , and to measure the amplitude of the oscillation, which we call A_{osc} . We find the ultimate gain, K_u , using a very simple formula:

$$K_u = \frac{4M}{\pi A_{osc}}$$

where M is the height of the ON–OFF relay controller characteristic.

The immediate aspect of the relay experiment which makes it such an attractive solution to the automated PID problem is its simplicity. It is *easy* to implement, and the computations needed to find K_u and P_u are simple in form. We can also identify some added bonuses: for example, the height of the relay, M , can be used to control the size of the process output oscillation, thereby reducing the disturbances to production operations. We also note that the relay experiment is a one-off procedure, so that we do not have to conduct a repeated sequence of test experiments. We now follow through the individual steps of this procedure.

19.5.1 The relay experimental procedure

Step 1: The process setup

The process must be in a quiescent condition and in closed-loop operation. During the experiment no process disturbances must occur and measurement noise should be minimal. If we find a process disturbance occurring during the relay experiment we will be able to see its effects fairly quickly. In some cases the oscillation becomes non-regular, and in more extreme cases the process oscillation will cease altogether. In this process controller technology, small modifications to the procedure have been introduced to mitigate the possible erroneous effects due to the presence of measurement noise. We do not consider these changes in our procedure.

Step 2: Apply the relay

For most situations, the relay height M is selected by trial and error, but we must select it to be sufficiently large to set up the limit cycle oscillation. We can fine-tune the amplitude of the process oscillation A_{osc} by changing the size of M . We show the ON-OFF relay closed-loop control experiment in Figure 19.23.

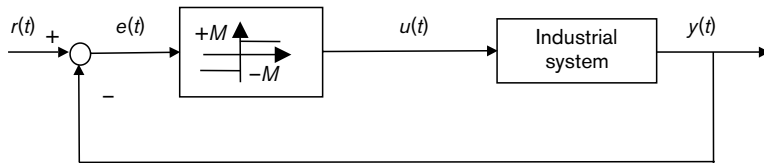


Figure 19.23 The relay experiment loop setup.

Step 3: Processing the data

We use a small pulse at the reference input to activate the oscillation, and the recorded data can be of the error, $e(t)$, or the system output, $y(t)$. We use the measured output signal $y(t)$ as shown in the trace in Figure 19.24.

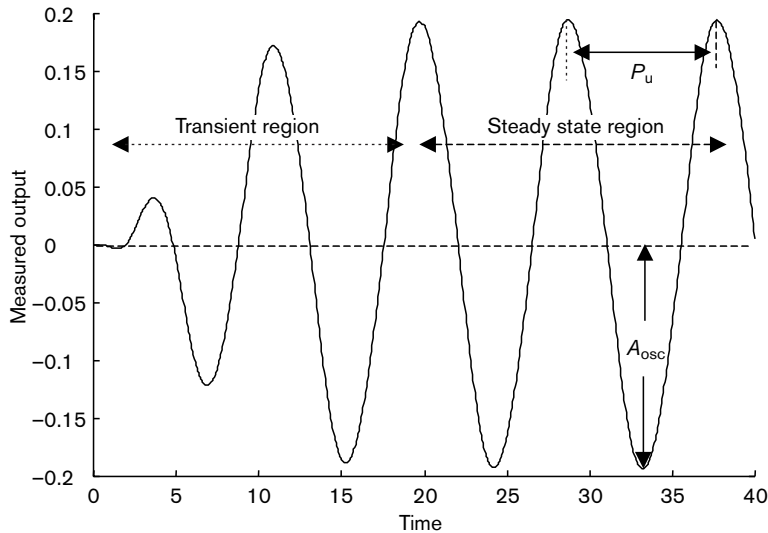


Figure 19.24 Output response from relay experiment.

From the figure we can see that the transient behaviour of the system dynamics must be allowed to settle out before attempting to process the data. We must use the steady state portion of the response for measuring the values of P_u and A_{osc} . In real process situations, we are quite likely to find that the recorded traces are contaminated by noise and possible data outliers. Despite these small problems, the measurements should be made as best as possible. It is possible that this may require taking the mean of a set of sampled measurements. Two parameters are to be measured: the ultimate period, P_u , and the oscillations amplitude, A_{osc} , as shown in Figure 19.24.

Step 4: calculating the PID parameters

When we have completed the data processing step we will have values for the following parameters:

1. the recorded relay height, M
2. the measurement of the ultimate period, P_u
3. the measurement of the oscillation amplitude, A_{osc} .

We follow this by completing one computation for the ultimate gain using the formula:

$$K_u = \frac{4M}{\pi A_{osc}}$$

The values for K_u and P_u are then used in a rule base to compute an appropriate PID controller. The original rule base due to Ziegler and Nichols can be used, but here we augment it with some phase margin rules devised by Astrom in 1982 (Table 19.10).

Table 19.10 PID controller parameter rule base.

Controller structure	Performance criterion	Proportional gain, K_p	Integral time constant, τ_i	Derivative time constant, τ_d
Case (i) P	¼ decay	$0.5K_u$	–	–
Case (ii) PI	¼ decay	$0.45K_u$	$0.833P_u$	–
Case (iii) PID	¼ decay	$0.6K_u$	$0.5P_u$	$0.125P_u$
Case (iv) PID	$\phi_{PM} = 30^\circ$	$0.87K_u$	$0.55P_u$	$0.14P_u$
Case (v) PID	$\phi_{PM} = 45^\circ$	$0.71K_u$	$0.77P_u$	$0.20P_u$
Case (vi) PID	$\phi_{PM} = 60^\circ$	$0.50K_u$	$1.29P_u$	$0.30P_u$

Step 5: implement and assess controller

We would determine the structure of the appropriate PID controller from the nature of the application and the type of performance required and implement the structure using the usual K_p , τ_i , τ_d industrial form given by

$$u(t) = K_p \left[e(t) + \frac{1}{\tau_i} \int_0^t e(\tau) d\tau + \tau_d \frac{de}{dt} \right]$$

As before, we would conduct an assessment of the performance of the design using a set of real-time tests or an analysis of real-time data. We would be seeking to see how the controller copes with significant plant events, such as reference changes, process disturbances or measurement noise.

Problem At the Clyde Power company, the difficulties of controlling the fan pressure system had long been known. A new graduate engineer determined that new, more powerful fan drive motors were needed, and that better locations for the sensors could be found. As there was some uncertainty about the process model, the engineer decided to introduce relay experiment tuning and use a phase margin PID design law. In the commissioning the engineer devised a new company data sheet for processing by the Instrumentation and Controls Department. The experimental data collected is shown opposite.

- (a) Examine the experimental relay trace obtained and use the 45° phase margin rule to design a PID control law.
- (b) The engineer has produced a transfer function model for the process as

$$G(s) = \frac{20(s^2 + 2s + 2)}{(s + 3)^3(s^2 + s + 4.25)}$$

Use a Simulink simulation to obtain more insight of the success of the PID tuning.

- (c) Use root locus analysis to see the inherent difficulties of the new process.

Solution (a) We can see from the experimental trace that the required data have been taken after the period of the initial transient. These data can be tabulated in two tables: one for the experimental results, and one for the calculation of the PID parameters.

Table 19.11 Phase margin PID controller tuning rule.

(a)

Recorded relay height, M	Ultimate period, P_u	Oscillation amplitude, A_{osc}
$M = 0.5$	$P_u = 1.38$	$A_{osc} = 0.1$
Ultimate gain calculation	$K_u = \frac{4M}{\pi A_{osc}} = \left(\frac{4 \times 0.5}{\pi \times 0.1}\right) = 6.4$	

(b)

Controller structure	Performance criterion	Proportional gain, K_p	Integral time constant, τ_i	Derivative time constant, τ_d
Case (v) PID	$\phi_{PM} = 45^\circ$	$0.71K_u$	$0.77P_u$	$0.20P_u$
Calculation		K_p	$\tau_i = 1.1$	$\tau_d = 0.28$

- (b) We use the transfer function model for the process in a Simulink simulation to obtain more insight into the success of the PID tuning. The Simulink simulation is shown in Figure 19.25 and the output response to a 5% step change in input is given in Figure 19.26. We can see that the new process modifications have increased the speed of response from over 2500

Test sheet	Power for Scotland's industry		Clyde Power plc Glasgow, Scotland
Position	Section Leader	Authorised By	R. MacSonne
Comments	Measurement free of noise. Relay Experiment performed. Pulse test. Peak-to-peak measurements taken after 7 s. Relay height = 0.5		
Process unit	Unit 445-E, Colville	Test date	5 January 2000
Loop details	Fan pressure loop		
Input reference	5% of nominal	Start value	20 bar
Data taken from trace			
Time	Peak	Amplitude	Period
9.23	20.1	0.1	1.35
10.58	20.1	0.1	1.41
11.99	20.1	0.1	Mean = 1.38 s
TRACE			

seconds to about 6 seconds. This is a great improvement. We can also see that the inverse system response has also disappeared, since there are no RHP zeros now in the new system. We must remember that these rule-based PID design methods have a success which depends on the degree of model match between the actual system and the assumed system. Model mismatch will be likely to increase as the system becomes more complicated.

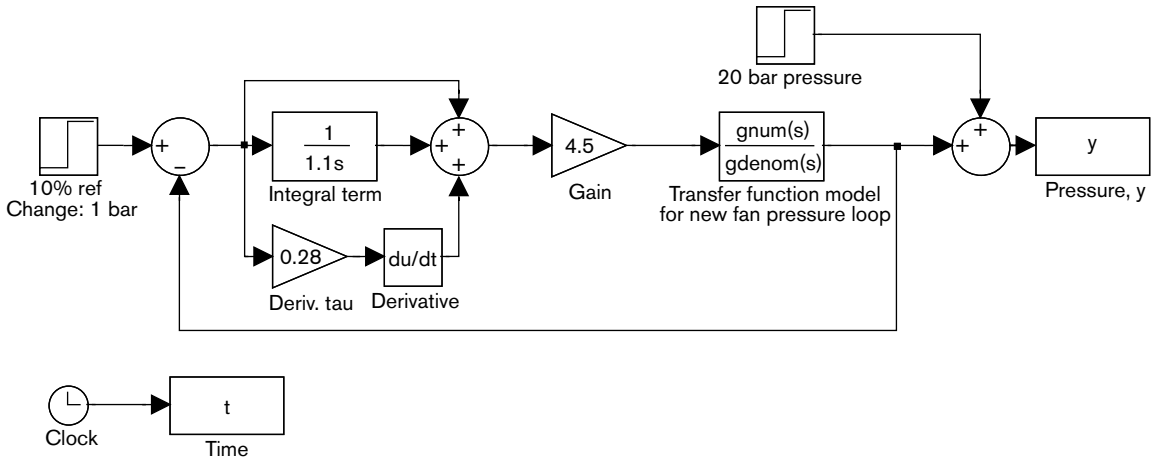


Figure 19.25 Simulink model for PID control of new fan pressure loop.

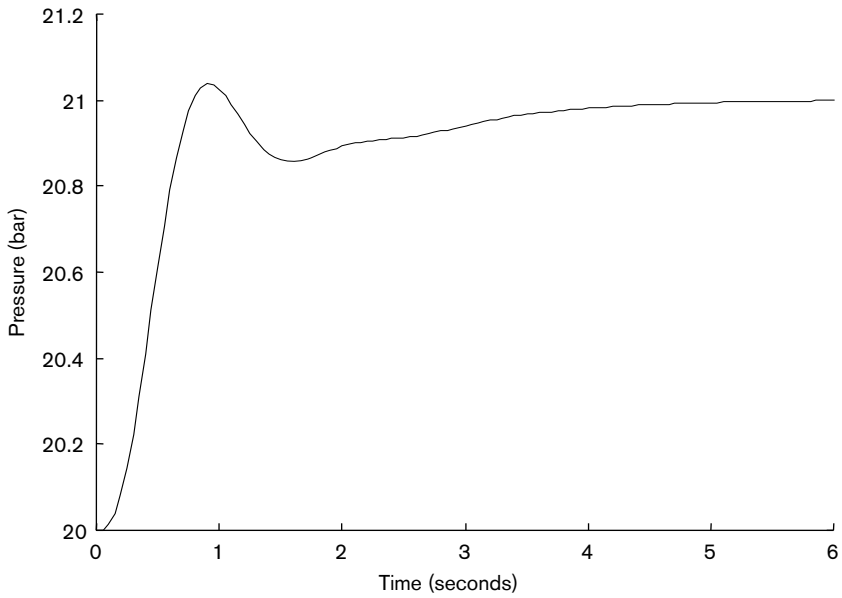


Figure 19.26 Output response under PID control with parameters from relay experiment.

- (c) To use root locus analysis to see the inherent difficulties of the new process we use a simple MATLAB rlocus command. For the new fan pressure system:

$$G(s) = \frac{20(s^2 + 2s + 2)}{(s + 3)^3(s^2 + s + 4.25)}$$

we have:

The open-loop poles: solve $(s + 3)^3(s^2 + s + 4.25) = 0$ and we find $s = -3, -3, -3$ and $s = -0.5 \pm 2j$ as the open-loop poles. This complex conjugate pair of poles is close to the imaginary axis and will give rise to a very underdamped type of response.

The zeros: solve $s^2 + 2s + 2 = 0$ and we find $s = -1.0 \pm 1j$ as the process zeros. This complex conjugate pair of zeros is close to the imaginary axis. As the proportional gain is wound up this pair of zeros will attract a pair of closed-loop poles. This will limit the speed of response.

The process gain: we find $G(0) = (20 \times 2)/(3^3 \times 4.25) = 0.348$. This is still low, but is an improvement on the previous fan pressure system, where this gain was 0.0174.

Using the simple root locus command with the given transfer function we obtain the root locus plot in Figure 19.27.

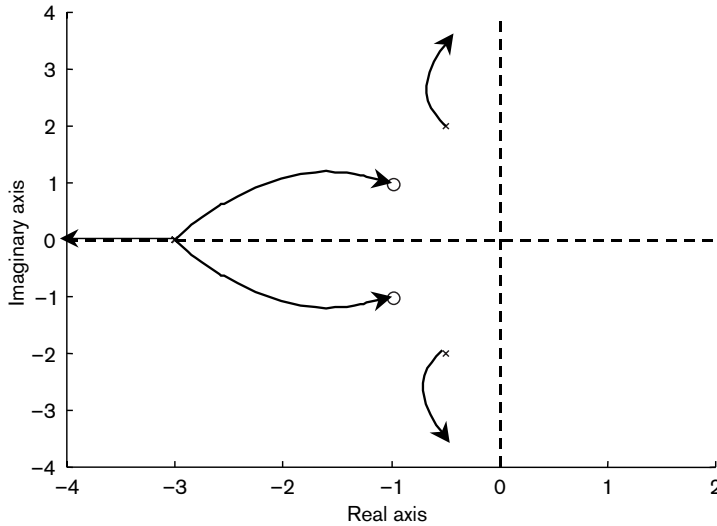


Figure 19.27 Root locus plot for fan pressure system.

While we can see that the closed-loop stability situation is much better than the old fan pressure system, we are still prevented from achieving high gain performance. The LHP zeros are attracting two closed-loop poles, and two closed-loop poles appear to go off into the right RHP as the proportional gain increases without bound.

Table 19.12 gives a summary of the relay tuning configuration and the considerations that should be made when applying it to a process.

19.6 Conclusions; or is the PID tuning problem solved?

The change from analogue to digital technology over the past twenty years was the enabling factor in the evolution of the PID autotune culture. This was the catalyst which led the end-user in the process industries to demand something better than the existing manual procedures, like the reaction curve method. We have worked with students and talked to industrial control engineers in local industries to see whether there are still

Table 19.12 Summary table: relay experiment method.

System setup	Closed-loop configuration Use of ON–OFF relay controller Device with adjustable relay height, M
System restrictions	There are few system restrictions, but the relay experiment formula is an approximation. Accuracy needs $ G(jn\omega_{pco}) \ll G(j\omega_{pco}) $ for $n > 1$
Test signal	Automatically generated by the ON–OFF relay One experimental test is usually sufficient
Test signal issues	No process disturbances during the test, but easily identified if they occur during test Minimal process measurement noise must be present Must allow transient to settle out to reach steady state stage of response before measurements made Measurement of P_u and A_{osc} very simple to take and to automate
PID design rules	Can use existing sustained oscillation Ziegler–Nichols rules or Astrom's phase margin rules New improved rules available in the technical literature
Advantages	Closed-loop procedure involving a stable limit cycle oscillation. The method overcomes key disadvantage of the sustained oscillation method and avoids the verge of instability Test signal is self-generating and adjustable to minimise production disruption Existing and new PID rule-based available for use with the procedure The procedure works with a large class of industrial process loops The process is conceptually simple and is easily implemented using digital technology
Disadvantages	Susceptible to poor and/or incorrect performance in the presence of process disturbances and measurement noise The computational formula for K_u is an approximation, and for a few processes can give poor results

problems with the PID tuning of process loops. These surveys have shown that despite the existence of automated technology there are still difficulties in industry. One problem is that technicians and engineers still do not fully understand what PID can and cannot achieve. We hope this chapter has clarified some of these limitations.

We have also seen how automated PID tuning technology does not solve *all* problems, it simply makes the solution of a subset of control problems easier. At present, the tuning technology is a push button method and still requires a very limited amount of intervention. The next stage in the technology will be to remove the push button and simply have plug-in PID control – a completely autonomous and self-configuring PID controller! Such technology when it arrives will not mean the end of control, since we have also learnt that there are some much more difficult control problems to be solved. These problems are usually nonlinear and multivariable and involve complex, highly interactive

technological processes. Trying to solve these problems keeps the control engineer in employment, so we must not complain at the challenge as we work at these harder problems!

What we have learnt

- ✓ The general principles of manual and automated tuning procedures for PID controller tuning.
- ✓ The concepts of a simple system identification routine, and the use of a PID controller parameter rule bases of formulas.
- ✓ Some of the real-world features of these test procedures, such as process disturbances, measurement noise and data outliers.
- ✓ The routines of PID tuning were linked to process control technology and concluded with the relay experiment, which is a typical example of current autotune technology used in today's commercial controllers.

Multiple choice

- M19.1** We can find the PID parameters using the autotuning methods if:
- (a) a system model is not available
 - (b) test signals can be injected to the process
 - (c) the process is open loop stable
 - (d) all of the above

- M19.2** We cannot use the reaction curve method if:
- (a) the system has a delay time
 - (b) the system is minimum phase
 - (c) the system is unstable
 - (d) (a) and (b)

- M19.3** The sustained oscillation method can only be used with systems where their Nyquist plot:
- (a) intersects the imaginary axis
 - (b) intersects the negative real axis
 - (c) intersects the unity gain circle
 - (d) has a large gain margin

- M19.4** The relay tuning method is:
- (a) an open-loop method
 - (b) a closed-loop method
 - (c) both (a) and (b)
 - (d) neither (a) nor (b)

- M19.5** The robustness of a control system can be studied using:
- (a) the GM, PM and the peak of the sensitivity function
 - (b) the poles on the $j\omega$ axis
 - (c) the zeros on the real axis
 - (d) the steady state error

- M19.6** The control system design specification for a chemical process is described as short settling time, improved damping and zero steady state error. What control do we choose?
- (a) P
 - (b) PI
 - (c) ID
 - (d) PID

- M19.7** The PID gain tables suggested in this chapter:
- (a) guarantee closed-loop stability for any system
 - (b) do not guarantee closed-loop stability for any system
 - (c) guarantee closed-loop stability if the open-loop system is stable
 - (d) guarantee closed-loop stability if the open-loop system is unstable

M19.8 The captain of a car ferry requests your help to tune the ferry's PID autopilot while the system is in closed loop control. Which method do you suggest?

- (a) reaction curve method
- (b) relay method
- (c) sustained oscillation method
- (d) none of the above

M19.9 When a system's frequency response passes the -1 point:

- (a) the GM is 1 dB
- (b) the GM is ∞
- (c) the PM is zero
- (d) the PM is 180°

M19.10 When a closed loop system oscillates:

- (a) all the closed-loop poles are on the real axis
- (b) at least a pair of closed-loop poles are on the imaginary axis
- (c) at least a pair of the closed-loop poles is on the real axis
- (d) all closed-loop poles are in the RHP

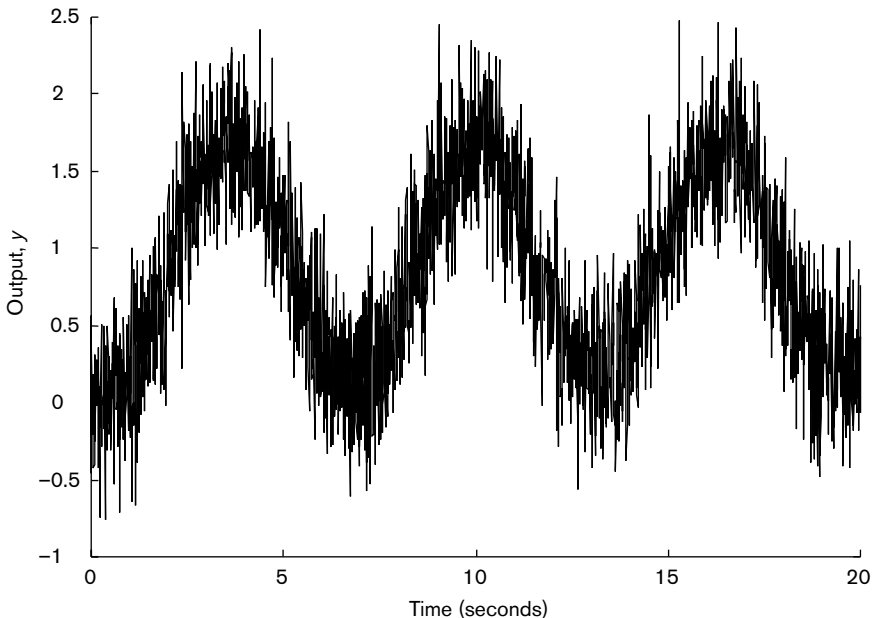
Questions: practical skills

Q19.1 There are strong links between the method of sustained oscillations and the calculation of the system's gain margin, GM . To establish the connection consider the system

$$G(s) = \frac{5}{s^3 + 3s^2 + 3s + 1}$$

- (a) Use a Bode plot or Nyquist plot to establish the gain margin for this system.
- (b) What is a physical system interpretation for this gain margin value?
- (c) Construct a simple Simulink closed-loop system simulation for the transfer function $G(s)$, with a step reference signal and a slider proportional gain block. Monitor the output signal as the proportional gain is increased.
- (d) How do the results of the experiment match up with the gain margin value calculated?

Q19.2 In a sustained oscillation experiment, the ultimate gain was found to be 9.75 and the oscillating output trace is shown below.



- (a) What decisions about the structure of the controller are derived from an assessment of the data trace?
- (b) Design a PID controller for this system using a PID rule so that the resulting response has quarter-decay overshoot. Clearly state the values for the controller coefficients of the design obtained.
- (c) On implementing the PID controller designed, some performance tests were conducted. These gave the following performance figures:

Unit step load disturbance: Peak disturbance = 1.427
5% settle time = 53.2 s

What does this test show? Use the performance figures to give two clear statements about the PID controller's performance.

Q19.3 It is claimed that the relay experiment and the method of sustained oscillations work for a large class of practical industrial systems.

- (a) Many industrial systems are modelled as a first-order transfer function model. Construct a Bode or Nyquist plot for any first-order system transfer function, $G(s) = K/(\tau s + 1)$, and determine why the method of sustained oscillation will not work for this system. Construct a simple Simulink experiment to confirm this result.
- (b) Consider the Bode plot or Nyquist plot for the transfer function

$$G(s) = \frac{Ke^{-sT_D}}{\tau s + 1}$$

using sensible parameter values. Determine whether the method of sustained oscillation will work for this system. Construct a simple Simulink experiment to confirm this result.

- (c) What features do real industrial processes have that might cause the method of sustained oscillations to actually work in practice? Give two possibilities.

Q19.4 The relay height used in a relay experiment was ± 5 , the measured oscillating output amplitude was $a = 0.189$ and the $P_u = 4.32$ s.

- (a) Use the Ziegler–Nichols table to design a PI controller.
- (b) On implementing the PI controller designed, some performance tests were conducted. These gave the following outcomes:

Unit step load disturbance: Peak disturbance = -0.954
5% settle time = 74.4 s

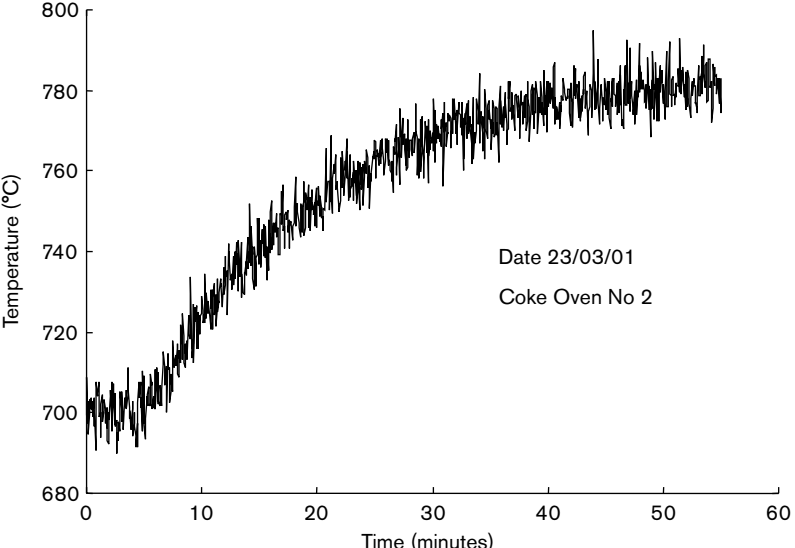
Unit step reference change: Overshoot = 110.4%
5% settle time = 40.09 s

What do these two tests attempt to show?

- (c) Give two clear assessments of the PI controller's performance.

Problems

P19.1 A control design for the coke oven at WJK Coking plant involved a reaction curve tuning exercise. The company form (below) has all the data.

WJK Coking Plant – Industrial Controls Department			
Date of test	23 March 2001	Process unit code	Coke Oven No. 2
Input level – start	50	Output level – start	700 °C
Input level – end	55 (10% step)	Output level – end	785 °C
			
Comments Coke oven on startup schedule. Maintenance completed 15 March 2001 Thermocouple burnt out. Replaced and test was successful. Noisy data, no load disturbances.			
Authorised by	M. Hamish	Date	23/03/01

- (a) Use the reaction curve method to process the data file and produce the tuning parameters for a PI controller.
- (b) Fit an appropriate first-order lag-plus-deadtime model transfer function, given by

$$G(s) = \left[\frac{Ke^{-sT}}{(\tau s + 1)} \right]$$

for which K is the d.c. gain, τ is the time constant and T is the deadtime of the process. Use a Simulink simulation to obtain some idea of the likely success of the PI tuning designed from the reaction curve method.

P19.2 A flow loop on a wastewater treatment plant unit has its transfer function estimated as

$$G_{\text{Unit}}(s) = \frac{1}{5s^3 + 15.5s^2 + 11.5s + 1}$$

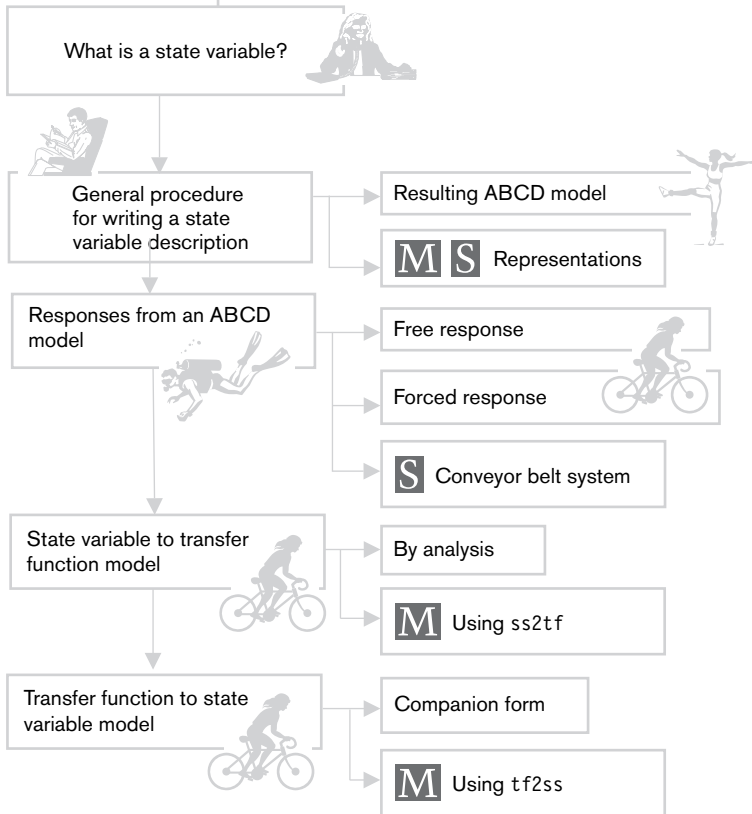
- (a) Determine the ultimate gain and ultimate period for the sustained oscillation experiment.
- (b) Use the original Ziegler–Nichols rules for PI design and implement the controller using a Simulink simulation.
- (c) Assess the reference tracking response obtained and give an opinion on the response obtained.

P19.3 A power station engineer is finding difficult operational problems on the PID control of a boiler loop. The loop had been tuned by an external contractor, and the engineer asks the junior engineer to investigate. The contractor has identified the open-loop transfer function as

$$G_{\text{BL}}(s) = \frac{s + 1.5}{5s^4 + 40s^3 + 56.5s^2 + 58.5s + 5}$$

- (a) Determine the ultimate gain and ultimate period from the relay experiment.
- (b) Use the Modified Ziegler–Nichols rules for a PID design with no overshoot and calculate the appropriate PID coefficients.
- (c) Construct a Simulink simulation and implement the controller designed.
- (d) Assess the reference tracking response obtained.
- (e) Examine an open-loop step response of the system. How does this affect your assessment?

20 Introducing a state variable description of a system



Although the representation of systems in terms of *transfer functions* is a very convenient way to describe control systems and their individual components, there is an alternative form for writing the system model. This common alternative is referred to as a *state variable* representation of the system. There are some advantages and disadvantages of using either transfer function or state variable representation; however, to appreciate the differences we must first learn what a state variable is and how to use state variable notation.

Learning objectives

- To identify a feasible set of states, inputs and outputs from a description of a system.
- To express a system in state variable form.
- To be able to convert from a state variable model to a transfer function model and to convert from transfer function model to state variable model.

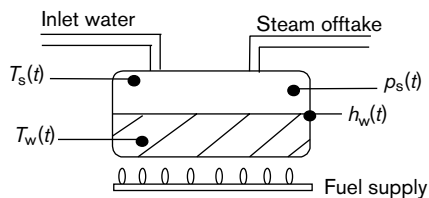
20.1 What is a state variable?

The *state* or set of system variables provides us with the status of a particular system at any instant in time. In real life, the *state* of your finances may be represented by the balance of your bank account; the *state* of your education may be represented by the number of exams or assessments that have been passed.

In the industrial example of a steam boiler system (Figure 20.1), the *state* of the boiler system is the collection of state variables which describes its status. Therefore, at time t , the steam boiler state is given by

$$\mathbf{x}_{sb}(t) = [T_s(t); p_s(t); T_w(t); h_w(t)]^T$$

where we have introduced the system *state*, $\mathbf{x}_{sb}(t)$, which is a set of *state variables*. In this example, the state variables have been chosen to be $T_s(t)$, $p_s(t)$, $T_w(t)$ and $h_w(t)$, which represent the steam temperature, the steam pressure, the water temperature and the water level respectively. Since we are usually concerned with how a system behaviour



State variables

$T_s(t)$	Temperature of steam
$p_s(t)$	Pressure of steam
$T_w(t)$	Temperature of water
$h_w(t)$	Level of water

Figure 20.1 Steam boiler.

changes with time, we find that the most useful *state variables* are often the *rate-of-change* variables within a system or combinations of these variables and their derivatives.

1. In a mechanical system, we are often interested in the position and/or velocity (rate of change of position) of a component which moves due to an input force. If we are able to write down the differential equations relating the acceleration, velocity and position, we would often let the state variables be the position and velocity within the system.
2. In an electrical circuit, the rate-of-change variables are the rate of change of current in an inductor, and the rate of change of voltage across a capacitor. The state variables can be chosen as the inductor current and capacitor voltage.
3. In a chemical engineering system, we find that the rate of change variables are often temperature, pressure and flow. These would usually become state variables in a state variable model.

What we will find out in this chapter is that although these may be common choices for state variables, there are many different state variable representations for the same system. However, although the states may be different, the inputs to the system and the defined outputs will remain the same! Therefore, in terms of input–output behaviour, all the state representations will produce equivalent responses for the same inputs. In defining a state variable model, we first define the inputs, outputs and state variables for the system.

We should note that although we use the term *state variable model*, the representation is also referred to as a *state space system*. We consider a *state space* simply as a working region where system descriptions are represented by states and the rules for the space include rules for operations on these states.

20.2 State vectors and matrices of coefficients

Before we proceed to writing down our first state variable model, we need to revise some vector–matrix notation. Why? Because sets of state variables are almost always written conveniently in terms of a vector with a system description which uses matrices of coefficients. We note that vectors and matrices will be denoted by bold characters.

20.2.1 Vectors

Consider a set of system variables: temperature, $T(t)$, pressure, $p(t)$, and flow, $q(t)$. Instead of continuing to write all three variables, we can develop a succinct vector notation.

Let $x_1(t) = T(t)$, $x_2(t) = p(t)$, $x_3(t) = q(t)$. We can define the vector $\mathbf{x}(t)$:

$$\mathbf{x}(t) = \begin{bmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \end{bmatrix} = \begin{bmatrix} T(t) \\ p(t) \\ q(t) \end{bmatrix}$$

The vector $\mathbf{x}(t)$ contains the information on the temperature, pressure and flow at each time instant, t .

For example, we find that, given an n th-order differential equation which represents the system behaviour, a state variable description would use a set of n first-order differential

equations, giving n state variables. We then use the vector notation to collect together these n state variables as a state vector, $\mathbf{x}(t) = [x_1(t), x_2(t), \dots, x_n(t)]^T$.

20.2.2 Matrices of coefficients

Often we find that the system variables are linked to the system inputs, outputs and other state variables through matrices of coefficients. Consider the following equation, where $\mathbf{x}(t) = [x_1(t), x_2(t), x_3(t)]^T$ is a function of variables $h(t)$, $m(t)$ and $p(t)$

$$x_1(t) = -3h(t) + 2m(t) + 6p(t)$$

$$x_2(t) = 4h(t) - 3m(t) + 9p(t)$$

$$x_3(t) = h(t) - p(t)$$

This can be written as

$$\begin{bmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \end{bmatrix} = \begin{bmatrix} -3 & 2 & 6 \\ 4 & -3 & 9 \\ 1 & 0 & 1 \end{bmatrix} \begin{bmatrix} h(t) \\ m(t) \\ p(t) \end{bmatrix} = \begin{bmatrix} -3 & 2 & 6 \\ 4 & -3 & 9 \\ 1 & 0 & 1 \end{bmatrix} \begin{bmatrix} u_1(t) \\ u_2(t) \\ u_3(t) \end{bmatrix}$$

which has the succinct form:

$$\mathbf{x}(t) = \begin{bmatrix} -3 & 2 & 6 \\ 4 & -3 & 9 \\ 1 & 0 & -1 \end{bmatrix} \mathbf{u}(t)$$

where we have let $\mathbf{u}(t) = [h(t), m(t), p(t)]^T$ and $\mathbf{x}(t) = [x_1(t), x_2(t), x_3(t)]^T$. This equation can also be written as

$$\mathbf{x}(t) = \mathbf{A}\mathbf{u}(t) \text{ where } \mathbf{A} = \begin{bmatrix} -3 & 2 & 6 \\ 4 & -3 & 9 \\ 1 & 0 & -1 \end{bmatrix}$$

We will use this type of vector–matrix notation whenever we meet state variable systems with more than one state.

20.3 General procedure for writing a state variable representation

We would like to form a state variable description of a general linear system (Figure 20.2). The system of Figure 20.2 can be represented in more detail by Figure 20.3.

We usually use the notation $\mathbf{u}(t)$ for inputs and $\mathbf{y}(t)$ for outputs, and we usually denote the state of the system by $\mathbf{x}(t)$. Up till now the systems analysed in this book have been represented by an *input–output* transfer function relationship between $Y(s)$ and $U(s)$. By representing the system in terms of the system states we not only have access to the inputs and outputs but we can produce a record of the internal variables – the system states – within the system. Previously we may have been restricted to knowledge of the measurable output, $y(t)$, but by forming a state variable model we can calculate the values of the measurable *and unmeasurable* system states. We find that in place of *output feedback*, which we have covered in this book up till now, we can move into the realm of *state feedback*, which can provide options which were not available previously. We investigate the control of a system

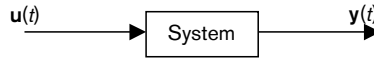


Figure 20.2 General system block diagram.

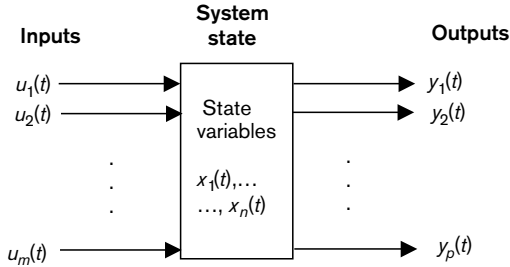


Figure 20.3 Inputs, outputs and state variables.

using state variables in Chapter 23. Another advantage of state variable notation is that this succinct notation gives us the means of writing a *multivariable* system description in a concise form. We have often met single-input single-output systems within this book, but the state variable notation, with its vector–matrix form is an immediate aid to writing multi-input multi-output system models.

We now provide a procedure for creating a state variable description of a system.

20.3.1 General procedure

(a) Define the system equations

We start by deriving the system equations, which are usually combinations of differential and algebraic equations. We consider ways in which they might be ordered in preparation for step (b).

(b) Identify the system inputs, outputs and states

Since the state variable description relies on inputs, outputs and system states, we must define these for the system and reformulate the system equations using the new state variable notation

(c) Rewrite the new system equations in standard state variable vector–matrix notation

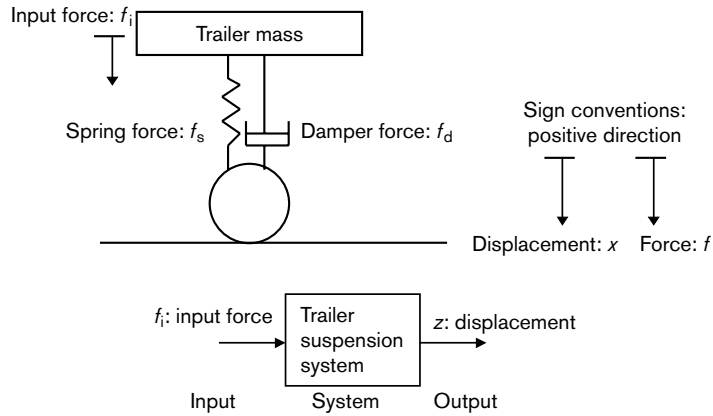
We introduce a systematic vector–matrix notation which uses four matrices: **A**, **B**, **C** and **D**. We collect together the equations to define the **A**, **B**, **C** and **D** matrices for the system.

We illustrate the procedure by applying it to the trailer suspension model of Chapter 7.

20.3.2 State variable model of trailer suspension system

(a) Define the system equations

We first examine the system and the equations which describe its dynamic behaviour. We remember that the model represents a second-order system where the position of the trailer mass is dependent on the input force, f_1 (Figure 20.4).



System	Trailer suspension system
Differential equation	$\frac{M}{K_s} \frac{d^2z}{dt^2} + \frac{B}{K_s} \frac{dz}{dt} + z(t) = \frac{1}{K_s} f_i(t)$
Parameter K_s	80 000 N/m
Parameter B	3464 N/ms ⁻¹
Parameter M	75 kg

Figure 20.4 Trailer suspension system and parameters.

The differential equation that describes the system is given by

$$\frac{M}{K_s} \frac{d^2z}{dt^2} + \frac{B}{K_s} \frac{dz}{dt} + z(t) = \frac{1}{K_s} f_i(t)$$

or

$$9.375 \times 10^{-4} \frac{d^2z}{dt^2} + 4.55 \times 10^{-3} \frac{dz}{dt} + z(t) = 1.25 \times 10^{-5} f_i(t)$$

(b) Identify the inputs, outputs and states

System inputs

We use the notation $\mathbf{u}(t)$ to represent the input signal to the system. In this example, there is one input signal, $f_i(t)$; therefore we define

$$\mathbf{u}(t) = [f_i(t)]$$

If there were more than one input signal, such as two forces $f_1(t)$ and $f_2(t)$, then the inputs would have been

$$u_1(t) = f_1(t)$$

$$u_2(t) = f_2(t)$$

and the input vector $\mathbf{u}(t)$ would have been given by

$$\mathbf{u}(t) = \begin{bmatrix} u_1(t) \\ u_2(t) \end{bmatrix} = \begin{bmatrix} f_1(t) \\ f_2(t) \end{bmatrix}$$

State variables

The system equation for the trailer suspension system is a second-order differential equation, and therefore we look for two states for the system, $x_1(t)$ and $x_2(t)$. We then aim to write two first-order differential equations for $\dot{x}_1(t)$ and $\dot{x}_2(t)$.

Let the first state be the position and the second state be the velocity

$$x_1(t) = z(t) \quad (\text{position})$$

$$x_2(t) = \frac{dz}{dt} \quad (\text{velocity})$$

From the definition of velocity we find that the first differential equation is given by

$$\dot{x}_1(t) = \frac{dz}{dt} = x_2(t).$$

For the second differential equation, we use the equation in terms of the highest derivative and rewrite it in the form

$$\frac{d^2z}{dt^2} = -\frac{B}{M} \frac{dz}{dt} - \frac{K_s}{M} z(t) + \frac{1}{M} f_i(t)$$

We now substitute in this equation the state variable notation:

$$x_1(t) = z(t) \quad (\text{position})$$

$$x_2(t) = \frac{dz}{dt} \quad (\text{velocity})$$

$$\dot{x}_2(t) = \frac{d^2z}{dt^2} \quad \text{is the derivative of the highest state (acceleration)}$$

$$\text{Input signal: } \mathbf{u}(t) = f_i(t)$$

Substituting these variables gives

$$\dot{x}_2(t) = -\frac{B}{M} x_2(t) - \frac{K_s}{M} x_1(t) + \frac{1}{M} \mathbf{u}(t)$$

This is a first-order differential equation for $\dot{x}_2(t)$. We also use the first-order equation relating states 1 and 2 (position and velocity) for the other first-order differential equation in $x_1(t)$.

Summary of state equations

$$\dot{x}_1(t) = x_2(t)$$

$$\dot{x}_2(t) = -\frac{B}{M} x_2(t) - \frac{K_s}{M} x_1(t) + \frac{1}{M} \mathbf{u}(t)$$

System outputs

We use the notation $\mathbf{y}(t)$ to represent the system's output signal. We have only one output signal, the position $z(t)$, which, in this example, is also the first state variable, $x_1(t)$:

$$\mathbf{y}(t) = z(t) = x_1(t)$$

Note: if there were more than one output signal, such as position *and* velocity ($x_1(t)$ and $x_2(t)$), then the outputs would have been

$$y_1(t) = x_1(t)$$

$$y_2(t) = x_2(t)$$

and the output vector $\mathbf{y}(t)$ would be given by

$$\mathbf{y}(t) = \begin{bmatrix} y_1(t) \\ y_2(t) \end{bmatrix}$$

(c) Vector–matrix ABCD notation

Although we have written the second-order differential equation in terms of two first-order equations, the notation is still not very concise. We adopt a matrix convention by rewriting the state variable equations in vector–matrix form, followed by the output equation.

We have two first-order equations replacing the second-order differential equation

$$\begin{aligned} \dot{x}_1(t) &= x_2(t) \\ \dot{x}_2(t) &= -\frac{B}{M}x_2(t) - \frac{K_s}{M}x_1(t) + \frac{1}{M}\mathbf{u}(t) \end{aligned}$$

We define the state vector

$$\mathbf{x}(t) = \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix}$$

with

$$\dot{\mathbf{x}}(t) = \begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \end{bmatrix}$$

and go to a vector–matrix form directly from the above equations:

$$\begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -K_s/M & -B/M \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} + \begin{bmatrix} 0 \\ 1/M \end{bmatrix} \mathbf{u}(t)$$

or

$$\dot{\mathbf{x}}(t) = \begin{bmatrix} 0 & 1 \\ -K_s/M & -B/M \end{bmatrix} \mathbf{x}(t) + \begin{bmatrix} 0 \\ 1/M \end{bmatrix} \mathbf{u}(t)$$

This can be written as

$$\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t)$$

where

$$\mathbf{A} = \begin{bmatrix} 0 & 1 \\ -K_s/M & -B/M \end{bmatrix} \quad \mathbf{B} = \begin{bmatrix} 0 \\ 1/M \end{bmatrix}$$

The output equation is given by

$$\mathbf{y}(t) = x_1(t)$$

and this can be written in terms of the state vector $\mathbf{x}(t) = [x_1(t), x_2(t)]^T$ and the control input $\mathbf{u}(t)$ as

$$\mathbf{y}(t) = [1 \quad 0] \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} = [1 \quad 0] \mathbf{x}(t)$$

However, for completeness we can write the output equation as a function of both the state $\mathbf{x}(t)$ and the input $\mathbf{u}(t)$: $\mathbf{y}(\mathbf{x}(t), \mathbf{u}(t))$. In this example, we have

$$\mathbf{y}(t) = [1 \quad 0] \mathbf{x}(t) + [0] \mathbf{u}(t)$$

This leads us to the form

$$\mathbf{y}(t) = \mathbf{C}\mathbf{x}(t) + \mathbf{D}\mathbf{u}(t)$$

where $\mathbf{C} = [1, 0]$ and $\mathbf{D} = [0]$.

We summarise the complete state variable system model description in the following.

Key result: State variable notation summary

Given a system of

m inputs

n states

r outputs

the full state space system is given by

$$\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t)$$

$$\mathbf{y}(t) = \mathbf{C}\mathbf{x}(t) + \mathbf{D}\mathbf{u}(t)$$

where \mathbf{A} (size $n \times n$) is the system matrix

\mathbf{B} (size $n \times m$) is the input matrix

\mathbf{C} (size $r \times n$) is the output matrix

\mathbf{D} (size $r \times m$) is the direct feedthrough matrix

The matrix \mathbf{D} represents any direct connections between the input and the output. However, in many simple cases, such as the trailer suspension example, the \mathbf{D} matrix is zero.

Skill section

Writing vector–matrix forms for state variable ABCD matrices

The rewriting of state variable equations in vector–matrix form will occur often in state space work. The step of identifying the number of states (n), inputs (m) and outputs (r) automatically sets up the size of the **ABCD** matrices to be filled:

$$\dot{\mathbf{x}}(t) = \mathbf{A}_{n \times n} \mathbf{x}(t) + \mathbf{B}_{n \times m} \mathbf{u}(t)$$

$$\mathbf{y}(t) = \mathbf{C}_{r \times n} \mathbf{x}(t) + \mathbf{D}_{r \times m} \mathbf{u}(t)$$

Problem For the following two systems, determine the dimensions of the **A**, **B**, **C** and **D** matrices.

System 1: Number of states is 3, number of inputs is 2, number of outputs is 1.

System 2: Number of states is 6, number of inputs is 6, number of outputs is 3.

Solution System 1:

We are given the information: $n = 3$, $m = 2$ and $r = 1$. Therefore

A: (3×3) ; **B:** (3×2) ; **C:** (1×3) ; and **D:** (1×2)

System 2:

We are given the information: $n = 6$, $m = 6$ and $r = 3$. Therefore

A: (6×6) ; **B:** (6×6) ; **C:** (3×6) ; and **D:** (3×6)

20.4 State variable diagram

We can represent the state variable model diagrammatically as in Figure 20.5. We have often used block diagrams to represent control systems. The state variable diagram has a particular format. This started from the time when digital computers were not available and analogue circuits were used to form integrating components and the gain blocks.

The forms of Figure 20.5 have been used to represent an integral component.

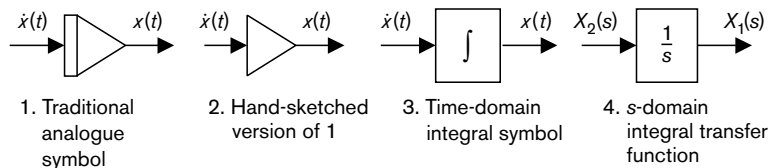


Figure 20.5 State variable block representations.

We will use blocks 3 and 4, depending on whether we are using the time domain or the s -domain in our block diagram. In the time domain, the general state variable diagram then looks like Figure 20.6. We have shown the lines connecting the **D** matrix as dotted, since for many examples in this book the **D** matrix will be zero, and these connections will not be present.

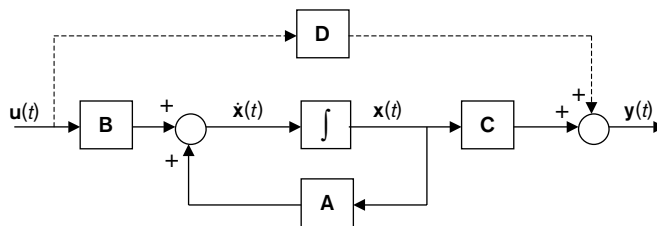


Figure 20.6 General ABCD block diagram.

The ABCD model represents a set of first-order differential equations which can be integrated to find the values of the system states. We require a set of initial conditions, $\mathbf{x}(0)$, to be able to solve the differential equations exactly. Since there are n differential equations we have an n -dimensional vector containing the n initial conditions. Very often these initial conditions are zero and we do not put them in our diagram. However, to be fully correct, we would add $\mathbf{x}(0)$ to the diagram of Figure 20.6 to give Figure 20.7.

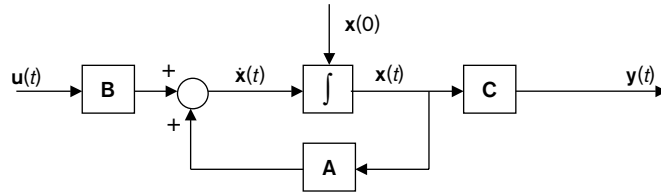


Figure 20.7 Initial conditions in a state variable system.

20.5 MATLAB–Simulink representation of state variable models

In MATLAB there is a block, called state-space (under Simulink, Continuous, Statespace), that represents a state space or state variable model.

$$\left. \begin{array}{l} \dot{x} = Ax + Bu \\ y = Cx + Du \end{array} \right\} \text{State space}$$

We can see that the block icon shows the state variable equations. It has one input port for $u(t)$ and one exit port for $y(t)$.

MATLAB requires all four matrices to be entered into the system. Even if there is no **D** matrix in the model, MATLAB will require a matrix of zeros to be entered. The **D** matrix should be the size [number of outputs \times number of inputs].

Example: Trailer suspension model

We wish to enter the following trailer suspension model into a Simulink format:

$$\mathbf{x}(t) = \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} \quad \mathbf{y}(t) = [y(t)] \quad \mathbf{u}(t) = [u(t)]$$

$$\mathbf{A} = \begin{bmatrix} 0 & 1 \\ -K_s/M & -B/M \end{bmatrix} \quad \mathbf{B} = [0 \quad 1/M] \quad \mathbf{C} = [1 \quad 0]$$

where $K_s = 80\,000 \text{ N/m}$, $B = 3464 \text{ N/m s}^{-1}$ and $M = 75 \text{ kg}$.

Since we intend using an ABCD model, we must first ensure that we know the size and values of all four matrices. The **D** matrix often has zero entries, but we must check the size of the matrix. In this example, the **D** matrix will be zero and the size is given by $r \times m$ where r and m represent the number of outputs and number of inputs respectively. The trailer suspension example has only one input and one output, giving a size for **D** of 1×1 .

We can enter now this model into a Simulink ABCD block easily. However, by using features of MATLAB we can make life easier for ourselves.

1. Direct entry of parameter values

The most obvious way to enter the matrices is by double-clicking on the ABCD-Simulink block, which reveals the data entry table (Figure 20.8).

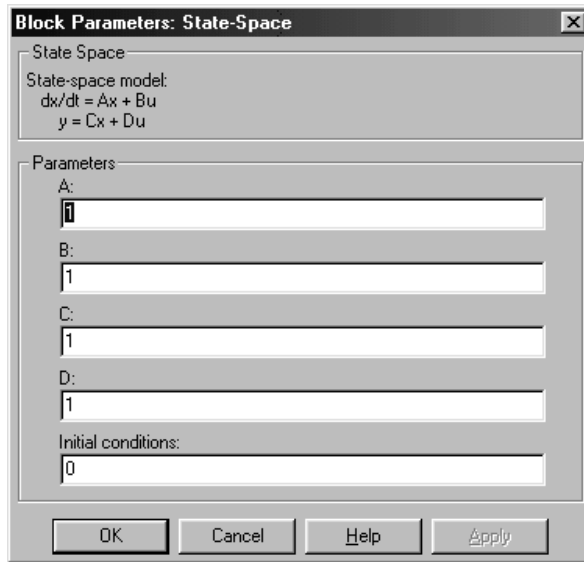


Figure 20.8 Data entry table for Simulink ABCD block.

We can then enter the matrices directly as:

A: [0 1; -1066.7 -46.187]

B: [0; 0.01333]

C: [1 0]

D: [0]

2. Direct entry of parameter calculations

We could have defined variables *Ks*, *M* and *B* in the MATLAB workspace, and entered the data as

A: [0 1; -*Ks*/*M* -*B*/*M*]

B: [0; 1/*M*]

C: [1 0]

D: [0]

3. Entry of matrix names

The Simulink data entry table could contain simply the matrix notation *A*, *B*, *C* and *D* in the corresponding entry places and we could enter the parameter data and calculations in a MATLAB M-file which we would call before running the simulation. This is in most cases preferable, since we do not need to retype data and any changes to the data are made in one place and stored on file.

The M-file would contain

```
Ks = 80 000;      % Stiffness value in N/m
B = 3464;        % Damping value in N/ms-1
M = 75;         % Mass in kg
A = [0 1; -Ks/M -B/M]; B=[0; 1/M];
C = [1 0]; D=[0];
```


We tend to develop and use state variable descriptions in their physical parameter form rather than insert the numerical data. This allows us to follow the development of the equations more clearly, and only at the final step will we substitute the parameter values in to the equations. As shown above, it is easy to use data files containing parameter values and subsequent calculations rather than to enter data directly. This allows us the flexibility to change parameters more easily in one data file rather than throughout the simulation files, as well as having a file which records the values used.

In the next problem, taken from electrical engineering, the modelling equations have been given to us; this enables even the non-electrical engineer to complete the exercise.

20.6 Example of the development of a state variable model

Problem The following set of electrical equations describe the behaviour of the currents and voltages in the electrical circuit in Figure 20.9.

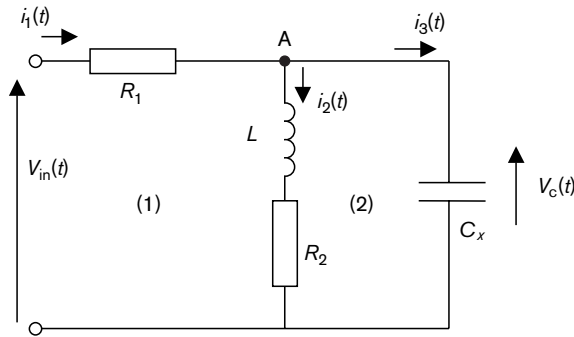


Figure 20.9 RLC circuit.

System equations

The table provides the system equations for the circuit.

Voltage round loop (1)	$V_{in}(t) = i_1(t)R_1 + L \frac{di_2(t)}{dt} + i_2(t)R_2$
Voltage across the capacitor	$C_x \frac{dV_c(t)}{dt} = i_3(t)$
Voltage round combined loops	$V_{in}(t) = i_1(t)R_1 + V_c(t)$
Current at node A	$i_1(t) = i_2(t) + i_3(t)$

Let the input $u(t)$ be the applied voltage, $V_{in}(t)$, the output signal be the voltage across the capacitor, C_x and the system states be given by

$x_1(t) = i_2(t)$: current through the inductor

$x_2(t) = V_c(t)$: voltage across the capacitor

Produce a state variable ABCD representation for this system.

Solution We follow the general procedure for creating an ABCD model: develop the system equations, define the system inputs, outputs and states and rewrite the system equations in these variables and finally, produce the ABCD model.

(a) Develop the system equations

In this problem, the modelling equations have been defined, which completes the first step of the general procedure.

(b) System inputs, outputs and states

The inputs, outputs and system states are defined in the example; however, we need to incorporate these in the system equations and find the set of first-order differential equations in the system states.

System input:

The input to the circuit is the applied voltage $V_{in}(t)$:

$$u(t) = V_{in}(t)$$

States:

The problem specification gives the two states as

$$x_1(t) = i_2(t): \quad \text{current through the inductor}$$

$$x_2(t) = V_c(t): \quad \text{voltage across the capacitor}$$

In the four system equations we see that we have two algebraic equations and two differential equations. We would like to produce a set of two first-order differential equations for this system in terms of the derivatives of the system states: $\dot{x}_1(t)$ and $\dot{x}_2(t)$. We firstly substitute in the modelling equations for $u(t)$, $x_1(t)$ and $x_2(t)$.

Replacing physical variables by state and input variables		
Physical variable equations	State variable equations	Eqn. No.
$V_{in}(t) = i_1(t)R_1 + L \frac{di_2(t)}{dt} + i_2(t)R_2$	$u(t) = i_1(t)R_1 + L \dot{x}_1(t) + x_1(t)R_2$	(1)
$C_x \frac{dV_c(t)}{dt} = i_3(t)$	$C_x \dot{x}_2(t) = i_3(t)$	(2)
$V_{in}(t) = i_1(t)R_1 + V_c(t)$	$u(t) = i_1(t)R_1 + x_2(t)$	(3)
$i_1(t) = i_2(t) + i_3(t)$	$i_1(t) = x_1(t) + i_3(t)$	(4)

We note that the variables $i_1(t)$ and $i_3(t)$ are not inputs, outputs or states. We now eliminate these from the set of equations. Substituting for $u(t)$ from (3) in equation (1) gives

$$x_2(t) = + L \dot{x}_1(t) + x_1(t)R_2$$

Rearranging gives

$$\dot{x}_1(t) = -\frac{R_2}{L} x_1(t) + \frac{1}{L} x_2(t)$$

This is one state equation and we can see the linear dependency on both $x_1(t)$ and $x_2(t)$. To find a second first-order equation we use equation (4) to replace $i_3(t)$ in (2):

$$C_x \dot{x}_2(t) = i_1(t) - x_1(t)$$

Then, by using equation (3), $i_1(t)$ can be replaced by $i_1(t) = (1/R)u(t) - (1/R)x_2(t)$ to give

$$C_x \dot{x}_2(t) = \frac{1}{R}u(t) - \frac{1}{R}x_2(t) - x_1(t)$$

Rearranging gives

$$\dot{x}_2(t) = \frac{1}{R_1 C_x} u(t) - \frac{1}{R_1 C_x} x_2(t) - \frac{1}{C_x} x_1(t)$$

This gives a set of two first-order differential equations:

$$\dot{x}_1(t) = -\frac{R_2}{L} x_1(t) + \frac{1}{L} x_2(t)$$

$$\dot{x}_2(t) = \frac{1}{R_1 C_x} u(t) - \frac{1}{R_1 C_x} x_2(t) - \frac{1}{C_x} x_1(t)$$

System output:

The output of the system is the voltage across the capacitor, $V_c(t)$, which is the second state in this example:

$$y(t) = V_{in}(t) = x_2(t)$$

(c) Vector-matrix ABCD notation

We note that the dimensions of the required ABCD form are $n = 2$, $m = 1$, $r = 1$. we therefore expect to define:

- System matrix **A**, size (2×2)
- Input matrix **B**, size (2×1)
- Output matrix **C**, size (1×2)
- Direct feedthrough matrix **D**, size (1×1)

We rewrite the set of two differential equations in ABCD form:

$$\begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \end{bmatrix} = \begin{bmatrix} -\frac{R_2}{L} & \frac{1}{L} \\ \frac{1}{C_x} & -\frac{1}{R_1 C_x} \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{R_1 C_x} \end{bmatrix} \mathbf{u}(t) \text{ with } y(t) = [0 \quad 1] \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} + [0] \mathbf{u}(t)$$

or

$$\dot{\mathbf{x}}(t) = \begin{bmatrix} -\frac{R_2}{L} & \frac{1}{L} \\ \frac{1}{C_x} & -\frac{1}{R_1 C_x} \end{bmatrix} \mathbf{x}(t) + \begin{bmatrix} 0 \\ \frac{1}{R_1 C_x} \end{bmatrix} \mathbf{u}(t) \text{ with } \mathbf{y}(t) = [0 \quad 1] \mathbf{x}(t) + [0] \mathbf{u}(t)$$

From this we can identify the **A**, **B**, **C** and **D** matrices.

$$\mathbf{A} = \begin{bmatrix} -\frac{R_2}{L} & \frac{1}{L} \\ \frac{1}{C_x} & -\frac{1}{R_1 C_x} \end{bmatrix} \quad \mathbf{B} = \begin{bmatrix} 0 \\ \frac{1}{R_1 C_x} \end{bmatrix} \quad \mathbf{C} [0 \quad 1] \quad \mathbf{D} = [0]$$

The choice of states, although a common one, was arbitrary; we could have equally well chosen the following:

$$x_1(t) = Li_2(t) \quad x_2(t) = C_x V_c(t)$$

This would have given the different state variable system

$$\begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \end{bmatrix} = \begin{bmatrix} -\frac{R_2}{L} & \frac{1}{C_x} \\ -\frac{1}{L} & -\frac{1}{R_1 C_x} \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{R_1} \end{bmatrix} \mathbf{u}(t) \quad \text{with } \mathbf{y}(t) = [0 \quad 1/C_x] \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix}$$

However, although the states of the system are not unique, both state variable systems will provide the same output $y(t)$ for the same input signal $u(t)$. The internal states will be different depending on our choice of representation, but the system output response will not change. To examine this further we look at the problem of how to calculate the state $\mathbf{x}(t)$ once we have a state variable model in ABCD form.

20.7 State variable free and forced responses

Given a system described by the ABCD state variable notation:

$$\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t)$$

$$\mathbf{y}(t) = \mathbf{C}\mathbf{x}(t) + \mathbf{D}\mathbf{u}(t)$$

we would like to find out how the state variables behave both for zero-valued input and non-zero-valued input signals. As before, when we looked at this problem for first- and second-order transfer function models, we referred to the zero-input and non-zero input problems as the *free response* and the *forced response*.

To find the solution $\mathbf{x}(t)$, we need to *solve* the differential equation; in this case, however, it is a matrix equation, though we will find that familiar integration rules still apply. The variable of integration in the following is τ . We have used this variable of integration here to be consistent with other textbooks, where we may find similar calculations. We begin from

$$\dot{\mathbf{x}}(\tau) = \mathbf{A}\mathbf{x}(\tau) + \mathbf{B}\mathbf{u}(\tau) \quad 0 \leq \tau \leq t$$

and we note that

$$\dot{\mathbf{x}}(\tau) - \mathbf{A}\mathbf{x}(\tau) = \mathbf{B}\mathbf{u}(\tau)$$

The key result we will need is:

$$\frac{d}{d\tau} [e^{-\mathbf{A}\tau} \mathbf{x}(\tau)] = -\mathbf{A}e^{-\mathbf{A}\tau} \mathbf{x}(\tau) + e^{-\mathbf{A}\tau} \dot{\mathbf{x}}(\tau) = e^{-\mathbf{A}\tau} [\dot{\mathbf{x}}(\tau) - \mathbf{A}\mathbf{x}(\tau)]$$

Now if we substitute for $[\dot{\mathbf{x}}(\tau) - \mathbf{A}\mathbf{x}(\tau)]$ from the above, we find

$$\frac{d}{d\tau} [e^{-\mathbf{A}\tau} \mathbf{x}(\tau)] = e^{-\mathbf{A}\tau} \mathbf{B}\mathbf{u}(\tau)$$

Integrating this equation over the interval $0 \leq \tau \leq t$ gives

$$[e^{-A\tau} \mathbf{x}(\tau)]_0^t = \int_0^t e^{-A\tau} \mathbf{B} \mathbf{u}(\tau) d\tau$$

Substituting in the limits of integration gives

$$e^{-At} \mathbf{x}(t) - \mathbf{x}(0) = \int_0^t e^{-A\tau} \mathbf{B} \mathbf{u}(\tau) d\tau$$

Rearranging gives the solution for $\mathbf{x}(t)$

$$\mathbf{x}(t) = e^{At} \mathbf{x}(0) + \int_0^t e^{-A(t-\tau)} \mathbf{B} \mathbf{u}(\tau) d\tau$$

At a quick glance this equation might look quite complicated, but if we pause and examine the components, we see that $\mathbf{x}(t)$ is comprised of two additive terms:

1. the term $e^{At} \mathbf{x}(0)$, which is dependent on the initial condition $\mathbf{x}(0)$; if the initial condition is zero this term will also be zero
2. the term $\int_0^t e^{-A(t-\tau)} \mathbf{B} \mathbf{u}(\tau) d\tau$ which is dependent on the input signal $u(t)$ over the interval $[0, t]$; if there is no forced input (that is $u(t) = 0$), this term will be zero

Therefore, as with the response from the transfer function model, we find that we have two components due to the free ($u(t) = 0$) and forced ($u(t) \neq 0$) responses:

$$\mathbf{x}(t) = \mathbf{x}_{\text{free}}(t) + \mathbf{x}_{\text{forced}}(t)$$

To gain some confidence in the use of this equation, we study the first-order example of the control of liquid level in a tank (Chapter 5) in the next problem.

Problem: Finding an ABCD model and examining the state behaviour

A typical liquid level system is shown in Figure 20.10.

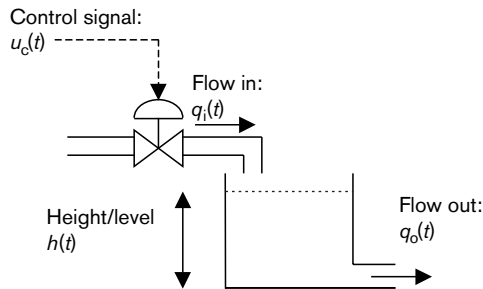


Figure 20.10 Liquid level filling system.

The modelling equation which relates the input flow, $q_i(t)$, in m^3/s , to the output height, $h(t)$, in metres, is given by

$$AR \frac{dh}{dt} + h(t) = Kq_i(t)$$

where $K = 140 \text{ s/m}^2$ and $AR = 29.3 \text{ min} = 1758 \text{ s}$ (Chapter 5).

- (a) Find a state variable model for this system and express the model in ABCD form.
 (b) Solve $\dot{x}(t) = \mathbf{A}x(t) + \mathbf{B}u(t)$ to find how the system behaves over time.

Solution We follow the procedure for identifying a state variable model.

(a) Develop system equations.

We note that the modelling equation has already been given for the liquid level system problem.

(b) Define inputs, outputs, states and write system equations in state variables.

Inputs

This system has one input, the flow $q_i(t)$: $u(t) = q_i(t)$

States

Since there is only one first-order differential equation, we look for one state variable, $x(t)$. In this case, we will let $x(t)$ be the height in the tank, then $x(t) = h(t)$.

The differential equation then becomes:

$$AR \frac{dx}{dt} + x(t) = Ku(t)$$

or

$$\dot{x}(t) = -\frac{1}{AR} x(t) + \frac{K}{AR} u(t)$$

Outputs

The output in this system is also the height of the liquid in the tank: $y(t) = h(t) = x(t)$.

(c) Rewrite state variable equations in ABCD format.

We take the differential equation and the output equation

$$\dot{x}(t) = -\frac{1}{AR} x(t) + \frac{K}{AR} u(t)$$

$$y(t) = x(t)$$

and note that

$$\dot{x}(t) = \left[-\frac{1}{AR} \right] x(t) + \left[\frac{K}{AR} \right] u(t)$$

$$y(t) = [1]x(t) + [0]u(t)$$

From which we identify:

$$\mathbf{A} = \left[-\frac{1}{AR} \right] \quad \mathbf{B} = \left[\frac{K}{AR} \right] \quad \mathbf{C} = [1] \quad \mathbf{D} = [0]$$

We now examine the free and forced responses for the one-dimensional system.

Key result: General state response

From the previous development we have derived the solution to the general state variable differential equation as

$$\mathbf{x}(t) = e^{At}\mathbf{x}(0) + \int_0^t e^{-A(t-\tau)}\mathbf{B}\mathbf{u}(\tau) d\tau$$

In our case, with one state, this reduces to

$$x(t) = \underbrace{e^{at}x(0)}_{\text{Free response}} + \underbrace{\int_0^t e^{-a(t-\tau)}bu(\tau) d\tau}_{\text{Forced response}}$$

20.7.1 Free response (no input signal)

Consider firstly the one-dimensional system with no input ($u(t) = 0$):

$$\dot{x}(t) = ax(t)$$

The solution is given by

$$x(t) = e^{at}x(0)$$

which in our liquid level example with $a = -1/AR$ becomes

$$x(t) = e^{-t/AR}x(0)$$

This represents the *free response of the system*; that is, the response of the system if there was no input and the initial condition were non-zero. In other words, if the initial height were 1.2 m, we would set $x(0) = 1.2$, and if there was no input flow and the drain tap was open, then the level would gradually fall (Figure 20.11) according to:

$$x(t) = 1.2e^{-0.000569t}$$

where $0.000569 = 1/AR = 1/1758 \text{ s}^{-1}$ and t is measured in seconds, or, equivalently, $x(t) = 1.2e^{-0.034t}$ where t is measured in minutes.

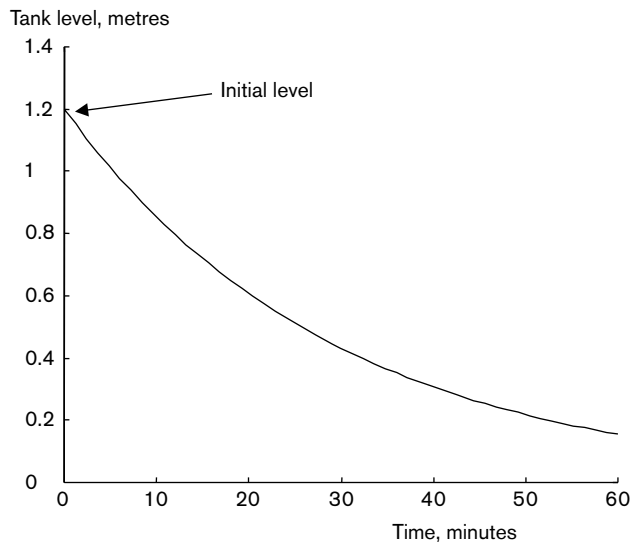


Figure 20.11 Level in tank falling.

20.7.2 Forced response (input signal applied)

This time we apply an input step signal of $u(t) = 0.005 \text{ m}^3/\text{s}$ and zero initial condition, $x(0) = 0$. This corresponds to the situation where the inflow is at the constant flow of $0.005 \text{ m}^3/\text{s}$ and the tank is empty so that $x(0) = 0$, and we solve

$$\dot{x}(t) = ax(t) + bu(t)$$

The solution is given by

$$x(t) = \int_0^t e^{a(t-\tau)} bu(\tau) d\tau$$

which in our tank example, with $a = -1/AR$, $b = K/AR$ and $u(t) = 0.005$ becomes

$$x(t) = \int_0^t e^{-(t-\tau)/AR} \frac{K}{AR} u(\tau) d\tau = \int_0^t e^{-(t-\tau)/AR} \frac{K}{AR} \times 0.005 d\tau$$

Rearranging gives

$$\begin{aligned} x(t) &= 0.005e^{-t/AR} \frac{K}{AR} \int_0^t e^{\tau/AR} d\tau \\ &= 0.005e^{-t/AR} \frac{K}{AR} [AR e^{\tau/AR}]_0^t \end{aligned}$$

Substituting in the limits of integration we find:

$$\begin{aligned} x(t) &= 0.005e^{-t/AR} \frac{K}{AR} [AR e^{t/AR} + AR] \\ &= 0.005K(1 - e^{-t/AR}) \end{aligned}$$

Substituting the values for the parameters K and AR gives:

$$x(t) = 0.7(1 - e^{-0.000569t})$$

The response $x(t)$ is shown in Figure 20.12. This shows how the tank fills up to the steady state level of 0.7 m.

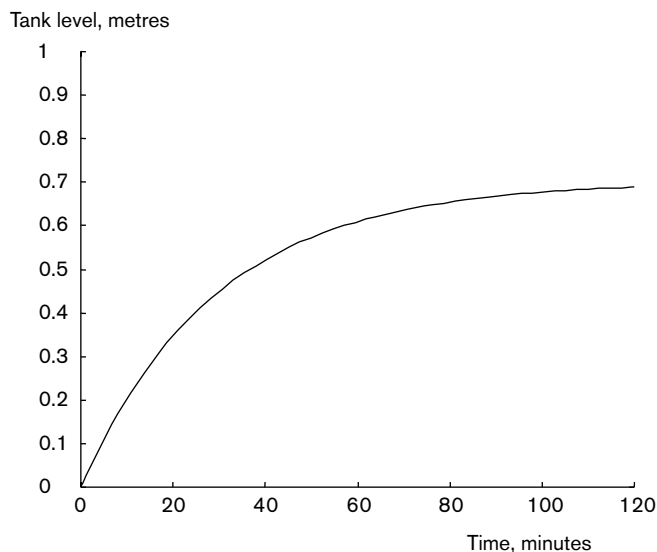


Figure 20.12 Forced response of liquid level system.

We do not usually perform the n -dimensional integration required for state variable systems; we use numerical packages such as MATLAB and Simulink to do this for us. MATLAB and Simulink have specific code which performs the integration of sets of first-order differential equations, such as those given by a state variable description.

20.8 Modelling and simulation in Simulink using an ABCD form

Problem Consider the conveyor belt example presented in Chapter 5, which has the Actuator–Process–Transducer model shown in Figure 20.13. The system equations and parameters are listed in Table 20.1.

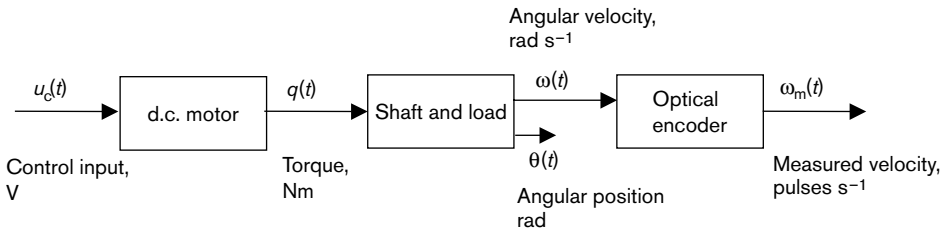


Figure 20.13 Actuator–Process–Transducer model for conveyor belt.

Table 20.1 Equations and parameters for the actuator and process systems.

Equations and parameters for the actuator and process systems	Eqn. No.
d.c. motor	
Field circuit	$V_f(t) = R_f i_f(t) + L_f \frac{di_f(t)}{dt}$ (1)
Rotational torque	$T_A(t) = (K_1 K_2 I_a) i_f(t)$ (2)
Shaft and load	
Angular velocity	$J \frac{d\omega}{dt} + B\omega(t) = T_A(t)$ (3)
Angular position	$\omega(t) = \frac{d\theta}{dt}$ (4)
Parameters (Chapter 5)	$R_f = 2\Omega$ $L_f = 0.5\text{H}$ $B = 0.5\text{N m s}^{-1}$ $J = 0.1\text{kg m}^2$ $K_1 K_2 I_a = 2\text{ N m / A (= const.)}$

- (a) Produce the actuator–process equations in the state variable ABCD format.
- (b) Use a Simulink state variable model to produce the step response of the shaft velocity, $\omega(t)$.
- (c) Investigate what would happen to the shaft velocity if the inertia, J , were increased by 50%.

Solution (a) Find an ABCD model. Once again, we follow the general procedure.

1. Develop the system equations: these are given in the example.
2. Define the inputs, outputs and states and rewrite system equations in state variables.

System input

The system input is given by the voltage $V_f(t)$. Therefore we can write

$$u(t) = V_f(t)$$

System states

The actuator–process system comprises three related first-order differential equations and several algebraic equations. We look for three *rate-of-change* variables as the system states. We rewrite equations (1), (3) and (4) in terms of the derivative

$$\begin{aligned}\frac{di_f(t)}{dt} &= -\frac{R_f}{L_f} i_f(t) + \frac{1}{L_f} V_f(t) \\ \frac{d\omega}{dt} &= -\frac{B}{J} \omega(t) + \frac{1}{J} T_A(t) \\ \frac{d\theta}{dt} &= \omega(t)\end{aligned}$$

We therefore let the state variables be

$$x_1(t) = i_f(t), \quad x_2(t) = \omega(t) \quad \text{and} \quad x_3(t) = \theta(t)$$

We can then directly rewrite the three differential equations in terms of the three states and the input, $u(t) = V_f(t)$.

$$\begin{aligned}\dot{x}_1(t) &= -\frac{R_f}{L_f} x_1(t) + \frac{1}{L_f} u(t) \\ \dot{x}_2(t) &= -\frac{B}{J} x_2(t) + \frac{1}{J} (K_1 K_2 I_a) x_1(t) \quad \text{using equation (2)} \\ \dot{x}_3(t) &= x_2(t)\end{aligned}$$

System outputs

The system has two outputs:

$$y_1(t) = \omega(t) = x_2(t)$$

$$y_2(t) = \theta(t) = x_3(t)$$

3. Rewriting the equations in ABCD format

The dimensions of the required ABCD form are $n=3$, $m=1$, $r=2$. We therefore expect to define:

System matrix **A**, size (3×3)

Input matrix **B**, size (3×1)

Output matrix **C**, size (2×3)

Direct feedthrough matrix **D**, size (2×1)

We use the vector matrix notation:

$$\begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \\ \dot{x}_3(t) \end{bmatrix} = \begin{bmatrix} -R_f & 0 & 0 \\ (K_1 K_2 I_a) / J & -B / J & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \end{bmatrix} + \begin{bmatrix} 1 / L_f \\ 0 \\ 0 \end{bmatrix} u(t)$$

$$\begin{bmatrix} y_1(t) \\ y_2(t) \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \end{bmatrix} u(t)$$

System responses

We can use the ABCD block in Simulink to simulate this system (Figure 20.14). We can see that this notation provides a very compact form of expressing system equations.

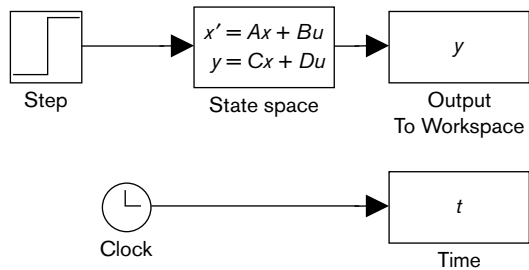


Figure 20.14 Simulink ABCD model implementation.

The parameters are given as:

$$\begin{aligned} R_f &= 2\Omega & L_f &= 0.5 \text{ H} \\ B &= 0.5 \text{ N m s}^{-1} & J &= 0.1 \text{ kg m}^2 \\ K_1 K_2 I_a &= 2 \text{ N m/A (= const.)} \end{aligned}$$

We enter the system parameters into the MATLAB workspace:

```
Rf=2; Lf=0.5; B=0.5; J=0.1; const=2;
```

We enter the **A**, **B**, **C** and **D** matrices into the state space block. We could have worked out the parameters directly and entered the numerical values, but if we wish to change any parameters, for example, to work out how sensitive a system is to one parameter, it is easier not to have entered the parameters in numerical values.

$$\begin{aligned} A &= [-Rf/Lf \ 0 \ 0 ; \ \text{const}/J \ -B/J \ 0 ; \ 0 \ 1 \ 0] \\ B &= [1/Lf ; \ 0 ; \ 0] \\ C &= [0 \ 1 \ 0 ; \ 0 \ 0 \ 1]; \\ D &= [0 ; \ 0] \end{aligned}$$

We set the maximum time to three seconds and after running the simulation we can plot the output. Remembering that the simulation output, *y*, is a vector, the command

```
plot(t,y(:,1))
```

will produce a plot of the velocity as shown in Figure 20.15.

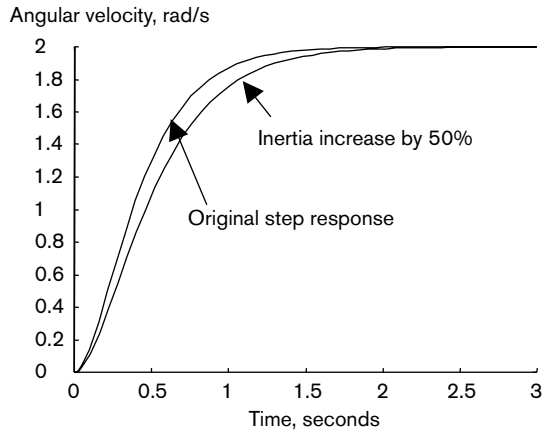


Figure 20.15 Simulation of conveyor belt velocity.

If we increase the inertia, J , by 50% and plot the velocity again (Figure 20.15) we find the speed of response has decreased, which is what we would expect for an increase in inertia.

20.9 State variable model to transfer function model

We have learnt how to represent a system in the concise state variable format. However, by so doing, we initially lose any intuition gained in the development and manipulation of transfer functions and the knowledge of poles/zeros and stability. We address these matters now and show how to change a state variable model to transfer function form and vice versa.

We start by developing the connection between the general state variable model and our Laplace transform representations of systems.

20.9.1 State variables and Laplace transform representations.

Consider the standard state variable description of a control system

$$\frac{d\mathbf{x}}{dt} = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t)$$

$$\mathbf{y}(t) = \mathbf{C}\mathbf{x}(t)$$

Taking Laplace transforms of this equation gives

$$s\mathbf{X}(s) - \mathbf{x}(0) = \mathbf{A}\mathbf{X}(s) + \mathbf{B}\mathbf{U}(s)$$

$$\mathbf{Y}(s) = \mathbf{C}\mathbf{X}(s)$$

Rearranging the expression for $\mathbf{X}(s)$ gives

$$(s\mathbf{I} - \mathbf{A})\mathbf{X}(s) = \mathbf{x}(0) + \mathbf{B}\mathbf{U}(s)$$

$$\mathbf{X}(s) = (s\mathbf{I} - \mathbf{A})^{-1}\mathbf{x}(0) + (s\mathbf{I} - \mathbf{A})^{-1}\mathbf{B}\mathbf{U}(s)$$

Taking inverse Laplace transforms of $\mathbf{X}(s)$ gives

$$\mathbf{x}(t) = \mathcal{L}^{-1}\{(s\mathbf{I} - \mathbf{A})^{-1}\}\mathbf{x}(0) + \mathcal{L}^{-1}\{(s\mathbf{I} - \mathbf{A})^{-1}\mathbf{B}\mathbf{U}(s)\}$$

If we equate this with the time domain solution

$$\mathbf{x}(t) = e^{\mathbf{A}t}\mathbf{x}(0) + \int_0^t e^{\mathbf{A}(t-\tau)}\mathbf{B}\mathbf{u}(\tau) d\tau$$

we find

$$\mathcal{L}^{-1}\{(s\mathbf{I} - \mathbf{A})^{-1}\} = e^{\mathbf{A}t} \quad \text{and} \quad \mathcal{L}^{-1}\{(s\mathbf{I} - \mathbf{A})^{-1}\mathbf{B}\mathbf{U}(s)\} = \int_0^t e^{\mathbf{A}(t-\tau)}\mathbf{B}\mathbf{u}(\tau) d\tau$$

Key result: Laplace transform model from ABCD form

The output equation is given by

$$\mathbf{Y}(s) = \mathbf{C}\mathbf{X}(s) = \mathbf{C}(s\mathbf{I} - \mathbf{A})^{-1}\mathbf{B}\mathbf{U}(s) + \mathbf{C}(s\mathbf{I} - \mathbf{A})^{-1}\mathbf{x}(0)$$

If we set the input conditions to zero, $\mathbf{x}(0) = 0$, we note that the output $\mathbf{Y}(s)$ is related to the input $\mathbf{U}(s)$ as follows:

$$\mathbf{Y}(s) = \mathbf{C}(s\mathbf{I} - \mathbf{A})^{-1}\mathbf{B}\mathbf{U}(s) = \mathbf{G}(s)\mathbf{U}(s)$$

where

$$\mathbf{G}(s) = \mathbf{C}(s\mathbf{I} - \mathbf{A})^{-1}\mathbf{B}$$

This is a transfer function matrix which represents the input–output transfer functions. For a single-input single-output system, the matrix $\mathbf{G}(s)$ would contain only one transfer function. Many of our examples up till now have been single-input single-output systems.

Problem: state space system to transfer function by analysis

Convert the following state space system to transfer function form given that the initial conditions are zero.

$$\begin{aligned} \dot{\mathbf{x}}(t) &= \begin{bmatrix} 2.2 & 1 \\ 3 & 6.5 \end{bmatrix} \mathbf{x}(t) + \begin{bmatrix} 3 \\ 1 \end{bmatrix} \mathbf{u}(t) \\ \dot{\mathbf{y}}(t) &= [2 \quad 1]\mathbf{x}(t) \end{aligned}$$

Solution We remember that the Laplace transfer function form is given by

$$\mathbf{Y}(s) = \mathbf{C}(s\mathbf{I} - \mathbf{A})^{-1}\mathbf{B}\mathbf{U}(s)$$

Therefore

$$\begin{aligned} \mathbf{Y}(s) &= [2 \quad 1] \left(\begin{bmatrix} s & 0 \\ 0 & s \end{bmatrix} - \begin{bmatrix} 2.2 & 1 \\ 3 & 6.5 \end{bmatrix} \right)^{-1} \begin{bmatrix} 3 \\ 1 \end{bmatrix} \mathbf{U}(s) \\ &= [2 \quad 1] \begin{bmatrix} s-2.2 & -1 \\ -3 & s-6.5 \end{bmatrix}^{-1} \begin{bmatrix} 3 \\ 1 \end{bmatrix} \mathbf{U}(s) \end{aligned}$$

We remember that for a 2×2 matrix, the inverse is easily found by

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix}^{-1} = \frac{1}{ad-bc} \begin{bmatrix} d & -b \\ -c & a \end{bmatrix}$$

Applying this gives

$$\begin{aligned} \mathbf{Y}(s) &= [2 \quad 1] \frac{1}{(s-2.2)(s-6.5)-3} \begin{bmatrix} s-6.5 & 1 \\ 3 & s-2.2 \end{bmatrix} \begin{bmatrix} 3 \\ 1 \end{bmatrix} \mathbf{U}(s) \\ &= [2 \quad 1] \frac{1}{(s^2-8.7s+11.3)} \begin{bmatrix} 3s-18.5 \\ s+6.8 \end{bmatrix} \mathbf{U}(s) \\ &= \frac{7s-30.2}{(s^2-8.7s+11.3)} \mathbf{U}(s) = \mathbf{G}(s)\mathbf{U}(s) \end{aligned}$$

We note that the resulting transfer function gives us the 'input-output' representation. We could implement this transfer function in a Simulink model but we would not have access to the internal states within the system. One of the advantages of using state variable models is the ability to access not only the outputs, $y_1(t)$, $y_2(t)$, ..., but to access the internal variables, $x_1(t)$, $x_2(t)$, These variables may not be measured, and therefore by using the state variable model we gain added information about the system.

It is also important to realise that although we can have many different state variable models of the same system, depending on our choice of system states, the input-output transfer function model will remain the same.

We have demonstrated how to produce a transfer function model from a state variable model through Laplace transform analysis. Obviously, if the system were more than 2×2 , it would become very difficult to solve by hand and the advantage of using computer packages becomes readily apparent. We now illustrate how to use a MATLAB command to effect the conversion from state variable model to transfer function.

20.10 MATLAB function ss2tf: state space to transfer function conversion

We note that the function `ss2tf` (and its counterpart `tf2ss`) are functions in the control toolbox in MATLAB. The form of the expression for `ss2tf` is given by

$$[\text{num}, \text{den}] = \text{ss2tf}(\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}, \text{iu})$$

The inputs are the state variable matrices \mathbf{A} , \mathbf{B} , \mathbf{C} and \mathbf{D} . If there is no \mathbf{D} matrix in the model, then a \mathbf{D} matrix must be created with zeros. The size of the \mathbf{D} matrix is $r \times m$, where r is the number of outputs and m is the number of inputs.

The input `iu` is the input we are interested in, that is input number 1 or 2, etc. If we need to find the transfer function matrix for all inputs we would have to enter the command several times, changing the value of `iu`.

The output is given in the matrices `num` and `den`. The denominator of each transfer function with a particular input will be the same; therefore `den` is a vector which contains the coefficients of the denominator polynomial. `num` contains rows of the numerator coefficients. The number of rows will be the same as the number of outputs of the system.

Example We now use MATLAB to repeat the same conversion from the given state variable model to transfer function form where the initial conditions are zero.

$$\dot{\mathbf{x}}(t) = \begin{bmatrix} 2.2 & 1 \\ 3 & 6.5 \end{bmatrix} \mathbf{x}(t) + \begin{bmatrix} 3 \\ 1 \end{bmatrix} \mathbf{u}(t)$$

$$\dot{\mathbf{y}}(t) = [2 \ 1] \mathbf{x}(t)$$

We use the MATLAB command: `[num,den] = ss2tf(A,B,C,D,iu)`. We enter
`A=[2.2 1; 3 6.5]; B=[3;1]; C=[2 1];`

We must enter a D matrix with size of 'outputs by inputs' – in this example 1×1 :

$$D=[0];$$

We only have one input so `iu` will be 1.

$$[\text{num,den}] = \text{ss2tf}(A,B,C,D,1)$$

MATLAB result:

```
num =
    0 7.0000 -30.2000
den =
    1.0000 -8.7000 11.3000
```

This is interpreted as

$$G(s) = \frac{0s^2 + 7s - 30.2}{1s^2 - 8.7s + 11.3}$$

Problem: conversion of multi-input single-output model

Convert the following state variable model which has two inputs to transfer function format. Assume zero initial conditions.

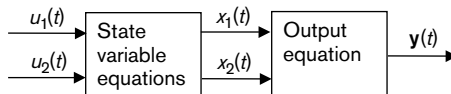
$$\dot{\mathbf{x}}(t) = \begin{bmatrix} -6.3 & 3 \\ 0.5 & -5.4 \end{bmatrix} \mathbf{x}(t) + \begin{bmatrix} 1 & 0.1 \\ 0.2 & 1 \end{bmatrix} \mathbf{u}(t)$$

$$\dot{\mathbf{y}}(t) = [1 \ 0] \mathbf{x}(t)$$

Solution We enter the **A**, **B**, **C** and **D** matrices as follows:

$$A=[-6.3 \ 3; 0.5 \ -5.4]; B=[1 \ 0.1; 0.2 \ 1]; C=[1 \ 0]; D=[0 \ 0];$$

We note that the system has the form



In our problem, we will find a transfer function model of the form:

$$\begin{bmatrix} X_1(s) \\ X_2(s) \end{bmatrix} = \begin{bmatrix} G_{11}(s) & G_{12}(s) \\ G_{21}(s) & G_{22}(s) \end{bmatrix} \begin{bmatrix} U_1(s) \\ U_2(s) \end{bmatrix}$$

$$Y(s) = [1 \ 0] \begin{bmatrix} X_1(s) \\ X_2(s) \end{bmatrix}$$

Multiplying out the transfer functions gives

$$Y(s) = [1 \ 0] \begin{bmatrix} X_1(s) \\ X_2(s) \end{bmatrix} = X_1(s) = [G_{11}(s) \ G_{12}(s)] \begin{bmatrix} U_1(s) \\ U_2(s) \end{bmatrix} = G_{11}(s)U_1(s) + G_{12}(s)U_2(s)$$

The output $Y(s)$ will depend on each input $U_1(s)$ and $U_2(s)$ through the transfer functions $G_{11}(s)$ and $G_{12}(s)$ respectively. The function `ss2tf` only provides one transfer function at each call of the function, so we must use it twice:

```
[num1,den1]=ss2tf(A,B,C,D,1)
num1 =
    0 1 6
den1 =
    1.0000 11.7000 32.5200
```

This gives

$$G_{11}(s) = \frac{s+6}{s^2+11.7s+35.52}$$

Applying the function again for input 2 gives

```
[num2,den2]=ss2tf(A,B,C,D,2)
num2 =
    0 0.1000 3.5400
den2 =
    1.0000 11.7000 32.5200
```

We find that

$$G_{12}(s) = \frac{s+3.54}{s^2+11.7s+35.52}$$

Therefore

$$Y(s) = \frac{s+6}{s^2+11.7s+35.52} U_1(s) + \frac{s+3.54}{s^2+11.7s+35.52} U_2(s)$$

We note that the denominator in each case remains the same, since it holds the key to the system stability. The numerator of each transfer function does alter. This will become clearer in Chapter 22 when we look at the analysis of state variable models.

20.11 From transfer function to state variable model

We have found that a state space representation for a system is not unique. We emphasise this again here when we present one method for determining a state space system from a transfer function description. We illustrate this with a particular example before listing the general case.

The following transfer function describes the dynamics of an actuator. We see that there is one real pole and a pair of complex poles and that the system is of third order.

$$Y(s) = \frac{1}{(s+4)(s^2+4s+7)} U(s) = \frac{1}{s^3+8s^2+23s+28} U(s)$$

If we write

$$(s^3 + 8s^2 + 23s + 28)Y(s) = U(s)$$

we can see that this is equivalent to the differential equation given by

$$\frac{d^3y(t)}{dt^3} + 8 \frac{d^2y(t)}{dt^2} + 23 \frac{dy}{dt} + 28y(t) = u(t)$$

We let the first state, $x_1(t)$ be equivalent to the output $y(t)$, the second state equal its derivative, the third state equal the next derivative and so on (Figure 20.16).

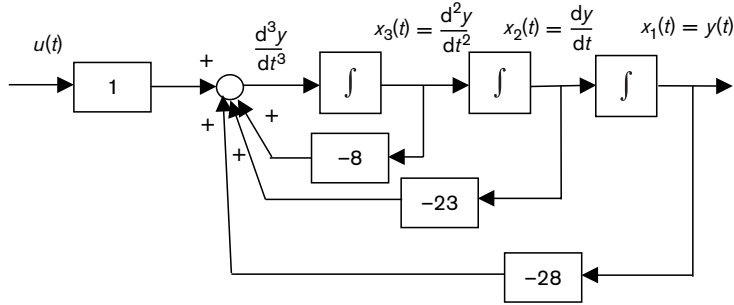


Figure 20.16 State variable model.

Therefore we can write

$$\begin{aligned} x_1(t) &= y(t) \\ \frac{dx_1}{dt} &= x_2(t) = \frac{dy}{dt} \\ \frac{dx_2}{dt} &= x_3(t) = \frac{d^2y}{dt^2} \end{aligned}$$

The full differential equation above can be rewritten in terms of its highest derivative:

$$\frac{d^3y(t)}{dt^3} = -8 \frac{d^2y(t)}{dt^2} - 23 \frac{dy}{dt} - 28y(t) + u(t)$$

and the state variable notation introduced to give

$$\frac{dx_3(t)}{dt} = -8x_3(t) - 23x_2(t) - 28x_1(t) + u(t)$$

This gives us the following state variable system

$$\begin{aligned} \begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \\ \dot{x}_3(t) \end{bmatrix} &= \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -28 & -23 & -8 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} u(t) \\ y(t) &= [1 \ 0 \ 0] \mathbf{x}(t) \end{aligned}$$

This is called the **companion** or **phase variable** form of the state variable model. The last row of the **A** matrix comprises all of the coefficients of the denominator of the transfer function and the diagonal row above the main diagonal of the **A** matrix always contains unity terms. It is therefore very easy to write this state variable description down directly. The disadvantage with this form is that the state variables do not always represent physical quantities.

In the above example, the numerator was simply a '1'. We would like to know how to deal with situations where the numerator is a polynomial in s .

For illustration we consider the same transfer function as above but add a lead term in the numerator; this gives

$$Y(s) = \frac{28(2s+1)}{(s+4)(s^2+4s+7)} U(s) = \frac{(56s+28)}{(s^3+8s^2+23s+28)} U(s)$$

We can rewrite this expression as

$$\frac{Y(s)}{U(s)} = \frac{X(s)}{U(s)} \frac{Y(s)}{X(s)}$$

where

$$X(s) = \frac{1}{(s^3+8s^2+23s+28)} U(s)$$

and

$$Y(s) = (56s + 28)X(s)$$

The transfer function from $U(s)$ to $X(s)$ is similar to the example above. However, the additional equation is for $Y(s)$. This can be converted back to a differential equation to give

$$y(t) = 56 \frac{dx}{dt} + 28x(t)$$

If we include this in the state variable diagram of Figure 20.16 we have the new diagram as shown in Figure 20.17.

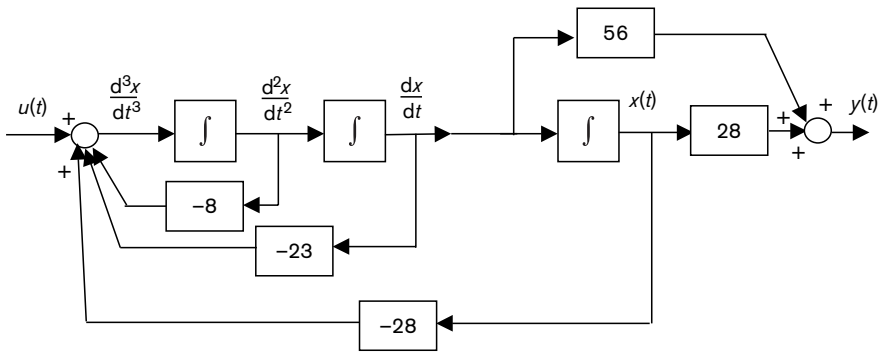


Figure 20.17 State variable model for transfer function description.

Using the state variable notation

$$y(t) = 56x_2(t) + 28x_1(t)$$

Therefore in matrix form we have the output equation as

$$y(t) = [28 \ 56 \ 0]\mathbf{x}(t)$$

with the state equations given as before:

$$\begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \\ \dot{x}_3(t) \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -28 & -23 & -8 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} u(t)$$

We have worked through the transformation from transfer function to state variable model for a particular transfer function. For completeness, we now summarise the procedure for application to a general transfer function model.

20.11.1 From a general transfer function to state variable model

Consider the general transfer function given by the following:

$$Y(s) = \frac{1}{s^n + a_{n-1}s^{n-1} + \dots + a_1s + a_0} U(s)$$

This is equivalent to the differential equation given by

$$\frac{d^n y(t)}{dt^n} + a_{n-1} \frac{d^{n-1}y(t)}{dt^{n-1}} + \dots + a_1 \frac{dy}{dt} + a_0 y(t) = u(t)$$

We let the first state, $x_1(t)$, be equivalent to the output $y(t)$, the second state equal its derivative, the third state equal the next derivative and so on (Figure 20.18).

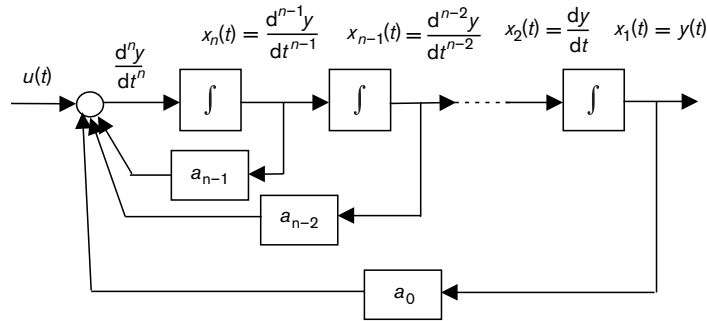


Figure 20.18 State variable model.

We can write

$$\begin{aligned} x_1(t) &= y(t) \\ \frac{dx_1}{dt} &= x_2(t) = \frac{dy}{dt} \\ \frac{dx_2}{dt} &= x_3(t) = \frac{d^2y}{dt^2} \\ \frac{dx_3}{dt} &= x_4(t) = \frac{d^3y}{dt^3} \\ &\dots \\ \frac{dx_{n-1}}{dt} &= x_n(t) = \frac{d^{n-1}y}{dt^{n-1}} \end{aligned}$$

The full output differential equation can be rewritten in terms of its highest derivative:

$$\frac{d^n y(t)}{dt^n} = -a_{n-1} \frac{d^{n-1}y(t)}{dt^{n-1}} - \dots - a_1 \frac{dy}{dt} - a_0 y(t) + u(t)$$

and the state variable notation introduced to give

$$\frac{dx_n(t)}{dt} = -a_{n-1}x_n(t) - a_{n-2}x_{n-1}(t) - \dots - a_1x_2(t) - a_0x_1(t) + u(t)$$

This gives us the following state variable system:

$$\begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \\ \dot{x}_3(t) \\ \vdots \\ \dot{x}_{n-1}(t) \\ \dot{x}_n(t) \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & \dots & 0 & 0 \\ 0 & 0 & 1 & \dots & 0 & 0 \\ 0 & 0 & 0 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & 0 & 1 \\ -a_0 & -a_1 & -a_2 & \dots & -a_{n-2} & -a_{n-1} \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \\ \vdots \\ x_{n-1}(t) \\ x_n(t) \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} u(t)$$

$$y(t) = [1 \ 0 \ 0 \ \dots \ 0 \ 0] \mathbf{x}(t)$$

If we add a polynomial in s to the numerator:

$$Y(s) = \frac{c_m s^m + c_{m-1} s^{m-1} + \dots + c_1 s + c_0}{s^n + a_{n-1} s^{n-1} + \dots + a_1 s + a_0} U(s)$$

where $m < n$, we can rewrite this expression as

$$\frac{Y(s)}{U(s)} = \frac{X(s)}{X(s)} \frac{Y(s)}{X(s)}$$

where

$$X(s) = \frac{1}{s^n + a_{n-1} s^{n-1} + \dots + a_1 s + a_0} U(s)$$

and

$$Y(s) = (c_m s^m + c_{m-1} s^{m-1} + \dots + c_1 s + c_0) X(s)$$

The transfer function from $U(s)$ to $X(s)$ is similar to the example above. However, the additional equation is for $Y(s)$. This can be converted back to differential equations to give

$$y(t) = c_m \frac{d^m x}{dt^m} + c_{m-1} \frac{d^{m-1} x}{dt^{m-1}} + \dots + c_1 \frac{dx}{dt} + c_0 x(t)$$

If we include this in our state variable diagram we obtain Figure 20.19.

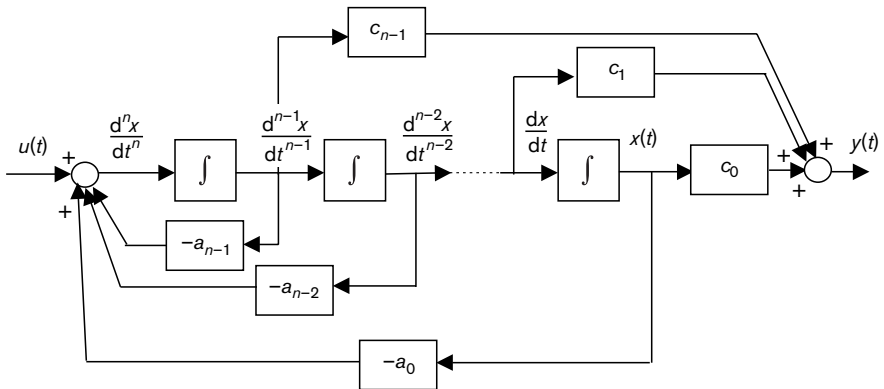


Figure 20.19 State variable model for transfer function description.

Using the state variable notation:

$$y(t) = c_m x_{m+1}(t) + c_{m-1} x_m(t) + \dots + c_1 x_2(t) + c_0 x_1(t)$$

Therefore in matrix form we have

$$y(t) = [c_0 \ c_1 \ \dots \ c_{m-1} \ c_m \ 0 \ \dots \ 0] \mathbf{x}(t)$$

with

$$\begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \\ \dot{x}_3(t) \\ \vdots \\ \dot{x}_{n-1}(t) \\ \dot{x}_n(t) \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & \dots & 0 & 0 \\ 0 & 0 & 1 & \dots & 0 & 0 \\ 0 & 0 & 0 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & 0 & 1 \\ -a_0 & -a_1 & -a_2 & \dots & -a_{n-2} & -a_{n-1} \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \\ \vdots \\ x_{n-1}(t) \\ x_n(t) \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} u(t)$$

Problem The following transfer function represents a fourth-order system:

$$Y(s) = \frac{3s^3 + 2s + 1}{s^4 + 3s^3 + 5s^2 + 0.4s + 28} U(s)$$

- (a) Write down an equivalent state space representation.
- (b) Sketch the ABCD block diagram.

Solution Since we have a polynomial in s on the numerator, we can separate the transfer function into two cascaded transfer functions:

$$X(s) = \frac{1}{s^4 + 3s^3 + 5s^2 + 0.4s + 28} U(s)$$

and

$$Y(s) = (3s^3 + 2s + 1)X(s)$$

If we examine the transfer function for $X(s)$ first, we find that the denominator coefficients are 1, 3, 5, 0.4 and 28. They will appear in reverse order in the fourth-order state matrix (with the unity coefficient of s^4 neglected). Using the companion form gives

$$\begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \\ \dot{x}_3(t) \\ \dot{x}_4(t) \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -28 & -0.4 & -5 & -3 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \\ x_4(t) \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} u(t)$$

The polynomial transfer function relating $Y(s)$ to $X(s)$ has the coefficients in descending order: 3 0 2 1. These will appear in reverse order in the output equation:

$$y(t) = [1 \ 2 \ 0 \ 3] \mathbf{x}(t)$$

The state variable diagram is in Figure 20.20.

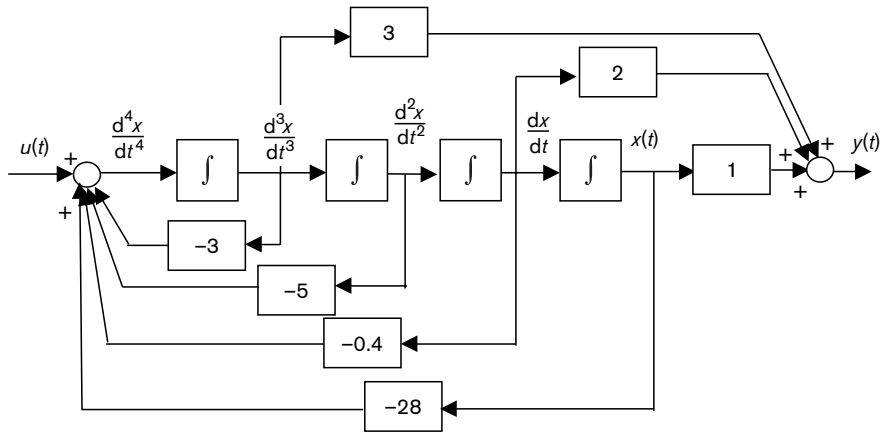


Figure 20.20 State variable diagram for model.

20.12 MATLAB command tf2ss: transfer function to state space conversion

The form of the MATLAB expression is: $[A, B, C, D] = \text{tf2ss}(\text{num}, \text{den})$. The required inputs are:

1. num: a matrix which contains the numerator coefficients for each transfer function in a particular row of the transfer function matrix. For a SISO case, num will just be a single polynomial.
2. den is a vector containing the denominator polynomial coefficients (The denominator is the same for any row in the transfer function matrix.)

The resulting outputs are the state variable matrices **A**, **B**, **C** and **D**.

Problem The following transfer function represents a fourth-order system:

$$G(s) = \frac{3}{s^4 + 2s^3 + 10s^2 + 6s + 3}$$

- (a) Write down an equivalent state space representation.
- (b) Enter this model in MATLAB as an A, B, C, D representation.
- (c) Use the MATLAB command `tf2ss` to create a different state variable model.
- (d) Use the MATLAB `step` command for both models (a) and (c) and comment on the output from both models.

Solution (a) We note the coefficients in the denominator and numerator polynomials and, using the companion form, can quickly write down the equivalent state space form:

$$\begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \\ \dot{x}_3(t) \\ \dot{x}_4(t) \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -3 & -6 & -10 & -2 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \\ x_4(t) \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} u(t)$$

$$y(t) = [3 \ 0 \ 0 \ 0]\mathbf{x}(t)$$

- (b) Note that MATLAB requires you to enter values for the D matrix. You will have to enter a D matrix as a matrix of zeros of size $r \times m$, where r is the number of outputs and m is the number of inputs.

```
A1= [0 1 0 0 ; 0 0 1 0; 0 0 0 1; 3 6 10 2]; B1= [0;0;0;1];
C1=[3 0 0 0]; D1=[0];
system1= ss(A1,B1,C1,D1);
```

- (c) num=[3]; den=[1 2 10 6 3]
[A2,B2,C2,D2]=tf2ss([3],[1 2 10 6 3])

MATLAB gives the following results

$$\begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \\ \dot{x}_3(t) \\ \dot{x}_4(t) \end{bmatrix} = \begin{bmatrix} -2 & -10 & -6 & -3 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \\ x_4(t) \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} u(t)$$

$$y(t) = [0 \ 0 \ 0 \ 3]\mathbf{x}(t)$$

```
system2=ss(A2,B2,C2,D2);
```

- (d) The commands `step(system1)` or `step(system2)` or `step(g)` all produce the same output response (Figure 20.21).

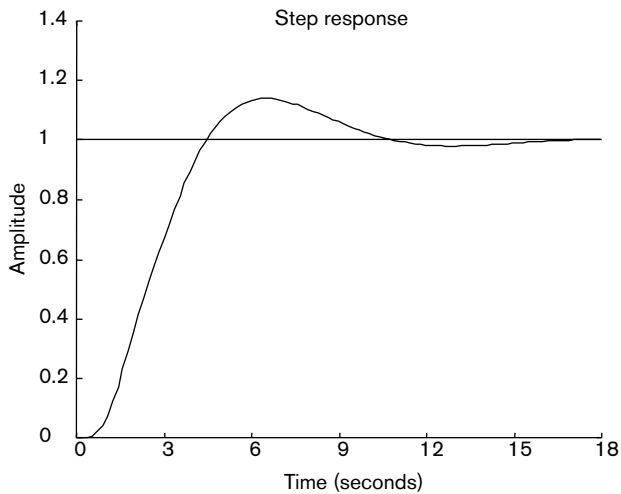


Figure 20.21 Step response from state variable and transfer function models.

What we have learnt

- ✓ To identify the inputs, $u(t)$, outputs, $y(t)$, and state vector, $\mathbf{x}(t)$, of a system.
- ✓ To rewrite the system equations using the state variable notation.

- ✓ To reform the equations in state variables in the standard vector matrix ABCD format:

$$\dot{\mathbf{x}}(t) = \mathbf{Ax}(t) + \mathbf{Bu}(t)$$

$$\mathbf{y}(t) = \mathbf{Cx}(t) + \mathbf{Du}(t)$$

- ✓ To go from a state variable model to a transfer function representation.
- ✓ To convert a system representation from transfer function to state space.
- ✓ To use the MATLAB functions `tf2ss` and `ss2tf` to make the conversion between system representations easier.

Multiple choice

M20.1 A system state variable representation:

- (a) can only be used for SISO systems
- (b) cannot be used for nonlinear systems
- (c) provides a matrix–vector description of the system
- (d) is a multivariable transfer function description of a system

M20.2 Inputs, outputs and states are usually given the notation:

- (a) $u(t)$, $x(t)$, $y(t)$
- (b) $u(t)$, $y(t)$, $x(t)$
- (c) $u(t)$, $z(t)$, $x(t)$
- (d) $u(t)$, $y(t)$, $z(t)$

M20.3 In an ABCD model, which matrix is often zero?

- (a) A
- (b) B
- (c) C
- (d) D

M20.4 A two-input, three-output state variable model:

- (a) has two states
- (b) has three states
- (c) has a 2×3 state matrix
- (d) the information does not define the number of states

M20.5 What are the dimensions of the output **C** matrix for a single-input, two-output, three-state model?

- (a) 3×3
- (b) 2×3
- (c) 3×2
- (d) 3×1

M20.6 The free response of a system is the response of the system with:

- (a) a step input
- (b) any input
- (c) no input
- (d) a bounded input signal

M20.7 The forced response of a system is:

- (a) the output of $\dot{\mathbf{x}}(t) = \mathbf{Ax}(t)$
- (b) the output of a system with no input forces
- (c) the output of a system with a non-zero input, $u(t)$
- (d) the output of a system with non-zero initial conditions

M20.8 The MATLAB function to transform a state variable model to a transfer function description is:

- (a) `tf2ss`
- (b) `sv2tf`
- (c) `ss2tf`
- (d) `tf2sv`

M20.9 The MATLAB state space ABCD block can be used to implement:

- (a) a SISO model
- (b) a MIMO model
- (c) a SIMO model
- (d) all of the above

M20.10 A simple state variable model with $A_{3 \times 3}$, $B_{3 \times 2}$, $C_{1 \times 3}$ and $D_{1 \times 2}$ has the following number of inputs, outputs and states:

- (a) 2, 1, 3
- (b) 1, 2, 3
- (c) 2, 2, 3
- (d) 3, 1, 3

Questions: practical skills

Q20.1 Organise the following system equations as a state space system and identify the matrices, **A**, **B**, **C**, **D**.

$$\begin{aligned}\dot{x}_1(t) &= -4.27x_2(t) \\ \dot{x}_2(t) &= -2.21x_1(t) - 3.96x_2(t) + 3.04u(t) \\ y(t) &= x_1(t) - 0.56x_2(t) + u(t)\end{aligned}$$

Q20.2 A tubular reactor study results in a state space model given by:

$$\begin{aligned}c_p \frac{dx_1}{dt} &= \delta_{C1}x_1(t) + v_T x_2(t) + \left(\frac{4.03\delta_{C1}}{\omega_F} \right) u_1(t) + 0.4\eta_T u_2(t) \\ c_T \frac{dx_2}{dt} &= \delta_{C2}x_1(t) + \eta_T x_1(t) + 3.04u_2(t) \\ y(t) &= \eta_T x_2(t)\end{aligned}$$

Organise the system equations and identify the matrices **A**, **B**, **C**, **D**.

Q20.3 Verify that the two state space descriptions below have the same input–output transfer functions.

$$\begin{aligned}\dot{x} &= \begin{bmatrix} 0 & 1 \\ -1 & 2 \end{bmatrix} x + \begin{bmatrix} 0 \\ 1.125 \end{bmatrix} u \\ y &= [1 \quad 1]x \\ \dot{x} &= \begin{bmatrix} 1 & 1.5 \\ 0 & 1 \end{bmatrix} x + \begin{bmatrix} 1.125 \\ 1.5 \end{bmatrix} u \\ y &= [1 \quad 0]x\end{aligned}$$

Q20.4 Find a state space model for the following equations given that the variable $\alpha(t)$ is the system input, and $\beta(t)$ is the system output. Define the matrices **A**, **B**, **C**, **D** for the system.

$$\begin{aligned}\dot{p}(t) &= q(t), \quad \dot{q}(t) = r(t), \quad \dot{r}(t) = -5r(t) - 9q(t) - 5p(t) + 2\alpha(t) \\ \beta(t) &= p(t) + r(t)\end{aligned}$$

Use MATLAB to find the system transfer function from input to output.

Q20.5 Find a state space model in companion form for the transfer function model

$$G(s) = \frac{(3.5s + 1)}{(s + 0.5)(s + 1.5)(s + 2.5)}$$

Draw the state space diagram for the solution found.

Use MATLAB to find a state space model from the transfer function.

Problems

P20.1 A liquid level process has the transfer function equation

$$H(s) = \left(\frac{4.65}{s + 2} \right) Q_{in}(s)$$

where the liquid height, $h(t)$, is measured in metres, and the inflow, $q_{in}(t)$, is in litres per minute.

(a) Find the d.c. gain and the time constant for this system.

- (b) If the inflow is constant at 0.45 litres/min, sketch the step response for the system, showing the steady state level expected.
- (c) Use the transfer function to create a state space system description for the liquid level system.
- (d) Use Simulink and the state space icon to generate a system step response for the constant inflow.
- (e) Verify that the predicted steady state level is reached and that the time constant is as calculated.

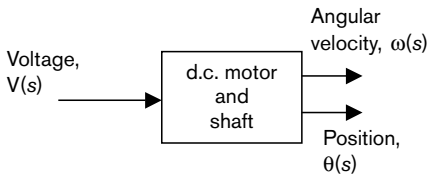
P20.2 A model for a system in the steel industry relates steel product thickness, $h(t)$, and product tension, $\sigma(t)$, as follows:

$$\dot{h}(t) + \eta \dot{\sigma}(t) = (1 - 2\eta)h(t) - 3\eta\sigma(t) + 0.94v(t)$$

$$\alpha_T \dot{h}(t) + 4.25 \dot{\sigma}(t) = (\alpha_T - \delta)h(t) - 12\sigma(t) + 4.75v(t)$$

The input for this system is the motor drive voltage, $v(t)$, the measured output is the thickness, $h(t)$, and η , α_T and δ are physical system parameters. If an engineer defines the state variables as $x_1(t) = h(t)$ and $x_2(t) = \sigma(t)$ with control input, $u(t) = v(t)$, what state space model results? Define the matrices **A**, **B**, **C**, **D** for the system.

P20.3 A simple d.c. motor-shaft system is described by the following block diagram, where there is a tachogenerator and position transducer on the shaft to measure angular velocity and position, respectively.

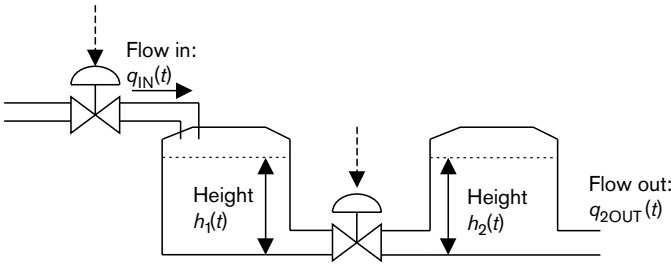


The model description is given as in the table below.

Variable description	Notation
Input voltage	$v(t)$
Angular velocity of shaft	$\omega(t)$
Angular position of shaft	$\theta(t)$
Inertia of the load	J
Motor constants	K, R
System equations	
Position equation	$\dot{\theta} = \omega$
Shaft velocity equation	$\dot{\omega} = \left(\frac{-K^2}{JR}\right)\omega + \left(\frac{K}{JR}\right)v$

- (a) Define suitable state and input variables to give a state space model. Define matrices **A** and **B** algebraically.
- (b) If the angular velocity is taken as one output, and the position is taken as a second output, define an output equation for the system. Define matrices **C** and **D**.
- (c) If $-K/(JR) = 1.35$ and $K = 0.04$, evaluate the state space matrices **A**, **B**, **C**, **D**.
- (d) Find the transfer function from the input to the outputs.

P20.4 Two storage tanks in an oil depot are shown in the diagram and the notation, data and equations are shown in the table.



Variable description	Notation	Data
TANK 1	Head of oil in Tank 1	$h_1(t)$
	Inflow of oil in Tank 1	$q_{IN}(t)$ $1 \text{ m}^3 \text{ min}^{-1}$
	Area of cross-section (circular)	A_1 Radius: 2 m; Height, 10 m
	Outflow of oil in Tank 1	$q_{1OUT}(t)$
TANK 2	Head of oil in Tank 2	$h_2(t)$
	Area of cross-section (circular)	A_2 Radius: 5 m; Height: 10 m
	Outflow of oil in Tank 2	$q_{2OUT}(t)$
	Proportionality constant for $q_{2OUT}(t)$	k $0.25 \text{ m}^2 \text{ min}^{-1}$
Flow constant for interconnecting pipe	R	$0.5 \text{ m}^2 \text{ min}^{-1}$

System equations: analysis stopcock fully open

Tank 1 level	$\frac{d(A_1 h_1)}{dt} = q_{IN}(t) - q_{1OUT}(t)$
Tank 2 level	$\frac{d(A_2 h_2)}{dt} = q_{1OUT}(t) - q_{2OUT}(t)$
Interconnecting pipe	$q_{1OUT}(t) = R(h_1(t) - h_2(t))$
Tank 2 outflow	$q_{2OUT}(t) = k h_2(t)$

(a) If the state variables are $x_1(t) = h_1(t)$ and $x_2(t) = h_2(t)$, with input $u(t) = q_{IN}(t)$, show that the state space model is given by

$$\dot{x}(t) = \begin{bmatrix} -R/A_1 & R/A_1 \\ R/A_2 & -(R+k)/A_2 \end{bmatrix} x(t) + \begin{bmatrix} 1/A_1 \\ 0 \end{bmatrix} u(t)$$

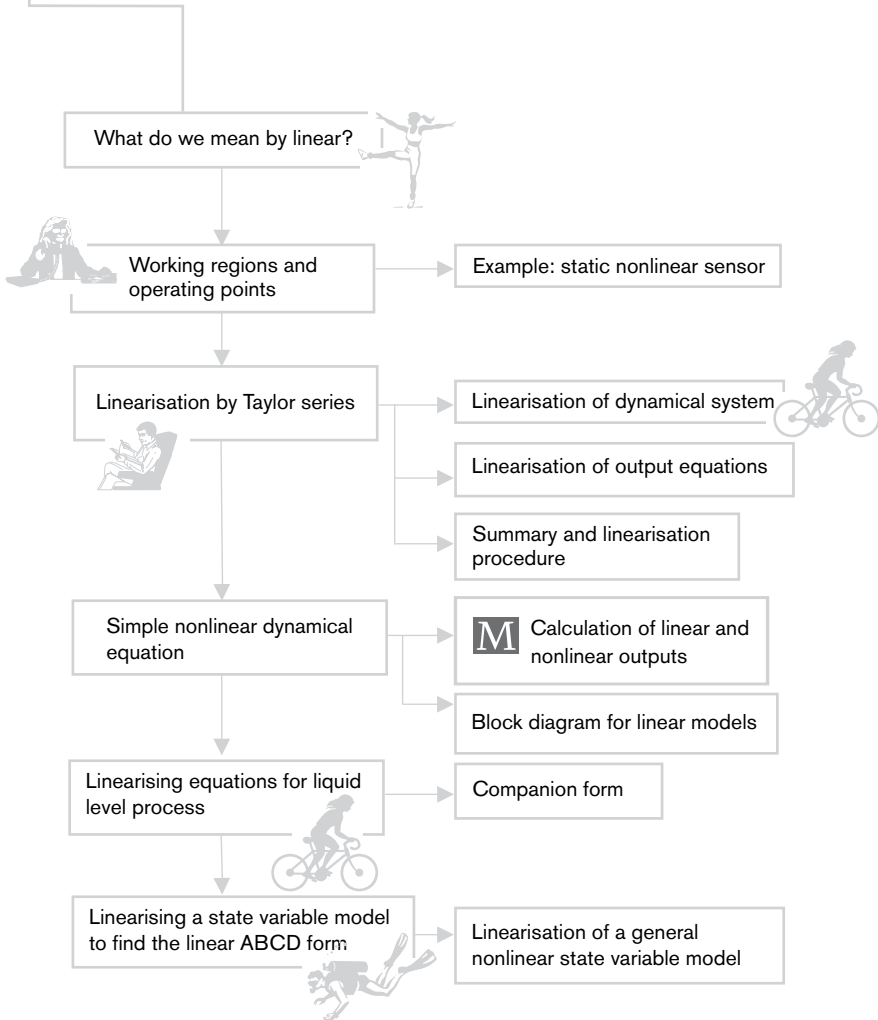
- (b) If the measurements on the system are the levels in the two tanks and a flow meter measuring $q_{1OUT}(t)$, determine an output equation for the system. Define matrices **C** and **D**.
- (c) If the stop cock is fully open, and the tanks start from empty, use Simulink to determine the steady levels that the in-flowing oil reaches in each tank.
- (d) In an assessment of safety at the depot, the following scenario was investigated:

Tank 1: Full to a height of 9.5 metres.	Tank 2: Empty
Stop cock: Was closed but fails and opens fully.	Inflow supply: Turned fully on.

Use a Simulink simulation to determine the outcome of this situation. Is the system safe?

21

Linearisation of systems from the real nonlinear world



Previously in this book, we have met systems whose descriptions can be represented by a set of linear differential equations. These equations have then been transformed into a transfer function or a linear state variable representation. In practice, most system models comprise nonlinear equations or components with nonlinear descriptions. There is a wealth of knowledge on the control of nonlinear systems (beyond the scope of this book), but in many cases the first stage of analysis would be to consider whether the system could be approximated by a linear representation. If a linear form is valid then this linear representation is convenient and has the advantage that we are able to use linear analysis techniques. In this chapter we focus on simple nonlinear features in a system, how a linear approximation can be derived and the region over which this approximation remains valid.

We therefore need to know how to take a set of nonlinear equations and write down a series of linear equations describing the system. We do this by a series of examples:

1. Linearisation of a depth sensor's static nonlinear characteristic
2. Linearisation of a simple nonlinear dynamic equation
3. Linearisation of the nonlinear model equations for a liquid level process
4. Case Study: Linearisation of a nonlinear state variable model

Learning objectives

- To understand what a nonlinear equation or nonlinear component is.
- To appreciate what an operating range and an *operating point* is for a system.
- To linearise a simple static equation.
- To linearise the dynamic equations for a system.

21.1 What do we mean by *linear*?

Consider the simple expression

$$y = 0.5x$$

We can graph the relationship between y and x as shown in Figure 21.1. We find that there is a linear relationship between y and x . For every unit increment in x there is a corresponding increment of 0.5 in y . This is also true in the case where we provide an offset to the graph:

$$y = 0.5x + 3$$

Although the graph is offset, the relationship is still linear, since, as before, a unit increment in x causes the same increment of 0.5 in y . This is more easily seen through the block diagram (Figure 21.2), where the offset is merely added to the linear representation.

So, how do we formally define a linear system? We define *linear* as follows:

A variable y given by $y = ax$, for some constant or function or system a , is linear in x if, (a) when

$$y_1 = a(x_1) \text{ and } y_2 = a(x_2)$$

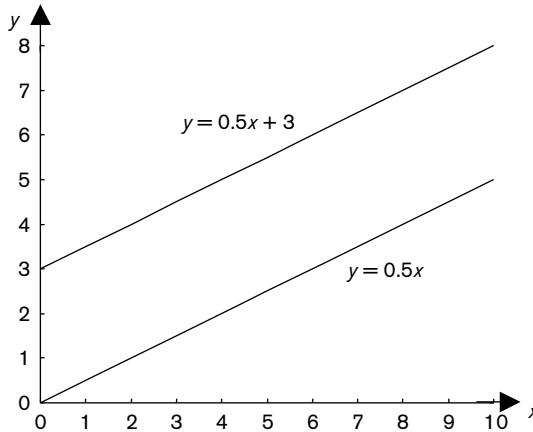


Figure 21.1 Linear relationship between y and x .

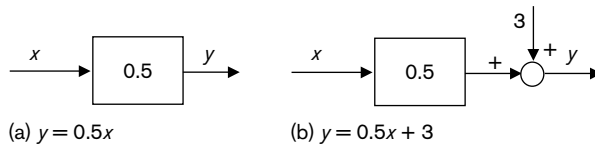


Figure 21.2 Linear systems.

then

$$y = a(x_1 + x_2) = ax_1 + ax_2 = y_1 + y_2$$

and (b) if $y = a(cx)$ then $y = ca(x)$ for scalars c .

We can write this pictorially using our block diagram notation as in Figure 21.3.

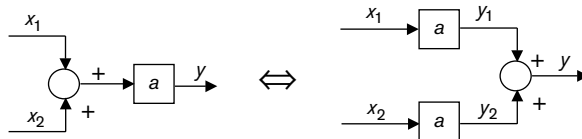
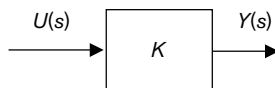


Figure 21.3 Linear system property.

Example The gain block represents a linear relationship between $Y(s)$ and $U(s)$.



Sometimes we input sinusoids, such that $u(t) = \cos \omega t$, giving $U(s) = s/(s^2 + \omega^2)$. A sinusoid is nonlinear. However, *the relationship* between $Y(s)$ and $U(s)$ will still be linear, even if the input signal $U(s)$ is nonlinear.

21.2 Nonlinear examples

There are many examples of nonlinearities in physical systems and equipment. These nonlinearities will appear as nonlinear equations and blocks in the control system models. We give a few examples.

Example 1 In a liquid level system, the flow out, $q(t)$, is proportional to the square root of the level, $h(t)$.

$$q(t) = K\sqrt{h(t)}$$

where K represents a constant dependent on the geometry of the orifice. This flow behaviour is not a linear relationship in $h(t)$ since if we let

$$q_1(t) = K\sqrt{h_1(t)} \quad \text{and} \quad q_2(t) = K\sqrt{h_2(t)}$$

then if $q_3(t) = K\sqrt{h_1(t) + h_2(t)}$, it does not equal $q_1(t) + q_2(t)$. This contradicts our rule for linearity. We find that square roots and squared functions are common nonlinearities in model equations.

Example 2 We consider the ideal gas law:

$$\frac{P(t)V(t)}{T(t)} = \text{constant} = k$$

where P , V and T indicate pressure, volume and temperature respectively. The equation states that

$$P(t) = KT(t)/V(t)$$

Hence pressure is linearly proportional to temperature, provided $V(t)$ is constant. We also note that $P(t) \propto 1/V(t)$. $P(t)$ is therefore not linear in $V(t)$, since we cannot find an equation of the form $P(t) = kV(t)$.

This is an interesting example, since it illustrates that variables may be linearly related *provided* other variables are held at constant values.

Example 3 The resistance output, R , of a thermistor (temperature measuring device) varies with temperature, T :

$$R(t) = Ae^{-b/T(t)}$$

The relation between $R(t)$ and $T(t)$ is not linear. If we write

$$R_1(t) = Ae^{-b/T_1(t)} \quad \text{and} \quad R_2(t) = Ae^{-b/T_2(t)}$$

then if $R_3(t) = Ae^{-b/(T_1(t)+T_2(t))}$, we find that $R_3(t) \neq R_1(t) + R_2(t)$. The exponential function is a nonlinear function.

Example 4 We meet nonlinear components, especially actuators, in our control studies. Figure 21.4 shows an example of saturation limits. For this explanation, we let the actuator slope be unity over the range $0 \leq u(t) \leq d$. When the controller output signal $u(t)$, which represents the actuator input signal, lies in the range 0 to $+d$, the output $u_c(t)$ will equal the input. In this range for $u(t)$, the component is linear. However, when either $u(t) < 0$ or $u(t) > d$, then the output *saturates* and we find that the output signal magnitude is limited to 0 or $+M$, respectively. When the range of $u(t)$ includes the saturation range, the actuator is no longer linear. In practice we can think of a valve which reaches its end points, that is, it is either fully closed or fully open. Once the valve needle, for example, has travelled over its range, it can move no further and any further increase in input will result in no change in the output. In practice, we often try to design controllers so that the valve movement stays within its linear range.

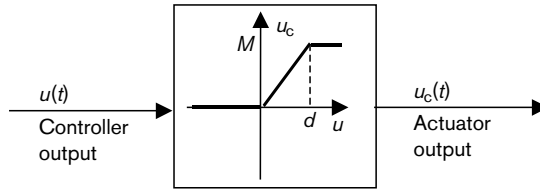


Figure 21.4 Non-linear actuator component.

Skill section

Recognising nonlinear equations

Problem Determine whether the following functions are nonlinear. Use the test: given a function $f(x)$ and any scalar c , does (i) $f(x+y) = f(x) + f(y)$ and (ii) $f(cx) = cf(x)$? If yes, then the function $f(\cdot)$ is linear; if no, then $f(\cdot)$ is nonlinear.

(a) $f(x) = x_1x_2$, where $x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$

(b) $f(x) = a(t)x(t)$

Solution (a) We need to test if

(i) $f(x+y) = f(x) + f(y)$, where $x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$ and $y = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}$

Left-hand side of expression:

$$f(x+y) = f(z) \text{ where } z = \begin{bmatrix} x_1 + y_1 \\ x_2 + y_2 \end{bmatrix}$$

$$f(z) = (x_1 + y_1)(x_2 + y_2) = x_1x_2 + y_1x_2 + x_1y_2 + y_1y_2$$

Right-hand side of expression:

$$f(x) + f(y) = x_1x_2 + y_1y_2$$

Since $x_1x_2 + y_1y_2 \neq x_1x_2 + y_1x_2 + x_1y_2 + y_1y_2$, the function $f(\cdot)$ is not linear, and we do not need to apply test (ii).

(b) We need to test (i) if $f(x+y) = f(x) + f(y)$, where $f(x) = a(t)x(t)$ and $f(y) = a(t)y(t)$.

Left-hand side of expression:

$$f(x+y) = a(t)[x(t) + y(t)] = a(t)x(t) + a(t)y(t)$$

Right-hand side of expression:

$$f(x) + f(y) = a(t)x(t) + a(t)y(t)$$

Both the left- and right-hand sides are equal, so we proceed to test (ii).

(ii) Test if $f(cx) = cf(x)$

Left-hand side of expression: $a(t)[cx(t)] = c[a(t)x(t)]$

Right-hand side of expression: $c[a(t)x(t)]$

Both the left- and right-hand sides are equal. Since both tests (i) and (ii) have been proved true, the function is linear.

21.3 Working regions and operating points

The example of the nonlinear actuator showed us that system components (and systems themselves) may have a range where the component may be considered to be linear. If we can identify this range and find the linear model that corresponds to it, we can then apply linear analysis techniques to design controllers and study the system behaviour. However, our analysis will only remain correct provided our system remains within the range of linearity we have found.

Example An engineer is working on a boiler control project and produces the plot in Figure 21.5 which shows the process temperature, y ($^{\circ}\text{C}$), for varying amounts of injected steam, x (%). The plot is obviously nonlinear. However, if the engineer notices that the system operates close to point B for most of its running time, a *nominal linear model* could be found around *operating point B*. Point B corresponds to an input signal x_p which results in the output signal y_p . We refer to the values of x_p and y_p as the nominal operating conditions at the operating point. We can see that the linear model would be appropriate between input points x_1 and x_2 . The input range x_1 to x_2 would then correspond to a linear operating range, while the output temperature will lie between y_1 to y_2 .

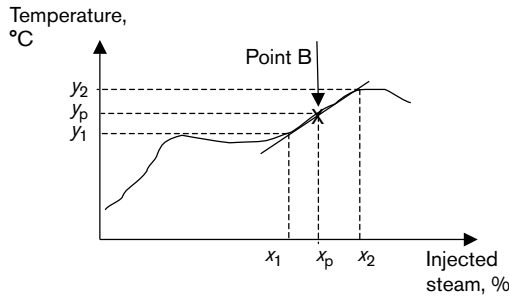


Figure 21.5 Process temperature varying with injected steam.

The following example illustrates how we can identify a linear working region for a simple nonlinear sensor.

Example: A static nonlinear sensor characteristic

The tank system in Figure 21.6 shows a pump controlling the flow of water into a tank.

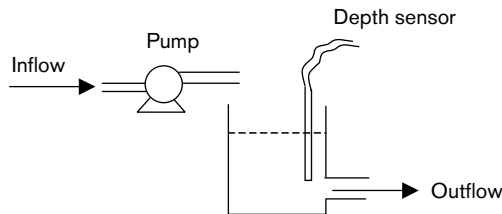


Figure 21.6 Tank level system.

The depth sensor produces an output voltage which is related to the depth of liquid in the tank. A calibration of the depth sensor involved taking output voltage readings for every 1 cm increase in

depth. This calibration started at 3 cm and continued till 28 cm. The resulting calibration curve is plotted in Figure 21.7.

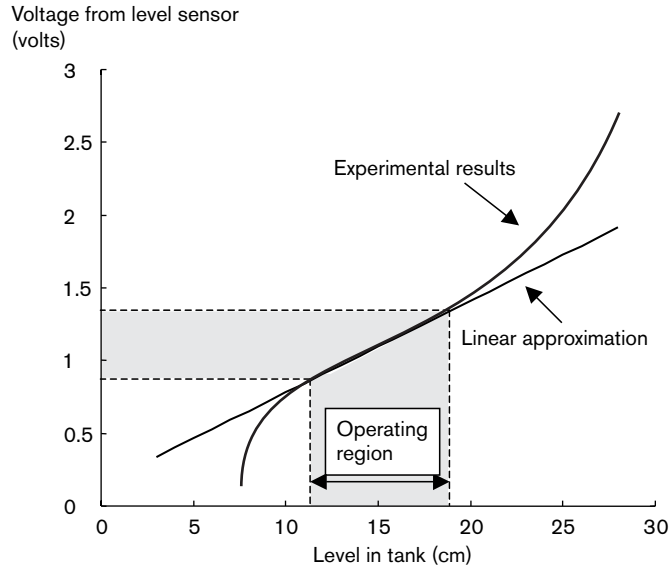


Figure 21.7 Graph of output from nonlinear sensor.

In the middle region between approximately 11.5 and 18.5 cm depth, the output sensor has a reasonable linear approximation given by

$$v(t) = md(t) + c$$

where $v(t)$ = sensor voltage (volts)

$d(t)$ = depth of tank (cm)

m = linear relationship = 0.063 V/cm

c = offset = 0.15 V

Therefore

$$v(t) = 0.063 d(t) + 0.15$$

If we operate the system such that the level stays within the operating region between 11.5 and 18.5 cm, we can use this linear approximation. However, outside this region the linear formula would no longer be valid.

For many simple nonlinear characteristics which depend on only one input, we can often form linear approximations without going to the mathematical rigour of least-squares approximations or other approximation techniques. However, in industrial process systems we often find that the system itself is described mathematically by several nonlinear differential equations from which it is more difficult to form a linear model. Many industrial processes operate within a specified range of a set of operating conditions. Within this range, a linear model may be valid. For example, we saw, in the depth sensor example, that the linear approximation was valid over a particular range of depth. Provided the system operated such that the depth remained within this range, the sensor output could be considered as linear. What we are doing when we linearise a system

description is effectively the same as we saw for the nonlinear sensor above: we are approximating a nonlinear curve by a straight line – that is, we are saying that one variable is related linearly to one or more other variables.

When we linearise a system, we must choose an *operating point* around which we provide the linear approximation.

If we denote the system description by a set of states $x(t)$ and a set of inputs, $u(t)$, then the differential equations for the system are given by

$$\frac{dx}{dt} = f(x, u)$$

These equations represent the nonlinear dynamic behaviour of the process and indicate how the process outputs change with time. If we set the derivatives to zero ($dx/dt = 0$), we find the *steady state* conditions. For example, if we consider a simple, linear differential equation

$$3\frac{dx}{dt} + x(t) = 5u(t)$$

and we set the derivative to zero, we are effectively saying that there is no change in the state $x(t)$ over time; that is, the system is in steady state given by the equation

$$x(t) = 5u(t)$$

We find that this corresponds to our understanding of first-order system responses: the transient portion of the response was the response due to the dynamic differential equation, whereas the steady state portion was when the dynamics had decayed to zero and we were left with the steady state response.

Key result

The operating conditions at which we base our linear models are often the *steady state* values of the system, that is, for example: $x_p(t) = x_p$ and $u_p(t) = u_p$. These *operating points* correspond to the steady state condition

$$f(x_p, u_p) = 0$$

Example: Operating condition for magnetic suspension system

Figure 21.8 shows a magnetic suspension system where the iron ball is suspended a distance h below the actuator producing the magnetic field. The actuator (electromagnet) requires a current $i(t)$ to create the magnetic field which supports the iron ball against the downward force of gravity. The differential equation for this system is simply derived by providing a force balance on the ball:

$$m\ddot{h}(t) = mg - \alpha \frac{i^2(t)}{h^2(t)} - \beta \dot{h}(t)$$

where the right-hand side contains three terms: the first due to gravity, the second due to the inverse square law attracting the ball and the last term due to air resistance opposing the fall of the ball. The ball's position is measured by a photocell light detection sensor.

The ball is initially suspended a distance h_p from the electromagnet by applying a current i_p . If the ball moves slightly from this position, the amount of light detected on the opposite side of the ball will change. The controller will then act on this change to alter the current to restore the ball to its original position. The amount of current required will be a small change from the initial

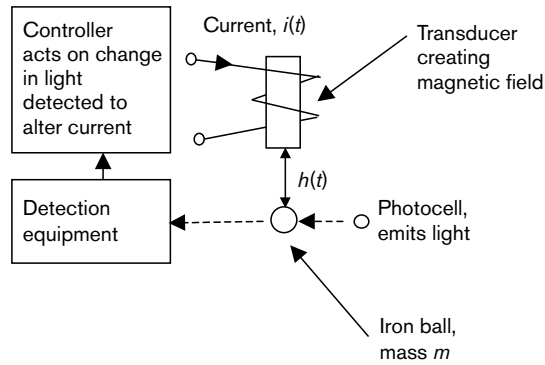


Figure 21.8 A magnetic suspension system.

operating point, i_p , that is, $i(t) = i_p + \delta i(t)$. In this case, the operating point is given by (h_p, i_p) . The current i_p is required to maintain the ball at its operating point, h_p .

Once we have defined the set of operating conditions, we can move forwards to develop a linear approximation.

21.4 Linear approximation through Taylor series

Our linear models for $x(t)$ often represent the small variations, $\delta x(t)$ and $\delta u(t)$, around the operating point (x_p, u_p) .

$$x(t) = x_p + \delta x(t) \quad \text{giving} \quad \delta x(t) = x(t) - x_p$$

$$u(t) = u_p + \delta u(t) \quad \text{giving} \quad \delta u(t) = u(t) - u_p$$

We note the following:

$f(x, u)$: Nonlinear function which can be the set of state derivatives which may be nonlinear in terms of the states or the inputs

$f_L(x, u)$: Linear function

$f(x_p, u_p)$: Value of function at operating conditions ($= f_L(x_p, u_p)$)

We then form a Taylor series expansion and truncate at the first-order terms. Given a nonlinear function $f(x, u)$, the Taylor series expansion is given by

$$f(x, u) = f(x_p, u_p) + \left. \frac{\partial f}{\partial x} \right|_{x_p, u_p} (x - x_p) + \left. \frac{\partial f}{\partial u} \right|_{x_p, u_p} (u - u_p) + \dots$$

The linear approximation is then given by

$$f_L(x, u) = f(x_p, u_p) + a(x - x_p) + b(u - u_p)$$

where

$$a = \left. \frac{\partial f}{\partial x} \right|_{x_p, u_p} \quad \text{and} \quad b = \left. \frac{\partial f}{\partial u} \right|_{x_p, u_p}$$

The linear approximation can be written as

$$f_L(x, u) = f(x_p, u_p) + df(x - x_p, u - u_p)$$

and a small change in state variable $x(t)$, or a small change in input $u(t)$, is linearly related to a small change in the output of function $f_L(x, u)$. The linear variation around the operating point of (x_p, u_p) is given by $df(x - x_p, u - u_p)$. Remembering that $x - x_p = \delta x$ and $u - u_p = \delta u$, we can write

$$df(x - x_p, u - u_p) = df(\delta x, \delta u) = a\delta x + b\delta u$$

We refer to this model as the ‘small change model’ to indicate that it represents the small variations in the function around the operating condition.

21.4.1 Linearisation procedure

Given a nonlinear function $f(x_1, x_2, \dots, x_n, u_1, u_2, \dots, u_m)$:

1. Define the operating condition at which we will produce a linear model.
This condition is often found by setting $f(x_1, x_2, \dots, x_n, u_1, u_2, \dots, u_m) = 0$ and considering the resulting values of x_1, x_2, \dots, x_n for a set of input values u_1, u_2, \dots, u_m .
2. Use partial differentiation to find the linear coefficients and evaluate these at the operating points.

$$\text{For example, } a_1 = \left. \frac{\partial f}{\partial x_1} \right|_{x_p, u_p}, \quad a_2 = \left. \frac{\partial f}{\partial x_2} \right|_{x_p, u_p}, \quad \dots, \quad a_n = \left. \frac{\partial f}{\partial x_n} \right|_{x_p, u_p}$$

$$b_1 = \left. \frac{\partial f}{\partial u_1} \right|_{x_p, u_p}, \quad b_2 = \left. \frac{\partial f}{\partial u_2} \right|_{x_p, u_p}, \quad \dots, \quad b_m = \left. \frac{\partial f}{\partial u_m} \right|_{x_p, u_p}$$

3. Form the small change model equations:

$$df(\delta x, \delta u) = a_1\delta x_1 + a_2\delta x_2 + \dots + a_n\delta x_n + b_1\delta u_1 + b_2\delta u_2 + \dots + b_m\delta u_m$$

Remark

1. In control systems we occasionally use large capital letters in the time domain to represent the actual process variables. We then use small letters to denote small changes about an operating point or reference point or reference level. This notation makes a clear distinction between the output of the linear model and the nonlinear model equations.
2. We find that our linear models will often be written in terms of $x(t)$ and $u(t)$. If the system were linear, then the states and inputs, $x(t)$ and $u(t)$, would be correct for small and large changes in process values. However, if the system has been linearised, then the linear model in $x(t)$ and $u(t)$ is only valid for small variations about x_p and u_p . It is up to us to make the correct interpretation for the system we are working with.
3. In this book, many of the examples have used linear models and the notation for the variables has been given in small letters.

21.5 Where do we apply linearisation?

Linearisation of systems

Consider the following set of nonlinear differential equations:

$$\dot{x}(t) = f(x(t), u(t))$$

The condition at the operating point gives $\dot{x}_p(t) = f(x_p, u_p)$ and for steady state conditions we set $\dot{x}_p(t) = 0$ and solve $f(x_p, u_p) = 0$. We note that the nonlinear set of equations can be approximated by

$$\dot{x}_L(t) = f(x_p, u_p) + df(\delta x(t), \delta u(t))$$

or

$$\dot{x}_L(t) = \frac{d}{dt}(x_p + \delta x) = f(x_p, u_p) + df(\delta x(t), \delta u(t))$$

and

$$\dot{x}_p + \delta \dot{x} = f(x_p, u_p) + df(\delta x(t), \delta u(t))$$

Since $\dot{x}_p = f(x_p, u_p)$, we find

$$\delta \dot{x}(t) = df(\delta x(t), \delta u(t)) = a\delta x(t) + b\delta u(t)$$

where

$$a = \left. \frac{\partial f}{\partial x} \right|_{x_p, u_p} \quad \text{and} \quad b = \left. \frac{\partial f}{\partial u} \right|_{x_p, u_p}$$

However, engineers are sometimes lazy in their notation and, once the linear model is established, tend to use $x(t)$ and $u(t)$ rather than $\delta x(t)$ and $\delta u(t)$ in expressing the linear equations.

Linearisation of output equations

Consider the function

$$y(t) = g(x(t), u(t))$$

At the operating point we have $y_p = g(x_p, u_p)$. Linearisation about this operating point gives

$$y_L(t) = g(x_p, u_p) + dg(\delta x(t), \delta u(t))$$

If we consider small changes about the operating point, we can write

$$y_L(t) = y_p + \delta y(t) = g(x_p, u_p) + dg(\delta x(t), \delta u(t))$$

Since $y_p = g(x_p, u_p)$, we find that

$$\delta y(t) = dg(\delta x(t), \delta u(t))$$

where we have

$$dg(\delta x(t), \delta u(t)) = c\delta x(t) + d\delta u(t)$$

with

$$c = \left. \frac{\partial g}{\partial x} \right|_{x_p, u_p} \quad \text{and} \quad d = \left. \frac{\partial g}{\partial u} \right|_{x_p, u_p}$$

Key result: Linearisation notation summary

Given a system: $\dot{x}(t) = f(x(t), u(t))$
 $y(t) = g(x(t), u(t))$

The operating condition solves: $\dot{x}_p(t) = f(x_p, u_p), y_p = g(x_p, u_p)$

For steady state conditions, set: $\dot{x}_p(t) = 0$ and solve $f(x_p, u_p) = 0$

Use the small change expressions: $x(t) = x_p + \delta x(t)$
 $u(t) = u_p + \delta u(t)$
 $y(t) = y_p + \delta y(t)$

We find the small changes around the operating point given by the model

$$\delta \dot{x}(t) = df(\delta x(t), \delta u(t)) = a\delta x(t) + b\delta u(t)$$

$$\delta y(t) = dg(x(t), \delta u(t)) = c\delta x(t) + d\delta u(t)$$

with

$$a = \left. \frac{\partial f}{\partial x} \right|_{x_p, u_p}, b = \left. \frac{\partial f}{\partial u} \right|_{x_p, u_p}$$

$$c = \left. \frac{\partial g}{\partial x} \right|_{x_p, u_p} \text{ and } d = \left. \frac{\partial g}{\partial u} \right|_{x_p, u_p}$$

21.6 Linearisation of a simple nonlinear dynamic equation

We are going to apply the linearisation method to a simple dynamical equation to understand more clearly the role of the operating point and the errors involved when the input moves from the operating condition. Let us consider a system that has the nonlinear description:

$$\dot{x}(t) = f(x, u) = -x^2(t) + 9u(t)$$

In this example, we note that $\dot{x}(t)$ is a function of $x(t)$ and $u(t)$, both of which can vary. We would like to linearise this system and we follow the general procedure.

1. Identify the operating condition at which we obtain a linear model

The operating condition will be provided by a set of operating points (x_p, u_p) for the system variables. We shall choose the steady state condition, $\dot{x}(t) = 0$. This gives

$$-x^2(t) + 9u(t) = 0 \quad \text{or} \quad x(t) = 3\sqrt{u(t)}$$

If we let the input signal be $u_p = 1$, then the steady state value of $x(t)$ will be given by $x_p = 3$. The operating point is then $(x_p, u_p) = (3, 1)$. The value of the function at the operating point is zero, as we have used the definition of steady state for our operating point: $f(x_p, u_p) = 0$.

2. Apply partial differentiation to the model to find the linear coefficients

Applying a Taylor series expansion to $f(x(t), u(t))$ around u_p and x_p gives the linear approximation $\dot{x}_L = f_L(x, u)$:

$$\begin{aligned} \dot{x}_L &= f(x_p, u_p) + df(\delta x, \delta u) = f(x_p, u_p) + a(x - x_p) + b(u - u_p) \\ \frac{d(x_p + \delta x(t))}{dt} &= f(x_p, u_p) + a(x - x_p) + b(u - u_p) \end{aligned}$$

Noting $\dot{x}_p = f(x_p, u_p)$ then the 'small change model' for $\delta\dot{x}(t)$ is given by

$$\delta\dot{x}(t) = a(x - x_p) + b(u - u_p) = a\delta x + b\delta u$$

with

$$a = \left. \frac{\partial f}{\partial x} \right|_{x_p, u_p} = (-2x|_{x_p, u_p}) = -6$$

and

$$b = \left. \frac{\partial f}{\partial u} \right|_{x_p, u_p} = 9$$

Therefore,

$$\delta\dot{x}(t) = -6 \delta x(t) + 9 \delta u(t)$$

21.6.1 Retrieving the 'actual' process value

The model $\delta\dot{x}(t) = -6\delta x(t) + 9\delta u(t)$ from example 2 gives the small changes around the operating point for small changes in $\delta x(t)$ and $\delta u(t)$. For example, since the operating value of $x(t)$ was $x_p = 3$, a value of $\delta x(t) = 0.1$ would correspond to an absolute value of $x(t) = x_p + \delta x(t) = 3.1$. Likewise, the input signal $\delta u(t)$ to the small change model should be the small variation from the operating condition. In this case, with

$$u(t) = u_p + \delta u(t)$$

and $u_p = 1$, then to input the corresponding value of $u(t) = 1.3$ to the small change model we would enter only $\delta u(t) = 0.3$.

If we wished to plot or use the actual values of the linear function, $f_L(x, u)$, we would use the following equations:

$$\begin{aligned} f_L(x, u) &= f(x_p, u_p) + a\delta x(t) + b\delta u(t) \\ &= f(x_p, u_p) - 6\delta x(t) + 9\delta u(t) \\ &= f(x_p, u_p) - 6(x - x_p) + 9(u - u_p) \end{aligned}$$

We note that since in our example $f(x_p, u_p) = 0$,

$$f_L(x, u) = -6(x - x_p) + 9(u - u_p)$$

M 21.6.2 Calculation of the linear and nonlinear output values

Let us calculate both the nonlinear and linear output for values of x around x_p . We use the following MATLAB code to create a vector of nonlinear and a vector of linear values around the operating point.

```
x=0:0.5:6;           % Create vector of x values spaced 0.5 apart
u=1;                 % Let u(t) be a constant value for the example
fnl=-x.*x + 9*u;    % Calculate nonlinear function
fl=-6*(x-3)+9*(u-1); % Calculate linear function
plot(x,fnl);         % Plot nonlinear function
hold                 % Current plot held
plot(x,fl)           % Superimpose linear function
```


Figure 21.9 shows both the linear and nonlinear function plotted against varying values of x with u held constant at $u(t) = 1$. We can see that the functions take the same value at the operating point and that the error between the nonlinear and linear function increases as we move further from the operating point.

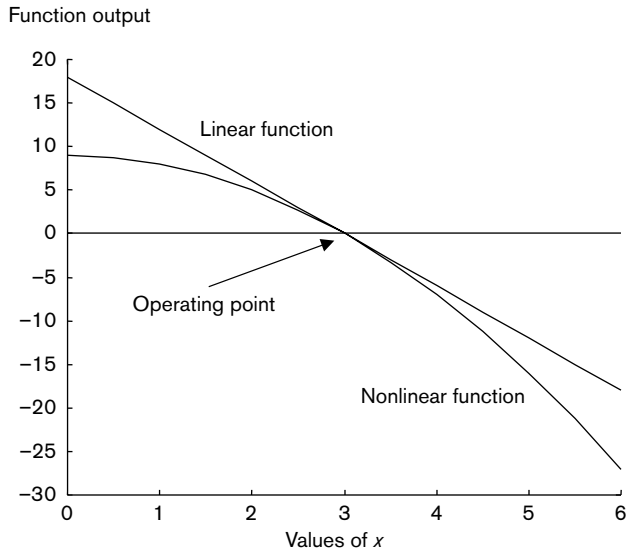


Figure 21.9 Non-linear and linear functions.

21.6.3 Block diagrams for linear models

The small change linear model

$$\delta\dot{x}(t) = -6\delta x(t) + 9\delta u(t)$$

can be expressed in block diagram format as shown in Figure 21.10.

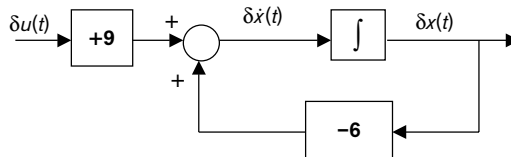


Figure 21.10 Block diagram of small change linear model.

What we must note is that the output $\delta x(t)$ will only represent the *small changes* from the operating point. We must remember that to find the correct output we must add on the steady state value, x_p . We show this in Figure 21.11. This becomes important in situations where we derive linear models for control design, simulate them in Simulink and then must remember that the output is not the actual voltage or pressure, for example, but the small change from the operating point. Likewise, if we wished to input the *actual* process value $u(t)$, we would alter the input shown in Figure 21.11.

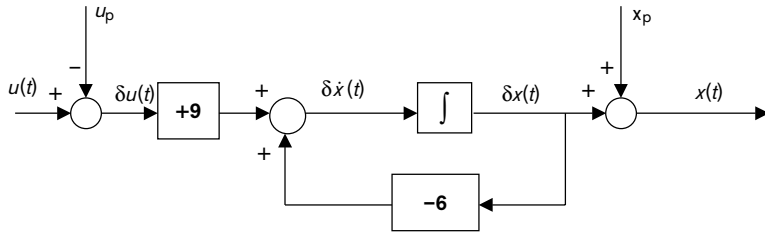


Figure 21.11 Block diagram of small change model showing the use of actual process input and output values.

21.7 Linearising the model equations for a liquid level process

We now apply the linearisation method to an engineering example. We use a simple liquid level example. The nonlinear equations for variations in the tank variables are:

$$Q_i - Q_o = A \frac{dH}{dt}$$

and

$$Q_o = C_d a_d \sqrt{2gH}$$

where

Q_i	flow in ($\text{m}^3 \text{s}^{-1}$)
Q_o	flow out ($\text{m}^3 \text{s}^{-1}$)
A	area of tank (m^2)
H	height of tank (m)
C_d	discharge coefficient
a_d	area of discharge orifice (m^2)
g	acceleration due to gravity (m s^{-2})

We can rearrange the differential equation to give the expression for the change in output level as:

$$\frac{dH}{dt} = \frac{Q_i}{A} - \frac{R}{A} \sqrt{H} = f(H, Q_i)$$

where $R = C_d a_d \sqrt{2g}$ is a constant dependent on the orifice. We provide the following system values:

$$C_d = 0.6, a_d = 0.0314 \text{ m}^2, A = 0.97 \text{ m}^2, g = 9.81 \text{ m s}^{-2}$$

giving $R = C_d a_d \sqrt{2g} = 0.0834$. The function $f(H, Q_i)$ is linear in Q_i but is not a linear expression in H , and we will now linearise to produce a linear model. We follow the general procedure:

1. Define the operating condition for the linear model

We first need to define the operating point of the process. We will assume that we wish to control the tank level to be at a depth of $h(t) = 15 \text{ cm} = 0.15 \text{ m}$. Therefore we will set the operating condition at $h_p = 0.15$. From the original equation we find that in steady state, $dH/dt = 0$, giving

$$0 = \frac{Q_i}{A} - \frac{R}{A} \sqrt{H}$$

or

$$Q_i = R\sqrt{H}$$

Therefore to achieve the steady state level of $h_p = 0.15$ m, we need to input a constant flow of $Q_i = q_p$, where

$$q_p = 0.0834 \times \sqrt{0.15} = 0.0323 \text{ m}^3/\text{s}$$

We assume that we are working around an operating point, given by (h_p, q_p) and consider small changes, $h(t)$, around this point:

$$H(t) = h_p + h(t)$$

$$Q(t) = q_p + q_i(t)$$

2. Apply partial differentiation to evaluate the linear coefficients and evaluate these coefficients at the values given by the operating condition

We linearise the function dH/dt to find

$$\frac{d(h_p + h(t))}{dt} = f(h_p, q_p) + df(h(t), q_i(t))$$

Since $\dot{h}_p = f(h_p, q_p)$, the small change model is given by

$$\frac{dh}{dt} = df(h(t), q_i(t)) = ah(t) + bq_i(t)$$

where a and b are the linear model coefficients given by:

$$a = \left. \frac{\partial f}{\partial H} \right|_{q_p, h_p} = \left[\frac{-R}{A} \frac{1}{2\sqrt{H}} \right]_{q_p, h_p} = -\frac{R}{2A\sqrt{h_p}}$$

$$b = \left. \frac{\partial f}{\partial Q} \right|_{q_p, h_p} = \left[\frac{1}{A} \right]_{q_p, h_p} = \frac{1}{A}$$

Then

$$\frac{dh}{dt} = -\frac{R}{2A\sqrt{h_p}} h(t) + \frac{1}{A} q_i(t)$$

3. Form the linear model equations.

The linear model is given by

$$\frac{dh}{dt} = \frac{1}{A} q_i(t) - \frac{R}{2A\sqrt{h_p}} h(t)$$

This is a linear first-order differential equation expression in the small variations of flow and depth, $q_i(t)$ and $h(t)$, respectively. We can represent this in state variable form by using the following notation change:

input $u(t)$: $q_i(t)$, inflow

state, $x(t)$: $h(t)$, level

output, $y(t)$: $h(t)$, level

This gives

$$\dot{x}(t) = -\frac{R}{2A\sqrt{h_p}}x(t) + \frac{1}{A}u(t)$$

$$y(t) = x(t)$$

or

$$\dot{x}(t) = ax(t) + bu(t)$$

$$y(t) = cx(t)$$

$$a = \left[-\frac{R}{2A\sqrt{h_p}} \right] \quad b = \left[\frac{1}{A} \right] \quad c = [1]$$

Using the system values gives $a = -0.1110$, $b = 1.0309$.

We use the state space block in Simulink to provide a step response for the system (Figure 21.12). Since we have a linearised model, we have chosen the input step change in flow to be $0.001 \text{ m}^3/\text{s}$. This represents a 3% change in flow from the operating condition of $q_p = 0.0323 \text{ m}^3/\text{s}$. Likewise, the output signal from the ABCD model is the change in level from the operating condition of $0.15 \text{ m} = 15 \text{ cm}$. We should remember to add the operating point values if we wish to report the actual tank input flow and the resulting tank level.

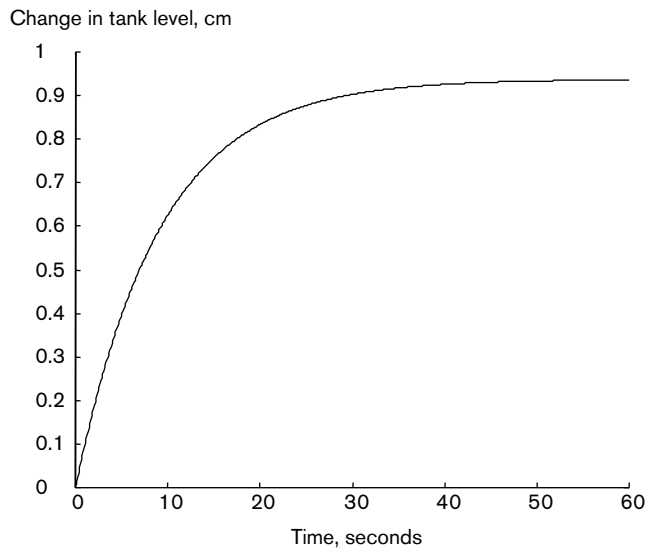


Figure 21.12 Change in level of tank contents due to 3% inflow change.

21.8 Linearisation of a more general nonlinear state variable model

If we generalise the above linearisation we find that given a system

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t))$$

$$\mathbf{y}(t) = \mathbf{g}(\mathbf{x}(t), \mathbf{u}(t))$$

then the linear **ABCD** model will be given by (for a 3×3 system with two inputs and two outputs):

$$\begin{bmatrix} \delta \dot{x}_1(t) \\ \delta \dot{x}_2(t) \\ \delta \dot{x}_3(t) \end{bmatrix} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \frac{\partial f_1}{\partial x_3} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \frac{\partial f_2}{\partial x_3} \\ \frac{\partial f_3}{\partial x_1} & \frac{\partial f_3}{\partial x_2} & \frac{\partial f_3}{\partial x_3} \end{bmatrix}_{x_p, u_p} \begin{bmatrix} \delta x_1(t) \\ \delta x_2(t) \\ \delta x_3(t) \end{bmatrix} + \begin{bmatrix} \frac{\partial f_1}{\partial u_1} & \frac{\partial f_1}{\partial u_2} \\ \frac{\partial f_2}{\partial u_1} & \frac{\partial f_2}{\partial u_2} \\ \frac{\partial f_3}{\partial u_1} & \frac{\partial f_3}{\partial u_2} \end{bmatrix}_{x_p, u_p} \begin{bmatrix} \delta u_1(t) \\ \delta u_2(t) \end{bmatrix}$$

$$\begin{bmatrix} \delta y_1(t) \\ \delta y_2(t) \end{bmatrix} = \begin{bmatrix} \frac{\partial g_1}{\partial x_1} & \frac{\partial g_1}{\partial x_2} & \frac{\partial g_1}{\partial x_3} \\ \frac{\partial g_2}{\partial x_1} & \frac{\partial g_2}{\partial x_2} & \frac{\partial g_2}{\partial x_3} \end{bmatrix}_{x_p, u_p} \begin{bmatrix} \delta x_1(t) \\ \delta x_2(t) \\ \delta x_3(t) \end{bmatrix} + \begin{bmatrix} \frac{\partial g_1}{\partial u_1} & \frac{\partial g_1}{\partial u_2} \\ \frac{\partial g_2}{\partial u_1} & \frac{\partial g_2}{\partial u_2} \end{bmatrix}_{x_p, u_p} \begin{bmatrix} \delta u_1(t) \\ \delta u_2(t) \end{bmatrix}$$

Although this may look like complicated mathematics, the process of linearising the nonlinear model is not difficult; once we have a nonlinear state variable model and the operating conditions, it is a simple case of using partial differentiation with each state and each input variable. We illustrate this with the magnetic suspension example.

21.9 Linearising a nonlinear state variable model to produce a linear ABCD model

We consider the magnetic suspension example, which has a second-order differential equation describing the behaviour of the ball under gravity and the magnetic force.

$$m\ddot{h}(t) = mg - \alpha \frac{i^2(t)}{h^2(t)} - \beta \dot{h}(t)$$

or

$$\ddot{h}(t) = g - a \frac{i^2(t)}{h^2(t)} - b\dot{h}(t)$$

where $a = \alpha/m$ and $b = \beta/m$. To convert the model to state variable form, we first identify the inputs, outputs and state variables.

The input signal is the current, $i(t)$.

The output signal will be the distance, $h(t)$.

We will let the states be the ball height and velocity.

$$x_1(t) = h(t)$$

$$x_2(t) = \dot{h}(t) = \dot{x}_1(t)$$

We now re-examine the equation for $\ddot{h}(t)$, substituting in for the states $x_1(t)$, $x_2(t)$ and input, $u(t)$, to give

$$\dot{x}_2(t) = g - a \frac{u^2(t)}{x_1^2(t)} - bx_2(t)$$

Therefore the two state variable nonlinear differential equations are

$$\begin{aligned} \dot{x}_1(t) &= x_2(t) & \text{or} & \quad \dot{x}_1(t) = f_1(x_1, x_2, u) \\ \dot{x}_2(t) &= g - a \frac{u^2(t)}{x_1^2(t)} - bx_2(t) & \text{or} & \quad \dot{x}_2(t) = f_2(x_1, x_2, u) \end{aligned}$$

with the output equation given by

$$y(t) = x_1(t) \quad \text{or} \quad y(t) = g_1(x_1, x_2, u)$$

This model, although represented in terms of state variables, is not in a linear ABCD format. We also note that since we have two state differential equations, we have two functions f_1 and f_2 . The output is given by g_1 . We apply the general linearisation procedure to the nonlinear system.

1. Identify the operating condition for linearisation

To produce a linear ABCD model from the nonlinear state variable and output equations, we must linearise these equations about an operating point. We note that in this example, the output equation, $y(t) = x_1(t)$, and the first state variable equation is already linear, and therefore we only consider the second state variable differential equation, which is given by

$$\dot{x}_2(t) = f_2(x_1(t), x_2(t), u(t))$$

The operating point is given by the values of u_p and $x_p = [x_{1p} \ x_{2p}]^T$. In this example, we set the operating current to be $i(t) = 0.12$ A, giving $u_p = 0.12$ A. We let the rates of the state variables be zero, effectively saying that at the operating point the ball has a fixed position but with no velocity, that is, $\dot{h}(t) = 0$. In state variables this corresponds to $x_2(t) = 0 = x_{2p}$. If $\dot{x}_2(t) = 0$, then the nonlinear differential equation for $\dot{x}_2(t)$ gives

$$0 = g - a \frac{u_p^2}{x_{1p}^2}$$

which results in

$$x_{1p} = \sqrt{\frac{a}{g}} u_p$$

If we let $m = 0.1$ kg, $\beta = 0.95$ kg/s, $\alpha = 0.9$ m³ s⁻² kg/A² and $u_p = 0.12$ A, then $b = 9.5$ s⁻¹ and $a = 9$ m³ s⁻²/A². This gives

$$x_{1p} = \sqrt{\frac{9}{9.81}} \times 0.12 = 0.1101 \text{ m}$$

Therefore we have $u_p = 0.12$, $x_p = [0.1101 \ 0]^T$.

2. Apply partial differentiation to the nonlinear model equations to produce the linear coefficients, and evaluate these coefficients at the operating condition values.

We consider firstly the differential equation for state $x_1(t)$:

$$\dot{x}_1(t) = f_1(x_1(t), x_2(t), u(t)) = x_2(t)$$

The linear approximation to this function gives

$$\frac{d(x_{1p} + \delta x_1)}{dt} = f_1(x_{1p}, x_{2p}, u_p) + df_1(x_1(t), x_2(t), u(t))$$

The small change model can be determined as

$$\delta \dot{x}_1(t) = df_1(x_1(t), x_2(t), u(t)) = a_{11}\delta x_1(t) + a_{12}\delta x_2(t) + b_{11}\delta u(t)$$

where a_{11} , a_{12} and b_{11} are the linear coefficients given by

$$a_{11} = \left. \frac{\partial f_1}{\partial x_1} \right|_{x_p, u_p} = 0$$

$$a_{12} = \left. \frac{\partial f_1}{\partial x_2} \right|_{x_p, u_p} = [1] = 1$$

$$b_{11} = \left. \frac{\partial f_1}{\partial u} \right|_{x_p, u_p} = 0$$

This gives the small change model for $\dot{x}_1(t)$ as

$$\delta \dot{x}_1(t) = \delta x_2(t)$$

We note that the original differential equation was linear and that linearising this equation merely resulted in a similar equation in the small change variables.

We can see this also if we consider the linear output equation

$$y(t) = g_1(x_1(t), x_2(t), u(t)) = x_1(t)$$

Applying the linearisation procedure produces the small change model

$$\delta y(t) = c_{11}\delta x_1(t) + c_{12}\delta x_2(t) + d_{11}\delta u(t)$$

where the linear coefficients are given by

$$c_{11} = \left. \frac{\partial g_1}{\partial x_1} \right|_{x_p, u_p} = 1$$

$$c_{12} = \left. \frac{\partial g_1}{\partial x_2} \right|_{x_p, u_p} = 0$$

$$d_{11} = \left. \frac{\partial g_1}{\partial u} \right|_{x_p, u_p} = 0$$

This gives the small change model for $y(t)$ as

$$\delta y(t) = \delta x_1(t)$$

and we note once again that applying the linearisation procedure to a linear equation produces the same linear equation, but in terms of the small change variables.

We now consider the nonlinear function given by

$$\dot{x}_2(t) = f_2(x_1(t), x_2(t), u(t))$$

The linear approximation is given by

$$\frac{d(x_{2p} + \delta x_2)}{dt} = f_2(x_{1p}, x_{2p}, u_p) + df_2(x_1(t), x_2(t), u(t))$$

The small change model is given by

$$\delta \dot{x}_2(t) = df_2(x_1(t), x_2(t), u(t)) = a_{21}\delta x_1(t) + a_{22}\delta x_2(t) + b_{21}\delta u(t)$$

where a_{21} , a_{22} and b_{21} are the linear coefficients given by

$$a_{21} = \left. \frac{\partial f}{\partial x_1} \right|_{x_p, u_p} = \left[2a \frac{u^2}{x_1^3} \right]_{x_p, u_p} = 2a \frac{u_p^2}{x_{1p}^3} = \frac{2 \times 9 \times 0.12^2}{0.1101^2} = 21.38$$

$$a_{22} = \left. \frac{\partial f}{\partial x_2} \right|_{x_p, u_p} = [-b] = -9.5$$

$$b_{21} = \left. \frac{\partial f}{\partial u} \right|_{x_p, u_p} = \left[\frac{-2au_p}{x_1} \right]_{x_p, u_p} = \frac{-2au_p}{x_{1p}} = -\frac{2 \times 9 \times 0.12}{0.1101} = -19.62$$

This results in a model

$$\delta \dot{x}_2(t) = 21.38 \delta x_1(t) - 9.5 \delta x_2(t) - 19.62 \delta u(t)$$

3. Form the linear equation

The linear model equations are

$$\delta \dot{x}_1(t) = \delta x_2(t)$$

$$\delta \dot{x}_2(t) = 21.38 \delta x_1(t) - 9.5 \delta x_2(t) - 19.62 \delta u(t)$$

This can now be written in an ABCD linear model, where the coefficients of the ABCD matrices correspond to the linearisation coefficients a_{11} , a_{12} , ...

$$\begin{bmatrix} \delta \dot{x}_1(t) \\ \delta \dot{x}_2(t) \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 21.38 & -9.5 \end{bmatrix} \begin{bmatrix} \delta x_1(t) \\ \delta x_2(t) \end{bmatrix} + \begin{bmatrix} 0 \\ -19.62 \end{bmatrix} \delta u(t)$$

$$\delta y(t) = [1 \quad 0] \delta x(t)$$

Once again, the output of this system only represents the movement of the ball from the operating point of $y_p = x_{1p} = 0.1101$ m. We would have to add this value to the output $\delta y(t)$ to find the actual position $y(t)$ of the ball.

$$y(t) = y_p + \delta y(t)$$

What we have learnt

- ✓ To recognise typical nonlinearities within a system.
- ✓ To identify the operating point, for example $f(x_p, u_p)$, for a linearisation and realise that the variables $\delta x(t)$ and $\delta u(t)$ then represent small variations round that operating condition.
- ✓ To linearise a nonlinear set of dynamical equations:

$$\dot{x}(t) = f(x(t), u(t))$$

$$y(t) = g(x(t), u(t))$$

leading to the linear dynamical model

$$\dot{x}_L(t) = f(x_p, u_p) + df(x(t) - x_p, u(t) - u_p)$$

$$y_L(t) = g(x_p, u_p) + dg(x(t) - x_p, u(t) - u_p)$$

where

$$df(x - x_p, u - u_p) = df(\delta x(t), \delta u(t)) = a\delta x(t) + b\delta u(t)$$

and

$$dg(x - x_p, u - u_p) = dg(\delta x(t), \delta u(t)) = c\delta x(t) + d\delta u(t)$$

with

$$a = \left. \frac{\partial f}{\partial x} \right|_{x_p, u_p}, \quad b = \left. \frac{\partial f}{\partial u} \right|_{x_p, u_p}, \quad c = \left. \frac{\partial g}{\partial x} \right|_{x_p, u_p} \quad \text{and} \quad d = \left. \frac{\partial g}{\partial u} \right|_{x_p, u_p}$$

✓ To recognise the 'small change' model given by

$$df(x - x_p, u - u_p) = df(\delta x(t), \delta u(t)) = a\delta x(t) + b\delta u(t)$$

as representing the small changes of the process variables from the operating points.

Multiple choice

M21.1 If a_1 , a_2 and a_3 are constants, which of the following equations is linear in x ?

- (a) $y = a_1x + a_2x^2 + a_3x^3$
- (b) $y = (a_1 + a_2 + a_3)x$
- (c) $y = \sqrt{x}$
- (d) $y = a_1/x$

M21.2 Saturation effects:

- (a) are found in actuators
- (b) are nonlinear effects
- (c) limit the control output signal
- (d) all of the above

M21.3 In most cases, a nominal linear model can only be used

- (a) over a wide operating range
- (b) over a specified operating range
- (c) at several operating points
- (d) where the system output shows large nonlinearity

M21.4 A linear approximation, at the operating point $x_p = 1.5$, to $y = 3x^2$, is:

- (a) $\delta y = 6\delta x$
- (b) $\delta y = 9\delta x$
- (c) $\delta y = 6.75\delta x$
- (d) $\delta y = 13.5\delta x$

M21.5 The linear function δx_L , at the operating point $u_p = 3$, of $x(u) = 2u^3 + 4$ is:

- (a) $\delta x_L = 6\delta u$
- (b) $\delta x_L = 10\delta u$
- (c) $\delta x_L = 54\delta u$
- (d) $\delta x_L = 58\delta u$

M21.6 A linear approximation, at the operating point $x_p = 0.5$, $u_p = 1.5$, to $y(x, u) = 2x^2 + xu + u^2$, is:

- (a) $\delta y = 3.5\delta x + 3.5\delta u$
- (b) $\delta y = 4\delta x + 2\delta u$
- (c) $\delta y = 2\delta x + 3\delta u$
- (d) $\delta y = 4.75\delta x + 3.75\delta u$

M21.7 If we have defined a linear model $\delta x = 3.4\delta u_1 + 6\delta u_2$, δx represents:

- (a) the value of $x(t)$ at the operating point
- (b) the large variations in $x(t)$ after the linearisation process
- (c) the small variations in $x(t)$ around the operating point
- (d) the change in the operating point

M21.8 The ABCD matrices resulting from a linearisation of a nonlinear model:

- (a) contain the nonlinear terms
- (b) contain the operating points
- (c) contain the linearisation coefficients
- (d) contain the small change inputs

M21.9 If we use a linearised model, the output, δy , of the ABCD system will represent:

- (a) the change in output from the operating point
- (b) the step response of the nonlinear system
- (c) the step response of the linear system
- (d) the change in output from zero

- M21.10** Given a SISO nonlinear model, and a linearisation of this model, then if we double the size of the input step to both the nonlinear and linear system:
- the output from each system will always double
 - the output from the nonlinear system will always double
 - the output from the linear system will always double
 - the output from the nonlinear system will always be double the output from the linear system

Questions: practical skills

Q21.1 Prove that the following functions are linear:

(a) $f(x_1, x_2) = 4x_1 + 7x_2$

(b) $f(x) = \begin{bmatrix} 3.5x_1 - 2.4x_2 \\ 7.2x_1 + 0.5x_2 \end{bmatrix}$ where $x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$

Q21.2 Prove that the following functions are nonlinear:

(a) $f(x) = \frac{x_1}{x_2}$ where $x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$

(b) $f(x, u) = \begin{bmatrix} x_1u_1 + x_2^2 + u_2 \\ x_2 + u_1 \end{bmatrix}$ where $x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$ and $u = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$

Q21.3 The resistance relationship for a thermistor is given by $R = Ae^{-b/T}$.

(a) If the following variables are defined:

$$y = \ln R \text{ and } x = 1/T$$

show that the resistance relationship can be transformed into the linear relationship

$$y = \ln A - bx$$

(b) Suggest what experimental use this transformation might have in calibrating a thermistor device.

Q21.4 A process furnace unit has a dynamical state variable model given by

$$\dot{x}(t) = -10x(t) + 0.2u(t)$$

$$y(t) = 2550x(t)$$

where $u(t)$ represents the fuel flow in m^3/min , $x(t)$ represents the thermocouple output in volts and $y(t)$ represents the measured temperature in $^\circ\text{C}/\text{volt}$

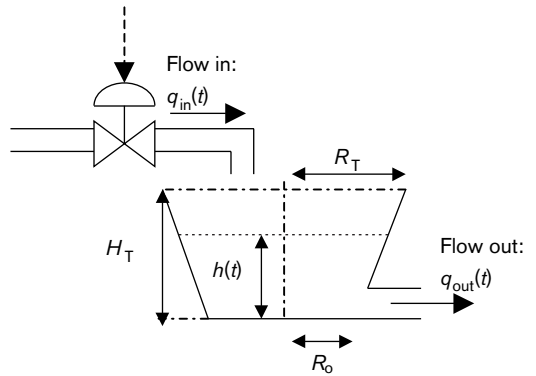
A steady operating condition arises when the fuel flow input is $10 \text{ m}^3/\text{min}$. Determine the furnace operating temperature.

Problem

P21.1 Consider a simple nonlinear tank filling operation. The dimensions of the tank, which is of circular cross-section are shown opposite.

Dimensions, data and features:

- R_o : radius at the tank base = 0.2 m
- R_T : radius at the tank top = 3.2 m
- H_T : height of tank = 3 m
- $q_{in}(t)$: inflow = 0.2 m³/s
- $q_{out}(t)$: outflow = $q_{out} = \sqrt{h(t)} / R_f$
- R_f : outflow proportionality constant = 2.74



(a) Use the volumetric conservation principle:

$$\frac{dV}{dt} = q_{in}(t) - q_{out}(t)$$

to prove

$$\frac{dh}{dt} = \frac{R_f q_{in}(t) - \sqrt{h(t)}}{\pi(\alpha^2 h^2(t) + 2\alpha R_o h(t) + R_o^2) R_f}$$

where

$$\alpha = \frac{R_T - R_o}{H_T}$$

(b) If the level $h(t)$ is defined to be state $x(t)$ and a flow meter is located at the orifice, define the nonlinear system equations

$$\dot{x}(t) = f(x(t), u(t))$$

$$y(t) = g(x(t), u(t))$$

(c) For constant inflow $q_{in}(t) = q_{IN}$, prove that the operating level of the tank satisfies

$$h_p = R_f^2 q_{IN}^2$$

(d) If the operating inflow $q_{IN} = 0.2$ m³/s, derive a linear state variable model for the tank system.

P21.2 The magnetic suspension system has a state variable model given by

$$\dot{x}_1(t) = x_2(t)$$

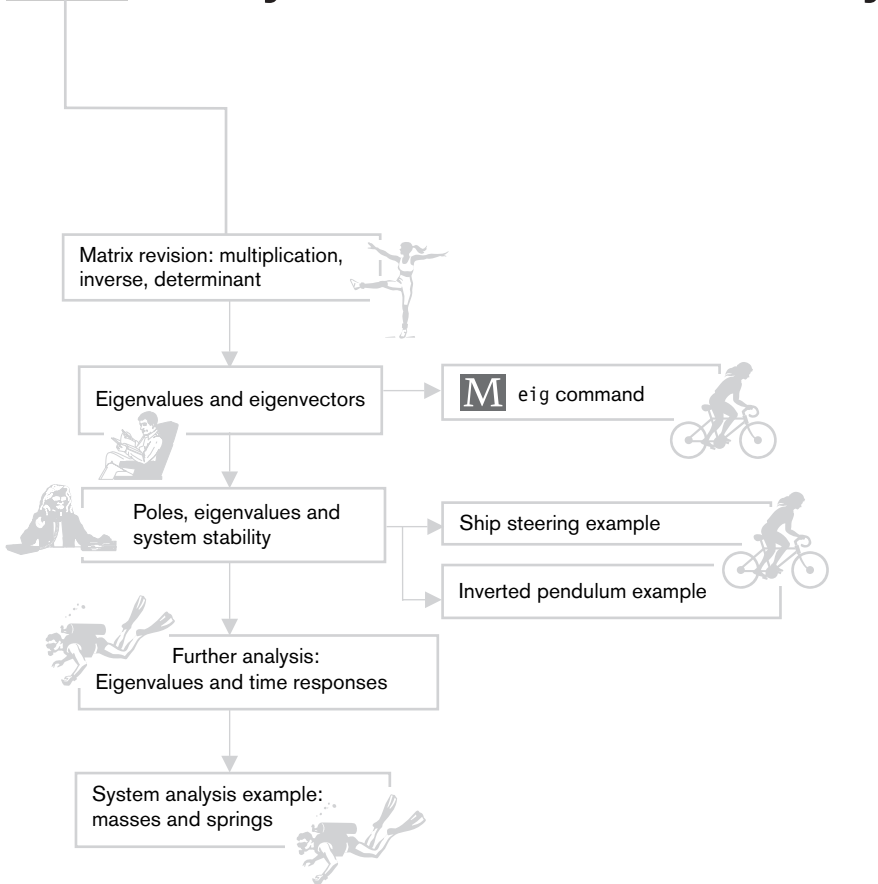
$$\dot{x}_2(t) = g - a \frac{u^2(t)}{x_1^2(t)} - b x_2(t)$$

where the system data gives $a = 9$ units; $b = 9.5$ s⁻¹ and $g = 9.81$ m s⁻². The input for the system was in the range 0.08 to 0.175 A.

(a) Calculate a set of linear models for the input signal changing in steps of 0.02 from 0.08 to 0.18 and examine the change in model parameters over the range.

(b) If a control design is to follow the modelling exercise, which model would you use and why?

22 Analysis of state variable systems



Industrial systems are found in a range of sizes and complexity. Liquid level systems tend to be simple, cold rolling steel mills and aircraft systems tend to be complicated, and utility systems like water distribution networks tend to be large-scale applications of process units that are in themselves fairly simple. What we have learnt, though, from the previous chapter is that the realistic system models are generally nonlinear. We therefore spent some considerable time and intellectual effort understanding the method of finding a representative linear model so that we can use linear control techniques. Even before that we learnt the rather abstract methods of how to represent a control system model in the concise state variable format. All this is fairly mathematical, and by concentrating on these formal processes we have perhaps lost sight of the insight we gained from the development and manipulation of transfer functions, and our knowledge of the system poles and zeros and their relation to system stability. We address these issues in this chapter by providing some analysis of our state variable model and, in particular, linking our results on stability to those we developed for transfer function models.

It turns out that the *eigenvalues* of the system model matrix, \mathbf{A} , are important indicators of the stability of a system. This means that matrix operations and matrix analysis are going to be very important in this chapter. We begin the chapter with some revision material on these topics. We look at matrix multiplication, the matrix determinant and inverse matrix calculations. We then introduce eigenvalues and eigenvectors. It might seem surprising, but the more abstract expressions for many of the results in this chapter say exactly what needs to be said with far fewer symbols than long-winded explanations. But we also find that we need to practice finding deeper interpretations in our mathematical formulas, and this takes time and patience. MATLAB and Simulink can remove one difficulty from these issues by allowing us direct calculation of many of the more abstract matrix operations and expressions, and we have included commands where appropriate.

Learning objectives

- To revise some elementary matrix operations.
- To introduce eigenvalues and eigenvectors.
- To relate eigenvalues to system poles and system stability.
- To analyse state variable system time responses in more detail.

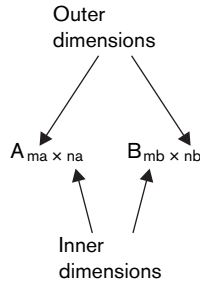
22.1 Matrix revision

We look at some of the elementary rules for matrix multiplication, and then at how to calculate the determinant and the inverse of a 2×2 matrix. These formulas will be useful for simple bookwork examples. For more complicated examples we would use the MATLAB commands given.

22.1.1 Matrix multiplication

Matrix multiplication is not always defined, so we must be far more careful to check that the dimensions of any matrix expression are compatible.

Compatibility rule: if matrix **A** has ma rows and na columns, and matrix **B** has mb rows and nb columns, then to be able to form the matrix product, **AB**, the inner dimensions must be equal:



that is, $na = mb$. The resulting matrix, **AB**, will have dimensions given by the outer dimensions of the product, $ma \times nb$.

Skill section

For the following matrix multiplications, **AB**, **BC**, **CB**, **BD**, **EC**, **CE**, we need to check whether the proposed product is an allowable matrix calculation. When we have a compatible product, we will perform the required calculation.

$$\mathbf{A} = \begin{bmatrix} 1 & -1 \\ 2 & 3 \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} 3 \\ 2 \end{bmatrix}, \quad \mathbf{C} = [3 \ 2 \ 5], \quad \mathbf{D} = \begin{bmatrix} 0 & 7 \\ -1 & 0 \end{bmatrix}, \quad \mathbf{E} = \begin{bmatrix} 1 & 0 \\ 3 & 4 \\ 5 & -1 \end{bmatrix}$$

We first check the dimensions of the matrices and determine which of the products are compatible (Table 22.1).

Table 22.1 Matrix multiplication skills test.

Dimensions of first matrix	Dimensions of second matrix	Are inner dimensions equal?	Is product allowable?	If yes, what are the dimensions of the final matrix product (outer dimensions)?
A: 2×2	B: 2×1	Yes	Yes	AB: 2×1
B: 2×1	C: 1×3	Yes	Yes	BC: 2×3
C: 1×3	B: 2×1	No	No	
B: 2×1	D: 2×2	No	No	
E: 3×2	C: 1×3	No	No	
C: 1×3	E: 3×2	Yes	Yes	CE: 1×2

The allowable calculations are:

$$\mathbf{AB} = \begin{bmatrix} 1 \\ 12 \end{bmatrix}, \quad \mathbf{BC} = \begin{bmatrix} 9 & 6 & 15 \\ 6 & 4 & 10 \end{bmatrix}, \quad \mathbf{CE} = [34 \ 3]$$

When we manipulate matrix expressions we must ensure that the dimensions of the resulting equations are compatible. We must preserve this compatibility property when we use the operations of matrix pre-multiplication and matrix post-multiplication. Thus we have to bring extra rigour to our thinking about matrix equations, because matrix multiplication is not commutative. This means that if \mathbf{A} and \mathbf{B} are compatible and both products \mathbf{AB} and \mathbf{BA} can be defined, then, in general, $\mathbf{AB} \neq \mathbf{BA}$.

Example Let

$$\mathbf{A} = \begin{bmatrix} 1 & 3 \\ 2 & -3 \end{bmatrix} \text{ and } \mathbf{B} = \begin{bmatrix} 4 & 1 \\ 0 & -2 \end{bmatrix}$$

Then, since both \mathbf{A} and \mathbf{B} are 2×2 matrices, their products \mathbf{AB} and \mathbf{BA} will be defined. However, we find that

$$\mathbf{AB} = \begin{bmatrix} 1 & 3 \\ 2 & -3 \end{bmatrix} \begin{bmatrix} 4 & 1 \\ 0 & -2 \end{bmatrix} = \begin{bmatrix} 4 & -5 \\ 8 & 8 \end{bmatrix}$$

and

$$\mathbf{BA} = \begin{bmatrix} 4 & 1 \\ 0 & -2 \end{bmatrix} \begin{bmatrix} 1 & 3 \\ 2 & -3 \end{bmatrix} = \begin{bmatrix} 6 & 9 \\ -4 & 6 \end{bmatrix}$$

Obviously $\mathbf{AB} \neq \mathbf{BA}$.

Thus, we cannot interchange the order of matrix multiplication at will in the way that we can for our usual scalar algebra, because this will invalidate the matrix expression or equation.

22.1.2 Inverse matrix calculation

The equivalent of '1' in matrices is the unit matrix or the identity matrix. The identity matrix is a square matrix that contains ones on the diagonal and zeros elsewhere. It is usually given the symbol \mathbf{I} , sometimes it is written as $\mathbf{I}_{n \times n}$, where the subscript shows the size of the unit matrix being used. For example, the identity matrix $\mathbf{I}_{3 \times 3}$ is given by:

$$\mathbf{I}_{3 \times 3} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

We use the identity matrix to define the inverse matrix when it exists.

Inverse matrix: any square ($n \times n$) matrix, \mathbf{A} , that has an inverse matrix, denoted \mathbf{A}^{-1} , produces an ($n \times n$) identity matrix from the following multiplication operations:

$$\mathbf{AA}^{-1} = \mathbf{A}^{-1}\mathbf{A} = \mathbf{I}_{n \times n}$$

Matrix pre-multiplication: Let \mathbf{A} ($n \times n$), \mathbf{B} ($n \times r$) and \mathbf{X} ($n \times r$) be matrices such that \mathbf{A}^{-1} exists and that $\mathbf{AX} = \mathbf{B}$. Then to find \mathbf{X} , we must *pre-multiply* both sides by \mathbf{A}^{-1} :

$$\mathbf{AX} = \mathbf{B}$$

$$\mathbf{A}^{-1} \times (\mathbf{AX}) = \mathbf{A}^{-1} \times \mathbf{B}$$

$$\mathbf{A}^{-1}\mathbf{AX} = \mathbf{A}^{-1}\mathbf{B}$$

$$\mathbf{I}_{n \times n}\mathbf{X} = \mathbf{A}^{-1}\mathbf{B}$$

$$\mathbf{X} = \mathbf{A}^{-1}\mathbf{B}$$

Matrix post-multiplication: Let \mathbf{A} ($n \times n$), \mathbf{B} ($r \times n$) and \mathbf{X} ($r \times n$) be matrices such that \mathbf{A}^{-1} exists and that $\mathbf{XA} = \mathbf{B}$. Then to find \mathbf{X} , we must *post-multiply* both sides by \mathbf{A}^{-1} :

$$\mathbf{XA} = \mathbf{B}$$

$$(\mathbf{XA}) \times \mathbf{A}^{-1} = \mathbf{B} \times \mathbf{A}^{-1}$$

$$\mathbf{XAA}^{-1} = \mathbf{BA}^{-1}$$

$$\mathbf{XI}_{n \times n} = \mathbf{BA}^{-1}$$

$$\mathbf{X} = \mathbf{BA}^{-1}$$

Determinant of a 2×2 matrix: If $\mathbf{A} = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$,

then the determinant is calculated as $\det(\mathbf{A}) = ad - bc$. The determinant of matrix \mathbf{A} is denoted by $\det(\mathbf{A})$.

Inverse of a 2×2 matrix: If matrix $\mathbf{A} = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$,

and $\det(\mathbf{A}) \neq 0$, then the inverse matrix, \mathbf{A}^{-1} , is calculated as

$$\mathbf{A}^{-1} = \frac{1}{\det(\mathbf{A})} \begin{bmatrix} d & -b \\ -c & a \end{bmatrix} = \frac{1}{ad-bc} \begin{bmatrix} d & -b \\ -c & a \end{bmatrix}$$

This can be verified by proving that $\mathbf{AA}^{-1} = \mathbf{A}^{-1}\mathbf{A} = \mathbf{I}_{2 \times 2}$.

Inverse of an $n \times n$ matrix: We can generalise the 2×2 matrix result as follows. If matrix $\mathbf{A} = [a_{ij}]$ and $\det(\mathbf{A}) \neq 0$, then the inverse matrix, \mathbf{A}^{-1} , is formally calculated as

$$\mathbf{A}^{-1} = \frac{\text{adj}(\mathbf{A})}{\det(\mathbf{A})}$$

The numerator term, $\text{adj}(\mathbf{A})$, in the expression is known as the adjoint matrix. In words, the adjoint matrix is the transposed matrix of signed co-factors of matrix \mathbf{A} .

M The MATLAB command for matrix inversion is simple. To find the inverse of matrix \mathbf{A} , we write

$$\text{inv}(\mathbf{A})$$

Skill section

Simple 2×2 matrix examples will be found in your class tests, assessed tutorials and even examinations, so it is really useful to polish your 2×2 matrix skills.

For the matrices

$$\mathbf{A} = \begin{bmatrix} 1 & -1 \\ 2 & 3 \end{bmatrix}, \quad \mathbf{D} = \begin{bmatrix} 0 & 7 \\ -1 & 0 \end{bmatrix}$$

we look at the determinant and, if possible, the matrix inverse calculations.

$$\mathbf{A} = \begin{bmatrix} 1 & -1 \\ 2 & 3 \end{bmatrix}, \quad \det(\mathbf{A}) = (1 \times 3 - (-1) \times 2) = 5$$

Since $\det(\mathbf{A}) \neq 0$, we can compute the matrix inverse as

$$\mathbf{A}^{-1} = \left(\frac{1}{5}\right) \begin{bmatrix} 3 & 1 \\ -2 & 1 \end{bmatrix}$$

$$\mathbf{D} = \begin{bmatrix} 0 & 7 \\ -1 & 0 \end{bmatrix}, \det(\mathbf{D}) = (0 \times 0 - 7 \times (-1)) = 7$$

Since $\det(\mathbf{D}) \neq 0$, we can compute the matrix inverse as

$$\mathbf{D}^{-1} = \left(\frac{1}{7}\right) \begin{bmatrix} 0 & -7 \\ 1 & 0 \end{bmatrix}$$

Many matrices have structure in the elements that make some matrix calculations simpler. Consider calculating \mathbf{F}^{-1} , where

$$\mathbf{F} = \begin{bmatrix} 3 & 0 \\ 0 & -2 \end{bmatrix}$$

The matrix \mathbf{F} is *diagonal*, so that only the diagonal elements of the matrix have non-zero values. If we compute the determinant, we find

$$\det(\mathbf{F}) = (3 \times (-2) - 0 \times 0) = -6$$

Since $\det(\mathbf{F}) \neq 0$, we can compute the matrix inverse as

$$\mathbf{F}^{-1} = \left(\frac{1}{-6}\right) \begin{bmatrix} -2 & 0 \\ 0 & 3 \end{bmatrix} = \begin{bmatrix} \frac{1}{3} & 0 \\ 0 & -\frac{1}{2} \end{bmatrix}$$

We find that the inverse of the diagonal matrix \mathbf{F} is just the matrix \mathbf{F} with the diagonal entries inverted. We could have easily written this down without completing the formal inversion calculation.

22.2 Eigenvalues and eigenvectors

For a square matrix \mathbf{A} which is of dimension $n \times n$ we can find a set of n eigenvalues and eigenvectors. Eigenvalue and eigenvector tell us about properties of the \mathbf{A} matrix that are not immediately obvious from just looking at the matrix. The equation which defines an eigenvector and its corresponding eigenvalue takes the form

$$\mathbf{A}\mathbf{x} = \lambda\mathbf{x}, \mathbf{x} \neq 0$$

The eigenvector \mathbf{x} has dimensions of $n \times 1$ and the eigenvalue λ is a scalar. We can interpret the eigenvector as a vector direction that, when multiplied by \mathbf{A} , gives a scalar multiple, λ , of itself.

22.2.1 Using determinants to find eigenvalues

We can rewrite the eigenvalue–eigenvector equation as

$$(\lambda\mathbf{I} - \mathbf{A})\mathbf{x} = 0$$

where \mathbf{I} is an identity matrix of the same dimension as \mathbf{A} . Any value of λ which makes the determinant of $\lambda\mathbf{I} - \mathbf{A}$ zero is a solution of the above equation. We call the solutions the eigenvalues.

Key result: Eigenvalues of matrix \mathbf{A}

To find the eigenvalues of a matrix \mathbf{A} , we solve the following equation

$$\det(\lambda\mathbf{I} - \mathbf{A}) = 0$$

This method leads to an n th order polynomial equation for which the n solutions will be the eigenvalues of the matrix \mathbf{A} . We would only recommend this method for low values of n , say $n = 2$ or 3 . This polynomial link also leads us to realise that a real matrix \mathbf{A} can have complex eigenvalues and that these always occur in complex conjugate pairs.

22.2.2 Using eigenvectors to diagonalise a matrix

If \mathbf{A} is a 3×3 matrix for which we have found the three eigenvalue–eigenvector pairs $(\lambda_1, \mathbf{m}_1)$, $(\lambda_2, \mathbf{m}_2)$, $(\lambda_3, \mathbf{m}_3)$, then we can introduce an eigenvector matrix, \mathbf{M} , defined by $\mathbf{M} = [\mathbf{m}_1, \mathbf{m}_2, \mathbf{m}_3]$. We can then set up an equation where we pre-multiply \mathbf{M} by matrix \mathbf{A} and obtain the following analysis:

$$\begin{aligned} \mathbf{AM} &= \mathbf{A}[\mathbf{m}_1 \quad \mathbf{m}_2 \quad \mathbf{m}_3] = [\mathbf{A}\mathbf{m}_1 \quad \mathbf{A}\mathbf{m}_2 \quad \mathbf{A}\mathbf{m}_3] \\ &= [\lambda_1\mathbf{m}_1 \quad \lambda_2\mathbf{m}_2 \quad \lambda_3\mathbf{m}_3] = [\mathbf{m}_1\lambda_1 \quad \mathbf{m}_2\lambda_2 \quad \mathbf{m}_3\lambda_3] \\ &= [\mathbf{m}_1 \quad \mathbf{m}_2 \quad \mathbf{m}_3] \begin{bmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \lambda_3 \end{bmatrix} \\ &= \mathbf{M}\mathbf{\Lambda} \end{aligned}$$

where

$$\mathbf{\Lambda} = \begin{bmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \lambda_3 \end{bmatrix}$$

is a diagonal matrix of eigenvalues.

It is not difficult to realise that this analysis will generalise to the $n \times n$ case. We can use the relationship $\mathbf{AM} = \mathbf{M}\mathbf{\Lambda}$ in two different ways:

- (a) Post-multiply by \mathbf{M}^{-1} to give $\mathbf{A} = \mathbf{M}\mathbf{\Lambda}\mathbf{M}^{-1}$
- (b) Pre-multiply by \mathbf{M}^{-1} to give $\mathbf{M}^{-1}\mathbf{AM} = \mathbf{\Lambda}$

Case (b) illustrates that we can use the inverse of the eigenvector matrix, \mathbf{M} , to diagonalise the matrix \mathbf{A} .

M 22.2.3 MATLAB eig command

Evaluating eigenvalues and eigenvectors is time-consuming to do for more than a simple second-order system, and the following MATLAB commands are useful:

- The command `eig(A)` produces a vector containing the eigenvalues of a square matrix **A**.
- `[M,lambda] = eig(A)` produces a diagonal matrix **lambda** of eigenvalues and a full matrix **M** whose columns are the corresponding eigenvectors, so that

$$\mathbf{A} * \mathbf{M} = \mathbf{M} * \mathbf{lambda}$$

Problem Use the MATLAB command `eig(A)` to calculate the eigenvalues and eigenvectors for

$$\mathbf{A} = \begin{bmatrix} 1 & 0 & 3 \\ 4 & 6 & 9 \\ -7 & 1 & 0 \end{bmatrix}$$

Solution We enter the matrix **A** first as

$$A = [1 \ 0 \ 3; \ 4 \ 6 \ 9; \ -7 \ 1 \ 0];$$

We could simply type

$$\text{eig}(A)$$

which would give us the following vector of eigenvalues:

$$\begin{aligned} & -0.0222 + 4.2793i \\ & -0.0222 - 4.2793i \\ & 7.0444 \end{aligned}$$

We note that we have two complex eigenvalues and one real eigenvalue. To find the eigenvectors, we use

$$[M, \text{lambda}] = \text{eig}(A);$$

The matrix **lambda** will give us a square matrix with the eigenvalues on the diagonal and the eigenvector matrix **M** will appear as:

$$\mathbf{M} = \begin{bmatrix} -0.3552 - 0.1884i & -0.3552 + 0.1884i & 0.0469 \\ -0.6015 + 0.3589i & -0.6015 - 0.3589i & 0.9944 \\ 0.3897 - 0.4425i & 0.3897 + 0.4425i & 0.0945 \end{bmatrix}$$

Note that this has three eigenvector columns, and that two of the eigenvectors are complex vectors.

In some matrices the eigenvalues can simply be read from the matrix, for example:

1. If **A** is diagonal, the eigenvalues will be equal to the diagonal entries:

$$\mathbf{A} = \begin{bmatrix} -0.22 + 4.28i & 0 & 0 \\ 0 & -0.22 - 4.28i & 0 \\ 0 & 0 & 7.04 \end{bmatrix}$$

has eigenvalues $-0.22 + 4.28i$, $-0.22 - 4.28i$, 7.04 .

2. Matrices with upper or lower triangular structure have eigenvalues as the diagonal entries.

$$\mathbf{A}_U = \begin{bmatrix} 4.8 & 0.5 & 7.2 \\ 0 & -0.22 + 8.5i & 0.6 \\ 0 & 0 & -2.2 + 8.5i \end{bmatrix}$$

has eigenvalues 4.8, $-2.2 + 8.5i$, $-2.2 - 8.5i$.

$$\mathbf{A}_L = \begin{bmatrix} 8.1 & 0 & 0 \\ 5.2 & 3.9 & 0 \\ 5.6 & 2.8 & -12.8 \end{bmatrix}$$

has eigenvalues 8.1, 3.9, -12.8 .

22.3 Poles, eigenvalues and system stability

We have found that many investigations in which we have developed models for industrial systems and components benefited from using the extra precision of a state variable representation. The first stage in developing these system models was the routine step of finding the state variable matrices, \mathbf{A} , \mathbf{B} , \mathbf{C} and \mathbf{D} , but once we have these matrices, what can they tell us about the system? To obtain a deeper understanding of this we have to further develop our state variable analysis. In the sections above we revised some matrix concepts and introduced the idea of eigenvalues and eigenvectors of a matrix. Now we will link the system eigenvalues to the poles of a system transfer function. Once we have this link we can make a direct connection to system stability. Let us assume that we have developed a state variable system model, and that, as often happens, the \mathbf{D} matrix of the model is zero, so we have the equations

$$\begin{aligned} \dot{\mathbf{x}}(t) &= \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t) \\ \mathbf{y}(t) &= \mathbf{C}\mathbf{x}(t) \end{aligned}$$

We will introduce some standard Laplace transform relations for the state, output and control vectors, as

$$\begin{aligned} \text{State vector transform} & \quad \mathbf{X}(s) = \mathcal{L}\{\mathbf{x}(t)\} \\ \text{Output vector transform} & \quad \mathbf{Y}(s) = \mathcal{L}\{\mathbf{y}(t)\} \\ \text{Control vector transform} & \quad \mathbf{U}(s) = \mathcal{L}\{\mathbf{u}(t)\} \end{aligned}$$

We also use the Laplace transform of the derivative of the state vector, as

$$\mathcal{L}\{\dot{\mathbf{x}}(t)\} = s\mathbf{X}(s) - \mathbf{x}_0$$

where $\mathbf{X}(s)$ is the state vector transform and \mathbf{x}_0 is an initial condition for the state vector. If we now take the Laplace transform of the state variable system model, we obtain

$$\begin{aligned} \mathcal{L}\{\dot{\mathbf{x}}(t)\} &= \mathcal{L}\{\mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t)\} = \mathbf{A}\mathcal{L}\{\mathbf{x}(t)\} + \mathbf{B}\mathcal{L}\{\mathbf{u}(t)\} \\ s\mathbf{X}(s) - \mathbf{x}_0 &= \mathbf{A}\mathbf{X}(s) + \mathbf{B}\mathbf{U}(s) \end{aligned}$$

We must now use our matrix manipulation skills to rearrange this as follows:

$$\begin{aligned} s\mathbf{X}(s) - \mathbf{A}\mathbf{X}(s) &= \mathbf{x}_0 + \mathbf{B}\mathbf{U}(s) \\ (s\mathbf{I} - \mathbf{A})\mathbf{X}(s) &= \mathbf{x}_0 + \mathbf{B}\mathbf{U}(s) \end{aligned}$$

We pre-multiply by the inverse matrix, $(s\mathbf{I} - \mathbf{A})^{-1}$, to obtain the state transfer function

$$\mathbf{X}(s) = (s\mathbf{I} - \mathbf{A})^{-1}\mathbf{x}_0 + (s\mathbf{I} - \mathbf{A})^{-1}\mathbf{B}\mathbf{U}(s)$$

Now to Laplace transform the output equation:

$$\begin{aligned} \mathcal{L}(\mathbf{y}(t)) &= \mathcal{L}(\mathbf{C}\mathbf{x}(t)) = \mathbf{C}\mathcal{L}(\mathbf{x}(t)) \\ \mathbf{Y}(s) &= \mathbf{C}\mathbf{X}(s) \end{aligned}$$

If we set the initial condition for the state to zero (so that $\mathbf{x}_0 = \mathbf{0}$), we obtain the transfer functions of the state variable system equations as

$$\begin{aligned} \text{State equation: } \quad \mathbf{X}(s) &= (s\mathbf{I} - \mathbf{A})^{-1}\mathbf{B}\mathbf{U}(s) \\ \text{Output equation: } \quad \mathbf{Y}(s) &= \mathbf{C}\mathbf{X}(s) \end{aligned}$$

We can combine these two equations to find the system input–output transfer function relation as

$$\mathbf{Y}(s) = \mathbf{C}(s\mathbf{I} - \mathbf{A})^{-1}\mathbf{B}\mathbf{U}(s)$$

The system transfer function is then identified as

$$\mathbf{G}(s) = \mathbf{C}(s\mathbf{I} - \mathbf{A})^{-1}\mathbf{B}$$

We should note that, in general, we have arrived at a *matrix* transfer function $\mathbf{G}(s)$ with dimensions $r \times m$, where r and m are the numbers of output and inputs respectively. We mainly deal with so-called square systems in this book, where the numbers of inputs and outputs are the same. If we use the formula for the $n \times n$ matrix inverse that we revised earlier, we can give a result for this system transfer function as

$$\mathbf{G}(s) = \mathbf{C}(s\mathbf{I} - \mathbf{A})^{-1}\mathbf{B} = \mathbf{C} \frac{\text{adj}(s\mathbf{I} - \mathbf{A})}{\det(s\mathbf{I} - \mathbf{A})} \mathbf{B} = \frac{\mathbf{N}(s)}{d(s)}$$

where the numerator polynomial matrix is $\mathbf{N}(s) = \mathbf{C}\text{adj}(s\mathbf{I} - \mathbf{A})\mathbf{B}$ and the denominator polynomial is $d(s) = \det(s\mathbf{I} - \mathbf{A})$. The poles of the transfer function $\mathbf{G}(s)$ are then roots of the denominator polynomial, $d(s)$. Thus we have the first link, that to find the poles of the transfer function of the state variable system model we must solve

$$d(s) = \det(s\mathbf{I} - \mathbf{A}) = 0$$

This clearly links the system poles to the system matrix, \mathbf{A} . But we also know, from the section above on matrix eigenvalues, that the eigenvalues of matrix \mathbf{A} are given by $\det(\lambda\mathbf{I} - \mathbf{A})$. We conclude, therefore, that the pole locations are precisely the eigenvalues of the system matrix, \mathbf{A} . We can now make the final connection with system stability and proceed to use the eigenvalues of system matrix \mathbf{A} to determine the system stability.

Key result: Link between model type and system stability

Representation of model equations	Stability determined by
Laplace transform descriptions	Poles of transfer functions
State variables	Eigenvalues of \mathbf{A} matrix

We make a formal statement about system stability:

System stability: We call the state variable system model given by $(\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D})$ stable if and only if all the eigenvalues of the system matrix \mathbf{A} have negative real parts. We write this precisely as requiring that $\text{Re}(\lambda_i(\mathbf{A})) < 0$ for $i = 1, \dots, n$.

Example A state variable model for a compressor unit is given by the following data:

$$\mathbf{A} = \begin{bmatrix} -1.78 & -1.39 \\ 0.81 & -3.22 \end{bmatrix}, \mathbf{B} = \begin{bmatrix} 1.2 \\ 2.3 \end{bmatrix} \text{ and } \mathbf{C} = [1 \quad 4]$$

We first find the transfer function model using the MATLAB command

$$[\text{num}, \text{den}] = \text{ss2tf}(\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}, 1)$$

where we have used $\mathbf{D} = 0$ and a '1' in the `ss2tf` command since we are using the first input (in our case we have only one input!). This produces the following output:

```
[num,den]=ss2tf(a,b,c,d,1)
num =
    0 10.4000 19.7810
den =
    1.0000 5.0000 7.2625
```

This gives us the information to write the transfer function model as

$$G(s) = \frac{10.4000s + 9.7810}{1.0000s^2 + 5.0000s + 7.2625}$$

Since the poles are the roots of the denominator of the transfer function, we use the following MATLAB command to obtain the two transfer function poles:

```
roots([1 5.0 7.2625])
ans =
   -2.5000 + 1.0062i
   -2.5000 - 1.0062i
```

This system is stable because we can see that the poles have negative real parts and therefore lie in the LHP of the s -domain. We check this system stability result, but this time using the system eigenvalues. First, we compute the eigenvalues of the system matrix, \mathbf{A} , and we use MATLAB to do this.

We enter the matrix \mathbf{A} first as

$$\mathbf{A} = [-1.78 \quad -1.89; 0.81 \quad -3.22]$$

We simply type

```
eig(A)
ans =
   -2.5000 + 1.0062i
   -2.5000 - 1.0062i
```

These eigenvalues are the same as the transfer function poles, which is what our theory predicts. Furthermore, the eigenvalues have negative real parts, so we are able to declare the system stable.

Problem: Stability of ship steering system

The aim of a ship steering system is to follow a prescribed course with minimal deviation due to wind, waves or current. However, in reality there is a trade-off between excessive use of the rudder to maintain a *tight* heading and wear and tear on the actuation systems. Although the ship has *six degrees of freedom*, that is, it has six independent motions, the ship steering problem involves only three of these motions: the forward surge motion, $u(t)$, the sideways velocity, $v(t)$, and the rotational motion, called the yaw rate, $r(t)$. A change in rudder angle, $\delta(t)$ will produce a

change in heading angle, $\psi(t)$. We note that the yaw rate, $r(t)$, is the derivative of the heading angle. In this problem we examine the stability of the system in terms of the eigenvalues and the poles of the vessel's model. A particular 250 000 tonne tanker at full load has the following state variable description:

$$\begin{bmatrix} \dot{v}(t) \\ \dot{r}(t) \end{bmatrix} = \begin{bmatrix} -0.13 & -8.6 \\ -9.3 \times 10^{-5} & -4.3 \times 10^{-2} \end{bmatrix} \begin{bmatrix} v(t) \\ r(t) \end{bmatrix} + \begin{bmatrix} 7.03 \times 10^{-2} \\ -4.59 \times 10^{-4} \end{bmatrix} \delta(t)$$

$$\begin{bmatrix} v(t) \\ r(t) \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v(t) \\ r(t) \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \end{bmatrix} \delta(t)$$

Use MATLAB to examine the stability of the system:

- (a) by finding the system matrix eigenvalues
- (b) by transforming the system to transfer function form and finding the poles of the system.

Solution (a) We identify matrix **A** as **A** = [-0.13 -8.6; -9.3 × 10⁻⁵ -4.3 × 10⁻²]. The MATLAB command eig(A) produces two eigenvalues:

```
0.00401249755955
-0.06001249755955
```

We see that both eigenvalues are real, so the stability condition relies on whether an eigenvalue is positive (unstable) or negative (stable). We immediately see that one of the real eigenvalues is positive and hence that overall this model represents an unstable system.

(b) Using the following MATLAB command we can produce a transfer function model:

```
[NUM,DEN] = ss2tf(A,B,C,D,1)
```

This produces the following MATLAB output:

```
NUM =
    0    0.07030000000000    0.00697030000000
    0   -0.00045900000000   -0.00001250490000
DEN =
    1.00000000000000    0.05600000000000   -0.00024080000000
```

This gives us the information to write the following model transfer function description:

$$\begin{bmatrix} v(t) \\ r(t) \end{bmatrix} = \begin{bmatrix} \frac{0.703s + 0.0069703}{s^2 + 0.056s - 2.408 \times 10^{-4}} \\ \frac{-4.59 \times 10^{-3}s - 1.25 \times 10^{-5}}{s^2 + 0.056s - 2.408 \times 10^{-4}} \end{bmatrix} \delta(t)$$

Since the poles are the roots of the denominator transfer function and both denominators are the same, we use the following MATLAB command:

```
roots([1 0.056 -2.408e-4])
ans =
   -0.0600
    0.0040
```

These poles are the same as the eigenvalues, which is what we would expect from our theory. The poles are wholly real and one lies in the LHP ($s = -0.06$), indicating a stable pole, and one lies in the RHP ($s = 0.004$), indicating an unstable pole. Overall the system is therefore unstable.

Problem: An inverted pendulum system

Figure 22.1 shows a cart with an inverted pendulum. This is a typical problem given to many control students. The cart is driven by the force F and has a state variable model given by

$$\dot{\mathbf{x}}(t) = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & -0.2 & 2.7 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & -0.5 & 31 & 0 \end{bmatrix} \mathbf{x}(t) + \begin{bmatrix} 0 \\ 1.8 \\ 0 \\ 4.5 \end{bmatrix} \mathbf{u}(t)$$

$$\mathbf{y}(t) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \mathbf{x}(t)$$

where $\mathbf{y} = [y_1(t) \ y_2(t)]^T$, with $y_1(t)$ as the cart position and $y_2(t)$ as the pendulum position, and $\mathbf{u}(t)$ is the force input, $F(t)$.

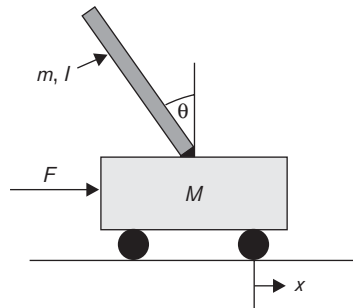


Figure 22.1 Inverted pendulum.

- Calculate the transfer functions from $U(s)$ to $Y_1(s)$ and $Y_2(s)$.
- Calculate the poles of the open-loop system. Comment on the stability of the system.
- Calculate the eigenvalues of the system and verify that they are the same as the system poles. Comment on the stability of the system.

Solution (a) System transfer function

Note that the system has two outputs: the cart position in metres and the pendulum position in radians. We are seeking a transfer function expression of the form

$$\begin{pmatrix} Y_1(s) \\ Y_2(s) \end{pmatrix} = \begin{pmatrix} g_{11}(s) \\ g_{12}(s) \end{pmatrix} U(s)$$

We use MATLAB to calculate the transfer function. We enter matrices **A**, **B**, **C** and **D** and use the command:

```
[num,den]=ss2tf(A,B,C,D,1);
g11 = tf(num(1,:),den)
```

We note that in this case, num is a matrix of numerator polynomials.

Ignoring very small coefficients, the transfer function element, $g_{11}(s)$ is given by

$$g_{11}(s) = \frac{1.818s^2 - 44.55}{s^4 + 0.1818s^3 - 31.18s^2 - 4.455s}$$

The transfer function element $g_{21}(s)$ is given by the following command:

```
tf(num(2,:),den)
```

This produces, approximately,

$$g_{21}(s) = \frac{4.455s^2}{s^4 + 0.1818s^3 - 31.18s^2 - 4.455s}$$

(b) Open-loop poles

The poles are the roots of the denominator polynomials, which are the same for both g_{11} and g_{21} . Using the MATLAB roots command gives

```
roots([1 0.1818 -31.18 -4.455 0])
```

$$p_1 = 0, p_2 = -5.6039, p_3 = 5.565, p_4 = -0.1429$$

We can see that the open-loop system is unstable, with one pole (p_3) in the RHP and one pole (p_1) at the origin.

(c) System eigenvalues

The MATLAB command `eig(A)` gives

$$\lambda_1 = 0, \lambda_2 = -5.6039, \lambda_3 = 5.565, \lambda_4 = -0.1429$$

The eigenvalues are the same as the system poles, and we see that they are all real. To assess the stability status, we only have to check whether the eigenvalues are negative (stable) or positive (unstable). We can see that the open-loop system is unstable with one positive eigenvalue ($\lambda_3 = 5.565$) and one eigenvalue at the origin ($\lambda_1 = 0$).

To close this section on the links between system transfer function poles, state variable models and system matrix eigenvalues we must say that there is still more to be said on these connections. However, these deeper issues must be left to other courses and books. We now turn to look at the use we can make of the property where the eigenvector matrix diagonalises a state variable system matrix.

22.4 More on state variable system time responses

We have already learnt that the state variable representation is not unique, and that for the same input–output system behaviour we can have different internal state descriptions. However, we must not get confused over this. If we derive a model for a physical system, then the state equations must and will represent the actual physical processes taking place. What we are saying is that *mathematically* we can find many different internal state representations for a given input–output representation. Within this set of multiple state representations some things are fixed, and that is where the eigenvalue–eigenvector theory enters the picture. It is the eigenvalues that are fixed inside all this flexibility of mathematical structure, and, as we have seen, the eigenvalues can be linked to system poles and system stability. In this next short piece of analysis we

show how the time domain state variable equations can be unravelled to reveal the internal dynamical structure and the relationship to the system eigenvalues.

22.4.1 System eigenvalues and state variable system dynamics

We consider how to transform the state variable system into one where we have a diagonal system matrix, \mathbf{A} . This is found to be useful, since the eigenvalues of a diagonal matrix are simply the elements on the diagonal. We start from the state variable equation:

$$\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t)$$

with an initial condition, \mathbf{x}_0 .

We have learnt that the system matrix, \mathbf{A} , has a set of n eigenvalue–eigenvector pairs, $(\lambda_i, \mathbf{m}_i)$, $i = 1, \dots, n$. From the eigenvectors, we can form an eigenvector matrix defined as $\mathbf{M} = [\mathbf{m}_1, \mathbf{m}_2, \dots, \mathbf{m}_n]$. If we pre-multiply \mathbf{M} by matrix \mathbf{A} , we can derive the equation $\mathbf{A}\mathbf{M} = \mathbf{M}\mathbf{\Lambda}$, where the matrix $\mathbf{\Lambda}$ is a diagonal matrix of the eigenvalues. Typically we have:

$$\mathbf{\Lambda} = \begin{bmatrix} \lambda_1 & 0 & \dots & 0 \\ 0 & \dots & \dots & \dots \\ \dots & \dots & \dots & 0 \\ 0 & \dots & 0 & \lambda_n \end{bmatrix}$$

Further analysis leads to an eigenvalue–eigenvector expression for the system matrix, \mathbf{A} , given by

$$\mathbf{A} = \mathbf{M}\mathbf{\Lambda}\mathbf{M}^{-1}$$

We use this in the state variable equation to obtain

$$\begin{aligned} \dot{\mathbf{x}}(t) &= \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t) \\ &= \mathbf{M}\mathbf{\Lambda}\mathbf{M}^{-1}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t) \end{aligned}$$

We now pre-multiply this equation by \mathbf{M}^{-1} to find

$$\mathbf{M}^{-1}\dot{\mathbf{x}}(t) = \mathbf{\Lambda}\mathbf{M}^{-1}\mathbf{x}(t) + \mathbf{M}^{-1}\mathbf{B}\mathbf{u}(t)$$

We introduce the state transformation, $\mathbf{z}(t) = \mathbf{M}^{-1}\mathbf{x}(t)$, for which $\dot{\mathbf{z}}(t) = \mathbf{M}^{-1}\dot{\mathbf{x}}(t)$. Substituting this into the above state variable model gives

$$\dot{\mathbf{z}}(t) = \mathbf{\Lambda}\mathbf{z}(t) + \mathbf{M}^{-1}\mathbf{B}\mathbf{u}(t)$$

Let us assume that $n = 3$ and that we have just one control input, so that $m = 1$, and let us write out the structure of this transformed state variable system. We get

$$\begin{bmatrix} \dot{z}_1(t) \\ \dot{z}_2(t) \\ \dot{z}_3(t) \end{bmatrix} = \begin{bmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \lambda_3 \end{bmatrix} \begin{bmatrix} z_1(t) \\ z_2(t) \\ z_3(t) \end{bmatrix} + \begin{bmatrix} \beta_1 \\ \beta_2 \\ \beta_3 \end{bmatrix} u(t)$$

We see that the eigenvalues of the system matrix, \mathbf{A} , are the diagonal entries in the $\mathbf{\Lambda}$ matrix and that the complicated interactions of the original state variable equation have been resolved into a set of very simple dynamical equations. If we take just one of these equations:

$$\dot{z}_2(t) = \lambda_2 z_2(t) + \beta_2 u(t)$$

we can recall that this can be solved very easily using the integrating factor method that was outlined in Chapter 20. Using this method its solution is

$$z_2(t) = e^{\lambda_2 t} z_2(0) + \beta_2 \int_0^t e^{\lambda_2 (t-t_1)} u(t_1) dt_1$$

where $z_2(0)$ is the initial condition for state z_2 . We see that this equation has two parts: one an initial condition response and the other a forced term due to the control input, $u(t)$. We see that each system eigenvalue gives rise to an exponential time function, $e^{\lambda_i t}$, within the complete system response. We should not be too surprised at this result because we already know that each eigenvalue is linked to a system pole, and we associate pole locations with particular exponential time responses. This state variable result is another interpretation of this time domain finding. So, eigenvalues and poles are just different ways at looking at the time responses in linear system responses.

Example A compressor unit has a state variable model given as

$$\begin{aligned} \begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \end{bmatrix} &= \begin{bmatrix} -1.55 & 0.3 \\ -0.175 & -0.45 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} + \begin{bmatrix} 2.1 \\ 1.2 \end{bmatrix} [u(t)] \\ [y(t)] &= [0.9 \quad 1.1] \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} \end{aligned}$$

The input, $u(t)$, is the reference gas pressure, and the output, $y(t)$, is the actual gas pressure.

We are going to show how all the aspects of this chapter come together in one example. We first calculate the eigenvalues of the system matrix. We do this by stepping through a hand calculation first.

$$\begin{aligned} \det(\lambda \mathbf{I} - \mathbf{A}) &= \det \begin{bmatrix} \lambda + 1.55 & -0.3 \\ +0.175 & \lambda + 0.45 \end{bmatrix} \\ &= (\lambda + 1.55) \times (\lambda + 0.45) - (0.175) \times (-0.3) \\ &= \lambda^2 + 2\lambda + 0.75 \\ &= (\lambda + 0.5) \times (\lambda + 1.5) = 0 \end{aligned}$$

Hence

$$\lambda = -1.5 \text{ and } \lambda = -0.5$$

We see that the system is stable, since both eigenvalues are real and negative. From our time domain interpretation we expect to see two exponential functions $e^{-1.5t}$ and $e^{-0.5t}$ occurring in the time response.

If we calculate the transfer function, which we do using MATLAB, we arrive at

$$G(s) = \mathbf{C}(s\mathbf{I} - \mathbf{A})^{-1}\mathbf{B} = \frac{3.21s + 2.8162}{s^2 + 2s + 0.75} = \frac{n(s)}{d(s)}$$

from which $d(s) = s^2 + 2s + 0.75$ and the poles are $p_1 = -1.5$ and $p_2 = -0.5$.

This links up the state variable eigenvalues, the transfer function poles and the stability status. Finally, to resolve the time domain form, we use MATLAB to find the eigenvector matrix, \mathbf{M} , and its inverse, \mathbf{M}^{-1} . The details of the eigenvalue–eigenvector structure of \mathbf{A} are given as

$$\mathbf{A} = \begin{bmatrix} -1.55 & 0.3 \\ -0.175 & -0.45 \end{bmatrix} = \mathbf{M}\mathbf{\Lambda}\mathbf{M}^{-1} = \begin{bmatrix} -0.9864 & -0.2747 \\ -0.1644 & -0.9615 \end{bmatrix} \begin{bmatrix} -1.5 & 0 \\ 0 & -0.5 \end{bmatrix} \begin{bmatrix} -1.0645 & 0.3041 \\ 0.1820 & -1.0920 \end{bmatrix}$$

When we use a transformation of the state vector, we use $\mathbf{z}(t) = \mathbf{M}^{-1}\mathbf{x}(t)$. Applying this to the state variable model of the compressor unit gives

$$\dot{\mathbf{z}}(t) = \mathbf{M}^{-1}\dot{\mathbf{x}}(t) = \mathbf{M}^{-1}\mathbf{A}\mathbf{x}(t) + \mathbf{M}^{-1}\mathbf{B}u(t) = \mathbf{M}^{-1}\mathbf{A}(\mathbf{M}\mathbf{z}(t)) + \mathbf{M}^{-1}\mathbf{B}u(t)$$

Equivalently:

$$\dot{\mathbf{z}}(t) = \mathbf{\Lambda}\mathbf{z}(t) + \mathbf{M}^{-1}\mathbf{B}u(t)$$

We have calculated the eigenvalue matrix, $\mathbf{\Lambda}$, so we only need calculate the new input matrix, $\mathbf{M}^{-1}\mathbf{B}$. Thus the transformed state variable equation is

$$\begin{bmatrix} \dot{z}_1(t) \\ \dot{z}_2(t) \end{bmatrix} = \begin{bmatrix} -0.5 & 0 \\ 0 & -1.5 \end{bmatrix} \begin{bmatrix} z_1(t) \\ z_2(t) \end{bmatrix} + \begin{bmatrix} 1.275 \\ 3.075 \end{bmatrix} u(t)$$

We see that the original state variable equation has been resolved into two very simple dynamical equations:

$$\dot{z}_1(t) = -1.5z_1(t) - 1.8704u(t)$$

$$\dot{z}_2(t) = -0.5z_2(t) - 0.9282u(t)$$

Using the integrating factor method that was outlined in Chapter 20, we find that these give transformed state responses

$$z_1(t) = e^{-1.5t} z_1(0) - 1.8704 \int_0^t e^{-1.5(t-t_1)} u(t_1) dt_1$$

$$z_2(t) = e^{-0.5t} z_2(0) - 0.9282 \int_0^t e^{-0.5(t-t_1)} u(t_1) dt_1$$

Quite clearly the eigenvalues have appeared as exponential time functions in the final result.

22.5 Case study: Eigenvalues, eigenvectors and time responses

We conclude this chapter with the detailed working for a state variable example. This example investigates the way eigenvalues, eigenvectors and time responses are all inter-related. Although the example has only two sets of second-order dynamics, it uses MATLAB and Simulink in quite an advanced way.

Example We consider the system of masses and springs as shown in Figure 22.2.

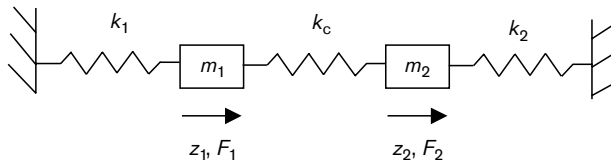


Figure 22.2 System of masses and springs.

The dynamic equations are given by

$$m_1 \ddot{z}_1(t) = -k_1 z_1(t) + k_c (z_2(t) - z_1(t)) + F_1(t)$$

$$m_2 \ddot{z}_2(t) = -k_2 z_2(t) - k_c (z_2(t) - z_1(t)) + F_2(t)$$

We let the input signals $u_1(t)$ and $u_2(t)$ be $F_1(t)$ and $F_2(t)$ respectively. The states are given by

$$\begin{aligned} x_1(t) &= z_1(t) \\ x_2(t) &= \dot{z}_1(t) \\ x_3(t) &= z(t) \\ x_4(t) &= \dot{z}_2(t) \end{aligned}$$

The state variable model can be written as

$$\begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \\ \dot{x}_3(t) \\ \dot{x}_4(t) \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ (-k_1 - k_c)/m_1 & 0 & k_c/m_1 & 0 \\ 0 & 0 & 0 & 1 \\ k_c/m_2 & 0 & (-k_2 - k_c)/m_2 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 1/m_1 & 0 \\ 0 & 0 \\ 0 & 1/m_2 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

$$\begin{bmatrix} y_1(t) \\ y_2(t) \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \\ x_4(t) \end{bmatrix}$$

If we let $k = k_1 = k_2 = k_c$ and $m = m_1 = m_2$, we find

$$\begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \\ \dot{x}_3(t) \\ \dot{x}_4(t) \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -2k/m & 0 & k/m & 0 \\ 0 & 0 & 0 & 1 \\ k/m & 0 & -2k/m & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 1/m & 0 \\ 0 & 0 \\ 0 & 1/m \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

If we further let $k = 100 \text{ N/m}$ and $m = 0.2 \text{ kg}$, the system becomes

$$\begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \\ \dot{x}_3(t) \\ \dot{x}_4(t) \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -1000 & 0 & 500 & 0 \\ 0 & 0 & 0 & 1 \\ 500 & 0 & -1000 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 5 & 0 \\ 0 & 0 \\ 0 & 5 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

Simulink simulation of mass-spring system

We can construct a simple Simulink state variable model to analyse the system under different initial and input conditions (Figure 22.3).

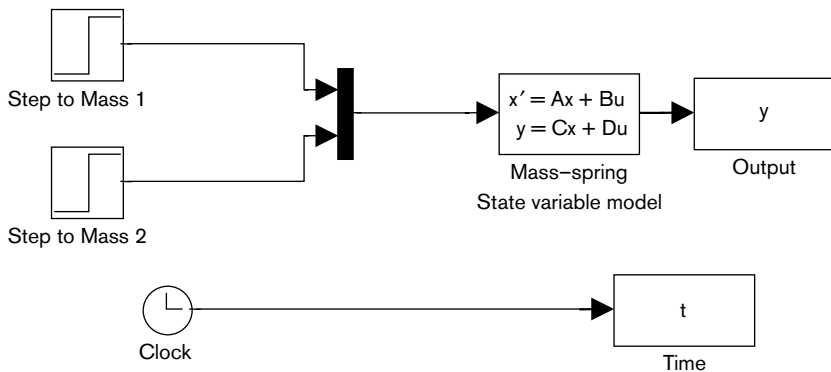


Figure 22.3 Simulink state variable model of spring-mass system.

Figure 22.4 shows a fun picture in which we have given mass 1 an initial offset of 10 cm and then let the system oscillate with no force input. Since there is no damping, the oscillations do not decay, but the energy is transferred from one mass to the other and back again.

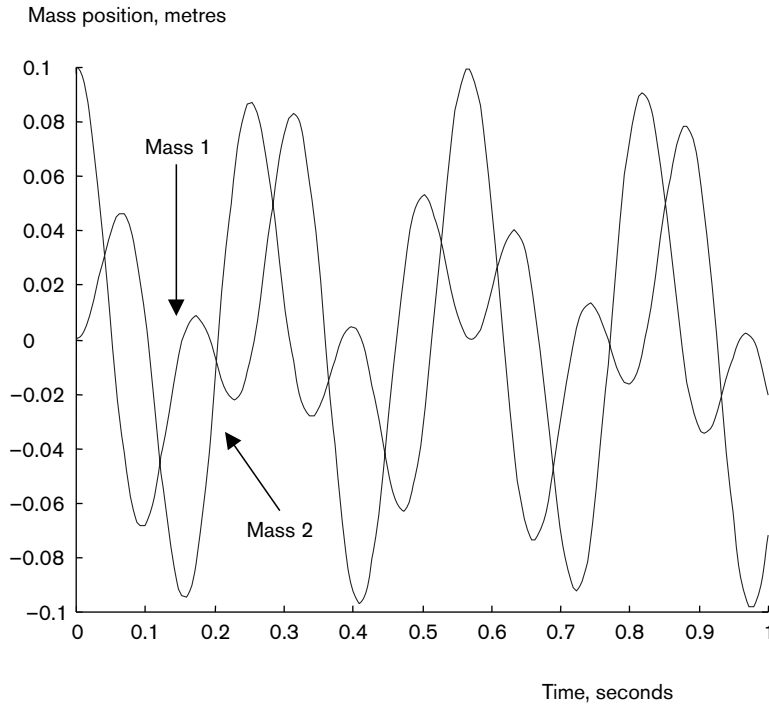


Figure 22.4 Mass position after initial offset given to mass 1.

22.5.1 System analysis for mass–spring system

We can analyse this system further by examining the eigenvalues and eigenvectors of the system matrix. We use MATLAB to do this analysis (Part A). For those who are interested, we also include a section on the derivation of the eigenvalues and eigenvectors from first principles (Part B).

Part A: analysis using MATLAB

The eigenvalues are found using MATLAB by using the `eig(A)` command. This gives $\lambda_{1,2} = \pm j38.7298$ and $\lambda_{3,4} = \pm j22.3607$. We see that we have two pairs of wholly complex eigenvalues with no damping present. We can also find the eigenvectors for this matrix. We use the MATLAB command

```
[M, lambda]=eig(A);
M =
0.0183      0.0183      0.0316      0.0316
0 + 0.7069i  0 - 0.7069i  0 + 0.7064i  0 - 0.7064i
-0.0183     -0.0183     0.0316     0.0316
0 - 0.7069i  0 + 0.7069i  0 + 0.7064i  0 - 0.7064i
```

What we have is a matrix \mathbf{M} of four eigenvectors, $\mathbf{M} = [\mathbf{m}_1, \mathbf{m}_2, \mathbf{m}_3, \mathbf{m}_4]$. We find that each eigenvector has four components relating to the four states (position and velocities of the masses). The actual *values* of the eigenvector components are not important, since they can be arbitrarily scaled; what is important is the relation between the components. We see from the first eigenvector, \mathbf{m}_1 (the eigenvector relating to the first eigenvalue, $\lambda_1 = +38.7298i$), that the components relating to the position of mass 1 and the position of mass 2 are equal and opposite (+0.0183, -0.0183). If we examine the eigenvector \mathbf{m}_3 , relating to $\lambda_3 = +22.3607i$, we find that the components relating to the position of each mass are equal and have the same sign (0.0316, 0.0316). What we can see in our simulation is that the system has two modes. One mode has the masses moving in the same direction with the same magnitude at one particular frequency. A second mode has the masses moving in opposite directions, but with the same magnitude, at a higher frequency. You can see this if you actually construct the system using toy trains and elastic bands! For this example it was fairly easy to connect the mathematics to the actual example. In more complex situations, it is more difficult to relate the information to physical variables but the basic state variable techniques will still hold.

For those who are more familiar with matrix algebra and who may wish to understand more about the derivation of the eigenvalues and eigenvectors, we give the analysis in the following. However, it can be readily missed and we can proceed to the Simulink simulation of the mass-spring example with forces active.

Part B: derivation of eigenvalues and eigenvectors for mass-spring example

The eigenvalues are given by the solutions of $\det(\lambda\mathbf{I} - \mathbf{A}) = 0$:

$$\det \begin{bmatrix} \lambda & -1 & 0 & 0 \\ 2k/m & \lambda & -k/m & 0 \\ 0 & 0 & \lambda & -1 \\ -k/m & 0 & 2k/m & \lambda \end{bmatrix} = 0$$

If we are familiar with calculating determinants of matrices of size greater than 2×2 , we can find:

Expanding about the first row:

$$\begin{aligned} \det(\lambda\mathbf{I} - \mathbf{A}) &= +\lambda \times \det \begin{bmatrix} -\lambda & -1 & 0 & 0 \\ 2k/m & \lambda & -k/m & 0 \\ 0 & 0 & \lambda & -1 \\ -k/m & 0 & 2k/m & \lambda \end{bmatrix} - (-1) \times \det \begin{bmatrix} -\lambda & 1 & 0 & 0 \\ 2k/m & \lambda & -k/m & 0 \\ 0 & 0 & \lambda & -1 \\ -k/m & 0 & 2k/m & \lambda \end{bmatrix} \\ &= \lambda \left[\lambda \left(\lambda^2 - (-1) \times \frac{2k}{m} \right) \right] + \left[2 \frac{k}{m} \left(\lambda^2 - (-1) \times \frac{2k}{m} \right) + \left(\frac{-k}{m} \right) \left(\frac{-k}{m} \times (-1) - 0 \times \lambda \right) \right] \\ &= \left[\lambda^2 \left(\lambda^2 + 2 \frac{k}{m} \right) \right] + \left[\frac{2k}{m} \left(\lambda^2 + \frac{2k}{m} \right) - \left(\frac{k^2}{m^2} \right) \right] \\ &= \lambda^4 + 2 \frac{k}{m} \lambda^2 + 2 \frac{k}{m} \lambda^2 + \frac{4k^2}{m^2} - \frac{k^2}{m^2} = \lambda^4 + 4 \frac{k}{m} \lambda^2 + \frac{3k^2}{m} \\ &= \left(\lambda^2 + 3 \frac{k}{m} \right) \left(\lambda^2 + \frac{k}{m} \right) = 0 \end{aligned}$$

This gives

$$\lambda_{1,2} = \pm j\sqrt{\frac{k}{m}} \quad \text{and} \quad \lambda_{3,4} = \pm j\sqrt{\frac{3k}{m}}$$

In a typical end order mass–damping–stiffness system, the period of oscillation is given by $\sqrt{k/m}$. We will find through simulation that the mass–spring example has two natural modes: one with a frequency of $\sqrt{k/m}$ rad/s and another with a frequency of $\sqrt{3k/m}$ rad/s. Given the values in our example, these values work out to be $\sqrt{500} = 22.3607$ rad/s and $\sqrt{1500} = 38.7298$ rad/s.

Calculation of eigenvectors

We have one eigenvector $\mathbf{m} = [v_1 \ v_2 \ v_3 \ v_4]^T$ associated with each eigenvalue. We find them by replacing the value of λ in

$$\begin{bmatrix} \lambda & -1 & 0 & 0 \\ 2k/m & \lambda & -k/m & 0 \\ 0 & 0 & \lambda & -1 \\ -k/m & 0 & 2k/m & \lambda \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \end{bmatrix} = \begin{bmatrix} \lambda & -1 & 0 & 0 \\ 1000 & \lambda & -500 & 0 \\ 0 & 0 & \lambda & -1 \\ -500 & 0 & 1000 & \lambda \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \end{bmatrix} = 0$$

and solving for v_1 to v_4 . We find that when we substitute for a value of λ , the equations will not be independent; that is, we are not able to solve explicitly for each component v_1 , v_2 , v_3 and v_4 . Rather, we can find the relationship between them.

For $\lambda_1 = \sqrt{500}j$, we have from the first row

$$\sqrt{500}jv_1 - v_2 = 0 \quad \text{or} \quad v_2 = \sqrt{500}jv_1$$

The second equation gives

$$1000v_1 + \sqrt{500}jv_2 - 500v_3 = 0$$

Substituting for v_2 gives

$$1000v_1 - 500v_1 - 500v_3 = 0 \quad \text{or} \quad v_3 = v_1$$

From the third equation we find that

$$\sqrt{500}jv_3 - v_4 = 0 \quad \text{or} \quad v_4 = \sqrt{500}jv_3 = \sqrt{500}jv_1$$

Since we cannot solve for a numerical value we set an arbitrary value on v_1 and this gives the values for the remaining components. In this case, let us set $v_1 = 1$.

$$\text{The eigenvector } \mathbf{m}_1 = \begin{bmatrix} 1 \\ \sqrt{500}j \\ 1 \\ \sqrt{500}j \end{bmatrix}$$

If we do likewise for $\lambda_3 = \sqrt{1500}j$, we have

$$\sqrt{1500}jv_1 - v_2 = 0 \quad \text{or} \quad v_2 = \sqrt{1500}jv_1$$

The second equation gives

$$1000v_1 + \sqrt{1500}jv_2 - 500v_3 = 0$$

Substituting for v_2 gives

$$1000v_1 - 1500v_1 - 500v_3 = 0 \quad \text{or} \quad v_3 = -v_1$$

From the third equation we find that

$$\sqrt{1500} j v_3 - v_4 = 0 \quad \text{or} \quad v_4 = \sqrt{1500} j v_3 = -\sqrt{1500} j v_1$$

Since we cannot solve for a numerical value we set an arbitrary value on v_1 and this gives the values for the remaining components. In this case, let us set $v_1 = 1$.

$$\text{The eigenvector } \mathbf{m}_3 = \begin{bmatrix} 1 \\ \sqrt{1500} j \\ -1 \\ -\sqrt{1500} j \end{bmatrix}$$

What does this tell us about the system? The eigenvalues of the system are related to different *modes* in the system. In this example, the eigenvalues are related to the two natural frequencies within the system. If the system was oscillating at the lower frequency the two masses would be moving in the same direction and with the same magnitude. This is indicated by the values of v_1 and v_3 in \mathbf{m}_1 having the same magnitude and same sign (v_1 and v_3 are the components relating to the position of the masses). For the higher frequency of oscillation, we have the masses moving in opposite directions but with the same magnitude, demonstrated by the values of $v_1 = 1$ and $v_3 = -1$ in \mathbf{m}_3 .

Scaling of eigenvectors: relation to MATLAB eigenvector solution in Part A

The eigenvector \mathbf{m}_3 corresponded to the eigenvalue $\lambda_3 = \sqrt{1500} j$ and had the value

$$\mathbf{m}_3 = [1 \quad j\sqrt{1500} \quad -1 \quad -j\sqrt{1500}]^T$$

The eigenvector produced from the MATLAB solution which corresponded to the same eigenvalue was:

$$\mathbf{m}_1 = [0.0183 \quad +j0.7069 \quad 0.0183 \quad j0.7069]^T$$

We see that they are not equal in magnitude. However, as we saw from Part B when we calculated the eigenvalues, it was the ratio between each eigenvector element that was important; therefore we can scale the eigenvectors by any constant number. If we scaled our eigenvector \mathbf{m}_3 by 0.0183, we would find

$$\mathbf{m}_3 = [0.0183 \quad +j0.7088 \quad -0.0183 \quad -j0.7088]^T$$

which corresponds to the eigenvector given by MATLAB in Part A (given the number of significant figures used throughout all the calculations).

22.5.2 Simulation of mass–springs with force inputs

In particular we can examine what happens if:

- (a) we inject the same force input to both masses
- (b) we inject the same magnitude of force but in different directions

If we inject a force of 10 N to each mass, we find that the masses oscillate with exactly the same magnitude and direction (the plot lines overlay each other). Figure 22.5(a) shows the position of the masses oscillating ± 20 cm. If we apply a force of +10 N to mass 1 and -10 N

to mass 2, we see, in Figure 22.5(b), the masses oscillating at higher frequency and in opposite directions. We can use MATLAB to evaluate the frequency of oscillation of the masses.

For graph (a):

$$\text{Period} = 0.28 \text{ seconds}$$

$$\text{Frequency (Hz)} = 1/0.28 = 3.57 \text{ Hz}$$

$$\text{Frequency (rad/s)} = 3.57 \times 2 \times 3.14 = 22.42 \text{ rad/s}$$

For graph (b):

$$\text{Period} = 0.16 \text{ seconds}$$

$$\text{Frequency (Hz)} = 1/0.16 = 6.25 \text{ Hz}$$

$$\text{Frequency (rad/s)} = 6.25 \times 2 \times 3.14 = 39.25 \text{ rad/s}$$

If we compare these with the values of the eigenvalues of the system, we find that

$$\lambda_1 = 22.36 \text{ rad/s} \quad \text{and} \quad \lambda_2 = 38.73 \text{ rad/s}$$

which are close to our values given the resolution of the Simulink simulation.

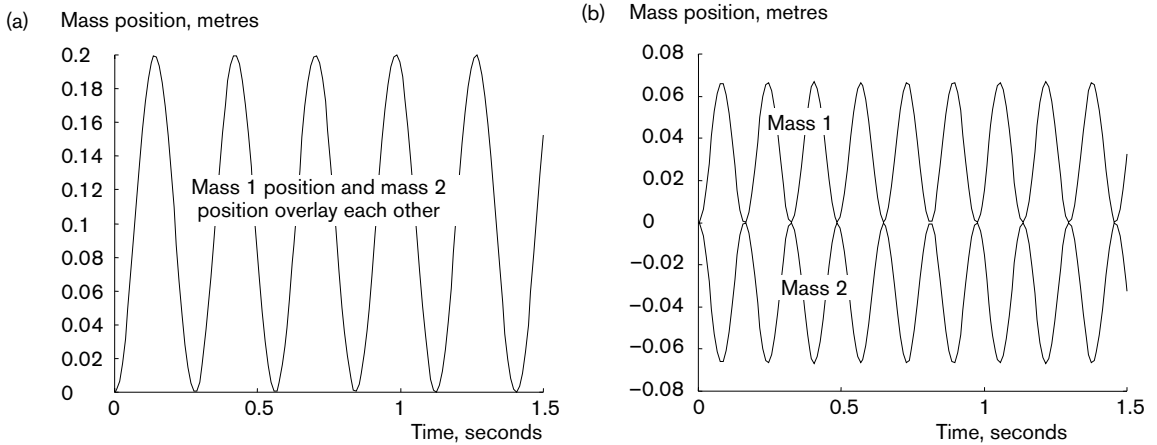


Figure 22.5 Mass positions under (a) the same and (b) opposite applied forces.

What we have learnt

- ✓ To remember the extra rigour needed with matrix equations.
- ✓ To find the eigenvalues and eigenvectors of a system matrix.
- ✓ To realise the connection between a diagonal matrix and its eigenvalues.
- ✓ To understand that the system poles and the system eigenvalues have the same interpretation.
- ✓ To relate the system eigenvalues to a system's stability.
- ✓ To analyse state variable system time responses in more detail.

Multiple choice

M22.1 The determinant of

$$\mathbf{A} = \begin{bmatrix} 2 & 3 \\ 1 & -2 \end{bmatrix}$$

is

- (a) -1
- (b) -7
- (c) -4
- (d) -8

M22.2 Given the vector–matrix equation $\mathbf{AX} = \mathbf{B}$, how do we calculate the vector \mathbf{X} ?

- (a) $\mathbf{X} = \mathbf{B}/\mathbf{A}$
- (b) $\mathbf{X} = \mathbf{BA}^{-1}$
- (c) $\mathbf{X} = \mathbf{A}^{-1}\mathbf{B}$
- (d) $\mathbf{X} = \mathbf{BA}$

M22.3 If a system has matrix dimension $\mathbf{A}_{3 \times 3}$, $\mathbf{B}_{3 \times 2}$ and $\mathbf{C}_{1 \times 3}$, how many eigenvalues will there be?

- (a) 1
- (b) 2
- (c) 3
- (d) 9

M22.4 The inverse of \mathbf{A} , \mathbf{A}^{-1} , is given by:

- (a) $\mathbf{BA}^{-1} = \mathbf{I}$
- (b) $\mathbf{M}^{-1}\mathbf{A}\mathbf{M} = \mathbf{I}$
- (c) $\mathbf{A}^{-1}\mathbf{B} = \mathbf{I}$
- (d) $\mathbf{A}^{-1}\mathbf{A} = \mathbf{I}$

M22.5 The system stability is determined from:

- (a) the \mathbf{A} matrix
- (b) the \mathbf{A} and \mathbf{B} matrices
- (c) the \mathbf{A} and \mathbf{C} matrices
- (d) the \mathbf{A} , \mathbf{B} and \mathbf{C} matrices

M22.6 The eigenvalues of

$$\mathbf{A} = \begin{bmatrix} 2 & 1 \\ 4 & 5 \end{bmatrix}$$

can be found by solving

- (a) $\lambda^2 - 7\lambda + 6 = 0$
- (b) $\lambda^2 - 7\lambda + 10 = 0$
- (c) $\lambda^2 - 5\lambda + 4 = 0$
- (d) $\lambda^2 - 2\lambda + 4 = 0$

M22.7 If the eigenvalues of the system matrix are $\lambda_1 = -2$, $\lambda_2 = -0.3$ and $\lambda_3 = 5$:

- (a) the system is unstable
- (b) the system is stable
- (c) the system is sometimes stable
- (d) the system is partially stable

M22.8 If the system eigenvalues are $\lambda_1 = 2$, $\lambda_{2,3} = -3 \pm 2j$, then the poles of the associated transfer function are:

- (a) $p_1 = -2$, $p_{2,3} = 3 \pm 2j$
- (b) $p_1 = 2$, $p_{2,3} = 3 \pm 2j$
- (c) $p_1 = 2$, $p_{2,3} = -3 \pm 2j$
- (d) $p_1 = -2$, $p_{2,3} = -3 \pm 2j$

M22.9 Given the system equation $\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u}$, the eigenvalues can be found by solving:

- (a) $(\lambda\mathbf{I} - \mathbf{A})^{-1} = 0$
- (b) $(\lambda\mathbf{I} - \mathbf{A})^{-1}\mathbf{B} = 0$
- (c) $|\lambda\mathbf{I} - \mathbf{A}| = 0$
- (d) $(\lambda\mathbf{I} - \mathbf{A})\mathbf{B} = 0$

M22.10 If $\mathbf{A} = \mathbf{M}^{-1}\mathbf{A}\mathbf{M}$ where \mathbf{A} is the diagonal matrix of system eigenvalues, then:

- (a) \mathbf{M} is the matrix of eigenvectors
- (b) \mathbf{M}^{-1} is the matrix of eigenvectors
- (c) \mathbf{M} is the matrix given by $(\lambda\mathbf{I} - \mathbf{A})^{-1}$
- (d) \mathbf{M} is the matrix given by $\mathbf{A}^{-1}\mathbf{B}$

Questions: practical skills

Q22.1 Use the determinant formula to find the system eigenvalues and determine the stability of the following systems:

$$\begin{aligned} \text{(a)} \quad \dot{\mathbf{x}}(t) &= \begin{bmatrix} 2.94 & -8.01 \\ 3.33 & -6.95 \end{bmatrix} \mathbf{x}(t) + \begin{bmatrix} 1.5 \\ 1.3 \end{bmatrix} \mathbf{u}(t) \\ \mathbf{y}(t) &= [2.8 \quad 5.2] \mathbf{x}(t) \end{aligned}$$

$$(b) \quad \dot{\mathbf{x}}(t) = \begin{bmatrix} -4.07 & 3.53 \\ -3.53 & 4.07 \end{bmatrix} \mathbf{x}(t) + \begin{bmatrix} 5.1 \\ 9.4 \end{bmatrix} \mathbf{u}(t)$$

$$\mathbf{y}(t) = [1.0 \quad 5.7] \mathbf{x}(t)$$

$$(c) \quad \dot{\mathbf{x}}(t) = \begin{bmatrix} 5.29 & -5.33 \\ 2.22 & -1.29 \end{bmatrix} \mathbf{x}(t) + \begin{bmatrix} 0.36 \\ 0.58 \end{bmatrix} \mathbf{u}(t)$$

$$\mathbf{y}(t) = [0.84 \quad 0.91] \mathbf{x}(t)$$

Q22.2 Use MATLAB to find the system eigenvalues and determine the stability of the following systems:

$$(a) \quad \dot{\mathbf{x}}(t) = \begin{bmatrix} -1.4 & 2.6 & 0 \\ -2.6 & -1.4 & 0 \\ 0 & 0 & -2.5 \end{bmatrix} \mathbf{x}(t) + \begin{bmatrix} 6.4 \\ 2.5 \\ 1.8 \end{bmatrix} \mathbf{u}(t)$$

$$\mathbf{y}(t) = [0.36 \quad -0.92 \quad -0.23] \mathbf{x}(t)$$

$$(b) \quad \dot{\mathbf{x}}(t) = \begin{bmatrix} -19.02 & 8.70 & 3.11 \\ -24.95 & 10.67 & 5.14 \\ -32.73 & 17.22 & 3.04 \end{bmatrix} \mathbf{x}(t) + \begin{bmatrix} 0.55 \\ -0.32 \\ 0.58 \end{bmatrix} \mathbf{u}(t)$$

$$\mathbf{y}(t) = [0.36 \quad -0.96 \quad -0.45] \mathbf{x}(t)$$

Q22.3 A system's state space model is given by

$$\dot{\mathbf{x}}(t) = \begin{bmatrix} 0 & 1.0 \\ -0.75 & -2.0 \end{bmatrix} \mathbf{x}(t) + \begin{bmatrix} 1.5 \\ 0.5 \end{bmatrix} \mathbf{u}(t)$$

$$\mathbf{y}(t) = [0.2 \quad 0.8] \mathbf{x}(t)$$

- Find the system eigenvalues using the determinant formula, $\det(s\mathbf{I} - \mathbf{A}) = 0$.
- Find the system eigenvalues using the MATLAB command `eig(A)`.
- Find the system transfer function using $G(s) = C(s\mathbf{I} - \mathbf{A})^{-1}B$.
- Use the system transfer function, $G(s)$, to identify the system poles.
- Make as many connections as possible between the various findings.

Q22.4 A system's state space model is given by

$$\dot{\mathbf{x}}(t) = \begin{bmatrix} -27.61 & 12.77 & 5.58 \\ -20.67 & 7.45 & 4.61 \\ -68.89 & 36.52 & 12.67 \end{bmatrix} \mathbf{x}(t) + \begin{bmatrix} 0.55 \\ -0.32 \\ 0.58 \end{bmatrix} \mathbf{u}(t)$$

$$\mathbf{y}(t) = [0.36 \quad -0.96 \quad -0.45] \mathbf{x}(t)$$

- Find the system eigenvalues using the MATLAB command `eig(A)`.
- Find the system transfer function using the MATLAB command `ss2tf(a,b,c,d,iu)`.
- Use the system transfer function, $G(s)$, to identify the system poles.
- Make as many connections as possible between the various findings.

Q22.5 Use the integrating factor method to investigate the time dynamics of a state variable model given by

$$\dot{x}(t) = [-0.75]x(t) + [2.5]u(t)$$

- If $x(0) = 150$, find the state free response equation.
- If the input is set to $u(t) = 1.5, 0 \leq t$, find the forced state response.
- Use an eigenvalue analysis to determine the stability status of the system.
- How is the system stability reflected in the system responses?

Problems

P22.1 In a control system design exercise for a marine engine, an engineer is using the state space framework. The system model is given as

$$\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}u(t)$$

$$\mathbf{y}(t) = \mathbf{C}\mathbf{x}(t)$$

The data for the model are:

$$\mathbf{A} = \begin{bmatrix} 13/6 & 4/3 \\ -4/3 & -7/6 \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} 0 \\ 0.5 \end{bmatrix}, \quad \text{and} \quad \mathbf{C} = [1 \quad 1]$$

where the system eigenvector matrices are given by

$$\mathbf{M} = \begin{bmatrix} -1/3 & 2/3 \\ 2/3 & -1/3 \end{bmatrix} \quad \text{and} \quad \mathbf{M}^{-1} = \begin{bmatrix} 1 & 2 \\ 2 & 1 \end{bmatrix}$$

- (a) (i) Determine the system eigenvalues
(ii) Find the decomposed form for the system as

$$\dot{\mathbf{z}}(t) = \mathbf{\Lambda}\mathbf{z}(t) + \mathbf{M}^{-1}\mathbf{B}u(t)$$

$$\mathbf{y}(t) = \mathbf{C}\mathbf{M}\mathbf{z}(t)$$

- (iii) Examine the structure of the system using the form

$$\begin{bmatrix} \dot{z}_1(t) \\ \dot{z}_2(t) \end{bmatrix} = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} \begin{bmatrix} z_1(t) \\ z_2(t) \end{bmatrix} + \begin{bmatrix} \beta_1 \\ \beta_2 \end{bmatrix} u(t)$$

$$y(t) = [\chi_1 \quad \chi_2] \begin{bmatrix} z_1(t) \\ z_2(t) \end{bmatrix}$$

- (iv) Compute the system transfer function,

$$G(s) = \mathbf{C}(s\mathbf{I}_2 - \mathbf{A})^{-1}\mathbf{B}$$

- (v) Does the transfer function have the same pole locations as the eigenvalues of the system matrix?

- (b) The engineer decides to experiment with a different method of actuation for the system and this leads to a different input matrix:

$$\mathbf{B}_1 = \begin{bmatrix} -1/6 \\ 1/3 \end{bmatrix}$$

- (i) Find the decomposed form for the system which uses \mathbf{B}_1 as the input matrix.
(ii) Examine the structure of the system using the form in part (a)(iii). What differences can be found between this new decomposition and the one found in part (a)?
(iii) Compute the system transfer function for the new actuation system:

$$G(s) = \mathbf{C}(s\mathbf{I}_2 - \mathbf{A})^{-1}\mathbf{B}_1$$

- (iv) What differences can be found between this new transfer function and the one found in part (a)?

- (v) Would this difference have any implication for the design of closed-loop control?

- (c) The engineer then decides to experiment with a different method of measurement for the system and this leads to a different output matrix: $\mathbf{C}_0 = [1.14 \quad 2.28]$.

- (i) Find the decomposed form for the system.
- (ii) Examine the structure of the system using the form in part (a)(iii). What differences can be found between this new decomposition and the one found in part(a)?
- (iii) Compute the system transfer function for the new measurement system in place:

$$G_0(s) = \mathbf{C}_0(s\mathbf{I}_2 - \mathbf{A})^{-1}\mathbf{B}$$

- (iv) What differences can be found between this new transfer function and the one found in part (a)?
- (v) Would this difference have any implication for the design of closed-loop control?

P22.2 The temperature of an ingot in a reheat furnace in a steel mill obeys a state space dynamical equation:

$$\dot{T}(t) = [-0.5]T(t) + [600]f_{\text{oil}}(t)$$

where the ingot temperature $T(t)$ is measured in °C, and the fuel flow variable, $f_{\text{oil}}(t)$, takes values in the range $0 < f_{\text{oil}}(t) < 1$, where unity flow corresponds to the fuel valve fully open. In this process, time is measured in hours.

Use Simulink to determine the following system temperature responses.

- (a) The ingot is charged cold at an ambient temperature of 0 °C, and the furnace operated for 5 hours. What is the steady state temperature of the ingot when it leaves the reheat furnace?
- (b) The ingot is allowed to stand for 20 minutes after discharge from the furnace. By how much does the temperature fall?
- (c) Which of the above represents a free state response, and which represents a forced response?

P22.3 Consider the state space equation

$$\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t)$$

$$\mathbf{y}(t) = \mathbf{C}\mathbf{x}(t)$$

with an initial condition, \mathbf{x}_0 .

The eigenvalue–eigenvector decomposition for \mathbf{A} is given by $\mathbf{A}\mathbf{M} = \mathbf{M}\mathbf{\Lambda}$, where the matrix $\mathbf{\Lambda}$ is a diagonal matrix of the eigenvalues, and matrix \mathbf{M} contains the eigenvectors. This further gives $\mathbf{A} = \mathbf{M}\mathbf{\Lambda}\mathbf{M}^{-1}$. If the state is replaced everywhere by $\mathbf{x}(t) = \mathbf{M}\mathbf{z}(t)$, then the following transformation occurs:

$$\begin{aligned} \dot{\mathbf{x}}(t) &= \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t) & \Rightarrow & \quad \dot{\mathbf{z}}(t) = \mathbf{\Lambda}\mathbf{z}(t) + \mathbf{M}^{-1}\mathbf{B}\mathbf{u}(t) \\ \mathbf{y}(t) &= \mathbf{C}\mathbf{x}(t) & & \quad \mathbf{y}(t) = \mathbf{C}\mathbf{M}\mathbf{z}(t) \end{aligned}$$

- (a) If $n = 3$ and

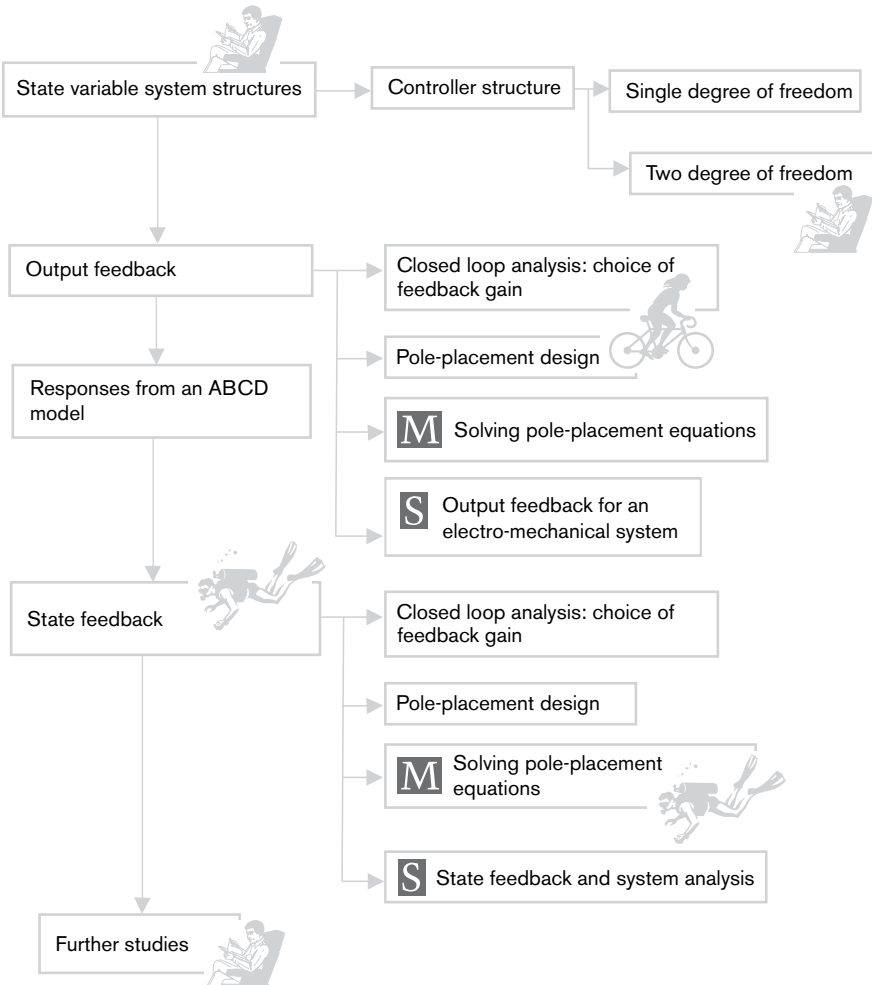
$$\mathbf{\Lambda} = \begin{bmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \lambda_3 \end{bmatrix}$$

draw out the structure of the transformed system.

- (b) Consider the implications for the system if a row of the transformed input matrix $[\mathbf{M}^{-1}\mathbf{B}]$ is completely zero. What would this imply for the control of the associated eigenvalue and its time response?
- (c) Consider the implication for the system if a column of the transformed output matrix $[\mathbf{C}\mathbf{M}]$ is completely zero. Would the associated eigenvalue and its time response be seen in the output vector?

23

An introduction to control using state variable system models



Gaining confidence



Help?



Going deeper



Skill section



Time to read

State space system models are popular because of the precision they bring to the task of modelling systems or processes we wish to control. Simulink and other simulation software tools usually have a state variable system icon or routine to help us exploit the structure and notation when we wish to load the state variable matrices **A**, **B**, **C**, **D** with system data. The very fact that we describe a system using a state variable equation model leads directly to new terms and analysis methods when we discuss system stability and system responses.

We saw in the last chapter that we used system eigenvalues when we discussed system stability. We also saw that these eigenvalues took the same values as the system poles. When we looked at the initial condition and forced system responses we found that these naturally occurred in the state variable framework. One powerful result we found was that the state variable method easily accommodated multiple-input and multiple-output systems.

With state variable systems we are using an internal description of the system dynamics as opposed to a straightforward input–output transfer system model. We will find that we gain from the knowledge of these internal states and benefit from the use of this information in control system design.

When we discuss the control of a plant or process using a state system model we will find many new fundamental insights about control theory. As always we must relate these insights back to the practical problem of why controlling real systems is such a difficult task. We will also learn how state variable systems lead to new methods for the design of compensators. These developments are relatively new; whereas PID control dates from about 1940 onwards, state variable methods emerged from around 1960 and underwent many refinements before reaching the highly developed level available today. In this last chapter of the book we introduce state variable control methods and point the way forward to new methods that will be found in more advanced courses and books on control engineering.

Learning objectives

- To understand output feedback and its practical limitations.
- To understand the basics of state feedback and its potential advantages.
- To appreciate the way forward in control system engineering studies.

23.1 State variable system structure

We begin our introduction to state variable control methods by looking once more at the general state variable diagram as shown in Figure 23.1.

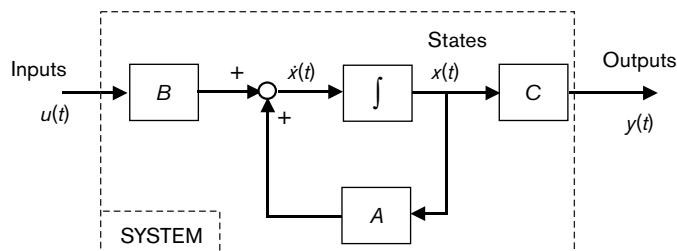


Figure 23.1 General state variable system diagram.

With the state variable method it becomes increasingly important to appreciate and use the link between the real world and the rather abstract representation of the state variable model. Thus we must relate this general schematic to the dynamical behaviour of any particular practical or industrial system.

(a) *The system state*

We usually denote the system state by the vector $\mathbf{x}(t)$. It is the important set or list of system variables. It is the system state and the system matrix, \mathbf{A} , that govern how the system internal dynamics will operate. This means that the state is the key *information-carrying* vector in a state variable system model.

(b) *The system inputs*

We list the system inputs through the input vector, $\mathbf{u}(t)$. These inputs are the control and actuator inputs to the system. They are able to influence and shape the dynamical behaviour of the system. The inputs are connected to the internal states of the system through the input matrix \mathbf{B} . We sometimes find this matrix being called the driving matrix or control matrix to emphasise its purpose as a link between controls and state. In real systems, it is usual to find that we do not have many control inputs available to us to manipulate and control the states and output values.

(c) *The system outputs*

The system outputs are generally those variables we can measure and wish to control. We list these as the vector $\mathbf{y}(t)$. The state is connected to the system outputs via the output matrix, \mathbf{C} . As with the control inputs, we usually find that there are many fewer outputs than states in real systems. Thus the matrix \mathbf{C} is *information-reducing* in the sense that \mathbf{C} specifies just how much we are able to view of the whole system state vector, $\mathbf{x}(t)$.

This fundamental appreciation of the system structure is a key feature of the way state variable methods work. We have seen in the previous chapters that state variable system models are very general and that we can deal with general multi-input, multi-output (MIMO) systems very easily. However, throughout most of this book we have dealt with single-input single-output (SISO) systems, where we have a single input available to control a single output. For example, in an electro-mechanical system we might use motor voltage to control the output shaft position. In this chapter we assume that our system has the same number of control input variables as the number of output variables that we wish to control. We call these *square* systems, because if we worked out the input to output transfer function we would find that it is a square matrix. Examples of square systems are quite common in industry. For example, in a distillation column we might use input feedstock flow and recycle flow to control the column top and bottom feedstock temperatures. This is a two-input, two-output (2I2O) system and the system is square.

23.2 State variable controller structure

We start by looking at an output feedback structure that we have generally used in this book. We saw that in Chapter 18 on the practical aspects of PID that we often split the pathways of the PID controller to achieve particular practical effects. We follow and extend this idea for this last chapter in the book. Past chapters have used what is called a

single degree of freedom controller structure. For example we commonly used proportional control as

$$\mathbf{u}(t) = \mathbf{K}_P(\mathbf{r}(t) - \mathbf{y}(t))$$

The 'single degree of freedom' refers to the choice of only *one* controller, K_P , in the design problem (Figure 23.2(a)).

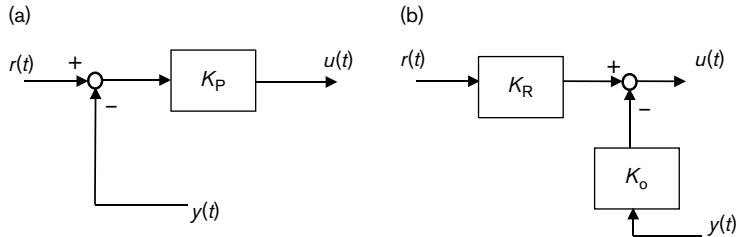


Figure 23.2 Feedback controller structures: (a) single degree of freedom structure; (b) two degree of freedom structure.

However, we can use a similar idea, except we use two pathways for the controller. This structure has two separate controllers, \mathbf{K}_R and \mathbf{K}_O , and the control law would be written as

$$\mathbf{u}(t) = \mathbf{K}_R\mathbf{r}(t) - \mathbf{K}_O\mathbf{y}(t)$$

This controller has different gains in the reference path and the feedback path (Figure 23.2(b)). We use the reference path gain, \mathbf{K}_R , to ensure that the steady state output values are correctly met for constant reference signals, \mathbf{r} . The feedback path gain \mathbf{K}_O we use to meet the dynamic design specification of the system. We give this structure a proper technical name and call it a *two degree of freedom controller*.

The structures in Figure 23.2 are examples of *output* feedback controllers, and this is where we start our investigation into the control of systems with state variable models.

23.3 A state variable investigation of output feedback

Output feedback and the use of an output feedback controller has been the common control structure applied in this book so far. These controllers were used in the context of an input–output transfer function model. When we move to a state variable model for the system we have time domain information about the internal structure of the system available in the form of a state vector. To investigate the link between the state information and output feedback, we set up a simple two degrees of freedom output feedback control system, as shown in Figure 23.3.

The output feedback law is

$$\mathbf{u}(t) = \mathbf{K}_R\mathbf{r}(t) - \mathbf{K}_O\mathbf{y}(t)$$

It is useful to realise that this is a matrix gain control law because the control, $\mathbf{u}(t)$, and the output, $\mathbf{y}(t)$, can both be vectors. For example, if we have m controls, m outputs and m references, then the gains \mathbf{K}_R and \mathbf{K}_O are square gain matrices.

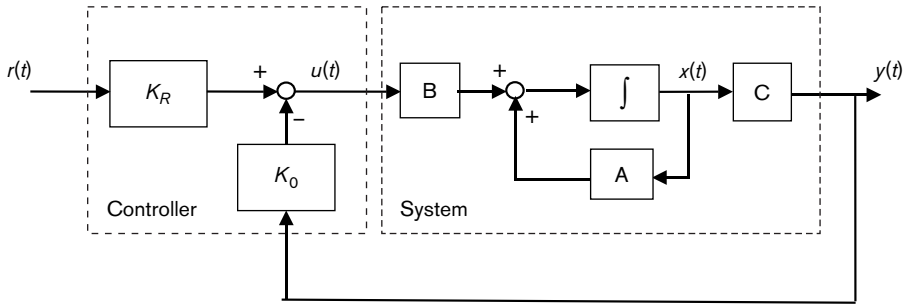


Figure 23.3 Output feedback control for state variable system model.

Example single-input, single-output system case

In the SISO case, we have one control and one output, so that $m = 1$. The gains in the control law are scalars or 1×1 matrices, giving

$$u(t) = k_R r(t) - k_O y(t)$$

Example multi-input, multi-output system case

In the MIMO case, we have more than one control and more than one output. Let us suppose that $m = 2$. The gains in the control law are now 2×2 matrices and we can write the control law in matrix form as

$$\mathbf{u}(t) = \mathbf{K}_R \mathbf{r}(t) - \mathbf{K}_O \mathbf{y}(t)$$

or we can use matrix element form as

$$\begin{bmatrix} u_1(t) \\ u_2(t) \end{bmatrix} = \begin{bmatrix} k_{R(1,1)} & k_{R(1,2)} \\ k_{R(2,1)} & k_{R(2,2)} \end{bmatrix} \begin{bmatrix} r_1(t) \\ r_2(t) \end{bmatrix} - \begin{bmatrix} k_{O(1,1)} & k_{O(1,2)} \\ k_{O(2,1)} & k_{O(2,2)} \end{bmatrix} \begin{bmatrix} y_1(t) \\ y_2(t) \end{bmatrix}$$

It is not difficult to see that the matrix element form is unnecessarily tedious to write and manipulate.

23.3.1 Closed-loop system analysis for output feedback system

We use the system structure shown in Figure 23.3 and directly write down the equations for the closed loop in matrix form as follows:

1. the system equations:

$$\begin{aligned} \dot{\mathbf{x}} &= \mathbf{Ax} + \mathbf{Bu} \\ \mathbf{y} &= \mathbf{Cx} \end{aligned}$$

2. the output feedback law:

$$\mathbf{u}(t) = \mathbf{K}_R \mathbf{r}(t) - \mathbf{K}_O \mathbf{y}(t)$$

Although we do not explicitly use the fact at this stage, we assume that the gain matrices \mathbf{K}_R and \mathbf{K}_O are constant and square. The gain matrix \mathbf{K}_R will be used to ensure that the correct state output values are reached for constant step reference signals, $\mathbf{r}(t) = \mathbf{r}_0$. If we now progress this analysis for the closed-loop system, we substitute for the control law, $\mathbf{u}(t) = \mathbf{K}_R \mathbf{r}(t) - \mathbf{K}_O \mathbf{y}(t)$, in the system equation as follows:

$$\begin{aligned}\dot{\mathbf{x}}(t) &= \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t) \\ &= \mathbf{A}\mathbf{x}(t) + \mathbf{B}(\mathbf{K}_R\mathbf{r}(t) - \mathbf{K}_O(\mathbf{y}(t)))\end{aligned}$$

We follow this by using the output equation $\mathbf{y}(t) = \mathbf{C}\mathbf{x}(t)$, giving

$$\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}(\mathbf{K}_R\mathbf{r}(t) - \mathbf{K}_O(\mathbf{C}\mathbf{x}(t)))$$

We tidy up to find

$$\begin{aligned}\dot{\mathbf{x}}(t) &= (\mathbf{A} - \mathbf{B}\mathbf{K}_O\mathbf{C})\mathbf{x}(t) + \mathbf{B}\mathbf{K}_R\mathbf{r}(t) \\ &= \mathbf{A}_{CL}\mathbf{x}(t) + \mathbf{B}\mathbf{K}_R\mathbf{r}(t)\end{aligned}$$

This equation is the state variable model for the closed-loop system.

Key result: Output feedback control

We define the closed-loop system matrix as

$$\mathbf{A}_{CL} = \mathbf{A} - \mathbf{B}\mathbf{K}_O\mathbf{C}$$

and the closed-loop system poles will depend on the eigenvalues of the closed-loop system matrix, \mathbf{A}_{CL} .

We now have two design problems: the choice of matrix \mathbf{K}_O and the choice of matrix \mathbf{K}_R . We can easily see that the closed-loop matrix depends on our choice of output feedback gain matrix, \mathbf{K}_O . If the matrix gain \mathbf{K}_O is varied, then the eigenvalues of the closed-loop system matrix $\mathbf{A}_{CL} = (\mathbf{A} - \mathbf{B}\mathbf{K}_O\mathbf{C})$ will also vary and move to new locations. We would like these new locations to maintain the stability of the closed loop and also achieve some desired performance measures like specified damping or natural frequency. This is the first problem we have: how do we choose \mathbf{K}_O , and what can we achieve in terms of performance with our choice?

The second part of our output feedback study concerns the choice of the reference gain matrix, \mathbf{K}_R . This is not so difficult if we first assume that we can find an output feedback gain \mathbf{K}_O that stabilises the closed-loop so that $\mathbf{A}_{CL} = [\mathbf{A} - \mathbf{B}\mathbf{K}_O\mathbf{C}]$ has all its eigenvalues with negative real parts. We can then use the closed-loop state variable equations

$$\begin{aligned}\dot{\mathbf{x}}(t) &= \mathbf{A}_{CL}\mathbf{x}(t) + \mathbf{B}\mathbf{K}_R\mathbf{r}(t) \\ \mathbf{y}(t) &= \mathbf{C}\mathbf{x}(t)\end{aligned}$$

We wish to find \mathbf{K}_R so that for step references, $\mathbf{r}(t) = \mathbf{r}_0$, the output $\mathbf{y}(t)$ reaches the correct output reference values, \mathbf{r}_0 .

23.3.2 Analysis for choice of \mathbf{K}_R

Let the constant steady state value of state $\mathbf{x}(t)$ be defined as \mathbf{x}_{ss} , then because the steady state is constant, $\dot{\mathbf{x}}_{ss} = \mathbf{0}$. This gives the equation

$$\dot{\mathbf{x}}_{ss} = \mathbf{A}_{CL}\mathbf{x}_{ss} + \mathbf{B}\mathbf{K}_R\mathbf{r}_0 = \mathbf{0}$$

This rearranges to give \mathbf{x}_{ss} as

$$\mathbf{x}_{ss} = -\mathbf{A}_{CL}^{-1}\mathbf{B}\mathbf{K}_R\mathbf{r}_0$$

Then, using the output equation,

$$\mathbf{y}_{ss} = \mathbf{C}\mathbf{x}_{ss} = \mathbf{C}(-\mathbf{A}_{CL}^{-1})\mathbf{B}\mathbf{K}_R\mathbf{r}_0 = [\mathbf{C}(-\mathbf{A}_{CL}^{-1})\mathbf{B}\mathbf{K}_R]\mathbf{r}_0$$

Since we require the value of \mathbf{y}_{ss} to equal the input reference, \mathbf{r}_0 , we set

$$[\mathbf{C}(-\mathbf{A}_{CL}^{-1})\mathbf{B}\mathbf{K}_R] = \mathbf{I}$$

Rearranging gives

$$\mathbf{K}_R = [\mathbf{C}(-\mathbf{A}_{CL}^{-1})\mathbf{B}]^{-1}$$

Thus to use the two degrees of freedom output control law

$$\mathbf{u}(t) = \mathbf{K}_R\mathbf{r}(t) - \mathbf{K}_0\mathbf{y}(t)$$

we have two calculations to consider:

1. Can we find feedback gain \mathbf{K}_0 such that the closed-loop system matrix $\mathbf{A}_{CL} = [\mathbf{A} - \mathbf{B}\mathbf{K}_0\mathbf{C}]$ is stable and gives satisfactory transient performance?
2. To attain the correct steady state reference levels in the output we must calculate the reference control gain $\mathbf{K}_R = [\mathbf{C}(-\mathbf{A}_{CL})^{-1}\mathbf{B}]^{-1}$, where $\mathbf{A}_{CL} = \mathbf{A} - \mathbf{B}\mathbf{K}_0\mathbf{C}$ is dependent on the chosen output controller gain \mathbf{K}_0 .

23.4 Pole placement design with output feedback

This is a simple idea that follows from the closed-loop state variable equations above. We have seen that the state variable system in closed loop has the equation

$$\begin{aligned} \dot{\mathbf{x}}(t) &= (\mathbf{A} - \mathbf{B}\mathbf{K}_0\mathbf{C})\mathbf{x}(t) + \mathbf{B}\mathbf{K}_R\mathbf{r}(t) \\ &= \mathbf{A}_{CL}\mathbf{x}(t) + \mathbf{B}\mathbf{K}_R\mathbf{r}(t) \end{aligned}$$

where the closed-loop system matrix is $\mathbf{A}_{CL} = \mathbf{A} - \mathbf{B}\mathbf{K}_0\mathbf{C}$. This matrix specifies the closed-loop poles of the control system. This condition can also be given through the closed-loop pole polynomial that specifies the eigenvalues of the closed-loop system matrix, \mathbf{A}_{CL} :

$$p_{CL}(s) = \det(s\mathbf{I} - \mathbf{A}_{CL}) = |s\mathbf{I} - \mathbf{A}_{CL}| = 0$$

However, we note that the closed-loop system matrix, $\mathbf{A}_{CL} = \mathbf{A} - \mathbf{B}\mathbf{K}_0\mathbf{C}$, is dependent on the controller gain, in this case, \mathbf{K}_0 . We also know that the closed-loop system matrix \mathbf{A}_{CL} will define the dynamic performance of the closed loop. So we can merge these two ideas to form a design method for the closed-loop dynamic behaviour of the state variable control system.

1. Turn the desired closed-loop design performance into a condition on closed-loop pole positions. Then put all of the desired pole locations into a design pole polynomial which we call $p_{design}(s)$.
2. Choose the available controller gain matrix so that the closed-loop pole polynomial, $p_{CL}(s)$ equals the design pole polynomial, $p_{design}(s)$.

Putting these two ideas together defines the pole placement design method. It is a fairly algebraic approach to closed-loop design, since it tends to lead to sets of equations which have to be solved for feedback gains. Indeed, we now continue to explore the output feedback control problem for systems describe by state variable models using a pole placement design framework.

Problem An engineer wishes to design the controller for a highly oscillatory electro-mechanical system. A modelling exercise leads to a state variable model with the following parameters:

$$\begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \end{bmatrix} = \begin{bmatrix} -0.25 & 6.00 \\ -6.00 & -0.25 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} + \begin{bmatrix} 0.4 \\ 1.5 \end{bmatrix} u(t)$$

and

$$y(t) = [0.73 \quad 1.82] \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix}$$

The engineer decides to investigate a two-degree-of-freedom output controller using the framework of Figure 23.3.

- What are the **A**, **B**, **C** data matrices for the electro-mechanical model?
- Use the open-loop data to show the response characteristics of the open-loop system.
- Use the general equations for **A_{CL}** and **K_R** to find expressions for these matrices from the system matrices.
- Use MATLAB to calculate the closed-loop system eigenvalues (poles) for different **K_o** values. Calculate the corresponding **K_R** values. Comment on the flexibility available in tuning closed-loop system parameters.
- Investigate whether pole placement design can be used for this example?
- Find the two-degrees-of-freedom design for $\zeta = 0.7$.
- Use Simulink to provide the transient responses for an example where damping takes the value 0.7.

Solution (a) From the given state variable model equations,

$$\begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \end{bmatrix} = \begin{bmatrix} -0.25 & 6.00 \\ -6.00 & -0.25 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} + \begin{bmatrix} 0.4 \\ 1.5 \end{bmatrix} u(t)$$

$$y(t) = [0.73 \quad 1.82] \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix}$$

we identify

$$\mathbf{A} = \begin{bmatrix} -0.25 & 6.00 \\ -6.00 & -0.25 \end{bmatrix} \quad \mathbf{B} = \begin{bmatrix} 0.4 \\ 1.5 \end{bmatrix} \quad \mathbf{C} = [0.73 \quad 1.82]$$

We note that the number of state variables, $n = 2$, that the number of control inputs $m = 1$, and the number of outputs is $r = 1$. This reveals the basic data for the system model and that this is a SISO system.

(b) To see the difficulty in controlling this problem, we find the open-loop system eigenvalues (poles). For this we solve

$$p_0(s) = \det(s\mathbf{I} - \mathbf{A}) = \begin{vmatrix} s+0.25 & -6 \\ 6 & s+0.25 \end{vmatrix} = s^2 + 0.5s + 36.0625 = 0$$

The open-loop pole polynomial, $p_0(s)$, shows a rather high value of natural frequency ($\omega_n = 6.005$) and a rather low value of damping ($\zeta = 0.5/(2 \times 6.005) = 0.04$). If we solve the quadratic equation, $p_0(s) = 0$, we find that the open-loop poles are located at $s = -0.25 \pm j6.005$. This leads us to expect highly oscillatory open-loop system responses that take a long time to die out. Such responses might be quite typical of an electro-mechanical system.

(c) The general equations for \mathbf{A}_{CL} and \mathbf{K}_R are

$$\mathbf{A}_{CL} = [\mathbf{A} - \mathbf{B}\mathbf{K}_0\mathbf{C}]$$

and

$$\mathbf{K}_R = [\mathbf{C}(-\mathbf{A}_{CL})^{-1}\mathbf{B}]^{-1}$$

We note from part (a) that $n = 2$ and $m = r = 1$, so that \mathbf{A} is 2×2 , \mathbf{B} is 2×1 , \mathbf{C} is 1×2 giving \mathbf{K}_0 as 1×1 , and \mathbf{A}_{CL} as 2×2 . If we let $\mathbf{K}_0 = [k_0]$ we can use the data of part (a) to obtain

$$\begin{aligned} \mathbf{A}_{CL} &= \mathbf{A} - \mathbf{B}k_0\mathbf{C} \\ &= \begin{bmatrix} -0.25 & 6.00 \\ -6.00 & -0.25 \end{bmatrix} - \begin{bmatrix} 0.4 \\ 1.5 \end{bmatrix} [k_0] \begin{bmatrix} 0.73 & 1.82 \end{bmatrix} \\ &= \begin{bmatrix} -0.25 - 0.29k_0 & 6.00 - 0.73k_0 \\ -6.00 - 1.09k_0 & -0.25 - 2.73k_0 \end{bmatrix} \end{aligned}$$

This gives \mathbf{A}_{CL} as a function of the output controller gain, k_0 , and once $\mathbf{K}_0 = [k_0]$ is selected the formula $\mathbf{K}_R = [\mathbf{C}(-\mathbf{A}_{CL})^{-1}\mathbf{B}]^{-1}$ can be used directly.

(d) The closed-loop system matrix is 2×2 and hence will have two closed-loop eigenvalues. We can use standard routines in MATLAB to work out the eigenvalues. Given that the closed-loop eigenvalues form a complex conjugate pair, we can also evaluate damping ζ and natural frequency ω_n for the closed-loop system. If we have eigenvalues, s , as $s_{1,2} = \alpha \pm j\beta = -\omega_n\zeta \pm j\omega_n\sqrt{1-\zeta^2}$ then ω_n can be calculated by remembering that the natural frequency is the distance of the pole (eigenvalue) from the origin:

$$\omega_n = |s| = (\alpha^2 + \beta^2)^{1/2}$$

and the damping ratio can then be calculated from the real part of the poles: $\zeta = -\alpha/\omega_n$.

M The above was coded as the MATLAB M-file:

```
% Set up System matrices
a(1,1)=-0.25;a(1,2)=6;
a(2,1)=-6; a(2,2)=-0.25;
b(1,1)=0.4;
b(2,1)=1.5;
c(1,1)=0.73; c(1,2)=1.82;
A=a;B=b;C=c;
%Input a value for K0
K0(1,1)=input('Input a value for K0')

%Calculate closed-loop system matrix and eigenvalues
ACL=A-B*K0*C
e=eig(ACL)
% Calculate omega_n and damping:
```

```

% since two complex eigenvalues, calculate magnitude of
% either to give omega_n
omega_n=abs(e(1,1))
damping =-real(e(1,1))/omega_n
%Reference Control Gain
KR=inv(-C*inv(ACL)*B)

```

The results from the MATLAB file may be tabulated as shown in Table 23.1.

Table 23.1 MATLAB results.

Output gain , k_o	Eigenvalues (A_{CL})	Damping ζ	Natural frequency ω_n	Reference gain $K_R(k_o)$
0	$-0.25 \pm j6$	0.04	6.005	12.19
0.5	$-1.01 \pm j6.04$	0.16	6.13	12.69
1.0	$-1.76 \pm j5.99$	0.28	6.25	13.19
1.5	$-2.52 \pm j5.85$	0.39	6.36	13.69
2.0	$-3.27 \pm j5.59$	0.51	6.48	14.19
2.5	$-4.03 \pm j5.22$	0.61	6.59	14.69
3.0	$-4.78 \pm j4.70$	0.71	6.70	15.19

What can be seen from the table is that we have only one gain parameter k_o that is changing two closed-loop eigenvalue locations. In fact, the table of results shows that as we change k_o we make the real part of the closed-loop eigenvalue position increasingly negative while not changing the imaginary value greatly. We find that the damping factor changes significantly while the value of ω_n has only marginal changes. We note that we cannot tune closed-loop system damping ζ and natural frequency ω_n independently.

(e) In this section of the problem, we are going to look at the theoretical relationship between closed-loop eigenvalue positions and the choice of k_o . We find the eigenvalue of A_{CL} from $\det[sI - A_{CL}] = 0$ as follows:

$$\begin{aligned}
 p_{CL}(s) = \det[sI - A_{CL}] &= \begin{vmatrix} s - (-0.25 - 0.29k_o) & -(6 - 0.73k_o) \\ -(-6 - 1.09k_o) & s - (-0.25 - 2.73k_o) \end{vmatrix} \\
 &= s^2 + (0.5 + 3.02k_o)s + (36.0625 + 2.915k_o)
 \end{aligned}$$

This second-order eigenvalue relationship can be compared with the usual second-order underdamped system relationship $p(s) = s^2 + 2\zeta\omega_n s + \omega_n^2$ to yield the equations

$$\begin{aligned}
 \omega_n &= \sqrt{(36.0625 + 2.915k_o)} \\
 \zeta &= \frac{(0.5 + 3.02k_o)}{(2 \times \omega_n)} = \frac{0.25 + 1.51k_o}{\sqrt{(36.0625 + 2.915k_o)}}
 \end{aligned}$$

These second-order system relationships show that ω_n and ζ cannot be tuned independently by selection of output gain, k_o . This implies that a full pole placement solution cannot be obtained and that we have to settle for only partial control over the closed-loop pole locations.

(f) To find the two degree of freedom output control law gain which yields $\zeta = 0.7$, we solve

$$\zeta = \frac{0.25 + 1.51k_o}{(36.0625 + 2.915k_o)^{1/2}} = 0.7$$

Squaring up and rearranging gives

$$\frac{0.0625 + 0.755k_o + 2.28k_o^2}{(36.0625 + 2.915k_o)} = 0.49$$

Further rearrangement gives a quadratic for k_o as $k_o^2 - 0.296k_o - 7.72 = 0$, from which two values are obtained for k_o , namely, $k_o = 2.93$ or $k_o = -2.63$. The negative value for k_o is inadmissible. A run of the MATLAB program devised above with $k_o = 2.93$ gives the following results for the closed-loop system parameters.

k_o	Eigenvalues	ζ	ω_n	$K_R(k_o = 2.93)$
2.93	$-4.68 \pm j4.78$	0.699	6.69	15.12

The required output feedback law is $u(t) = 15.12r(t) - 2.93 y(t)$. The Simulink simulation (part (g)) follows directly, as shown in Figure 23.4 and the output response is shown in Figure 23.5.

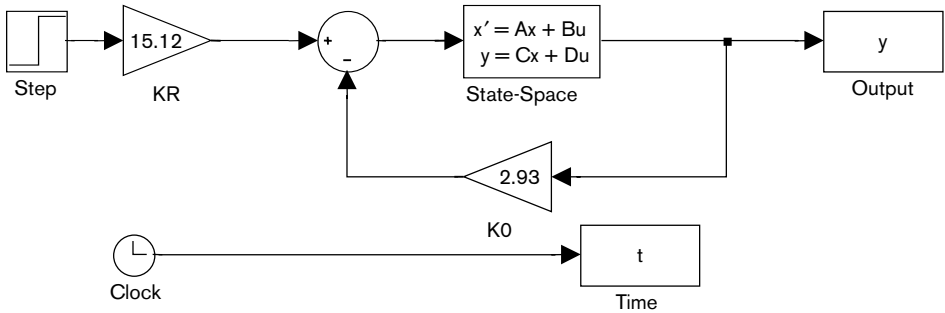


Figure 23.4 The Simulink simulation for output feedback solution.

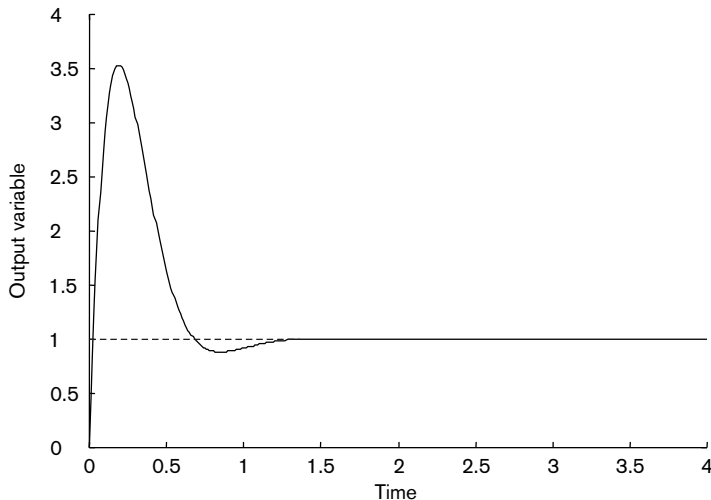


Figure 23.5 The output response for design where $\zeta = 0.7$.

23.4.1 Analysis of results

We see from Figure 23.5 that the output attains the correct steady state reference level. However, the response is seen to have an overshoot of over 350% which does not satisfy the design specification of $\zeta = 0.7$. We found from our analysis that we would not be able to satisfy specifications on both ζ and ω_n with our output feedback, but we did expect to have met the damping ratio specification. There will be reasons for this, for example:

1. The design specification was based on second-order system analysis: if the system is of higher order then we would not expect to have an exact match between our second-order analysis and results from higher order systems.
2. Use of state variable models can sometimes obscure the pole-zero knowledge of the system which we are more familiar with.
3. The model of the system may not have accurately represented the actual dynamics.

In this example, we find that the system is second-order (two states), but if we convert the state variable model to transfer function model, we find that the system transfer function is given by:

$$G(s) = \frac{3.025(s + 0.9787)}{s^2 + 0.5s + 36.06}$$

We note that there is one zero in the numerator at $s = -0.9787$. From our knowledge of poles and zeros, we remember that although the system stability is determined by the poles of the system, the transient response can be affected by the zeros. To gain further insight, we can look at a root locus plot (Figure 23.6).

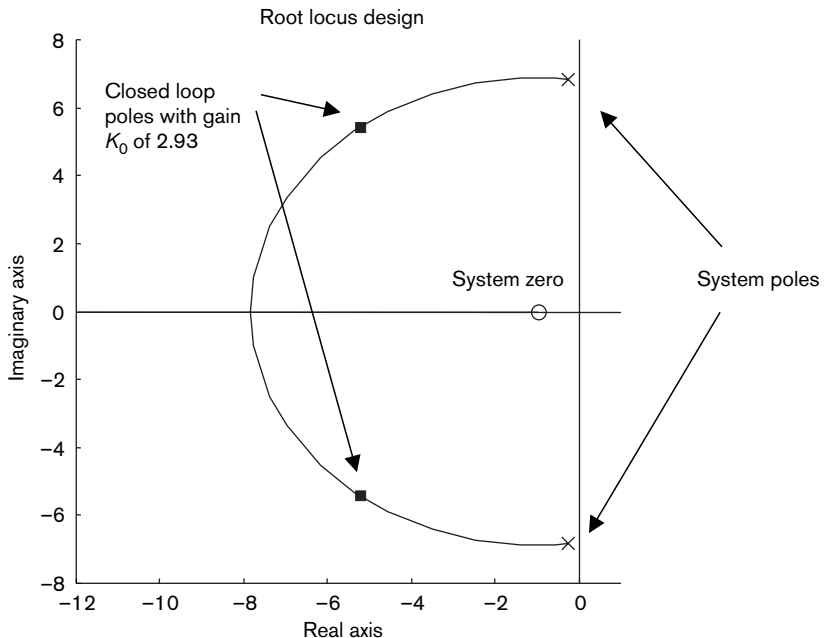


Figure 23.6 Root locus plot for output feedback of electro-mechanical system.

We find that for a gain of $\mathbf{K}_0 = 2.93$, the complex poles move along the locus to the positions shown. The closed-loop system still has complex poles, but the zero of the system transfer function remains. Since the zero is closer to the origin than the poles, the system does not have ‘dominant second-order poles’ and therefore the response will be significantly altered from the expected second-order response. This is what we have seen in Figure 23.5, where the overshoot is excessive. In practice, there would usually be an actuator in the loop that would have limited this overshoot, and also provided some saturation nonlinearity within the system.

This problem shows:

1. That we should be careful when performing our initial system analysis and that we should stop and analyse our results if they appear at odds with what we expect to see.
2. That output feedback in a state variable framework does not necessarily have sufficient degrees of freedom to satisfy all design requirements. We can see this from the general form of the closed-loop system matrix, $\mathbf{A}_{CL} = \mathbf{A} - \mathbf{B}\mathbf{K}_0\mathbf{C}$. The output feedback gain \mathbf{K}_0 will have $r \times m$ parameters to tune the n dynamic eigenvalues of \mathbf{A}_{CL} . In most real systems the order of the system n will be very much greater than the number of measurements and/or control, and so sufficient degrees of design freedom are not usually available in this form of state variable system method. One of the important issues here is that the output vector is only a partial view of the state vector, \mathbf{x} . As we have seen from the output equation, the state passes through the output matrix \mathbf{C} to give the output variables in the vector, \mathbf{y} .

In the next section we investigate what happens if we can remove this constraint and have direct access to all the state variables.

23.5 Investigating state feedback: using the state vector directly

Output feedback is the control structure that we have used for most of the book so far. In the investigation that we performed in the previous section for output feedback using the framework of a state variable system model, we found some restrictions in design freedom. The example we used was a single-input, single-output electromechanical system model with two states. A two-degrees-of-freedom control law enabled us to consider one control gain for achieving the correct steady state values, and a second control gain for shaping the dynamic response of the system. Simple calculations showed us that we were not able to tune the transient response of the system adequately. It was intimated that this was due to the information-reducing property of the output equation, which only allowed a partial view of the full information content of the state vector, $\mathbf{x}(t)$. In this section we follow this idea further and instead of using the output in a feedback law, we investigate the outcome of using the state vector *directly* in a state feedback law.

The two degrees of freedom *output* feedback control law was given by

$$\mathbf{u}(t) = \mathbf{K}_R\mathbf{r}(t) - \mathbf{K}_0\mathbf{y}(t)$$

If we assume that we have a set of measurements (or a system model) available that allow a *full* view of the state vector, we can consider using a two-degree-of-freedom *state* feedback law of the form

$$\mathbf{u}(t) = \mathbf{K}_R\mathbf{r}(t) - \mathbf{K}\mathbf{x}(t)$$

The closed-loop system block diagram for the state feedback law is given in Figure 23.7. This shows the same basic reference tracking structure as the output feedback case

shown in Figure 23.3. The main difference is that the internal system state is now providing the feedback action through a state feedback gain, \mathbf{K} . We should note that the gain, \mathbf{K} , is an $m \times n$ matrix and that this is generally larger in size than the corresponding output feedback gain, \mathbf{K}_O , which was only an $m \times r$ matrix. We anticipate that we should be able to achieve more with state feedback.

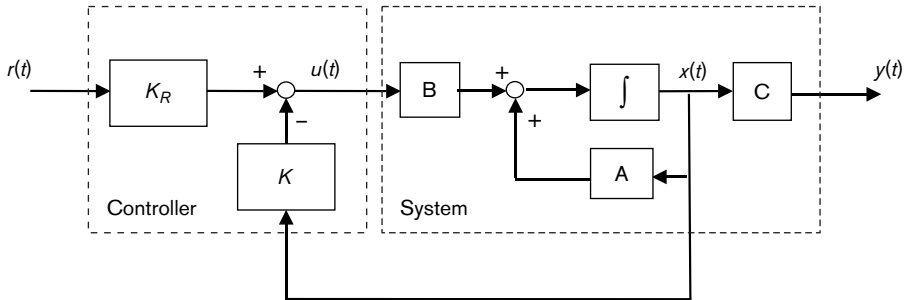


Figure 23.7 State feedback control for reference tracking.

Example: Extra design freedom through state feedback

We consider a system with one input, one output and two state variables. Therefore, for a single input and single output we set $m = r = 1$, and for two state variables we set $n = 2$. In the state feedback control law, the reference control gain matrix \mathbf{K}_R is now a 1×1 matrix, or a scalar, $[k_R]$. The state feedback gain, $\mathbf{K} = [k(1,1) \ k(1,2)]$, is a 1×2 matrix. Thus we can write the control law in matrix form as

$$\mathbf{u}(t) = \mathbf{K}_R r(t) - \mathbf{K} \mathbf{x}(t)$$

or we can use matrix element form as

$$u(t) = [k_R][r(t)] - [k(1,1) \ k(1,2)] \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix}$$

It is not difficult to see that the matrix form is generally easier to work with and that we usually only use the element-wise form when we consider the particular details of a given problem. We can recall that the equivalent output feedback law for a single-input, single-output system gave rise to a matrix form in which both the reference and the control gain matrices were scalars:

$$u(t) = [k_R]r(t) - [k_O]y(t)$$

Hence the state feedback law shows an extra degree of design freedom in the control gain by having an extra controller gain to tune, and we expect to be able to achieve better control tuning with the new flexibility.

23.5.1 Closed-loop system analysis for general state feedback system

The closed-loop control analysis follows from the system equations and is similar to that performed for output feedback, only we use the state feedback control law in place of the output feedback control equation:

$$\dot{\mathbf{x}}(t) = \mathbf{A} \mathbf{x}(t) + \mathbf{B} \mathbf{u}(t)$$

and the two degrees of freedom state feedback law:

$$\mathbf{u}(t) = \mathbf{K}_R \mathbf{r}(t) - \mathbf{K} \mathbf{x}(t)$$

1. To find the closed-loop system matrix we use direct substitution of the control law into the system equation:

$$\begin{aligned}\dot{\mathbf{x}}(t) &= \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t) \\ &= \mathbf{A}\mathbf{x}(t) + \mathbf{B}(\mathbf{K}_R\mathbf{r}(t) - \mathbf{K}\mathbf{x}(t)) \\ &= [\mathbf{A} - \mathbf{B}\mathbf{K}]\mathbf{x}(t) + \mathbf{B}\mathbf{K}_R\mathbf{r}(t) \\ &= [\mathbf{A}_{CL}]\mathbf{x}(t) + \mathbf{B}\mathbf{K}_R\mathbf{r}(t)\end{aligned}$$

The closed-loop system matrix is then given by $\mathbf{A}_{CL} = [\mathbf{A} - \mathbf{B}\mathbf{K}]$.

2. To find the reference controller \mathbf{K}_R we assume that a state feedback gain can be found to stabilise the closed-loop so that all the eigenvalues of \mathbf{A}_{CL} have a strictly negative real part. This allows the inversion of matrix $\mathbf{A}_{CL} = [\mathbf{A} - \mathbf{B}\mathbf{K}]$. We then assume that the constant reference signals $\mathbf{r}(t) = \mathbf{r}_o$ cause the states to reach constant levels as $t \rightarrow \infty$. We denote this steady level as \mathbf{x}_{ss} and use the property that $\dot{\mathbf{x}}_{ss} = \mathbf{0}$ to obtain

$$\dot{\mathbf{x}}_{ss} = \mathbf{A}_{CL}\mathbf{x}_{ss} + \mathbf{B}\mathbf{K}_R\mathbf{r}_o = \mathbf{0}$$

This rearranges to give

$$\mathbf{x}_{ss} = -\mathbf{A}_{CL}^{-1}\mathbf{B}\mathbf{K}_R\mathbf{r}_o$$

Then, using the output equation, $\mathbf{y}(t) = \mathbf{C}\mathbf{x}(t)$, we find

$$\mathbf{y}_{ss} = \mathbf{C}\mathbf{x}_{ss} = \mathbf{C}(-\mathbf{A}_{CL}^{-1})\mathbf{B}\mathbf{K}_R\mathbf{r}_o = [\mathbf{C}(-\mathbf{A}_{CL}^{-1})\mathbf{B}\mathbf{K}_R]\mathbf{r}_o$$

Since we required that $\mathbf{y}_{ss} = \mathbf{r}_o$, we find

$$[\mathbf{C}(-\mathbf{A}_{CL}^{-1})\mathbf{B}\mathbf{K}_R] = \mathbf{I}$$

and

$$\mathbf{K}_R = [\mathbf{C}(-\mathbf{A}_{CL}^{-1})\mathbf{B}]^{-1}$$

Key result: State feedback control

The important difference from the previous analysis for the output feedback case is that the closed-loop matrix here, $\mathbf{A}_{CL} = [\mathbf{A} - \mathbf{B}\mathbf{K}]$, is a function of the state feedback gain matrix, \mathbf{K} .

Thus overall, the state feedback closed-loop analysis also leads to two matrix expressions, a closed-loop matrix, $\mathbf{A}_{CL} = [\mathbf{A} - \mathbf{B}\mathbf{K}]$ and a reference controller gain matrix, $\mathbf{K}_R = [\mathbf{C}(-\mathbf{A}_{CL})^{-1}\mathbf{B}]^{-1}$.

The design problem is therefore similar to that for the output feedback case, and the pole placement method is a possible solution procedure.

To investigate the success or failure of state feedback, we re-examine the lightly damped single-input, single-output electro-mechanical example of the previous section that has two states. We use this example to be able to make a comparison of the outcomes of output feedback and state feedback designs for the same system.

Problem The state variable model for an electro-mechanical system has been derived with the following parameters:

$$\begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \end{bmatrix} = \begin{bmatrix} -0.25 & 6.00 \\ -6.00 & -0.25 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} + \begin{bmatrix} 0.4 \\ 1.5 \end{bmatrix} u(t)$$

$$y(t) = [0.73 \quad 1.82] \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix}$$

The engineer on this control project decides to investigate the use of state feedback control using the two degrees of freedom control law:

$$\mathbf{u}(t) = \mathbf{K}_R \mathbf{r}(t) - \mathbf{K} \mathbf{x}(t)$$

- Summarise the open-loop characteristics of this system model.
- Use the coefficient form of the state feedback matrix \mathbf{K} to determine the pole placement equations for closed-loop design.
- Discuss the flexibility available in the design equations. Use MATLAB to determine the state feedback gains and the reference controller gain for the two design cases:

Case (a) Underdamped design $\zeta = \frac{1}{\sqrt{2}}, \omega_n = \sqrt{2}$

Case (b) Critically damped design $\zeta = 1, \omega_n = 1$

- Develop a Simulink simulation to show the system unit step responses resulting from the designs. Comment on the results found.

Solution (a) The open-loop characteristics of the electro-mechanical system model are determined from the system matrix \mathbf{A} which was previously identified as

$$\mathbf{A} = \begin{bmatrix} -0.25 & 6.00 \\ -6.00 & -0.25 \end{bmatrix}$$

To see the difficulty in controlling this problem, the open-loop system eigenvalues (poles) can be examined from the open-loop eigenvalue polynomial

$$p_0(s) = \det(s\mathbf{I} - \mathbf{A}) = \begin{vmatrix} s+0.25 & -6 \\ 6 & s+0.25 \end{vmatrix} = s^2 + 0.5s + 36.0625 = 0$$

The open-loop poles are located at $s = -0.25 \pm j6.005$. The poles correspond to positions associated with low damping, $\zeta = 0.04$, and a high value of natural frequency, $\omega_n = 6.005$. Thus open-loop responses are highly oscillatory and will take a long time to settle. If the system transfer function is evaluated:

$$G(s) = \frac{3.025(s+0.9787)}{s^2 + 0.5s + 36.06}$$

it then becomes apparent that there is an open-loop zero at $s_z = -0.9787$. This zero will have effect on the time responses of the system.

- We use the following equations in our design procedure:

$$\mathbf{A}_{CL} = [\mathbf{A} - \mathbf{BK}] \quad \text{and} \quad \mathbf{K}_R = [\mathbf{C}(-\mathbf{A}_{CL})^{-1}\mathbf{B}]^{-1}$$

Since we have a second-order system, we can perform some general analysis to show the advantages of the state feedback design method. In particular, we use the coefficient form of the

state-feedback matrix \mathbf{K} to determine the general pole placement equations for our second-order closed-loop design.

The pole placement analysis follows the procedure:

1. Find the general pole placement coefficient equations for our second-order closed-loop design.
2. Equate the coefficients of the second-order closed-loop pole polynomial to a general second-order design polynomial.
3. Write down the linear set of equations that results in a form suitable for solution by MATLAB.
4. Then, evaluating the parameters of the design pole polynomial from the design specifications, run the program to find the resulting coefficients of the controller gain matrix, \mathbf{K} .
5. Use the controller gain \mathbf{K} in the equation for \mathbf{K}_R to find the reference tracking gain.

We begin the pole placement procedure at Step 1.

1. We wish to calculate the closed-loop system matrix expression, $\mathbf{A}_{CL} = [\mathbf{A} - \mathbf{BK}]$. From the state variable model we identify matrices \mathbf{A} and \mathbf{B} as:

$$\mathbf{A} = \begin{bmatrix} -0.25 & 6.00 \\ -6.00 & -0.25 \end{bmatrix} \quad \text{and} \quad \mathbf{B} = \begin{bmatrix} 0.4 \\ 1.5 \end{bmatrix}$$

where \mathbf{A} is 2×2 and \mathbf{B} is 2×1 and \mathbf{K} is 1×2 . We let the coefficient form of the state-feedback matrix be $\mathbf{K} = [k_1 \ k_2]$.

To investigate the design freedom we examine the eigenvalues of the closed-loop system, and first produce a set of equations which define all the possible pole placement positions for the closed-loop system matrix. To keep the procedure and the subsequent programs general we will perform the next steps algebraically. Although a little tedious, this will make the MATLAB program more useful, since we will be able to change the system data easily. The closed-loop system matrix is given by

$$\mathbf{A}_{CL} = \mathbf{A} - \mathbf{BK} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} - \begin{bmatrix} b_{11} \\ b_{21} \end{bmatrix} [k_1 \ k_2] = \begin{bmatrix} a_{11} - b_{11}k_1 & a_{12} - b_{11}k_2 \\ a_{21} - b_{21}k_1 & a_{22} - b_{21}k_2 \end{bmatrix}$$

The eigenvalue polynomial for the closed-loop system matrix can now be found:

$$\begin{aligned} \rho_{CL}(s) &= |s\mathbf{I} - \mathbf{A}_{CL}| = \begin{vmatrix} s - a_{11} + b_{11}k_1 & -a_{12} + b_{11}k_2 \\ -a_{21} + b_{21}k_1 & s - a_{22} + b_{21}k_2 \end{vmatrix} \\ &= (s - a_{11} + b_{11}k_1)(s - a_{22} + b_{21}k_2) - (-a_{21} + b_{21}k_1)(-a_{12} + b_{11}k_2) \\ &= s^2 + (-a_{11} + b_{11}k_1 - a_{22} + b_{21}k_2)s + (-a_{11} + b_{11}k_1)(-a_{22} + b_{21}k_2) \\ &\quad - (-a_{21} + b_{21}k_1)(-a_{12} + b_{11}k_2) \end{aligned}$$

We find that when multiplying these expressions out, the cross-term in k_1k_2 always disappears from this type of problem. Hence we have a closed-loop polynomial as

$$\begin{aligned} \rho_{CL}(s) &= s^2 + (b_{11}k_1 + b_{21}k_2 - a_{11} - a_{22})s + (-b_{11}a_{22} + b_{21}a_{12})k_1 \\ &\quad + (-a_{11}b_{21} + a_{21}b_{11})k_2 + (a_{11}a_{22} - a_{21}a_{12}) \end{aligned}$$

The quadratic, $\rho_{CL}(s)$, gives all the locations of the eigenvalue positions for the closed-loop system matrix \mathbf{A}_{CL} in terms of the elements of the state feedback gain, \mathbf{K} .

2. In a pole placement design, we can assume that the design specification have been transformed into a design closed-loop system eigenvalue quadratic, $p_{\text{design}}(s)$. We can then equate coefficients to obtain some equations for the elements of the gain matrix, \mathbf{K} . Let the design eigenvalue polynomial be given by

$$p_{\text{design}}(s) = s^2 + t_1 s + t_2$$

Equating coefficients of the design polynomial $p_{\text{design}}(s)$ and the closed-loop pole polynomial, $p_{\text{CL}}(s)$, gives

$$\text{Coefficients of } s^2: \quad 1 = 1$$

$$\text{Coefficients of } s^1: \quad t_1 = b_{11}k_1 + b_{21}k_2 - a_{11} - a_{22}$$

$$\text{Coefficients of } s^0: \quad t_2 = (-b_{11}a_{22} + b_{21}a_{12})k_1 + (-a_{11}b_{21} + a_{21}b_{11})k_2 + (a_{11}a_{22} - a_{21}a_{12})$$

3. We rewrite these equations into a set of linear equations for the gains k_1 and k_2 :

$$\begin{aligned} b_{11}k_1 + b_{21}k_2 &= t_1 + a_{11} + a_{22} \\ (-b_{11}a_{22} + b_{21}a_{12})k_1 + (-a_{11}b_{21} + a_{21}b_{11})k_2 &= t_2 - (a_{11}a_{22} - a_{21}a_{12}) \end{aligned}$$

This can be written as

$$\begin{bmatrix} X(1,1) & X(1,2) \\ X(2,1) & X(2,2) \end{bmatrix} \begin{bmatrix} k_1 \\ k_2 \end{bmatrix} = \begin{bmatrix} Y(1,1) \\ Y(2,1) \end{bmatrix}$$

where the elements are given by

$$\begin{aligned} X(1,1) &= b_{11} & X(1,2) &= b_{21} \\ X(2,1) &= -b_{11}a_{22} + b_{21}a_{12} & X(2,2) &= -a_{11}b_{21} + a_{21}b_{11} \\ Y(1,1) &= t_1 + a_{11} + a_{22} \\ Y(2,1) &= t_2 - (a_{11}a_{22} - a_{21}a_{12}) \end{aligned}$$

This can be given in matrix form as

$$\mathbf{XK} = \mathbf{Y}, \text{ giving } \mathbf{K} = \mathbf{X}^{-1}\mathbf{Y}$$

Although the above seems a little long-winded, it does give us a much neater MATLAB program. This program will have the steps as follows:

- Step 1: Set up \mathbf{A} , \mathbf{B} data.
- Step 2: Specify the design polynomial data, t_1 , t_2 .
- Step 3: Set up \mathbf{X} and \mathbf{Y} matrices.
- Step 4: Use $\mathbf{K} = \mathbf{X}^{-1}\mathbf{Y}$ to find \mathbf{K} .
- Step 5: Use $\mathbf{A}_{\text{CL}} = \mathbf{A} - \mathbf{BK}$ to find the closed-loop system matrix.
- Step 6: Use $\mathbf{K}_R = [\mathbf{C}(\mathbf{A}_{\text{CL}}^{-1})\mathbf{B}]^{-1}$ to find reference control gain.

Thus the element-wise expression for the closed-loop matrix, \mathbf{A}_{CL} , leads to the design equations, and the formula $\mathbf{K}_R = [\mathbf{C}(\mathbf{A}_{\text{CL}}^{-1})\mathbf{B}]^{-1}$ gives the value of the reference gain.

- (c) The design equations found above require the design pole polynomial, $p_{\text{design}}(s) = s^2 + t_1 s + t_2$, to be created from the design specification. We can look at the pole assignment from two viewpoints.

From a practical viewpoint we often have an intuitive idea of how fast we should like the modes to decay and what region of the s -plane we should like to see the closed-loop poles occupy. So we could specify the closed-loop system modes to take desired values, $\lambda = -\alpha \pm j\beta$. We can turn this directly into a target eigenvalue polynomial:

$$p_{\text{design}}(s) = s^2 + 2\alpha s + (\alpha^2 + \beta^2) = s^2 + t_1 s + t_2$$

From this we can specify coefficients $t_1 = 2\alpha$ and $t_2 = (\alpha^2 + \beta^2)$.

Alternatively, we could take the design route in which we specify a desired damping ζ and natural frequency, ω_n . We then use the usual second-order underdamped polynomial for specifying t_1 and t_2 . We set

$$p_{\text{design}}(s) = s^2 + 2\zeta\omega_n s + \omega_n^2 = s^2 + t_1 s + t_2$$

and obtain $t_1 = 2\zeta\omega_n$, $t_2 = \omega_n^2$.

M Presuming we have already set up the matrix coefficients in MATLAB, the design equations can then be added to the system setup.

```
%Design parameters
t1=2; t2=1;
%Gain Equation Set up
X(1,1)=b(1,1);
X(1,2)=b(2,1);
Z(1,1)=t1+a(1,1)+a(2,2);
X(2,1)=-b(1,1)*a(2,2)+b(2,1)*a(1,2);
X(2,2)=-b(2,1)*a(1,1)+b(1,1)*a(2,1);
Z(2,1)=t2-(a(1,1)*a(2,2)-a(2,1)*a(1,2));
%Gain Equation Solution
Y=inv(X)*Z;
%Closed-loop System Matrix
k(1,1)=Y(1,1); k(1,2)=Y(2,1);
K=k;
%Calculate closed-loop system matrix
ACL=A-B*K;
%Reference Control Gain
KR=inv(-C*inv(ACL)*B)
```

We can check the closed-loop eigenvalues as we did for the output feedback example by using `eig(ACL)`.

Program verification

It is important to verify the program and we use the data of the given model to check the program calculations. We have system data:

$$\begin{aligned} a_{11} &= -0.25 & a_{12} &= 6.00 & b_{11} &= 0.4 \\ a_{21} &= -6.00 & a_{22} &= -0.25 & b_{21} &= 1.5 \\ c_{11} &= 0.73 & c_{12} &= 1.82 \end{aligned}$$

We can verify the following element calculations,

$$X(1,1) = b_{11} = 0.4$$

$$X(1,2) = b_{21} = 1.5$$

$$X(2,1) = -b_{11}a_{22} + b_{21}a_{12} = 9.1$$

$$X(2,2) = a_{11}b_{21} + a_{21}b_{11} = -2.025$$

$$Y(1,1) = t_1 + a_{11} + a_{22} = t_1 - 0.5$$

$$Y(2,1) = t_2 - (a_{11}a_{22} + a_{21}a_{12}) = t_2 - 36.0625$$

(d) Finding the design parameters t_1 and t_2 .

Case (a): Underdamped design where $\zeta = 1/\sqrt{2}$, $\omega_n = \sqrt{2}$.

We seek a design polynomial of the form:

$$p_{\text{design}}(s) = s^2 + 2\zeta\omega_n s + \omega_n^2 = s^2 + t_1 s + t_2$$

and use $t_1 = 2\zeta\omega_n$, $t_2 = \omega_n^2$. We can calculate that:

$$t_1 = 2\zeta\omega_n = 2 \times \frac{1}{\sqrt{2}} \times \sqrt{2} = 2$$

$$t_2 = \omega_n^2 = (\sqrt{2})^2 = 2$$

Case (b): Critically damped design where $\zeta = 1$, $\omega_n = 1$.

We seek a target polynomial of the form

$$p_{\text{design}}(s) = s^2 + 2\zeta\omega_n s + \omega_n^2 = s^2 + t_1 s + t_2$$

and use $t_1 = 2\zeta\omega_n$, $t_2 = \omega_n^2$. We can calculate that

$$t_1 = 2\zeta\omega_n = 2 \times 1 \times 1 = 2 \text{ and } t_2 = \omega_n^2 = (1)^2 = 1$$

Using MATLAB we find the results shown in Table 23.2.

Table 23.2 Results for underdamped and critically damped design cases.

Case (a)	Underdamped design				
\mathbf{K}_R	$K(1,1)$	$K(1,2)$	CL poles	Calculated ζ	Calculated ω_n
0.6762	-3.323	1.886	$s = -1 \pm j1$	0.7071	1.414
Case (b)	Critically damped design				
\mathbf{K}_R	$K(1,1)$	$K(1,2)$	CL poles	Calculated ζ	Calculated ω_n
0.3381	-3.427	1.914	$s = -1 \pm j0$	1.0	1.0

(e) We present two versions for the Simulink simulation: one using the full matrix notation, and one showing the matrix coefficients. Both give the same results.

(i) The Simulink simulation using full matrix notation is given in Figure 23.8.

Implementation note: to use the state space block for our problem, we have set the \mathbf{C} matrix to be the identity matrix. This provides an output vector from the state space block which contains the states, $\mathbf{y}(t) = \mathbf{C}\mathbf{x}(t) = \mathbf{I}\mathbf{x}(t) = \mathbf{x}(t)$. This has additional implications for the size of the output vector $\mathbf{y}(t)$, which must now be the same dimensions as $\mathbf{x}(t)$, and the size of the (zero) \mathbf{D} matrix.

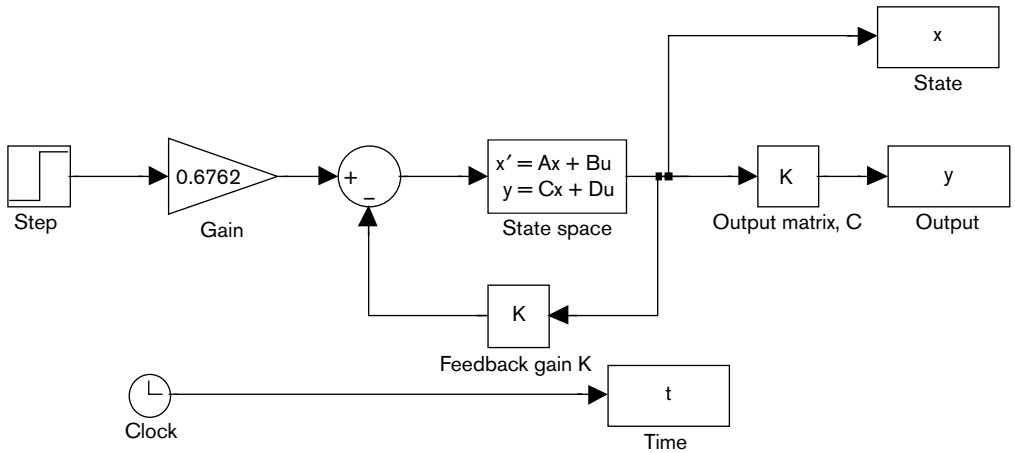


Figure 23.8 Simulink state feedback simulation using matrix notation.

(ii) The Simulink simulation giving gain coefficients is shown in Figure 23.9. Once again we have had to set the **C** matrix in the state space block to be an identity matrix in order that the output of the block is the state vector $x(t)$. We then pass this through a Demux block which allows us access to the individual state vector components.

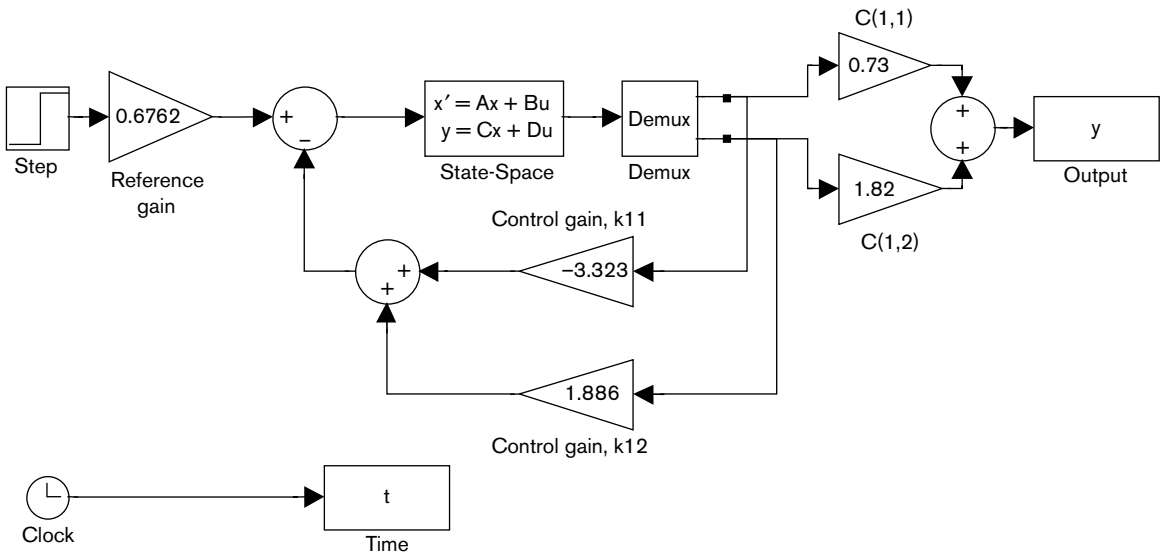


Figure 23.9 Simulink state feedback simulation using matrix gain coefficients.

Analysis of results

The Case (a) underdamped response is shown in Figure 23.10, and the Case (b) critically underdamped response is shown in Figure 23.11. We see that the shape of the dynamic response has been much tamed. We have an overshoot of 21.6% in the underdamped case and no overshoot in the critically damped case. In both cases the correct steady state values have been achieved. In the underdamped case the overshoot is higher than what we required, since a design damping of

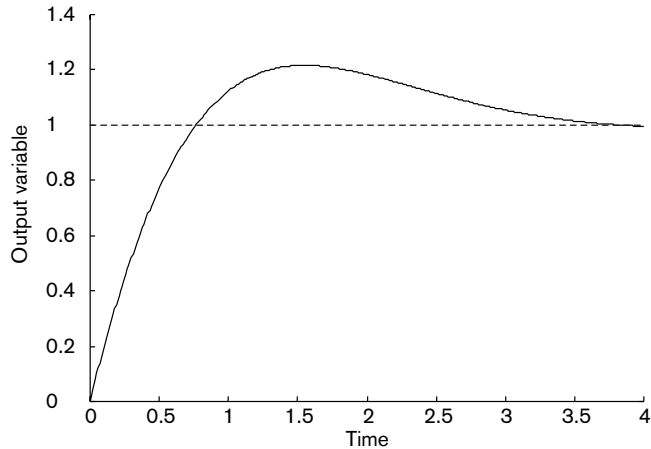


Figure 23.10 Case (a) underdamped output response.

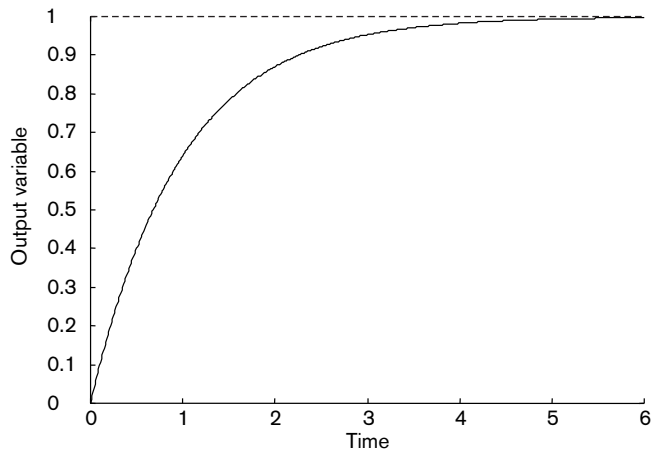


Figure 23.11 Case (b) critically damped output response.

0.7 translates as an overshoot of about 5%, and we have 21.6%. But we note from the results analysis at the end of the output feedback case that the system has a zero close to the origin which affects the transient response. The effect of the zero is to give a non-standard second-order system response. However, what we do notice is that the step response has improved greatly due to the use of state feedback.

M MATLAB has a command `acker` which provides a quick and convenient route to performing pole-placement design with a simple state feedback system. The command is

$$K = \text{acker}(A, B, P)$$

where A and B are the system and input matrices from the state variable model. P represents the vector containing the desired pole locations. If we apply this to the previous example we find:

```

A = [-0.25 6; -6 -0.25];    % Enter system matrices
B = [ 0.4; 1.5 ];
p1= roots( [ 1 2 2]);      % Calculate first desired pole locations
p2 = roots( [ 1 2 1]);      % Calculate second desired pole locations
KR1 = acker(A,B,p1)         % Calculate first control gain
    KR1
        -3.3234    1.8862
KR2 = acker(A,B,p2)         % Calculate second control gain
    KR2
        -3.4271    1.9139

```

We see that the results from the control gains KR1 and KR2 are the same as those given in Table 23.2 in the previous problem.

23.5.2 Summary: Output and state feedback in a state variable system framework

We have already discussed the similarities between the two laws:

$$\text{output feedback law} \quad \mathbf{u}(t) = \mathbf{K}_R \mathbf{r}(t) - \mathbf{K}_O \mathbf{y}(t)$$

$$\text{state feedback law} \quad \mathbf{u}(t) = \mathbf{K}_R \mathbf{r}(t) - \mathbf{K} \mathbf{x}(t)$$

The output feedback law is restricted in design achievements while the state feedback law is able to give total control over the system dynamics. Indeed, the design flexibility of the state feedback law can be supported by deep technical results to guarantee the design properties. But theory is always what it says it is: *just theory*, which has to be matched against the practical reality of the real world. The design flexibility of state feedback is achieved because it has been assumed that we can access each state variable. In practice this means that we must be able to measure every state variable, or we must have a more complicated control system where we include a model of the process which provides us with the state information. Further, we have also assumed that the model is a very accurate representation of the actual process being controlled. We find that most industrial control systems will have many unmeasurable system states and that industrial process models are not usually so well known or accurate. These problems lead us into the following three control design areas that rely on state feedback.

State feedback controllers

1. Observers

These are controllers which incorporate models of the process and are used when all the states are not able to be measured and we have a good model of the process.

2. Optimal control

We have used a simple pole-placement technique to introduce state feedback control. We can take this further by providing control design methods to choose the controller state feedback gains in an optimal manner.

3. Kalman filters

These are used to provide an estimate of the states when we have a process model but presume that there is some noise acting on the system states and outputs.

Therefore the promise of excellent design flexibility in state feedback has driven the subject of advanced control forward. The state variable framework and the use of models to design controls is now common in advanced control.

23.6 At the signpost of advanced control

State variable methods provided control theory with a deep and satisfying mathematical basis for the discipline. Of course, there is always a gap between theory and practice, but it is the interplay between them that makes the practical outcomes so interesting and useful. We need the precision of mathematical control tools to illuminate control application problems in industry and commerce. For example, all of the complex systems described in the opening chapter of this book would not have succeeded without the use of design and simulation methods like those we have developed and investigated in the past chapters of this book. We hope you have enjoyed learning about the control system descriptions, classical control design methods like root locus and Bode plots and state variable systems that you have found in this book. Although we have reached the end of a first-level course, it is highly likely that you will be looking at the modules in control engineering available to you next. What sort of topics can you expect?

As we have seen, control engineering uses a model of the system or process either for control design or for predicting control system performance. This basic feature is aided by the availability of computer simulation tools like Simulink, which use our model data, compute system responses, and display the outcomes in a variety of graphical formats. We use models of systems in our design and feasibility studies because of their low cost and wide availability. It is simply cheaper to use software simulations than to develop hardware prototypes or perform extensive trials on production lines. This dependence on models has led to new analysis methods and design procedures which try to overcome the various shortcomings of the model-based approach in control engineering. Some control techniques even incorporate online modelling into the architecture of the controller itself. It is these types of topic that you are likely to meet in your second-level control engineering courses. We should like to give you some brief comments on just three areas that you might meet.

System identification: how to find better system models

It would seem natural that, if we have an engineering approach based on using system models, that we should also investigate how to find these models efficiently and economically. You are quite likely to meet the least-squares method for system identification, where we use process data obtained from the production line or from carefully designed experiments, to compute the parameters and coefficients in our system models. The online version of this routine is known as recursive least-squares, and this method has been used in some online control techniques.

Robust control: dealing with modelling uncertainty

We met gain and phase margin measures in this book. We found that these were simple ways of defining the closed-loop stability margin we have in our control design. This safety margin was needed because our system models might not completely match the real system. Model mismatch and designing control systems to accommodate model discrepancy will be themes that will continue into your second-level course modules. This whole area is commonly termed robust control.

Digital control: where control, computing, signal processing and communications meet

In some chapters we made brief links to the field of digital control. The amazing growth of LANs and the Internet is having a dramatic effect on the way that we do digital control and the way that we control systems, particularly large-scale systems. For example, we recently read of a student who had set up links between a system digital controller and a mobile phone. This was described as a new method of real-time control monitoring! Despite the rapid changes and developments in digital control, your digital control modules are likely to discuss Kalman filtering, discrete-state variable systems and model-based predictive control, among other topics. Again there will be a continuing emphasis on using process and system models in all of these methods.

And your future in control engineering...

As well as looking at second-level course modules we hope you may even be considering possible career in control engineering. Almost all the professional engineering fields, mechanical, electrical, environmental, chemical, aeronautical and marine have control activities of some form or another. In some areas, for example electrical engineering, it is possible to be 'a control engineer'; in others it is always useful to have a good understanding of control topics. In closing, we hope that this book has provided the inspiration to find out more about the useful field of control engineering.

Multiple choice

- M23.1** How are the control or actuator connections to the internal states of the system represented?
- through the \mathbf{A} matrix
 - by diagonalisation of the system matrix
 - through the \mathbf{B} matrix
 - through the system eigenvalues
- M23.2** The 'two-degrees-of-freedom' title for the controller, $u(t) = \mathbf{K}_R r(t) - \mathbf{K}_O y(t)$, refers to the fact that:
- there are two states in the system
 - we use two different values for \mathbf{K}_O
 - there are two dependent controller parameters, \mathbf{K}_R and \mathbf{K}_O
 - there are two independent controller pathways
- M23.3** If we have two inputs, two outputs and two reference inputs:
- there are two states
 - there are four states
 - there are six states
 - the number of states is undefined
- M23.4** For the system $\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u}$, $\mathbf{y} = \mathbf{C}\mathbf{x}$ and the control law $\mathbf{u}(t) = \mathbf{K}_R r(t) - \mathbf{K}_O \mathbf{y}(t)$, the closed-loop system matrix is given by:
- $\mathbf{A} - \mathbf{B}\mathbf{K}_O \mathbf{C}$
 - $\mathbf{A} - \mathbf{B}\mathbf{K}_R \mathbf{C}$
 - $\mathbf{A} + \mathbf{B}\mathbf{K}_O \mathbf{C}$
 - $\mathbf{A} + \mathbf{B}\mathbf{K}_R \mathbf{C}$
- M23.5** A pole-placement design:
- places all the poles on the real axis in the LHP
 - chooses the pole locations to suit the design specification
 - can only be performed with a SISO system
 - can never be achieved with output feedback
- M23.6** The closed-loop pole polynomial is given by:
- $\det(s\mathbf{I} - \mathbf{A}) = 0$
 - $\det(s\mathbf{I} - \mathbf{A})^{-1} = 0$
 - $\det(s\mathbf{I} + \mathbf{A} - \mathbf{B}\mathbf{K}_O \mathbf{C}) = 0$
 - $\det(s\mathbf{I} - \mathbf{A} + \mathbf{B}\mathbf{K}_O \mathbf{C}) = 0$
- M23.7** For a second-order system, tuning the output feedback gain \mathbf{K}_O can:
- fix the values of ζ and ω_n independently
 - fix only the value of ζ
 - fix only the value of ω_n
 - only fix the values of ζ and ω_n dependently

M23.8 State feedback can be achieved by a control law of the form:

- (a) $u(t) = \mathbf{K}_R r(t) - \mathbf{K}x(t)$
- (b) $u(t) = \mathbf{K}_R r(t) - \mathbf{K}y(t)$
- (c) $u(t) = \mathbf{K}(r(t) - y(t))$
- (d) $u(t) = \mathbf{K}_R r(t) - \mathbf{K}_O y(t) - \mathbf{K}x(t)$

M23.9 To stabilise a state feedback system we require:

- (a) all eigenvalues of $\mathbf{A} - \mathbf{B}\mathbf{K}$ have negative real parts
- (b) all eigenvalues of \mathbf{A} have negative real parts
- (c) all eigenvalues of \mathbf{K} have negative real parts
- (d) all eigenvalues of $\mathbf{A} - \mathbf{B}\mathbf{K}_R$ have negative real parts

M23.10 To implement state feedback we need:

- (a) measurements of all the system states
- (b) an internal model of the process
- (c) one of (a) or (b)
- (d) both (a) and (b)

Questions: practical skills

Q23.1 An engineer who is developing a closed-loop control system based on state space methods has modelled a flow process in a chemical works. The flow process has a model:

$$\begin{aligned}\dot{f}(t) &= -0.2f(t) + 6.75f_{\text{in}} \\ f_m(t) &= 1.26f(t)\end{aligned}$$

where $f_m(t)$ represents a measured flow output, and f_{in} is a normalised flow input such that $0 < f_{\text{in}} < 1$. The engineer chooses to design a two-degrees-of-freedom control law using output feedback, $u(t) = k_R f_{\text{ref}} - k_O f_m(t)$, where the reference flow will take values in the range $4.5 < f_{\text{ref}} < 5.5$.

- (a) If the desired closed-loop system eigenvalue is $\lambda(\mathbf{A}_{\text{CL}}) = -1.45$, find the gains k_R and k_O to achieve the design specification.
- (b) Draw a block diagram for the system and control scheme.
- (c) Create a Simulink simulation for the complete control system and show if satisfactory responses are obtained.

Q23.2 A ship autopilot for rudder angle control is required. A simple linear model in state space format has been developed as

$$\begin{aligned}\dot{\phi}(t) &= 0.35\phi(t) + 105.5T_A \\ \phi_m(t) &= 1.05\phi(t)\end{aligned}$$

where $\phi_m(t)$ represents a measured rudder angle output and T_A is a normalised applied torque such that $-1 < T_A < 1$. The control engineer on the project chooses to design a two-degrees-of-freedom control law using state feedback, $u(t) = k_R \phi_{\text{ref}} - k\phi(t)$, where the reference rudder angle will take values in the range $-45^\circ < \phi_{\text{ref}} < 45^\circ$.

- (a) If the desired closed-loop system eigenvalue is $\lambda(\mathbf{A}_{\text{CL}}) = -2.25$, find the gains k_R and k to achieve the design specification.
- (b) Draw a block diagram for the system and control scheme.
- (c) Create a Simulink simulation for the complete control system and show if satisfactory responses are obtained.

Q23.3 An industrial system has a state space model with the following parameters:

$$\begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \end{bmatrix} = \begin{bmatrix} -5.26 & 7.32 \\ -7.32 & -5.26 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} + \begin{bmatrix} 3.6 \\ 2.4 \end{bmatrix} u(t)$$

$$y(t) = [0.36 \quad 0.37] \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix}$$

It is desired to design a two-degrees-of-freedom state feedback control law:

$$\mathbf{u}(t) = K_R r(t) - \mathbf{K} \mathbf{x}(t)$$

- Find the system eigenvalues and the open-loop transfer function for this system.
- Derive the closed-loop formulas for \mathbf{A}_{CL} and K_R .
- Derive the equations for the state feedback matrix \mathbf{K} in element form if pole placement design is to be used.
- Use MATLAB to determine the state feedback gains and the reference controller gain for the design case, which has the desired system eigenvalues $s = -4.5 \pm j3$.
- Develop a Simulink simulation to show the system unit step responses resulting from the designs. Comment on the results found.

Q23.4 In a control system design exercise for a marine engine, an engineer is using the state space framework. The system model is given as

$$\dot{\mathbf{x}}(t) = \mathbf{A} \mathbf{x}(t) + \mathbf{B} u(t)$$

$$\mathbf{y}(t) = \mathbf{C} \mathbf{x}(t)$$

The data for the model are

$$\mathbf{A} = \begin{bmatrix} 13/6 & 4/3 \\ -4/3 & -7/6 \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} 0 \\ 0.5 \end{bmatrix} \quad \text{and} \quad \mathbf{C} = [1 \quad 1]$$

- Determine the system eigenvalues and the stability status of the system.
 - Prove that arbitrary pole placement by state feedback is possible for this system.
- The engineer decides to experiment with a different method of actuation for the system and this leads to a different input matrix:

$$\mathbf{B}_1 = \begin{bmatrix} -1/6 \\ 1/3 \end{bmatrix}$$

- Prove that arbitrary pole placement is not possible with the new actuation system.
- Show that the open-loop system eigenvalue at $s = 1.5$ is unaltered by feedback.
- Comment on the practical implications of this finding.

Problems

P23.1 A modelling exercise leads to a state space model with the following parameters:

$$\begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \end{bmatrix} = \begin{bmatrix} -0.2 & 0 \\ 0 & -0.2 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} + \begin{bmatrix} 1.4 \\ 0.5 \end{bmatrix} u(t)$$

and

$$y(t) = [3.21 \quad 2.31] \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix}$$

Use a two-degrees-of-freedom output controller to design a closed-loop control system. Follow the steps indicated.

- What are the \mathbf{A} , \mathbf{B} , \mathbf{C} data matrices for the system model?
- Find the open-loop transfer function and use simulations, if necessary, to detail the response characteristics of the open-loop system to step inputs.
- Derive the closed-loop system equations using the two-degrees-of-freedom control law, $u(t) = \mathbf{K}_R r(t) - \mathbf{K}_o y(t)$. Find expressions for \mathbf{A}_{CL} and \mathbf{K}_R .
- Use MATLAB to calculate the closed-loop system eigenvalues (poles) for different \mathbf{K}_o values. Calculate the corresponding \mathbf{K}_R values. Comment on the flexibility available in tuning closed-loop system parameters. Comment on your findings.

P23.2 The technical manual gives the state space model for a hydraulic servo system as

$$\begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \end{bmatrix} = \begin{bmatrix} -0.125 & 7.85 \\ -7.85 & -0.125 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} + \begin{bmatrix} 4.0 \\ 3.5 \end{bmatrix} u(t)$$

$$y(t) = [0.85 \quad 0.76] \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix}$$

In an attempt to improve on the existing control scheme, an engineer decides to investigate the use of state feedback control using the two-degrees-of-freedom control law:

$$u(t) = \mathbf{K}_R r(t) - \mathbf{K} \mathbf{x}(t)$$

- Determine the open-loop characteristics of this system model.
- Briefly derive the closed-loop equations to determine expressions for \mathbf{A}_{CL} and \mathbf{K}_R .
- Use the coefficient form of the state feedback matrix \mathbf{K} to determine the pole placement equations for closed-loop design.
- Discuss the flexibility available in the design equations. Use MATLAB to determine the state feedback gains and the reference controller gain for the two design cases:

$$\text{Case (i) Underdamped design} \quad \zeta = 0.5, \omega_n = 1.8$$

$$\text{Case (ii) Critically damped design} \quad \zeta = 1, \omega_n = 1$$

- Develop a Simulink simulation to show the system unit step responses resulting from the designs. Comment on the results found.



Answers to multiple choice questions

Chapter 2

M2.1 (b) M2.2 (a) M2.3 (d) M2.4 (b) M2.5 (a)
M2.6 (b) M2.7 (b) M2.8 (c) M2.9 (a) M2.10 (b)

Chapter 3

M3.1 (b) M3.2 (d) M3.3 (d) M3.4 (d) M3.5 (c)
M3.6 (c) M3.7 (b) M3.8 (a) M3.9(a) M3.10 (a)

Chapter 4

M4.1 (a) M4.2 (b) M4.3 (a) M4.4 (c) M4.5 (a)
M4.6 (c) M4.7 (d) M4.8 (b) M4.9(b) M4.10 (d)

Chapter 5

M5.1 (c) M5.2 (b) M5.3 (d) M5.4 (c) M5.5 (c)
M5.6 (b) M5.7 (a) M5.8 (a) M5.9 (b) M5.10 (d)

Chapter 6

M6.1 (c) M6.2 (a) M6.3 (c) M6.4 (d) M6.5 (a)
M6.6 (c) M6.7 (c) M6.8 (a) M6.9(b) M6.10 (c)

Chapter 7

M7.1 (a) M7.2 (c) M7.3 (d) M7.4 (a) M7.5 (b)
M7.6 (d) M7.7 (c) M7.8 (b) M7.9 (c) M7.10 (c)

Chapter 8

M8.1 (c) M8.2 (d) M8.3 (a) M8.4 (c) M8.5 (a)
M8.6 (a) M8.7 (b) M8.8 (d) M8.9 (a) M8.10 (a)

Chapter 9

M9.1 (c) M9.2 (d) M9.3 (a) M9.4 (a) M9.5 (d)
M9.6 (c) M9.7 (a) M9.8 (c) M9.9 (a) M9.10 (b)

Chapter 10

M10.1 (d) M10.2 (a) M10.3 (c) M10.4 (d) M10.5 (a)
M10.6 (c) M10.7 (a) M10.8 (a) M10.9 (c) M10.10 (c)

Chapter 11

M11.1 (b) M11.2 (a) M11.3 (a) M11.4 (b) M11.5 (a)
M11.6 (c) M11.7 (c) M11.8 (b) M11.9 (c) M11.10 (d)

Chapter 12

M12.1 (b) M12.2 (c) M12.3 (d) M12.4 (a) M12.5 (c)
M12.6 (d) M12.7 (d) M12.8 (b) M12.9 (d) M12.10 (b)

Chapter 13

M13.1 (a) M13.2 (a) M13.3 (c) M13.4 (d) M13.5 (a)
M13.6 (b) M13.7 (c) M13.8 (d) M13.9 (d) M13.10 (a)

Chapter 14

M14.1 (a) M14.2 (d) M14.3 (b) M14.4 (d) M14.5 (c)
M14.6 (d) M14.7 (d) M14.8 (d) M14.9 (c) M14.10 (b)

Chapter 15

M15.1 (d) M15.2 (d) M15.3 (c) M15.4 (a) M15.5 (b)
M15.6 (a) M15.7 (a) M15.8 (c) M15.9 (a) M15.10 (a)

Chapter 16

M16.1 (a) M16.2 (a) M16.3 (b) M16.4 (c) M16.5 (b)
M16.6 (b) M16.7 (a) M16.8 (b) M16.9 (d) M16.10 (d)

Chapter 17

M17.1 (a) M17.2 (c) M17.3 (c) M17.4 (d) M17.5 (c)
M17.6 (a) M17.7 (a) M17.8 (a) M17.9 (a) M17.10 (b)

Chapter 18

M18.1 (a) M18.2 (d) M18.3 (b) M18.4 (d) M18.5 (a)
M18.6 (d) M18.7 (c) M18.8 (d) M18.9 (d) M18.10 (a)

Chapter 19

M19.1 (d) M19.2 (c) M19.3 (b) M19.4 (b) M19.5 (a)
M19.6 (d) M19.7 (b) M19.8 (d) M19.9 (c) M19.10 (b)

Chapter 20

M20.1 (c) M20.2 (b) M20.3 (d) M20.4 (d) M20.5 (b)
M20.6 (c) M20.7 (c) M20.8 (c) M20.9 (d) M20.10 (a)

Chapter 21

M21.1 (b) M21.2 (d) M21.3 (b) M21.4 (b) M21.5 (c)
M21.6 (a) M21.7 (c) M21.8 (c) M21.9 (a) M21.10 (c)

Chapter 22

M22.1 (b) M22.2 (c) M22.3 (c) M22.4 (d) M22.5 (a)
M22.6 (a) M22.7 (a) M22.8 (c) M22.9 (c) M22.10 (a)

Chapter 23

M23.1 (c) M23.2 (d) M23.3 (d) M23.4 (a) M23.5 (b)
M23.6 (d) M23.7 (d) M23.8 (a) M23.9 (a) M23.10 (c)

Answers to selected questions

Chapter 2

Q2.1 (3.61, 56.3°), (7.21, 56.3°), (1.12, 26.6°), (7, -90°)

Q2.2 5.20 +j3, 1.29 -j1.53, 2.06 + j1.41, 0.955 -j0.296

Q2.3 6∠-5°, 24∠50°, 60∠-60°

Q2.4 (i) 0.707, 45°; (ii) 0.464, -35.5°

Q2.5 (i) $-1 \pm j\sqrt{2}$, $-2 \pm j\sqrt{2}$, $0.167 \pm j0.799$; (ii) $(s+1)^2 + 1.414^2$, $(s+2)^2 + 1.414^2$, $(s-0.165)^2 + 0.802^2$

Q2.6 $6/(s+1)$, $2/s^2$, $(7s^2 + 4s + 8)/(s^2(s+2))$, $5/(s^2 + 25)$, $-s/(s^2 + 4)$

Q2.7 $3e^{-2t}$, $6e^{3t}$, $28 \sin 4t$, $0.33(1-e^{-3t})$, $3t$

Q2.8 0, 2

P2.1 (a) $x(t) = 0.5e^{-0.5t}$; (b) $x(t) = 1 - e^{-0.5t}$

P2.2 (a) $U(s) = 2/s$, $V(s) = 1/s$; (b) $X(s) = -7/s$, $Y(s) = 7/s$; (c) $p = 0$

P2.3 (a) Poles of $X_1(s)$: $p_1 = -1$, $p_2 = -2$, Poles of $X_2(s)$: $p_1, p_2 = -1 \pm j$;

(b) $x_{1ss} = 0$, $x_{2ss} = 0$

P2.4 $X(j\omega) = 1/(3 + j\omega)$

Chapter 3

Q3.2 (a) -2.2056, $0.1028 \pm 0.6655j$; (b) -1, $0.5 \pm 0.866j$; (c) -2, 0.3333

Q3.3 (a) $g_a = (4/3) * (s+3)/(s+4)$; (b) $g_b = 12.5 * (s+1)/((s+1-j*0.5) * (s+1+j*0.5))$;

(c) $g_c = (s+6)/s^2/(s+4)$

Chapter 4

Q4.2 (b) $K = 0.1$

Q4.3 (a) $H(s) = 1/(11.5s + 1)Q(s)$

Q4.4 (b) about 3000 sec (50 minutes)

P4.4 (e) Approx: $K_p = 0.1$

Chapter 5

Q5.1 (a) 2 volts/litre; (b) 1.33 mA/bar; (c) 40 $\mu V/^\circ C$

Q5.2 (a) $Y(s) = 10/(4s + 3)Q(s)$; (b) $M(s) = 2/(6s + 1)P(s) + (18/6s + 1)$;

(c) $Y(s) = (K/(\tau s + 1))U(s) + (\tau/\tau s + 1)y_0$

Q5.3 (a) $Y(s) = (K/(Ms^2 + Bs + K_s))F(s)$; (b) $\theta(s) = \frac{K}{(Js^2 + Bs + K_s)} T(s) = \frac{-2J}{(Js^2 + Bs + K_s)}$

P5.2 $M \frac{d^2x}{dt^2} + B \frac{dx}{dt} + K_s x(t) = M \frac{d^2z}{dt^2}$

P5.3 (b) $K_A = 300/28 = 10.7143$

Chapter 6

Q6.1 (a) $K = 0.3$, $\tau = 0.1$; (b) $K = 6$, $\tau = 1$; (c) $K = 0.667$, $\tau = 0.33$

Q6.2 (a) 30; (b) 2.25; (c) 10

Q6.3 (a) $(1 - s)/(1 + s)$; (b) $(1 - 5s)/(1 + 5s)$

P6.1 $0.3(dx/dt) + x = 2u(t)$, Input $u(t)$: °C, Output $x(t)$: mm

P6.2 $\tau = RC$, $R = 6 \text{ k}\Omega$

Chapter 7

Q7.1 (a) underdamped, $\omega_d = 0.4975 \text{ rad/s}$; (b) overdamped, no damped natural frequency; (c) underdamped, $\omega_d = 0.9539 \text{ rad/s}$

Q7.2 $G_1(s) = (3/7)/(s^2 + (4/7)s + 9/7)$, $G_2(s) = 2/(s^2 + 1.5s + 0.5)$,
 $G_3(s) = 2/(s^2 + 0.67s + 12)$

Q7.3 (a) Unity s^2 coefficient: $\omega_n = 3$, $\zeta = 0.33$, $K = 0.33$; (b) unity constant coefficient: $K = 4$, $\omega_n = 0.707$, $\zeta = 0.707$; (c) unity constant coefficient: $K = 100$, $\omega_n = 0.1$, $\zeta = 0.5$

Q7.4 (a) $K = 6$, $\omega_n = 1/6$, $\zeta = 0.5$, underdamped, two complex roots; (b) $K = 14$, $\omega_n = 1$, $\zeta = 1$, critically damped, two equal roots; (c) $K = 2$, $\omega_n = 1/\sqrt{6}$, $\zeta = 4/\sqrt{6}$, overdamped, two real roots; (d) $K = 15$, $\omega_n = 2$, $\zeta = 2$, overdamped, two real roots

Q7.5 Plot A and plot C are underdamped with complex roots of characteristic equation.

P7.1 (a) mm/bar; (b) $G(s) = 2.36/(0.25s^2 + 0.7s + 1)$

P7.2 (a) $K = 20$, $\omega_n = 31.6$, $\zeta = 9.49$; (b) system very overdamped; (c) roots will lie on negative real axis

P7.4 (a) $\Omega(s) = 0.15/[(0.3s + 1)(4s + 3)]U(s)$; (b) $i_{ss} = 2$, $t_{qss} = 0.2$, $\omega_{ss} = 0.05$

Chapter 8

Q8.2 (a) 7.68, 1.0, 0.8848; (b) 16.0, 4.0, 0.2462; (c) $K_1K_2K_3$, H_1 , $(K_1K_2K_3)/(1 + K_1K_2K_3H_1)$

Q8.3 (a) 1.2, 1, $Y/R = 0.5455$, $Y/d = 0.1818$; (b) 0.48, 3.2, $Y/R = 0.1893$, $Y/d = 0.1577$;
(c) 1.2, 6.0, $Y/R = 0.1463$, $Y/d = 0.1707$; (d) GK , H , $Y/R = GK/(1 + GKH)$,
 $Y/d = GF/(1 + GKH)$

Q8.4 (a) $G_{CL}(s) = 30/(3s + 31)$, $G_E(s) = (3s + 1)/(3s + 31)$;

(b) $G_{CL}(s) = (10s + 5)/(3s^2 + 11s + 5)$, $G_E(s) = (3s^2 + 1)/(3s^2 + 11s + 5)$

Q8.5 (a) $G_{CL}(s) = 80/(2s + 81)$, $G_E(s) = (2s + 1)/(2s + 81)$, $G_d(s) = (2s + 1)/(2s + 81)$;

(b) $G_{CL}(s) = 3/(2s^2 + 4s + 4)$, $G_E(s) = (2s^2 + 4s + 1)/(2s^2 + 4s + 4)$,

$G_d(s) = 2(s^2 + 4s + 1)/[(s + 1)(2s^2 + 4s + 4)]$

P8.2 $c = 0.23$

P8.4 $K = 0.01$: overdamped, $K = 1$: underdamped, oscillatory, overshoot

P8.5 overdamped: $K < 16/30$, underdamped: $K > 16/30$

Chapter 9

Q9.1 (a) 30; (b) 2.25; (c) unstable system, no steady state value

Q9.2 (a) 24; (b) 13.33; (c) unstable, no steady state value

Q9.3 Plot A: (a) 38%; (b) $\zeta = 0.2944$; (c) $t_r(10\%,90\%) = 0.9$ seconds;

(d) $t_s(5\%) = 5.2$ seconds; (e) $\omega_d = 1.904 \text{ rad/s}$; (f) $\omega_n = 1.9923 \text{ rad/s}$,

$G_A(s) = 3.9693/(s^2 + 1.1731s + 3.9693)$

Q9.4 System 1: $D_{peak} = 1.8$ volts, $D_{ts}(2\%) = 17.5$ seconds, $\omega_d = 0.6981 \text{ rad/s}$

P9.1 1st order, $c = 3$

P9.2 $K = 0.052$

Chapter 10

Q10.1 (a) $s = -3$, slow exponential decay; (b) $s = \pm j0.125$, zero at $s = 0$, constant oscillation; (c) $s = 0.1$, slow exponential growth; (d) $s = 10.6$, fast exponential growth;

(e) $s = -0.222$, slow exponential decay; (f) $s = -100$, fast exponential decay; (g) $s = \pm 8$, fast exponential growth; (h) $s = 4$, $s = \pm j8$, zeros at $s = -6$, $s = -4$: poles provide fast increasing sinusoidal growth

Q10.3 (a) $\omega_n = 0.707, K = 1, \zeta = 1.0605$; (b) $\omega_n = 1.25, K = 1, \zeta = 0.6$; (c) $\omega_n = 1, K = 1, \zeta = 0.4$

Q10.4 (a) $s = -1/4$; (b) $s = -3/2$; (c) $s = 4$; (d) $-1.5 \pm j2.6$; (e) $1.5 \pm j1.3$; (f) $-1, -1$
 (a) stable (b) stable (c) unstable (d) stable (e) unstable (f) stable

Q10.5 (a) *A* unstable; *B, C* and *D* all stable. (b) *D*

Q10.6 (a) *A* unstable, *B* unstable, *C* and *D* stable. (b) *A* unstable; *B* oscillating, no decay; *C* oscillatory, decays to a steady state value; *D* overdamped, settles at a steady state value

P10.1 Th_3 is the fastest. $\tau_3 = 1/3$

P10.2 $K \leq 2/3$

P10.4 (a) $z = -0.5, p_1 = -2$ and $p_2 = -0.0833$; (b) $z_1 = -0.5, z_2 = -0.1, p_1 = -2, p_2 = -0.0833, p_3 = 0$; (c) $p_{CL1} = -4.9891, p_{CL2} = -0.3247, p_{CL3} = -0.1029, z_1 = -0.5, z_2 = -0.1$

Chapter 11

Q11.1 (a) $G_{CL}(s) = \frac{32(K_p + K_d s)}{s^2 + (2.4 + 32K_d)s + 16 + 32K_p}, S(s) = \frac{(s^2 + 2.4s + 16)}{s^2 + (2.4 + 32K_d)s + 16 + 32K_p}$

Q11.2 (a) $3/(1 + 6K_p)$; (b) $3/(1 + 6K_p)$; (c) 0

Q11.3 (a) $e_{ss} = 1/(3K_p)$; (b) $e_{ss} = 0$

Q11.4 $K(s) = 0.75 + (2/s)$

Q11.5 $K(s) = (6.3 + 1.56s)$

P11.1 $K_p = 0.167$

P11.2 $K(s) = 0.44 + 0.46s$

P11.3 $K_1 = 0.16, K_2 = 0.012, K_p = 0.16, K_d = 0.012$

Chapter 12

Q12.4 (a) $K = 8, \tau = 26$ minutes; (b) $e_{ss} = 0.303$

Q12.5 (b) $K = 0.8, \tau = 10$ minutes; (d) manual controller: $K_C(s) = 3.5 + (0.4/s)$

P12.2 (a) $K = 0.25, \tau = 5$ minutes, $K_p = 8, K_i = 1.8$

P12.3 $K_p = 1.25, K_d = 4$

Chapter 13

Q13.1 (a) $p_{1,2} = -4 \pm 2j$, finite zeros: $z = -3, n_{z\infty} = 1$; (b) $p_1 = -4, p_2 = -5$, no finite zeros, infinite zeros: $n_{z\infty} = 2$; (c) $p_1 = 0, p_2 = -0.33, p_3 = -0.2$, no finite zeros, infinite zeros: $n_{z\infty} = 3$

Q13.2 $G(s) = K[(s + 1)(s - 1)(s + 0.5)]/[s(s + 3)(s + 5)]$

Q13.3 $G(s) = K/s^3$

Q13.4 (a) 2; (b) the plot is symmetric with respect to the real axis; (c) $p_1 = 0$ and $p_2 = -4, z = -8$; (d) the root locus starts at the poles and ends at infinity and at the zero at $s = -8$; (e) the locus exists only to the right of $z_1 = -8$ and to the right of $p_2 = -4$

Q13.5 $K = 0.1$

Q13.6 (a) $G_{\text{fict}}(s) = 3s/(4s^2 + 21)$; (b) $G_{\text{fict}}(s) = 2/(s^2 + s)$

P13.1 (c) $K = 0.8$

P13.2 (a) $K = 47.6$; (b) 2.82 rad/s; (c) for $K = 8, \zeta \sim 0.5$

Chapter 14

Q14.1 (a) $\omega = 3$ rad/s, mag = 5, phase = +0.2 rads; (b) $\omega = 1$ rad/s, mag = 3, phase = -0.3 rads; (c) $\omega = 6$ rad/s, mag = 10, phase = 0 rads; (d) $\omega = 1$ rad/s, mag = 20, phase = -1.2 rads

Q14.1 (a) 29.54 dB; (b) 9.54 dB; (c) -7.96 dB

Q14.3 (a) 2.25; (b) 10; (c) 1.20

Q14.4 (a) infinite; (b) $0 < \omega < 0.2$ rad/s; (c) $\omega = 2$ rad/s; (d) -20 dB/decade; (e) 20 dB and 0 dB

Q14.5 $\omega_{gco} = 1.28$ rad/s $\omega_{pco} = 1.05$ rad/s, GM ~ 36 dB, PM $\sim -45^\circ$, unstable

P14.1 Gain = 0.2, Phase = -89°

P14.2 $u(t) = 1.58 \cos(10t - 7\pi/6)$

P14.3 The system is stable

P14.4 PM about 70° , GM about 24 dB. Closed loop system will be stable

P14.5 $\zeta = 0.25$

P14.6 The closed loop system will be stable

Chapter 15

Q15.2 (a) $\omega_1 = 0.3$ rad/s, $\omega_2 = 0.5$ rad/s, $\omega_3 = 2$ rad/s, d.c. gain = 13.33; (b) $\omega_1 = 0.1$ rad/s, $\omega_n = 1$ rad/s, $\omega_3 = 4$ rad/s, d.c. gain $\rightarrow \infty$

Q15.3 $K = 10$, $\tau = 10$ seconds

Q15.6 $G_{OL}(s) = g(s)[g_1(s) + g_2(s)] = g(s)g_1(s) + g(s)g_2(s)$

P15.2 $|G(j\omega)| = 10/\sqrt{\omega^2 + 1}$ $\angle G(j\omega) = -\tan^{-1}(\omega)$. PM = 96° , GM = ∞

P15.3 $K = 0.18$ gives PM = 40°

P15.4 $K = 4.6$

Chapter 16

Q16.3 $k = 0.1$, PM = 78° (stable), $k = 1$, PM = 23° (stable), $k = 2$, PM = 0° (critically stable), $k = 10$, PM = -37° (unstable). As k increases the PM decreases

Q16.4 (b) As PM decreases, overshoot increases. As gain increases, PM decreases

Q16.5 PM increases to 75° , GM increases to ∞

Q16.6 $e_{ss} = 0.05$, OS(%) = 2%

Q16.7 $\tau = 2$, $\alpha = 0.1$, $K_1(s)$ is a lead controller. $\tau = 2$, $\beta = 10$, $K_2(s)$ is a lag controller

P16.2 Lead compensator does not meet requirements

P16.3 For example, $G_{lag}(s) = (8.6s + 1)/(40.9s + 1)$

P16.4 For example, $G_{lead}(s) = (0.09864s + 1)/(0.05401s + 1)$

Chapter 17

Q17.1 PM = 135° , GM = ∞

Q17.2 $K = 6.3$

Q17.3 $G(s)$: 78.5° , $H(s)$: -75.8°

Q17.4 $M_p = 0.6$ dB

Q17.5 $\omega_{bw} = 0.23$ rad/s, $M_p = 3$ dB

P17.1 $\omega_{bw} = 0.22$ rad/s, $M_p = 3$ dB

P17.2 $M_p = 3$ dB, PM = 40° and GM = ∞

P17.3 (a) $20 \log_{10}(S(j\omega)) = 20 \log_{10}(G_{CL}(j\omega)) - 20 \log_{10}(G(j\omega))$; (c) 0.15 to 1 rad/s

Chapter 18

Q18.1 (a) $K_p = K_p$, $K_i = K_p/T_i$, $K_d = K_p T_d$; (b) $G_P(s) \neq G_Q(s)$

Q18.2 $K_p = 0.2083 \times 10^5$

Q18.4 (a) $G_{PID}(s) = 5.054 + (0.455/s) + 2.688s$

Chapter 19

Q19.1 (a) GM_{dB} = 4.1 dB; (d) $K = 1.596$

Q19.2 (b) $K_p = 4.39$, $T_i = 5.41$

Q19.4 (a) $K_u = 33.690$, $K_p = 16.605$, $T_i = 3.600$

P19.1 $G(s) = 17e^{-5s}/(13s + 1)$, $G_{PI}(s) = 0.24 + (0.016/s)$

P19.2 (a) $K_u = 35$, $P_u = 4.15$; (b) $K_p = 15.75$, $T_i = 3.46$

P19.3 (a) $M = 20$, $A_{osc} = 0.28$, $K_u = 90.99$, $P_u = 3.32$; (b) PID, no overshoot: $K_p = 18.20$, $T_i = 1.10$, $T_d = 1.66$

Chapter 20

Q20.1 $\mathbf{A} = [0 \ -4.27; -2.21 \ -3.96]$, $\mathbf{B} = [0; 3.04]$, $\mathbf{C} = [1 \ 0.56]$, $\mathbf{D} = [1]$

Q20.2 $\mathbf{A} = [\delta_{C1}/c_P \ v_T/c_P; \eta_T/c_T \ \delta_{C2}/c_T]$, $\mathbf{B} = [4.03\delta_{C1}/\omega_P c_P \ 0.4\eta_T/c_T; \ 0 \ 3.04/c_T]$, $\mathbf{C} = [1 \ \eta_T]$,

$$\mathbf{D} = [0 \ 0]$$

Q20.4 (a) $\mathbf{A} = [0 \ 1 \ 0; 0 \ 0 \ 1; -5 \ -9 \ -5]$; $\mathbf{B} = [0; 0; 2]$; $\mathbf{C} = [1 \ 0 \ 1]$; $\mathbf{D} = [0]$;

$$(b) Y(s) = 2(s^2 + 1)/(s^3 + 5s^2 + 9s + 5) \times U(s)$$

Q20.5 $\mathbf{A} = [0 \ 1 \ 0; 0 \ 0 \ 1; -1.875 \ -5.75 \ -4.5]$; $\mathbf{B} = [0; 0; 1]$; $\mathbf{C} = [1 \ 3.5 \ 0]$; $\mathbf{D} = [0]$

P20.1 (a) Gain = 2.375, $\tau = 0.5$; (c) $A = -2$, $B = 1$, $C = 4.65$, $D = 0$

P20.2

$$\mathbf{A} = \begin{bmatrix} \frac{4.25(1-2\eta) - \eta\alpha_T + \eta\delta}{4.25 - \alpha_T\eta} & -0.75\eta \\ \frac{2\eta\alpha_T - \delta}{4.25 - \alpha_T\eta} & \frac{3\eta\alpha_T - 12}{4.25 - \alpha_T\eta} \end{bmatrix} \quad \mathbf{B} = \begin{bmatrix} \frac{3.995 - 4.75\eta}{4.25 - \alpha_T\eta} \\ \frac{0.94\alpha_T + 4.75}{4.25 - \alpha_T\eta} \end{bmatrix}$$

$$\mathbf{D} = 0 \text{ and } \mathbf{C} = [1 \ 0]$$

P20.3 (a) $\mathbf{A} = [0 \ 1; 0 \ -K^2/JR]$, $\mathbf{B} = [0; K/JR]$, $\mathbf{C} = [1 \ 0]$, $\mathbf{D} = [0]$; (c) $\mathbf{C} = [1 \ 0, 0 \ 1]$,

$$\mathbf{D} = [0; 0]; (d) Y_1(s) = 1.35/(s^2 - 0.054s), Y_2(s) = 1.35s/(s^2 - 0.054s)$$

Chapter 21

Q21.4 510 °C

Chapter 22

Q22.1 (a) $\lambda_{1,2} = -2.005 \pm j1.490$, stable; (b) $\lambda_{1,2} = \pm j1.99$, unstable;

$$(c) \lambda_{1,2} = +2.000 \pm j1.005, \text{ unstable}$$

Q22.2 (a) $\lambda_{1,2} = -1.4 \pm 2.6j$, $\lambda_3 = -2.5$, stable; (b) $\lambda_1 = 1.2469$, $\lambda_2 = -2.5334$,

$$\lambda_3 = -4.0236, \text{ unstable}$$

Q22.3 (a) $\lambda_1 = -1.5$, $\lambda_2 = -0.5$; (b) as (a); (c) $G(s) = (0.7s - 0.2)/(s^2 + 2s + 0.75)$;

$$(d) p_1 = -0.5, p_2 = -1.5$$

P22.2 (a) $T(0) = 0$ °C, $T(5 \text{ hours}) = 1102.5$ °C; (b) 168.5°C

Chapter 23

Q23.1 (a) $k_o = 0.147$, $k_R = 0.2148$

Q23.2 (a) $k = 0.0246$, $k_r = 0.0203$

Q23.3 (a) $\lambda_{1,2} = -5.26 \pm j7.32$, $G(s) = (2.184s + 8.0621)/(s^2 + 10.52s + 81.25)$;

$$(d) K = [-1.0630 \ 0.9612], K_R = 3.6281$$

Q23.4 (a) (i) $\lambda_1 = 1.5$, $\lambda_2 = -0.5$, system is open loop unstable, (ii) $\det(\mathbf{X}) = -0.333 \neq 0$. Thus arbitrary pole placement is possible; (b) $\det(\mathbf{X}) = 0$, \mathbf{X} is no longer invertible

P23.1 (a) $G(s) = (5.899s + 1.1798)/(s + 0.2)^2$; (d) $K_o = 1$, $\lambda(A_{CL}) = -6.1, -0.2$, $K_o = 2$, $\lambda(A_{CL}) = -11.9, -0.2$

P23.2 (d) (i) $K = [-0.7053 \ 1.2489]$, $K_R = 13.1041$, (ii) $K = [-0.7127 \ 1.3145]$, $K_R = 4.0445$

Index

- A/D interface 565
- ABCD state model
 - block diagram 629
 - state variable notation summary 628
- accelerometer 123, **128**
- actuator 108
- actuator saturation 538
 - nonlinear model 662
- actuator-process-transducer structure 106, 109
- aliasing 566
- amplification 398
- anti wind-up circuit 539, 541
- anti-alias filters 566
- attenuation 398
- autopilot 8, 10
 - control specification 317
- autotune 578
 - process reaction curve method 578
 - technology 578
- auxiliary equation 184

- bandwidth 406
 - open and closed loop 408
 - signal 407
- block diagram **103**
 - block
 - absolute value 542
 - multiplier 541
 - feedback path 208
 - forward path 208
 - manipulation 204
 - open loop 208
 - push-through rule 207
 - shorthand formula 208
 - signal equations 205
 - system block 103
- Bode plot 393, 426, 428
 - adding a pole at the origin 457
 - adding a real pole 458
 - adding a real zero 460
 - effect of damping ratio 454
 - finding the transfer function 446
 - gain and first order lag 445
 - identification of second order transfer function 455
 - s-term and first order lag 448
 - two first order lags 449
- boiler system 556
- bounded signals 264, 268
 - table 266

- cascaded system 105, 117, 456, 484
- causality 43
- characteristic equation
 - complex roots 185
 - real roots 184
- characteristic polynomial 281
- chemical system, state variables 622
- closed loop 197
 - control signals 203
 - control system 202
- closed loop bandwidth **509**
- closed loop control, block diagram analysis 280
- closed loop frequency response **461**
 - Nichols 513
- closed loop peak magnitude, M_p 509
- closed loop polynomial 354
- closed loop stability 281, 292
- closed loop system, PD control analysis 301
- commercial controller units 578
- comparator 201
- compensator, effect of 485
- complementary sensitivity function 461
- complex numbers 17
 - addition and subtraction 20
 - cartesian form 17
 - complex exponential form 17, 18
 - division 21
 - magnitude, gain and phase 22
 - modulus 18
 - multiplication 20
 - parameter (frequency) dependence 24
 - polar angle 18, 22
 - polar form 17
 - rectangular form 17
- contours
 - constant damping 250, 363
 - natural frequency 250, 363
- control improvements
 - natural resource usage 2
 - quality control 1
 - quality of life 2
- control matrix, **B** 712
- controller structure 473
 - selection of PID terms 313
- conveying system
 - actuator 139
 - complete block diagram 142
 - process 137
 - Simulink model 142
 - state variable model 640
 - transducer 138
- co-ordinate measuring machines, description 8–10
- corner frequency 441, 454
- critically damped 178, 252
- cut-off rate 400

- d.c. gain 399, 582
- d.c. motor **139**
- D/A interface 565
- damped oscillation method 601
 - advantages/disadvantages, table 606
 - procedure 600
 - rule-base 600
- damped oscillation, ω_d 134
- damper element 124, 174
- damping 126
- damping factor 127
- damping ratio, ζ 127, **176, 178**, 419, 456,
 - effect on Bode plot 454
- data outliers 585
- DCS system 530

- deadtime 165
 - Padé approximation 168
 - temperature model 165
- decibel 387
- delay time 582
- denominator, closed loop 281
- denominator polynomial 28, 243
- derivative, bandwidth limited 539
- derivative control 278, **299**
 - disturbance rejection 305
 - Laplace domain formula 299
 - noise amplification 545
 - reference tracking 302
 - relationship to damping ratio 302
 - time domain formula 299
- derivative kick 335, 539, 550
- derivative term
 - Bode plot 545
 - digital form 569
 - filter 539
 - incremental form 572
 - modified Bode plot 547
- derivative time constant 531, 579
- design pole polynomial 716
- design specifications 410, **470**, 220
- diagonalisation of a matrix 689
- differentiation, numerical approximation, backward form 567
- differentiator 278
- digital control 734
 - algorithm 565
 - PID control 565
 - recurrent relationship 565
- digital filters 566
- digital PID control
 - incremental form 572
 - positional algorithms 572
 - summary table 570
 - velocity form 572
 - summary table 573
- direct feedthrough matrix 628
- disturbance 198, 203, 279
 - input and output 204
- disturbance rejection 233, 279, 281
 - Laplace analysis 288
- disturbance rejection performance 296
 - derivative control 305
- driving matrix, **B** 712
- dynamics 148
- eigenvalue 688
 - determinant formula 689
- eigenvalue–eigenvector formula 689
- eigenvector 688
- electrical circuit, state variable model 632
- electrical system, state variables 622
- electromagnetic coil 122, 129
- electro-mechanical system model, state variable model 717
- error transfer function 221
- exothermic reactor system 556
- feedback 413
- feedback control system 200
- Final value theorem 221, 223
 - application 315
 - definition 45
- first order differential equation **150, 156**
- first order lag plus deadtime response 582, 584
- first order terms, magnitude and phase 440
- flight control systems description 6–8
- fluid flow system 110
- forced response 157
 - state-space 635, 639
- Fourier analysis 384
- free response 157
 - state-space 635, 638
- frequency, damped oscillation, ω_d 134
- frequency axis 390, 428
- frequency content of a signal 47
- frequency ranges 433
- frequency response **386, 397**, 426, 430
 - amplification 398
 - attenuation 398
 - MATLAB plot 430
 - roll-off rate 400
- frequency response plot, interpretation 409
- frequency scales 390
- functional specification 116
- gain, constant 104
- gain calculation 428
- gain crossover frequency 405
- gain margin **414**
 - Bode 415
 - Nichols 416
 - Nyquist 417
- gain terms, magnitude and phase 437
- gas turbine system control specification 318
- general polynomial form 181
- hertz 390
- high-frequency asymptote 400, 440
- hot strip rolling mills, description 11–13
- ideal gas law, nonlinear equation 662
- industrial heaters, description 14–15
- infinite zeros 354
- initial value theorem, definition 46
- input disturbances 203
- input matrix, **B** 628, 712
- integral control 278, **290**
 - closed loop system analysis 291
 - first order system analysis 291
 - Laplace domain formula 290
 - summary table 299
 - time domain formula 290
- integral controller, phase lag 486
- integral term
 - digital form 569
 - incremental form 572
- integral time constant 531, 579
- integral wind-up 539
- integrating factor 156
- integrating process 40
- integration, numerical approximation 568
- integrator, 1/s 278
- inverse response system 368, 557
- inverted pendulum system, state variable model 695
- Kalman filters 732
- lag term 444
- Laplace transform models 110
- Laplace transforms 16, **25**
 - 1/s, integrator 37
 - decaying exponential cosine signal 33
 - decaying exponential signal 26, 31
 - decaying exponential trigonometric signal 33
 - definition 25
 - derivations, first principles 30
 - derivative of a signal 37

- differential equation 38
- Final value theorem, example 45
- first derivative of a signal 37
- growing exponential trigonometric signal 33
- initial value theorem, definition 46
- integration of a signal 37
- inverse Laplace transform 28
- Laplace transform tables 29
- Laplace variable, s 26
- left half plane (LHP) 26
- partial fractions 44
- properties, linear combination 34
- properties, signal by a constant 34
- properties, table of 35
- ramp signal 31
- right half plane(RHP) 26
- s , differentiator 37
- second derivative of a signal 37
- sine signal 32
- s -plane 26, 27
- step signal 30
- superposition theorem 42
- lead term 444
- left half plane (LHP) 244
- limit cycle 607
- linear relationship 660
- linear system 41, 660
- linearisation 668
 - actual process values 671
 - general nonlinear state variable model 675
 - output equations 669
 - procedure 668
 - system dynamics 668
- liquid level model
 - actuator 117
 - complete block diagram 120
 - performance specification 116
 - process 112
 - Simulink model 121
 - transducer 114
- liquid level system
 - nonlinear model 662
 - state variable model 636
- load disturbance 279, 581
- logarithmic frequency 390
- low frequency asymptote 440
- low pass filter 546

- magnetic suspension model 666
- magnitude of sinusoidal signal 386
- magnitude axis 428
- magnitude plot 48, 393
- manipulated variable 104
- manufacturing systems 135
- mass and spring system model, state variable model 699
- matrices
 - determinant 687
 - diagonal eigenvalue matrix 688
 - inverse 686
 - inverse (2×2), formula 687
 - inverse ($n \times n$) formula 687
 - MATLAB eig command 689
 - MATLAB inv command 687
 - multiplication 684
 - compatibility 685
 - postmultiplication 687
 - premultiplication 686
 - unity matrix 686
- maximum peak value, y_{\max} 228
- M-contours 508
- mechanical system, state variables 622
- MIMO 104
- model
 - conveying system 137
 - earthquake disturbance on building 190
 - fan pressure system 601, 610
 - feeder tank simulation 254
 - first order with time delay 341
 - industrial furnace 279
 - liquid level 112
 - motor 332
 - RC circuit 159
 - shaker table 124
 - trailer suspension 174
- model parameters K and τ 158
- modelling 103, 109
- modified derivative term 546
- multivariable system 624

- natural frequency, ω_n 176, 179, 232
- N-contours 509
- negative feedback 200
- negative gain 557
- negative step 154
- Nichols chart 395, 505
- Nichols chart design
 - bandwidth specification 523
 - gain margin specification 517, 520
 - M_p specification 522
 - phase margin specification 518, 521
- Nichols plot 394, 507
- nominal linear model 664
- non-interacting form, PID 554
- nonlinear model, magnetic suspension 666
- nonlinear system 41, 154, 538, 660, 662
- non-minimum phase 369
- numerator polynomial 27, 243
- numerical approximations 567
- Nyquist plot 395

- observers 732
- on-off control 578
- on-off relay 605, 607
- open loop system 197
- operating points 664
- optical encoder 136
- optimal control 732
- oscillation 174
- oscillatory system 251
- output disturbances 203
- output feedback control 713
 - closed loop analysis 714
 - closed loop pole polynomial 716
 - closed loop system matrix 715
 - reference gain matrix 715
 - two-degrees-of-freedom output control law 713, 722
- output matrix, C 628, 712
- overdamped 178, 253

- Padé approximation 168
- parallel form, PID control 531
- parameter (frequency) dependent complex numbers 24
- parameter Root Locus 370
- partial fractions 44
- PD control
 - P on error, D on measured variable 335
 - procedure for type 1 model 336
 - textbook form 333
- peak disturbance, D_{peak} 234
- percent overshoot, OS(%) 228, 232
- phase axis 428
- phase calculation 386, 428, 437
- phase crossover frequency 405, 607
- phase graphs 401
- phase lag lead, summary 500
- phase margin
 - advantages/disadvantages, table 614
 - Bode 415

- example 610
- Nichols 416
- Nyquist 417
- PID tuning 609
- phase plot 48, 393
- phase shift 389
- phase-lag controller 482, 486
 - as integral controller 486
 - design example 489
 - design procedure 488
 - properties 482
- phase-lead controller 482
 - as derivative controller 493
 - controller 491
 - design example 496
 - design procedure 494
 - properties 483
- physical realisability 43
- PI control, design example 477
- PI control of first order model 327
 - general analysis 327
 - procedure (design) 329
 - steady state offset, specification 327
 - transient specification 328
- PID control 11
 - damped oscillation method 600
 - decoupled form 315
 - digital formula, summary table 570
 - Laplace domain formula 314, 531
 - manual tuning 318
 - manual tuning procedure 318
 - non-interacting form 554
 - parallel form 315, 531
 - pole-placement 340
 - pole-placement procedure 343
 - relay experiment 605
 - sustained oscillation method 592
 - term selection chart 483
 - textbook and industrial forms, summary 532
 - textbook form 314, 531
 - textbook form 531
 - time domain formula 531
 - time-constant form 531
 - Ziegler and Nichols rule base 586
- PID control terms, effects of, summary table 314
- PID controller family
 - Laplace domain formula 314
 - summary table 314
 - time domain formula 314
- PID industrial forms, summary table 555
- PID tuning
 - diagram 537
 - rule-based 579
 - SCADA interface 537
- pole-placement 340
 - design specification 343
- pole-placement design, state variable model 717
- poles 27, **243**
 - finding 245
 - link to system eigenvalues 691
 - on $j\omega$ axis 268
 - open loop and closed loop 259
- pole-zero conservation 354
- pole-zero map 245
 - second order poles 249
- process 106
- process controller unit
 - description 535
 - diagram 536
- process reaction curve
 - advantages/disadvantages, table 592
 - procedure 584
- proportional and derivative control 300
 - Laplace domain formula 300
 - time domain formula 300
- proportional band 533
 - formula 534
 - relationship with proportional gain 534
 - table of values 534
- proportional control 278, **282**
 - closed loop analysis 282
 - closed loop stability 283
 - design example, with lag term 471
 - disturbance rejection 286
 - disturbance time response 287
 - final value theorem 285
 - first order system analysis 283
 - Laplace domain formula 282
 - reference tracking 285
 - summary table 290
 - time domain formula 282
- proportional control of first order model 321
 - disturbance rejection 323
 - reference tracking 323
 - steady state offset, disturbance rejection 324
 - steady state offset, reference tracking 324
- proportional control of type 1
 - model 332
 - disturbance rejection 333
 - reference tracking 333
- proportional gain 531, 579
- proportional kick 338, 476, 547
- proportional term
 - digital form 568
 - incremental form 572
- quadratic equations 23
- quarter amplitude display 600
- rate of change equation 39, 559
- rate of change variables, for state variables 622
- RC circuit, model 159
- reference signal 200, 279
- reference tracking performance 279, 280, 293
 - derivative control 302
- regulator design 280
- relay experiment 605
 - PID rule-base 609
 - procedure 608
- reverse-acting controllers 556
 - PI control 562
- right half plane (RHP) 244
- rise-time, t_r 230
- robust control 733
- roll-off rate 400, 472
- root locus
 - adding a pole 366
 - adding a zero 365
 - asymptotes 359, 360
 - branch 359, 360
 - breakaway points 360
 - closed loop poles 356
 - MATLAB investigations 357
 - poles and dynamic response 352
 - rules 360
- rotational system 137
- sampled data values 565
- sampling 565
- sampling, notation 567
- saturation characteristic 538, 539
- SCADA system 530
- scaling, input scaling 472
- second order differential equations 183
- sensitivity, magnitude plot 464
- sensitivity transfer function **461, 210**
- servo-control design 280
- setpoint 200
- settling time, t_s 229

- disturbance input 235
- shaft encoder 138
- Shaker table model
 - actuator 129
 - complete block diagram 131
 - process 124
 - Simulink model 131
 - transducer 127
- ship autopilot design, control problems 10–11
- ship steering system, state variable model 693
- signal 103
- signal equations 222
- signal take-off 201
- single degree of freedom
 - controller 713
- sinusoidal signal 386
- SISO 103, 112
- sketching table, Bode plot 439
- small change model 668
- span, transducer 108
- specification 414
- speed of response 163
- spring element 124, 175
- square systems 712
- stability, eigenvalue condition 692
- stable system 268
- start-up procedure 560
- state 621
- state space to transfer function, Laplace analysis 691
- state variable 621
 - inputs 712
 - outputs 712
- state variable block diagram 629
- state variable feedback 722
 - closed loop analysis 723
 - closed loop system matrix 724
 - reference controller 724
 - two-degrees-of-freedom state control law 722
- state variable model, procedure for defining 624
- state variable notation 623
- state variable system model
 - eigenvalue and system dynamics 697
 - non-uniqueness 696
- state vector 622
- state-feedback design freedom 723
- state-space icon, MATLAB 630
- state-space to transfer function 643
 - example 644
 - general analysis 643
 - ss2tf, MATLAB command 645
- steady state 148, 151, 158, 220
- steady state
 - design specification 223
 - performance 221
 - specification 473
- steady state conditions 666
- steady state error, e_{ss} 221
- steady state output, y_{ss} 151
- steam boiler example 621
- stem position 118
- step response 151
 - second order examples 187
- step response tests, example 315
- s-terms, magnitude and phase 437
- stiffness constant 126
- summation symbol 105
- supply disturbance 581
- sustained oscillation
 - advantages/disadvantages, table 599
 - basic principles 593
 - Bode plots 593
 - gain margin 594
 - Nyquist plot 593
 - procedure 594
 - time response plot 593
 - tuning rules, PID 595
 - ultimate gain 594
- system gain, K 151, 152, 176, 177, 387, 456
 - dimensional units 115
 - units 104
- system identification 582, 733
- system matrix, \mathbf{A} 628, 712
- system matrices 624
- system overshoot 227
- system stability 243, 267, 412
 - Nichols 518
- system type 332
- tank level system, control specification 317
- tank system
 - model 39
 - transfer function 40
- Taylor series 667
- temperature model, with
 - deadtime 165
- thermistor, nonlinear equation 662
- time constant, τ 151, 152, 158, 582
 - relation to system pole 248
- time delay 165, 368
- time domain specifications 317
- time-constant form, PID control 531
- torque 137
- trailer suspension model 174
- trailer suspension system
 - state variable model 624
- transducer 107
 - level 114
 - pressure measurement 111
- transfer function 22
- transfer function components 434
 - magnitude and phase 436
- transfer function representation
 - gain-time constant form 162
 - general polynomial form 181
 - pole-zero form 162
 - unity constant coefficient form 181
 - unity s^2 coefficient form 181
- transfer function to state space 647
 - companion form 648
 - general analysis 650
 - phase variable 648
 - tf2ss, MATLAB command 653
- transient 148, 220
- transient performance 226
- transport delay 165
- two degrees of freedom controller 713
- unbounded signal 412, 265
 - table of 266
- underdamped 178, 251
- underdamped system, frequency response 451
- unity constant coefficient form 181
- unity feedback system 207
- unity s^2 coefficient form 181
- unstable closed loop 7
- unstable system, test by eigenvalues 696
- valve 112, 118
- viscous friction constant 125
- wastewater treatment plant, description 4–6
- wind turbine systems, description 3–4
- working regions 664
- zeros 27, 243
 - at infinity 354
 - blocking zeros 259
 - finding 245
 - how do they arise? 253

- in RHP 256, 368
- open loop and closed loop 260
- origin of zeros 253
- Ziegler and Nichols rule base
586

MATLAB and Simulink index

- command history 53, 69
- command window 53, 54
- constants 59
- current directory 53, 69

- functions 59
 - as M-files 72

- graph, co-ordinate finding 62

- help command 60
- help window 60, 70
- house heating system
 - comparator 84
 - setpoint 85
 - thermostat block 90
- house model **86**
 - disturbances 88
 - first order differential equation 87
 - first order dynamics 88
 - rate of change equation 87
 - system gain 88
 - time constant 88

- labels on plots, axes 62
- launch pad 53, 68

- MATLAB commands, table 78

- matrices
 - data entry 58
 - inverse 59
 - power 59
 - product 59
 - sum 59
 - transpose 59
- M-files 70
 - types 71
- multiple plots 62

- operations 55

- plotting, MATLAB 61
- polynomials
 - convolution 58
 - data entry 57
 - deconvolution 58
 - divide 58
 - manipulations 58
 - multiply 58

- running MATLAB 53

- Simulink
 - connecting blocks 95
 - continuous library 92
 - data input 95

- icon
 - constant 94
 - gain 94
 - relay 96
 - scope 94
 - sum 94
 - transfer function 94
- library browser 84
- model construction procedure 91
- parameters command 96
- scope 91
- simulation command 96
- SISO design tool 75
- state-space icon 630

- time responses, commands 66
- transfer function input 64
- transfer function manipulation 65

- vectors 55
 - column 55
 - manipulations 56
 - row 55
 - size of 55

- workspace 53, 68

MATLAB command index

There is a table of MATLAB commands on p. 78

- bode 431
- clear 61
- eig 689
- feedback 67, 212
- figure 62
- freqresp 430
- ginput 62
- hold on 62
- inv 687

- ngrid 511
- nichols 511
 - ngrid 511
- pade 168
- plot 61
- pole 66
- pzmap 66
- rlocus 375
- rltool 75, 363, 375, 432

- semilogx 63
- ss2tf 645
- step 66
- tf2ss 653
- variable, ans 61
- who 61
- zero 66