

Control Theory for Bioengineers

2016 Changes

Version 0.81 Corrections to Chapters 11, 13 with major editing of Chapters 1, 2 and 3.

Version 0.82 Corrections to Chapters 1, 2, 3 and 4.

Version 0.83 Corrections to all chapters, additions to complex numbers in appendix, additional material in negative feedback chapter and new chapter on linear pathways

2017 Changes

Version 0.84 Updated Laplace transform chapter, add more discussion on convergence criteria.

Version 0.841 Fixes to Chapter 1

Version 0.842 Fixes to Chapter 1

Version 0.843 Updated chapter 3 by including introduction to PID controllers

Version 0.844 Updated chapter 7 with some minor fixes and replaced font in figure 7.1 which wasn't displaying the symbol alpha.

Version 0.845 Fixed error on page 153, time constant should be τ .

Version 0.846 Redid the LTI linearity proof in chapter 8

Version 0.847 Redid exercise 7.7

Version 0.85 Added examples to Appendix F

Things to do:

1. Exercises for Chapter 8
2. Check additional examples, 10.8
3. Check gene figures use x not S

*Herbert M. Sauro
University of Washington
Seattle, WA*

Ambrosius Publishing

Copyright ©2011-2017 Herbert M. Sauro. All rights reserved.

First Edition, version 0.85

Published by Ambrosius Publishing

Typeset using L^AT_EX 2_&, TikZ, PGFPlots, WinEdt
and 11pt Math Time Professional 2 Fonts

pgf version is: 3.0.1a

Limit of Liability/Disclaimer of Warranty: While the author has used his best efforts in preparing this book, he makes no representations or warranties with respect to the accuracy or completeness of the contents of this book and specifically disclaim any implied warranties of merchantability or fitness for a particular purpose. The advice and strategies contained herein may not be suitable for your situation. Neither the author nor publisher shall be liable for any loss of profit or any other commercial damages, including but not limited to special, incidental, consequential, or other damages. No part of this book may be reproduced by any means without written permission of the author.

Library of Congress Cataloging-in-Publication Data:

ISBN 10: 0982477384 (paperback)

ISBN 13: 978-0982477380 (paperback)

Printed in the United States of America.

Contents

Preface	iv
1 Introduction to Modeling	1
1.1 Introduction	1
1.2 Build a Simulation Model	2
1.3 Models and Systems	5
1.4 Model Variables, Inputs, and Parameters	7
1.5 Classification of Models	15
1.6 Electrical Models	16
1.7 Block Diagrams	27
1.8 Analytical Solutions	29
1.9 Dimensions and Units	31
1.10 Approximations	32
1.11 Behavior	33
Exercises	34
2 Simulation Software	39
2.1 Software for Solving ODEs	39
2.2 Other Software	41
2.3 Domain Specific Tools: Tellurium/Python	44
Further Reading	46
Links to useful modeling tools	46
3 Introduction to Feedback Control	47
3.1 What is Control Theory?	47

3.2	Historical Perspective	48
3.3	Simple Quantitative Analysis	52
3.4	PID Controllers	56
3.5	Examples of control systems	58
	Exercises	62
4	Biological Networks	63
4.1	Gene Regulation	63
4.2	Metabolic Pathways	64
4.3	Protein Networks	65
4.4	Stoichiometric Networks	68
4.5	Reaction Kinetics	72
4.6	Elementary Mass-Action Kinetics	76
4.7	Chemical Equilibrium	76
4.8	Mass-action and Disequilibrium Ratio	78
4.9	Modified Mass-Action Rate Laws	79
4.10	Elasticity Coefficients	80
4.11	General Elasticity Rules	85
4.12	Mass-Balance Equations	88
4.13	Stoichiometry Matrix	92
4.14	The System Equation	97
4.15	Building Simple Gene Regulatory Models	98
	Further Reading	100
	Exercises	101
5	The Steady State	105
5.1	System States	105
5.2	Equilibrium	105
5.3	Transients	108
5.4	Steady State	110
5.5	Computing the Steady State	113
5.6	Effect of Different Perturbations	120

5.7	Stability and Robustness	122
5.8	Introduction to Stability	123
5.9	Sensitivity Analysis	126
	Further Reading	128
	Exercises	128
	Tellurium Scripts	130
6	Inputs to a System	135
6.1	Unit Step Function	136
6.2	Delayed Signals or Shifting	137
6.3	Addition of Signals	140
6.4	Ramp Signal	142
6.5	Impulse Signal	144
6.6	Sinusoidal	149
6.7	Exponential	153
	Exercises	157
6.8	Appendix	159
7	Linear Systems	161
7.1	Linear Systems	162
7.2	Time-Invariance	164
7.3	Linearization	166
	Exercises	174
8	State Space Representation	177
8.1	State Space Representation	177
8.2	Examples of State Space Models	182
	Exercises	187
9	Response of Linear Systems	189
9.1	Solutions to Linear Systems	189
9.2	Relationship to Linearized Equations	194

9.3	General Solutions	195
9.4	Relation to Eigenvalues and Eigenvectors	200
	Exercises	204
10	Laplace Transforms	207
10.1	Introduction	207
10.2	Definition	208
10.3	Examples of Laplace Transforms	210
10.4	Properties of the Laplace Transform	215
10.5	Inverse Transforms	220
10.6	Solving Differential Equations	225
10.7	Laplace Transforms of Common Signals	228
10.8	Additional Examples	229
	Exercises	231
10.9	Appendix	233
11	Zero-State Response	235
11.1	Introduction	235
11.2	Zero-State Response to an Impulse	235
11.3	Convolution	237
11.4	Relationship to State Equations	242
11.5	Transfer Function	249
11.6	Steady State in the Laplace Domain	253
11.7	Block Diagrams	256
11.8	Summary of Transfer Functions	259
	Exercises	259
12	Fourier Series	263
12.1	Introduction	263
12.2	Dirichlet Conditions	269
12.3	Exponential Form	270
12.4	Frequency Spectrum	270

12.5 APeriodic Signals: Fourier Integral and Transform	270
12.6 Return to the Laplace Transform	273
12.7 Fourier Transform of Discrete Data	274
13 Frequency Response	279
13.1 Introduction	279
13.2 Linear Systems and Sinusoids	280
13.3 Frequency Response	283
13.4 Laplace and Fourier Transforms	284
13.5 Bode Plots	286
13.6 Bode Plots using Python	301
13.7 Zero Frequency Response	307
Exercises	309
13.8 Appendix	310
14 Stability	313
14.1 Introduction	313
14.2 Internal Stability	313
14.3 External Stability	318
14.4 Phase Plots	318
14.5 Routh-Hurwitz Method	323
14.6 Numerical Determination of Roots	331
14.7 Stability from Frequency Response	332
Further Reading	336
Exercises	336
Tellurium Scripts	338
15 First-Order Systems	347
15.1 Introduction	347
15.2 Zero-Input Response	350
15.3 Impulse Response	350
15.4 Unit Response	351

15.5	Ramp Response	354
15.6	Frequency Response	354
	Exercises	357
16	Second-Order Systems	359
16.1	Introduction	359
16.2	Overdamped Case: $\zeta > 1$	362
16.3	Critically Damped Case: $\zeta = 1$	363
16.4	Underdamped Case: $0 < \zeta < 1$	363
16.5	No damping: $\zeta = 0$	364
16.6	Transient Response Metrics	365
16.7	Frequency Response	368
16.8	Example	369
	Further Reading	372
	Exercises	372
16.9	Appendix	372
16.10	Appendix	374
17	Cellular Networks	377
17.1	Systems Equation	377
17.2	Output Equations	383
17.3	Canonical Transfer Functions	386
18	Scaled Transfer Functions	391
18.1	Introduction	391
18.2	Scaled Transfer Functions	392
19	Structural Constraints	397
19.1	Introduction	397
19.2	Structural Constraints	398
20	Linear Pathways	403
20.1	Introduction	403

20.2	Front Loading	405
21	Negative Feedback	407
21.1	Introduction	407
21.2	Types of Feedback	408
21.3	Proportional Control	409
21.4	Reduced Gain	411
21.5	Accuracy	412
21.6	Linearization	416
21.7	Sensitivity	417
21.8	Sensitivity at Zero Frequency	419
21.9	Implications for Drug Targeting	421
21.10	Stability	422
21.11	Biological Interpretation of the Transfer Function	433
21.12	PID Controller	434
	Exercises	436
22	Positive Feedback	439
22.1	Basic Properties	439
22.2	Stability of Positive Feedback	443
	Further Reading	452
	Exercises	452
23	Oscillators	455
23.1	Introduction	455
23.2	Negative Feedback Oscillator	456
23.3	Relaxation Oscillators	459
23.4	Oscillator Classification	461
Appendix A	Symbols	465
A.1	List Of Symbols	465
Appendix B	Numerical Methods	467

B.1	Introduction	467
B.2	Numerical Solutions	468
	B.2.1 Euler Method	470
	B.2.2 Modified Euler or Heun Method	474
	B.2.3 Runge-Kutta	475
	B.2.4 Variable Step Size Methods	476
	B.2.5 Stiff Models	478
	Further Reading	479
	Exercises	479
	Appendix C Modeling with Python	481
C.1	Introduction to Python	482
C.2	Describing Reaction Networks using Antimony	486
	C.2.1 Initialization of Model Values	489
C.3	Using libRoadRunner in Python	489
	C.3.1 Time Course Simulation	490
	C.3.2 Plotting Simulation Results	492
	C.3.3 Applying Perturbations to a Simulation	492
	C.3.4 Steady State and Metabolic Control	493
	C.3.5 Other Model Properties of Interest	495
C.4	Generating SBML and Matlab Files	496
C.5	Exercise	497
	Appendix D Enzyme Kinetics in a Nutshell	499
D.1	Michaelis-Menten Kinetics	499
D.2	Reversibility and Product Inhibition	500
D.3	Reversible Rate laws	501
D.4	Haldane Relationship	501
D.5	Competitive Inhibition	501
D.6	Cooperativity	503
D.7	Allostery	505
	Further Reading	507

Appendix E Visualizing Simulations	509
E.1 Time Course Simulation	509
E.2 NullClines	509
E.3 Bifurcation Plots	510
Appendix F Math Notes	513
F.1 Polynomials	513
F.2 Absolute Values	515
F.3 Radian Measure	515
F.4 Common Integrals	517
F.5 Taylor Series	519
F.6 Complex Numbers	521
F.7 Eigenvalues and Eigenvectors	530
Appendix G Common Integrals	533
Appendix H Table of Laplace Transforms	535
References	541
History	547
Index	1

Preface

The choice of text books on control theory is very broad, however most books tend to focus on control systems found in mechanical and electrical systems. Far fewer text books apply control theory to biological systems. This book is an introduction to control theory that looks at control mechanisms found in living cells. The first half of the book is largely traditional using examples from electrical, mechanical and biochemical systems. The second half delves more deeply into the control theory of biochemical systems. The material presented in this book is suitable for late sophomore or early junior years, or late first year and second year for the UK university system.

As with my book on Enzyme Kinetics for Systems Biology, I have decided to publish this book myself. I have found that traditional publishers have yet to catch up with modern publishing trends, in particular the loss of copyright on the text as well as any figures and even more problematic, the inability to rapidly update text to correct errors or when new material needs to be added. Publishers still handle corrections via errata pages rather than updating the book itself. With today's print on demand technology, the restrictions imposed by publishers seem unnecessary.

There are many people and organizations who I should thank, but foremost must be my infinitely patient wife, Holly, and my two boys Theodore and Tyler who have put up with the many hours I have spent working alone. I would like to thank Holly in particular for helping me edit the text. I am also most grateful to the National Science Foundation and the National Institutes of Health who paid my summer salary so that I could allocate some time to write, edit and research. I would also like to thank the many undergraduates, graduates and colleagues who have contributed to this work. In particular I want to thank my two teachers, David Fell and the late Henrik Kacser who I had the privilege to work with as a graduate student and postdoctoral fellow. I had many hours of fruitful conversations with John Doyle, Mustafa Khammash and Joseph Hellerstein¹ at UW. I would also like to thank Kyung Kim, a gifted applied mathematician, who helped me understand some of the mathematical subtleties. Finally I would like to thank my many students who provided many useful suggestions for improving the text, and identifying errors in some of the examples. All remaining errors are as they say, my own responsibly.

Many thanks to the authors of the \TeX system, Mik \TeX (2.9), TikZ (2.1), PGFPlots (1.9), WinEdt (6.0), Inkscape (0.48.4), and Createspace for making available such amazing tools for technical authors. It is these tools that make it possible for individuals like myself to publish. Finally, I should thank Michael Corral (<http://www.mecmath.net/>) and Mike Hucka (www.sbml.org) whose \LaTeX work inspired some of the visual styles I used in the text.

August 2015
Seattle, WA

HERBERT M. SAURO

¹<https://sites.google.com/site/josephlhellerstein/>

1

Introduction to Modeling

“You take the blue pill, the story ends, you wake up in your bed and believe whatever you want to believe. You take the red pill, you stay in Wonderland, and I show you how deep the rabbit hole goes.”

– The Matrix, 1999

1.1 Introduction

We are surrounded by machines of all kinds, most of which are surprisingly reliable. Our mobile phone always connects, postings to social web sites always reach their audience, and space probes, almost without fail, reach their destination no matter how far they have to travel. The reliability of our machines can be traced to the engineering of control mechanisms that ensure that our machines behave in predictable ways and adapt in unpredictable environments.

The same can be said for biological systems; on average we breathe 672,768,000 times in a lifetime and our heart beats roughly 100,000 times a day without fail; we can stand, walk and run, usually without any incident even though standing is an inherently unstable position. A complex system such as a human, can remain fully functional for almost 100 years. Part of the reason for this lies in the sophisticated systems that exist to regulate, control and repair our tissues. We can understand how man-made and natural systems achieve these feats using the science of control. Control is concerned with understanding how systems behave predictably and adapt to external disturbances. One of the simplest examples of a regulated system is the temperature thermostat. A thermostat is able to regulate a heat source such that the temperature of a room is held at a steady level irrespective of what is happening elsewhere. Engineers have for many years possessed a well developed set of modeling

and theoretical tools under the umbrella of control theory that permits the design and construction of predictable systems. In more recent years control theory has started to be applied to understanding the control mechanisms we find in living organisms.

Students who are well versed in modeling and building computational models may omit the first chapter. For those who haven't, we will start by considering the building and simulation of a simple water tank model.

1.2 Build a Simulation Model

Water Tank Model

Figure 1.1 shows two cylindrical water tanks. The first tank is fed with water at a rate Q_1 ($\text{m}^3 \text{ s}^{-1}$). This tank drains into a second tank at a rate Q_2 which in turn drains to waste at a rate Q_3 . The second tank has an additional feed of water flowing in at a rate Q_4 . The height of the water level in each tank is given by h_1 and h_2 respectively. Each tank has a cross sectional area A and the volume of water in each tank is given by V_1 and V_2 .

Our objective is to build a quantitative model that will allow us to predict how long it will take to fill the tanks with water or how that rate of filling is affected by the various inflow and outflow rates.

To build a model we will assume that water is conserved as it flows from one tank to the next. That is there are no leaks or evaporation of water. This means we can state that the rate of change in the volume of water in a given tank is the rate at which the water enters minus the rate at which it leaves. For the first tank we can state:

$$\frac{dV_1}{dt} = Q_1 - Q_2 \quad (1.1)$$

Note that the units of flow, Q_i , are in volume per unit time. If we want the equation in terms of the rate of change of height then we should recall that the volume of a cylinder is $V = Ah$, or by differentiating with respect to time:

$$\frac{dV}{dt} = A \frac{dh}{dt}$$

We can use this result to express equation (1.1) in terms of the rate of change in height:

$$\frac{dh_1}{dt} = \frac{Q_1 - Q_2}{A}$$

Using Torrielli's Law we know that the rate of water flowing out of a given tank, i , is equal to:

$$Q = K_i \sqrt{h_i}$$

Where K_i is a constant related to the resistance of the output pipe. We can use this result to swap out Q_2 to give:

$$\frac{dh_1}{dt} = \frac{Q_1 - K_1 \sqrt{h_1}}{A} \quad (1.2)$$

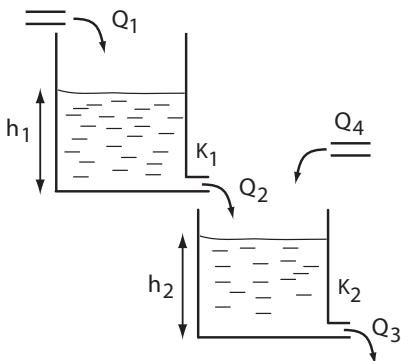


Figure 1.1 Water Tank Model

Likewise we can write out the rate of change of the height in the second tank, noting the additional flow Q_4 into the tank:

$$\frac{dh_2}{dt} = \frac{Q_2 + Q_4 - K_2 \sqrt{h_2}}{A} \quad (1.3)$$

The equations (1.2) and (1.3) represent the dynamical equations that can be used to describe the evolution of the height of water as the water tanks empty and fill.

With the differential equations in hand, the next stage is to assign values to the various parameters in the model. For example, the cross-sectional areas of the tanks, the flow Q_4 into the second tank, the flow Q_1 into the first tank, and the two tank constants, K_1 and K_2 . Once the parameters are assigned, we initialize the two heights h_1 and h_2 , then enter the equations into a computer program to solve the differential equations. We will leave the actual simulation task to a later Chapter. For now we just show the results from a simulation of the tank model using Matlab, Python, or a Python based simulation package called Tellurium (Figure 1.2). The plot displays the heights, h_1 and h_2 as water fills the tanks one at a time.

To summarize, we have learned a number of things from this exercise. First, models include assumptions and simplifications, and it is the careful selection of these that marks a good model from a bad one. The assumptions in this model include: 1) We assume that Torrielli's Law at low flow rates is a reasonable approximation; 2) The flows Q_1 and Q_4 are constant; and 3) there are no leaks or evaporation in the system. The other thing we have learned is that there are at least four different types of quantities in the model. We will go into more detail in the next section but for now, we can briefly indicate what these quantities are:

1. *Inputs* to the system such as Q_1 and Q_4 .

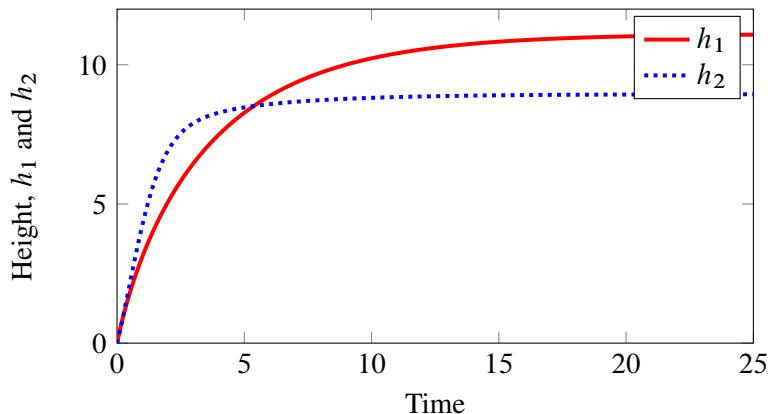


Figure 1.2 Simulation of the tank model.

2. *Model variables* which include the two heights, h_1 and h_2 , which change in time as the system evolves.
3. A number of *physical parameters* which are fixed during the study of the model but which we could in principle change. In the tank model these include the volume, cross-sectional area of each tank, and the diameter of the outflow pipes.
4. Finally, there are *parameters which we cannot change* such as the force of gravity.

Class Exercise 1

Given the model of the water tank system, answer the following questions:

- a) List some of the assumptions we made in building the model.
 - b) Plot the rate of outflow, Q_2 , as a function of the height of water, h_1 at a given resistance, K_1 .
 - c) Assuming that Q_1 and Q_4 are fixed, and we start with both tanks empty, what do you expect to happen over time as water flows in?
 - d) If you are well versed with solving differential equations using MATLAB or Python, build a computer model of the tank system, assign suitable values to the parameters in the model and run the simulation to plot the height of water in the tanks over time. Assume both tanks are empty at the start of the simulation.
 - e) Investigate the effect of increasing and decreasing the resistance parameters, K_1 and K_2 on the model.
-

1.3 Models and Systems

In the previous section we built a quantitative model of a water tank system but what exactly is a model?

A **model**, according to the Oxford English Dictionary is:

“A simplified or idealized description or conception of a particular system, situation, or process, often in mathematical terms, that is put forward as a basis for theoretical or empirical understanding, or for calculations, predictions, etc.”

This definition embodies a number of critical features that defines a model, the most important is that a model represents an **idealized description**, a simplification, of a real world process. This may at first appear to be a weakness but simplification is usually done on purpose. Simplification is important because it allows us to comprehend the essential features of a complex process without being burdened and overwhelmed by unnecessary detail. In the water tank model we assumed that we could account for all the water flowing in and out of the tanks, that is the tanks didn't leak and that evaporation of water was negligible.

Models come in various forms including verbal, written text, visual, mathematical and others. Biology, particular molecular and cell biology, has a long tradition of using visual models to represent cellular structure and processes; one need only look through a modern textbook to see instances of visual models on every page. Visual models have been immensely useful at describing complicated biological processes but are limited in their scope. A more interesting way to describe models is to use mathematics, a language designed for *logical reasoning*. Mathematical models are useful in biology for a number of reasons, but the three most important are increased **precision, prediction**, and the capacity for **analysis**. Analysis can be carried out either by simulation or by analytic means. Although visual models can be used to make predictions, the kinds of predictions that can be made are limited. The use of mathematical models opens up whole new vistas of study which visual models simply can not match.

One word that has been used but has been undefined is the word “System”. Here we define what we mean by the system:

System: The part of the Universe we are interested in studying with a defined boundary and set of processes within.

Open, Closed, and Isolated Systems

Like many other scientific studies, modeling divides the world into two parts, the system and the surroundings. The system is that part of the universe we wish to focus our attention. This might

be a single enzyme reaction, a pathway, a single cell, an organ, or an entire multicellular organism. The surroundings includes everything else. The boundary between the system and the surroundings can have different properties which determines whether the system is **isolated, closed or open** (See Figure 1.3). All living systems are open because they exchange energy and matter with their surroundings. An isolated system is one where nothing can enter or leave. Such systems can only be approximated as it is very difficult to completely isolate a part of the Universe. A thermos flask can be considered an isolated system, although an imperfect one. A closed system is one that can exchange energy with the surroundings. Any mass in the system is isolated from the surroundings. An example of a closed system is the bomb calorimeter.

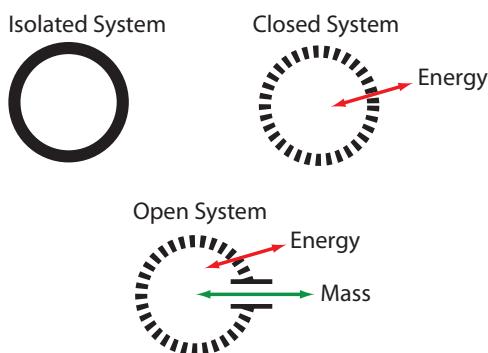


Figure 1.3 Open and Closed Systems.

Class Exercise 2

For each of the following systems, decide whether the system is isolated, closed or open. Comment on the nature of the surroundings.

- i) A system represented by a mechanical clock slowly winds down in a room controlled by a thermostat.
 - ii) A car engine running idle in the open air.
 - iii) A bacterial culture is grown in batch and kept in a sealed and insulated chamber.
-

Different Ways to Represent Physical Models

The tank model described in section 1.2 was built using a set of ordinary differential equations (ODEs) and solutions to these equations could be obtained using a digital computer. There are however many other ways to build and solve a model of a physical process. Table 1.1 lists some of the more common and interesting approaches that people have used in the past to construct and simulate models.

Table 1.1: Different ways to construct and solve physical models

Electrical Circuits	General purpose analog computer [66] WWII V2 guidance system [79] Neuromorphic electronics [4, 61]
Mechanical and Fluid	Slide rule [76] Curta [69] Tide predicting machine [78] Computing projectile trajectories [75] Differential analyzer (solves ODEs) [70] Antikythera mechanism (planetary motion) [67] Water tanks - MONIAC economic model [73]
Purely Mathematical	Algebraic Equations Linear differential equations Linear difference equations Partial differential equations Probabilistic models Statistical models
Digital Computer	Solving ODEs and PDEs Agent based models (multicellular systems) Cellular automata [68] Emergent systems (Ant models) [71] Fractal models [72] Neural networks [74]

1.4 Model Variables, Inputs, and Parameters

Notation:

All vectors will be indicated using bold lower case roman letters, for example \mathbf{x}, \mathbf{u} .

All matrices will be indicated using bold upper case roman letters, for example \mathbf{A}, \mathbf{B} .

Variables, Parameters and Constants

Figure 1.4 classifies the different kinds of quantities we find in a model. These include absolute constants, parameters, inputs, dependent variables, independent variables and outputs. This is a long list so a concrete example will help better explain each quantity.

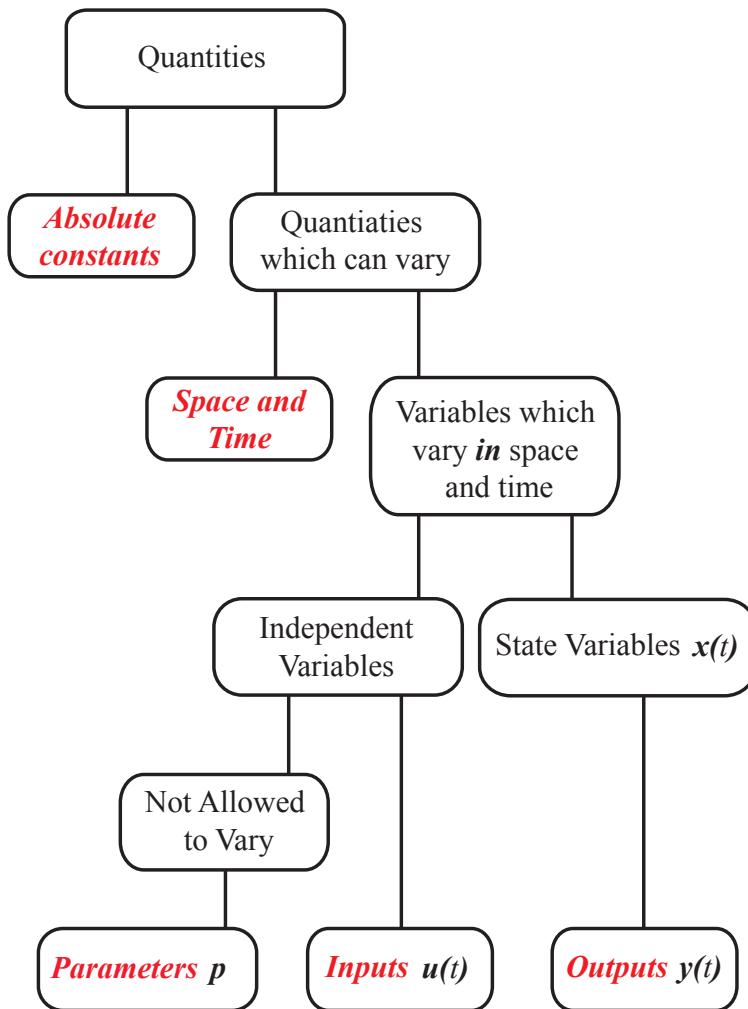


Figure 1.4 Classification of quantitative terms.

The following diagram illustrates a series of chemical reactions, designated, v_1 , to v_3 . Each chemical

reaction converts a reactant into a product at a rate v_i .



The rates are governed by the following three rate laws:

$$\begin{aligned} v_1 &= k_1 X_o \\ v_2 &= k_2 S_1 \left(1 - \frac{S_2/S_1}{e^{-\Delta G^o/RT}} \right) \\ v_3 &= k_3 S_2 \end{aligned}$$

where ΔG^o is the standard free energy, R the gas constant, and T the temperature. Note that $e^{-\Delta G^o/RT}$ equals the equilibrium constant, K_{eq} , see equation (4.10). We will make a number of assumptions in building a model of this system:

1. The reactions take place in a constant unit volume at a constant temperature.
2. Species X_o and X_1 are at fixed concentrations by some external and unspecified process.
3. Reactions occur in well-stirred volumes, that is there are no spatial gradients.

Let us list each type of quantity in this model:

Constants Constants usually refer to absolute constants in the model. In this case it would include Napier's constant e , and the gas constant, R . Constants cannot typically be changed by the experimenter.

Parameters The parameters of a model are those quantities which could, in principle, be changed by the experimenter but which *remain constant* when the model is used to make predictions. In the pathway model one can imagine that the ΔG^o and the reaction rate constants are parameters. However, these particular parameters are not easily changed. It might be possible to change them by altering the ionic composition, the solvent, or temperature. Usually however we treat kinetic and thermodynamic parameters as absolute constants. The exception to this is if the reactions are enzyme catalyzed. In this case one could change the enzyme concentration or through site-direct mutagenesis, change the enzyme kinetic properties.

The unchanging properties of a system are described by a set of **parameters**:

Inputs The inputs to the system are those quantities which are under direct control of the experimenter and can conceivably be changed by the experimenter during the course of a model simulation. In the pathway model, the inputs include X_o and X_1 . Other examples of inputs include nutrient sources, temperature, enzyme concentrations, and any kind of external effector such as a drug or inhibitor.

In biology the inputs are often clamped to some fixed values (*cf.* voltage clamp), but can also be varied in some controlled way by the experimenter. The clamping mechanism can simply be a large external reservoir so that any exchange of mass between the system and the external reservoir has a negligible effect on the external concentration. Alternatively, there may be active mechanisms maintaining an external concentration. A classic example of active maintenance of an external variable is the voltage clamp used in electrophysiology.

The inputs to a system at time t are described by a set of **input variables**:

$$\mathbf{u}(t)$$

Sometimes inputs are also represented using the symbol $\mathbf{p}(t)$.

External concentrations may also change slowly in time compared to the timescale of the model so that over the study period, the external concentrations change very little. A typical example is the study of a metabolic response over a timescale that is shorter than the change in gene expression. This permits a modeler to study a metabolic pathway without considering the effect of changes in gene expression. The external species inputs such as X_o are also called **boundary variables** because they are considered to be at the boundary of the system.

Molecular species that are not dependent on the action of the model are sometimes called **boundary species**. Often boundary species are fixed by the modeler but it is possible for the modeler to impose a particular change in a boundary species to simulate, for example the administration of a drug as a bolus or as a continuous infusion.

Dependent Variables The dependent variables, also called the **state variables**, are the minimum set of variables to describe the state of a system. In biochemical modeling these variables often include the concentrations of molecular species or voltages across membranes. In the pathway model (Figure 1.4), the two dependent variables are S_1 and S_2 . The distinguishing feature that separates the input variables from the dependent variables is that while the inputs can be directly controlled by the model observer, the only way the dependent variables can change is through the operation of the model itself, i.e. they depend on the model. In biochemical modeling the dependent variables are also called **floating species**.

Molecular species that change in time as a result of the action of the model are sometimes called **floating species**.

Initial Conditions

An important part in defining a model is to assign initial or starting values to all the state variables. In the dynamics literature the initial values are called the **initial conditions**. For a state variable $x(t)$, a common notation to indicate an initial condition is:

$$x(0) = 1.2345$$

That is at time zero ($t = 0$) we set the initial value for the state variable, $x(t)$ to 1.2345.

The distinction between the inputs and the dependent variables is important. Once the choice is made, the separation is strictly adhered to during the course of a study. This means for example that the environment surrounding the physical system will, *by definition*, be *unaffected* by the behavior of the system. If for some reason parts of the environment do change as a result of the system and can in turn affect the system in some way, then these parts must now be considered part of the system.

The state of a system at time t is described by a set of **state variables**:

$$\mathbf{x}(t)$$

They are the smallest set of variables that define the state of the system.

Independent Variables There are two main independent variables in biochemical modeling: time and space. In this book we will only be concerned with time dependent models.

Outputs The outputs are the readouts from the model, and are the quantities that an experimenter can actually measure. However the outputs are sometimes no different from the dependent variables, particularly in a computer model. Experimentally however, there are times when it is not possible to measure a particular dependent variable or when a derived measurement is required or measured. For example, we will often report the pH rather than the actual hydrogen ion concentration. In the case when we cannot make a direct measurement, we instead use a proxy, for example a fluorescence measurement or another molecular marker that follows the variable of interest. In the case of derived quantities, a very common one is the pathway flux. We will cover this in more detail later in the book. Separating the outputs from the independent variables is an important part of classical control and metabolic control theory.

Internal Variable	External Variable
State variable	Inputs
Dependent variable	Independent variable
Floating variable (species)	Boundary variable (species)

Table 1.2 Synonyms for internal and external variables.

The output of a system at time t is described by a set of **output variables**:

$$\mathbf{y}(t)$$

If possible we try to make sure that an output variable is directly proportional to one of the state variables.

The various quantities used in a model are summarized in equation 1.4.

Summary of model quantities:

$\mathbf{x}(t)$	state variables
$\mathbf{y}(t)$	output variables
$\mathbf{u}(t), \mathbf{p}$	inputs and parameters

(1.5)

Class Exercise 3

For the water tank model described in section 1.2, identify the state variables, outputs, inputs and parameters of the system.

Variables: Heights, h_1 and h_2 in tank one and two. Parameters: K_1 , K_2 and the cross-sectional area, A . Boundary parameters: Q_1 and Q_4 Independent variable: Time, t

Taken together if we choose to describe our system using a set of ordinary differential equations, then we can write the system equation shown in equations (1.6).

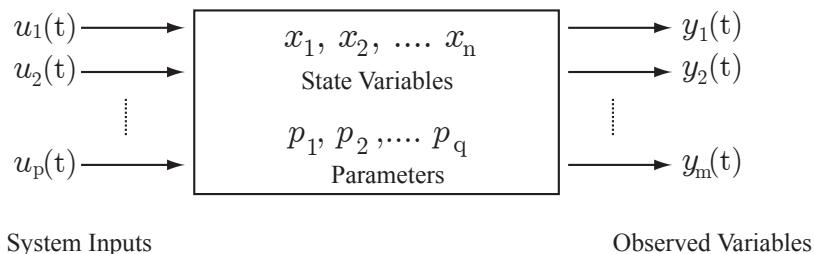


Figure 1.5 System Diagram with Inputs and Outputs.

System Equations:

$$\frac{dx}{dt} = f(x(t), u(t)) \quad (1.6)$$

$$y = g(x(t), u(t))$$

Mathematical Descriptions of Models

Table 1.1 lists the many different ways in which a model can be represented and simulated. However one thing they all have in common is that the models are first expressed in mathematical form. The form of this expression determines how the model will be constructed, and how it will be solved or simulated. In particular, decisions must be made about how the variables and parameters in the system are best described. Model variables can be described either using **discrete** or **continuous** variables. For example, the change in the level of water in a tank is more reasonably described using a continuous variable such as the height. On the other hand, it might be more realistic to describe the dynamics of lion predation on the Serengeti using a discrete model where individual lions can be represented. It does not make sense to refer to 34.67 lions in a model. The choice of whether to use a discrete or continuous description depends entirely on the system being studied and the questions posed.

A **discrete variable** is one that cannot take on all values within a given numeric range. For example, the number of airplanes in the sky at any one time is a discrete number. In statistics this is generalized further to a finite set of states, such as true/false or combinations in a die throw.

Continuous variables can assume all values within a given numeric range. For convenience we will often represent a measurement as a continuous variable. For example we will often use a continuous variable to represent the concentration of a solute as it is unwieldy to refer to the concentration of a solute as 5,724,871,927,315,193,634,656 molecules per liter.

Another important categorization is whether the model should be represented in a **deterministic** or **stochastic** form. A deterministic model is one where if we repeated the simulation using the same starting conditions, we would get exactly the same result again. That is, the future state of the model is completely determined by its initial starting point. The model of the water tanks filling up is an example of a deterministic model. In contrast, each time we run a computer simulation of a stochastic model, we get a slightly different outcome even though the starting conditions are the same.

The reason for this is that each step in the simulation is determined by one or more random processes. To give an example, modeling lion predation on the Serengeti could be modeled as a stochastic process. It is not guaranteed that a lion will catch its prey every time; instead, there is a probability it will succeed. To model this process the computer simulation would throw a die to determine whether the lion had succeeded or not. Running the simulation again would naturally give a slightly different outcome because the die throws will be different in each run.

A **deterministic model** is one where a given input will always produce the same output. For example, in the equation, $y = x^2$, setting x to 2 will always yield the output 4.

A **stochastic model** is one where the processes described by the model include a random element. This means that repeated runs of a model will yield slightly different outcomes.

We can classify a model as a combination of the above attributes. The water tank model uses a deterministic, continuous approach. The lion model might use a discrete and stochastic approach. In this book we will be primarily concerned with deterministic and continuous models. Table 1.3 shows the four combinations and examples where each combination might be appropriately used.

Type	Example
Continuous/Deterministic	Projectile motion
Continuous/Stochastic	Brownian motion
Discrete/Deterministic	Large population dynamics
Discrete/Stochastic	Small population dynamics

Table 1.3 Examples of different kinds of model.

Class Exercise 4

How you would describe each of the following systems in terms of how you might model them?

1. A game of chess
 2. Two teams playing American football
 3. A forest fire
 4. Gene regulation in a bacterium
 5. The beating of a healthy heart
 6. A tornado
 7. Growth of a tumor
 8. Upper atmosphere chemistry
 9. A digital computer
 10. Electron flow through a single transistor
 11. A population of cells from which emerges a cancerous cell
 12. Metabolism
 13. The swimming of a single *E. coli* through water
-

1.5 Classification of Models

In addition to classifying models as discrete/continuous and deterministic/stochastic there are additional properties of models that can be used to categorize them as shown in Table 1.4.

1. Linear or Non-Linear
2. Dynamic or Static
3. Time invariant or time dependent
4. Lumped or distributed parameter models
5. Causal

Table 1.4 Other model categories.

Dynamic and Static Models

A static model is one where the variables of the system do not change in time. For example, a circuit made up of only resistors can be modeled as a static system because there are no elements in the circuit that can store or dissipate charge. Given this, currents and voltages are considered instantaneous without any time evolution.

Time Invariant Systems

A time invariant model is one where the model does not explicitly depend on time. This means that running the model with a starting time of $t = 0$ or $t = 10$ makes no difference to the time evolution of the model. If a parameter of the system depends on time, then the model is called time dependent. An example of a time dependent model is where we apply a drug in the form of a pulse and the duration of the pulse depends on when the drug is administered. An example of a time dependent non-biological model is a parking lot where the price of a ticket depends on the time of day. We will have more to say about time invariant systems in a later chapter when we talk about linear time invariant systems (LTI).

Lumped and Distributed Parameter Models

Many complex models can be approximated with a single number. For example we often describe a resistor using a single value, its resistance. In reality, the resistor has a length, a diameter and a chemical composition. The resistance is a function of all these properties that make up the resistor. We could model the resistor by slicing it up into many small compartments and compute the resistance as a systemic property. In the former case we have what is called a lumped parameter model, in the second case a distributed parameter model.

Linear and Nonlinear Systems

A linear system is one where the output changes in proportion to the inputs. That is, doubling the input will result in a doubling of the output. We will discuss linear and non-linear systems in much more detail in Chapter 7.

Causal

A causal system is one where the output depends only on the current or past state of the system. Any time dependent physical system must therefore be causal. Noncausal systems can be mimicked by prerecording some known future event so that the system can respond to events that have not yet occurred. The importance of causal systems is that they can be implemented practically.

1.6 Electrical Models

In this section we are going to briefly review electrical systems. Throughout the book we will use both electrical and biochemical pathways to illustrate concepts in control theory. Box 1.0 reviews some of the more important results in basic electricity. Let us begin by looking at a simple circuit and

building the corresponding model. This circuit includes a voltage source, V , a capacitor, C , resistor, R and a switch, S . Initially with the switch in the off position there is no charge on the capacity. Once the switch is set to on, current flows around the circuit and in the process charges up the capacitor until electrical steady state is reached. During the charging event, the voltage across the capacitor, V_C changes. Our model will therefore describe how V_C changes in time.

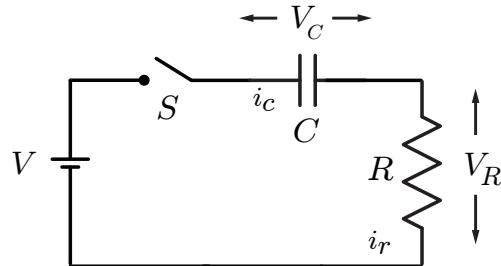


Figure 1.6 Resistor/Capacitor Circuit.

To begin the derivation we start by writing out Kirchhoff's voltage law around the circuit:

$$V = V_R + V_C$$

Now invoke Ohm's law to replace V_R :

$$\begin{aligned} V_R &= i_r R \\ V &= i_r R + V_C \end{aligned}$$

Using Kirchhoff's current law, we note that at any instant (charge cannot accumulate at a node):

$$i_r = i_c$$

In addition, the current through the capacitor is given by the capacitor law:

$$i_c = C \frac{dV_C}{dt}$$

Combining this and the last equation allows us to reexpress V_R as:

$$V_R = RC \frac{dV_C}{dt}$$

Substituting this into $V = V_R + V_C$ and rearranging yields the result we seek:

$$\frac{dV_C}{dt} = \frac{1}{RC}(V - V_C)$$

This is a first-order differential equation with initial condition, $V_C(0) = 0$.

One of the interesting applications of electrical circuits is to carry out computations. The following exercises will illustrate some simple computations that can be carried out by simple resistive circuits. Answers can be found at the end of the chapter.

Class Exercise 5

Given the resistor network shown in Figure 1.7, determine the voltage, v_0 .

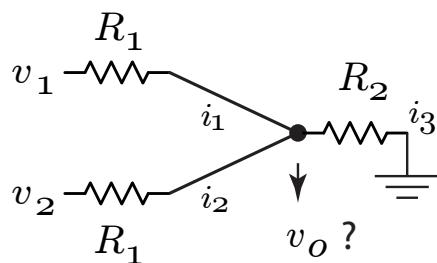


Figure 1.7 Resistor Network.

When you've derived the relation for v_0 , assume $R_2 \gg R_1$ and then infer what calculation could be done with the circuit if v_1 and v_2 are the inputs and v_0 the output.

Class Exercise 6

Show that the resistive circuit in Figure 1.8 can be used to compute a voltage times a constant.

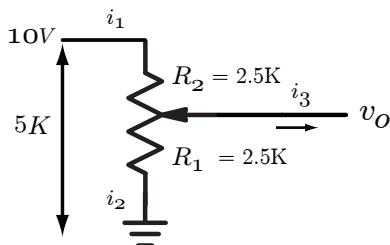


Figure 1.8 Circuit to multiply an input voltage by a constant.

Box 1.0 Review of Basic Electricity.

v = Voltage i = Current R = Resistance

C = Capacitance q = Charge

Ohm's Law: $v = R i$

Capacitor Law: $q = C v$ or $v = \frac{1}{C} q$

In addition, the charge on a capacitor is given by:

$$q = \int i dt$$

Combining the two previous relations and differentiating yields:

$$i = C \frac{dv}{dt} \quad \text{or} \quad \frac{dv}{dt} = \frac{1}{C} i \quad \text{or} \quad v = \frac{1}{C} \int i dt$$

Resistors in series/parallel:

In series: $R = \sum R_i$

In parallel: $1/R = \sum 1/R_i$

Capacitors in series/parallel:

In series: $1/C = \sum 1/C_i$

In parallel: $C = \sum C_i$

Kirchhoff's Laws - KCL and KVL

KCL: At a node $\sum i_{in} = \sum i_{out}$ or $\sum i_k = 0$

KVL: Around a cycle $\sum v_k = 0$

Class Exercise 7

What is the problem with all these circuits that could make them potentially poor at computing?

A detailed answer to Question 7 is given at the end of the chapter. In summary, a problem arises if the downstream circuit that is connected to the output, v_o , draws too much current from the resistor network. If this happens, the output voltage is affected which means that the calculation incurs significant error. As a result using simple resistor networks to carry out calculations is not always practical. We will call this the loading issue.

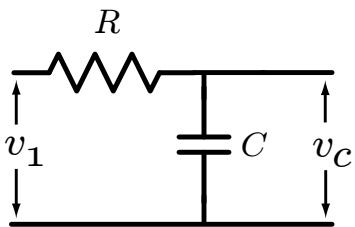
There are many other kinds of circuits one can build using passive components. Two of the most common are the integrator and differentiator circuits (Figure 1.9). Integrators carry out the mathematical operation:

$$\int v dt$$

while differentiators carry out the operation:

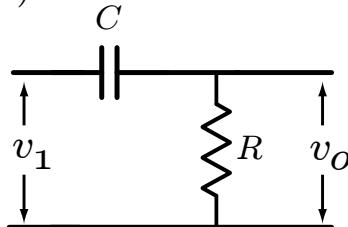
$$\frac{dv}{dt}$$

a)



Integrator

b)



Differentiator

Figure 1.9 Integrator and differentiator circuits.

Integrator Consider first the integrator circuit. Recall that the output voltage, v_C is the integral of the current:

$$v_C = \frac{1}{C} \int i dt$$

However we need integration to occur with respect to the input voltage. The easiest way to do this is to add a resistor R which will convert the input voltage v_1 into a current. According to Kirchhoff's voltage law, the voltage $v_1 = v_R + v_C$. Let us make the resistance on R large; this means that any

current flow through R will be small and this in turn means that the voltage drop across R_1 , v_R , will be small. This means that the voltage at the junction between R and C will be roughly equal to v_1 ($v_1 = v_R + v_C$). By Ohm's law, $i = v_1/R$, and inserting this into the integral given above yields;

$$v_C = \frac{1}{RC} \int v_1 dt$$

That is, v_C is the integral of the input voltage, v_1 .

Differentiator For the differentiator we apply similar reasoning except in reverse. The capacitor, via $i = Cdv/dt$, ensures that the current is the derivative of the input voltage. However we want the output voltage to reflect the derivative not the current. We therefore add a resistor to convert the current i into the output voltage v_o . Since v_o is proportional to the current, the output voltage is proportional to the derivative of the input voltage.

To find the proportionality constant, we assume that the resistance R is small. This means the voltage drop across R will also be large ($v_o \approx 0$) so that the voltage across the capacitor, v_C will roughly equal to v_1 ($v_1 = v_o + v_C$). The voltage across R will equal: $v_o = iR$, but the current, i is given by the capacitor law:

$$i = C \frac{dv_C}{dt}$$

Since we've assumed that $v_C = v_1$, then:

$$i = C \frac{dv_1}{dt}$$

Substituting in Ohm's law, $i = v_o/R$ we finally obtain:

$$v_o = RC \frac{dv_1}{dt}$$

That is, the output voltage is equal to the derivative of the input voltage via the constant RC .

As with the summer and multiplier circuits these passive circuits suffer from loading issues and are not practical as computational circuits.

Avoiding the Loading Issue: Op Amps

One way to avoid the loading issue described in the last section is to use some kind of buffer circuit such as an **Op Amp**. Op Amps, or operational amplifiers, are the workhorse of many analog circuits. They can be purchased very cheaply with varying characteristics to suit a myriad of different applications.

In its basic form, an op amp has five terminals. Two of these terminals are used to supply power to the circuit and are not normally shown when drawing op amp circuits. For the most common op amps, the

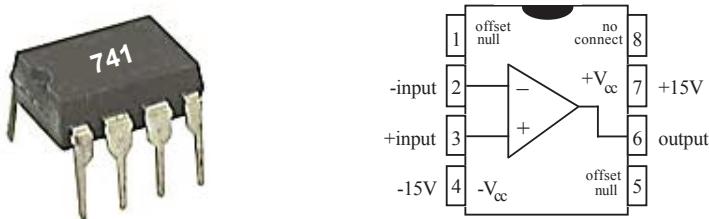


Figure 1.10 Common 741 Op Amp. Image from Jameco Electronics, cost \$0.25 each.

positive supply voltage is often +15 volts, and the negative supply voltage -15 volts. The remaining three terminals represent two input and one output terminal. The two input terminals are designated the inverting and non-inverting terminals, respectively. In diagrams the two input terminals are often indicated with a plus and a minus sign (Figure 1.11)

Op amps are readily available from electronic stores and the most popular, the 741, is priced at around 25 cents per unit.

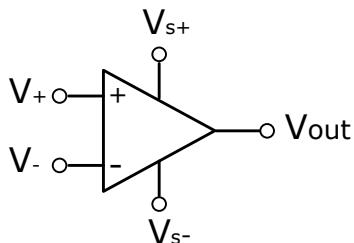


Figure 1.11 Symbol for an Op Amp. V_{s+} and V_{s-} are the positive and negative supply voltages. V_+ and V_- are the inverting and non-inverting inputs.

Basic Properties of an Op Amp

An op amp amplifies the difference in voltage between the positive and negative input terminals, that is:

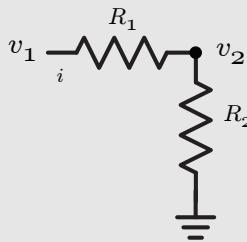
$$v_o = A(v_+ - v_-)$$

where A is the so-called gain of the amplifier. Figure 1.12 shows the basic response of an op amp. Note that the saturation point is where the output reaches the supply voltage. In real op amps the gain is usually of the order of 10^5 . For an op amp operating with a supply voltage of +15/-15 volts, the difference in voltage between the two input terminals cannot be much more than $150 \mu V$. If the input difference does exceed $150 \mu V$, then the output of the amplifier will saturate, meaning it will reach the supply voltage. Given a gain of 10^5 , a voltage difference of 0.00015 volts will give an output

voltage of 15 volts. This means that an op amp on its own is of little practical value because the slightest difference in voltage at the input terminals causes the op amp to saturate. When applied to real problems, op amps are almost always used with some kind of feedback circuit to avoid this issue.

Box 1.2 Effect of Low Output Impedance

Consider the circuit:



where v_2 is the output voltage and R_2 a load on the system. The two voltage drops across the circuit can be described by:

$$v_1 - v_2 = iR_1 \quad v_2 = iR_2$$

Combining the two equations and solving for v_2 yields:

$$v_2 = \frac{v_1}{1 + \frac{R_1}{R_2}}$$

That is when $R_1 \ll R_2$:

$$v_2 \approx v_1$$

In other words, if the output resistance R_1 is low, the output voltage at v_2 is unaffected by the load, R_2 .

Example 1.1

An op amp has a gain of 100,000. In each of the following situations, indicate the output voltage:

a) $V_+ = 4\mu V, V_- = 2\mu V$

$$V_{\text{out}} = 100000 \times (4 - 2) = 0.2V$$

b) $V_+ = 2\mu V, V_- = 6\mu V$

$$V_{\text{out}} = 100000 \times (2 - 6) = -0.4V$$

c) $V_+ = 4\mu V, V_- = 4\mu V$

$$V_{\text{out}} = 100000 \times (4 - 4) = 0V$$

Although the basic op amp is essentially a very high-gain amplifier, it has certain other properties which make it extremely useful. These properties are:

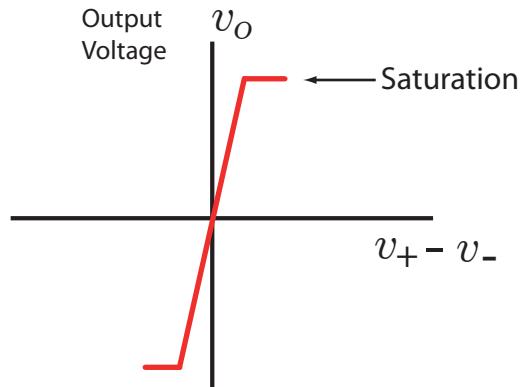


Figure 1.12 Basic Op Amp Response.

1. The resistance (or impedance) into the input terminals is extremely high.
2. The resistance (or impedance) of the output terminal is extremely low.

The high impedance on the input terminals means than an op amp will impose hardly any load on the circuit it is connected to. An op amp connected to a signal source will have little effect on the signal voltage so that the full strength of the signal can be fed into the op amp. The input resistance will often be of the order of a least a mega ohm, often more.

A low impedance on the output terminal of the order of 70 ohms or less means that attaching loads to the op amp will have hardly any effect on the output voltage (See Box 1.1). As a result of these properties, op amps are ideal components to use as buffering circuits for the computational circuits described in the previous section.

Summary of Op Amp Properties:

- $v_o = A(v_+ - v_-)$
- High gain
- High input impedance
- Low output impedance

Table 1.5 lists some characteristics for the common 741 op amp.

Input resistance	$2 \text{ M}\Omega$
Output resistance	75Ω
Gain	200,000

Table 1.5 Electrical characteristics for the common 741 op amp.

Negative Feedback and Op Amps

On its own an op amp is not very useful because the gain is so large that it only requires very small differences between the input terminals to saturate the output. In practice op amps are always combined with some kind of feedback.

To illustrate a simple example, consider the non-inverting amplifier shown in Figure 1.13. From the circuit we can determine the currents, i_1 and i_2 from the voltage drop across, R_1 and R_2 , that is:

$$i_1 = \frac{v_F - 0}{R_1} \quad i_2 = \frac{v_o - v_F}{R_2}$$

Note that the ground voltage is 0. In an ideal op amp, the amount of current flowing into the positive and negative inputs is zero, owing to the high input impedance. This means that the two currents, i_1 and i_2 must be equal, therefore it must be true that:

$$\frac{v_F}{R_1} = \frac{v_o - v_F}{R_2} \Rightarrow \frac{v_F}{R_1} + \frac{v_F - v_o}{R_2} = 0 \quad (1.7)$$

Another property of an op amp is that the output voltage is equal to the gain, A , times the difference in the input voltage:

$$v_o = A(v_I - v_F)$$

Rearranging this yields:

$$v_F = v_I - \frac{v_o}{A} \quad (1.8)$$

Inserting v_F into the current equation (1.7) gives:

$$\frac{v_I - \frac{v_o}{A}}{R_1} + \frac{v_I - \frac{v_o}{A} - v_o}{R_2} = 0$$

Since the gain, A , of an op amp is very large, ideally infinite, the above equation simplifies to:

$$\frac{v_I}{R_1} + \frac{v_I - v_o}{R_2} = 0$$

This can now be rearranged to solve for v_O :

$$v_o = v_I \left(1 + \frac{R_2}{R_1} \right)$$

The remarkable thing about this result is that the gain of the op amp, A , has vanished. This is an important effect because it means that the performance of the circuit is only dependent on the two resistors, R_1 and R_2 . Any variation in the op amp itself is eliminated. So long as the resistors are high quality, this circuit is immune to variation in the characteristics of the op amp and to environmental variation such as temperature changes.

Computationally the circuit takes an input voltage, v_I , and multiplies it by a scaling factor. The only possible disadvantage to the non-inverting circuit is that the scaling will be greater than one; it is not possible to scale v_I by a factor < 1 . The circuit does however make an excellent signal amplifier.

More importantly, given the high input and low output impedance, the circuit doesn't suffer the loading issues of a simple resistive circuit. Combining op amps with resistive/capacitive circuits is a common way to make analog computational modules that can be easily connected without the modules interfering with each other.

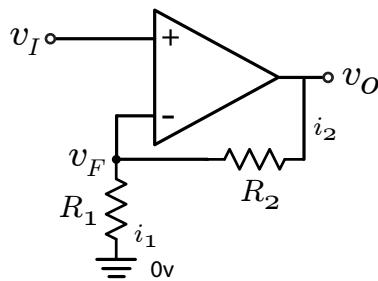


Figure 1.13 Non-Inverting Op Amp Circuit.

Equation 1.8 is of interest because it helps understand the operation of the circuit more intuitively. Given that the gain, A , of the op amp is huge, we can approximate equation 1.8 to:

$$v_I = v_F$$

This relationship tells us that whatever the input voltage, v_I is, v_F will match it. In order for this to happen, as the input voltage v_I changes the op amp must be going through a dynamic change so that v_F matches any change in v_I . The explanation is simple but crucial to an intuitive understanding of the circuit.

Recall that an op amp is a difference amplifier. Imagine we increase v_I above v_F . This means the difference ($v_I - v_F$) increases, which in turn is amplified resulting in an increase in v_O . Given the high gain of the op amp, the difference is likely to cause v_o to increase significantly. As v_o increases part of it is passed back via resistor R_2 to v_F so that v_F **increases**. If v_F increases, the difference ($v_I - v_F$) decreases, which in turn reduces the rise in v_o . This feedback cycle continues until v_F converges to v_I at which point the circuit reaches a steady state, where v_O stops changing and settles to a voltage that maintains $v_I = v_F$.

Figure 1.14 shows three computational circuits that use an op amp: a summer, an integrator and a differentiator.

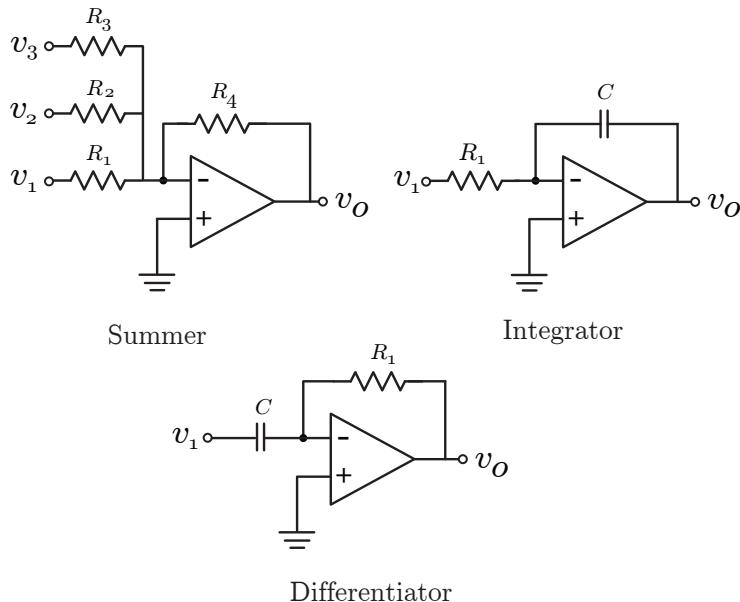


Figure 1.14 Three different computational circuits made from op amps. Note the similarity with the non-op amp circuits in Figures 1.7 and 1.9.

1.7 Block Diagrams

In many textbooks, block diagrams are used to illustrate a model. We've already seen that electrical circuits can be used to perform various computations and these can be represented in the form of block symbols. Figure 1.15 shows some common block symbols found in the literature. Using these symbols, let us draw a block diagram that represents the differential equation and its solution:

$$\frac{dx_1}{dt} = -kx_1$$

Solving a differential equation involves integration. Therefore, the first block to consider is the integral block. The input to this block will be the derivative, dx_1/dt , and the output will be x_1 . However x_1 has a negative term in the differential equation, $-kx_1$, therefore we will also invert x_1 at the same time. Once we have $-x_1$, we only need to add a ‘multiply by the constant k ’ to obtain

the value of dx_1/dt . This final operation closes the loop. Figure 1.16 illustrates the complete block diagram.

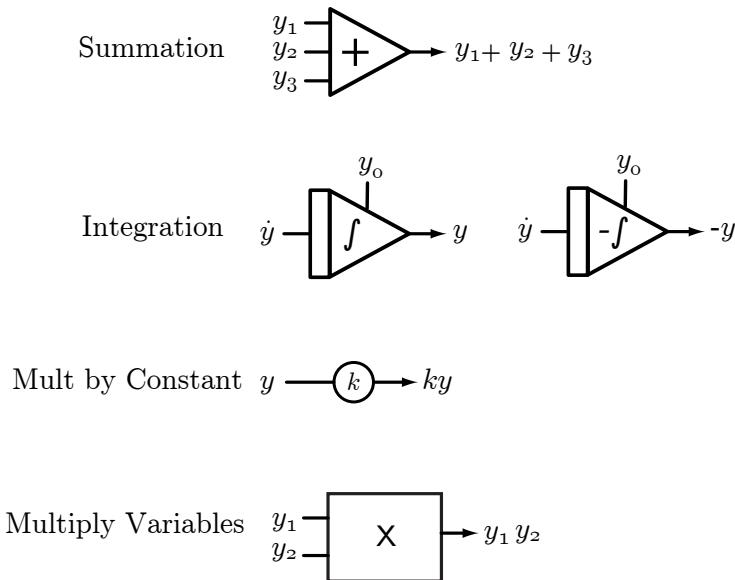


Figure 1.15 Common Block Symbols.

From the block diagram we can construct an op amps device such that the electrical circuit solves the differential equation. Before the advent of the digital computer, such ‘analog’ computers were commonplace and were used to solve a variety of problems in science and engineering. One of the advantages of an analog computer is that they are very fast and can be operated in a mode such that it is possible to interact with the model in real time; the analog computer automatically adjusts to the new solution.

With the advent of fast and mass-produced and therefore cheap digital computers, the need for analog computers has declined although there is recent interest in a new field called Neuromorphic electronics [4, 61]. The aim of this emerging field is to recreate brain-like computational systems from electrical analog circuitry.

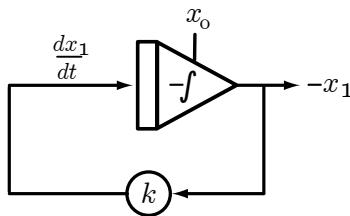
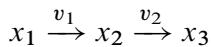


Figure 1.16 Analog System to Solve, $\frac{dx_1}{dt} = -kx_1$.

1.8 Analytical Solutions

Consider the following two consecutive chemical reactions involving three chemical species, x_1 , x_2 and x_3 .



Assume each chemical reaction is governed by simple mass action kinetics so that the reaction rates, v_1 and v_2 are given by:

$$v_1 = k_1 x_1 \quad v_2 = k_2 x_2$$

We can write a simple mathematical model to describe the time evolution of the three species by invoking the conservation of mass. That is:

$$\frac{dx_1}{dt} = -k_1 x_1$$

$$\frac{dx_2}{dt} = k_1 x_1 - k_2 x_2$$

$$\frac{dx_3}{dt} = k_2 x_2$$

In this case the model is simple enough that we can derive the analytical solution. Assuming that at

time zero, both x_2 and x_3 are at zero concentration; it can be shown that:

$$x_1(t) = x_1(0) e^{-k_1 t}$$

$$x_2(t) = \frac{k_1}{k_2 - k_1} x_1(0) \left(e^{-k_1 t} - e^{-k_2 t} \right)$$

$$x_3(t) = x_1(0) \left(1 - \frac{k_2 e^{-k_1 t} - k_1 e^{-k_2 t}}{k_2 - k_1} \right)$$

In the majority of cases it is not possible to derive such an analytical solution and we must therefore fall back to obtaining numerical solutions. To do this we must find a suitable software application that can solve differential equations. The plot shown in Figure 1.17 was obtained using Python (tellurium.analoglachine.org), but similar plots can be obtained using Matlab.

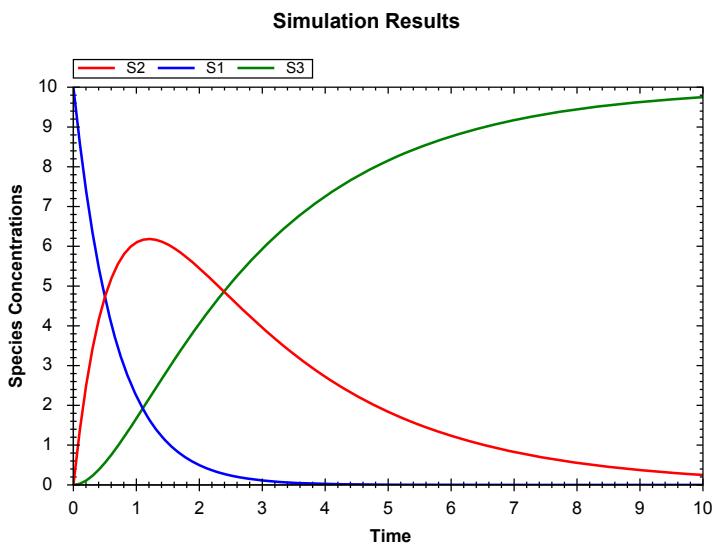


Figure 1.17 Computer Simulation of two Consecutive Chemical Reactions. $x_1(0) = 10$, $k_1 = 1.5$; $k_2 = 0.4$

Assignment 1

Obtain the same plot shown in Figure 1.17 using Matlab or Python.

1.9 Dimensions and Units

Variables and parameters that go into a model will be expressed in some standard of measurement. In science the recognized standard for units are the SI units. These include units such as the *meter* for length, *kilogram* for mass, *second* for time, *Joules* for energy, *kelvin* for temperature and the *mole* for amount. The mole is of particular importance because it is a means to measure the number of particles of substance irrespective of the mass of substance itself. Thus 1 mole of glucose is the same amount as 1 mole of the enzyme glucose-6-phosphate isomerase even though the mass of each molecule is quite different. The actual number of particles in 1 mole is defined as the number of atoms in 12 grams of carbon-12 which has been determined empirically to be 6.0221415×10^{23} . This definition means that 1 mole of substance will have a mass equal to the molecular weight of the substance. This makes it easy to calculate the number of moles using the following relation:

$$\text{moles} = \frac{\text{mass}}{\text{molecular weight}}$$

The concentration of a substance is expressed in moles per unit volume and is usually termed molarity. Thus a 1 molar solution means 1 mol of substance in 1 litre of volume.

Dimensional Analysis

Dimensional analysis is a simple but effective method for uncovering mistakes when formulating kinetic models. This is particularly true for concrete models where one is dealing with actual quantities and kinetic constants. Conceptual models are more forgiving and don't usually require the same level of attention because they tend to be simpler.

Amounts of substance is usually expressed in moles and concentrations in moles per unit volume ($\text{mol } l^{-1}$). Reaction rates can be expressed either in concentrations or amounts per unit time depending on the context ($\text{mol } t^{-1}$, $\text{mol } l^{-1} t^{-1}$).

Rate constants are expressed in differing units depending on the form of the rate law. The rate constants in simple first order kinetics are expressed in per unit time (t^{-1}), while in second order reactions the rate constant is expressed per concentration per unit time ($\text{mol}^{-1} t^{-1}$).

In dimensional analysis, units on the left and right-hand sides of expressions must have the same units (or dimensions). There are certain rules for combining units when checking consistency in units. Only like units can be added or subtracted, thus the expression $S + k_1$ cannot be summed because the units of S are likely to be $\text{mol } l^{-1}$ and the units for k_1 , t^{-1} . Even something as innocent looking as $1 + S$ can be troublesome because S has units of concentration but the constant value '1' is unit less. Quantities with different units can be multiplied or divided with the units for the overall expression computed using the laws of exponents and treating the unit symbols as variables.

Class Exercise 8

Determine the overall units for the expression, $k_1 S/K_m$, where the units for each variable are $k_1(t^{-1} l)$, $S(\text{mol } l^{-1})$ and $K_m(\text{mol } l^{-1})$.

In exponentials such as e^x , the exponent term must be dimensionless, or at least the expression should resolve to dimensionless. Thus given this, e^{kt} is permissible but e^k is not, if for example, k is a first-order rate constant. Trigonometric functions will always resolve to dimensionless quantities because the argument will be an angle, which can always be expressed as a ratio of lengths which will by necessity have the same dimension.

1.10 Approximations

By their very nature, models involve making assumptions and approximations. The best modelers are those who can make the most shrewd and reasonable approximations without compromising a model's usefulness. There are however some kinds of approximations which are useful in most problems:

- Neglect the small effects
- Assume the system environment is unchanged by the system itself
- Replace complex subsystems with lumped or aggregate laws
- Assume simple linear cause-effect relationships where possible
- Assume the physical characteristics of the system do not change with time
- Neglecting noise and uncertainty.

Neglecting small effects. This is the most common approximation to make. In many studies there will always be parts of the system that have a negligible effect on the properties of the system, at least during the period of study. For example, the rotation of the earth, the cycle of the moon or the rising and setting of the sun will most likely have negligible influence on our system. Assuming of course we are not studying circadian rhythms.

Assume that the system environment is unchanged by the system itself. This is a basic assumption in any study. The minute a system starts to affect the environment we have effectively extended the system boundaries to include more of the environment. Oftentimes the interface between the environment and the system will not be perfect so that there will be some effect that the system has on the environment. So long as this effect is small, we can assume that the environment is not affected by the system.

Replace complex subsystems with lumped or aggregate laws. Lumping subsystems is a commonly used technique in simplifying cellular models. The most important of these is the use of aggregate

rate laws such as Michaelis-Menten or Hill like equations to model cooperativity. Sometimes entire sequences of reactions can be replaced with a single rate law.

Assume simple linear cause-effect relationships. In some cases it is possible to assume a linear cause and effect between an enzyme reaction rate and the substrate concentration. This is especially true when the substrate concentration is below the K_m of the enzyme. Linear approximations make it much easier to understand a model.

Physical characteristics do not change with time. A modeler will often assume that the physical characteristics of a system do not change. For example, if we consider the volume of a cell, the values of the rate constants or the temperature of the system. In many cases such approximations are perfectly reasonable.

Neglecting noise and uncertainty. Most models make two important approximations. The first is that noise in the system is either negligible or unimportant. In many non-biological systems such an approximation is considered reasonable. However biological systems are particulate in nature and operate at the molecular level. As a result, biological systems are susceptible to noise generated from thermal effects as a result of molecular collisions. For many systems the large number of particles ensures that the noise generated in this way is insignificant. In these cases noise can be safely ignored. For other systems such as prokaryotic organisms, the number of particles can be very small. In this instance the effect of noise can be significant, and therefore must be included as part of the model. Thus the context of the question and the nature of the system must both be considered.

1.11 Behavior

We can classify the behavior of systems into three broad categories: thermodynamic equilibrium, steady state or transient. The steady state can be further classified into stable or unstable steady states. For example, if we pick biochemical systems as our example then:

Thermodynamic Equilibrium In this state the concentrations of reactants and products do not show change over time; in addition, the rates of all forward and reverse reactions are equal. This means that at thermodynamic equilibrium there is no net movement of mass from one part of the system to another and no net dissipation of energy. In thermodynamics, equilibrium is the state that maximizes the entropy of the system.

Steady State At steady state the concentrations of reactants and products show no net change over time; however, unlike thermodynamic equilibrium, there is a net flow of mass or energy between the system and the environment. At steady state the system will continually dissipate entropy to the external environment while the entropy level of the system itself remains constant.

Transient State Under a transient state, a system will be moving from either one steady state to

another, or from a steady state to thermodynamic equilibrium.

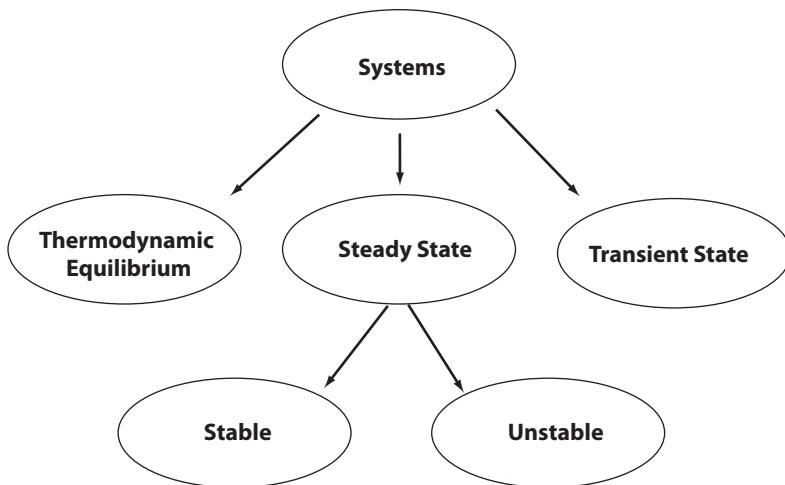


Figure 1.18 Classification of System Behavior.

Exercises

1. Watch the YouTube videos: RC Integrator Circuit¹ and RC Differentiator Circuit² (Figure 1.9).
2. Using the same approach to what was used in the non-inverting amplifier (Figure 1.13), derive the relationship between the input and output voltage for the inverting amplifier.
 - a) Show that the inverting amplifier behaves as a voltage scaler while at the same time inverting the signal.
 - b) Show that the voltage at v_F is zero. Using the explanation that $v_I = v_F$ for the non-inverting op amp, explain why v_F is zero in an inverting op amp.

¹<http://www.youtube.com/watch?v=JbpR5nGu0Gg>

²<http://www.youtube.com/watch?v=qtrYd0uJzyA>

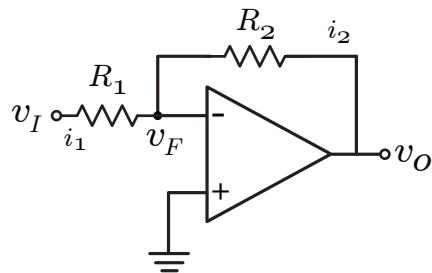


Figure 1.19 Inverting Op Amp Circuit.

3. Draw a block diagram of the following system of two differential equations.

$$\frac{dx_1}{dt} = -k_1 x_1$$

$$\frac{dx_2}{dt} = k_1 x_1 - x_2$$

4. Suggest state variables, inputs, and parameters in the following systems:

- a) A projectile fired from a cannon (Figure 1.20)

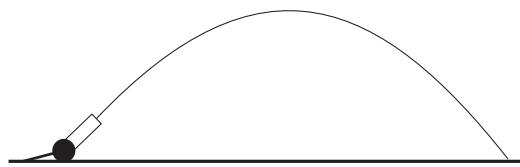


Figure 1.20 Projectile fired from a cannon.

- b) An RC circuit acting as a low pass filter (Figure 1.21)

- c) An ecosystem comprising of a population of rabbits and foxes where the model is based on the Lotka-Volterra system of equations.

- d) A chemical system based on the Brusselator model.

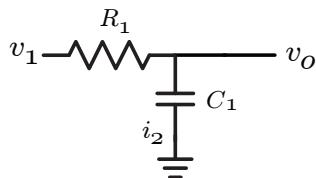


Figure 1.21 RC Circuit.

5. Predator/Prey Models

- a) Build a computer model of the predator/prey system described by the Lotka-Volterra system of equations:

$$\frac{d\text{Prey}}{dt} = \text{Prey } \alpha - \beta \text{ Prey Predator}$$

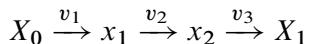
$$\frac{d\text{Predator}}{dt} = \delta \text{ Prey Predator} - \gamma \text{ Predator}$$

where α is the growth rate of the prey, β is the rate of predation, γ is the death rate of the predator and δ is the rate of growth in the predator population.

- b) Explore how the parameters affect the evolution of the populations. The parameter can range between 0.1 and 2.0. Set the initial population of prey and predators to 1 each.

6. Biochemical Models

Build a computer model of the following simple metabolic pathway:



Assume that the metabolites, X_o and X_1 are fixed. Assume also that v_1 to v_3 are governed by the rate laws:

$$v_1 = E_1(k_1 X_o - k_2 S x_1)$$

$$v_2 = E_2(k_3 x_1 - k_4 x_2)$$

$$v_3 = E_3 * k_5 * x_2$$

where E_i is the concentration of the i^{th} enzyme and k_i are the kinetic rate constants.

- Write out the state variables, suggested inputs, and parameters of the system.
- Write out the differential equations for this system.
- Assign arbitrary but realistic values to the various parameters in the model and compute the steady state, that is when $dx_i/dt = 0$. Report the steady state concentrations of x_1 and x_2 .
- What is the effect of doubling all E_i by a factor α on the steady state concentrations of x_1 and x_2 ?

7. Pharmokinetic Model

Figure 1.22 shows a simple compartmental model of what happens to a drug when it is injected intravenously into a patient. The model comprises of three compartments, the blood stream, the liver and the actual site of action. It is assumed that the drug can freely exchange between the liver and blood compartments but that the liver can also irreversible degrade the drug. In addition, some drug is lost by excretion through the kidneys. Finally, some drug accumulates irreversibly at the site of action.

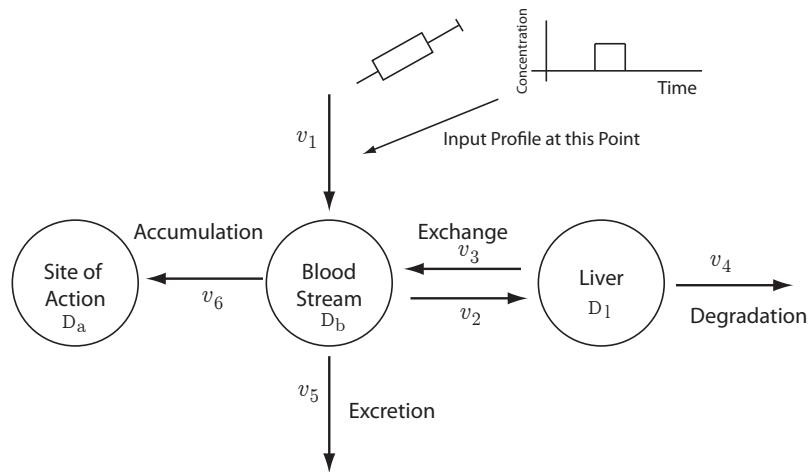


Figure 1.22 Simple model of a drug begin injected and distributed through the body.

Assume that movement of the drug between compartments obeys simple first-order kinetics. For example, the rate v_2 at which the drug enters the liver is given by $k_2 D_b$ where D_b , is the concentration of drug in the blood stream, and k_2 is the kinetic constant for the process. The

complete list of rates is:

$$v_2 = k_2 D_b$$

$$v_3 = k_3 D_l$$

$$v_4 = k_4 D_l$$

$$v_5 = k_5 D_b$$

$$v_6 = k_6 D_b$$

where D_b is the concentration of drug in the blood stream, D_l the concentration of drug in the liver, and D_a the concentration of drug at the site of action. Using the conservation of mass, write out three differential equations that describe the rate of change of drug in each of the three compartments.

Previous pharmokinetic studies of the drug indicate that the rate constants for the entry and exit of drug to and from the various compartments is given by: $k_2 = 6.12$, $k_3 = 0.2$, $k_4 = 0.45$, $k_5 = 1$, $k_6 = 5$.

All units in per second. Using a simulation model, answer the following questions.

a) Assume the nurse injects 1 mM of drug per second into the patient's blood stream. That is, in one second the concentration of drug in the blood stream increases by 1 mM. This is another way of saying that $v_1 = 1 \text{ mM sec}^{-1}$. The nurse can start and stop the injection very quickly such that the profile one observes in the rate at which drug enters the blood looks like a pulse as shown in Figure 1.23.

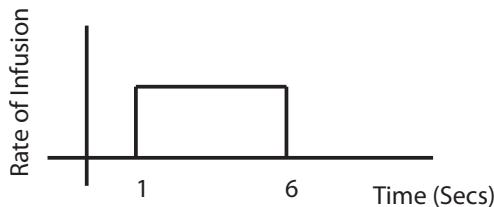


Figure 1.23 Drug Infusion Profile.

Assume that in order for the drug to be effective the concentration at the site of action must reach at least 2.5 mM. Use the computer model to tell the nurse how long she should inject for.

b) The model has a number of assumptions, for example, that mixing of drug is instantaneous in each compartment, that we've accounted for all the possible compartments and that the rate constants are accurate. But there is one particularly bad assumption which makes the model and its predictions highly suspect. What might that be?

2

Simulation Software

“Cogito ergo sum” – I think therefore I am

– René Descartes, 1664

Let's now turn to the use of programmable digital computers for modeling physical systems. Modern digital computers are programmable machines that can manipulate binary numbers. As such, they are very adept at handling problems involving numerical processing. There are a great variety of computer software packages that have been written specifically to solve ordinary differential equations both algebraically and numerically. In this brief chapter we will outline examples using Matlab, Mathematica and Python. A much fuller account of numerical methods for solving differential equations can be found in Appendix B. In addition, a more detailed description of Python can be found in Appendix C.

2.1 Software for Solving ODEs

All the models described in this book will be based on ordinary differential equations. For this reason this chapter will briefly describe some software tools that are available for solving differential equations. There are two categories of software to consider: open source and proprietary commercial tools. Of the commercial tools Matlab¹ and Mathematica² are well known so we will address these first.

¹<http://www.mathworks.com>

²<http://www.wolfram.com>

Using Matlab

Matlab is a numerical computing environment with significant strengths in matrix manipulation and plotting. Matlab offers a range of solvers to solve ordinary differential equations. The two most common solvers are `ode45` and `ode15s`.

The `ode45` solver implements a variable step size Runge-Kutta method. The variable step size is achieved by comparing solutions using a forth and fifth order Runge-Kutta. If the error between the two methods is too big, then the step size is reduced. Otherwise, if the step size is below a given threshold, the program will attempt to increase the step size. This method is also called the Dormand-Prince method [12]. The basic syntax for `ode45` is:

```
[t,y] = ode45(@myModel, [t0, tend], yo, [], p);
```

where

`myModel` is the function containing the differential equations.

`t0, tend` are the initial and final values for the independent variable, t .

`yo` is a vector of initial conditions.

`p` is the set of parameters for the model, this can be any size.

The empty vector, `[]` in the call is where additional options peculiar to `ode45` can be included.

For example, consider the set of ODEs:

$$\frac{dy_1}{dt} = v_o - k_1 y_1$$

$$\frac{dy_2}{dt} = k_1 y_1 - k_2 y_2$$

Listing 2.1 shows a simple Matlab based script for defining the set of ODEs and solving them.

```
function dy = myModel(t, y, p)
    dy = zeros (2,1);
    vo = p(1);
    k1 = p(2);
    k2 = p(3);
    dy(1) = vo - k1 y(1);
    dy(2) = k1 y(1) - k2 y(2);
end

% Parameter values
p = [10, 0.5, 0.35]

% Initial conditions
y0 = [0, 0]
```

```
% Call the ode solver  
[t, y] = ode45 (@myModel, [0, 20], y0, [], p)  
  
% Plot the results  
plot(t,x(1), t,x(2))  
legend ('y1', 'y2')
```

Listing 2.1 A Matlab script for solving a simple set of differential equations and plotting the results.

The alternative to `ode45` is `ode15s`. Although many problems can be solved using `ode45`, for models that have widely differing time scales (called stiff systems), `ode45` will fail to give the correct solution. In these cases `ode15s` is recommended. `ode15s` is a variable order solver and amongst other things, it uses a well known method called Gear's method [20]. Like `ode45`, `ode15s` is also a variable step size method. `ode45` might be faster than `ode15s` on simple problems, but with today's fast computers the difference is hardly noticeable. Therefore one might consider using `ode15s` for all problems. When writing code `ode15s` is used in the same way as `ode45`.

Mathematica

Mathematica is another commercial tool whose particular strength is algebraic manipulation. Listing 2.1 shows the same simulation described in Figure 2.1 but solved using Mathematica instead.

Both Matlab and Mathematica are costly to purchase. Therefore many researchers have either written their own solvers or use publicly available software libraries.

2.2 Other Software

Although sometimes it may seem to be the case, Matlab isn't the only software that can be used to solve differential equations. For those who require more control or who are unable to purchase a commercial tool, there are many free applications and professionally developed open source software libraries. Octave (<http://www.gnu.org/software/octave/>) is an open source tool that uses a script language almost identical to Matlab. SciLab (<http://www.scilab.org/>) is another popular free Matlab like application. If you like programming in Python then Sage (<http://www.sagemath.org/index.html>) is an excellent option. A new free scientific programming language is Julia which offers a Matlab like syntax, but with much faster computation times (<http://julialang.org/>). There are therefore many alternatives to using commercial products.

For those who require much more control and higher performance than can be achieved by the tools mentioned above, the Sundials C/C++ library developed by the Department of Energy is an excellent choice. Within Sundials there is the CVODE library that is used by many of the commercial tools.

CVODE implements an advanced Gear like algorithm using a variable order and variable step size approach. It is highly suited for stiff systems and is the preferred method for those who need to write their own code. One final library worth mentioning is the GPL (GNU General Public License) licensed GSL library (<http://www.gnu.org/software/gsl/>). Although very comprehensive, the GPL license is restrictive in the sense that any software that uses GSL must also acquire the same GPL licence. This generally precludes GSL from commercial use due to the requirement that all GPL source code must be made public.

```
NDSolve[{3.4 - 2.3 x1[t] == x1'[t],
         2.3 x1[t] - 0.4 x2[t] == x2'[t],
         x1[0] == 0, x2[0] == 0}, {x1, x2}, {t, 0, 5}]

Plot[Evaluate[{x1[t], x2[t]} /. s], {t, 0, 5},
      PlotRange->All]
```

Figure 2.1 Mathematica script for solving a simple set of differential equations.

One of the more popular non-Matlab like tools for numerical analysis is Python (python.org). Python is a popular programming language that is designed to be easy to learn and has found widespread use in commercial, scientific and teaching environments. In the next section we will learn how to use Matlab and Python to solve a set of differential equations. A more detailed description of Python can be found in Appendix C.

Solving the Lorenz Attractor

In this section we will illustrate the use of Matlab and Python to solve the famous chaotic model called the Lorenz system. The Lorenz system is a three variable system of differential equations given by the following:

$$\begin{aligned}\frac{dx}{dt} &= -\sigma x + \sigma y \\ \frac{dy}{dt} &= \rho x - y - yz \\ \frac{dz}{dt} &= -\beta z + xy\end{aligned}$$

Matlab This example illustrates how Matlab can be used to solve the differential equations. To solve differential equations we must first declare a function that computes the right-hand sides of the differential equations. It then returns the derivatives to the caller. This is shown:

```
function xprime = lorenz(t,x);
% Computes the derivatives of
% the differential equations
sig=10;
beta=8/3;
rho=28;
xprime=[-sig*x(1) + sig*x(2);
    rho*x(1) - x(2) - x(1)*x(3);
    -beta*x(3) + x(1)*x(2)];
```

To solve the Lorenz system one of the Matlab provided ODE solver routines can be used. In the script below ode45 is used which implements an adaptive step-size Runge-Kutta method (See Appendix B).

```
>>x0 = [-8 8 27];
>>tspan = [0, 20];
>>[t,x] = ode45 (@lorenz, tspan, x0)
>>plot (t, x)
```

The first line sets up the initial conditions for the three variables, the second sets up the duration of the simulation, and the third line carries out the actual simulation. The last line plots the simulation results.

Python This example shows how we can use Python to solve the Lorenz system. It relies on the SciPy and NumPy packages. The SciPy package includes a wide variety of numerical methods for solving different kinds of problem. The first part of the script imports the required libraries:

```
import scipy
from scipy.integrate import odeint
import numpy
import matplotlib.pyplot as plt
```

The second part of the script defines the system of differential equations:

```
def lorenz(x, t):
    sigma = 10
    rho = 28
    beta = 8.0/3
    return [ sigma * (x[1] - x[0]),
        x[0] * (rho -x[2]) - x[1],
        x[0] * x[1] - beta* x[2]]
```

Note the indentation of the lines which is important for Python. Finally, we invoke the ODE solver and plot the resulting simulation data.

```
xInitial = [0, 1, 1.05]
t = numpy.arange(0, 30, 0.01)
lorenzSolution = scipy.integrate.odeint(lorenz, xInitial, t)

plt.plot (lorenzSolution)
plt.show()
```

2.3 Domain Specific Tools: Tellurium/Python

Tellurium

Tellurium [52] is an integrated Python based environment for modeling in systems biology. The current version (July 2015) integrates a number of libraries including libRoadRunner (Simulator), libSBML (SBML support), libAntimony (Antimony support) and SBML2Matlab (SBML to Matlab converter). In addition, Tellurium distributes a number of standard Python packages such as Matplotlib (plotting), NumPy (array support), and sciPy (numerical analysis). All packages are integrated using spyder2 (<https://code.google.com/p/spyderlib/>) which offers a Matlab-like experience for users.

Visually, Tellurium has two main windows (Figure C.1): a console where commands can be issued and results returned, and an editor where control scripts and models are written. The application also has a plotting window which is used when graphing commands are issued.

Box 2.1 Systems Biology Markup Language

The systems biology markup language (SBML) is a standard machine exchangeable language for describing cellular networks such as metabolic, signalling and gene regulatory networks [29]. A large number of tools, including Matlab and Mathematica (via add on packages) support SBML.

Tellurium uses Antimony to let users describe biochemical pathways and Python coupled with libRoadRunner to do simulations and other analyses. Models can also be imported or exported as SBML (See Box 2.1). Many other capabilities are offered through libRoadRunner including support for metabolic control analysis, structural analysis of networks, and stochastic simulation [51]. A more detailed description of Tellurium is given in Appendix C. The following code shows the model we used previously but now expressed using Tellurium:

```
import tellurium as te
# Define a simple linear chain of reactions
```

```
r = te.loada ('''  
    $Xo -> S1; k1*Xo - k2*S1;  
    S1 -> S2; k3*S1/(k4 + S1);  
    S2 -> $X1; k5*S2^2;  
# Initialize parameters  
Xo = 10;  
k1 = 0.6; k2 = 0.4;  
k3 = 3.4; k4 = 0.1;  
k5 = 1.2;'''')  
  
# Carry out simulation and plot results  
# Three arguments = timeStart, timeEnd, numberofPoints  
m = r.simulate (0, 10, 100);  
r.plot (m)
```

The first part of the code shows the model expressed using Antimony and loaded into libRoadRunner, while the second part show two commands to simulate and plot the results via libRoadRunner.

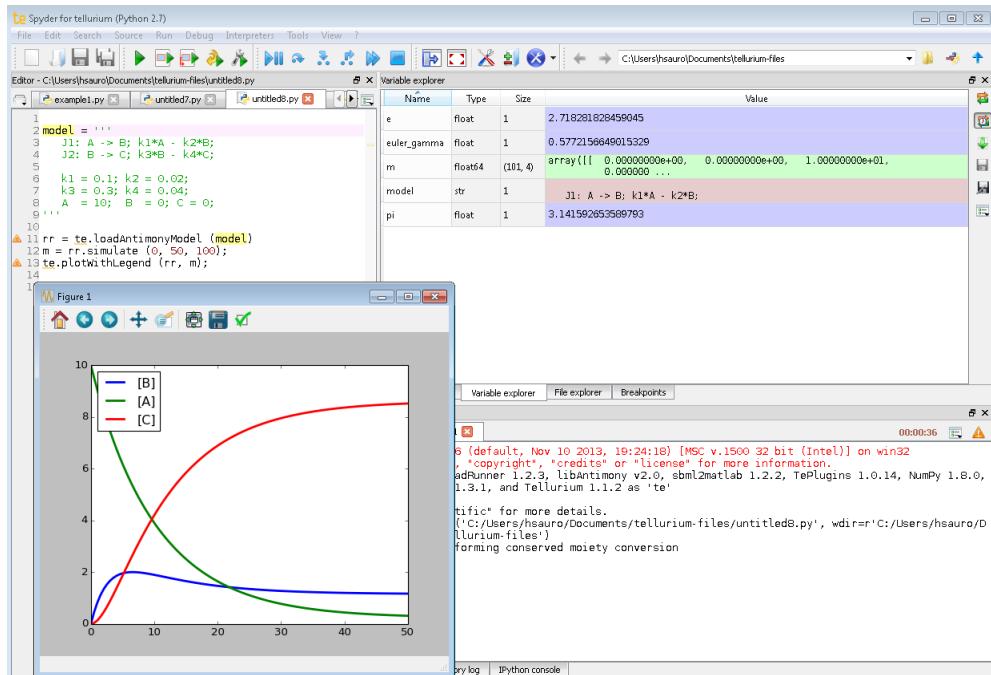


Figure 2.2 Screen shot of Tellurium with simulation results.

In addition to tools that can solve differential equations, there are also domain specific tools that allow a user to specify the physical problem in a way that will be more familiar to the user. From

this input, the software will derive the differential equations automatically. This helps reduce errors in the final equations. Examples of such tools include SPICE [77] for modeling electrical circuits, and Jarnac [49] or the Python based tool Tellurium for modeling biochemical networks (tellurium.analogmachine.org). In biochemical modeling there are also standard file formats for storing models. The most well known of these is the Systems Biology Markup Language (SBML) [29]. This means that applications that support SBML can access the large number of models available in model repositories such as Biomodels (<http://www.ebi.ac.uk/biomodels-main/>). Tellurium has the capability to convert SBML or Antimony models into Matlab. Many of the example models in this book will be described using Antimony scripts used in Tellurium which can be easily converted to SBML or Matlab. A simple example is given in the last section of appendix C.4.

Tellurium can be downloaded from <http://tellurium.analogmachine.org>. A more detailed description of Python and the tools associated with Tellurium can be found in Appendix C.

Further Reading

1. Numerical Recipes in C: The Art of Scientific Computing. Press WH, Flannery BP, Teukolsky, SA, and Vetterling WT, Second Edition, Cambridge University Press; 2 edition (October 30, 1992). I consider the 2nd edition to be superior to more recent editions especially in terms of the quality of the source code.

Appendix

1. Euler <http://euler.sourceforge.net/>
2. Julia scientific language <http://julialang.org/>
3. Python web site python.org
4. Python scientific library, scipy, <http://www.scipy.org/>
5. Scilab web site www.scilab.org
6. Tellurium web site <http://tellurium.analogmachine.org>.
7. Yorick <http://yorick.sourceforge.net/>

3

Introduction to Feedback Control

“He who controls the present, controls the past. He who controls the past, controls the future.”

— George Orwell, 1984

3.1 What is Control Theory?

Control theory in its broadest sense is about understanding how to influence the future behavior of a system using a regulatory mechanism. Controlled systems can be found in man-made and natural systems. Examples of man-made systems include temperature controlled rooms, landing of the curiosity probe on Mars, and autopilots. Biological systems employ control systems at many levels, for example glucose control via insulin/glucagon, body temperature control, intracellular homeostasis, protein signaling networks, gene regulation and more.

Systems that employ control mechanisms have a number of common features that make them useful. These include:

1. **Path Following.** The ability to follow a particular path, for example driverless cars, or the tracking of aircraft by radar or artillery.
2. **Stability.** The ability to maintain an unstable system in a stable state, for example, standing. Without stabilization control, an upright human would fall over.
3. **Disturbance Rejection.** Reduce the effect of external and often random influences. Homeostasis in biological systems is an example of disturbance rejection and is often imple-

mented using some kind of negative feedback system. The ability of an output signal (voltage, concentration) to withstand current or mass consumption is an example of disturbance rejection.

4. **Performance Improvement.** Faster response to some external cue, for example the time it takes to turn a gene on can be improved through the use of negative feedback.
5. **Robustness to System Variability.** A batch of resistors of a given resistance will display manufacturing variation between individual resistors. Likewise the concentration of a given biological protein will vary between cells of the same type. Control strategies can be employed to minimize the effect of natural variation in component properties.

Control Theory is a general body of knowledge and principles that can be applied to a huge variety of systems. One of its principle ideas is negative feedback and how negative feedback influences system dynamics. The historical roots of control theory lie in the early studies of using negative feedback in man-made machines.

Class Exercise 1

Design a water clock that can measure hours and minutes. The clock should be completely automatic and run continuously. You may use water, tanks, pipes, and levers. It must use a feedback mechanism of some kind.

3.2 Historical Perspective

Feedback is widespread in biochemical networks and physiological systems. Some form of feedback permeates almost every known biological process. On the face of it, feedback is a simple process that involves sending a portion of the output back to the input. If the portion sent back reduces the input then the feedback is called negative feedback. Otherwise, it is called positive feedback.

Negative feedback is where part of the output of a system is used to reduce the magnitude of the system input.

The concept of feedback control goes back at least as far as the Ancient Greeks. Of some concern to the ancient Greeks was the need for accurate time keeping, particularly in the law courts where water clocks were used to limit long speeches. In about 270 BC the Greek Ktesibios (also written as Ctesibius) (Tess-eeb-ius) invented a float regulator for a water clock. The role of the regulator was to keep the water level in a tank at a constant depth. This constant depth yielded a constant flow of water through a tube at the bottom of the tank which filled a second tank at a constant rate. The level of water in the second tank thus depended on time elapsed.

In 250 BC Philon of Byzantium is known to have kept a constant level of oil in a lamp using a float regulator and in the first century AD, Heron of Alexandria experimented with float regulators for water clocks. Philon and particularly Heron (13 AD) left us with a book (*Pneumatica*) detailing many amusing water devices that employed negative feedback.

It wasn't until the industrial revolution that feedback control, or devices for automatic control, became economically important. One of the most famous modern devices that employed negative feedback was the governor. Thomas Mead in 1787 took out a patent on a device that could regulate the speed of windmill sails. His idea was to measure the speed of the mill by the centrifugal motion of a revolving pendulum and then use it to regulate the position of the sail. Very shortly afterwards in early 1788, James Watt is told of this device in a letter from his partner, Matthew Boulton. Watt recognized the utility of the governor as a device to regulate the new steam engines that were rapidly becoming an important source of new power for the industrial revolution.

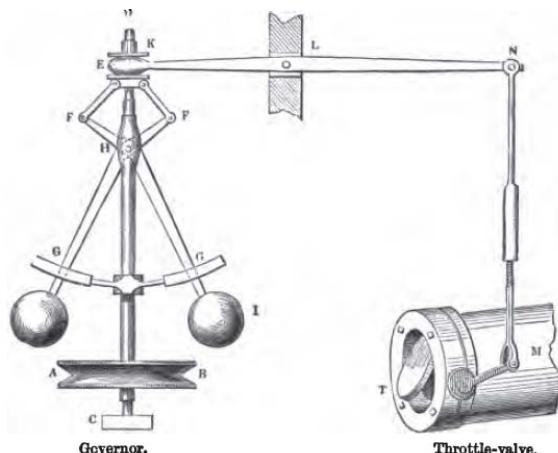


Figure 3.1 A typical governor from J. Farley, *A Treatise on the Steam Engine: Historical, Practical, and Descriptive* (London: Longman, Rees, Orme, Brown, and Green, 1827, p.436).

The novel device (Figure 3.1) employed two pivoted rotating fly-balls which were flung outward by centrifugal force. As the speed of rotation increased, the flyweights swung further out and up, operating a steam flow throttling valve which slowed the engine down. Thus, a constant engine speed was achieved automatically in the face of varying loads or steam pressure. So popular was this innovation that by 1868 it is estimated that 75,000 governors¹ were in operation in England. Many similar devices were subsequently invented to control a wide range of processes, including water wheels, telescope drives, and temperature and pressure controls.

The description of the governor illustrates the operational characteristics of **negative feedback**. The output of the device, in this case the steam engine speed, is “fed back” to control the rate of steam

¹A History of Control Engineering, 1800-1930 By Stuart Bennett, 1979

entering the steam engine and thus influencing the engine speed.

During this period devices for automatic control were designed through trial and error and little theory existed to understand the limits and behavior of feedback control systems. One of the difficulties with feedback control is the potential for instability. As the governor became more widespread, improvements were made in manufacturing mechanical devices which reduced friction. As a result engineers began to notice a phenomena they termed hunting. This was where, after a change in engine load, the governor would begin to ‘hunt’ in an oscillatory fashion for the new steam rate that would satisfy the load. This effect caused considerable problems with maintaining a stable engine speed and resulted in James Maxwell and independently Ivan Vyshnegradskii, undertaking the first theoretical analysis of a negative feedback system.

Until the 20th century, feedback control was generally used as a means to achieve automatic control, that is to ensure that a variable, such as a temperature or a pressure was maintained at some set value. However, entirely new applications for feedback control emerged early in the 20th century, these included artillery tracking on ships and communication. In naval warfare a major issue is being able to accurately fire a weapon from a moving ship to a moving target. In the early part of the 20th century mechanical analog computers called rangekeepers (<https://en.wikipedia.org/wiki/Rangekeeper>) were developed that could continuously compute a target bearing, predict the future target position and make adjustments to the weapon to account for other factors such as wind. Tracking uses negative feedback and when employed in this mode it is called a **servomechanism** or servo for short. Possibly the most important application of tracking came with the development of the feedback amplifier in the 1920s and 30s.

Feedback Amplifiers Amplification is one of the most fundamental tasks one can demand of an electrical circuit. One of the challenges facing engineers in the 1920’s was how to design amplifiers whose performance was robust to variation in the internal parameters of the system, external influences such as temperature and which could overcome inherent nonlinearities in the implementation. This problem was especially critical to the effort to implement long distance telephone lines across the USA.

These difficulties were overcome by the introduction of the feedback amplifier, designed in 1927 by Harold S. Black [39] who was an engineer for Western Electric (the forerunner of Bell Labs). The basic idea was to introduce a negative feedback loop from the output of the amplifier to its input. At first sight, the addition of negative feedback to an amplifier might seem counterproductive. Indeed Black had to contend with just such opinions when introducing the concept. His director at Western Electric dissuaded him from following up on the idea and his patent applications were at first dismissed. In his own words, “our patent application was treated in the same manner as one for a perpetual motion machine” [2].

While Black’s detractors were correct in insisting that the negative feedback would reduce the gain of the amplifier, they failed to appreciate his key insight that the reduction in gain is accompanied by increased robustness of the amplifier and improved fidelity of signal transfer.

Unlike the steam engine governor which is used to stabilize some system variable, negative feedback in amplifiers is used to accurately track an external signal. These two applications highlight the main ways in which negative feedback can be used, namely as a **regulator** or as a **servomechanism**.

Two modes of negative feedback:

Regulator: Maintain a given variable at a constant level, e.g. Thermostat

Servo: Track a reference input, e.g. Op amp acting as a voltage follower.

As a regulator, negative feedback is used to maintain a controlled output at some constant desired level, whereas a servomechanism will slavishly track a reference input. We can see both applications at work in the human eye. On the one hand there is the need to control the level of light entering the pupil where the diameter of the pupil is controlled by two antagonistic muscles. If the external light intensity increases, the muscles respond by reducing the pupil diameter, whereas the muscles increase the pupil diameter if the light intensity falls. The pupil reflex serves as an example of negative feedback used in a regulator mode, that is it maintains a constant level of light entering the eye. On the other hand the eye also needs to track objects that involves maintaining the eyeball fixed on the object. In this mode the eye functions as a servomechanism.

Both regulator and servomechanism can be implemented using the same operational mechanism. Figure 3.2 shows a generic negative feedback circuit. On the left of the figure can be found the input, sometimes termed the desired value or more often the **set point**. If the circuit is used as a servomechanism then the output tracks the set point. As the set point changes the output follows. If the circuit is used as a regulator or homeostatic device then the set point is held constant and the output is maintained at or near the set point even in the face of disturbances.

The central mechanism in the feedback circuit is the generation of the error signal, that is the difference between the desired output (set point) and the actual output. The error is fed into a controller (often something that simply amplifies the error) which is used to increase or decrease the process. For example, if a disturbance on the process block reduces the output, then the feedback operates by generating a positive error, this in turn increases the process and restores the original drop in the output.

Class Exercise 2

Figure 3.2 shows a generic feedback circuit. Identify the set point in a voltage follower circuit. See Figure 1.13.

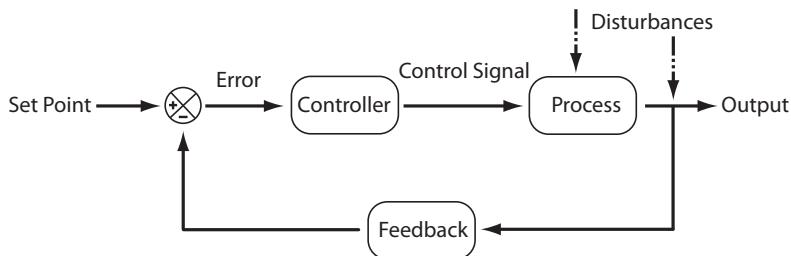


Figure 3.2 Generic structure of a negative feedback system.

3.3 Simple Quantitative Analysis

In the remainder of this section we will consider some basic properties of negative feedback systems. Later chapters will consider negative feedback in much more detail. The simplest way to think about feedback quantitatively is by reference to Figure 3.3.

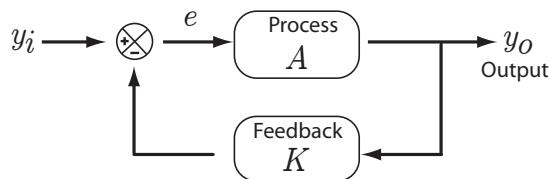


Figure 3.3 Simplified structure of a negative feedback system.

In this figure the process block which we wish to control has a gain of A and the feedback a gain of K . We will assume some very simple rules that govern the flow of information in this feedback system. For example, the output signal, y_o , will be the value of the gain A multiplied by the error, e . The feedback signal will be assumed to be proportional to y_o via gain K , that is Ky_o . Finally, the error signal, e will be given by the difference between the set point, y_i and the feedback signal, Ky_o (Figure 3.4).

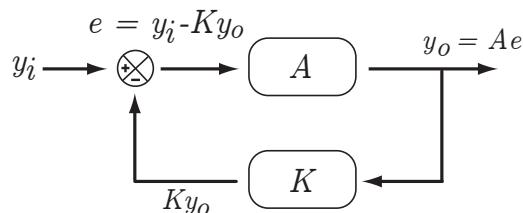


Figure 3.4 Negative feedback system with various gains listed.

Gain	Expression
Open Loop Gain	A
Loop Gain	AK
Closed Loop Gain	$\frac{A}{1 + AK}$

Table 3.1 Definition of Various Loop Gains

Noting that $e = y_o/A$ and substituting this into $e = -y_i - Ky_o$ and solving for y_o we obtain:

$$y_o = \frac{Ay_i}{1 + AK} \quad \text{or more simply} \quad y_o = Gy_i \quad (3.1)$$

where:

$$G = \frac{A}{1 + AK} \quad (3.2)$$

G is called the gain of the feedback loop, often called the **closed loop gain**. *Gain* is a term that is commonly used in control theory and refers to the scalar change between an input and output. Thus a gain of 2 simply means that a given output will be twice the input.

The **gain** is a measure of the change that occurs between a signal output and its input. A gain of two means that the output will change two times in magnitude compared to a change in the input.

In addition to the closed loop gain, engineers also define two other gain factors, the **open loop gain** and the important **loop gain**. The open loop gain is simply the gain generated by the process, A . It would be the gain between y_o and y_i if the feedback loop were absent. The loop gain is the gain from the feedback and process A combined, AK . The loop gain is an important quantity when discussing the stability and general performance of feedback circuits. Figure 3.4 illustrates the different types of gain in a feedback circuit, also summarized in Table 3.1.

We can use equation (3.1) to discover some of the basic properties of a negative feedback circuit. The first thing to note is that as the loop gain, AK , increases, the system behavior becomes more dependent on the feedback loop and less dependent on the rest of the system:

$$\text{when } AK \gg 1 \quad \text{then} \quad G \simeq \frac{A}{AK} = \frac{1}{K}$$

This apparently innocent effect has significant repercussions on other aspects of the circuit. To begin with, the system becomes less dependent on A . That is feedback makes the performance of the system independent of any variation in A . Such variation might include noise, temperature or variation as a

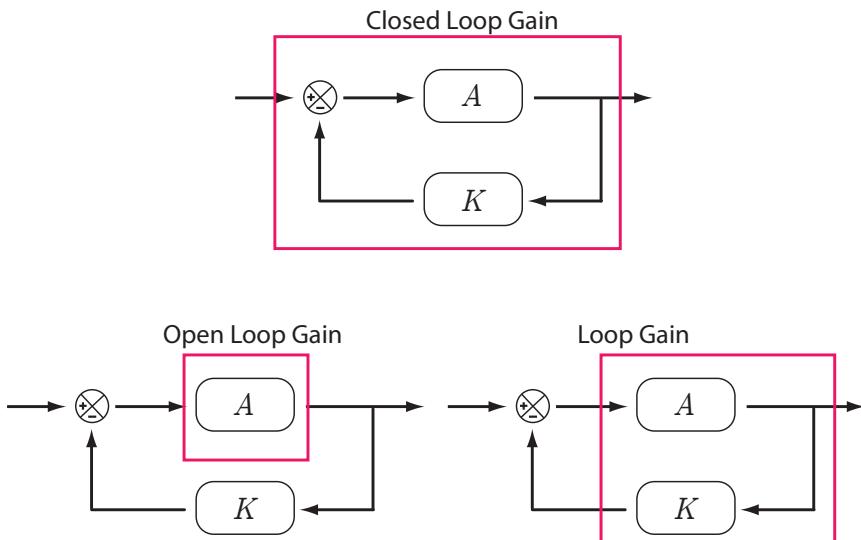


Figure 3.5 The various loop gains in a negative feedback system.

result of the manufacturing process or in the case of biological systems, genetic variation. To be more precise we can compute the sensitivity of the gain G with respect to variation in A .

$$\frac{\partial G}{\partial A} = \frac{\partial}{\partial A} \frac{A}{1 + AK} = \frac{1}{(1 + AK)^2}.$$

If we consider the relative sensitivity we find:

$$\frac{\partial G}{\partial A} \frac{A}{G} = \frac{1}{1 + AK}$$

From this we can see that as the loop gain increases the sensitivity decreases.

Functional Modules In addition to resistance to parameter variation, feedback also confers resistance to disturbances in the output. Suppose that a nonzero disturbance d affects the output. The system behavior is then described by

$$y = Ae - d \quad e = y_i - Ky.$$

Eliminating e , we find

$$y = \frac{Ay_i - d}{1 + AK}.$$

The sensitivity of the output to the disturbance is then

$$\frac{\partial y}{\partial d} = -\frac{1}{1 + AK}.$$

The sensitivity decreases as the loop gain AK is increased. In practical terms, this means that the imposition of a load on the output, for example a current drain in an electronic circuit, protein sequestration on a signaling network or increased demand for an amino acid will have less of an effect on the circuit as the feedback strength increases. In electronics this property essentially separates the network into **functional modules**.

Fidelity For a servo mechanism such as an amplifier where the output tracks the input, feedback confers a critical benefit, that is improved fidelity of the response. This means for a given change in the input, a system with feedback is more likely to faithfully reproduce the input at the output than a circuit without feedback. An ability to faithfully reproduce signals is critical in electronics communications and in fact it was this need that was one of the inspirations for the development of negative feedback in the early electronics industry. The next section on linearization will cover this in more detail.

Linearization Properties Related to the improvement in fidelity is linearization due to feedback. Consider the case where the amplifier A is nonlinear. For example a protein cascade pathway exhibiting a sigmoid response. Then the behavior of the system G (now also nonlinear) is described by

$$G(y_i) = y_o = A(e) \quad e = y_i - Ky_o = y_i - KG(y_i).$$

Differentiating we find

$$G'(y_i) = A'(y_i) \frac{de}{dy_i} \quad \frac{de}{dy_i} = 1 - KG'(y_i).$$

Eliminating $\frac{de}{dy_i}$, we find

$$G'(y_i) = \frac{A'(y_i)}{1 + A'(y_i)K}.$$

We find then, that if $A'(y_i)K$ is large ($A'(y_i)K \gg 1$), then

$$G'(y_i) \approx \frac{1}{K},$$

That is $G(y_i) = y_i/K$. This means that G is approximately linear (Recall that K is constant).² In this case, the feedback compensates for the nonlinearities $A(\cdot)$ and the system response is not distorted. Another feature of this analysis is that the slope of $G(\cdot)$ is less than that of $A(\cdot)$, i.e. the response is “stretched out”. For instance, if $A(\cdot)$ is saturated by inputs above and below a certain “active range”, then $G(\cdot)$ will exhibit the same saturation, but with a broader active range.

One objection to the implementation of feedback as described is that the system sensitivity is not actually reduced, but rather is shifted so that the response is more sensitive to the feedback K and

²I'd like to thank Brian Ingalls for his contribution here.

less sensitive to the amplifier A . However, in each of the cases described above, we see that it is the nature of the loop gain AK (and not just the feedback K) which determines the extent to which the feedback affects the nature of the system. This suggests an obvious strategy. By designing a system which has a small “clean” feedback gain and a large “sloppy” amplifier, one ensures that the loop gain is large and the behavior of the system is satisfactory. Engineers employ precisely this strategy in the design of electrical feedback amplifiers, regularly making use of amplifiers with gains several orders of magnitude larger than the feedback gain (and the gain of the resulting system). Because the amplifier A need not be precise, it means that the costs for manufacturing the amplifier can be greatly reduced. Instead the cost can be shifted to the feedback which is generally a much simpler mechanism. This is clearly seen in the use of op amps as amplifiers. The op amp is a complex circuit that exhibits a huge gain factor. However op amps are very cheap because negative feedback can be used to reduce the effect of manufacturing variation in the op amp. Instead, the cost is shifted to using high precision but cheap resistors to implement the negative feedback circuit.

Useful Properties Resulting from Negative Feedback

1. Amplification of signal.
2. Robustness to internal component variation.
3. High fidelity of signal transfer.
4. Low output impedance so that the load does not affect the performance of the circuit.

We will see in later chapters that negative feedback can confer additional useful properties.

3.4 PID Controllers

In deriving equation 3.1 it was assumed that the output from the process was directly proportional to the error: $y_o = Ae$. This is called **proportional control**. However, this isn’t the only way to respond to the error signal. Of particular importance is responding to the **integral** and **derivative** of the error. When all three are combined, the result is the **PID controller** where PID stands for proportional, integral and derivative control. This is a widespread control strategy used in a huge variety of control systems ranging from toy robots, and biological systems to entire petrochemical plants.

Briefly, the three modes of feedback operate in the following way:

Proportional Control - responds to the present

The signal transmitted to the process is directly proportional to the error.

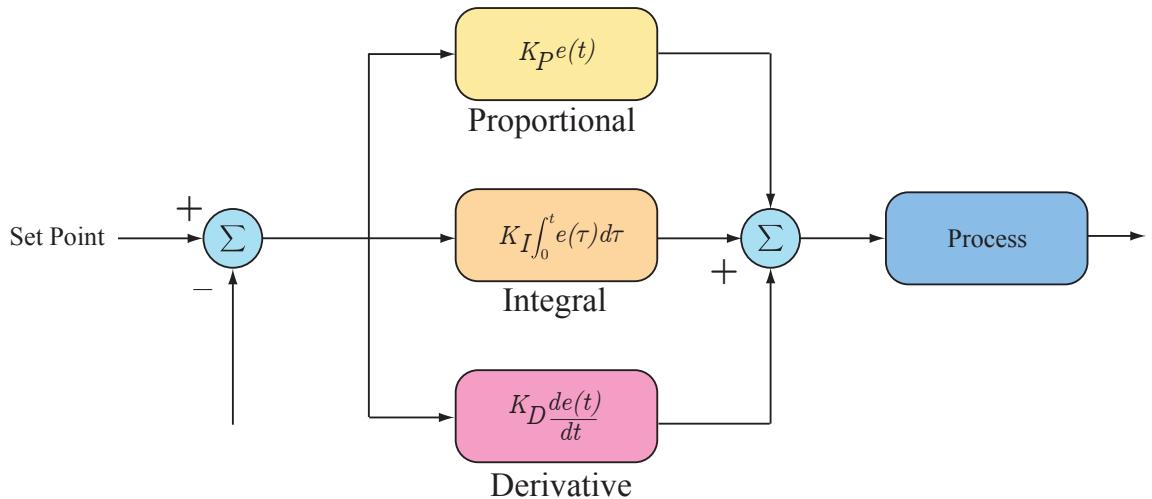


Figure 3.6 PID Control.

Integral Control - responds to the past.

The error signal is accumulated and the sum is used to control the process. One of the issues with proportional control is that it cannot eliminate disturbances completely so that there will always be some error, called the offset. The chief advantage of integral control is that when combined with proportional control, any disturbances are eliminated completely. Integral control is not without its issues. One problem that can occur is that the accumulated error can exceed the maximum integral that can be represented. This is particularly the case with mechanical or electrical integrators. In an electrical integrator a capacitor is frequently used to accumulate the error. Since a capacitor has a finite capacity saturation of the integrator can occur. This effect is called **integral windup**.

Derivative Control - responds to the future.

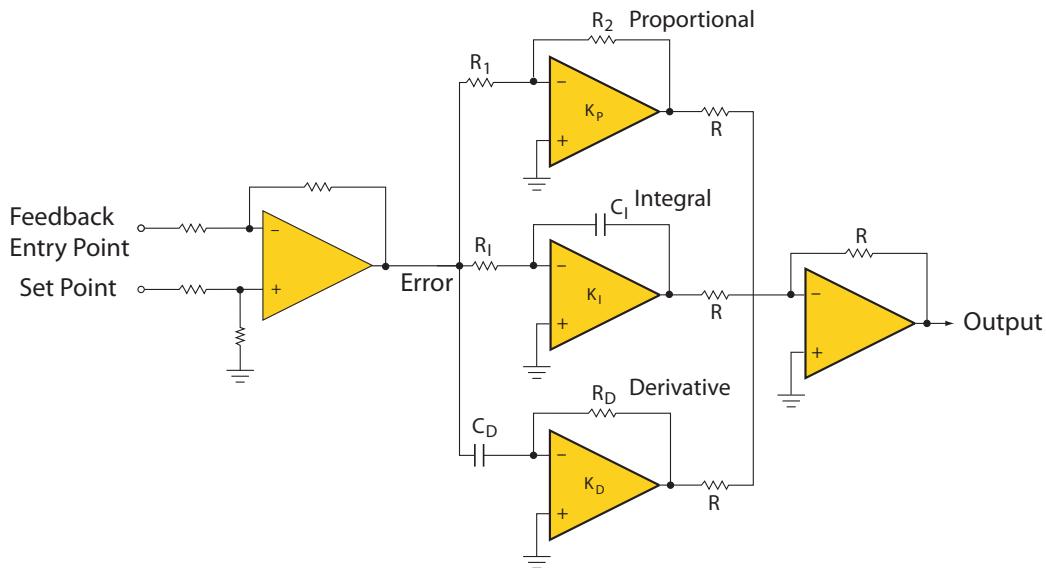
The derivative control prevents excessive under or overshoot by measuring how fast the error signal is changing. However it is susceptible to noise. Whereas integral will tend to average out noise, derivative control can cause the process to rapidly start and stop simply because of noise in the signal. For this reason, in many cases derivative control is avoided.

A PID controller ensures stability, accuracy and prevents overshoot and undershoot in transient changes. Mathematically a PID controller is represented as:

$$u = \underbrace{K_P e(t)}_{\text{Proportional}} + \underbrace{K_I \int e(t) dt}_{\text{Integral}} + \underbrace{K_D \frac{de(t)}{dt}}_{\text{Derivative}}$$

PID controllers can be made by using three op amp circuits in parallel. A typical circuit is shown in

Figure 21.15.

**Figure 3.7** Op Amp based PID Controller.

With the advent of lost-cost microcontrollers such as the Arduino or even a Raspberry Pi it is also straight forward to implement PID controllers in software. The pseudocode in Figure 3.8 shows how a PID controller might be implemented.

PID controllers will be considered in more detail in a later chapter.

3.5 Examples of control systems

1. Figure 3.9 shows one of the earliest uses of a control system.

Objective: Regulate the water inlet so that the flow is constant.

2. Steam Engine Governor (Figure 3.10) used to regulate the steam inlet in response to the load put on the steam engine.

Objective: Regulate the steam inlet so that the steam engine output is constant.

3. Feedback amplifier (Figure 3.11), used in servo mode where the control objective is to faithfully generate an amplified version of an input signal.

Objective: Ensure that the output is a faithful reproduction of the input.

4. Amino acid biosynthesis and glycolytic pathway, Figure 3.12.

```

previousError = 0; integral = 0; dt = timeStep
loop:
    // Compute the error signal
    errorSignal = setpoint - outputSignal

    // Accumulate integral of the error
    integral = integral + (errorSignal*dt)

    // Compute a rough derivative
    derivative = (errorSignal - previousError)/dt

    // Compute the overall process drive signal
    driveSignal = (errorSignal*KP) + (integral*KI) + (derivative*KD)
    previousError = errorSignal
    sendToProcess (driveSignal)
    wait(dt)
    goto loop

```

Figure 3.8 Pseudocode for a PID controller. K_P , K_I and K_D are the gains on the proportional, integral and derivative controllers.

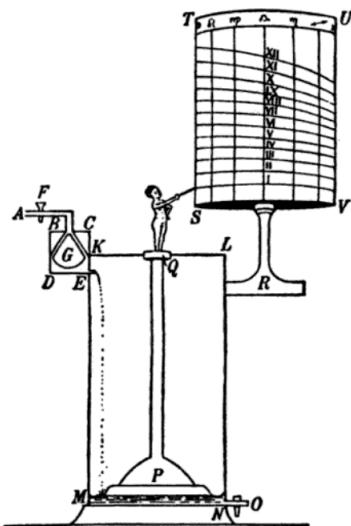


Figure 3.9 Ctesibios' Float Valve regulator, Water-clock, Alexandria 250BC. G marks the float valve. Image from Diels

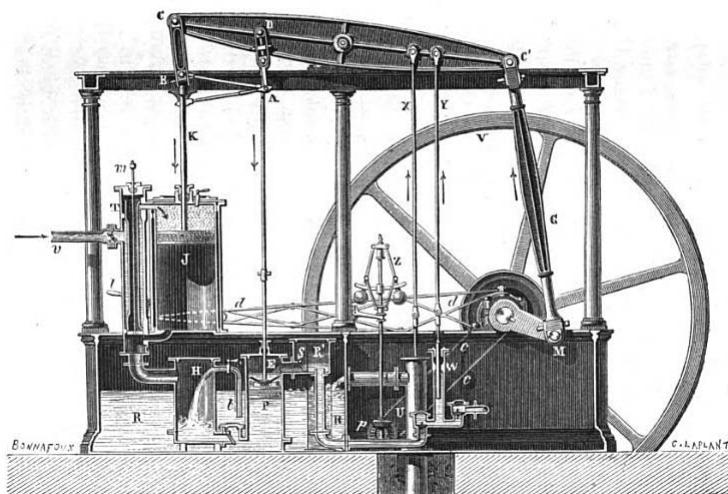


Figure 3.10 Engraving of a Steam Engine from *La vapeur*, by A. Guillemin, Paris, 1876. Note the governor at the center of the image, marked Z.

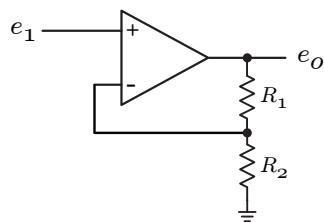


Figure 3.11 Negative feedback used in servo mode, tracking and amplifying an input signal.

Objective: Ensure that as demand for amino acids or ATP varies, the supply remains constant.

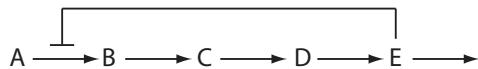


Figure 3.12 Negative feedback used to maintain a metabolite level, E , relatively constant even under varying demand.

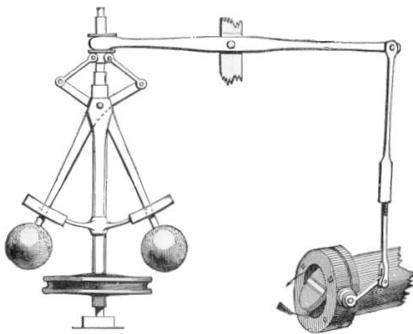


Figure 3.13 A governor controlling the steam inlet valve as a function of the speed of the engine. Illustration from An Elementary Treatise on Steam and the Steam-engine by Clark and Sewell published in 1892.

Class Question 2

Figure 3.2 shows a generic feedback circuit. Identify the set point in a voltage follower circuit.

Exercises

1. Define negative feedback.
2. Define each of the following terms:
 - (a) Open Loop Gain
 - (b) Closed Loop Gain
 - (c) Loop Gain
3. List four benefits of negative feedback
4. If the process block A is some nonlinear function $A(e)$, show that negative feedback (Figure 3.2) will still maintain linearity between y_0 and y_i so long as the gain of the process block is sufficiently high.
5. Investigate the physiology and molecular biology literature and describe two other systems where negative feedback is used. Suggest a possible evolutionary advantage for such feedback in your selected systems.
6. Build a computer simulation of a simple water tank model that uses negative feedback to control the water level.

4

Biological Networks

“In a system with many interactions many properties arise which cannot be assigned to any one isolable entity...”

– Henrik Kacser, 1963

4.1 Gene Regulation

At least in prokaryotes, the control of gene expression is relatively well understood. Transcription factors control gene expression by binding to special upstream DNA sequences called operator sites. Such binding results in the activation or inhibition of gene transcription. Multiple transcription factors can also interact to control the expression of a single gene. Such interactions can emulate simple logical functions (such as AND, OR etc.) or more elaborate computations. Gene regulatory networks can range from a single controlled gene to hundreds of genes interlinked with transcription factors forming a complex decision making circuit. Different classes of transcription factors exist, for example the binding of some transcription factors to operators sites is modulated by small molecules. A classic example of this is the binding of allolactose (a disaccharide very similar to lactose) to the lac repressor or cAMP to the catabolite activator protein (CAP). Alternatively, a transcription factor may be expressed by one gene that directly modulates a second gene (which can be itself), or via other transcription factors, integrating multiple signals onto another gene. Additionally, some transcription factors only become active when phosphorylated or unphosphorylated by protein kinases and phosphatases. Like protein signaling and control networks, gene regulatory networks can be elaborate, both structurally and computationally.

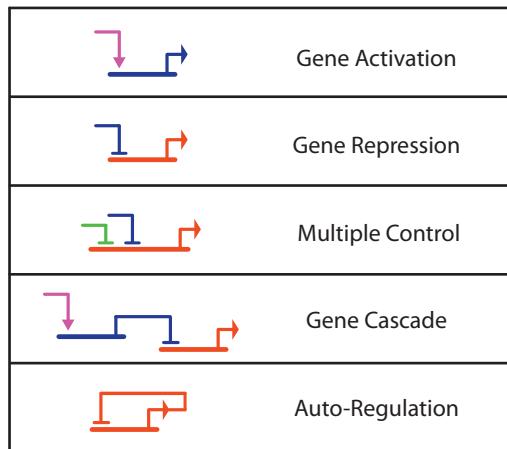


Figure 4.1 Simple gene regulatory patterns.

In general, gene regulatory networks are the slowest responding networks in a cell and work from minutes to hours depending on the organism, with bacterial gene regulatory networks tending to operate more rapidly.

The most extensive database on a gene regulatory network is RegulonDB [30, 17] which is a database on the gene regulatory network of *E. coli*. Detailed reviews that cover the structure of regulatory networks can be found in the reviews of Alon [57] and Seshasayee [56].

Although the description of the three main network types gives the impression that they act independently of each other, this is most definitely not the case. In general, the different networks will often act together. For example, Figure 4.2 includes an example taken from Caulobacter [6] showing a mixed gene regulatory and protein network.

4.2 Metabolic Pathways

The first cellular networks to be understood were the metabolic pathways glycolysis in the 1930s and the Calvin cycle in the 1940s. Early metabolic pathways were elucidated by a combination of enzymatic inhibitors and the use of radioisotopes such as Carbon-14. The Calvin cycle for example was discovered by following the fate of carbon when algae were exposed to ^{14}C -labeled CO_2 . With the development of microbial genetics, significant progress was also made in uncovering other pathways by studying mutants and complementing different mutants of a given pathway to determine the order of steps.

Traditionally, metabolism is classified into two groups, anabolic (synthesis) and catabolic (break-

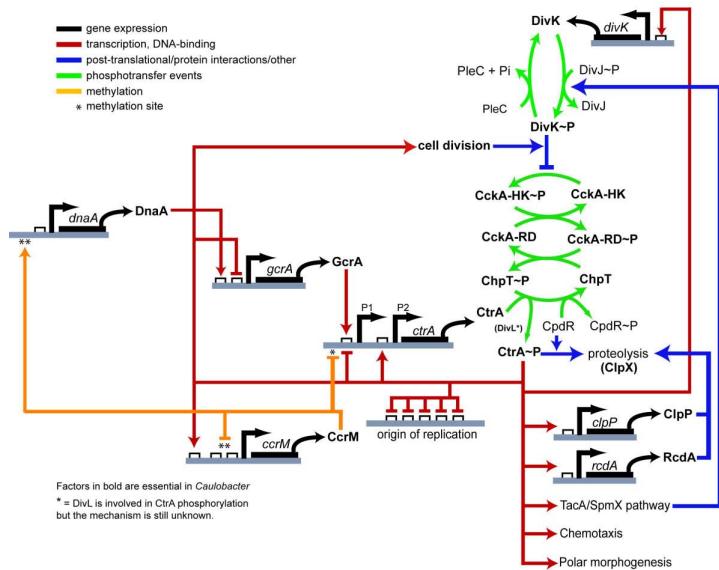


Figure 4.2 Example of a mixed network involving gene regulatory and protein phosphorylation networks in *Caulobacter*. Blunt ends to regulatory arcs indicate inhibition while arrow ends indicate activation. Image from BioMed Central [6].

down) metabolism. Coupling between the two metabolic groups is achieved through cofactors of which a great variety exist. Two well known and widely used cofactors include the pyridine nucleotides in the form of NAD^+ and NADP^+ , and the adenine nucleotides in the form of ATP, ADP and AMP. These cofactors couple redox and phosphate, respectively, by forming reactive intermediates that enables catabolism to drive anabolism.

Metabolic networks are by far the fastest (excluding ion transfer mechanisms) in terms of their response to perturbations and can operate in a time scale from microseconds to seconds. This reflects the need to rapidly adjust the supply of molecular building blocks and energy as supply and demand fluctuate. Physically the rapid response of metabolic networks is achieved by allosteric control where the fast diffusion of small molecules can bind and alter the activity of selected enzymes extremely rapidly.

4.3 Protein Networks

Protein networks are by far the most varied networks found in biological cells. They range from proteins involved in controlling gene expression, the cell cycle, coordinating and processing signals from

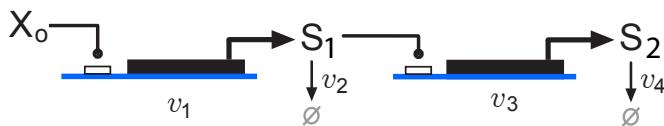


Figure 4.3 A cascade of two genes. v_1 represent the expression rate for protein S_1 . v_2 represents the degradation rate of S_1 . v_3 is the expression rate of protein S_2 . v_4 is the degradation rate of S_2 . X_o is the input inducer to the cascade.

the external and internal environments, to highly sophisticated nano-machines such as the ribosome or the bacterial flagella motor.

Protein networks can be studied on different levels, broadly classified as either stoichiometric or non-stoichiometric networks. The non-stoichiometric networks can be as simple as considering the physical associations between different proteins (often through the formation of protein complexes). Such networks, also termed interaction networks, have been elucidated largely with the help of high-throughput methods. An interaction is formed if two proteins, A and B are known to associate.

Another descriptive level involves functional and stoichiometric networks formed from a consideration of specific stoichiometric binding events, covalent modification (most notably phosphorylation) and degradation. Here two protein A and B might form a complex with a specific stoichiometric relationship and with a specific association constant.

Many protein-protein networks operate as signal processing networks and are responsible for sensing external signals such as nutritional (for example by changes in glucose levels) or cell to cell signals such as insulin. Other signalling networks include control networks that are concerned with monitoring and coordinating internal changes, the most well known of these includes the cell cycle control network. Many external signals act by binding to cell-surface receptor proteins such as the large family of receptor tyrosine kinases and G-protein coupled receptors. Once a signal is internalized through the cell-surface receptors, other proteins, including protein kinases and phosphatases, continue to process the signal often in coordination with other signaling networks. Eventually the signalling pathway terminates on target proteins that lead to a change in the cell's behavior. Such targets can include a wide variety of proteins involved in metabolic pathways, ion channels, cytoskeleton, motor proteins and gene regulatory proteins.

The molecular mechanisms employed by signalling and control pathways include covalent modification, degradation and complex formation. Covalent modification in particular is a common mechanism used in signaling networks and includes a variety of different modifications such as phosphorylation, acetylation, methylation, ubiquitylation, and possibly others [5]. As a result the structure and computational abilities [53] of such networks can be extremely elaborate. It has been estimated from experimental studies that in *E. coli*, 79 proteins can be phosphorylated [36] on serine, threonine and tyrosine side groups whereas in yeast, 4000 phosphorylation events involving 1,325 different proteins

have been recorded [45].

The cell cycle control network is an excellent example of a sophisticated protein control network that coordinates the replication of a biological cell. The cell cycle includes a number of common molecular mechanisms that are found in many other protein networks. These mechanisms can be grouped into three broad types and include phosphorylation, degradation and complex formation. Phosphorylation is a common mechanism for changing the state of a protein and involves phosphorylation on a number of sites on the protein surface including serine/threonine and tyrosine. In prokaryotes, histidine, arginine or lysine can also be phosphorylated. Phosphorylation is mediated by kinases of which the human genome may have over 500 kinase encoding genes [37]. The effect of phosphorylation is varied but generally causes the protein undergoing phosphorylation to either change catalytic activity, to change the protein's 'visibility' to other proteins, or to mark the protein for degradation. For example, src is a tyrosine kinase protein involved in cell growth. It has two states, active and inactive; when active it has the capacity to phosphorylate other proteins. Deactivation of src is achieved by phosphorylation of a tyrosine group on the C-terminal end of the protein. Dephosphorylation of the tyrosine group by tyrosine phosphatase results in the activation of the protein.

Phosphorylation can also be used to inactivate enzymes such as glycogen synthase by the glycogen synthase kinase 3 protein. In the yeast cell cycle, the protein Wee1 is phosphorylated and inactivated by the complex Cdc2-Cdc13. Active Wee1 in turn (that is the unphosphorylated form) can inactivate Cdc2-Cdc13 by phosphorylating the Cdc2 subunit.

In addition to changing the activity of proteins, phosphorylation can also be used to mark proteins for degradation. For example, the protein Rum1 that is part of the yeast cell cycle control network can be phosphorylated by Cdc2-Cdc13. Once phosphorylated, Rum1 is degraded. Degradation itself is an important mechanism used in protein networks and allows proteins to be rapidly removed from a network according to the state of the cell. Degradation is usually mediated by ubiquitylation. For example, Cdc2-Cdc13, via Ste9 and APC is marked for degradation by ubiquitylation (Rum1 is similarly processed once phosphorylated). Once marked this way, such proteins can bind to the proteasome where they are degraded. The binding of one protein to another can change the target protein's activity or visibility. An example of this is the inactivation of Cdc2-Cdc13 by Rum1. When unphosphorylated, Rum1 binds to Cdc2-Cdc13, and the resulting complex is inactive.

Different combinations of these basic mechanisms are also employed. For example, phosphorylation of complexes can lead to the dissociation of the complex, or the full activity of a protein may require multiple phosphorylation events. Although signalling networks can appear highly complex and varied, most of them can be reduced to the three fundamental mechanisms: covalent modification, selective degradation and complex formation (Fig 4.4).

In higher eukaryotic cells, particularly human, around 2% of the protein-coding part of the genome is devoted to encoding protein kinases, with perhaps 10% of the coding region dedicated to proteins involved in signalling networks. It has also been suggested that as much as 30% of all cellular proteins in yeast and human can be phosphorylated[10].

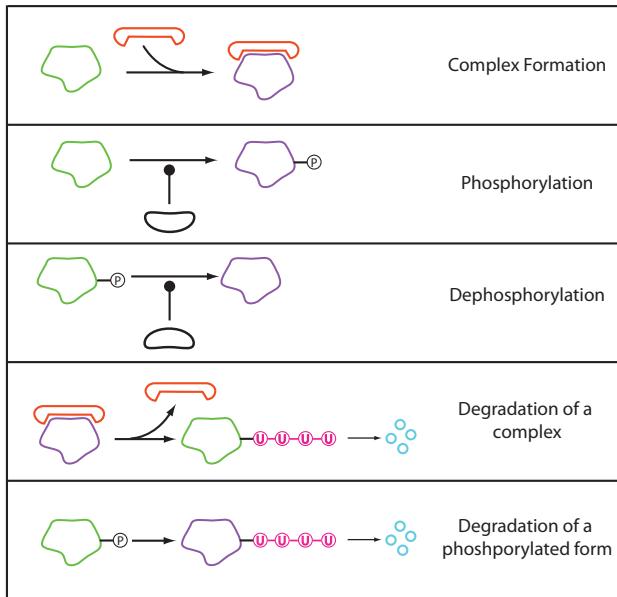


Figure 4.4 Fundamental Protein Mechanisms.

The actual size of the networks themselves is even larger than these numbers suggest because of the significant number of covalent variants and binding permutations. For example, p53, a well known tumor suppressor protein, has between 17 and 20 phosphorylation sites alone [60]. If every combination were phenotypically significant, though unlikely, that amounts to at least 131,072 different states.

Ptacek and Snyder [46] have published a review on elucidating phosphorylation networks where much more detailed information is given.

4.4 Stoichiometric Networks

A chemical reaction is usually depicted in the form of a chemical equation which describes the transformation of one or more **reactants** into one or more **products**. The reactants appear on the left of the equation and the products on the right. Both sides are separated by an arrow indicating the positive direction of the transformation. The simplest possible reaction is the conversion of a single reactant, *A*, into a single product, *B*, as depicted in the following way:



Such a reaction can be studied by observing the change in concentration of *A* and/or *B* in time. Experimentally there are a variety of ways to do this, for example by observing the emission or absorption of light at a specific wavelength, the change in pH, or the incorporation of a radioactive or heavy isotope into the product. An example of an actual biochemical reaction is the familiar interconversion of the adenine nucleotides:



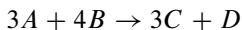
This describes two molecules of ADP being transformed into one molecule of ATP and one molecule of AMP. Sometimes a double arrow is used to explicitly indicate that a reaction is reversible, as in:



If a reaction is reversible (as almost all reactions are to some extent), then the reaction rate can be positive or negative. By convention, a positive rate means that the reaction progresses from left to right, whereas a negative rate indicates a right to left reaction.

Example 4.1

What does the following reaction notation mean:



This notation means that during a reaction event, 3 molecules of *A* and 4 molecules of *B* react to form 3 molecules of *C* and one molecule of *D*.

We now need to define a number of terms: the stoichiometric amount, rate of change, stoichiometric coefficient, and reaction rate.

Stoichiometry refers to the molar proportions of reactants and products in a chemical reaction. We will first distinguish two related measures of stoichiometry, stoichiometric amounts and stoichiometric coefficients.

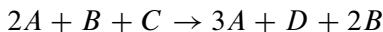
Stoichiometric Amounts

The **stoichiometric amount** is defined as the number of molecules of a particular reactant or product taking part in a reaction. Stoichiometric amounts will always be **positive** numbers. For example, in the reaction:



ADP has a stoichiometric amount of two, ATP a stoichiometric amount of one, and AMP also with a stoichiometric amount of one. If the same species occurs on the reactant and product side of a reaction

then it must be treated separately. For example, in the reaction:

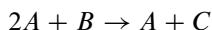


The stoichiometric amounts on the reactant side include: A with two, B with one and C with one. On the product side the stoichiometric amounts include: A with three, D with one and B with two.

The **stoichiometric amount** is the number of molecules of a particular reactant or product taking part in a reaction.

Example 4.2

List the stoichiometric amounts in the following reaction:



On the reactant side the stoichiometric amount for *A* is two and for *B* is one. On the product side, the stoichiometric amount for *A* is one and for *C* one.

Stoichiometric Coefficients

Stoichiometry deals with static information about the amounts of substances involved in a chemical transformation, whereas kinetics relates rates of change that occur in these amounts. To paraphrase a statement made by Aris [1], stoichiometry provides the framework within which chemical change takes place irrespective of the forces that bring them about, and by kinetics the speed of chemical change. Aris then went on to state, “Just as the latter can only be built on a proper understanding of the kinematics, so the analysis of stoichiometry must precede that of kinetics”. We will do the same here.

The stoichiometry coefficient refers to the **relative** amount of substance that is consumed and/or produced by a reaction. Given a reaction such as:

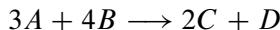


the stoichiometric amount of *A* is 2 and for *B*, 1. The species stoichiometry or **stoichiometric coefficient** however, is the difference between the stoichiometric amounts if a given species on the product side and the stoichiometric amount of the same species on the reactant side. The definition below summarizes this more clearly.

The **stoichiometric coefficient**, c_i , for a molecular species A_i , is the difference between the stoichiometric amount of the species on the product side and the stoichiometric amount of the same species on the reactant side, that is:

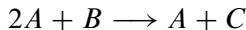
$$c_i = \text{Stoichiometric Amount of Product, } A_i - \text{Stoichiometric Amount of Reactant, } A_i$$

In the reaction, $2A \rightarrow B$, the stoichiometric amount of A on the product side is **zero** while on the reactant size it is two. Therefore the stoichiometric coefficient of A is given by $0 - 2 = -2$. In many cases a particular species will only occur on the reactant or product side and it is relatively uncommon to find situations where a species occurs simultaneously as a product and a reactant. As a result, reactant stoichiometric coefficients tend to be **negative** and product stoichiometric coefficients tend to be **positive**. To illustrate this further consider the more complex reaction:



Since A only appears on the reactant side, its stoichiometric coefficient will be -3 , similarly for B which will have a stoichiometric coefficient of -4 . Species C only occurs on the product side, therefore its stoichiometric coefficient is $+2$, and similarly for D which will have a stoichiometric coefficient of $+1$. In these cases the stoichiometric amounts and the stoichiometric coefficients are the same except for the sign difference on the reactant stoichiometric coefficients.

Finally consider the following reaction where a species occurs on both the reactant and product side:

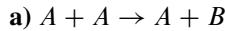


The stoichiometric coefficient of A must take into account the fact that A appears both as a reactant and a product. The overall stoichiometric coefficient of A is therefore $+1 - 2$ which gives -1 .

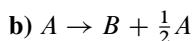
The last example highlights how information can be lost when computing stoichiometric coefficients. It is not possible to recreate the original reaction equation from the stoichiometric coefficients alone, and therefore underscores the danger of just supplying stoichiometric coefficients when communicating information on reaction equations to other researchers. One option is to store the stoichiometric amounts together with the associated reactant or product. Computer exchange formats, such as the Systems Biology Markup Language (SBML) [29] are specifically designed to preserve complete reaction equation information for this very reason.

Example 4.3

Write down the stoichiometric coefficients for the following reactions:



The stoichiometric amount of A on the reactant side is 2 and on the product side, 1. Therefore the stoichiometric coefficient for A is $1 - 2 = -1$. The stoichiometric amount of B on the product side is 1 and on the reactant side, 0, therefore the stoichiometric coefficient for B is $1 - 0 = 1$.



The stoichiometric amount of A on the reactant side is 1 and on the product side $\frac{1}{2}$, therefore the stoichiometric coefficient for A is $1/2 - 1 = -1/2$. The stoichiometric amount of B on the reactant side is 0 and on the product side, 1, therefore the stoichiometric coefficient for B is $1 - 0 = 1$.

Example 4.3 (b) highlights another fact about stoichiometric coefficients. The coefficients can be fractional amounts, often represented as rational fractions.

4.5 Reaction Kinetics

Reaction kinetics is the study of how fast chemical reactions take place, what factors influence the rate of reaction and what mechanisms are responsible. Many variables can affect the reaction rate including temperature, pressure and composition. In this chapter we will review a number of topics related to reaction kinetics that have a significant bearing on the development of mathematical models of cellular networks.

Rates of Change

The rate of change can be defined as the rate of change in concentration or amount (depending on units) of a designated species. If S is the species then the rate of change is given by:

$$\text{Rate} = \frac{\Delta S}{\Delta t}$$

Because rates change as reactants are consumed and products made, the rate of change is better defined as the instantaneous change in concentration, or a derivative:

$$\text{Rate} = \frac{dS}{dt} \tag{4.1}$$

If we were to plot the rate of product formation as a function of time, the rate of reaction would be given by the slope of the curve (Fig. 4.5). If concentrations are measured in moles per liter (L) and time in seconds (sec), then the rate of reaction is expressed in mol L⁻¹ sec⁻¹.

When reporting a rate of change, it is important to give the name of the species that was used to make the measurement. For example, in the reaction $2A \rightarrow B$, the rate of change of A is twice the rate of change of B . In addition, the rate of change of A is negative because it is consumed, whereas the rate of change of B is positive because it is being made.

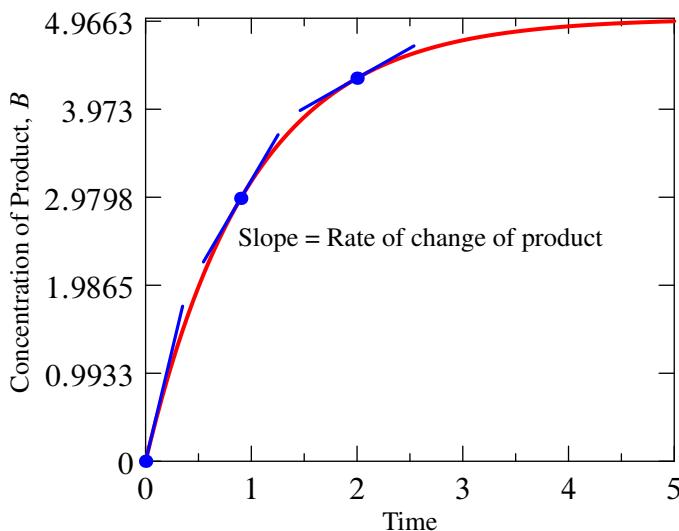


Figure 4.5 Progress curve for a simple irreversible reaction, $A \rightarrow B$. Initial reactant concentration, A , is set at 5 units. The plot shows the accumulation of product, B , as the reaction proceeds. The rate of change of product is given by the slope of the curve which changes over the course of the reaction.

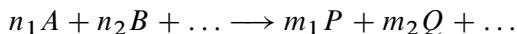
Reaction Rates

In this section we will introduce the concept of a **reaction rate**, denoted by v . The standard unit for the reaction rate is amount per volume per time. This is an intensive property, which does not depend on the amount of substance, for example $\text{mol L}^{-1} \text{ sec}^{-1}$. In the previous section we introduced the rate of change. In practice it is the rate of change that we measure experimentally. We also briefly mentioned that in the reaction $2A \rightarrow B$, A is consumed twice as fast as the production of product, B . This means that the sign and magnitude of the rates of change will vary depending on which species we choose to measure.

A simple way to avoid these differences is to divide each rate of change by the species **stoichiometric coefficient**. In this case the stoichiometric coefficient of A is -2 and for B is $+1$. If we do this we obtain:

$$\frac{1}{-2} \frac{dA}{dt} = \frac{1}{1} \frac{dB}{dt} = v$$

In general, for a reaction of the form:



where n_1, n_2, \dots and m_1, m_2, \dots represent the stoichiometric coefficients, the reaction rate is given

by:

$$\text{Rate} = v \equiv -\frac{1}{n_1} \frac{dA}{dt} = -\frac{1}{n_2} \frac{dB}{dt} \dots = \frac{1}{m_1} \frac{dP}{dt} = \frac{1}{m_2} \frac{dQ}{dt} \dots \quad (4.2)$$

Defined this way, a reaction rate is independent of the species used to measure it. The same applies if a given species appears on both sides of a reaction. For example, in the reaction $A \rightarrow 2A$, the stoichiometric coefficient is +1 so that the reaction rate, v , is:

$$v = \frac{1}{+1} \frac{dA}{dt}$$

To make the definition of the reaction rate more formal, let us introduce the **extent of reaction**, indicated by the symbol, ξ . We define a change from ξ to $\xi + d\xi$ in time dt to mean that $c_1 d\xi$ moles of A_1 , $c_2 d\xi$ moles of A_2 etc, react to form $c_n d\xi$ moles of A_n etc. By this definition we can state that for any component i , the following is true for the time interval dt :

$$dn_i = c_i d\xi \quad (4.3)$$

or

$$\frac{dn_i}{dt} = c_i \frac{d\xi}{dt}$$

where n_i equals the amount in moles of species i . From this relation we **define** the **extensive rate of reaction**, v_E , to be:

$$v_E \equiv \frac{d\xi}{dt}$$

In other words:

$$\frac{dn_i}{dt} = c_i v_E \quad (4.4)$$

For the moment we will use v_E and v_I to distinguish the extensive and intensive reaction rates. Note that ξ has units of **amount** and v_E has units of **amount per unit time** and is therefore an **extensive property**, being dependent on the size of the system. The advantage of introducing the extent of reaction is that it allows us to formally define the rate of reaction independently of the species we use to measure the rate. This convenient property can be expressed as:

$$v_E \equiv \frac{d\xi}{dt} = -\frac{1}{c_1} \frac{dn_1}{dt} = -\frac{1}{c_2} \frac{dn_2}{dt} \dots = \frac{1}{c_n} \frac{dn_n}{dt} = \frac{1}{c_{n+1}} \frac{dn_{n+1}}{dt} \dots$$

Example 4.4

Express the rate of reaction and the rates of change for the following biochemical reaction:



The rate of reaction is given by:

$$\begin{aligned} v &= \frac{d\xi}{dt} = \frac{dn(\text{ATP})}{dt} = \frac{dn(\text{AMP})}{dt} \\ &= -\frac{1}{2} \frac{dn(\text{ADP})}{dt} \end{aligned}$$

If the volume, V , of the system is constant we can also express the rate in terms of concentration, $C_i = n_i / V$. We can therefore rewrite the rate of reaction in the form:

$$\frac{v_E}{V} = -\frac{1}{c_1} \frac{dC_1}{dt} = \dots$$

where v_E has units of amount per unit time (mol s^{-1}). The relation v_E / V is the intensive version of the rate, v_I , with units of concentration per unit time ($\text{mol L}^{-1} \text{s}^{-1}$) and is the most commonly used form in biochemistry.

$$v_I = \frac{v_E}{V} = \frac{1}{c_i} \frac{dC_i}{dt}$$

or

$$\frac{dC_i}{dt} = c_i v_I \quad (4.5)$$

where C_i is the concentration of species i and v_I is the **intensive rate of reaction**. For constant volume, single compartment systems, this is a commonly encountered equation in models of cellular networks. The above equation may also be expressed as:

$$\frac{1}{V} \frac{dn_i}{dt} = c_i v_I \quad (4.6)$$

to emphasize the change in mass that accompanies a reaction. Recall that v_I is expressed using $\text{mol L}^{-1} \text{s}^{-1}$. If a E or I subscript is not used on v then the specific form should be clear from the context. In this book, where we use v , we will generally mean v_I , the intensive form.

In situations involving multiple compartments of different volumes or where there are specific mass conservation laws at work, the intensive rate is not appropriate. This is because the intensive version

is unable to keep track of the total number of moles undergoing transformation. In these situations it is necessary to deal explicitly with the extensive rate of reaction, in other words:

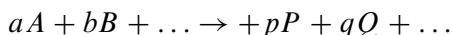
$$\frac{dn_i}{dt} = V c_i v_I$$

A Word on Notation

In many texts, the concentration (molarity) of a substance, X , is denoted using square brackets, as in $[X]$. To avoid unnecessary clutter in the current text, the use of square brackets to indicate molarity will be avoided.

4.6 Elementary Mass-Action Kinetics

An elementary reaction is one that cannot be broken down into simpler reactions. Such reactions will often display simple kinetics called mass-action kinetics. For a reaction of the form:



the mass-action kinetic rate law is given by:

$$v = k_1 A^a B^b \dots - k_2 P^p Q^q \dots$$

where k_1 and k_2 are the forward and reverse rate constants, respectively.

4.7 Chemical Equilibrium

In principle, all reactions are reversible, meaning transformations can occur from reactant to product or product to reactant. The net rate of a reversible reaction is the difference between the forward and reverse rates. Given a reversible reaction such as:



we can observe the concentrations of A and B approach equilibrium (Figure 4.7).

At chemical equilibrium the forward and reverse rates are equal and is described by the relation:

$$\frac{k_1}{k_2} = \frac{B}{A} = K_{eq} \quad (4.7)$$

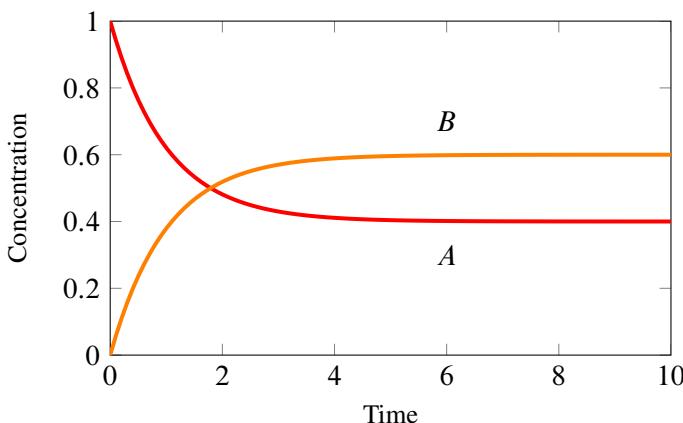
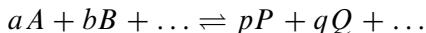


Figure 4.6 Approach to equilibrium for the reaction $A \rightleftharpoons B$, $k_1 = 0.6$, $k_2 = 0.4$, $A(0) = 1$, $B(0) = 0$. Progress curves calculated from the solution to the differential equation $dA/dt = k_2 B - k_1 A$.

This ratio has special significance and is called the **equilibrium constant**, denoted by K_{eq} . The equilibrium constant is also related to the ratio of the rate constants, k_1/k_2 . For a general reversible reaction such as:

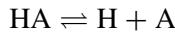


and using arguments similar to those described above, the ratio of the rate constants can be easily shown to be:

$$K_{eq} = \frac{P^p Q^q \dots}{A^a B^b \dots} = \frac{k_1}{k_2} \quad (4.8)$$

where the exponents are the stoichiometric *amounts* for each species.

For a bimolecular reaction such as:



chemists and biochemists will often distinguish between two kinds of equilibrium constants called association and dissociation constants. Thus the equilibrium constant for the above bimolecular reaction is often called the **dissociation constant**, K_d :

$$K_d = \frac{H \cdot A}{HA}$$

to indicate the degree that the complex is dissociated into its component molecules at equilibrium. The **association constant**, K_a , though less commonly used, describes the equilibrium constant for the reverse process $H + A \rightleftharpoons HA$, that is the formation of a complex from component molecules:

$$K_a = \frac{HA}{H \cdot A}$$

It should be evident that:

$$K_d = \frac{1}{K_a} \quad (4.9)$$

The equilibrium constant is also related to the standard free energy change, ΔG^o , such that:

$$\Delta G^o = -RT \ln K_{eq}$$

where R is the gas constant, and T the temperature. Rearranged we can also see that:

$$K_{eq} = e^{-\Delta G^o / RT} \quad (4.10)$$

4.8 Mass-action and Disequilibrium Ratio

Although in closed systems reactions tend to equilibrium, reactions occurring in living cells are generally out of equilibrium and the ratio of the products to the reactants *in vivo* is called the **mass-action ratio**, Γ . The ratio of the mass-action ratio to the equilibrium constant is called the **disequilibrium ratio**:

$$\rho = \frac{\Gamma}{K_{eq}} \quad (4.11)$$

At equilibrium the mass-action ratio will be equal to the equilibrium constant, that is $\rho = 1$. If the reaction is away from equilibrium ($B/A < K_{eq}$), then $\rho < 1$.

For a simple unimolecular reaction it was previously shown that the equilibrium ratio of product to reactant, B/A , is equal to the ratio of the forward and reverse rate constants. Substituting this into the disequilibrium ratio gives:

$$\rho = \Gamma \frac{k_2}{k_1} = \frac{B}{A} \frac{k_2}{k_1}$$

Therefore:

$$\rho = \frac{v_r}{v_f} \quad (4.12)$$

That is, the disequilibrium ratio is the ratio of the reverse and forward rates. If $\rho < 1$, the net reaction must be in the direction of product formation. If ρ is zero, the reaction is as out of equilibrium as possible with no product present.

4.9 Modified Mass-Action Rate Laws

A typical reversible mass-action rate law will require both the forward and the reverse rate constants to be fully defined. Often however, only one rate constant may be known. In these circumstances it is possible to express the reverse rate constant in terms of the equilibrium constant.

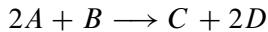
For example, given the simple unimolecular reaction, $A \rightleftharpoons B$, it is possible to derive the following:

$$\begin{aligned} v &= k_1 A - k_2 B \\ v &= k_1 A \left(1 - \frac{k_2 B}{k_1 A} \right) \\ \text{Since } K_{eq} &= \frac{k_1}{k_2} \\ v &= k_1 A \left(1 - \frac{\Gamma}{K_{eq}} \right) \end{aligned} \tag{4.13}$$

where Γ is the mass-action ratio. This can be generalized to an arbitrary mass-action reaction to give:

$$v = k_1 A^a B^b \dots \left(1 - \frac{\Gamma}{K_{eq}} \right) = k_1 A^a B^b \dots (1 - \rho)$$

where $A^a B^b \dots$ represents the product of all reactant species, a and b are the **corresponding** stoichiometric amounts, and ρ is the disequilibrium ratio. For example, for the reaction:



where k_1 is the forward rate constant, the modified reversible rate law is:

$$v = k_1 A^2 B (1 - \rho)$$

The modified formulation demonstrates how a rate expression can be divided up into functional parts to include both kinetic and thermodynamic components [28]. The kinetic component is represented by the term $k_1 A^a B^b \dots$, while the thermodynamic component is represented by the expression $1 - \rho$.

We can also derive the modified rate law in the following way. Given the net rate of reaction $v = v_f - v_r$, we can write this expression as:

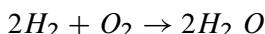
$$v = v_f \left(1 - \frac{v_r}{v_f} \right)$$

That is:

$$v = v_f (1 - \rho)$$

4.10 Elasticity Coefficients

A fundamental property of any reaction rate law is the **kinetic order**, sometimes called the reaction order. For simple mass-action chemical kinetics, the kinetic order is the power to which a species is raised in the kinetic rate law. Reactions with zero-order, first-order and second-order are commonly found in chemistry, and in each case the kinetic order is zero, one and two, respectively. For a reaction such as:



where the irreversible mass-action rate law is given by:

$$v = k H_2^2 \cdot O_2$$

the kinetic order with respect to hydrogen is two and oxygen one. In this case the kinetic order also corresponds to the stoichiometric amount of each molecule although this may not always be true.

It is possible to generalize the concept of the kinetic order by defining it as the scaled derivative of the reaction rate with respect to the species concentration, as follows:

$$\varepsilon_{S_i}^v = \left(\frac{\partial v}{\partial S_i} \frac{S_i}{v} \right)_{S_j, S_k, \dots} = \frac{\partial \ln v}{\partial \ln S_i} \approx v\% / S_i \% \quad (4.14)$$

From the definition it is apparent that elasticities are dimensionless quantities. When expressed this way, the kinetic order is often called the **elasticity coefficient** or in biochemical systems theory, the **apparent kinetic order**. The subscripts, S_j, S_k, \dots in definition (4.14) indicate that any species affecting the reaction must be held constant at their current value when species S_i is changed. This is also implied in the use of the partial derivative symbol, ∂ , rather than the derivative symbol, d . Elasticities can be derived from rate laws by differentiating and scaling the rate law equation (4.5).

Example 4.5

Determine the elasticities for the following mass-action rate laws:

1. $v = k$

Elasticity: $\varepsilon_A^v = \frac{\partial v}{\partial A} \frac{A}{v} = 0$

2. $v = kA$

Elasticity: $\varepsilon_A^v = \frac{\partial v}{\partial A} \frac{A}{v} = \frac{A k}{k A} = 1$

3. $v = kA^2$

Elasticity: $\varepsilon_A^v = \frac{\partial v}{\partial A} \frac{A}{v} = \frac{2kAA}{kA^2} = 2$

$$4. v = kA^n$$

$$\text{Elasticity: } \varepsilon_A^v = \frac{\partial v}{\partial A} \frac{A}{v} = \frac{n k A^{n-1} A}{k A^n} = n$$

Example (4.5) shows that the elasticity corresponds to the expected kinetic order for simple rate laws. The definition of the elasticity (4.14) also gives us a useful operational interpretation.

Operational Definition: The elasticity is the fractional change in reaction rate in response to a fractional change in a given reactant or product while keeping all other reactants and products constant.

That is, the elasticity measures how responsive a reaction is to changes in its immediate environment. Since the elasticity is expressed in terms of fractional changes, it is also possible to get an approximate value for the elasticity by considering percentage changes. For example, if we increase the substrate concentration of a particular reaction by 2% and the reaction rate increases by 1.5%, then the elasticity is given by $1.5/2 = 0.75$. The elasticity is however only strictly defined (See equation (4.14)) for infinitesimal changes and not finite percentage changes. So long as the changes are small, the finite approximation is a good estimate for the true elasticity.

For a given reaction, there will be as many elasticity coefficients as there are reactants, products and other effectors of the reaction. For species that cause reaction rates to increase, the elasticity is **positive**, while for species that cause the reaction rate to decrease, the elasticity is **negative**. Therefore, reactants generally have positive elasticities and products generally have negative elasticities.

For a species that **increases** the reaction rate: Elasticity is **positive**.

For a species that **decreases** the reaction rate: Elasticity is **negative**.

Log Form

The definition of the elasticity in equation (4.14) shows the elasticity expressed using a log notation:

$$\varepsilon_S^v = \frac{\partial \ln v}{\partial \ln S}$$

This notation is frequently used in the literature. The right panel (B) of Figure 4.7 shows one application of this notation, namely that a log-log plot of reaction rate versus reactant concentration yields a curve where the elasticity can be read directly from the slope. The origin of this notation will be explained here.

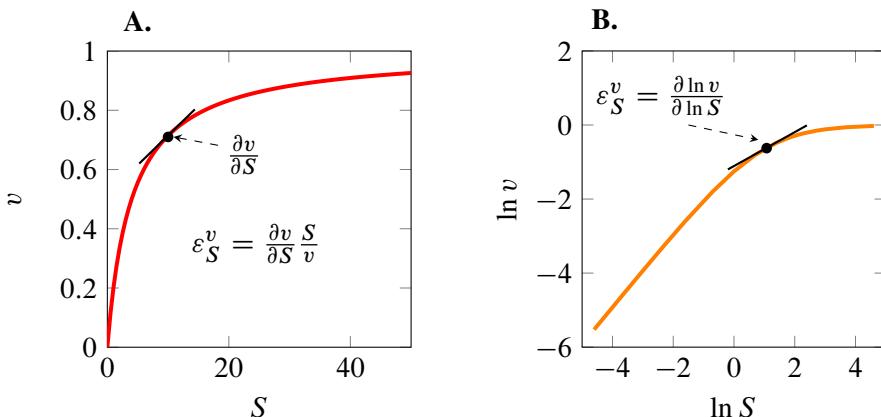


Figure 4.7 **A.** The slope of the reaction rate versus the reactant concentration scaled by both the reactant concentration and reaction rate yields the elasticity, ε_S^v . **B.** If the log of the reaction rate and log of the reactant concentration are plotted, the elasticity can be read directly from the slope of the curve. Curves are generated by assuming $v = S/(2 + S)$.

Consider a variable y to be some function $f(x)$, that is $y = f(x)$. If x increases from x to $(x + h)$ then the change in the value of y will be given by $f(x + h) - f(x)$. The **proportional** change however, is given by:

$$\frac{f(x + h) - f(x)}{f(x)}$$

The **rate of proportional change** at the point x is given by the above expression divided by the step change in the x value, namely h :

Rate of proportional change =

$$\lim_{h \rightarrow 0} \frac{f(x + h) - f(x)}{hf(x)} = \frac{1}{f(x)} \lim_{h \rightarrow 0} \frac{f(x + h) - f(x)}{h} = \frac{1}{y} \frac{dy}{dx}$$

From calculus we know that $d \ln y / dx = 1/y dy/dx$, therefore the rate of proportional change equals:

$$\frac{d \ln y}{dx}$$

and serves as a measure of the rate of *proportional* change of the function y . Just as dy/dx measures the gradient of the curve, $y = f(x)$ plotted on a linear scale, $d \ln y / dx$ measures the slope of the curve when plotted on a semi-logarithmic scale, that is the rate of proportional change. For example, a value of 0.05 means that the curve increases at 5% per unit x .

We can apply the same argument to the case when we plot a function on both x and y logarithmic scales. In such a case, the following result is true:

$$\frac{d \ln y}{d \ln x} = \frac{x}{y} \frac{dy}{dx}$$

Unscaled Elasticities

We can choose not to scale the elasticities in which case we can define the unscaled elasticity as:

$$\mathcal{E}_j^i = \frac{\partial v_i}{x_j} \quad (4.15)$$

We can also write the unscaled elasticity in terms of the scaled elasticity as:

$$\mathcal{E}_j^i = \varepsilon_j^i \frac{v_i}{x_i}$$

Mass-action Kinetics

Computing the elasticities for mass-action kinetics is straight forward. For a reaction such as $v = kS$, we showed earlier (4.5) that $\varepsilon_S^v = 1$. For a generalized irreversible mass-action law such as:

$$v = k \prod S_i^{n_i}$$

the elasticity for species S_i is n_i . For simple mass-action kinetic reactions, the kinetic order and elasticity are therefore identical and independent of species concentration.

For a simple reversible mass-action reaction rate law such as:

$$v = k_1 S - k_2 P \quad (4.16)$$

The elasticities for the substrate and product are given by:

$$\varepsilon_S^v = \frac{k_1 S}{k_1 S - k_2 P} = \frac{v_f}{v} \quad (4.17)$$

$$\varepsilon_P^v = -\frac{k_2 P}{k_1 S - k_2 P} = -\frac{v_r}{v} \quad (4.18)$$

One very important point to note is that as the reaction approaches equilibrium, the elasticities tend to plus infinity and negative infinity, respectively.

Elasticity for Michaelis-Menten Reaction

The irreversible Michaelis-Menten kinetic equation is given by:

$$v = \frac{V_m S}{K_m + S}$$

Substrate Elasticity. To compute the substrate concentration elasticity, ε_S^v , first differentiate the rate equation and then scale by S and v . The derivative of the simple Michaelis-Menten rate law is given by:

$$\frac{\partial v}{\partial S} = \frac{V_m K_m}{(K_m + S)^2}$$

Scaling yields:

$$\varepsilon_S^v = \frac{\partial v}{\partial S} \frac{S}{v} = \frac{K_m}{K_m + S} \quad (4.19)$$

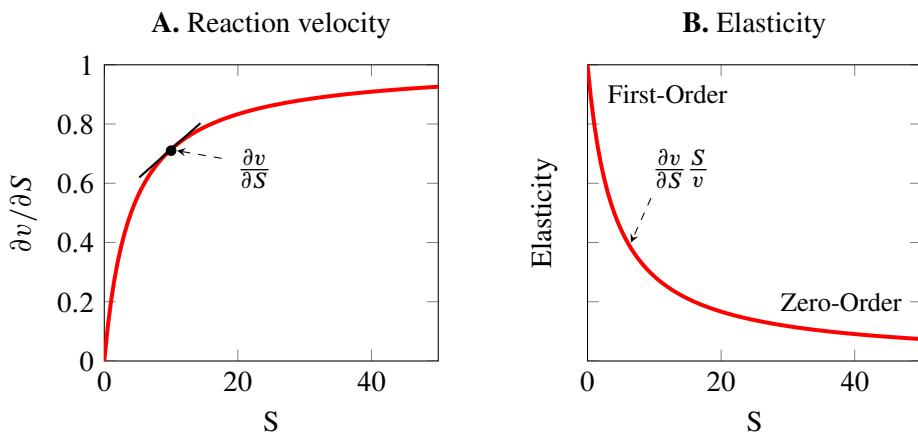


Figure 4.8 **A.** Left panel: the reaction velocity for an irreversible Michaelis-Menten rate law as a function of substrate concentration. The curve is also marked by the slope $\partial v / \partial S$. **B.** Right panel: the substrate elasticity is plotted as a function of substrate concentration. $K_m = 4$ and $V_m = 1$. Note the elasticity starts at one, then decreases to zero as S increases.

The substrate elasticity shows a range of values (Figure 4.8) from **zero** at high substrate concentrations to **one** at low substrate concentrations. When the enzyme is near saturation it is naturally unresponsive to further changes in substrate concentration, hence the elasticity is near zero. The reaction behaves

as a zero-order reaction at this point. When the elasticity is close to one at low S , the reaction behaves with first-order kinetics. In addition, the reaction order changes depending on the substrate concentration.

We can also express the elasticity in terms of the degree of saturation. Recall that the degree of saturation is given by:

$$\frac{ES}{E + ES} = \frac{S}{K_m + S}$$

A slight rearrangement of equation (4.19) allows us to write the elasticity in the following form:

$$\varepsilon_S^v = 1 - \frac{S}{K_m + S} = 1 - \frac{v}{V_m}$$

so that:

$$\varepsilon_S^v = 1 - \text{fractional saturation} \quad (4.20)$$

The elasticity is therefore inversely related to the fractional saturation. If we know the degree of fractional saturation, we can estimate the elasticity.

Enzyme Elasticity. We can also compute the elasticity with respect to enzyme concentration since *in vivo* enzyme concentrations can change. Given that the $V_m = E_t k_{\text{cat}}$ the enzyme elasticity is derived as follows:

$$\frac{\partial v}{\partial E_t} = \frac{k_2 K_m}{K_m + S}$$

Scaling by E_t and v yields:

$$\frac{\partial v}{\partial E_t} \frac{E_t}{v} = \frac{k_2 S}{K_m + S} E_t \frac{K_m + S}{E_t k_2 S} = 1$$

Hence the **enzyme elasticity** is one 4.21.

$$\varepsilon_E^v = 1 \quad (4.21)$$

4.11 General Elasticity Rules

Just as there are rules for differential calculus, there are similar rules for computing elasticities. These rules can be used to simplify the derivation of elasticities for complex rate law expressions. Table 4.1

shows some common elasticity rules, where a designates a constant and x the variable. For example, the first rule says that the elasticity of a constant is zero.

-
1. $\varepsilon(a) = 0$
 2. $\varepsilon(x) = 1$
 3. $\varepsilon(f(x) \pm g(x)) = \varepsilon(f(x)) \frac{f'(x)}{f(x)+g(x)} \pm \varepsilon(g(x)) \frac{g'(x)}{f(x)+g(x)}$
 4. $\varepsilon(x^a) = a$
 5. $\varepsilon(f(x)^a) = a\varepsilon(f(x))$
 6. $\varepsilon(f(x)g(x)) = \varepsilon(f(x)) + \varepsilon(g(x))$
 7. $\varepsilon(f(x)/g(x)) = \varepsilon(f(x)) - \varepsilon(g(x))$
-

Table 4.1 Transformation rules for determining the elasticity of a function a is a constant, x is the dependent variable.

We can illustrate the use of these rules with a simple example. Consider the reversible mass-action rate law (4.16):

$$v = k_1 S - k_2 P$$

To determine the elasticity we first apply rule 3 to give:

$$\varepsilon_S^v = \varepsilon_S(k_1 S) \frac{k_1 S}{k_1 S - k_2 P} - \varepsilon_S(k_2 P) \frac{-k_2 P}{k_1 S - k_2 P}$$

where $\varepsilon_S(f)$ means the elasticity of expression f with respect to variable S .

Now transform the elasticity terms by applying additional rules. Let us apply rule 6 to the expression $\varepsilon_S(k_1 S)$ to give:

$$\varepsilon_S(k_1 S) = \varepsilon_S(k_1) + \varepsilon_S(S)$$

We can now apply rule 1 to the first term on the right and rule 2 to the second term on the right to give:

$$\varepsilon_S(k_1 S) = 0 + 1$$

Since we're evaluating the elasticity of S, P in this situation is a constant, therefore:

$$\varepsilon_S(k_2 P) = \varepsilon_S(k_2) + \varepsilon_S(P) = 0 + 0$$

Combining these results yields:

$$\varepsilon_S^v = \frac{k_1 S}{k_1 S - k_2 P}$$

which corresponds to the first equation in (4.17).

Now consider a simple enzyme kinetic rate equation. One of the most famous is the Michaelis-Menten equation:

$$v = \frac{V_m S}{K_m + S}$$

where V_m is the maximal velocity and K_m the substrate concentration at half maximal velocity.

The elasticity for this equation can be derived by first using the quotient rule (rule 7) which gives:

$$\varepsilon_S^v = \varepsilon(V_m S) - \varepsilon(K_m + S)$$

The rules can now be applied to each of the sub-elasticity terms. For example, we can apply rule 6 to the first term, $\varepsilon(V_m S)$, and rule 3 to the second term, $\varepsilon(K_m + S)$, to yield:

$$\varepsilon_S^v = (\varepsilon(V_m) + \varepsilon(S)) - \left(\varepsilon(K_m) \frac{K_m}{K_m + S} + \varepsilon(S) \frac{S}{K_m + S} \right)$$

Applying rules 1 and 2 allows us to simplify ($\varepsilon(V_m) = 0$; $\varepsilon(K_m) = 0$; $\varepsilon(S) = 1$) the equation to:

$$\varepsilon_S^v = 1 - \left(\frac{S}{K_m + S} \right)$$

or

$$\varepsilon_S^v = \frac{K_m}{K_m + S}$$

Example 4.6

Determine the elasticity expression for the rate laws using log-log rules:

1. $v = k(A + 1)$

Begin with the product rule 6:

$$\varepsilon_A^v = \varepsilon(k) + \varepsilon(A + 1) = \varepsilon(A + 1)$$

Next use the summation rule 3 and rule 2:

$$\begin{aligned} \varepsilon_A^v &= \varepsilon(A + 1) = \varepsilon(A) \frac{A}{A + 1} + \varepsilon(1) \frac{1}{A + 1} \\ &= \frac{A}{A + 1} + 0 = \frac{A}{A + 1} \end{aligned}$$

2. $v = k/(A + 1)$

Begin with the quotient rule 6 followed by Rule 3 and 2:

$$\begin{aligned} \varepsilon_A^v &= \varepsilon(k) - \varepsilon(A + 1) = -\frac{1}{A + 1} \\ &= -\frac{A}{A + 1} \end{aligned}$$

$$3. \ v = A(A + 1)$$

Begin with the quotient rule 6:

$$\varepsilon_A^v = \varepsilon(A) + \varepsilon A + 1$$

Next use Rule 2, 3 and 1:

$$\varepsilon_A^v = 1 + \frac{1}{A+1}$$

To make matters even simpler we can define the elasticity rules using an algebraic manipulation tool such as Mathematica (<http://www.wolfram.com/>) to automatically derive the elasticities [80]. To do this we must first enter the rules in Table 4.1 into Mathematica. The script shown in Figure 4.9 shows the same rules (with a few additional ones) in Mathematica format.

```
(* Define elasticity evaluation rules *)
el[x_, x_] := 1
el[k_, x_] := 0
el[Log[u_, x_]] := el[Log[u], x] = el[u, x]/Log[u]
el[Sin[u_], x_] := el[Sin[u], x] = u el[u, x]Cos[u]/Sin[u]
el[Cos[u_], x_] := el[Sin[u], x] = -u el[u, x]Sin[u]/Cos[u]
el[u_*v_, x_] := el[u*v, x] = el[u, x] + el[v, x]
el[u_/v_, x_] := el[u/v, x] = el[u, x] - el[v, x]
el[u_+v_, x_] := el[u+v, x] = el[u, x]u/(u+v) + el[v, x]v/(u+v)
el[u_-v_, x_] := el[u-v, x] = el[u, x]u/(u-v) - el[v, x]v/(u-v)
el[u_~v_, x_] := el[u~v, x] = v (el[u, x] + el[v, x] Log[u])
```

Figure 4.9 Elasticity rules expressed as a Mathematica script.

The notation $f[x_, y_] := g()$ means define a function that takes two arguments, $x_$ and $y_$. The underscore character in the argument term is essential. Note also the symbol ‘:’ in the assignment operator.

Typing $el[k1 S - k2 P, S]$ into Mathematica will result in the output:

$$k1 S/(-k2 P + k1 S)$$

4.12 Mass-Balance Equations

In the last chapter we briefly considered the various ways in which biochemical networks can be depicted. Ultimately there is the desire to convert a visual map of a biochemical network into a mathematical representation. An increasingly common need is to create quantitative models where

one can either describe the distribution of flows in a network or investigate how the concentration of different species change in time. A quantitative model can be used to study different perturbations, such as knockouts, on the network's phenotype. In order to create such mathematical models we must consider a fundamental principle in biochemical networks which is **mass conservation**.

Consider a simple network comprising two reactions, v_1 and v_2 , with a common species, S . We assume that the first reaction, v_1 , produces S , and the second reaction, v_2 , consumes S (Figure 4.10).



Figure 4.10 Simple Two Step Pathway.

According to the law of conservation of mass, any observed change in the amount of species, S must be due to the difference between the inward rate, v_1 , and outward rate, v_2 . That is, the change in concentration of S is given by the differential equation:

$$\frac{dS}{dt} = v_1 - v_2$$

The above equation is called a **mass-balance equation**. In general for more complex systems such as the one shown in Figure 4.11 where there are multiple inflows and outflows, the mass-balance equation is given by:

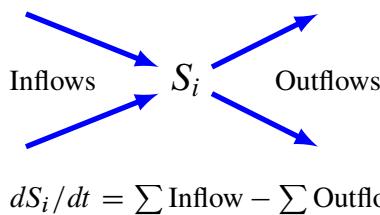


Figure 4.11 Mass Balance: The rate of change in species S_i is equal to the difference between the sum of the inflows and the sum of the outflows.

$$\frac{dS_i}{dt} = \sum \text{Inflows} - \sum \text{Outflows} \quad (4.22)$$

For an even more general representation, we can write the mass-balance equations using stoichiometric coefficients. The rate at which a given reaction, v_j contributes to change in a species, S_i is given by the stoichiometric coefficient of the species, S_i with respect to the reaction, c_{ij} , multiplied by the reaction rate, v_j . That is, a reaction j contributes, $c_{ij} v_j$ rate to changes in species S_i . For a species S_i

with multiple reactions producing and consuming S_i , the mass-balance equation (assuming constant volume conditions) is given by:

$$\frac{dS_i}{dt} = \sum_j c_{ij} v_j \quad (4.23)$$

where c_{ij} is the stoichiometric coefficient for species i with respect to reaction, j . For reactions that consume a species, the stoichiometric coefficient is often negative. (See “Enzyme Kinetics for Systems Biology”, [50]). In considering the simple example in Figure 4.10, the stoichiometric coefficient for S with respect to v_1 is +1 and for v_2 is -1. That is

$$\frac{dS}{dt} = (+1)v_1 + (-1)v_2 = v_1 - v_2$$

The way in which the mass-balance equation is described may seem overly formal, however the formality allows software to automatically convert network diagrams into mass-balance differential equations.

Example 4.7

Consider a more complex linear chain of reactants from S_1 to S_5 shown in Figure 4.12. Write out the mass-balance equations for this simple system.

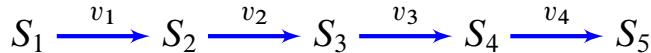


Figure 4.12 Simple Straight Chain Pathway.

$$\begin{aligned} \frac{dS_1}{dt} &= -v_1 & \frac{dS_2}{dt} &= v_1 - v_2 \\ \frac{dS_3}{dt} &= v_2 - v_3 & \frac{dS_4}{dt} &= v_3 - v_4 \\ \frac{dS_5}{dt} &= v_4 \end{aligned} \quad (4.24)$$

Each species in the network is assigned a mass-balance equation which accounts for the flows into and out of the species pool.

For a branched system such as the following:

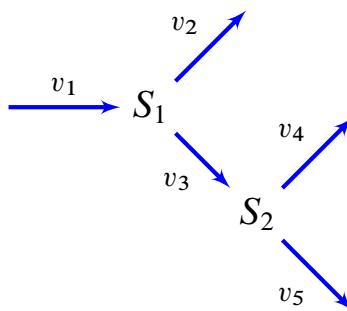


Figure 4.13 Multi-Branched Pathway.

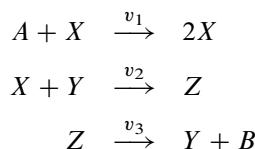
the mass-balance equations are given by:

$$\frac{dS_1}{dt} = v_1 - v_2 - v_3$$

$$\frac{dS_2}{dt} = v_3 - v_4 - v_5$$

Example 4.8

Write out the mass-balance equation for the more complex pathway:



This example is more subtle because we must be careful to take into account the stoichiometry change between the reactant and product side in the first reaction (v_1). In reaction v_1 , the overall stoichiometry for X is +1 because two X molecules are made for every one consumed. Taking this into account, the rate of change of species X can be written as:

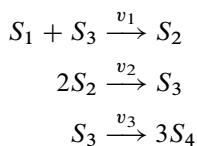
$$\frac{dX}{dt} = -v_1 + 2v_1 - v_2$$

or more simply as $v_1 - v_2$. The full set of mass-balance equations can therefore be written as:

$$\begin{aligned}\frac{dA}{dt} &= -v_1 & \frac{dX}{dt} &= v_1 - v_2 \\ \frac{dY}{dt} &= v_3 - v_2 & \frac{dZ}{dt} &= v_2 - v_3 \\ \frac{dB}{dt} &= v_3\end{aligned}$$

Example 4.9

Write out the mass-balance equation for pathway:



In this example we have non-unity stoichiometries in the second and third reaction steps. The mass-balance equations are given by:

$$\begin{aligned}\frac{dS_1}{dt} &= -v_1 & \frac{dS_2}{dt} &= v_1 - 2v_2 \\ \frac{dS_3}{dt} &= -v_1 + v_2 - v_3 & \frac{dS_4}{dt} &= 3v_3\end{aligned}$$

It is therefore fairly straight forward to derive the balance equations from a visual inspection of the network. Many software tools exist that will assist in this effort by converting network diagrams, either represented visually on a computer screen (for example, pathwayDesigner – www.pathwayDesigner.org) or by providing a text file listing the reactions in the network (for example via Jarnac).

4.13 Stoichiometry Matrix

When describing multiple reactions in a network, it is convenient to represent the stoichiometries in a compact form called the **stoichiometry matrix**, traditionally denoted by **N**. The symbol **N** refers

to number, although some recent researchers use the symbol \mathbf{S} . The stoichiometry matrix is a m row by n column matrix, where m is the number of species and n the number of reactions:

$$\mathbf{N} = m \times n \text{ matrix}$$

The columns of the stoichiometry matrix correspond to the individual chemical reactions in the network, the rows to the molecular species, one row per species. Thus the intersection of a row and column in the matrix indicates whether a certain species takes part in a particular reaction or not, and, according to the sign of the element, whether there is a net loss or gain of substance, and by the magnitude, the relative quantity of substance that takes part in that reaction. The elements of the stoichiometry matrix do not concern themselves with the rate of reaction.

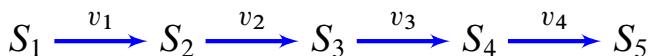
In general, the stoichiometry matrix has the form:

$$\mathbf{N} = \begin{matrix} & \xleftarrow{\quad} & v_j & \xrightarrow{\quad} \\ S_i & \uparrow & \left[\begin{matrix} c_{ij} & \dots & \dots \\ \vdots & & \end{matrix} \right] \\ & \downarrow & \end{matrix}$$

where c_{ij} is the stoichiometry coefficient for the i^{th} species and j^{th} reaction.

Example 4.10

Write out the stoichiometry matrix for the simple chain of reactions which has five molecular species and four reactions as shown below. The four reactions are labeled, v_1 to v_4 .



The stoichiometry matrix for this simple system is given by:

$$\mathbf{N} = \begin{bmatrix} v_1 & v_2 & v_3 & v_4 \\ -1 & 0 & 0 & 0 \\ 1 & -1 & 0 & 0 \\ 0 & 1 & -1 & 0 \\ 0 & 0 & 1 & -1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{matrix} S_1 \\ S_2 \\ S_3 \\ S_4 \\ S_5 \end{matrix}$$

Example 4.11

Write out the stoichiometry matrix for the multibranched pathway shown in Figure 4.13.

$$\mathbf{N} = \begin{bmatrix} v_1 & v_2 & v_3 & v_4 & v_5 \\ 1 & -1 & -1 & 0 & 0 \\ 0 & 0 & 1 & -1 & -1 \end{bmatrix} \begin{matrix} S_1 \\ S_2 \end{matrix}$$

To illustrate that the stoichiometry matrix can be applied to other kinds of networks, let us look at a simple signaling network and two simple gene regulatory networks.

Signaling Networks

Figure 4.14 illustrates a simple protein signaling network, comprising two double phosphorylation cycles coupled by inhibition by protein *C* on the lower double cycle (*D*, *E* and *F*). In this model all species are proteins, and we can assume that protein *A* and *D* are unphosphorylated, *B* and *E* singly phosphorylated, and *C* and *F* doubly phosphorylated. *C* acts as a kinase and phosphorylates *D* and *E*. The reverse reactions, v_2 , v_4 , v_7 and v_8 , are assumed to be catalyzed by phosphatases.

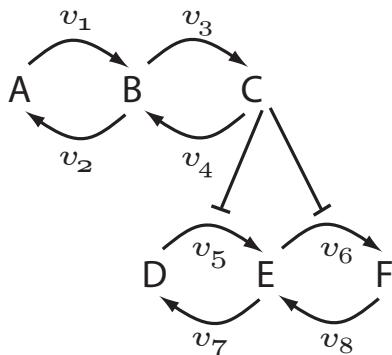


Figure 4.14 Simple Signaling Network. Protein *C* inhibits the activity of reactions v_5 and v_6 .

There is no specified stoichiometric mechanism for the inhibition on v_5 and v_6 . Therefore the stoichiometric matrix will contain no information on this. The stoichiometric matrix for this system will look like:

$$\mathbf{N} = \begin{matrix} & v_1 & v_2 & v_3 & v_4 & v_5 & v_6 & v_7 & v_8 \\ A & -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ B & 1 & -1 & -1 & 1 & 0 & 0 & 0 & 0 \\ C & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 \\ D & 0 & 0 & 0 & 0 & -1 & 0 & 1 & 0 \\ E & 0 & 0 & 0 & 0 & 1 & -1 & -1 & 1 \\ F & 0 & 0 & 0 & 0 & 0 & 1 & 0 & -1 \end{matrix} \quad (4.25)$$

The stoichiometric matrix can be seen to be composed of two separate blocks corresponding to the two cycle layers. It is important to note that whenever there are regulatory interactions in a pathway diagram, these do not appear in the stoichiometry matrix. Instead, such information will reside in the rate law that describes the regulation. If however the mechanism for the regulation is made explicit, then details of the regulation will appear in the stoichiometry matrix. Figure 4.15 will show a simple example of an inhibitor I regulating a reaction, S to P . On the left is displayed the implicit regulatory interaction. All we see is a blunt ended arrow indicating inhibition. In this case, details of the regulation will be found in the rate law governing the conversion of S to P . On the right is displayed an explicit mechanism, a simple competitive inhibition. In this case all details of the mechanism will find its way into the stoichiometry matrix. Figure 4.16 shows a comparison of the implicit and explicit models in terms of the stoichiometry matrix.

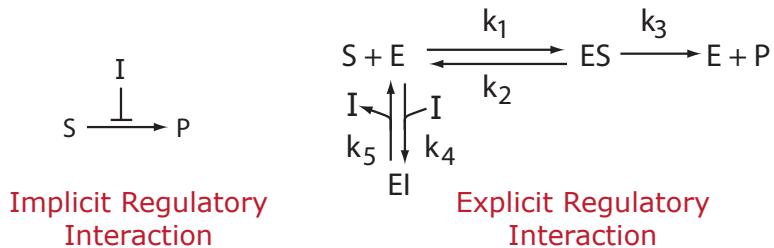


Figure 4.15 Example of implicit and explicit depiction of a regulatory interaction. The left-hand mechanism involving inhibitor I will not appear in the stoichiometry matrix whereas the explicit mechanism will.

Gene Regulatory Networks

Consider a transcription factor P_1 that regulates v_3 by repressing the rate of gene expression (Figure 4.17). In this model we have production of P_1 from reaction v_1 and degradation of P_1 via v_2 . The construction of the stoichiometry matrix will depend on how we represent the regulated step, v_3 . If regulation is implied, i.e. there is no explicit kinetic mechanism, then the regulation will not appear

$$\mathbf{N} = \begin{matrix} S \\ P \\ I \end{matrix} \begin{bmatrix} v_1 \\ -1 \\ 1 \\ 0 \end{bmatrix} \quad \mathbf{N} = \begin{matrix} S \\ P \\ I \\ ES \\ EI \end{matrix} \begin{bmatrix} -1 & 1 & 0 & -1 & 1 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & -1 & 1 \\ 1 & -1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & -1 \end{bmatrix}$$

Figure 4.16 Stoichiometry matrices corresponding to the two models in Figure 4.15.

in the stoichiometry matrix. For the network on the left in Figure 4.17, the stoichiometry matrix is given below:

$$\mathbf{N} = P_1 \begin{bmatrix} v_1 & v_2 \\ 1 & -1 \end{bmatrix} \quad (4.26)$$

The stoichiometry matrix has only one row indicating that there is only one species in the model, P_1 and there is no hint in the stoichiometry matrix that there is regulation.

Imagine the interaction between P_1 and v_3 is made mechanistically explicit. The right hand network in Figure 4.17 shows one possible way to represent the interaction of the transcription factor, P_1 with gene v_3 . In the explicit model the transcription factor P_1 is assumed to bind to a repressor site preventing gene expression.

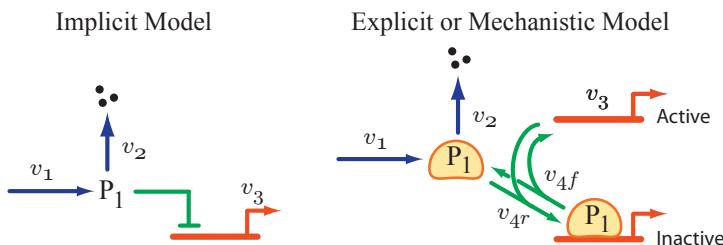


Figure 4.17 Two simple gene regulatory networks involving gene repression. On the left side is the implicit model where P_1 represses v_3 . On the right side is the explicit model showing a more detailed regulation model.

In the explicit model there are two new species, designated active gene and inactive gene. The stoichiometry matrix will therefore include two additional rows corresponding to these two species. The

stoichiometry matrix for the explicit model is:

$$\mathbf{N} = \begin{matrix} P_1 \\ P_1(\text{Active}) \\ P_1(\text{InActive}) \end{matrix} \left[\begin{array}{cccc} v_1 & v_2 & v_{4r} & v_{4f} \\ 1 & -1 & -1 & 1 \\ 0 & 0 & -1 & 1 \\ 0 & 0 & 1 & -1 \end{array} \right] \quad (4.27)$$

4.14 The System Equation

Equation 4.23, which describes the mass balance equation, can be reexpressed in terms of the stoichiometry matrix to form the **system equation**.

$$\frac{d\mathbf{x}}{dt} = \mathbf{Nv(x, p)} \quad (4.28)$$

\mathbf{N} is the $m \times n$ stoichiometry matrix and \mathbf{v} is the n dimensional rate vector, whose i th component gives the rate of reaction i as a function of the species concentrations. \mathbf{x} is the m vector of species. The term \mathbf{p} represents the vector of parameters and inputs that can affect the reaction rate.

Looking again at the simple chain of reactions in Figure 4.12, the system equation can be written as:

$$\frac{d\mathbf{x}}{dt} = \mathbf{Nv(x, p)} = \left[\begin{array}{cccc} -1 & 0 & 0 & 0 \\ 1 & -1 & 0 & 0 \\ 0 & 1 & -1 & 0 \\ 0 & 0 & 1 & -1 \\ 0 & 0 & 0 & 1 \end{array} \right] \left[\begin{array}{c} v_1 \\ v_2 \\ v_3 \\ v_4 \end{array} \right] \quad (4.29)$$

If stoichiometry matrix is multiplied into the rate vector, the mass-balance equations shown earlier (4.24) are recovered.

All stoichiometric interactions are placed in the stoichiometry matrix. The example shown in Figure 4.14 and Figure 4.17 illustrated non-stoichiometric interactions, namely two inhibition interactions from C to reactions v_5 and v_6 and repression on v_3 by P_1 . As was noted, these interactions do not occur in the stoichiometry matrix. Instead, they will be found in the rate vector, \mathbf{x} , in the form of a particular rate law.

The stoichiometry matrix represents the mass transfer connectivity of the network and contains information on the network's structural characteristics. These characteristics fall into two groups, relationships among the species and relationships among the reaction rates.

4.15 Building Simple Gene Regulatory Models

In subsequent chapters we will be using simple gene regulatory network models to illustrate various concepts. In order to prepare the reader, a brief summary of building such models is given here. Figure 4.18 shows a single gene expressing a protein x at a rate v_1 .

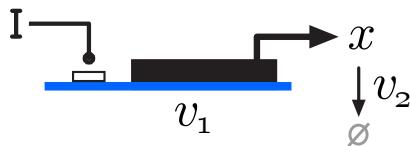


Figure 4.18 Single Gene Cassette with Gene Expression and Degradation.

Gene expression is induced by inducer, I . Protein x is degraded at a rate, v_2 . This simple unit is termed a **gene cassette**. We can write down the differential equation that describes the rate of change of x as follows:

$$\frac{dx}{dt} = v_1 - v_2$$

In this equation we have not specified the specific rate laws for v_1 or v_2 . These decisions will be based on specific biological knowledge. In this example we can make the following assumptions. The first is that v_2 is a first-order reaction such that $v_2 = kx$ where k is a first-order rate constant. The second assumption is that the rate of gene expression is a sigmoidal function of the inducer concentration. The most commonly used sigmoid function in gene expression models is the Hill equation. This is described in detail in the companion textbook Enzyme Kinetic for Systems Biology [50]. For a positive inducer the Hill equation takes the form:

$$v = \frac{V_m \text{Ind}^h}{K_d + \text{Ind}^h}$$

where V_m is the maximal expression rate, Ind the concentration of inducer, and K_d the dissociation constant for inducer binding to the operator site. h is called the Hill coefficient and determines the sigmoidicity of the response. Figure 4.19 shows how the Hill equation behaves as a function of inducer and different values for the Hill coefficient.

The Hill equation can also be used to model repression. This requires only a simple modification where the induction term in the numerator is removed:

$$v = \frac{V_m}{K_d + \text{Ind}^h}$$

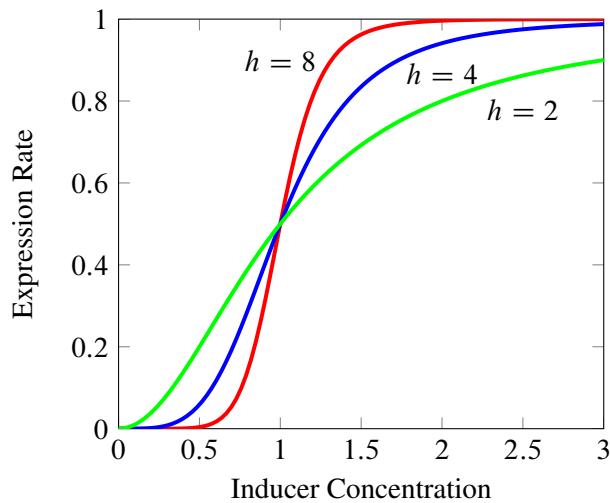


Figure 4.19 Plot showing the response Hill equation with respect to inducer concentration and different Hill coefficients. $K_d = 1$.

A more complex model is shown in Figure 4.21 where we see a cascade of two gene cassettes. This system has two proteins expressed, x_1 and x_2 . x_1 expression is induced by X_o , and x_2 expression is induced by x_1 . Both x_1 and x_2 are degraded. The differential equations for this model are given by:

$$\frac{dx_1}{dt} = v_1 - v_2$$

$$\frac{dx_2}{dt} = v_3 - v_4$$

If we assign first-order kinetics to the degradation steps and Hill equations to the gene expression steps, we obtain:

$$\frac{dx_1}{dt} = \frac{V_{m1} X_o^h}{K_{d1} + X_o^h} - k_1 x_1$$

$$\frac{dx_2}{dt} = \frac{V_{m2} x_1^h}{K_{d2} + x_1^h} - k_2 x_2$$

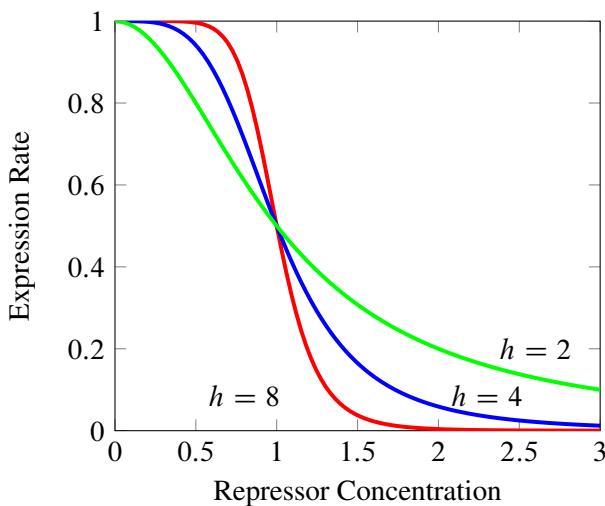


Figure 4.20 Plot showing the response Hill equation with respect to repressor concentration and different Hill coefficients. $K_d = 1$.

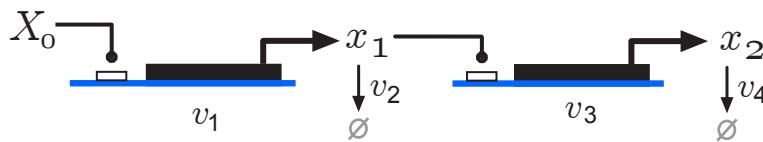


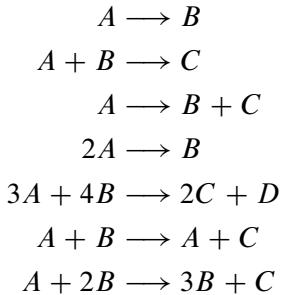
Figure 4.21 Two genes in a cascade formation.

Further Reading

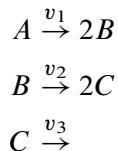
1. Bray D (2011) Wetware: A Computer in Every Living Cell. Yale University Press. ISBN: 978-0300167849
2. Goodsell D S (2009) The machinery of life. Springer, 2nd edition. ISBN 978-0387849249
3. Phillips R, Kondev J and Theriot J (2010) Physical Biology of the Cell. Garland Science. ISBN 978-0-8153-4163-5
4. Sauro HM (2011) Enzyme Kinetics for Systems Biology. ISBN: 978-0982477311

Exercises

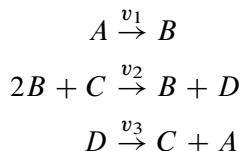
1. Explain the difference between the terms: Stoichiometric amount, Stoichiometric coefficient, rate of change (dX/dt) and reaction rate (v_i).
2. Determine the stoichiometric amount and stoichiometric coefficient for each species in the following reactions:



3. Derive the set of differential equations for the following model in terms of the rate of reaction, v_1 , v_2 and v_3 :

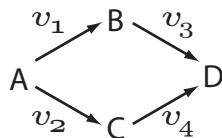


4. Derive the set of differential equations for the following model in terms of the rate of reaction, v_1 , v_2 and v_3 :

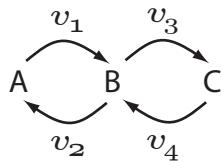


5. Write out the stoichiometry matrix for the networks in question 3 and 4.
6. Derive the stoichiometry matrix for each of the following networks. In addition, write out the mass-balance equations in each case.

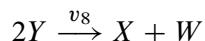
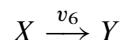
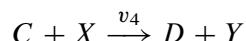
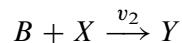
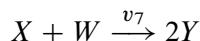
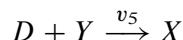
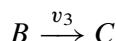
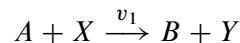
(a)



(b)



(c)



7. A gene G_1 expresses a protein p_1 at a rate v_1 . p_1 forms a tetramer (4 subunits), called p_1^4 at a rate v_2 . The tetramer negatively regulates a gene G_2 . p_1 degrades at a rate v_3 . G_2 expresses a protein, p_2 at a rate v_9 . p_2 is cleaved by an enzyme at a rate v_4 to form two protein domains, p_2^1 and p_2^2 . p_2^1 degrades at a rate v_5 . Gene G_3 expresses a protein, p_3 at a rate v_6 . p_3 binds to p_2^2 forming an active complex, p_4 at a rate v_{10} , which can activate G_1 . p_4 degrades at a rate v_7 . Finally, p_2^1 can form a complex, p_5 , with p_4 at a rate v_8 .

- (a) Draw the network represented in the description given above.
- (b) Write out the differential equation for each protein species in the network in terms of v_1, v_2, \dots
- (c) Write out the stoichiometric matrix for the network.

8. Write out the differential equations for the system depicted in equation 4.29.
9. Build a differential equation computer model of the system shown in Figure 4.3. Assume the following rate laws for the four rates:

$$\begin{aligned}v_1 &= V_{m1}X_o^h/(K_1 + X_o^h) \\v_2 &= K_2S_1 \\v_3 &= V_{m2}S_1^h/(K_3 + S_1^h) \\v_4 &= K_4S_2\end{aligned}$$

where h is the Hill coefficient, V_{mx} the maximal gene expression rate and K_i various kinetic constants. Assume the following values:

$$K_1 = 100; K_2 = 0.6; K_3 = 20; K_4 = 2.5$$

$$V_{m1} = 4; V_{m2} = 15; h = 1$$

- a) Plot the steady state values for S_1 and S_2 as a function of inducer molecule X_o . Vary X_o between zero and 50.
- b) Repeat question a) but set $h = 8$ and now vary X_o between zero and 3.0
- c) What does h do in the expression rate laws?
- d) What effect does h have on the response of S_2 to X_o .
- e) From your answer in d), what advantage can you see in using two stages rather than one?

5

The Steady State

“Steady as she goes, Number One. ”

– Captain Ericson in The Cruel Sea, 1953

5.1 System States

As we proceed through the book we will encounter different kinds of behavior. At this stage however it is worth describing the states that are fundamental to all systems. These states fall into three groups: (thermodynamic) equilibrium, steady state, and transients. In the literature the terms equilibrium and steady state are often used to mean the same thing, but here they will be used to describe two very different states. The simplest and arguably the least interesting is equilibrium, or more precisely thermodynamic equilibrium.

5.2 Equilibrium

Thermodynamic equilibrium, or simply equilibrium, refers to the state of a system when all energy gradients are dissipated. Thermodynamically it means that the entropy of the system has reached its maximum and is constant. In chemistry, thermodynamic equilibrium is when all forward and reverse rates are equal. This also means that the concentration of chemical species is unchanging and all net flows are zero. Equilibrium is easily achieved in a closed system. For example, consider the simple

chemical isomerization reaction:



This system is closed because x_1 and x_2 cannot exchange mass with the surrounding. Let the net forward rate of the reaction, v , be equal to $v = k_1 x_1 - k_2 x_2$. The rates of change of x_1 and x_2 are given by:

$$\frac{dx_1}{dt} = -v \quad \frac{dx_2}{dt} = v$$

At equilibrium $v = 0$, therefore dx_1/dt and dx_2/dt both equal zero. The solution to the differential equations can be derived as follows. Given that the system is closed we know that the total mass in the system, $x_1 + x_2$, is constant. This constant is given by the sum of the *initial concentrations* of x_1 and x_2 which we will define as $x_1^0 + x_2^0$. We assume that the volume is constant and set to unit volume, allowing us to state that the sum of the concentrations is conserved. The differential equation for x_1 is given by:

$$\frac{dx_1}{dt} = k_2 x_2 - k_1 x_1$$

Let us replace x_2 by the term $x_1^0 + x_2^0 - x_1$ to yield:

$$\frac{dx_1}{dt} = k_2 x_1^0 + k_2 x_2^0 - k_2 x_1 - k_1 x_1 = k_2(x_1^0 + x_2^0) - x_1(k_1 + k_2)$$

We now have an equation only in terms of x_1 and a set of parameters. The easiest way to solve this equation is to use Mathematica or Maxima. The Mathematica command is:

```
DSolve[X1'[t]==k2 (X1o + X2o) - X1[t] (k1 + k2), X1[0]==X1o, X1[t], t]
```

where $X1[0] == X1o$ sets the initial condition for the concentration of x_1 to be x_1^0 . By implication, the initial condition for x_2^0 is $(x_1^0 + x_2^0) - x_1^0 = x_2^0$. The result of applying Mathematica yields the following solution:

$$x_1(t) = \frac{(x_1^0 + x_2^0)k_2}{k_1 + k_2} + \frac{e^{-(k_1+k_2)t} v_{\text{initial}}}{k_1 + k_2}$$

The first term in the solution is the equilibrium concentration of x_1 . The second term is an adjustment to the concentration over time as a result of the non-equilibrium initial conditions which tends to zero at infinite time. In later chapters we will see this kind of structure appear again and again because it is characteristic of linear systems.

At $t = 0$, the second term is equal to $v_{\text{initial}}/(k_1 + k_2)$ where v_{initial} is the net reaction rate of the reversible reaction at the initial conditions, that is $k_1 x_1^0 - k_2 x_2^0$. At $t = 0$, the value of the second term must be the difference between the initial concentration S_1^0 and the equilibrium concentration, x_{1eq} . The second term also has an exponential component which approaches zero as time goes to infinity. At this point we are left with the first term which equals the concentration of x_1 when $dx_1/dt = dx_2/dt = 0$.

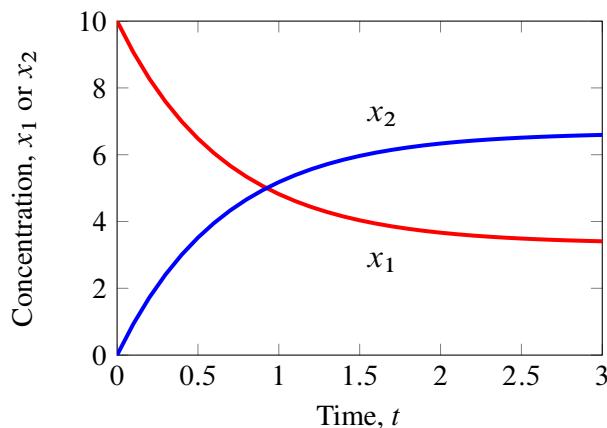


Figure 5.1 Time course for equilibration of the reversible reaction, $x_1 \xrightleftharpoons[k_2]{k_1} x_2$ where $k_1 = 1, k_2 = 0.5, x_1^o = 10, x_2^o = 0$. The ratio of the equilibrium concentration is given by k_1/k_2 . Python model listing: (5.1).

At equilibrium the reaction rate can be computed by substituting the equilibrium concentration of x_1 and x_2 into the reaction rate, $v = k_2 x_2 - k_1 x_1$. The equilibrium concentration of x_1 is given by:

$$x_{1eq} = \frac{(x_1^o + x_2^o)k_2}{k_1 + k_2}$$

The equilibrium concentration of x_2 can be obtained by subtracting x_{1eq} from $x_1^o + x_2^o$. When the x_{1eq} and x_{2eq} relations are substituted into v , the result is:

$$v = 0$$

From this somewhat long-winded analysis, it has been determined for the closed reversible system, at infinite time, the concentrations of x_1 and x_2 reach some constant value and that the net rate, v , is zero. The system is therefore at thermodynamic equilibrium.

In biochemical models it is often assumed that when the forward and reverse rates for a particular reaction are very fast compared to the surrounding reactions, the reaction is assumed to be in **quasi-equilibrium**. That is, although the entire system may be out of equilibrium, there may be parts of the system that can be approximated as though they were in equilibrium. This is often done to simplify the modeling process. Living organisms are not themselves at thermodynamic equilibrium, if they were then they would technically be dead. Living systems are open so that there is a continual flow of mass and energy across the system's boundaries.

5.3 Transients

The second behavior that a system can show is a transient. A transient is usually the change that occurs in the species concentrations as the system moves from one state, often a steady state, to another. Equation 5.3 shows the solution to a simple system (5.2) that describes the transient behavior of species x_1 and x_2 . Figure 5.2 illustrates the transient from an initial non-steady state condition to a steady state.

To convert the simple reversible model described in the last section into an open system, we only need to add a source reaction and a sink reaction as shown in the following scheme:



In this case simple mass-action kinetics is assumed for all reactions. It is also assumed that the source reaction, with rate constant, k_o , is irreversible and originates from a boundary species, X_o . X_o is **fixed**. Although X_o is being constantly consumed, there is some outside mechanism that maintains X_o constant. The rate of reaction of the source reaction (k_o) is therefore fixed at $k_o X_o$ which we will call v_o .

In addition it is assumed that the sink reaction, with rate constant, k_3 is also irreversible. For the purpose of making it easier to derive the time course solution, the reverse rate constant, k_2 will be assumed to equal zero and we will set the initial conditions for x_1 and x_2 to both equal zero. The mathematical solution for the system under these conditions is given by equation 5.3:

$$x_1(t) = v_o \frac{1 - e^{-k_1 t}}{k_1}$$
(5.3)

$$x_2(t) = v_o \frac{k_1 (1 - e^{-k_3 t}) + k_3 (e^{-k_1 t} - 1)}{k_3 (k_1 - k_3)}$$

Note that $v_o = k_o X_o$. As t tends to infinity, $x_1(t)$ tends to v_o/k_1 , and $x_2(t)$ tends to v_o/k_3 . The reaction rate through each of the three reaction steps is v_o . This can be confirmed by substituting the solutions for x_1 and x_2 into the reaction rate laws. Given that v_o is greater than zero and that x_1 and x_2 reach constant values given sufficient time, we conclude that this system eventually settles to a steady state rather than thermodynamic equilibrium. The system displays a continuous flow of mass from the source to the sink. This can continue undisturbed so long as the source material, X_o , never runs out and that reaction k_3 is irreversible. Figure 5.2 shows a simulation of this system.

Thermodynamic Equilibrium and Steady State

Thermodynamic equilibrium and the steady state are distinct states of a chemical system. In equilibrium, both the rate of change of species and the net flow of mass through the system is zero. That is:

$$\frac{d\mathbf{x}}{dt} = 0$$

for all i : $v_i = 0$

where v_i is the net reaction rate for the i^{th} reaction step. At equilibrium there is therefore no dissipation of gradients or energy fields. When a biological system is at equilibrium, we say it is dead.

The steady state has some similarities with equilibrium except there is a net flow through the system such that gradients and energy fields are continuously being dissipated. This also means that one or more v_i 's must be non-zero.

The steady state is defined when all dx_i/dt are equal to zero while one or more reaction rates are non-zero:

$$\frac{d\mathbf{x}}{dt} = 0$$

$$v_i \neq 0$$

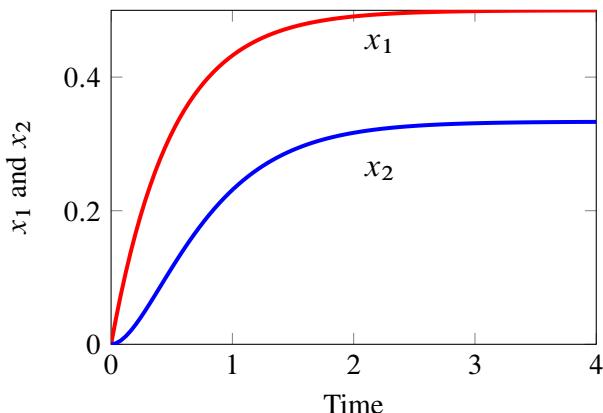


Figure 5.2 Time course for an open system reaching steady state. $X_o \xrightarrow{k_o} x_1 \xrightarrow{k_1} x_2 \xrightarrow{k_3}$ where $v_o = 1, k_1 = 2, k_3 = 3, x_{1o} = 0, x_{2o} = 0$. X_o is assumed to be fixed. Python model listing: (5.2).

5.4 Steady State

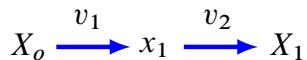
In the previous section we saw a transient system evolve towards a state called the steady state. In a dynamical system, the **steady state** is one of the most important states to consider. In the literature it is also sometimes referred to as the stationary solution or state, singular points, fixed points, or even equilibrium. We will avoid the use of the term equilibrium because of possible confusion with thermodynamic equilibrium.

The steady state is the primary reference point from which to consider a model's behavior. At steady state the concentrations of all molecular species are constant, and there is a net flow of mass through the network. This is in contrast to systems at thermodynamic equilibrium, where, although concentrations are constant, there is no net flow of mass across the system's boundaries.

The steady state is where the rates of change of all species, dS/dt are zero but at the same time the net rates are non-zero, that is $v_i \neq 0$. This situation can only occur in an open system, that is a system that can exchange matter with the surroundings. Thermodynamically a system is at steady state when the entropy production of the *system* is a constant zero or even negative. Note that the entropy of the surrounding will be positive such that the net production of entropy in the system and environment is positive. This later requirement satisfies the second law of thermodynamics.

Graphical Procedure

We can also illustrate the steady state using a graphical procedure. Consider the simple model below:



where X_o and X_1 are fixed boundary species and x_1 is a species that can change. For illustration purposes we will assume some very simple kinetics for each reaction, v_1 and v_2 . Let us assume that each reaction is governed by simple first order mass-action kinetics:

$$\begin{aligned} v_1 &= k_1 X_o \\ v_2 &= k_2 x_1 \end{aligned}$$

where k_1 and k_2 are both first-order reaction rate constants. In Figure 5.3 both reaction rates have been plotted as a function of the floating species concentration, x_1 .

Note that the reaction rate for v_1 is a horizontal line because it is unaffected by changes in x_1 (no product inhibition). The second reaction, v_2 , is shown as a straight line with slope, k_2 . Notice that the lines intersect. The intersection marks the point when both rates, v_1 and v_2 , are equal. This point marks the steady state concentration of x_1 . By varying the value of k_2 we can observe the effect it has on the steady state. For example, Figure 5.3 shows that as we decrease k_2 , the concentration of

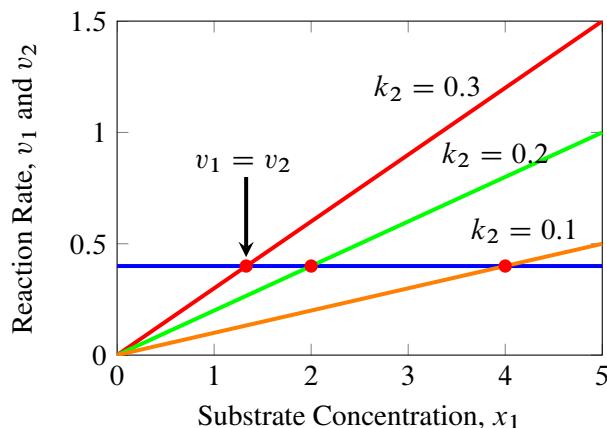


Figure 5.3 Plot of reaction rates versus concentration of x_1 and different values for k_2 for the system $X_o \rightarrow x_1 \rightarrow X_1$. The intersection of the two lines marks the steady state point where $v_1 = v_2$. $X_o = 1, k_1 = 0.4$. Note that as k_2 is decreased, the steady state level of x_1 increases.

x_1 increases. This should not be difficult to understand, as k_2 decreases, the activity of reaction v_2 also decreases. This causes x_1 to build up in response.

In this simple model it is also straightforward to determine the steady state of x_1 mathematically which amounts to finding a mathematical equation that represents the intersection point of the two lines. The model for this system comprises a single differential equation:

$$\frac{dx_1}{dt} = k_1 X_o - k_2 x_1$$

At steady state set $dx_1/dt = 0$, from which we can solve for the steady state concentration of x_1 as:

$$x_1 = \frac{k_1 X_o}{k_2} \quad (5.4)$$

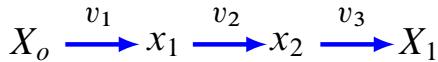
This solution tells us that the steady state concentration of x_1 is a function of *all* the parameters in the system. We can also determine the steady state rate, usually called the **pathway flux** and denoted by J , by inserting the steady state value of x_1 into one of the rate laws, for example into v_2 :

$$J = k_2 \frac{k_1 X_o}{k_2} = k_1 X_o$$

This answer is identical to v_1 which is not surprising since in this model, the pathway flux is completely determined by the first step. The second step has no influence whatsoever on the flux. This simple example illustrates a rate limiting step in the pathway, that is one step, and one step only, that has complete control over the pathway flux. More realistically the first step would either be reversible or be subject to product inhibition. In this case rate limitingness is distributed between the first and second step.

More Complex Model

A more complicated model is the following:



where the rate law for the first step is now reversible and given by:

$$v_1 = k_1 X_o - k_2 x_1$$

The remaining steps are governed by simple irreversible mass-action rate laws, $v_2 = k_3 x_1$ and $v_3 = k_4 x_2$. The differential equations for this system are:

$$\begin{aligned}\frac{dx_1}{dt} &= (k_1 X_o - k_2 x_1) - k_3 x_1 \\ \frac{dx_2}{dt} &= k_3 x_1 - k_4 x_2\end{aligned}$$

The steady state solution for x_1 and x_2 can be obtained by setting both differential equations to zero to yield:

$$\begin{aligned}x_1 &= \frac{k_1 X_o}{k_2 + k_3} \\ x_2 &= \frac{k_3 k_1 X_o}{(k_2 + k_3) k_4}\end{aligned}$$

The steady state flux, J , can be determined by inserting one of the solutions into the appropriate rate law. The easiest is to insert the steady state level of x_2 into v_3 to yield:

$$J = \frac{k_3 k_1 X_o}{k_2 + k_3}$$

Once the first step is reversible, we see that the steady state flux is a function of all the parameters except k_4 , indicating that the first step is no longer the rate limiting step. The equation shows us that the ability to influence the flux is shared between the first and second steps. There is no rate limiting step in this pathway. Note that if we set $k_2 = 0$, the solution reverts to the earlier, simpler model.

We can also make all three steps reversible ($k_f S_i - k_r S_{i+1}$), so that the solution is given by:

$$\begin{aligned}x_1 &= \frac{X_o k_1 (k_4 + k_5) + X_1 k_4 k_6}{k_3 k_5 + k_2 (k_4 + k_5)} \\ x_2 &= \frac{X_1 k_6 (k_2 + k_3) + X_o k_1 k_3}{k_3 k_5 + k_2 (k_4 + k_5)}\end{aligned}$$

The last example illustrates the increase in complexity of deriving a mathematical solution after only a modest increase in model size. In addition, once more complex rate laws as used, such as Hill equations or Michaelis-Menten type rate laws, the solutions become exceedingly difficult to derive. As a result, steady states tend to be computed numerically rather than analytically.

5.5 Computing the Steady State

In those (many) cases where we cannot derive an analytical solution for the steady state, we must revert to numerical methods. There are at least two methods that can be used. The simplest approach is to run a time course simulation for a sufficiently length of time so that the time course trajectories eventually reach the steady state. This method works as long as the steady state is stable; it cannot be used to locate unstable steady states because such trajectories diverge (See Chapter 14). In addition, the method can sometimes be very slow to converge depending on the kinetics of the model. As a result, many simulation packages provide an alternate method for computing the steady state where the model differential equations are set to zero and the resulting equations solved for the concentrations. This type of problem is quite common in many fields and is often represented mathematically as the quest to find solutions to equations of the following form:

$$f(x, p) = 0 \quad (5.5)$$

where x is the unknown, and p is one or more parameters in the equations. All numerical methods for computing solutions to equation 5.5 start with an initial estimate for the solution, say x_o . The methods are then applied iteratively until the estimate converges to the solution. One of the most well known methods for solving equation 5.5 is called the **Newton-Raphson method**. It can be easily explained using a geometric argument, Figure 5.4. Suppose x_1 is the initial guess for the solution to equation 5.5. The method begins by estimating the slope of equation 5.5 at the value x_1 , that is df/dx . A line is then drawn from the point $(x_1, f(x_1))$, with slope df/dx until it intersects the x axis. The intersection, x_2 , becomes the next guess for the method. This procedure is repeated until x_i is sufficiently close to the solution. For brevity the parameter p is omitted from the following equations. From the geometry shown in Figure 5.4, one can write down the slope of the line, $\partial f / \partial x_1$ as:

$$\frac{\partial f}{\partial x_1} = \frac{f(x_1)}{x_1 - x_2}$$

This can be generalized to:

$$\frac{\partial f}{\partial x_k} = \frac{f(x_k)}{x_k - x_{k+1}}$$

or by rearrangement:

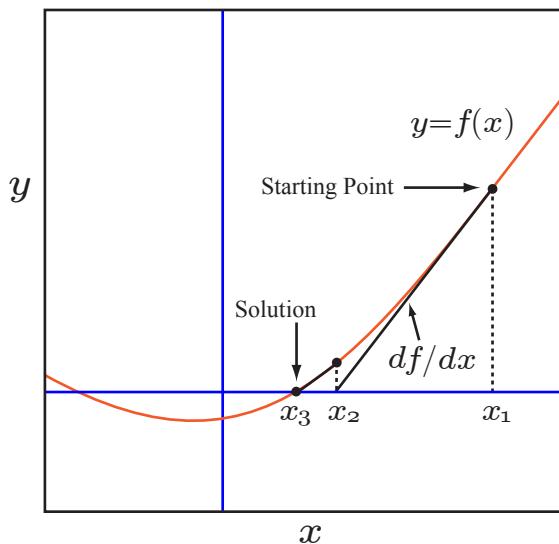


Figure 5.4 The geometry of Newton-Raphson's method.

$$x_{k+1} = x_k - \frac{f(x_k)}{\frac{df}{dx_k}} \quad (5.6)$$

In this form (5.6) we see the iterative nature of the algorithm.

Before electronic calculators had a specific square root button, calculator users would exploit the Newton method to estimate square roots. For example, if the square root of a number a is equal to x , that is $\sqrt{a} = x$, then it is true that:

$$x^2 - a = 0$$

This equation looks like an equation of the form 5.5 where we wish to estimate the value for x . We can apply the Newton formula (equation 5.6) to this equation to obtain:

$$x_{k+1} = \frac{1}{2} \left(x_k + \frac{a}{x_k} \right) \quad (5.7)$$

Table 5.1 shows a sample calculation using this equation to compute the square root of 25. Note that only a few iterations are required to reach convergence.

One important point to bear in mind, the Newton-Raphson method is not guaranteed to converge to the solution, this depends heavily on the starting point and the nature of the system being solved. In the case when convergence fails it is often useful to halt the method after a maximum of iterations, say one hundred iterations. In a case like this, a new initial start is given and the method tried again. In biochemical models we can always run a time course simulation for a short while and use the end

Iteration	Estimate
0	15
1	8.33333
2	5.666
3	5.0392
4	5.0001525
5	5.0

Table 5.1 Newton method used to compute the square root of 25, using equation 5.7 with a starting value of 15.

point of that as the starting point for the Newton method. This approach is much more reliable. There are various ways to decide whether convergence has been achieved. Two such tests include:

1. Difference between Successive Solution Estimates. We can test for the difference between solution x_i and the next estimate, x_{i+1} . If the absolute difference $|x_i - x_{i+1}|$ is below some threshold, then we assume convergence has been achieved. Alternatively, we can check whether the relative error is less than a certain threshold (say, 1%). The relative error is given by:

$$\epsilon = \frac{x_{i+1} - x_i}{x_{i+1}} \times 100\%$$

The procedure can be made to stop at the i -th step if $|f(x_i)| < \epsilon_f$ for a given ϵ_f .

2. Difference between Successive dS_i/dt Estimates. Here we estimate the rates of change as the iteration proceeds, and assume convergence has been achieved when the difference between two successive rates of change are below some threshold. If we are dealing with a model with more than one state variable, then we can construct the sum of squares of the rates of change:

$$\sum \left(\frac{dx_i}{dt} \right)^2$$

The Newton method can be easily extended to systems of equations so that we can write it in matrix form as shown in (5.8).

Matrix form of the Newton-Raphson Method:

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \left[\frac{\partial \mathbf{f}(\mathbf{x})}{\partial \mathbf{x}} \right]^{-1} \mathbf{f}(\mathbf{x}_k) \quad (5.8)$$

If m is the number of state variables or floating species in the model, then \mathbf{x}_k is a m dimensional vector of species concentrations, $\mathbf{f}(\mathbf{x})$ is a vector containing the m rates of change, and $\partial\mathbf{f}(\mathbf{x})/\partial\mathbf{x}$ is a square $m \times m$ matrix called the **Jacobian matrix**.

Newton Algorithm

1. Initialize the values of the concentrations, \mathbf{x} , to some initial guess obtained perhaps from a short time course simulation.
2. Compute the values for $\mathbf{f}(\mathbf{x})$, that is the left-hand side of the differential equation ($d\mathbf{x}/dt$).
3. Calculate the matrix of derivatives, $\partial\mathbf{f}/\partial\mathbf{x}$, that is $d(d\mathbf{x}/dt)/d\mathbf{x}$, at the current estimate for \mathbf{x} .
4. Compute the inverse of the matrix, $\partial\mathbf{f}/\partial\mathbf{x}$.
5. Using the information calculated so far, compute the next guess, \mathbf{x}_{k+1} .
6. Compute the sum of squares of the new value of $\mathbf{f}(\mathbf{x})$ at \mathbf{x}_{k+1} . If the value is less than some error tolerance, assume the solution has been reached, else return to step 3 using \mathbf{x}_{k+1} as the new starting point.

Although the Newton method is seductively simple, it requires the initial guess to be sufficiently close to the solution in order for it to converge. In addition, convergence can be slow or may not occur at all. A common problem is that the method can overshoot the solution and will then begin to rapidly diverge.

One way to provide a good starting point for the Newton method is to first use a short time course simulation to bring the initial estimate closer to the steady state. The assumption here is that the steady state is stable. The final point computed in the time course is used to seed a Newton-like method. If the Newton method fails to converge, a second time course simulation is carried out. This can be repeated as many times as desired. If there is a suspicion that the steady state is unstable, one can also run a time course simulation backwards in time. In general however, there is no sure way of computing the steady state automatically, and sometimes human intervention is required to supply good initial estimates.

Jacobian Matrix

The Jacobian is a common matrix used in many fields especially control theory and dynamical systems theory. We will frequently use it in this book. Given a set of equations:

$$\begin{aligned} y_1 &= f_1(x_1, \dots, x_n) \\ y_2 &= f_2(x_1, \dots, x_n) \\ &\vdots \\ y_m &= f_m(x_1, \dots, x_n) \end{aligned} \tag{5.9}$$

The Jacobian matrix is defined as the matrix of partial differentials:

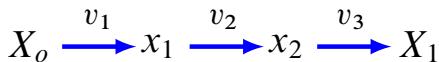
$$\mathbf{J} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_m}{\partial x_1} & \cdots & \frac{\partial f_m}{\partial x_n} \end{bmatrix}$$

As a result of these issues, the unmodified Newton method is rarely used in practice for computing the steady state of biochemical models. One common variant called the Damped Newton method is sometimes used. Both Gepasi [38] [48] and SCAMP use the Damped Newton method for computing the steady state. This method controls the derivative, df/dx , by multiplying the derivative by a factor α , $0 < \alpha < 1$ and can be used to prevent overshoot. There are many variants on the basic Newton method and good simulation software will usually have reasonable methods for estimating the steady state.

In the past ten years, more refined Newton like methods have been devised. One that is highly recommended is NLEQ2 (<http://www.zib.de/en/numerik/software/ant/nleq2.html>). This is used by both Jarnac [49], libRoadRunner [58] and PySCeS [42] for computing the steady state. The stiff solver suite sundials (<https://computation.llnl.gov/casc/sundials/main.html>) also incorporates an equation solver, however experience has shown that it doesn't converge as well as NLEQ2. More recent versions may have improved this.

Solving the Steady State for a Simple Pathway

We are going to illustrate the use of the Newton-Raphson method to solve the steady state for the following simple pathway. Assume that all three reactions are governed by simple mass-action reversible rate laws. Species X_o and X_1 are assumed to be fixed, and only x_1 and x_2 are floating species or state variables.



The differential equations for the model are as follows:

$$\begin{aligned}\frac{dx_1}{dt} &= (k_1 X_o - k_2 x_1) - (k_3 x_1 - k_4 x_2) \\ \frac{dx_2}{dt} &= (k_3 x_1 - k_4 x_2) - (k_5 x_2 - k_6 X_1)\end{aligned}\tag{5.10}$$

The values for the rate constants and the boundary conditions are given in Table 5.2.

Parameter	Value
k_1	3.4
k_2	0.2
k_3	2.3
k_4	0.56
k_5	5.6
k_6	0.12
X_o	10
X_1	0

Table 5.2 Values for example (5.10).

This is a problem with more than one variable (x_1 and x_2) which means we must use the Newton-Raphson matrix form (5.8) to estimate the steady state. To use this we require two vectors, \mathbf{x}_k and $\mathbf{f}(\mathbf{x}_k)$ and one matrix, $\partial \mathbf{f}(\mathbf{x}) / \partial \mathbf{x}$. The \mathbf{x}_k vector is simply:

$$\mathbf{x}_k = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

The $\mathbf{f}(\mathbf{x}_k)$ vector is given by the values of the differential equations:

$$\mathbf{f}(\mathbf{x}_k) = \begin{bmatrix} (k_1 X_o - k_2 x_1) - (k_3 x_1 - k_4 x_2) \\ (k_3 x_1 - k_4 x_2) - (k_5 x_2 - k_6 X_1) \end{bmatrix}$$

The $\partial \mathbf{f}(\mathbf{x}) / \partial \mathbf{x}$ matrix is the 2 by 2 Jacobian matrix. To compute this we need to form the derivatives which in this case is straight-forward given that the differential equations are simple. In cases involving more complex rates laws, software will usually estimate the derivatives by numerical means. In

this case however it is easy to differentiate the equations to obtain the following Jacobian matrix:

$$\frac{\partial \mathbf{f}(\mathbf{x})}{\partial \mathbf{x}} = \begin{bmatrix} \frac{d(dx_1/dt)}{dx_1} & \frac{d(dx_1/dt)}{dx_2} \\ \frac{d(dx_2/dt)}{dx_1} & \frac{d(dx_2/dt)}{dx_2} \end{bmatrix} = \begin{bmatrix} -k_2 - k_3 & -k_4 \\ k_3 & -k_4 - k_5 \end{bmatrix}$$

Notice that the elements of the Jacobian contain only rate constants. This is because the model we are using is linear. This also means we need only evaluate the Jacobian and its inverse *once*. If we used nonlinear rate laws such as the Michaelis-Menten rate law, the Jacobian matrix would also contain terms involving the species concentrations and in this case, the Jacobian would need to be reevaluated at each iteration because the value for the species concentration will change at each iteration. For the current problem the Jacobian and its inverse is given by:

$$\text{Jacobian} = \begin{bmatrix} -2.86 & 5.6 \\ -0.56 & -11.2 \end{bmatrix}$$

$$\text{Jacobian}^{-1} = \begin{bmatrix} -0.3876 & -0.1938 \\ -0.01938 & -0.09898 \end{bmatrix}$$

Table 5.3 shows the progress of the iteration as we apply equation 5.8. What is interesting is that convergence only takes one iteration. This is because the model is linear. Nonlinear models may require more iterations. We can also see that after the first iteration the rates of change have very small values, this is usually due to very small numerical errors in the computer arithmetic; anything as small as 10^{-14} can be considered zero.

Iteration	x_1	x_1	dx_1/dt	dx_2/dt
0	1	1	36.74	-10.64
1	13.18	0.6589	2.8×10^{-14}	-1.16×10^{-13}

Table 5.3 Newton-Raphson applied to a three step pathway with linear kinetics. Starting values for x_1 and x_2 are both set at one. Convergence occurs within one iteration. Note that the values for the rates of change are extremely small at the end of the first iteration, indicating we have converged.

Computing the Steady State Using Software

The previous section explained how to compute the steady state using the Newton method. In practice we would not write our own solver but instead, use existing software to accomplish the same thing. To illustrate this, the following Python script will define and compute the steady state all at once:

```
import tellurium as te

rr = te.loada ('''
$Xo -> x1; k1*Xo - k2*x1;
x1 -> x2; k3*x1 - k4*x2;
x2 -> $X3; k5*x2 - k6*X3;

// Initialize value
Xo = 10; X3 = 0;
k1 = 3.4; k2 = 2.3;
k3 = 0.56; k4 = 5.6;
k5 = 0.12; k6 = 0.02;

// Initial starting point
x1 = 1; x2 = 1;
''')

# Compute steady state
rr.steadyState()
# Print out steady state values
print rr.x1, rr.x2
```

Running the above script yields steady state concentrations of 13.1783 and 0.658915 for x_1 and x_2 respectively, which is the same if we compare these values to those in Table 5.3. Other tools will have other ways to compute the steady state, for example graphical interfaces will generally have a button marked steady state that when selected will compute the steady state for the current loaded model.

When using Matlab, the function `fsolve` can be used to solve systems of nonlinear equations and in Mathematica one would use `FindRoot`.

5.6 Effect of Different Perturbations

When we discuss model dynamics, we mean how species levels and reaction rates change in time as the model evolves. There are a number of ways to elicit a dynamic response in a model, the two we will consider here are perturbations to species and perturbations to model parameters.

If the initial concentration of x_1 is zero, we can run a simulation and allow the system to come to steady state. This is illustrated in Figure 5.5 based on the model $X_o \rightarrow x_1 \rightarrow$.

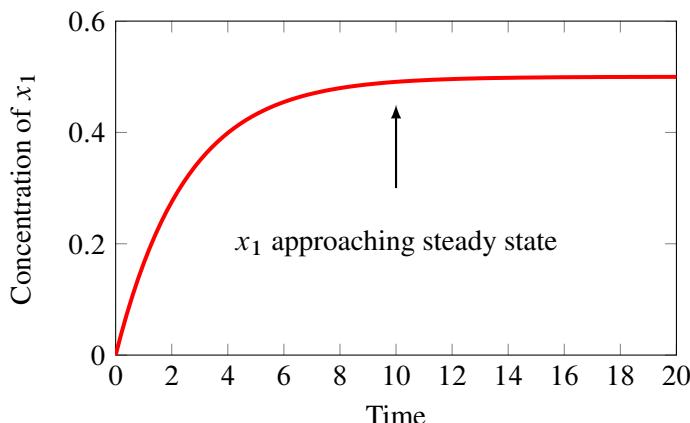


Figure 5.5 x_1 approaching steady state. Python model: 5.3

Effect of Perturbing Floating Species

Once at steady state we can consider applying perturbations to see what happens. For example, Figure 5.8 illustrates the effect of injecting 0.25 units of x_1 at $t = 20$ and watching the system evolve. The Python script used to generate this graph is in the chapter Appendix. In practice this could be accomplished by injecting 0.25 units of x_1 from an external source. We observe that the concentration of x_1 relaxes back to its steady state concentration before the perturbation was made (Figure 5.8). When we apply perturbations to species concentrations and the change relaxes back to the original state, we call the system **stable**.

Figure 5.8 illustrates one kind of perturbation that can be made to a biochemical pathway, in this case perturbing one of the floating molecular species by physically adding a specific amount of the substance to the pathway. In many cases we will find that the system will recover from such perturbations. We are not limited to single perturbations, however. Figure 5.6 shows multiple perturbations, both positive and negative. Not all systems show this behavior, and those that do not are called **unstable**. That is, when we perturb a species concentration, instead of the perturbation relaxing back, it begins to diverge.

Effect of Perturbing Model Parameters

In addition to perturbing floating species, we can also perturb the model parameters. Such parameters include kinetic constants and boundary species. Equation 5.4 shows how the concentration of species x_1 depends on all the model parameters. Moreover, changing any of the parameters results in a change to the steady state concentration of x_1 and in turn, the steady state flux. When changing a parameter we can do it in two ways; we can make a permanent change or we can make a change, then at a

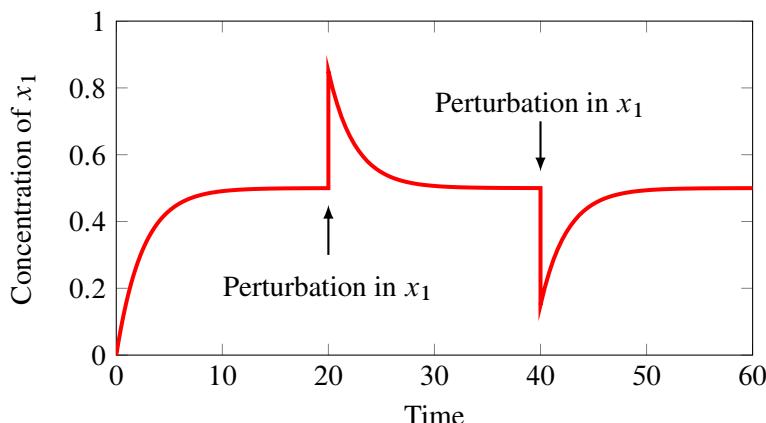


Figure 5.6 Multiple Perturbations. The steady state concentration of the species x_1 is 0.5 and a perturbation is made to x_1 by adding an additional 0.35 units of x_1 at time = 20 and removing 0.35 units at time = 40. In both cases the system relaxes back. Python script: (5.5)

later time we can return the parameter value to its original value. Assuming that the steady state is stable, a temporary change will result in the steady state changing, then recovering to the original state when the parameter is changed back. Figure 5.7 shows the effect of perturbing the rate constant, k_1 and then restoring the parameter back to its original value at some time later. Listing 5.6 shows the Python script that was used to generate this plot. In some applications other types of perturbations are made. For example, in studying the infusion of a drug where the concentration of the drug is a model parameter, one might use a slow linear increase in the drug concentration. Such a perturbation is called a ramp. More sophisticated analyses might require a sinusoidal change in a parameter, an impulse or an exponential change. These inputs are described more fully in Chapter 6. The main point to remember is that parameter changes will usually result in changes to the steady state concentrations and fluxes.

5.7 Stability and Robustness

Biological organisms are continually subjected to perturbations. These perturbations can originate from external influences such as changes in temperature, light or the availability of nutrients. Perturbations can also arise internally due to the stochastic nature of molecular events or by genetic variation. One of the most remarkable and characteristic properties of living systems is their ability to resist such perturbations and maintain very steady internal conditions. For example, the human body can maintain a constant core temperature of $36.8^\circ\text{C} \pm 0.7$ even though external temperatures may vary widely. The ability of a biological system to maintain a steady internal environment is called

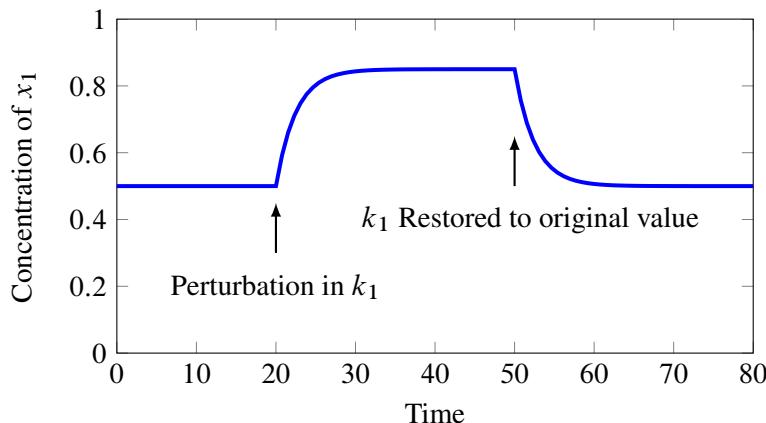


Figure 5.7 Effect of Perturbing Model Parameters using the Python script 5.6.

homeostasis, a phrase introduced by Claude Bernard almost 150 years ago. Modern authors may also refer to this behavior as **robustness**.

5.8 Introduction to Stability

A closely related concept to robustness is stability. We can define the stability of a pathway in the following way:

A biochemical pathway is dynamically stable at steady state if small perturbations in the floating species concentrations relax back to the original state.

We can illustrate a stable system using a simple two step model. Let us assume that the two step pathway has the following form:

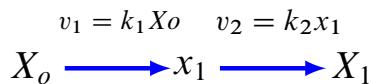


Figure 5.8 illustrates the results from a simulation of a simple two step biochemical pathway with one floating species, x_1 . The Python script used to generate this graph is given in the Python listing (5.6). The initial concentrations of the model are set so that it is at steady state, meaning no transients are seen between $t = 0$ and $t = 20$. At $t = 20$ a perturbation is made to the concentration of x_1 by adding 0.25 units of x_1 . This could be accomplished by injecting 0.25 units of x_1 into the volume where the

pathway resides. The system is now allowed to evolve further. If the system is stable, the perturbation will relax back to the original steady state, as it does in the simulation shown in Figure 5.8. This system therefore appears to be stable.

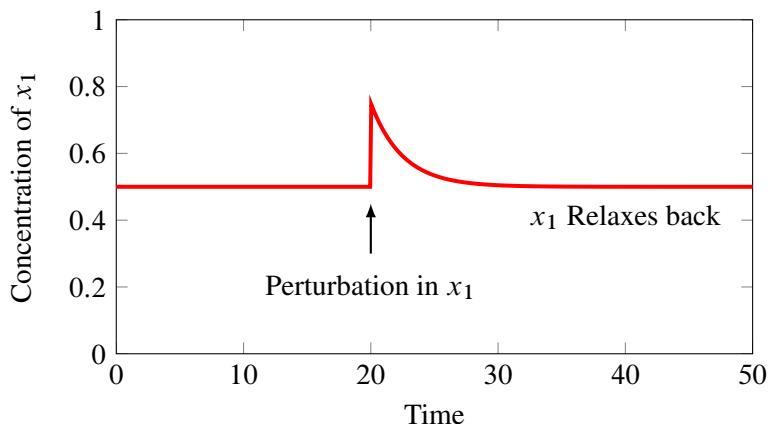


Figure 5.8 Stability of a simple biochemical pathway at steady state. The steady state concentration of the species x_1 is 0.5. A perturbation is made to x_1 by adding an additional 0.25 units of x_1 at time = 20. The system is considered stable because the perturbation relaxes back to the original steady state. See the Python listing in (5.6).

The differential equation for the single floating species, x_1 , is given by:

$$\frac{dx_1}{dt} = k_1 X_o - k_2 x_1 \quad (5.11)$$

with a steady state solution $x_1 = k_1 X_o / k_2$. We can show that the two step model is stable by using the following mathematical argument. If the system is at steady state, let us make a small perturbation to the steady state concentration of x_1 , δx_1 , and ask what is the rate of change of $x_1 + \delta x_1$ as a result of this perturbation. That is, what is $d(x_1 + \delta x_1)/dt$? The new rate of change equation is rewritten as:

$$\frac{d(x_1 + \delta x_1)}{dt} = k_1 X_o - k_2(x_1 + \delta x_1)$$

If we insert the solution for x_1 (Equation 5.4) into the above equation we are left with:

$$\frac{d\delta x_1}{dt} = -k_2 \delta x_1 \quad (5.12)$$

In other words, the rate of change of the disturbance δx_1 is negative, that is, the system attempts to reduce the disturbance so that the system returns back to the original steady state. Systems with this

kind of behavior are called **stable**. If the rate of change in x_1 had been positive instead of negative however, the perturbation would have continued to diverge away from the original steady state and the system would then be considered **unstable**.

Dividing both sides of equation 5.12 by δx_1 and taking the limit $\delta x \rightarrow 0$, we find that $\partial(dx_1/dt)/\partial x_1$ is equal to $-k_2$. To determine the stability of this system, we need to look at the sign of k_2 . Given that it is negative, δx_1 must get smaller as the system continues to evolve. We can also see that recovery from a perturbation in x_1 is only dependent on the value of the rate constant, k_2 . Since k_2 itself is always positive, the system will recover no matter what. In addition, the rate at which it recovers depends on the size of k_2 . For larger systems the stability can be determined by looking at all the terms $\partial(dx_i/dt)/\partial x_i$ which are given collectively by the expression:

$$\frac{d(\delta \mathbf{x})}{d\mathbf{x}} = \mathbf{J} \quad (5.13)$$

where \mathbf{J} is called the Jacobian matrix containing elements of the form $\partial(dS_i/dt)/\partial S_i$. Equation 5.12 can also be written as:

$$\frac{d(\delta \mathbf{x})}{dt} = \mathbf{J} \delta \mathbf{x} \quad (5.14)$$

This is a set of linear differential equations that describes the rate of change in the perturbation δS . \mathbf{J} is given by:

$$\mathbf{J} = \begin{bmatrix} \frac{\partial x_1/dt}{\partial x_1} & \dots & \frac{\partial x_1/dt}{\partial x_m} \\ \vdots & \ddots & \vdots \\ \frac{\partial x_m/dt}{\partial x_1} & \dots & \frac{\partial x_m/dt}{\partial x_m} \end{bmatrix} \quad (5.15)$$

Equation 5.14 is an example of an homogeneous linear differential equation and has the general form:

$$\frac{d\mathbf{x}}{dt} = \mathbf{A}\mathbf{x}$$

As we will see in a later chapter, solutions to such equations are well known and take the form:

$$x_j(t) = \sum_{k=1}^n \beta_{jk} e^{\lambda_k t}$$

This solution involves the sum of exponentials, $e^{\lambda_k t}$. The exponents, λ_k , of the exponentials are given by the eigenvalues of the matrix, \mathbf{A} , namely the Jacobian matrix (5.15). If the eigenvalues are

negative then the exponents decay (stable), whereas if they are positive the exponents grow (unstable). We can therefore determine the stability properties of a given model by computing the eigenvalues of the Jacobian matrix and looking for any positive eigenvalues. Note that the elements of the Jacobian matrix will often be a function of the species levels. It is important therefore that the Jacobian is evaluated at the steady state of interest. We will cover this topic in detail later.

There are many software packages that will compute the eigenvalues of a matrix, and there are a small number of packages that can compute the Jacobian directly from the biochemical model. For example, the script below is taken from Python. It defines a simple model, initializes the model values, computes the steady state, and then prints out the eigenvalues of the Jacobian matrix. For a simple one variable model, the Jacobian matrix only has a single entry and the eigenvalue corresponds to that entry. The output from running the script shows that the eigenvalue is -0.3 . Since we have a negative eigenvalue, the pathway must be stable to perturbations in x_1 .

```
import tellurium as te
rr = te.loada ('''
$X0 -> x1; k1*X0;
x1 -> $X2; k2*x1;
// Set up the model initial conditions
X0 = 1; X2 = 0;
k1 = 0.2; k2 = 0.3;
''')

# Evaluation of the steady state
rr.getSteadyStateValues()

# print the eigenvalues of the Jacobian matrix
print rr.getEigenvalues()

# Output follows:
[[-0.3  0. ]]
```

5.9 Sensitivity Analysis

Sensitivity analysis at steady state looks at how particular model variables are influenced by model parameters. There are at least two main reasons why it is interesting to examine sensitivities. The first is a practical one. Many kinetic parameters used in building biochemical models can have a significant degree of uncertainty about them. By determining how much each parameter has an influence on the model's state, we can decide whether we should improve our confidence in a particular parameter. A parameter that has considerable influence but at the same time has significant uncertainty is a parameter that should be examined more carefully by additional experimentation. On the other hand,

a parameter that has little influence but has significant uncertainty associated with it, is relatively unimportant. A sensitivity analysis can therefore be used to highlight parameters that need better precision.

The second reason for measuring sensitivities is to provide insight. The degree to which a parameter can influence a variable tells us something about how the network is responding to perturbations and to what degree. Such a study can be used to answer questions about robustness and adaptation.

How are sensitivities represented? Traditionally there are two way, one based on absolute sensitivities and the second based on relative sensitivities. Absolute sensitivities are the ratio of the absolute change in the variable to the absolute change in the parameter. That is:

$$S = \frac{\Delta V}{\Delta p}$$

where V is the variable and p the parameter. This equation shows finite changes to the parameter and variable. Unfortunately because most systems are nonlinear, the value for the sensitivity will be a function of the size of the finite change. To make the sensitivity independent of the size of the change, the sensitivity is usually defined in terms of infinitesimal changes:

$$S = \frac{dV}{dp}$$

Although absolute sensitivities are simple, they have one drawback which is that the value can be influenced by the units used to measure the variable and parameter. Often in making experimental measurements we won't be able to measure the quantity using the most natural units. Instead, we may have measurements in terms of fluorescence, colony counts, staining on a gel and so on. It is most likely that the variable and parameter units will be quite different, and each laboratory may use different units. Absolute sensitivities are therefore quite difficult to compare.

To get around the problem of units, many people will use relative sensitivities. These are simple scaled absolute sensitivities:

$$S = \frac{dV}{dp} \frac{p}{V}$$

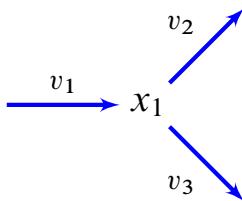
The sensitivity is defined in terms of infinitesimal changes for the same reason cited before. The reader may also recall that elasticities are measured in this way. Relative sensitivities are immune from the units we use to measure quantities. Relative sensitivities correspond more closely to how many measurements are made, often in terms of relative or fold changes. In practice, steady state relative sensitivities should be measured at the operating steady state, making a perturbation (preferably a small one), waiting for the system to reach a new steady state, and then measuring the system again. It is important to note that steady state sensitivities measure how a perturbation in a parameter moves the system from one steady state to another.

Further Reading

1. Sauro HM (2011) Enzyme Kinetics for Systems Biology. ISBN: 978-0982477311
2. Kipp E, Herwig R, Kowald A, Wierling C and Lehrach H (2005) Systems Biology in Practice, Wiley-VCH Verlag
3. Tellurium/Python web site <http://tellurium.analogmachine.org/>

Exercises

1. Consider the following simple branched network:



where $v_1 = v_o$, $v_2 = k_1 x_1$ and $v_3 = k_2 x_1$.

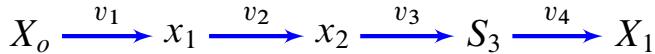
- (a) Write the differential equation for x_1 .
 - (b) Derive the equation that describes the steady state concentration for x_1 .
 - (c) Derive the equations for the steady state fluxes through v_1 and v_2 .
 - (d) Determine algebraically the *scaled sensitivity* of the steady state concentration of x_1 with respect to v_o and k_1 .
 - (e) Explain why the signs of the sensitivity with respect to v_o and k_1 are positive and negative, respectively.
 - (f) Assuming values for $v_o = 1$; $k_1 = 0.5$ and $k_2 = 2.5$, compute the values for the sensitivities with respect to k_1 and k_2 .
 - (g) What happens to the sensitivity with respect to k_1 as k_1 increases?
2. Derive equation (5.7).
 3. Using a suitable programming language, implement the Newton-Raphson algorithm and use it to find one solution to the quadratic equation: $4x^2 + 6x - 8 = 0$.

4. By changing the initial starting point of the Newton-Raphson algorithm, find the second solution to the quadratic equation from the previous question.
5. Using Python/Telluirum or some other suitable tool, find the steady state for the following model:

$$X_0 \rightarrow x_1; k_1*X_0; x_1 \rightarrow X_2; k_2*x_1; x_1 \rightarrow X_3; k_3*x_1;$$

Assume that X_0 , X_2 and X_3 have fixed concentrations. Assign suitable values to the rate constants and compute the steady state concentration of x_1 .
6. Using Python/Telluirum or some other suitable tool, perturb the value of X_0 in the above model. Apply the perturbation as a square pulse; that is, the concentration of X_0 rises, stays constant, then falls back to its original value. Make sure the system is at steady state before you apply the perturbation.
7. Explain what is meant by a stable and unstable steady state.
8. The steady state of a given pathway is stable. Explain the effect in general terms on the steady state if:
 - a) A bolus of floating species is injected into the pathway.
 - b) A permanent change is applied to a kinetic constant.
9. Why are scaled sensitivities sometimes more advantageous than unscaled sensitivities?
10. Construct a simple linear pathway with four enzymes as shown below:

20 points



Assume that the edge metabolites, X_o and X_1 , are fixed. Assign reversible Michaelis-Menten kinetics to each step and arbitrary values to the kinetics constants. Assign a modest value to the boundary metabolite, X_o , of 10 mM. Compute the steady state for your pathway. If the software fails to find a steady state, adjust the parameters. Once you have the steady state, use the model to compute the sensitivity of the steady state flux with respect to each of the enzyme maximal activities. You can compute each sensitivity by perturbing each maximal activity and observing what this does to the steady state flux.

How might you use the flux sensitivities in a practical application? Compute the sum of the four sensitivities. What value do you get? Can you make a statement about the sum?

11. Figure 5.9 shows a simple resistor/capacitor circuit connected to a fixed voltage source, V . Write out the equation that describes the rate of change of the output voltage: V_R . Using this equation, determine the steady state voltage, V_R .

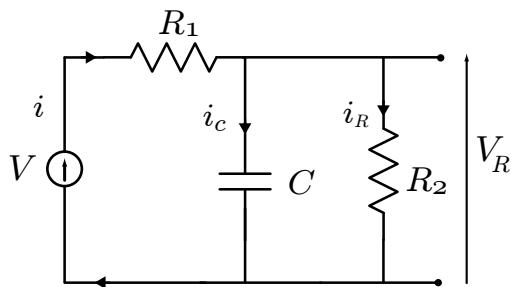


Figure 5.9 Circuit with capacitor and resistor connected to a fixed supply voltage, V .

Appendix

See <http://tellurium.analogmachine.org> for more details of Tellurium.

```
import tellurium as te

# Simulation of a simple closed system
rr = te.loada ('''
    A -> B; k1 * A;
    B -> A; k2 * B;

    A = 10; B = 0;
    k1 = 1; k2 = 0.5;
''')

result = rr.simulate(0, 3, 100)
te.plotWithLegend(rr, result)
```

Listing 5.1 Script for Figure 5.1.

```
import tellurium as te

# Simulation of an open system
rr = te.loada ('''
    $Xo -> x1; vo;
    x1 -> x2; k1*x1 - k2*x2;
    x2 -> $X3; k3*x2;

    vo = 1
    k1 = 2; k2 = 0; k3 = 3;
''')
```

```
''')

result = rr.simulate(0, 6, 100)
te.plotWithLegend(rr, result)
```

Listing 5.2 Script for Figure 5.2.

```
import tellurium as te

# Simple steady state system
rr = te.loada ('''
    $Xo -> x1; k1*Xo;
    x1 -> $X2; k2*x1;

    k1 = 0.2; k2 = 0.4;
    Xo = 1;    x1 = 0.0;
''')

result = rr.simulate(0, 20, 100, ["time", "x1"])
pylab.ylim([0, 0.6])
te.plotWithLegend(rr, result)
```

Listing 5.3 Script for Figure 5.5.

```
import tellurium as te
import numpy

# Perturbing a species concentration
rr = te.loada ('''
    $Xo -> x1; k1*Xo;
    x1 -> $X1; k2*x1;

    Xo = 1;
    x1 = 0.5;
    k1 = 0.2;
    k2 = 0.4;
''')

# Simulate the first part up to 20 time units
m1 = rr.simulate(0, 20, 100, ["time", "x1"])

# Perturb the concentration of x1 by 0.35 units
rr.x1 = rr.x1 + 0.35
```

```
# Continue simulating from last end point
m2 = rr.simulate(20, 50, 100, ["time", "x1"])

# Merge and plot the two halves of the simulation
result = numpy.vstack((m1, m2))
te.plotWithLegend(rr, result)
```

Listing 5.4 Script for Figure 5.8.

```
import tellurium as te
import numpy

# Multiple species perturbations
rr = te.loada ('''
$Xo -> x1; k1*Xo;
x1 -> $X1; k2*x1;

Xo = 1;
x1 = 0.0;
k1 = 0.2;
k2 = 0.4;
''')

# Simulate the first part up to 20 time units
m1 = rr.simulate(0, 20, 100, ["time", "x1"])

# Perturb the concentration of x1 by 0.35 units
rr.x1 = rr.x1 + 0.35

# Continue simulating from last end point
m2 = rr.simulate(20, 40, 50, ["time", "x1"])

# Merge the data sets
m3 = numpy.vstack((m1, m2))

# Do a negative perturbation in x1
rr.x1 = rr.x1 - 0.35

# Continue simulating from last end point
m4 = rr.simulate(40, 60, 50, ["time", "x1"])

# Merge and plot the final two halves of the simulation
result = numpy.vstack((m3, m4))
```

```
te.plotWithLegend(rr, result)
```

Listing 5.5 Script for Figure 5.6.

```
import tellurium as te
import numpy
import pylab

rr = te.loada ('''
    $Xo -> x1; k1*Xo;
    x1 -> $X2; k2*x1;

    Xo = 1;
    x1 = 0.5;
    k1 = 0.2;
    k2 = 0.4;
''')

# Simulate the first part up to 20 time units
m1 = rr.simulate(0, 20, 5, ["time", "x1"]).copy()

# Perturb the parameter k1
rr.k1 = rr.k1 * 1.7

# Simulate from the last point
m2 = rr.simulate(20, 50, 40, ["time", "x1"]).copy()

# Restore the parameter back to ordinal value
rr.k1 = 0.2

# Carry out final run of the simulation
m3 = rr.simulate(50, 80, 40, ["time", "x1"])

# Merge all data sets and plot
result = numpy.vstack((m1, m2, m3))
pylab.ylim([0,1])
te.plotWithLegend(rr, result)
```

Listing 5.6 Script for Figure 5.7.

```
import tellurium as te
import numpy
```

```
# Stability illustration
rr = te.loada ('''
    $Xo -> x1;  k1*Xo;
    x1 -> $X1;  k2*x1;

    Xo = 1;
    x1 = 0.5;
    k1 = 0.2;
    k2 = 0.4;
''')

# Simulate the first part up to 20 time units
m1 = rr.simulate(0, 20, 100, ["time", "x1"]).copy()

# Perturb the concentration of x1 by 0.35 units
rr.x1 = rr.x1 + 0.35

# Continue simulating from last end point
m2 = rr.simulate(20, 50, 100, ["time", "x1"]);

# Merge and plot the two halves of the simulation
result = numpy.vstack ((m1, m2))
te.plotWithLegend(rr, result)
```

Listing 5.7 Tellurium script used to generate Figure 5.8.

6

Inputs to a System

“It’s designed to disrupt your input/output carrier signal.”

– The Matrix, 1999

In this chapter we are going to look at the different kinds of inputs that can be applied to a system. In control engineering such inputs are also called signals, forcing functions or driving functions. Inputs to a system can be applied in a variety of different ways, some of which are critical to a proper understanding of how the system operates. In particular we are going to look at the **step function**, the **impulse**, the **ramp** and the **sinusoidal** input. Along the way we will also introduce **shifting** or time delays in signals. In the following text, t refers to time.

Figure 6.1 illustrates the main types of inputs that are commonly found in engineering with perhaps the step function being the most common. Increasing the expression of a gene by changing the inducer concentration is an example of a step function.

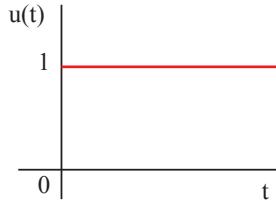
Example 6.1

What type of signal is represented in the following situations:

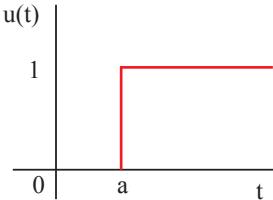
- a) The concentration of glucose entering a chemostat is increased from 5 mM to 8 mM and remains at 8 mM.
Step input
- b) A drug is very rapidly injected into a patient’s arm.
Approximates an impulse
- c) The concentration of an inducer molecule is slowly increased over time until it reaches 0.2 mM. The con-

centration of inducer then remains at this concentration to the end of the experiment.

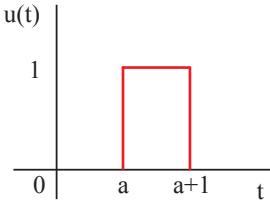
Ramp followed by constant level.



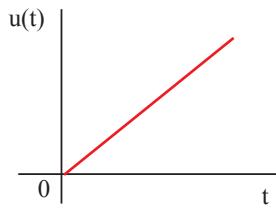
Unit Step Function



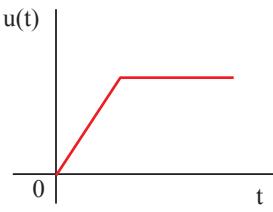
Delayed Unit Step Function



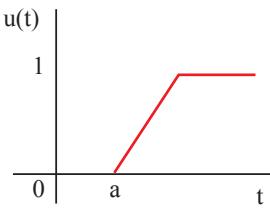
Delayed Unit Pulse Function



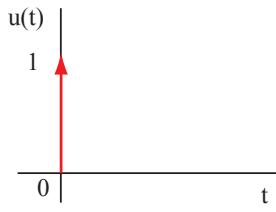
Ramp Function



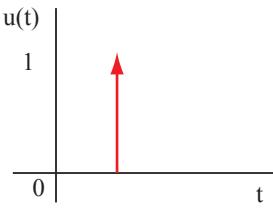
Ramp with Stop



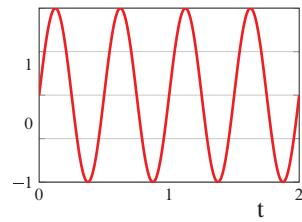
Delayed Ramp



Unit Impulse



Delayed Unit Impulse



Sinusoidal Input

Figure 6.1 Variety of signal inputs.

6.1 Unit Step Function

The unit step function or **Heaviside function**, $u(t)$, is described as a function in time that has a value of zero before $t = 0$, but for $t \geq 0$, it has a constant value of **one** (Figure 6.2). We can summarize the properties of the unit step function in the following way:

Unit Step Function

$$u(t) = \begin{cases} 0 & t < 0 \\ 1 & t \geq 0 \end{cases}$$

Scaling

For steps other than unity, such as a step size of α , we can write the step function as:

$$u_\alpha(t) = \alpha u(t)$$

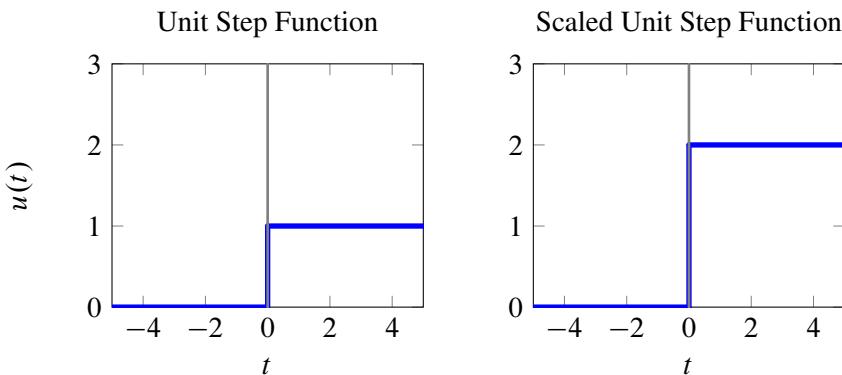


Figure 6.2 Unit Step Function, $u(t)$ left panel. Scaled Unit Step Function, $2u(t)$ right panel.

Examples of step functions in biology are common. Increasing the expression of a particular protein or increasing the concentration of a drug in a cell culture can all be modeled as a step function.

The unit step function has other uses. The expression $f(t) u(t)$ has the effect of ‘switching’ on the function, $f(t)$ at time zero. Before time zero, the product, $f(t) u(t)$, is zero. For example, $\sin(t)u(t)$ is zero before $t = 0$ but exhibits sinusoidal behavior for $t \geq 0$. See Figure 6.3.

6.2 Delayed Signals or Shifting

It is often desirable to delay or **shift** a signal. For example rather than have a step signal occur at $t = 0$, we might want the step to occur at some later time. For example, Figure 6.4 shows a step signal delayed until $t = 2$. Delayed inputs are so common that a particular notation is used to describe them:

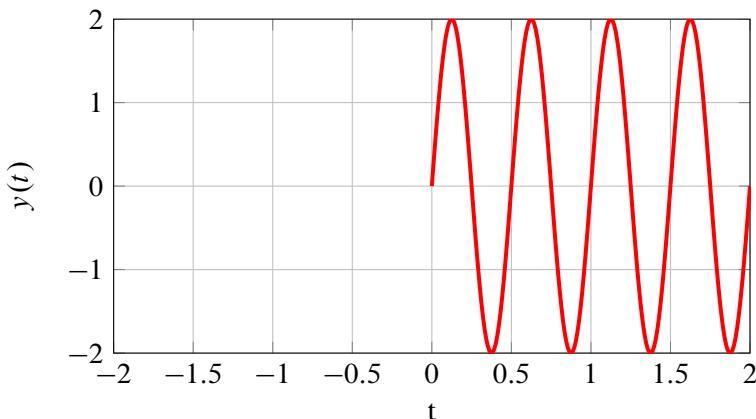


Figure 6.3 Sine wave starting at $t = 0$ using the unit step function: $\sin(t) u(t)$.

$$u(t - a)$$

where a is the amount of delay. Note that t is the variable and a is a constant. By analogy, the expression:

$$u(t + a)$$

shifts the signal to an earlier time.

$$u(t - a) = \begin{cases} 0 & t < a \\ 1 & t \geq a \end{cases}$$

To explain how the delay notation works, consider the following. When $t = a$, we know that $t - a = 0$, that is $u(t - a) = u(0)$. Therefore, when $t = a$, $u(t - a)$ is the value of u at time zero. Similarly, if $t < a$, then the value of u corresponds to a time before $t = 0$, which by definition is also zero. If we move t beyond a by an amount α , that is, $t = a + \alpha$, then $t - a = \alpha$. Thus $u(t - a) = u(\alpha)$. The entire function therefore moves forward by a time units.

The expression $u(t - 2)$ means that the step function has moved to $t = 2$.

We can generalize shifting to delay any function. To shift a function $f(t)$, we shift the function itself, then ensure that the function is zeroed out before the shift time point. To shift a function, $f(t)$ by a , we apply the expression:

$$f(t - a)u(t - a)$$

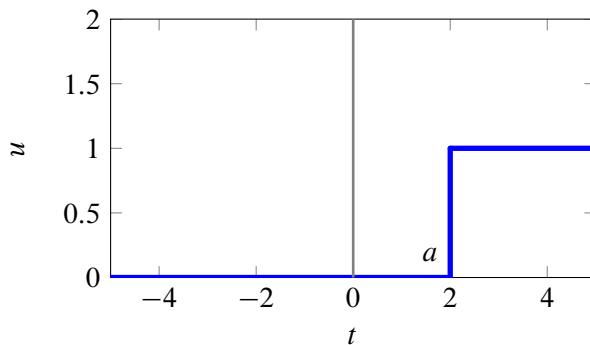


Figure 6.4 Delayed Unit Step Function, $u(t - 2)$.

where $f(t - a)$ shifts the function and $u(t - a)$ zeros out the function until a . For example (See also Figure 6.22), to shift the function x^2 two time units into the future, we would write:

$$(x - 2)^2 u(t - 2)$$

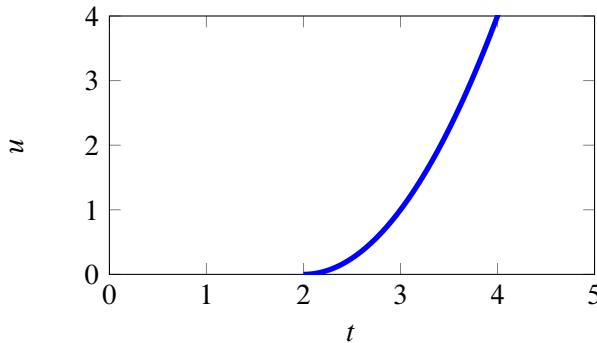


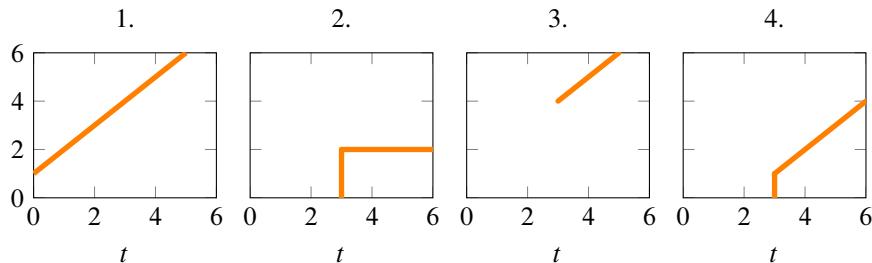
Figure 6.5 Shifted (delayed) function, x^2 using $(x - 2)^2 u(t - 2)$.

Example 6.2

Sketch the graphs for the following functions, t is the independent variable on the x-axis:

1. $y = t + 1$
2. $y = 2u(t - 3)$
3. $y = (t + 1)u(t - 3)$

4. $y = ((t - 3) + 1)u(t - 3)$



6.3 Addition of Signals

A very useful operation is adding signals. For example, consider the signal:

$$y = u(t - 1) + u(t - 2)$$

where we have summed two delayed step functions. To illustrate how addition works, let's plot each signal separately (Figure 6.6, left panel). The right panel in the same figure is the sum of the two individual terms. Signals may also be subtracted as we will see in the next section.

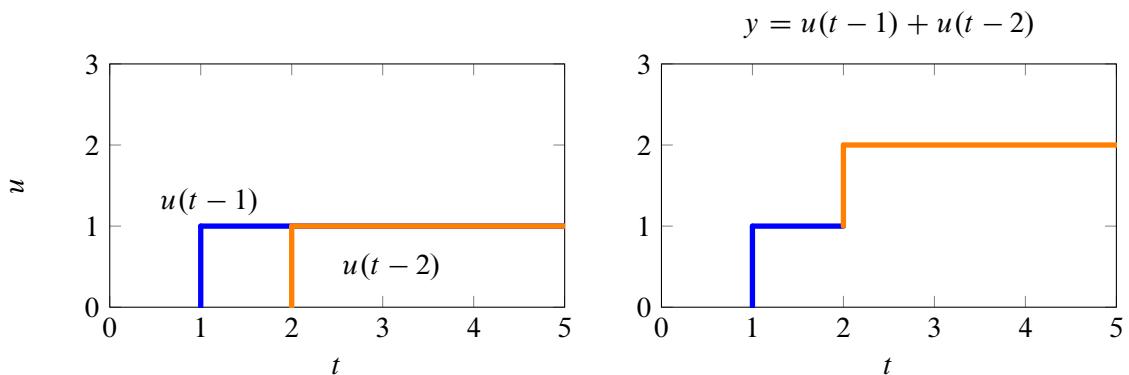


Figure 6.6 Individual terms left panel. Sum of terms right panel.

Pulses

If a drug is administered via an intravenous line for a set period, we can model the input of the drug into the blood stream as a pulse with a given width. We can define pulse signals by combining step

functions and delays. For example, consider a pulse that starts at $t = a$ and stops at a later time, $t = b$. We can represent this as follows:

$$\text{pulse}(a, b) = u(t - a) - u(t - b) \quad \text{where } a < b$$

That is, at time a the unit step function rises to 1.0. At this point, $u(t - b)$ is still zero. At $t = b$, $u(t - b)$ goes to one, but it subtracts this value from $u(t - a)$, thereby reducing the signal back to zero. Figure 6.7 illustrates such a response.

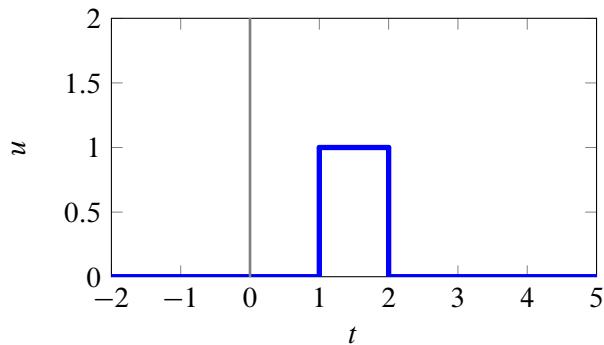


Figure 6.7 Shifted Unit Pulse: $u(t - 1) - u(t - 2)$.

A pulse can be used to extract part of a function. For example:

$$\sin(t)(u(t - 1) - u(t - 2))$$

will extract part of the sine function.

Pulse Train

Once we define a single pulse, it is possible to describe a train of pulses. Figure 6.8 shows two pulses which are constructed by summing two separate pulses. We can generalize this to n pulses using the equation:

$$f(t) = \sum_{k=1}^n [(-1)^{k+1} u(t - dk)]$$

where d equals the width of a single pulse.

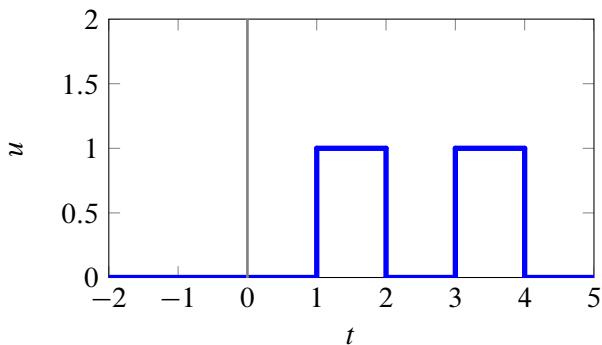


Figure 6.8 Shifted Unit Pulse Signal, $[u(t - 1) - u(t - 2)] + [u(t - 3) - u(t - 4)]$.

6.4 Ramp Signal

If a drug is administered to a patient intravenously over time but at an increasing level, we have a ramp signal. The ramp is a function of time that rises or falls at a constant rate. If we have a ramp that rises at a rate k , then the ramp signal can be described by kt . It is more convenient however to define the ramp in terms of the unit step response:

$$\text{ramp}(t) = kt u(t)$$

Since the unit step function is zero for $t < 0$, the ramp is also zero. Only when $t \geq 0$ does the ramp begin to rise at a rate k (Figure 6.9). Like the step function, the ramp can also be delayed. We might consider delaying the ramp using the expression:

$$kt u(t - a)$$

This means that the ramp starts to rise at $t = a$ where the slope of the ramp is given by k . Note however that the ramp will appear to start at a y value of kt (Figure 6.9, right panel), that is the first part of the ramp has been removed. As described before we are using the step function to **mask** part of a function.

More often we will want the ramp to start from zero at a given time in the future, that is the signal is shifted to a future point in time.

A *shifted* ramp, where the y value starts at $y = 0$ (not half way up as it did in the last example), can be created with the following expression (Figure 6.10: left panel):

$$k(tu(t - 2) - 2u(t - 2))$$

The first term describes a ramp starting at $t = 2$, with slope k and a y value of $2k$. In order for

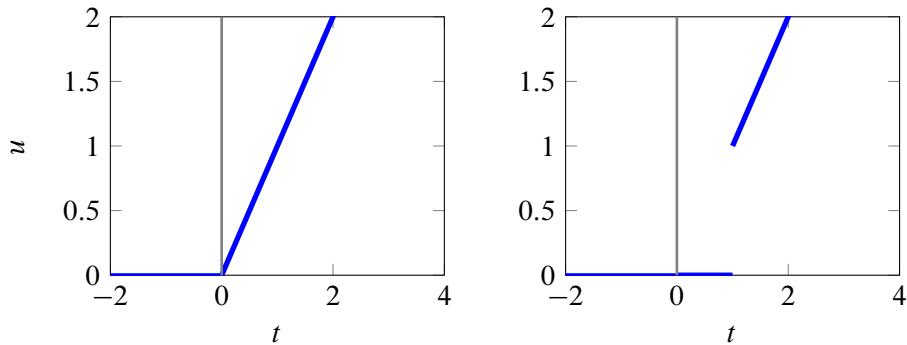


Figure 6.9 Left panel: Unit Ramp Signal: $tu(t)$. Right panel: shift unit ramp signal: $tu(t - 1)$

the ramp to start at $y = 0$, we must translate the ramp to the x -axis. To do this we just subtract 2 k starting at $t = 2$.

To create a ramp that rises and stops (Figure 6.10: right panel), we can use the expression:

$$k(tu(t) - (t - a)u(t - a))$$

The first term describes a ramp starting at time, $t = 0$ with slope k . At time a , the same slope is now subtracted from the original rising slope which gives us a horizontal line. The term $u(t - a)$ ensures that the negative slope, $-(t - a)$, is only subtracted at time a . Think of the term $-(t - a)$, i.e. $(-t + a)$, as a line with negative unit slope and a y intercept of a . Compare this to the generalized shifting function, $f(t - a)u(t - a)$, we encountered earlier. In this case the function is simply kt , so that the shifted function is $k(t - a)u(t - a)$.

By combining a ramp and unit steps we can create a sawtooth signal. The first part of a sawtooth signal will be a ramp:

$$r(t) = tu(t)$$

Let's assume that the first tooth lasts until $t = 1$ at which point the value falls to zero. To accomplish this we can subtract from ramp $tu(t - 1)$:

$$t(u(t) - u(t - 1))$$

We now repeat this for as many sawtooths as we need. For example, the second sawtooth is given by:

$$(t - 1)(u(t - 1) - u(t - 2))$$

and a third:

$$(t - 2)(u(t - 2) - u(t - 3)) \quad (6.1)$$

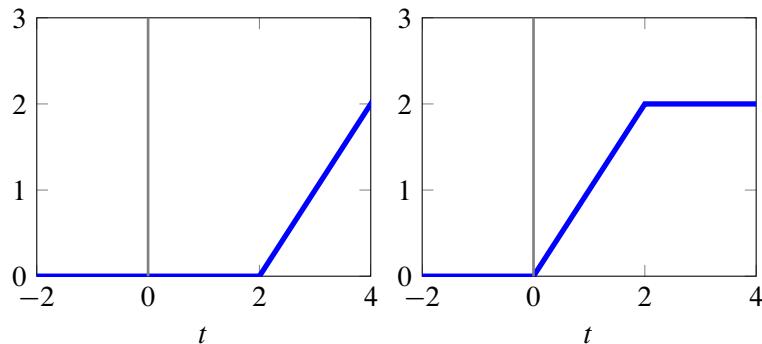


Figure 6.10 Left Panel: Ramp Function with a slope of 1 that starts at time 2: $tu(t - 2) - 2u(t - 2)$. Right Panel: Ramp Function with a slope of 1 that stops at time 2: $tu(t) - (t - 2)u(t - 2)$.

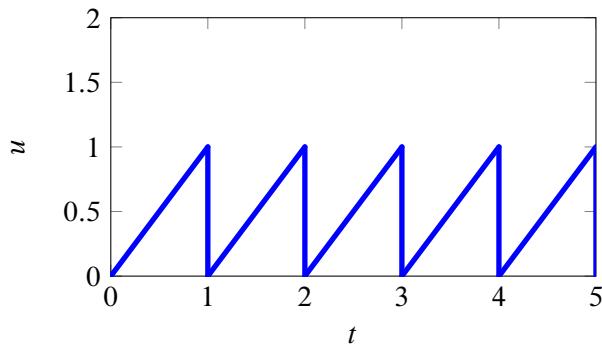


Figure 6.11 Sawtooth signal, see equation 6.1.

6.5 Impulse Signal

The impulse¹, $\delta(t)$, is probably the most important signal but at the same time one of the stranger functions to consider. The perfect impulse is a signal that lasts an infinitesimally small amount of time but whose magnitude appears to be infinite. Such a thing doesn't exist in the physical world and yet it is extremely important in control theory. In practice the impulse can only be realized approximately, though a hammer suddenly hitting an anvil is a good approximation. Mathematically, we can say that the impulse has zero value everywhere except at $t = 0$ where its height is very large. As such, the impulse function is not a normal function because it doesn't have a definite value at $t = 0$. For this reason an impulse is defined in a special way by describing its properties, which include:

¹Also known as the Dirac delta function in the physics community.

$$\delta(t) = 0 \quad \text{for } t \neq 0$$

$$\int_{-\infty}^{\infty} \delta(t) dt = 1$$

The first expression states that the value of the impulse at all values other than zero *is* zero. This is an odd statement because it just tells us where the impulse isn't, not what it is. The second expression gives us more information by stating that the area of an impulse is one. This arises by first considering a pulse where the height of the pulse is $1/T$ and its width T . Note that the area of the pulse = $(1/T)T = 1$. As the width of the pulse is reduced, the area of the pulse remains one. The width can be reduced to a value as small as we like (Figure 6.12), but still the area will remain one. These arguments are somewhat weak, and a more formal description of an impulse involves defining it in terms of a distribution or generalized function [32]. For practical purposes however, the definition in terms of a limiting rectangle is sufficient for all of the work we will do in control theory.

The description of an impulse can be further generalized to any time, a :

$$\delta(t - a) = 0 \quad \text{for } t \neq a$$

$$\int_{-\infty}^{\infty} \delta(t - a) dt = 1$$

The strength of an impulse is defined by its area and we can change the area of an impulse such that:

$$\int_{-\infty}^{\infty} K\delta(t) dt = K$$

An impulse with a given strength is often indicated using the **notation** $K\delta(t)$ where K is the strength. Don't consider the $K\delta(t)$ notation as a product, it is not. It is *only* a convenient notation. Such an expression *only* makes sense under integration.

As already stated the impulse can be delayed. To specify that a impulse should be delayed until time a , use the notation:

$$\delta(t - a)$$

A series of impulses can be represented as a sum of impulses at different times:

$$\delta(t - a) + \delta(t - b) + \dots$$

To generate a series of impulses at regular intervals we can write:

$$\sum_{k=0}^{k=n} \delta(t - kT)$$

where T is the period of the pulse train and n the number of pulses.

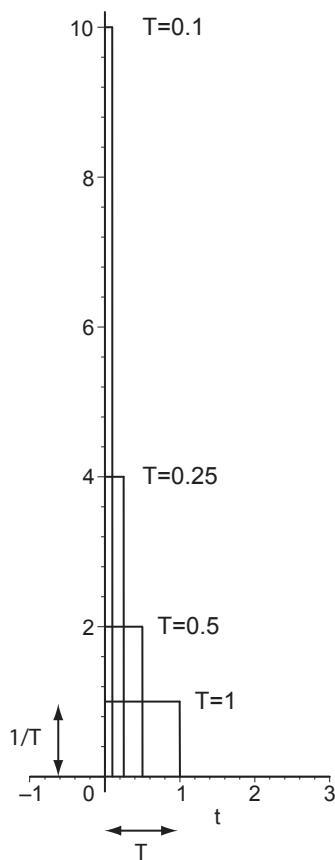


Figure 6.12 Pulses converging to an impulse while maintaining the area fixed at 1.0: $T \times 1/T$

Sifting

Probably one of the most important properties of an impulse is its ability to pick out a value from a given function, called **sifting**. Consider a continuous function, $f(t)$ that we multiply by $\delta(t)$. This product will be zero except for the point marked by the impulse.

Consider the integral:

$$\int_{-\infty}^{\infty} f(t) \delta(t - a) dt$$

From the definition of an impulse we know that the impulse is zero everywhere except when $t = a$.

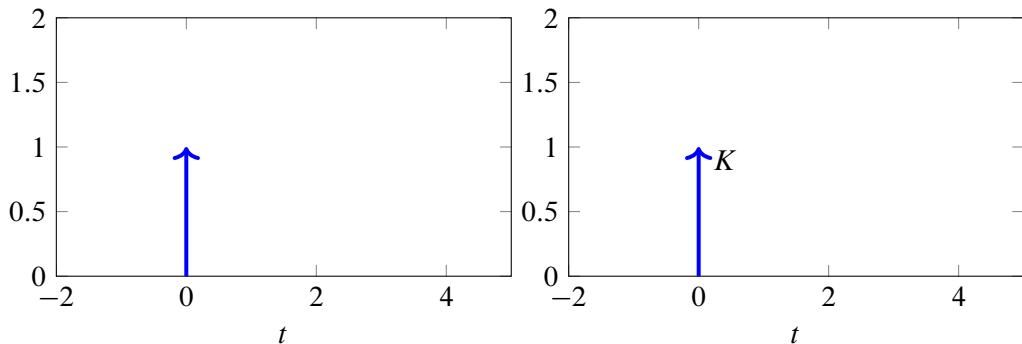


Figure 6.13 Left Panel: Depiction of an impulse Signal, $\delta(t)$. Right Panel: Depiction of impulse $K\delta(t)$, with strength, K .

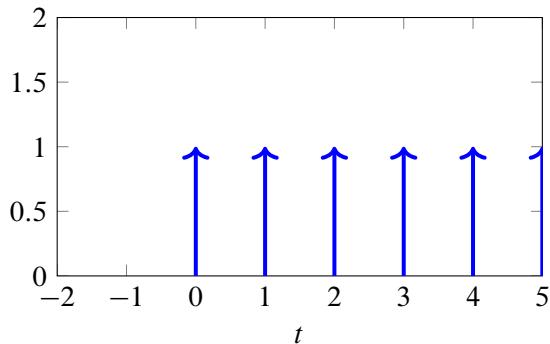


Figure 6.14 A series of impulse signals, $\delta(t - k)$.

Therefore the only value of $f(t) \delta(t - a)$ that isn't zero is $f(a) \delta(t - a)$. This allows us to write:

$$\int_{-\infty}^{\infty} f(t) \delta(t - a) dt = \int_{-\infty}^{\infty} f(a) \delta(t - a) dt$$

Since $f(a)$ is a constant we can move it out of the integral:

$$\int_{-\infty}^{\infty} f(t) \delta(t - a) dt = f(a) \int_{-\infty}^{\infty} \delta(t - a) dt$$

Since we know that:

$$\int_{-\infty}^{\infty} \delta(t - a) dt = 1$$

then:

$$\int_{-\infty}^{\infty} f(t) \delta(t - a) dt = f(a) \quad (6.2)$$

This result tells us that the area under the product of a function, $f(t)$, with a unit impulse, is equal to the value of that function at the point where the unit impulse is located. In other words the impulse function plucks out or sifts out the function value at the point of the impulse. The function, $f(t)$, must be continuous where the impulse is located.

At first sight it result may appear completely redundant since we can more easily compute $f(a)$ by inserting a into $f(a)$. However this result is used when we derive the convolution integral (11.3) which is used to understand a system's response to an input.

Relationship to the Step Function

One final property worth noting is the relationship between an impulse and the step function. Figure 6.15 illustrates the derivative of a ramp. As the ramp approaches a unit step, the pulse approaches an impulse. From this we can suggest that the relationship between a unit step and impulse is:

$$\frac{du(t)}{dt} = \delta(t)$$

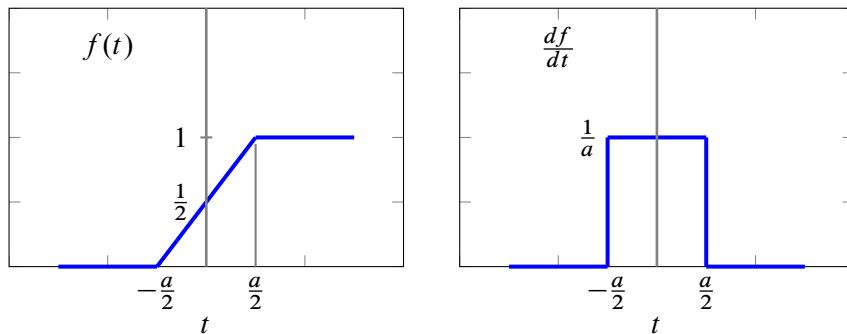


Figure 6.15 As a approaches zero, $f(t)$ approaches a unit step and df/dt approaches a unit impulse.

Main Properties of the Impulse:

$$\delta(t - a) = 0 \quad \text{for } t \neq a$$

$$\int_{-\infty}^{\infty} \delta(t - a) dt = 1$$

$$\int_{-\infty}^{\infty} f(t) \delta(t - a) dt = f(a)$$

6.6 Sinusoidal

A sinusoidal signal has the form of a sine or cosine function and is the most fundamental periodic signal. Any other periodic signal can be constructed from a summation of sinusoidal signals by using the Fourier series (See Chapter 12). We can describe a sinusoidal signal by an equation of the general form:

$$y(t) = A \sin(\omega t + \theta)$$

The equation has three terms:

1. **Amplitude, A**
2. **Angular frequency, ω** (rads sec $^{-1}$)
3. **Phase, θ** (radians)

including the derived terms:

4. **Period, T** (seconds)
5. **Frequency, f** (cycles sec $^{-1}$)

Frequency The angular frequency is the rate of the periodic signal and is expressed in radians per second. A sine wave traverses a full cycle (peak to peak) in 2π radians (circumference of a circle) so that the number of complete cycles traversed in one second is then $\omega/2\pi$. This is termed the frequency, f and has units of cycles sec $^{-1}$ also called **Hertz** (Hz). The angular frequency is conveniently expressed as $\omega = 2\pi f$ which has units of radians sec $^{-1}$. A sinusoid will therefore sometimes also be written in the form:

$$y(t) = A \sin(2\pi f t + \omega)$$

The horizontal distance peak to peak is referred to as the **period, T** . and is usually expressed in

seconds. The inverse of the period, $1/T$ sec $^{-1}$ is equal to the frequency, f in Hz.

$$T = \frac{2\pi}{\omega}; \quad f = \frac{1}{T}$$

Example 6.3

What is the frequency and period of $y = \sin(3t)$?

Since the angular frequency, $\omega = 2\pi f$, then: $\omega = 3 = 2\pi f$. That is the frequency is given by:

$$f = \frac{3}{2\pi}$$

and the period, $T = 1/f$ is given by:

$$T = \frac{2\pi}{3}$$

Example 6.4

A sine wave has a frequency of 10 cycles per second. What is its angular frequency and period of the wave?

The angular frequency is given by $\omega = 2\pi f$. Therefore $\omega = 2\pi \cdot 10 = 62.83$ radians per second.

The period of the wave is $T = 1/f$, therefore $T = 1/10 = 0.1$ seconds.

The period allows us to define a periodic signal as follows. A signal $g(t)$ is periodic if:

$$g(t + T) = g(t)$$

That is the signal repeats itself exactly every T time units.

Amplitude The amplitude is the extent to which the periodic function changes in the y direction, that is the maximum height of the curve *from the origin* (Figure 6.18).

Figure 6.17 shows a typical plot of the sinusoidal function, $y(t) = A \sin(\omega t + \theta)$, where the amplitude is set to two. The frequency, $\omega/(2\pi)$, is 2 cycles per second and the period is therefore 0.5 seconds. The angular frequency is $\omega = 4\pi$ radians per second. Note the phase, θ , is zero. The left panel in Figure 6.18 shows the effect of varying the amplitude. The right panel shows two signals of different frequency.

Phase The phase for a sinusoidal signal need not be zero. The two sine waves shown in Figure 6.19 have the same frequency and amplitude but one of them is shifted to the right by 90 degrees (or $\pi/2$ radians), that is, phase shifted.

Example 6.5

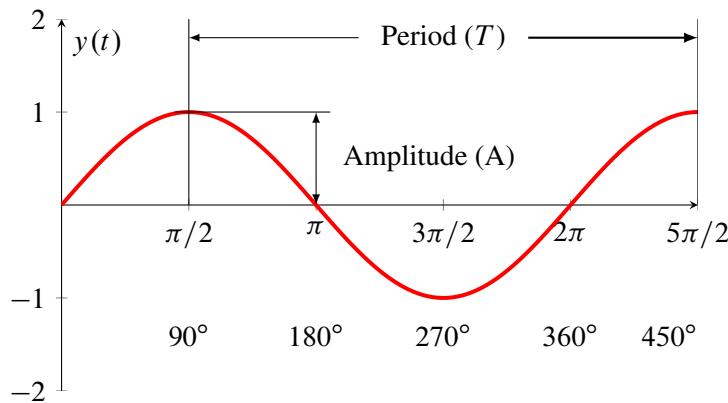


Figure 6.16 Basic Sine Wave: $y(t) = A \sin(t)$ where $A = 1$.

Write out the amplitude, phase, angular frequency, period and frequency in Hz for the sinusoid (time is expressed in seconds):

$$y(t) = 24 \cos(50t + 0.4\pi)$$

Amplitude, $A = 24$

Phase, $\theta = 0.4\pi$ radians

Angular frequency $\omega = 50$ rad/s

Period $T = 2\pi/\omega = 2\pi/50 = 0.1257$ s

Frequency (Hz) $f = 1/T = 7.958$ Hz

Harmonics

Consider a sinusoidal signal with a frequency, f Hz. Integer multiples of f are called **harmonics**. In this case $2f, 3f, 4f, \dots$ are all harmonics of the base frequency f . In the literature the base frequency is called the **fundamental frequency** and can also be considered the first harmonic waveform. The concept of harmonics will arise in later chapters when we briefly discuss the Fourier series. Figure 6.20 illustrates a set of five harmonics.

Masking and Time Shifting

The sine signal can be masked (Figure 6.21) by multiplying it by a delayed unit step response, for example:

$$u(t - a) \sin(\omega t)$$

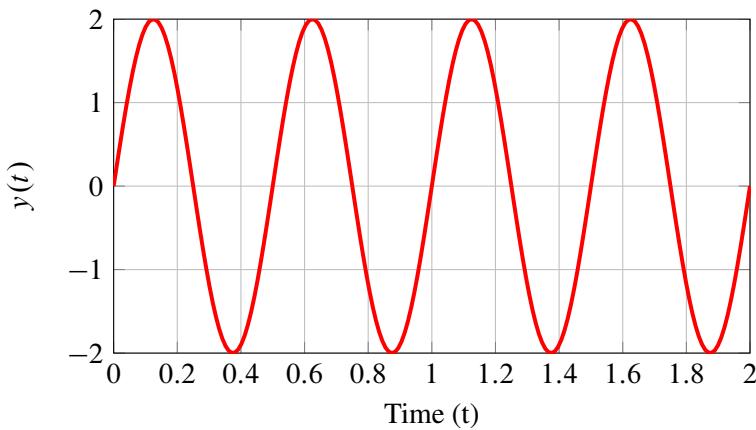


Figure 6.17 $y(t) = A \sin(\omega t + \theta)$ where $A = 1$, $f = 2$ Hz so that $\omega = 4\pi$, $\theta = 0$.

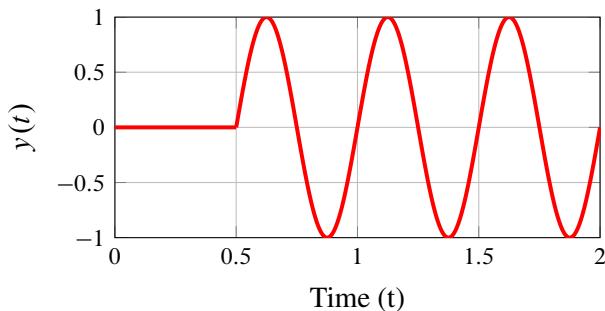


Figure 6.21 A masked Sine Function, $u(t - 0.5) \sin(4\pi t)$. Note that a frequency of 4π ensures that a full cycle starts every 0.5 s.

Shifting a sine function can be accomplished using the expression:

$$\sin(t - a)u(t - a)$$

Sum of Sinusoidal Functions

One very important property of sinusoidal signals is that the sum of two sinusoidal signals of the same frequency but different phase and amplitude will result in another sinusoidal with a different phase and amplitude **but identical frequency**. That is:

$$A_1 \sin(\omega t + \theta_1) + A_2 \sin(\omega t + \theta_2) = A_3 \sin(\omega t + \theta_3) \quad (6.3)$$

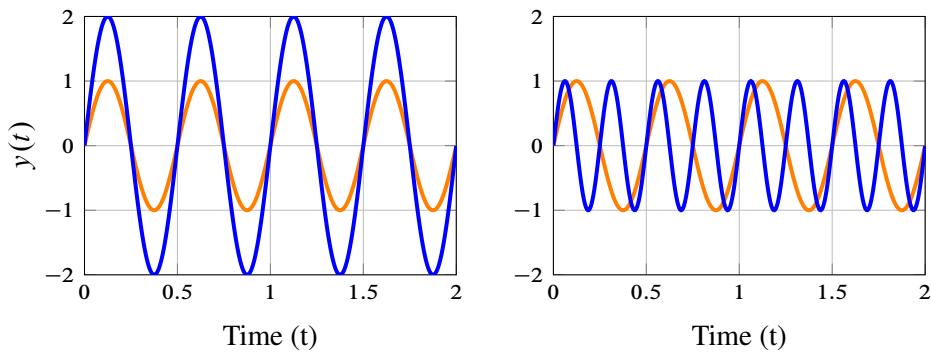


Figure 6.18 Left Panel: Amplitude change $y(t) = A \sin(\omega t + \theta)$ where $A = 2$, $f = 2$ Hz, $\theta = 0$; Right Panel: Frequency Change $y(t) = A \sin(\omega t + \theta)$ where $A = 1$, $f = 4$ Hz, $\theta = 0$

where:

$$A_3 = \sqrt{A_1^2 + A_2^2 + 2A_1 A_2 \cos(\theta_1 - \theta_2)}$$

and:

$$\theta_3 = \arctan\left(\frac{A_1 \sin \theta_1 + A_2 \sin \theta_2}{A_1 \cos \theta_1 + A_2 \cos \theta_2}\right)$$

The proof for this relationship is given in the chapter Appendix. This result provides a key insight when we consider the propagation of sinusoidal signals through linear systems.

6.7 Exponential

Although we may not think of common situations in biology where exponentials are used as inputs to a system, their theoretical importance is crucial to the study of systems. The common or real exponential is given by:

$$y = Ae^{ax}$$

where A and a are constants.

e is Euler's number and is given approximately by 2.71828182845904523536... and defined as the limit of $(1 + 1/n)^n$ as n approaches infinity. Typical plots of the common exponential are given in Figure 6.23. The constant A determines the value of y at $t = 0$. The constant a determines whether the exponential grows ($a > 0$) or decays ($a < 0$), and the degree to which it does. In the science and engineering fields the exponential function is also expressed as:

$$y = Ae^{x/\tau}$$

where τ is called the **time constant** (See 15.5).

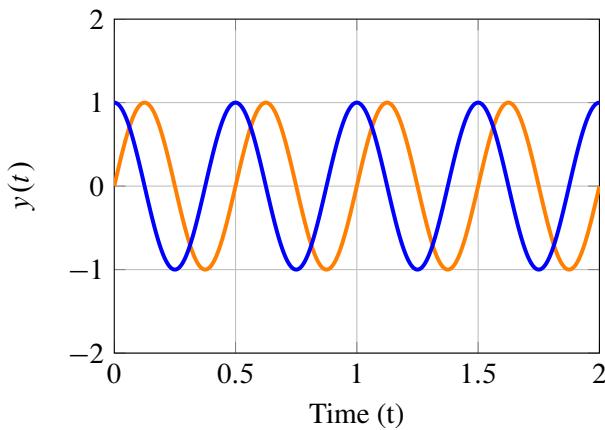


Figure 6.19 Phase change: $y(t) = A \sin(\omega t + \theta)$ where $A = 1$, $f = 2$ Hz, $\theta = 90^\circ$. The lighter curve is shifted 90° to the right relative to the darker curve.

The exponential can be defined in terms of the Taylor series:

$$e^x = 1 + \frac{x^1}{1!} + \frac{x^2}{2!} + \frac{x^3}{3!} + \dots = \sum_{n=0}^{\infty} \left(\frac{1}{n!} x^n \right) \quad (6.4)$$

such that it is possible to prove the following properties:

$$e^{x_1+x_2} = e^{x_1} e^{x_2}$$

$$e^{x_1-x_2} = e^{x_1}/e^{x_2}$$

Complex Exponential: What about the exponential of a complex number such as $i\omega$, that is $e^{i\omega}$? How can we interpret such an expression? If we substitute x in the exponential Taylor series for $i\omega$ and assume for now that the series converges, then we obtain:

$$\begin{aligned} e^{i\omega} &= 1 + i\omega + \frac{(i\omega)^2}{2!} + \frac{(i\omega)^3}{3!} + \frac{(i\omega)^4}{4!} + \frac{(i\omega)^5}{5!} + \frac{(i\omega)^6}{6!} + \frac{(i\omega)^7}{7!} + \dots \\ &= 1 + i\omega - \frac{\omega^2}{2!} - \frac{i\omega^3}{3!} + \frac{\omega^4}{4!} + \frac{i\omega^5}{5!} - \frac{\omega^6}{6!} - \frac{i\omega^7}{7!} + \dots \\ &= \left(1 - \frac{\omega^2}{2!} + \frac{\omega^4}{4!} - \frac{\omega^6}{6!} + \frac{\omega^8}{8!} - \dots \right) + i \left(\omega - \frac{\omega^3}{3!} + \frac{\omega^5}{5!} - \frac{\omega^7}{7!} + \dots \right) \\ &= \cos \omega + i \sin \omega . \end{aligned}$$

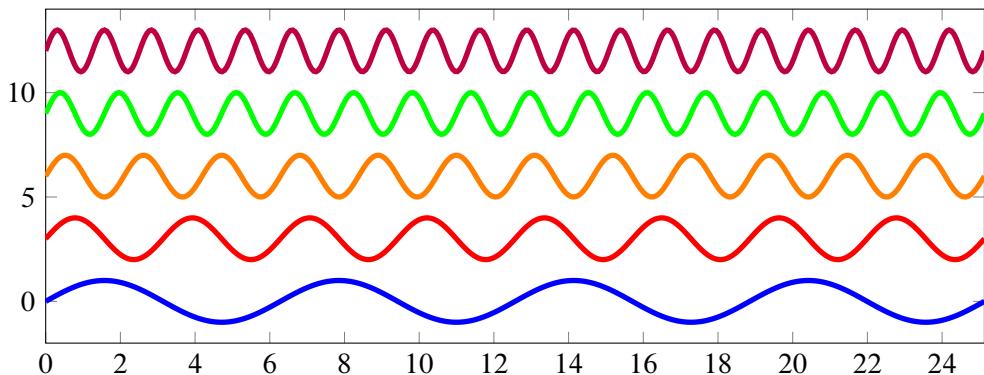


Figure 6.20 Set of five Harmonics based on the fundamental frequency f . From top down, the harmonics are $5f$, $4f$, $3f$, $2f$ and f .

Since both trigonometric series converge, the complex exponential also converges. The final result is called Euler's theorem and shows us that the complex exponential is related to the trigonometric functions and therefore has **periodic properties**.

Euler's Theorem:

$$e^{i\omega} = \cos \omega + i \sin \omega$$

We can generalize further and consider the complex exponential: $e^{(\sigma+i\omega)t}$. This is called the **complex exponential function**

$$y = Ae^{(\sigma+i\omega)t} = Ae^{\sigma t} e^{i\omega t}$$

where $s = \sigma + i\omega$

The symbol s is commonly used to indicate the complex number in an exponential. If the complex term $i\omega$ is zero, then the complex exponential reduces to the real exponential. Although we know what a plot of a real exponential looks like, how does the complex exponential behave?

We can write the complex exponential as follows (assuming $A = 1$):

$$y = e^{st} = e^{(\sigma+i\omega)t} = e^{\sigma t} e^{i\omega t}$$

Using Euler's theorem $e^{i\omega t} = \cos \omega t + i \sin \omega t$:

$$y = e^{\sigma t} (\cos \omega t + i \sin \omega t)$$

If for the moment we set $\sigma = 0$, then we can look at the complex number:

$$y = \cos \omega t + i \sin \omega t$$

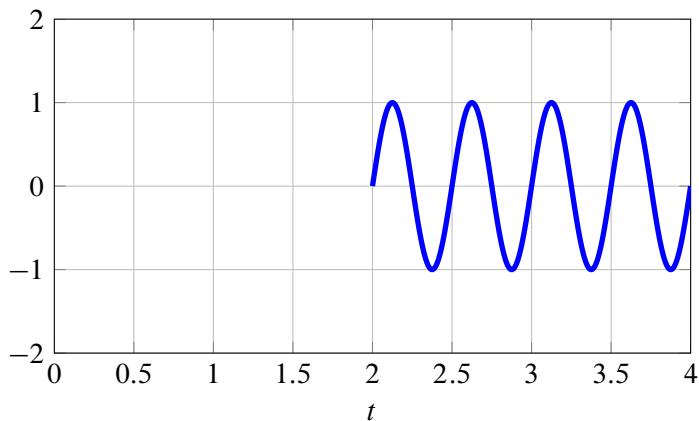


Figure 6.22 Shifted (delayed) Sin wave $(x - 2)^2 u(t - 2)$

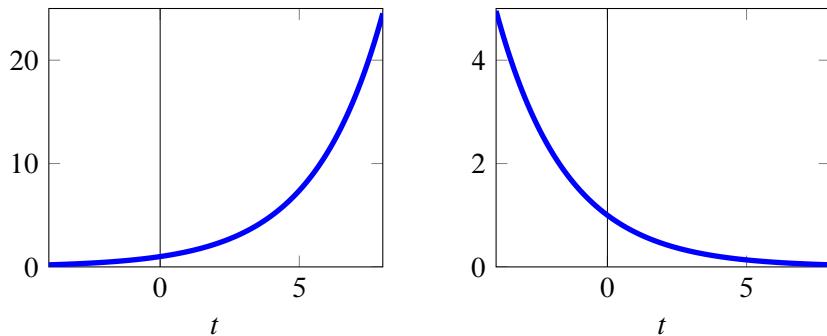


Figure 6.23 Exponential Function, $C e^{at}$. Left panel $a = 0.4$, Right panel $a = -0.4$. In each case $C = 1$.

This function has two sinusoidal components, one in the real domain and the other in the complex domain. At $t = 0$, only the real part has a value at 1.0, the imaginary part is zero. As we increase t , both real and imaginary parts oscillate, the real part as a sine wave and the imaginary part as a cosine wave. We can plot a 3D graph to show how the real and imaginary parts behave as a function of t . This plot is shown in Figure 6.24.

If we now add back the real part of the complex exponential such that $\sigma \neq 0$, that is:

$$y = e^{\sigma t} (\cos \omega t + i \sin \omega t)$$

We have spiral behavior that can either expand or contract depending on the value of σ . See Figure 6.25.

In summary, the exponential function encompasses a wide range of behaviors. Given the expression

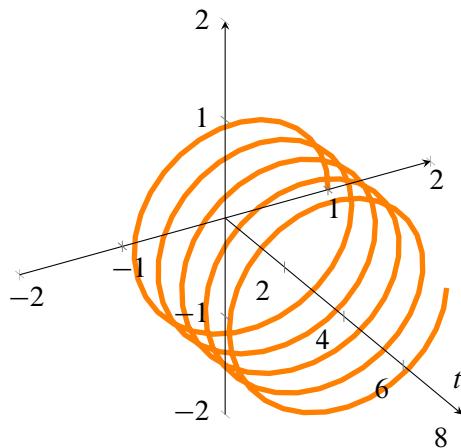


Figure 6.24 Behavior of the complex exponential: $e^{i\omega t}$.

$y = Ae^{(\sigma+i\omega)t}$, these include:

- Constant: $\sigma = 0; \omega = 0$, $y = A$
- Monotonic exponential: $\sigma \neq 0; \omega = 0$
- Sinusoidal: $\sigma = 0; \omega \neq 0$
- Exponentially varying sinusoidal: $\sigma \neq 0; \omega \neq 0$

A purely real sinusoid can be expressed as the sum of two complex sinusoids. Using Euler's formula:

$$e^{i\omega} = \cos \omega + i \sin \omega$$

$$e^{-i\omega} = \cos \omega - i \sin \omega$$

and solving for $\sin \omega$ and $\cos \omega$ yields:

$$\cos \omega = \frac{e^{i\omega} + e^{-i\omega}}{2}$$

$$\sin \omega = \frac{e^{i\omega} - e^{-i\omega}}{2i}$$

This allows us to interpret sine and cosine functions as combinations of complex sinusoids.

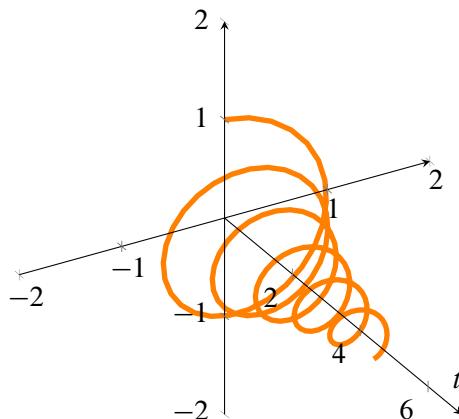


Figure 6.25 Behavior of the complex exponential: $e^{(-0.05+i\omega)t} = e^{-0.05t}e^{i\omega t}$.

Exercises

1. Write out the signal equation for a ramp with a slope k , that starts rising at 5 seconds, lasts for 6, then stops increasing.
2. Write out the signal equation for a ramp with slope k that lasts for T seconds then falls abruptly to zero.
3. Sketch the following signals:
 - (a) $u(t) - u(t - 5)$
 - (b) $e^{-t}u(t)$
 - (c) $\delta(t) + \delta(t - 3)$
 - (d) $8tu(t) - 16(t - 1)u(t - 1) + 8(t - 2)u(t - 2)$
 - (e) $2tu(t) - 2(t - 12)u(t - 12)$
4. Write out the signal function that will behave as a sawtooth. Assume the sawtooth starts at $t=0$, with height 1.0 and lasts 2.0 t units.
5. Write a Python program (or other suitable computer language) that displays sawtooth behavior using step functions and delays.
6. Using the sifting property of an impulse, compute the sifted value for e^{-t} at $t = 0$. Show your working.

7. Using the sifting property of an impulse, compute the sifted value for $\sin(t)$ at $t = 1.1$. Show your working.
8. Show that by differentiating the exponential series 6.4 that the derivative of e^x is e^x .
9. Write out the amplitude, phase, angular frequency, period and frequency in Hz for the sinusoidal:

$$y = \sin(t)$$

10. The frequency of a sine wave is 5 cycles per second. Compute the angular frequency, ω and the period, T .
11. The peak to peak voltage of a sinusoidal signal is 5 volts, what is the amplitude of the signal?
12. A flagellar motor rotates at 6000 rpm. What is the frequency in Hz and radians per minute?
13. Explain why it is true that $\sin(wt) = \cos(wt - \pi/2)$.
14. A sinusoidal voltage with a frequency of 60 Hz reaches its peak voltage of 20 volts at 2 ms after the start of the experiment. Write the cosine equation of voltage as a function of time.
15. Express the exponential decay function in terms of the time constant, τ

6.8 Appendix

Proof for Equation 6.3. Show that:

$$A_1 \sin(\omega t + \theta_1) + A_2 \sin(\omega t + \theta_2) = A \sin(\omega t + \theta)$$

We begin with the known identity:

$$\sin(\alpha + \beta) = \sin \alpha \cos \beta + \cos \alpha \sin \beta \quad (6.5)$$

From this we can write:

$$A_1 \sin(\omega t + \theta_1) = A_1 (\sin \omega t \cos \theta_1 + \cos \omega t \sin \theta_1)$$

$$A_2 \sin(\omega t + \theta_2) = A_2 (\sin \omega t \cos \theta_2 + \cos \omega t \sin \theta_2)$$

Summing the two equations and pulling out the common terms, $\sin \omega t$ and $\cos \omega t$ we obtain:

$$\begin{aligned} A_1 \sin(\omega t + \theta_1) + A_2 \sin(\omega t + \theta_2) \\ = \sin \omega t (A_1 \cos \theta_1 + A_2 \cos \theta_2) + \cos \omega t (A_1 \sin \theta_1 + A_2 \sin \theta_2) \end{aligned}$$

Let $X = A_1 \cos \theta_1 + A_2 \cos \theta_2$ and $Y = A_1 \sin \theta_1 + A_2 \sin \theta_2$:

$$A_1 \sin(\omega t + \theta_1) + A_2 \sin(\omega t + \theta_2) = \sin \omega t X + \cos \omega t Y$$

We recognize the right-hand side to be in the form of identity 6.5 such that:

$$\sin(\omega t + \theta) = \sin \omega t X + \cos \omega t Y$$

Multiply both sides by A :

$$A \sin(\omega t + \theta) = \sin \omega t AX + \cos \omega t AY$$

We can now state that:

$$X = A \cos \theta \quad Y = A \sin \theta$$

Square each term and summing we obtain:

$$X^2 + Y^2 = A^2 \cos^2 \theta + A^2 \sin^2 \theta = A^2 (\cos^2 \theta + \sin^2 \theta) = A^2$$

That is $A = \sqrt{X^2 + Y^2}$. The ratio:

$$\begin{aligned} \frac{A \sin \theta}{A \cos \theta} &= \frac{Y}{X} \\ \theta &= \tan^{-1} \frac{Y}{X} \end{aligned}$$

But $X = A_1 \cos \theta_1 + A_2 \cos \theta_2$ and $Y = A_1 \sin \theta_1 + A_2 \sin \theta_2$, squaring and summing these two terms yields:

$$\begin{aligned} (A_1 \cos \theta_1 + A_2 \cos \theta_2)^2 + (A_1 \sin \theta_1 + A_2 \sin \theta_2)^2 \\ = A_1^2 + A_2^2 + 2A_1 A_2 \cos(\theta_1 - \theta_2) \end{aligned}$$

Therefore:

$$A = \sqrt{A_1^2 + A_2^2 + 2A_1 A_2 \cos(\theta_1 - \theta_2)}$$

Likewise, the ratio, Y/X yields:

$$\frac{Y}{X} = \frac{A_1 \sin \theta_1 + A_2 \sin \theta_2}{A_1 \cos \theta_1 + A_2 \cos \theta_2}$$

so that

$$\theta = \arctan \left(\frac{A_1 \sin \theta_1 + A_2 \sin \theta_2}{A_1 \cos \theta_1 + A_2 \cos \theta_2} \right)$$

We conclude that:

$$A_1 \sin(\omega t + \theta_1) + A_2 \sin(\omega t + \theta_2) = A \sin(\omega t + \theta) \tag{6.6}$$

7

Linear Systems

“Linear systems are important because we can solve them.”

– Richard Feynman, *The Feynman Lectures on Physics*

Most of the phenomena in nature is the result of processes governed by nonlinear laws. So why the emphasis on linear systems? The answer is simple; there is no general theory for dealing with nonlinear systems. Whether there ever will be is a matter of speculation, but our inability to rationalize nonlinear systems is a huge hinderance to our understanding of complex systems such as biological cells or the brain. All is not lost however. The last three hundred years has seen the development of a complete understanding of linear systems. Given our inability to deal effectively with nonlinear systems but our complete mastery of linear systems, many studies in science and engineering focus on systems that are linear.

Engineers are particularly fond of linear systems. Many of the devices that engineers build are by design linear, for the very reason that they can be understood. Unfortunately, we don't have this luxury when studying biological systems. One way to study nonlinear systems is to simulate them on a digital or analog computer. However, anyone who has tried this will realize attempting to actually understand what is going on in the system simply from time course plots knows how difficult the task is. As an adjunct to simulation, many bioengineers will “linearize” a nonlinear system and study its linear properties. Between computer simulation and linearizing a system, we can get a better way to understanding how a particular system operates. In this chapter we will focus on linear systems, what they are, what properties they have, and when confronted with a nonlinear system how to use linearization to make the system amenable to analytical study.

7.1 Linear Systems

A linear system is one that obeys the following two constraints:

- Homogeneity
- Additivity

The two conditions can be combined to form the **superposition principle** described below.

Homogeneity

If we change an input signal by a factor α (i.e multiply by α), and the output is also changed by the same factor, then the system is said to be homogenous. If H is the system and x the input signal, then the following is true if the system is homogenous:

$$H(\alpha x) = \alpha H(x)$$

Homogeneity also implies that when the input is zero the output is also zero.

Additivity

Apply two separate inputs x_1 and x_2 to yield two outputs y_1 and y_2 . Now apply both inputs, x_1 and x_2 simultaneously to yield a new output, y_3 . If y_3 is equal to the sum $y_1 + y_2$, then the system obeys additivity. That is:

$$H(x_1 + x_2) = H(x_1) + H(x_2)$$

We can combine both rules to yield the **superposition** property:

$$H(\alpha x_1 + \beta x_2) = \alpha H(x_1) + \beta H(x_2)$$

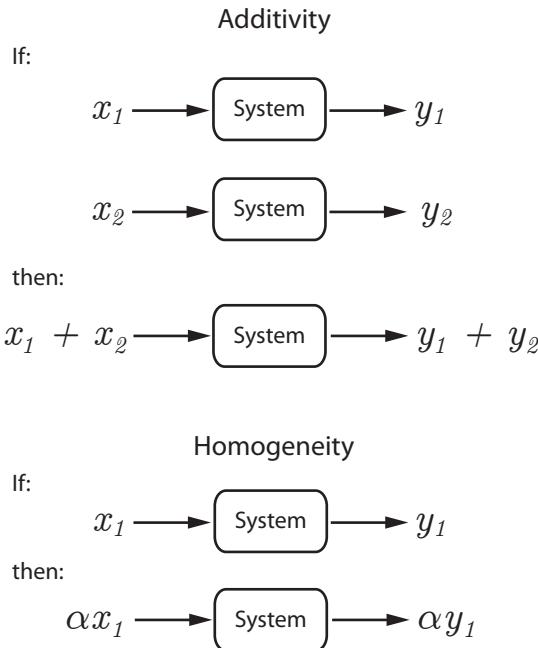
That is, the output is a superposition of the inputs.

It is additivity and homogeneity that make linear systems much easier to understand compared to nonlinear systems. In a nonlinear system the application of two different inputs will yield an output which is not the sum of the two individual responses, but instead is a more complex function, often a function that cannot be derived analytically.

Example 7.1

Consider the simple equation: $y = mx$. Homogeneity is easily satisfied since $m(\alpha x) = \alpha(mx) = \alpha y$. To check for additivity, consider two separate inputs:

$$y_1 = mx_1 \quad y_2 = mx_2$$

**Figure 7.1** Linearity.

If we now apply both inputs simultaneously, we see that additivity is also obeyed.

$$y_3 = m(x_1 + x_2) = y_1 + y_2$$

Therefore, the equation is linear.

Example 7.2

Consider the simple equation, $y = x^2$. The homogeneity test for this equation fails because:

$$(\alpha x)^2 = \alpha^2 x^2 = \alpha^2 y \neq \alpha y$$

$y = x^2$ is therefore a non-linear equation.

Non-linear equations are fairly easy to spot. Any equation where the state variables are raised to a non-unity power, or systems that involve the state variables of trigonometric or root functions are generally non-linear.

Table 7.1 illustrates some functions that are nonlinear.

Class Exercise 1

x^n	$\sqrt[n]{x}$
xy	$\sin(x)$
e^x	$\log(x)$
$(dy/dy)^n$	$V_m S/(S + K_m)$

Table 7.1 Examples of nonlinear functions.

Show that the function e^x is nonlinear.

We first form the sum of separately applied inputs, that is:

$$e^{x_1} + e^{x_2}$$

We now apply the sum simultaneously, that is:

$$e^{x_1+x_2}$$

To obey additivity, these two expressions must be equal. However, $e^{x_1+x_2} = e^{x_1}e^{x_2}$, which is not the same as $e^{x_1} + e^{x_2}$. Therefore e^x is a nonlinear function.

Similarly, we can also easily show that homogeneity is not true because it should be evident that:

$$ae^x \neq e^{ax}$$

7.2 Time-Invariance

Another property of a system that is often included as part of linearity is **time-invariance**. Systems that are linear and time-invariant are called Linear Time-Invariant Systems or LTIs. Such systems are important to engineers because the response of a LTI system can be completely characterized by the system's response to an impulse or step response. We will come back to this important topic in another chapter.

A time-invariant system is one where the system isn't an explicit function of time. This means that if a system is subjected to a perturbation at time t_0 , and experiences a response at time t_1 , then applying the same perturbation at a time $t_0 + \delta t$ will elicit the same response but now at time $t_1 + \delta t$. That is, the system's response is simply shifted in time with no other qualitative difference in the actual response. In a computer simulation it means that we can start the simulation clock at any point in time, $0, t_1, t_2, \dots$, etc and we would get the same time course solutions. For these reasons time-invariance is also sometimes called shift-invariant.

To test a system for time-invariance, we carry out the following procedure.

Step 1: Determine the output for a delayed input $x(t - a)$. This means replace every occurrence of $x(t)$ with $x(t - a)$. Let us call the resulting output $y_1(t)$.

Step 2: Delay the output itself, i.e. $y(t - a)$. This means replace every occurrence of t with $t - a$. Let us call the resulting output $y_2(t)$.

Step 3: The system is time-invariant if $y_1(t) = y_2(t)$.

Operational definition of time-invariance: If the input, $x(t)$ to a system yields a response $y(t)$, and we shift the time at which we apply the input to $x(t + \Delta t)$, then the same response will occur at $y(t + \Delta t)$.

$$x(t) \longrightarrow [H_1] \longrightarrow y(t) \qquad x(t - \tau) \longrightarrow [H_1] \longrightarrow y(t - \tau)$$

Figure 7.2 Time-Invariance: Two inputs, one at t and a second delayed by τ . In a time-invariant system, both outputs are the same except that the second one is delayed by τ .

Example 7.3

Is the following system time-invariant?

$$y(t) = \sin(x(t))$$

We first input a delayed $x(t)$, that is $x(t - a)$, to generate $y_1(t)$. Replace every occurrence of $x(t)$ with $x(t - a)$.

$$y_1(t) = \sin(x(t - a))$$

Next, we generate a time shifted output to yield $y_2(t)$. Replace every occurrence of t with $(t - a)$:

$$y_2(t) = \sin(x(t - a))$$

We can see that $y_1(t) = y_2(t)$, therefore the system is time-invariant.

Example 7.4

Is the following system time-invariant?

$$y(t) = tx(t)$$

We first input a delayed $x(t)$, that is $x(t - a)$, to generate $y_1(t)$. Replace every occurrence of $x(t)$ with $x(t - a)$.

$$y_1(t) = t(x(t - a))$$

Next, we generate a time shifted output to generate $y_2(t)$. Replace every occurrence of t with $(t - a)$:

$$y_2(t) = (t - a)(x(t - a))$$

We see that $y_1(t) \neq y_2(t)$, therefore the system is time varying.

Example 7.5

Is the following system time-invariant?

$$y(t) = e^{-t}(x(t))$$

We first input a delayed $x(t)$, that is $x(t-a)$, to generate $y_1(t)$. Replace every occurrence of $x(t)$ with $x(t-a)$.

$$y_1(t) = e^{-t}(x(t-a))$$

Next we generate a time shifted output to generate $y_2(t)$. Replace every occurrence of t with $(t-a)$:

$$y_2(t) = e^{-(t-a)}(x(t-a))$$

We see that $y_1(t) \neq y_2(t)$, therefore the system is time varying.

Spotting a time variant system is often a case of observing if the system equations are an explicit function of time. Linearity and time-invariance are important because the dynamics of systems which agree with these assumptions can be fully described and understood. As such we can say:

A Linear Time-Invariant System or LTI is a system that is both linear and time-invariant.

7.3 Linearization

To linearize means replacing the nonlinear version of the equation with a linear approximation. Such approximations are only valid when subjected to **small changes** around the point of linearization, also called the **operating point**. It should be emphasized that in the process, we lose valuable information but enough is retained to make linearization an extremely useful tool.

To appreciate the difference between linear and nonlinear functions, consider the system $y = x^2$. Let us apply two separate inputs, x_1 and x_2 to give outputs x_1^2 and x_2^2 . If we now apply the inputs simultaneously, that is $y = (x_1 + x_2)^2$, we obtain $x_1^2 + x_2^2 + 2x_1x_2$. We see that the output is not simply $x_1^2 + x_2^2$, but includes an additional term, $2x_1x_2$. This term is the nonlinear contribution. Imagine that this difference now enters further nonlinear processes, leading to further changes. Eventually the output looks nothing like the input. This small change makes most nonlinear systems difficult to understand and analyze.

Unless the system is degenerate (i.e. has an infinite number of solutions) or has the trivial solution (i.e./ the solution is zero), linear systems generally admit only one solution. In contrast, it is possible for nonlinear systems to admit multiple solutions, that is given a single input, a nonlinear system can regurgitate one of a number of possible distinct outputs. To makes matters worse, in the majority of

mathematical models we find in biochemical networks, it is not even possible to find the solutions mathematically. That is, we cannot actually describe mathematically how an output depends on an input other than by doing brute-force computer simulation. Understanding nonlinear systems whether we find them in biology or elsewhere is a huge unresolved problem.

While there is a complete body of theory for linear systems, no such equivalent exists for nonlinear systems. When dealing with nonlinear systems we are often forced to use either computer simulation or linear approximations.

If we were to draw a nonlinear curve on a graph and then zoom into the graph, the curve would eventually look like a straight line; that is we can turn a nonlinear system into a linear one but only in small regions of the system's behavior where linearity dominates. This process is called **linearization** and is a powerful technique for studying nonlinear systems.

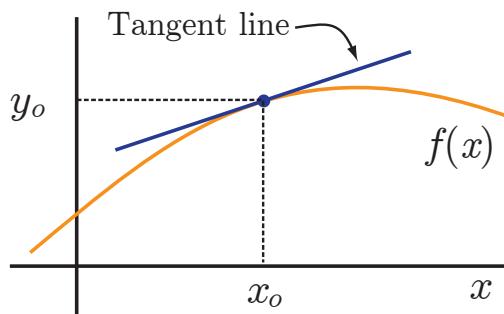


Figure 7.3 Linearization at point x_o by drawing the tangent line.

The essence of linearization is to select a point on the nonlinear function curve and draw a tangent at that point. The tangent represents the linearized version (Figure 7.3). Notice that the approximation is only exact at the tangent point, x_o . As we move away from x_o the approximations gets worse and worse. It is quite easy to derive the equation for the tangent line if we remember that the slope of a line is given by:

$$m = \frac{df}{dx}$$

where f is the function and x the independent variable. The line at a point x_o is given by:

$$m = \frac{y - y_o}{x - x_o}$$

where y and x are points that lie on the tangent line. Rearranging yields:

$$y = y_o + m(x - x_o) \tag{7.1}$$

For example, let's say we wanted to linearize the square function, $y = x^2$ at $x = x_o$. The first thing to do is find the slope, m , of the function at x_o . That is the derivative of x^2 at x_o :

$$m = 2x_o$$

Inserting m into the tangent line equation (7.1) yields:

$$y = y_o + 2x_o(x - x_o)$$

We know that $y_o = x_o^2$ so that:

$$y = x_o^2 + 2x_o x - 2x_o^2 = 2x_o x - x_o^2$$

In this equation x_o is strictly a constant. For example, we might linearize x^2 around $x_o = 4$, which will yield:

$$y = 8x - 16$$

In more advanced texts, linearization is normally introduced by way of the Taylor series (See Appendix B for review). This is a way of approximating a mathematical function by using an infinite polynomial series such as the following:

$$f(x) = c_o + c_1 x + c_2 x^2 + c_3 x^3 + \dots \quad (7.2)$$

It is possible to represent any continuous function using such a polynomial. For example, we can represent $\sin(x)$ using the formula:

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

Without going into the details (See Appendix B), the c_i terms in the polynomial series (7.2) can be determined such that the series will approximate any given continuous function. The Taylor series is always defined around some operating point, x_o , and is expressed in terms of a distance from that operating point, $x - x_o$. The Taylor series is given by:

$$f(x) = f(x_o) + \frac{\partial f}{\partial x}(x - x_o) + \frac{1}{2!} \frac{\partial^2 f}{\partial x^2}(x - x_o)^2 + \dots + \frac{1}{n!} \frac{\partial^n f}{\partial x^n}(x - x_o)^n + \dots \quad (7.3)$$

The various derivatives in the Taylor series **must** be evaluated at x_o . The function $f(x)$ must be continuous in order to compute the derivatives; there should be no holes or sudden breaks (discontinuity) in the curve that the function describes. The number of terms in the Taylor series determines the degree of approximation. The fewer the number of terms, the more approximate the series is. For example, the most approximate expression is given by using only the first term, $f(x_o)$. However,

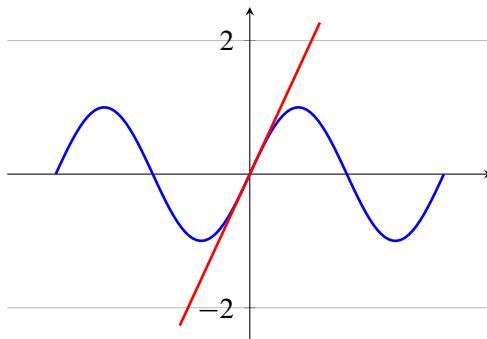


Figure 7.4 Linearized $\sin(t)$ function represented by the straight line through zero.

$f(x_o)$ is a constant so this represents a very poor approximation. To make the approximation more useful, we can include the first two terms of the Taylor series:

$$f(x) \approx f(x_o) + \frac{\partial f}{\partial x_o}(x - x_o) \quad (7.4)$$

This equation has the same form as the tangent line equation shown previously, Equation (7.1). We emphasize again that the derivative must be computed at the operating point, x_o . Provided x is close to x_o , the approximation is good. For example, let us form the Taylor series for the function, $y = \sin(x)$ around $x_o = 0$. We should recall that $\sin(0) = 0$ and $\cos(0) = 1$, then:

$$\begin{aligned} y &= \sin(0) + \frac{\partial \sin(x_o)}{\partial x}(x - 0) + \frac{1}{2!} \frac{\partial^2 \sin(x)}{\partial x^2}(x - 0) + \dots \\ y &= 0 + 1x + 0 - \frac{1}{3!}x^3 + 0 + \frac{1}{5!}x^5 + \dots \end{aligned}$$

The linear approximation is simply, $y = x$ (Figure 7.4). We have linearized the sin function.

To illustrate linearization with another example consider the equation we looked at before, $y = x^2$. To start we must first choose an operating point around which we will linearize, for example, $x_o = 2$. According to the second term in the Taylor series, we need to find the derivative, df/dx so that the first two terms of the Taylor series (Equation 7.4) become:

$$f(x) = f(2) + 2x_o(x - 2)$$

To obtain the linear approximation we evaluate the derivative at the operating point ($x_o = 2$), that is $df/dx_o = 2x_o = 4$. The final linear approximation is given by:

$$f(x) = 4 - 4(x - 2) = 4x - 4$$

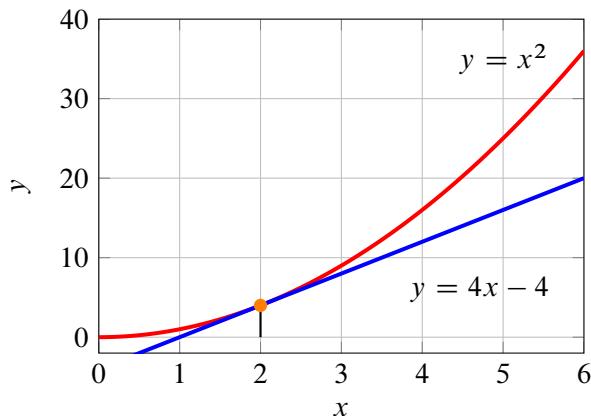


Figure 7.5 First order Taylor series approximation ($y = 4x - 4$) of $y = x^2$ at the operating point, $x_o = 2$.

Figure 7.5 shows the original nonlinear function together with the linear approximation.

Equation 7.4 is also commonly written in the form:

$$f(x) \simeq f(x_o) + \frac{df}{dx_o} \delta x \quad (7.5)$$

where $\delta x = (x - x_o)$. If the equation f is a function of more than one variable, then additional terms appear. For example, the linearization of $f(x, y)$ will give:

$$f(x, y) = f(x_o, y_o) + \frac{\partial f}{\partial x} \delta x + \frac{\partial f}{\partial y} \delta y$$

As stated previously the derivatives must be evaluated at the operating point, x_o, y_o .

Linearizing a Single ODE

In the previous section we explained how to linearize simple algebraic equations. Here we will show that differential equations can also be linearized. Consider the ordinary differential equation where u is an input to the system, and x a state variable:

$$\frac{dx}{dt} = f(x, u)$$

Let us linearize around the steady state operating point x_o and u_o , that is around the point $f(x_o, u_o) = 0$, such that:

$$\frac{dx}{dt} = f(x_o, u_o) = 0$$

The Taylor expansion (See F.12 for more on multidimensional Taylor series) around the steady state is:

$$\frac{dx}{dt} \approx \frac{\partial f(x_o, u_o)}{\partial x}(x - x_o) + \frac{\partial f(x_o, u_o)}{\partial u}(u - u_o)$$

Note that the first term $f(x_o, u_o)$ is zero and doesn't appear in the expression. Let us define $\delta x = x - x_o$ and $\delta u = u - u_o$. Differentiating $\delta x = x - x_o$ with respect to time yields:

$$\frac{d\delta x}{dt} = \frac{dx}{dt}$$

Note that x_o is independent of time. We can therefore write the Taylor series approximation as:

$$\frac{d\delta x}{dt} \approx \frac{\partial f(x_o, u_o)}{\partial x}\delta x + \frac{\partial f(x_o, u_o)}{\partial u}\delta u$$

The important point to note about this result is that the Taylor approximation gives us an equation that describes the **rate of change of a perturbation** in x , that is δx . Let us replace the partial derivatives by the symbols A and B so that:

$$\frac{d\delta x}{dt} \approx A\delta x + B\delta u$$

An important point to emphasize is that around the steady state equilibrium point, the linearization has *no offset*, $f(x_o)$ as we saw in equation (7.5). This is an important point to remember in later chapters when we linearize systems around the steady state.

Linearizing a System of ODEs

Linearizing a system of differential equations is a simple extension of linearizing a single equation. Consider the set of two differential equations:

$$\frac{dx}{dt} = f(x, y)$$

$$\frac{dy}{dt} = g(x, y)$$

Let us linearize around the steady state for this system where the steady state is given by (x_o, y_o) . Expand these equations using a Taylor series and ignore higher terms to yield:

$$\frac{dx}{dt} \approx F(x_o, y_o) + \frac{\partial F(x_o, y_o)}{\partial x_o}(x - x_o) + \frac{\partial F(x_o, y_o)}{\partial y_o}(y - y_o)$$

$$\frac{dy}{dt} \approx G(x_o, y_o) + \frac{\partial G(x_o, y_o)}{\partial x_o}(x - x_o) + \frac{\partial G(x_o, y_o)}{\partial y_o}(y - y_o)$$

However, at steady state $F(x_o, y_o)$ and $G(x_o, y_o)$ equal zero so that:

$$\begin{aligned}\frac{dx}{dt} &\approx \frac{\partial F(x_o, y_o)}{\partial x_o}(x - x_o) + \frac{\partial F(x_o, y_o)}{\partial y_o}(y - y_o) \\ \frac{dy}{dt} &\approx \frac{\partial G(x_o, y_o)}{\partial x_o}(x - x_o) + \frac{\partial G(x_o, y_o)}{\partial y_o}(y - y_o)\end{aligned}\quad (7.6)$$

Chapter 5 introduced the matrix called the Jacobian (5.9)(5.15):

$$\mathbf{J} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_m}{\partial x_1} & \cdots & \frac{\partial f_m}{\partial x_n} \end{bmatrix}$$

Substituting the Jacobian in equation (7.6) yields:

$$\begin{bmatrix} \frac{dx}{dt} \\ \frac{dy}{dt} \end{bmatrix} \approx \begin{bmatrix} \frac{\partial F}{\partial x_o} & \frac{\partial F}{\partial y_o} \\ \frac{\partial G}{\partial x_o} & \frac{\partial G}{\partial y_o} \end{bmatrix} \begin{bmatrix} x - x_o \\ y - y_o \end{bmatrix}$$

If the differential equations are also a function of one or more inputs ($u_1 \dots u_m$), then the derivation can be extended such that:

$$\begin{bmatrix} \frac{dx}{dt} \\ \frac{dy}{dt} \end{bmatrix} \approx \begin{bmatrix} \frac{\partial F}{\partial x_o} & \frac{\partial F}{\partial y_o} \\ \frac{\partial G}{\partial x_o} & \frac{\partial G}{\partial y_o} \end{bmatrix} \begin{bmatrix} x - x_o \\ y - y_o \end{bmatrix} + \begin{bmatrix} \frac{\partial F}{\partial u_1^o} & \cdots & \frac{\partial F}{\partial u_m^o} \\ \frac{\partial G}{\partial u_1^o} & \cdots & \frac{\partial G}{\partial u_m^o} \end{bmatrix} \begin{bmatrix} u_1 - u_1^o \\ \vdots \\ u_m - u_m^o \end{bmatrix}$$

where u_i^o is the nominal value for the i th input. If we represent the Jacobian by the matrix symbol \mathbf{A} , the matrix related to the input as \mathbf{B} (often called the control matrix), and the change in state variable and inputs by $\delta\mathbf{x}$ and $\delta\mathbf{u}$, is then:

$$\frac{d}{dt}\delta\mathbf{x} \approx \frac{\partial \mathbf{F}_i}{\partial \mathbf{x}_j}\delta\mathbf{x} + \frac{\partial \mathbf{F}_i}{\partial \mathbf{u}_j}\delta\mathbf{u} \quad (7.7)$$

$$\frac{d}{dt}\delta\mathbf{x} \approx \mathbf{A}\delta\mathbf{x} + \mathbf{B}\delta\mathbf{u} \quad (7.8)$$

This equation tells us how a disturbance $\delta\mathbf{x}$ changes in time given some initial disturbance to the state $\delta\mathbf{x}$ and/or a disturbance to the system inputs $\delta\mathbf{u}$.

Example 7.6

Linearize the following system of differential equations around the steady state point ($dx/dt = 0, dy/dt = 0$) when $x = x_o$ and $y = y_o$, with given state x, y and input u :

$$\frac{dx}{dt} = u - x^2$$

$$\frac{dy}{dt} = x^2 - y$$

Compute the Jacobian, \mathbf{A} , with respect to the state and the control matrix, \mathbf{B} , with respect to the input, u .

$$\mathbf{A} = \begin{bmatrix} -2x & 0 \\ 2x & -1 \end{bmatrix} \quad \mathbf{B} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

Combining these terms into equation 7.8 yields the linearized system:

$$\frac{d}{dt} \delta \mathbf{x} \approx \begin{bmatrix} -2x & 0 \\ 2x & -1 \end{bmatrix} \begin{bmatrix} \delta x \\ \delta y \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \end{bmatrix} \delta u$$

Example 7.7

Consider a simple nonlinear system, a two step pathway with the first step governed by an irreversible mass-action rate law and the second step by an irreversible Michaelis-Menten rate laws. The pathway has a single species, x which is governed by the nonlinear differential equation:

$$\frac{dx}{dt} = k_1 X_o - \frac{Vm x}{Km + x}$$

Let us assume a single input, the Vm . The steady state level for x is denoted by x_{ss} . We now linearize around this point by evaluating the \mathbf{A} and \mathbf{B} matrices:

$$\delta \dot{x} = \frac{\partial(dx/dt)}{\partial x} \delta x + \frac{\partial(dx/dt)}{\partial Vm} \delta Vm$$

Each derivative is computed at x_{ss} . Evaluating the derivatives yields the linear differential equation:

$$\delta \dot{x} = - \left(\frac{Vm Km}{(Km + x_{ss})^2} \delta x + \frac{x_{ss}}{(Km + x_{ss})} \delta Vm \right)$$

which describes the rate of change of perturbation in response to a step change in Vm . Note that the terms that include x_{ss} are constant and equivalent to A and B in the state space representation, so that the equation can be reduced to

$$\delta \dot{x} = -(A \delta x + B \delta Vm)$$

which is a linear equation. The solution to this equation is given by:

$$\delta x(t) = [e^{-At} - 1] \frac{B \delta Vm}{A} + \delta x_o e^{-At}$$

where δx_o is the delta initial perturbation in x_{ss} . This equation describes the time evolution in δx as a result of a perturbation in δV_m and/or δx . Note that the equation only applies to small changes in δV_m and δx because of the linearization.

If we assume that the initial condition for δx to equal to zero (i.e. $\delta x_o = 0$, no perturbation in x), then the steady state solution (obtained as t goes to infinity) is:

$$\delta x = -\frac{B\delta V_m}{A}$$

At $t = 0$ the perturbation is zero as defined by the initial condition but as t advances, $\delta x(t)$ goes negative indicating that a perturbation in V_m results in a decline in the steady state level of x . As time continues to advance, δx reaches a new steady state given by $-B\delta V_m/A$. Note, this is the delta change in x not the absolute value of the new level of x , which is why the negative sign makes sense here. The absolute level of the new steady state level of x is given by $x_{ss} - A/(B\delta V_m)$.

If on the other hand δV_m is zero but δx_o is not, then $B\delta V_m = 0$ so that the evolution equation is given by $\delta x_o e^{-At}$. As t advances, this term decays to zero so that at the new steady state $\delta x = 0$, that is the system relaxes back to its original state.

Exercises

1. Show whether the following relationships are linear or nonlinear with respect to x :

- (a) $y = x^2$
- (b) $y = \frac{dx}{dt}$
- (c) $\frac{dy}{dt} = ay + x$
- (d) $y = mx + b$

2. Show algebraically that the following equation is time varying:

$$y(t) = tx(t)$$

Show graphically that the equation is time varying.

3. Show whether the following systems are time-invariant or varying:

- (a) $y(t) = Ax(t) + B$
- (b) $y(t) = \cos(3t)x(t)$
- (c) $y(t) = x(t-1) + 2tx(t-2)$

4. Linearize the following equations:

$$(a) y = \sqrt{x} \text{ at } x = 4$$

- (b) $y = \sqrt[3]{x}$ at $x = 8$
(c) $y = 2 \ln(x)$ at $x = 1$
(d) $f(x, y) = y^2 e^{x+y}$ at $x = 2; y = 1$

5. Linearize the following differential equations at the operating point x_o :

(a)

$$\frac{dx}{dt} = x^2 + 10$$

$$\frac{dx}{dt} = x^3 + 2x - 6$$

6. Linearize the following system of differential equations at the (0,0) operating point:

$$\frac{dx}{dt} = 3x - y^2$$

$$\frac{dy}{dt} = \sin(y) - 4x$$

8

State Space Representation

“Space is big. You just won’t believe how vastly, hugely, mind-bogglingly big it is.”

– Douglas Adams, *The Hitchhiker’s Guide to the Galaxy*

8.1 State Space Representation

As we’ve seen in a previous chapter, we can represent a general system as follows (Figure 8.1):

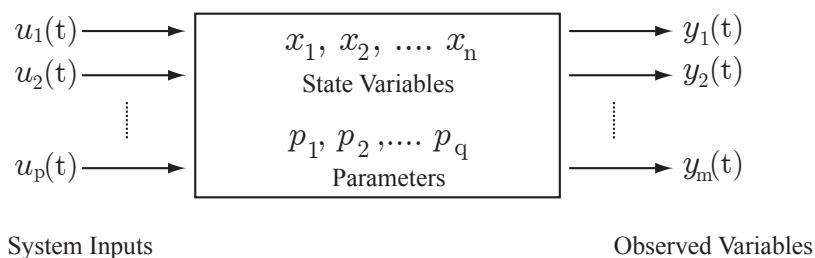


Figure 8.1 Systems Diagram with Inputs and Outputs.

This representation has four types of symbols: the state variables, x_i that represent the quantities that define the behavior of the system; the inputs, u_i , which represent quantities that are fixed properties that can be controlled by the observer; the fixed system parameters p_i , and the system outputs, y_i ,

which represent the actual observables. In simple systems the state variables are often the same as the observables, but sometimes one can only indirectly access the state variable via other kinds of measured quantities (y_i).

To give a concrete example of a system, consider a simple genetic circuit that expresses green fluorescence protein that responds to changes in an inducer molecule. Let the state variable of the system be the concentration of expressed protein; this would be a x_i type quantity in Figure 8.1. However we can't actually observe the protein directly, only as a fluorescent measurement in an instrument. The fluoresce we observe in the experiment is represented by a y_i variable in Figure 8.1. Finally, the inducer that we apply to the system to change the level of gene expression is an external parameter, u_i .

Let us represent the vector of state variables, x , by:

$$\mathbf{x}(t) = [x_1(t), x_2(t), \dots, x_n(t)]^T$$

the vector of inputs, u , by:

$$\mathbf{u}(t) = [u_1(t), u_2(t), \dots, u_p(t)]^T$$

and the vector of output variables by:

$$\mathbf{y}(t) = [y_1(t), y_2(t), \dots, y_m(t)]^T$$

In general the relationship between the state variables, x_i , and the input parameters, u_i , is nonlinear and can be written as:

$$\frac{d\mathbf{x}}{dt} = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t))$$

where \mathbf{x} and \mathbf{u} are vectors. Likewise, the vector of observables, \mathbf{y} , is related to the state variables and parameters by some nonlinear function, \mathbf{g} :

$$\mathbf{y}(t) = \mathbf{g}(\mathbf{x}(t), \mathbf{u}(t))$$

\mathbf{x} , \mathbf{u} and \mathbf{y} are in general functions of time. For brevity, we will omit the (t) in the following discussion. It is common to linearize these nonlinear system equations around the equilibrium or steady state point. If we let \mathbf{x}_o and \mathbf{u}_o be the values at steady state, then by definition it must be the case that:

$$0 = \mathbf{f}(\mathbf{x}_o, \mathbf{u}_o)$$

We now expand $\mathbf{f}(\mathbf{x}, \mathbf{u})$ around \mathbf{x}_o and \mathbf{u}_o using a Taylor series to obtain:

$$\frac{d\mathbf{x}}{dt} = \mathbf{f}(\mathbf{x}_o, \mathbf{u}_o) + \frac{\partial \mathbf{f}}{\partial \mathbf{x}_o}(\mathbf{x} - \mathbf{x}_o) + \frac{\partial \mathbf{f}}{\partial \mathbf{u}_o}(\mathbf{u} - \mathbf{u}_o) + \dots$$

where $\mathbf{x} - \mathbf{x}_o = \delta\mathbf{x}$ and $\mathbf{u} - \mathbf{u}_o = \delta\mathbf{u}$ and define:

$$\mathbf{A} = \frac{\partial \mathbf{f}}{\partial \mathbf{x}_o}$$

$$\mathbf{B} = \frac{\partial f}{\partial \mathbf{u}_o}$$

Since $\mathbf{x} = \delta\mathbf{x} - \mathbf{x}_o$ and \mathbf{x}_o is a constant:

$$\frac{d\mathbf{x}}{dt} = \frac{d\delta\mathbf{x}}{dt}$$

substituting these terms yields:

$$\frac{d}{dt}\delta\mathbf{x} = \mathbf{A}\delta\mathbf{x} + \mathbf{B}\delta\mathbf{u}$$

This is a linear time independent equation (to be justified later) that describes the *rate of change of the perturbation* in \mathbf{x} around the steady state as a result of initial perturbations in \mathbf{x} and/or \mathbf{u} .

Likewise, the function \mathbf{g} can also be linearized to obtain:

$$\delta\mathbf{y} = \mathbf{C}\delta\mathbf{x} + \mathbf{D}\delta\mathbf{u}$$

where:

$$\mathbf{C} = \frac{\partial \mathbf{g}}{\partial \mathbf{x}_o} \quad \text{and} \quad \mathbf{D} = \frac{\partial \mathbf{g}}{\partial \mathbf{u}_o}$$

The matrices \mathbf{A} , \mathbf{B} , \mathbf{C} and \mathbf{D} are constant and *do not* depend on time or the time dependent state variables. In the control literature, these matrices are given the following labels:

- \mathbf{A} $n \times n$ State Matrix, or Jacobian
- \mathbf{B} $n \times p$ Control Matrix
- \mathbf{C} $m \times n$ Output Matrix
- \mathbf{D} $m \times p$ Feed-forward Matrix

If the \mathbf{C} matrix is the identity matrix and the \mathbf{D} matrix is zero, then the state variable vector is equal to the output vector, \mathbf{y} . The linearized state equations are often represented in the literature by equation (8.1). For many systems the \mathbf{D} matrix is empty. A block diagram equations (8.1) is given in Figure 8.2, and shows why the \mathbf{D} matrix is called the feed-forward matrix.

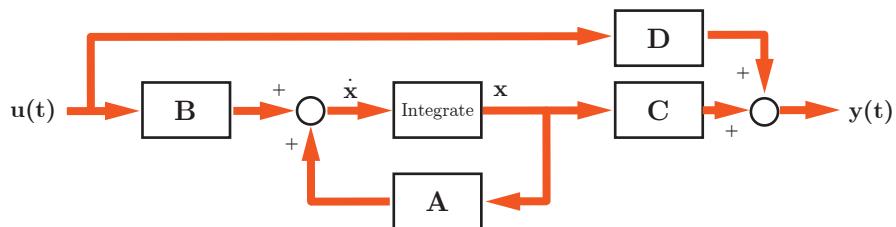


Figure 8.2 Block diagram representation of the state space formulation.

State Space Equations:

$$\frac{d\mathbf{x}}{dt} = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t) \quad (8.1)$$

$$\mathbf{y}(t) = \mathbf{C}\mathbf{x}(t) + \mathbf{D}\mathbf{u}(t)$$

As we will discover in the next section, these equations are both linear and time-invariant and are called the **state space equations**.

Written out, the state space equation for \mathbf{x} looks like:

$$\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} + \begin{bmatrix} b_{11} & b_{12} & \dots & b_{1p} \\ b_{21} & b_{22} & \dots & b_{2p} \\ \vdots & \vdots & & \vdots \\ b_{n1} & b_{n2} & \dots & b_{np} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_p \end{bmatrix}$$

Note that the \mathbf{A} matrix is *always square*. The \mathbf{A} matrix is also called the **Jacobian matrix**. Note that the number of inputs u , is entirely at the discretion of the model builder. It is possible for example to have no inputs, one input or many inputs. The easiest way to construct the \mathbf{A} and \mathbf{B} matrices is to differentiate the differential equations with respect to the state variables for the \mathbf{A} matrix and the inputs for the \mathbf{B} matrix.

Each entry in \mathbf{A} is given by:

$$A_{ij} = \frac{dx_i/dt}{dx_j} \quad (8.2)$$

The \mathbf{A} matrix in the state space representation is called the Jacobian Matrix, where:

$$A_{ij} = \frac{dx_i/dt}{dx_j}$$

if $f_i = dx_i/dt$ then:

$$\mathbf{A} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \dots & \frac{\partial f_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_m}{\partial x_1} & \dots & \frac{\partial f_m}{\partial x_n} \end{bmatrix}$$

Each entry in \mathbf{B} is given by:

$$\mathbf{B}_{ij} = \frac{dx_i/dt}{du_j} \quad (8.3)$$

Linearity and Time Invariance

Before proceeding we should ask whether the state space equations (8.1) are linear and time invariant. Let us first look at the state variable equation:

$$\frac{d\mathbf{x}}{dt} = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t) \quad (8.4)$$

Linearization ensures that the \mathbf{A} and \mathbf{B} matrices are constant and therefore not functions of time. This means that the left-hand side of the equation is independent of time. The equation is therefore time invariant.

To test for linearity we need to show that a combined input $\mathbf{u}(t) = \alpha\mathbf{u}_1(t) + \beta\mathbf{u}_2(t)$ yields the output $\mathbf{x}(t) = \alpha\mathbf{x}_1(t) + \beta\mathbf{x}_2(t)$. Let us first apply two separate and independent inputs, \mathbf{u}_1 and \mathbf{u}_2 such that this results in the response \mathbf{x}_1 and \mathbf{x}_2 respectively. We can write these as:

$$\frac{d\mathbf{x}_1}{dt} = \mathbf{A}\mathbf{x}_1 + \mathbf{B}\mathbf{u}_1 \quad \text{and} \quad \frac{d\mathbf{x}_2}{dt} = \mathbf{A}\mathbf{x}_2 + \mathbf{B}\mathbf{u}_2$$

Let us consider the combined response, where $\mathbf{x}(t) = \alpha\mathbf{x}_1(t) + \beta\mathbf{x}_2(t)$, substituting this into equation (8.4) yields:

$$\frac{d(\alpha\mathbf{x}_1 + \beta\mathbf{x}_2)}{dt} = \mathbf{A}(\alpha\mathbf{x}_1(t) + \beta\mathbf{x}_2(t)) + \mathbf{B}\mathbf{u}(t)$$

To discover the nature of the expression $\mathbf{B}\mathbf{u}(t)$, we expand terms and move everything except $\mathbf{B}\mathbf{u}(t)$ to the left side:

$$\frac{\alpha d\mathbf{x}_1}{dt} + \frac{\beta d\mathbf{x}_2}{dt} - \mathbf{A}(\alpha\mathbf{x}_1(t) + \beta\mathbf{x}_2(t)) = \mathbf{B}\mathbf{u}(t)$$

Collecting α and β terms:

$$\alpha \left(\frac{d\mathbf{x}_1}{dt} - \mathbf{A}\mathbf{x}_1(t) \right) + \beta \left(\frac{d\mathbf{x}_2}{dt} - \mathbf{A}\mathbf{x}_2(t) \right) = \mathbf{B}\mathbf{u}(t)$$

However

$$\frac{d\mathbf{x}_1}{dt} - \mathbf{A}\mathbf{x}_1(t) = \mathbf{B}\mathbf{u}_1(t) \quad \text{and} \quad \frac{d\mathbf{x}_2}{dt} - \mathbf{A}\mathbf{x}_2(t) = \mathbf{B}\mathbf{u}_2(t)$$

so that:

$$\alpha\mathbf{B}\mathbf{u}_1(t) + \beta\mathbf{B}\mathbf{u}_2(t) = \mathbf{B}\mathbf{u}(t) \quad \text{and} \quad \alpha\mathbf{u}_1(t) + \beta\mathbf{u}_2(t) = \mathbf{u}(t)$$

Hence the system (8.4) is linear. This doesn't preclude $\mathbf{u}(t)$ from being nonlinear.

8.2 Examples of State Space Models

Mechanical Model

Consider the simple mechanical system shown in Figure 8.3.

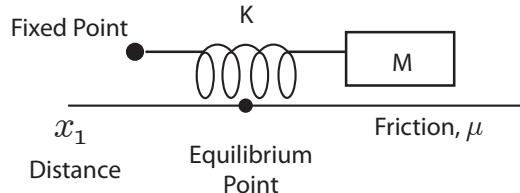


Figure 8.3 A mass, M , is attached to spring (K) which in turn is attached to a fixed point. The mass rests on a surface with friction, f .

Let us designate the distance moved relative to the equilibrium point as x_1 . If the system moves to the left of the equilibrium point we assume $x_1 < 0$, otherwise we assume $x_1 > 0$. At equilibrium $x_1 = 0$. dx_1/dt is the velocity of the movement. If we apply an impulse, $\delta(t)$, to the mass, it will begin to oscillate as it attempts to return to the equilibrium point. If we include a frictional force on the table, then the movement of the mass will slow down until it rests again at the equilibrium point. We will assume that the velocity of the mass is given by x_2 . This means the acceleration, given by the rate of change of the velocity, will be:

$$a = \frac{dx_2}{dt}$$

We will write two differential equations, one for the velocity and another for the acceleration. To do this we must consider all forces that contribute to the dynamics in this system. There are three forces involved: the force we apply when we displace the mass, the force that is exerted by the spring, and the force exerted by the friction on the table. The total force, F_t , on the mass is therefore:

$$F_t = F_{\text{impulse}} + F_{\text{spring}} + F_{\text{friction}}$$

First, consider the spring. Hooke's law states that the displacement experienced by a spring from its equilibrium point is proportional to the applied force, F . Moreover, because the force exerted by the spring opposes the direction of the displacement, we include a negative sign in the relationship. That is:

$$F_s = -kx$$

where k is called the spring constant. The next force to consider is the frictional force. Friction always opposes motion and studies indicate that the frictional force is proportional to the rate at which the

mass moves, in other words:

$$F_f = -\mu \frac{dv}{dt} = -\mu \frac{dx_1}{dt} = -\mu x_2$$

where μ is the coefficient of friction. The force relationship is negative because friction opposes movement. The final force to consider is the displacement we apply to the mass at $t = 0$. This will be an impulse such that at $t < 0$, the system is at equilibrium but at $t = 0$, we apply an impulse, $\delta(t)$, that causes the system to respond. We can therefore write that the total force on the mass is:

$$F_{total} = \delta(t) - kx_1 - \mu x_2$$

From Newton's second law we know that the force on a mass is equal to the acceleration times the mass of the object. That is:

$$\frac{dx_2}{dt} M = \delta(t) - kx_1 - \mu x_2$$

or

$$\frac{dx_2}{dt} = \frac{\delta(t)}{M} - \frac{k}{M}x_1 - \frac{\mu}{M}x_2$$

This equation combined with the velocity equation gives us the complete system of equations for this system:

$$\begin{aligned}\frac{dx_1}{dt} &= x_2 \\ \frac{dx_2}{dt} &= \frac{\delta(t)}{M} - \frac{k}{M}x_1 - \frac{\mu}{M}x_2\end{aligned}$$

In this case it should be evident how to construct the A and B matrices by inspection. However they can also be constructed by differentiation (See Equations 8.2 and 8.3). This would be a necessity for nonlinear models. We can rewrite these equations in the state space representation as:

$$\begin{bmatrix} \frac{dx_1}{dt} \\ \frac{dx_2}{dt} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\frac{k}{M} & -\frac{\mu}{M} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{M} \end{bmatrix} \begin{bmatrix} \delta(t) \end{bmatrix}$$

Note that the impulse function is separated out from the rest of the model because it represents the input to the system.

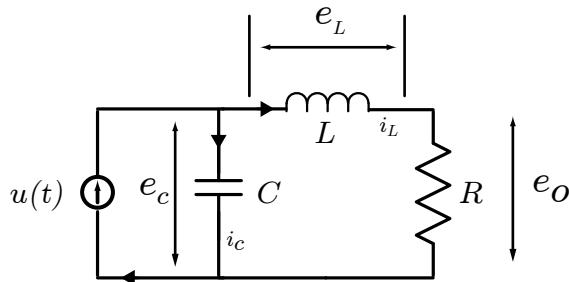


Figure 8.4 Electrical circuit comprising a fixed current source, $u(t)$ together with a capacitor (C), inductor (L) and resistor (R). i is the current and e the voltage.

Electrical Model

Consider the electrical circuit shown in Figure 8.4. Let us note the following basic relationships:

$$e_o = R i_L$$

$$i_c = C \frac{de_c}{dt}$$

$$e_L = L \frac{di_L}{dt}$$

$$e_c = e_L + e_o$$

$$u(t) = i_c + i_L$$

where L is the inductance, R the resistance, i the current, e the voltage and $u(t)$ the constant current source at $t > 0$. The circuit can be described by two differential equations, one describing the rate of change of the voltage across the capacitor and another that describes the rate of change of current through the inductor. From the capacitance law we can write:

$$\frac{de_c}{dt} = \frac{i_c}{C}$$

but since $i_c = u(t) - i_L$:

$$\frac{de_c}{dt} = \frac{u(t)}{C} - \frac{i_L}{C} \quad (8.5)$$

This provides one of the equations we need. Given that $e_c = e_L + e_o$, the inductor law, $e_L = L di_L/dt$, can be rewritten as:

$$L \frac{di_L}{dt} = e_c - e_o = e_c - Ri_L$$

or:

$$\frac{di_L}{dt} = \frac{e_c}{L} - \frac{R_i}{L} i_L \quad (8.6)$$

which yields the second equation.

We will designate the capacitor voltage (e_c , equation 8.5) by x_1 and the inductor current (i_L , equation 8.6) by x_2 . We also note that the input, $u(t)$ acts on the capacitor, C , therefore we can rewrite the state space form as:

$$\begin{bmatrix} \frac{dx_1}{dt} \\ \frac{dx_2}{dt} \end{bmatrix} = \begin{bmatrix} 0 & -\frac{1}{C} \\ \frac{1}{L} & -\frac{R}{L} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} \frac{1}{C} \\ 0 \end{bmatrix} [u]$$

Pharmokinetic Model

Consider the pharmokinetic system shown in Figure 8.5.

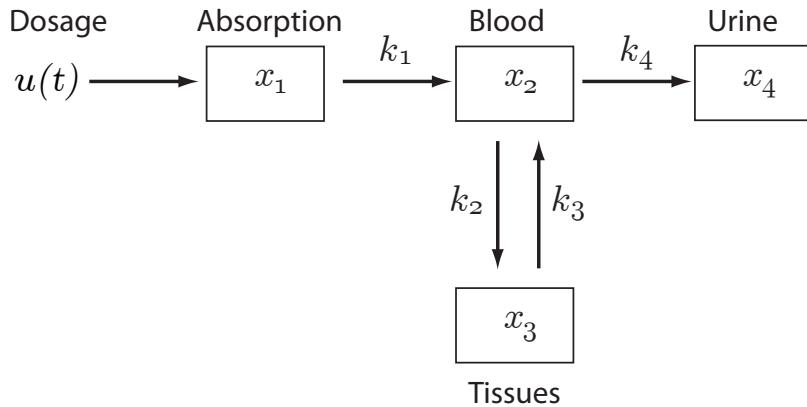


Figure 8.5 Pharmokinetic Model

We will assume first-order kinetics for each of the processes. The dosage will be assumed to be a step response so that:

$$\frac{dx_1}{dt} = u(t) - k_1 x_1$$

The remaining equations can be written in a similar fashion:

$$\frac{dx_2}{dt} = k_1x_1 - k_2x_2 - k_4x_2 + k_3x_3$$

$$\frac{dx_3}{dt} = k_2x_2 - k_3x_3$$

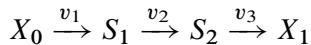
$$\frac{dx_4}{dt} = k_4x_2$$

We can write the model in state space form as:

$$\begin{bmatrix} \frac{dx_1}{dt} \\ \frac{dx_2}{dt} \\ \frac{dx_3}{dt} \\ \frac{dx_4}{dt} \end{bmatrix} = \begin{bmatrix} -k_1 & 0 & 0 & 0 \\ k_1 & -(k_2 + k_4) & k_3 & 0 \\ 0 & k_2 & -k_3 & 0 \\ 0 & k_4 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} [u]$$

Biochemical Model

Consider the simple biochemical pathway shown below:



where we assume that X_o and X_1 are fixed species. Assume that the rate laws are given by:

$$v_1 = E_1(X_o k_1 - k_2 S_1)$$

$$v_2 = E_2(k_3 S_1 - k_4 S_2)$$

$$v_3 = E_3 k_5 S_2$$

where E_i is the concentration of enzyme i in reaction i . We will build the state equations around the steady state operating point. We could have used nonlinear Michaelis-Menten rate laws but we can make the algebra simpler by assuming simple linear rate laws. We will cover non-linear rate laws in a later chapter. The differential equations for S_1 , and S_2 are:

$$\frac{dS_1}{dt} = v_1 - v_2$$

$$\frac{dS_2}{dt} = v_2 - v_3$$

Writing out the differential equations fully we have:

$$\frac{dS_1}{dt} = E_1(X_o k_1 - k_2 S_1) - E_2(k_3 S_1 - k_4 S_2)$$

$$\frac{dS_2}{dt} = E_2(k_3 S_1 - k_4 S_2) - E_3 k_5 S_2$$

We will consider three parameters that serve as inputs, E_1 , E_2 and E_3 . As discussed before the easiest way to construct the A and B matrices is to differentiate the differential equations with respect to the state variables for the A matrix and the inputs for the B matrix. For these equations, A can then be derived as:

$$A = \begin{bmatrix} -(E_1 k_2 + E_2 k_3) & E_2 k_4 \\ E_2 k_3 & -(E_2 k_4 + E_3 k_5) \end{bmatrix}$$

B is derived by assuming E_1 , E_2 and E_3 are the inputs which leads to a 2 by 3 matrix:

$$B = \begin{bmatrix} k_1 X_o - k_2 S_1 & -(k_3 S_1 + k_4 S_2) & 0 \\ 0 & k_3 S_1 - k_4 S_2 & -k_5 S_2 \end{bmatrix}$$

The \mathbf{u}_o vector is $[E_1 \ E_2 \ E_3]^T$. Fully written out the state space equation looks like:

$$\begin{bmatrix} \frac{dS_1}{dt} \\ \frac{dS_2}{dt} \end{bmatrix} = \begin{bmatrix} -(E_1 k_2 + E_2 k_3) & E_2 k_4 \\ E_2 k_3 & -(E_2 k_4 + E_3 k_5) \end{bmatrix} \begin{bmatrix} S_1 \\ S_2 \end{bmatrix}$$

$$+ \begin{bmatrix} k_1 X_o - k_2 S_1 & -(k_3 S_1 + k_4 S_2) & 0 \\ 0 & k_3 S_1 - k_4 S_2 & -k_5 S_2 \end{bmatrix} \begin{bmatrix} E_1 \\ E_2 \\ E_3 \end{bmatrix}$$

There is one important point to bear in mind about the above equation. Even though matrix B appears to be a function of S_1 and S_2 , this is not strictly true. The values for S_1 and S_2 must be taken from the operating point (steady state). As a result, the B matrix contains only constant values. Likewise, even though the A matrix contains input terms, these are assigned the nominal values at steady state, therefore A also only contains constants.

Exercises

1. Consider the following two species biochemical pathway:

$$\xrightarrow{v_o} x_1 \xrightarrow{k_1 x_1} x_2 \xrightarrow{k_2 x_2}$$

with differential equations:

$$\frac{dx_1}{dt} = v_o - k_1 x_1$$

$$\frac{dx_2}{dt} = k_1 x_1 - k_2 x_2$$

Write out the state space representation of this model. You can assume that the term, v_o , is the input to the system and that the operating point is the steady state. Assume the C matrix is the identity matrix and the D matrix is zero.

2. Write out the state space model for the gene cascade show in Figure 8.6. Assume the input to the model is X_o and that the state variables are x_1 and x_2 . Assume the C matrix is the identity matrix and the D matrix is zero. The operating point for the system will be the steady state. The individual rate laws are given by:

$$v_1 = V_{m_1} \frac{X_o}{K_{m_1} + X_o}$$

$$v_2 = k_1 x_1$$

$$v_3 = V_{m_2} \frac{x_1}{K_{m_2} + x_1}$$

$$v_4 = k_2 x_2$$

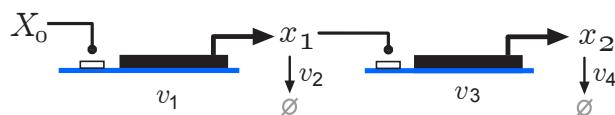


Figure 8.6 Two genes in a cascade formation.

9

Response of Linear Systems

"A child of five could understand this. Send someone to fetch a child of five."

– Groucho Marx

9.1 Solutions to Linear Systems

In the last chapter we looked at the state space representation. In this chapter we are going to look at solutions to the state space equations. Recall that a linearized differential equation can be represented using the state space representation (in non-vector form) as:

$$\frac{dx}{dt} = Ax(t) + Bu(t) \quad (9.1)$$

So long as the terms A and B are constant and not time dependent, this is a linear time invariant system. In text books on ordinary differential equations, a linear differential equation is usually written in the form:

$$\frac{dx}{dt} + P(t)x = Q(t)$$

If we remove the time dependence from P and replace it with $-A$ and replace $Q(t)$ with $Bu(t)$ we obtain:

$$\frac{dx}{dt} - Ax = Bu(t)$$

where $u(t)$ is a time dependent input. This equation can be rearranged to equal equation (9.1). That is, in its time invariant form (where $P(t)$ is replaced by P), the linear differential equation is a linear

time invariant system. When $B = 0$, the linear system is called a homogeneous system. Otherwise, it is called a non-homogeneous system. In matrix form they are written in the following form:

Homogeneous linear system:

$$\frac{d\mathbf{x}}{dt} = A\mathbf{x}(t)$$

Non-Homogeneous linear system:

$$\frac{d\mathbf{x}}{dt} = A\mathbf{x}(t) + B\mathbf{u}(t)$$

Example 9.1

Convert the following homogeneous system into matrix form:

$$\begin{aligned}\frac{dx_1}{dt} &= -2x_1 + 3x_2 \\ \frac{dx_2}{dt} &= 4x_1 - 6x_2\end{aligned}$$

We can write these equations in matrix form as:

$$\frac{d\mathbf{x}}{dt} = \begin{bmatrix} -2 & 3 \\ 4 & -6 \end{bmatrix} \mathbf{x}(t)$$

Example 9.2

Convert the following nonhomogeneous system into matrix form:

$$\begin{aligned}\frac{dx_1}{dt} &= -2x_1 + 3x_2 + 5u \\ \frac{dx_2}{dt} &= 4x_1 - 6x_2 - 15u\end{aligned}$$

We can write these equations in matrix form as:

$$\frac{d\mathbf{x}}{dt} = \begin{bmatrix} -2 & 3 \\ 4 & -6 \end{bmatrix} \mathbf{x}(t) + \begin{bmatrix} -5 \\ -15 \end{bmatrix} u$$

The solutions to linear differential equations are well known. For example, the solution to the following system, where X_o is fixed:

$$X_o \xrightarrow{k_1 X_o} x \xrightarrow{k_2 x}$$

with linear differential equation:

$$\frac{dx}{dt} = k_1 X_o - k_2 x$$

is given by:

$$x(t) = x(0)e^{-k_2 t} + \frac{X_o k_1}{k_2} (1 - e^{-k_2 t}) \quad (9.2)$$

Solution (9.2) is called the **total or complete solution** because it is made up of components involving the initial condition, $x(0)$, and a component involving the input, X_o , to the system. We can summarize some generalities from equation (9.2) that apply to all LTI systems. The first thing to note is that the equation is in two parts. The first part is given by:

$$x(0)e^{-k_2 t}$$

This is called the **zero-input response** because it doesn't include the input to the system ($X_o k_1$), and assumes that the initial conditions are not necessarily zero, $x(0)$. It is the response of the system in the absence of any change in input. Figure 9.1 shows the zero-input response for a positive initial condition. The natural response is to decay to zero in the absence of any input.

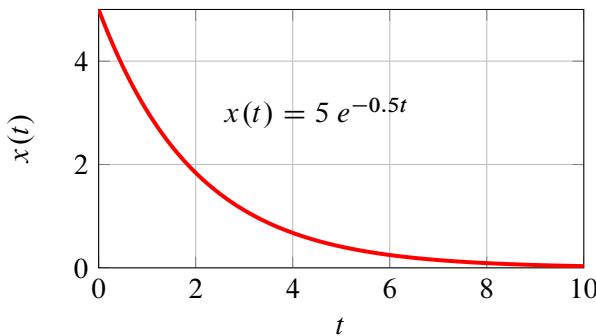


Figure 9.1 Zero-Input Response: $x(0) = 5, k_2 = 0.5$.

The second part of the total response is given by:

$$\frac{X_o k_1}{k_2} (1 - e^{-k_2 t})$$

This part of the equation is called the **zero-state response**. In this case the equation assumes that the initial condition, $x(0)$, is zero, but with an input step function with respect to X_o . Figure 9.2 shows an example of a zero-state response.

The zero-state response equation can also be divided into two parts:

$$\frac{X_o k_1}{k_2} - \frac{X_o k_1}{k_2} e^{-k_2 t}$$

These two parts refer to the steady state and transient responses. Thus the first term, $X_o k_1 / k_2$, corresponds to the steady state solution that is obtained at infinite time. We can see this if we take $t \rightarrow \infty$ where the second part, $(X_o k_1 / k_2) e^{-k_2 t}$ tends to zero.

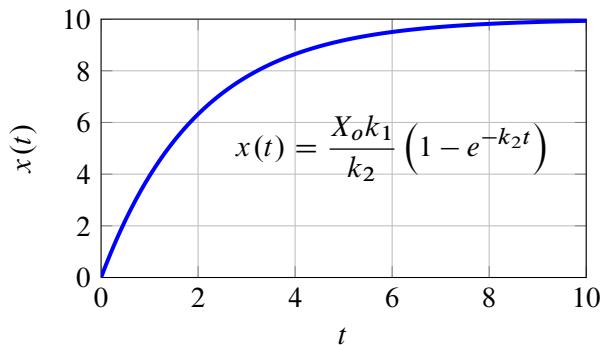


Figure 9.2 Zero-State Response: $X_o = 5; k_1 = 1, k_2 = 0.5$.

When we combine both the zero-input and zero-state parts of the equation, we obtain the *total response* shown in Figure 9.3.

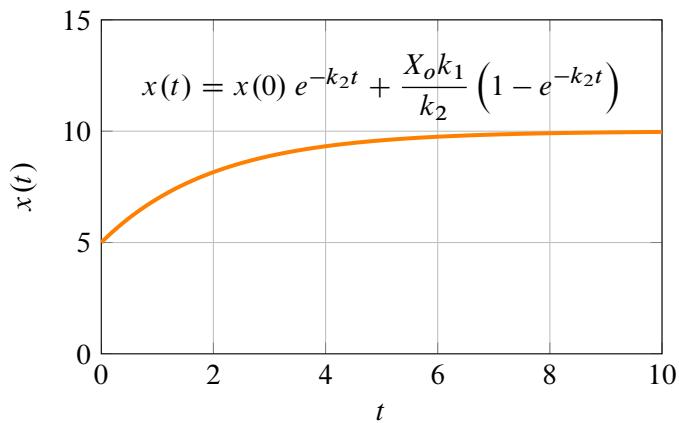


Figure 9.3 Total Response: $x(0) = 5, X_o = 5, k_1 = 1, k_2 = 0.5$.

$$x(t) = \underbrace{x(0)e^{-k_2 t}}_{\text{Zero-Input Response}} + \underbrace{\frac{X_o k_1}{k_2}}_{\text{Steady State}} - \underbrace{\frac{X_o k_1}{k_2} e^{-k_2 t}}_{\underbrace{\text{Forced Transient}}_{\text{Zero-State Response}}} \quad (9.3)$$

Equation (9.3) summarizes the various parts in the full response equation. That is:

$$\text{Total Response} = \text{Zero-Input Response} + \text{Zero-State Response}$$

The reason we can split the total response into the various behavioral modes is because of the **principle of superposition**.

Zero-input response:

1. Input is zero
2. Initial conditions are non-zero

Zero-state response:

1. Input is non-zero
2. Initial conditions are zero.

In other disciplines such as applied mathematics, there is a slightly different emphasis on how the total response is divided up. Without going into details here, the general solution for a homogeneous equation is:

$$x(t) = c_1 e^{\lambda_1 t} + c_2 e^{\lambda_2 t} + c_3 e^{\lambda_3 t} + \dots$$

The individual exponential terms in the solution are called the **characteristic modes** and the individual λ terms the characteristic roots. The zero-input response is made of a linear combination of the characteristic modes, the particular combination depending on the initial conditions. In equation (9.2) the first term is $x(0)e^{-k_2 t}$ and contains the mode: $e^{-k_2 t}$. Note however that we can also see the same mode in the zero-state response, that is:

$$\frac{X_o k_1}{k_2} (1 - e^{-k_2 t})$$

In differential equation theory, common modes across the zero-state and zero-input responses are grouped together to form the **natural response**. What is left in the zero-state response after these modes have been removed is called the **forced response**. For example, if a system is found to have a total response of:

$$x(t) = \underbrace{(-4e^{-t} + 2e^{-2t})}_{\text{zero-input response}} + \underbrace{(-6e^{-t} + 10e^{-2t} - 25e^{-3t})}_{\text{zero-state response}}$$

We see there are two modes, e^{-t} and e^{-2t} in the zero-input response, but we can also see the same modes in the zero-state response. We can combine these common modes to form the natural response. Now remove them from the zero-state response to yield the forced response:

$$x(t) = \underbrace{(-10e^{-t} + 12e^{-2t})}_{\text{natural response}} + \underbrace{(-25e^{-3t})}_{\text{forced response}}$$

The separation according to zero-input and zero-output is more useful for engineering applications than the separation using natural and forced responses. In addition, the fact that the zero-state response shares some modes with the zero-input response tells us that any input to a system also operates by exciting the natural modes during the input transient. The degree to which the natural modes are excited will be different compared to the same modes in the zero-input response.

From now on we will only consider the separation of the total response into zero-input and zero-state terms. The reason to bring up the topic of natural and forced responses is that there is common confusion between the two ways of dividing up the total response¹.

9.2 Relationship to Linearized Equations

When a system is linearized there is a subtle but very important change to how we interpret the zero-input and zero-state solutions. When a system is linearized all changes are relative to the steady state operating point around which the linearization was made. This is reflected in the fact the LTI for a linearized system measures changes in a disturbance (Figure 9.4) rather than the absolute change in a state variable. The linearized the LTI system becomes:

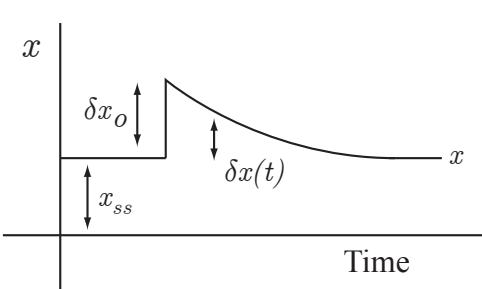
$$\frac{d\delta\mathbf{x}}{dt} \approx \mathbf{A}\delta\mathbf{x} + \mathbf{B}\delta\mathbf{u}$$

This means that for a zero-state response, the state variables are at the nominal values at the steady state but are not perturbed from that state. Instead the inputs are perturbed by an amount δu relative to the input values at steady state.

For the zero-input response it means that the inputs are at their nominal values at steady state and are not perturbed from that state. Instead the values of the state variables are perturbed by an amount δx_o relative to the values for the state variables at steady state.

¹For example the Wikipedia page (date: Aug, 2015) on zero state response incorrectly defines the forced response.

a) Zero-Input Response



b) Zero-State Response

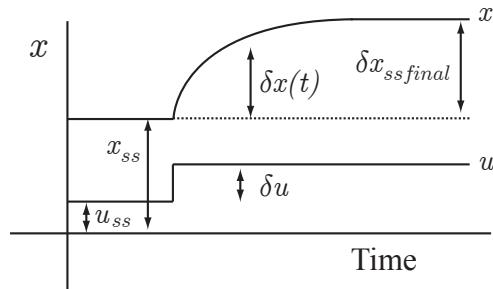


Figure 9.4 Perturbations on a linearized system. δx_o represents the initial perturbation in the state variable x around the steady state x_{ss} . δu represents the perturbation in the input u around the value of u that sustains the steady state which results in a change to the steady state value of x . u_{ss} represents the value of the input that sustains nominal steady state.

9.3 General Solutions

Homogeneous Solutions

In the previous section it was found that the solution to a first-order system could be split into zero-input and zero-state parts. The zero-input solution corresponds to no input, and non-zero initial conditions and reflects the relaxation properties of the system without any inputs. Here we will generalize these results to systems of linear differential equations.

In terms of the state space representation, the zero-input solution refers to the solution of the following homogeneous system:

$$\frac{d\mathbf{x}}{dt} = A\mathbf{x}$$

To solve this equation we first recall that the exponential function, e^{at} , can be written using the well known series:

$$e^{at} = 1 + ta + \frac{t^2 a^2}{2!} + \frac{t^3 a^3}{3!} + \dots \quad (9.4)$$

For the simple differential equation:

$$\frac{dx}{dt} = ax$$

the solution is known to be $x(t) = e^{at}x(0)$. Using the exponential series we can write this solution in the form:

$$x(t) = \left(1 + ta + \frac{t^2 a^2}{2!} + \frac{t^3 a^3}{3!} + \dots\right)x(0) \quad (9.5)$$

By analogy, we might propose that for the system of equations:

$$\frac{d\mathbf{x}}{dt} = \mathbf{A}\mathbf{x} \quad (9.6)$$

the solution might look like:

$$\mathbf{x}(t) = \left(\mathbf{I} + \mathbf{A}t + \frac{\mathbf{A}^2 t^2}{2!} + \frac{\mathbf{A}^3 t^3}{3!} + \dots\right)\mathbf{x}(0) \quad (9.7)$$

We can check if this is the correct solution by differentiating the proposed solution with respect to t , and seeing if we recover the differential equation (9.6). Differentiating equation (9.7) yields:

$$\begin{aligned} \frac{d\mathbf{x}}{dt} &= \left(\mathbf{0} + \mathbf{A} + \mathbf{A}^2 t + \frac{\mathbf{A}^3 t^2}{2!} + \dots\right)\mathbf{x}(0) \\ &= \mathbf{A} \left(\mathbf{I} + \mathbf{A}t + \frac{\mathbf{A}^2 t^2}{2!} + \dots\right)\mathbf{x}(0) \\ &= \mathbf{A}\mathbf{x}(t) \end{aligned}$$

That is, equation (9.7) is the solution to $d\mathbf{x}/dt = \mathbf{A}\mathbf{x}$. By analogy, the matrix series in equation (9.7) is referred to as the **matrix exponential** and is represented by the curious expression:

$$e^{\mathbf{A}t} = \mathbf{I} + \mathbf{A}t + \frac{1}{2!}\mathbf{A}^2 t^2 + \frac{1}{3!}\mathbf{A}^3 t^3 + \dots = \sum_{k=0}^{\infty} \frac{1}{k!} \mathbf{A}^k \quad (9.8)$$

Differentiating this definition of the matrix exponential with respect to t leads to the following result:

$$\frac{d}{dt} e^{\mathbf{A}t} = \mathbf{A} e^{\mathbf{A}t} = e^{\mathbf{A}t} \mathbf{A} \quad (9.9)$$

Although it may seem odd to raise e to the power of a matrix, the term, $e^{\mathbf{A}t}$ has properties very similar to a usual power. Given the definition of the matrix exponential, we can write the solution to $d\mathbf{x}/dt = \mathbf{A}\mathbf{x}$, equation (9.7), in a compact way, namely:

$$\mathbf{x}(t) = e^{\mathbf{A}t} \mathbf{x}_0 \quad (9.10)$$

To simplify the notation, the exponential matrix is also referred to as the **state transition matrix**, $\phi(t)$ so that the solution is sometimes written as:

$$\mathbf{x}(t) = \phi(t) \mathbf{x}_o \quad (9.11)$$

The matrix exponential has some useful properties given in the table below.

Properties of the Matrix Exponential

1. $e^{\mathbf{0}} = \mathbf{I}$ (This is equivalent to stating that $\phi(0) = \mathbf{I}$)
2. $e^{A t_1} e^{A t_2} = e^{A(t_1 + t_2)}$
3. $e^{A t} e^{-A t} = \mathbf{I}$
4. $(e^{A t})^{-1} = e^{-A t}$
5. $\frac{d}{dt} e^{A t} = A e^{A t} = e^{A t} A$

Example 1

Find the solution to the following system of differential equations assuming initial conditions: $x_1(0) = 2, x_2(0) = 3$:

$$\frac{dx_1}{dt} = -2x_1$$

$$\frac{dx_2}{dt} = x_1 - x_2$$

From the equations we determine that the A matrix is:

$$\begin{bmatrix} -2 & 0 \\ 1 & -1 \end{bmatrix}$$

Using equation (9.7) we can write out the matrix exponential:

$$\begin{aligned}
 e^{\mathbf{A}t} &= \left(\mathbf{I} + \mathbf{A}t + \frac{\mathbf{A}^2 t^2}{2!} + \frac{\mathbf{A}^3 t^3}{3!} + \dots \right) \\
 &= \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} + \begin{bmatrix} -2 & 0 \\ 1 & -1 \end{bmatrix} t + \begin{bmatrix} 4 & 0 \\ -3 & 1 \end{bmatrix} \frac{t^2}{2!} + \begin{bmatrix} -8 & 0 \\ 7 & -1 \end{bmatrix} \frac{t^3}{3!} + \dots \\
 &= \begin{bmatrix} 1 - 2t + \frac{4t^2}{2!} - \frac{8t^3}{3!} + \dots & 0 \\ 0 + t - \frac{3t^2}{2!} + \frac{7t^3}{3!} + \dots & 1 - t + \frac{t^2}{2!} - \frac{t^3}{3!} + \dots \end{bmatrix} \tag{9.12}
 \end{aligned}$$

The solutions, $x_1(t)$ and $x_2(t)$ are obtained by multiplying the matrix by the vector of initial conditions, [2, 3]. The elements in the final matrix can be seen as series representation for exponentials (9.4). In particular, the first series in the matrix represents $2e^{-2t}$, and the forth element, e^{-t} . The third element is a bit more difficult to spot but is the sum $e^{-t} - e^{-2t}$. Therefore, the solution to the original set of differential equations that incorporates the given initial conditions is:

$$\begin{aligned}
 x_1(t) &= 2e^{-2t} \\
 x_2(t) &= 2(e^{-t} - e^{-2t}) + 3e^{-t} \\
 &= 5e^{-t} - 2e^{-2t}
 \end{aligned} \tag{9.13}$$

The approach used in this example is not straightforward because it involves spotting the appropriate patterns in the series solution. It is possible to use tools such as Mathematica to compute the matrix, but there are better ways to solve for the solutions including the use of Laplace transforms which we will visit in Chapter 10 and the use of eigenvalues and eigenvectors described later in this chapter.

Sometimes the series in equation (9.7) will admit solutions that look like:

$$t - \frac{t^3}{3!} + \frac{t^5}{5!} - \dots$$

For those familiar with trigonometric series, this series will be recognized as representing $\sin(t)$. In other words, the solutions can admit periodic behavior. In terms of the generalized exponential solution, this corresponds to complex λ terms in the exponents.

Example 1 shows the solution to a simple set of linear differential equations without any input terms. In state space these solutions represent the zero-input response. The solution to the equations was a weighted (v_{jk}) sum of exponentials ($e^{\lambda_k t}$) which can be generalized to:

$$x_j(t) = \sum_{k=1}^n v_{jk} e^{\lambda_k t} \tag{9.14}$$

Non-Homogeneous Solutions

Let us now consider the solution to the full state space system:

$$\frac{dx}{dt} = Ax(t) + Bu(t)$$

First, we rewrite the above equation as:

$$\frac{dx}{dt} - Ax(t) = Bu(t)$$

and multiplying both sides by e^{-At} yields:

$$e^{-At} \frac{dx}{dt} - e^{-At} Ax(t) = e^{-At} Bu(t) \quad (9.15)$$

A known result in matrix algebra is the following relationship:

$$\frac{d(PQ)}{dt} = \frac{dP}{dt}Q + \frac{dQ}{dt}P$$

From this we can show that:

$$\frac{d(e^{-At}x)}{dt} = \frac{d(e^{-At})}{dt}x + e^{-At} \frac{dx}{dt}$$

From equation (9.9) we can simplify the above to:

$$\frac{d(e^{-At}x)}{dt} = -e^{-At}Ax + e^{-At} \frac{dx}{dt}$$

The right-hand expression is the left-hand expression in equation (9.15), therefore (and to avoid confusion we use τ as the independent variable time):

$$\frac{d(e^{-At}x)}{dt} = e^{-At}Bu(t)$$

Integrating both sides from t_0 to t gives:

$$e^{-At}x \Big|_{t_0}^t = \int_{t_0}^t e^{-A\tau}Bu(\tau)d\tau$$

that is:

$$e^{-At}x(t) - e^{-At_0}x(t_0) = \int_{t_0}^t e^{-A\tau}Bu(\tau)d\tau$$

Premultiply by $e^{\mathbf{A}t}$ yields:

$$\mathbf{x}(t) - e^{\mathbf{A}(t-t_0)}\mathbf{x}(t_0) = e^{\mathbf{A}t} \int_{t_0}^t e^{-\mathbf{A}\tau} \mathbf{B}\mathbf{u}(\tau) d\tau$$

or

$$\underbrace{\mathbf{x}(t)}_{\text{Total Response}} = \underbrace{e^{\mathbf{A}(t-t_0)}\mathbf{x}(t_0)}_{\text{Zero-Input Response}} + \underbrace{\int_{t_0}^t e^{\mathbf{A}(t-\tau)} \mathbf{B}\mathbf{u}(\tau) d\tau}_{\text{Zero-State Response}} \quad (9.16)$$

We can split the solution into the zero-input and zero-state responses. The zero-input response depends only on the initial conditions with the inputs set to zero. The zero-state response depends on the inputs but where the initial conditions are zero. It is worth noting that time (t) on the right-hand side is relative, that is the equation involves $(t - t_0)$ and $(t - \tau)$ terms, and is a result of time invariance. In other words, the actual start time where the solution is evaluated does not affect the qualitative form of the solution curves. We could set $t_0 = 0$ and in fact many textbooks do assume this, so that the solution can be written as:

$$\mathbf{x}(t) = e^{\mathbf{A}t}\mathbf{x}(0) + \int_0^t e^{\mathbf{A}(t-\tau)} \mathbf{B}\mathbf{u}(\tau) d\tau$$

It is possible to use this approach to solve non-homogeneous systems, however in practice it is rarely used due to the potential difficulties in deriving the integral for the zero-state response. As we will see in the next chapter, the use of Laplace transforms greatly simplifies the entire procedure for solving linear non-homogeneous differential equation.²

9.4 Relation to Eigenvalues and Eigenvectors

The question we wish to ask here is where do the exponents in the general solution (9.14) comes from?

See Appendix F.7 for a reminder on eigenvalues and eigenvectors. Given the homogeneous equation:

$$\frac{d\mathbf{x}}{dt} = \mathbf{A}\mathbf{x} \quad (9.17)$$

²One could even argue that with the advent of algebraic software tools like Mathematica, even the use of Laplace transforms may be redundant. Note however that pen and pencil is much more cost effective and intellectually more satisfying and educational than the use of expensive software tools.

let us write the matrix \mathbf{A} in the form of an eigenvalue problem:

$$\mathbf{A}\mathbf{v} = \lambda\mathbf{v}$$

Rearranging and multiplying both sides by $e^{\lambda t}$ yields:

$$\lambda\mathbf{v}e^{\lambda t} = \mathbf{A}\mathbf{v}e^{\lambda t} \quad (9.18)$$

By comparison with the solution to a single differential equation, let us suppose that equation (9.17) has a solution of the form

$$\mathbf{x}(t) = \mathbf{v}e^{\lambda t} \quad (9.19)$$

Substituting the term $\mathbf{v}e^{\lambda t}$ on the right side of equation (9.18) with $x(t)$ from equation (9.19) we obtain:

$$\lambda\mathbf{v}e^{\lambda t} = \mathbf{A}\mathbf{x}(t) \quad (9.20)$$

Differentiating equation (9.19) yields:

$$\frac{d\mathbf{x}}{dt} = \lambda\mathbf{v}e^{\lambda t}$$

However this is the same as the term in equation 9.20 therefore:

$$\frac{d\mathbf{x}}{dt} = \mathbf{A}\mathbf{x}(t)$$

which is the original homogeneous equation. Therefore not only is $\mathbf{x}(t) = \mathbf{v}e^{\lambda t}$ the solution to the homogeneous equation, but we have also reduced the problem to a basic eigenvalue/eigenvector problem. The solution $\mathbf{x}(t) = \mathbf{v}e^{\lambda t}$ corresponds to the case when the initial condition, $x(0)$ is equal to one of the eigenvectors of \mathbf{A} . If we assume for the moment that all the eigenvalues, λ_i of \mathbf{A} are distinct then there will be the equivalent number of linearly independent eigenvectors, \mathbf{v}_i . This means there will be a number of independent solutions to a homogeneous equation corresponding to each eigenvalue, eigenvector pair. That is the solutions will be:

$$\mathbf{x}_1(t) = \mathbf{v}_1 e^{\lambda_1 t}$$

$$\mathbf{x}_2(t) = \mathbf{v}_2 e^{\lambda_2 t}$$

⋮

$$\mathbf{x}_n(t) = \mathbf{v}_n e^{\lambda_n t}$$

Given that the system is linear and the eigenvectors linearly independent, there will be a general solution that is made up of a weighted sum of the individual solutions:

$$\mathbf{x}(t) = c_1 \mathbf{v}_1 e^{\lambda_1 t} + c_2 \mathbf{v}_2 e^{\lambda_2 t} + \dots + c_n \mathbf{v}_n e^{\lambda_n t} = \sum_{i=1}^n \beta_i e^{\lambda_i t} \mathbf{v}_i$$

The weighting factors, c_i , are determined by the initial conditions. This is illustrated in the next example.

Example 9.3

Solve the following system of homogenous linear differential equations:

$$\frac{d\mathbf{x}}{dt} = \begin{bmatrix} 1 & 2 \\ 3 & 2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

given initial conditions:

$$\begin{bmatrix} 0 \\ -2 \end{bmatrix}$$

We first compute the eigenvalues for the \mathbf{A} matrix:

$$\det(\mathbf{A} - \lambda \mathbf{I}) = \lambda^2 - 3\lambda - 4 = (\lambda + 1)(\lambda - 4)$$

$$\lambda_1 = -1, \quad \lambda_2 = 4$$

For each eigenvalue we compute the corresponding eigenvector using the relation $(\mathbf{A} - \lambda \mathbf{I})\mathbf{v} = 0$. For example for $\lambda_1 = -1$ we must solve:

$$\begin{bmatrix} 2 & 2 \\ 3 & 3 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

Solving for v_1 and v_2 yields, $v_1 = -1$ and $v_2 = 1$. We can do a similar evaluation for the second eigenvalue, λ_2 that gives: $v_1 = 2$ and $v_2 = 3$. We can now combine the eigenvalues and eigenvectors into the general solution (9.19):

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = c_1 e^{-t} \begin{bmatrix} -1 \\ 1 \end{bmatrix} + c_2 e^{4t} \begin{bmatrix} 2 \\ 3 \end{bmatrix}$$

The c_i terms can be determined from the initial conditions. We can do this by setting $t = 0$ to obtain (note that at $t = 0$ we know the values of x_1 and x_2 from the initial conditions):

$$\begin{bmatrix} 0 \\ -2 \end{bmatrix} = c_1 \begin{bmatrix} -1 \\ 1 \end{bmatrix} + c_2 \begin{bmatrix} 2 \\ 3 \end{bmatrix}$$

Solving these for c_1 and c_2 yields: $c_1 = -4/5$, $c_2 = -2/5$. The final solution is therefore:

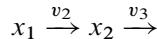
$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = -\frac{4}{5} e^{-t} \begin{bmatrix} -1 \\ 1 \end{bmatrix} - \frac{2}{5} e^{4t} \begin{bmatrix} 2 \\ 3 \end{bmatrix}$$

Although solving a homogenous system this way seems straightforward, edge cases can arise if the system admits repeated eigenvalues or complex conjugates. We will not cover these here. The point of this section is to show that the exponents in the solution are the eigenvalues of the \mathbf{A} matrix. Since the exponents determine the decay and/or growth of the solutions, an inspection of the eigenvalues can tell us something about the future behavior of the system.

Rather than compute the exponential matrix which is tedious, we can instead determine the eigenvalues from the Jacobian. The following example illustrates the solution to a system of homogeneous equations using the eigenvalue/eigenvector method.

Example 9.4

Given the following sequence of reactions, solve the accompanying differential equations:



where we assume that each reaction, v_i , is first-order order and irreversible. The state variables are x_1 and x_2 . Initial conditions are given by:

$$x_1(0) = 2, \quad x_2(0) = 0$$

$$\frac{dx_1}{dt} = -k_1 x_1$$

$$\frac{dx_2}{dt} = k_1 x_1 - k_2 x_2$$

Expressed in matrix form:

$$\begin{bmatrix} \frac{dx_1}{dt} \\ \frac{dx_2}{dt} \end{bmatrix} = \begin{bmatrix} -k_1 & 0 \\ k_1 & -k_2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \mathbf{A}\mathbf{x}$$

We first determine the eigenvalues for the \mathbf{A} matrix.

$$\lambda_1 = -k_1, \quad \lambda_2 = -k_2$$

The eigenvectors, \mathbf{v} , are computed from the equation $(\mathbf{A} - \lambda \mathbf{I})\mathbf{v} = 0$ so that the corresponding eigenvectors are found to be:

$$\mathbf{v}_1 = \begin{bmatrix} \frac{k_2 - k_1}{k_1} \\ 1 \end{bmatrix} \quad \mathbf{v}_2 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

We can now write the solution without consideration of the initial conditions:

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = c_1 e^{-k_1 t} \begin{bmatrix} \frac{k_2 - k_1}{k_1} \\ 1 \end{bmatrix} + c_2 e^{-k_2 t} \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

To find the constants, c_1 and c_2 , we set $t = 0$ and insert the initial conditions. For example, for the first row that evaluates x_1 we have:

$$x_1 = c_1 e^{-k_1 t} \left(\frac{k_2 - k_1}{k_1} \right)$$

At $t = 0$, $x_1(0) = 2$, therefore:

$$2 = c_1 \left(\frac{k_2 - k_1}{k_1} \right)$$

$$c_1 = 2 \frac{k_1}{k_2 - k_1}$$

Inserting this into the solution yields:

$$x_1 = \left(2 \frac{k_1}{k_2 - k_1}\right) e^{-k_1 t} \left(\frac{k_2 - k_1}{k_1}\right) = 2e^{-k_1 t}$$

The same can be done to evaluate c_2 at $t = 0$ for x_2 where $x_2(0) = 0$:

$$0 = c_1 + c_2$$

Since we know c_1 we can derive c_2 . Therefore x_2 is given by:

$$x_2 = 2 \frac{k_1}{k_2 - k_1} e^{-k_1 t} - 2 \frac{k_1}{k_2 - k_1} e^{-k_2 t}$$

or:

$$x_2 = 2 \frac{k_1}{k_2 - k_1} (e^{-k_1 t} - e^{-k_2 t})$$

Exercises

- Convert the following homogeneous system into matrix form:

$$\frac{dx_1}{dt} = 6x_1 - x_2$$

$$\frac{dx_2}{dt} = -3x_1 + 5x_2$$

- Define the terms zero-input response, zero-state response and total response.
- Why is it possible to separate out the response of a linear system into the zero-input and zero-state responses?
Because the system is linear, independent changes result in a simple sum of responses.
- Prove equation (9.9).
- Given the following A matrix for the homogeneous equation: $dx/dt = Ax$, determine the solutions $x_1(t)$ and $x_2(t)$. Assume initial conditions: $x_1(0) = 1$, $x_2(t) = 0$

$$A = \begin{bmatrix} -7 & 4 \\ -6 & 3 \end{bmatrix}$$

6. Given the following system:

$$\begin{aligned}\frac{dx_1}{dt} &= x_1 \\ \frac{dx_2}{dt} &= x_1 - x_2\end{aligned}$$

with initial conditions, $x_1(0) = 1$ and $x_2(0) = 1$. Determine x_1 and x_2 in the form of a series derived from e^{At} (see equation (9.12)). Three terms will be sufficient when you give your answer.

Confirm that the elements of the exponential matrix are given by:

$$\begin{bmatrix} e^t & 0 \\ \frac{1}{2}e^{-t}(e^{2t}-1) & e^{-t} \end{bmatrix}$$

e^t is given by the standard series $1 + t + t^2/2! + \dots$

To find the others, eg $1 - t + t^2/2 + \dots$, expand e^{-t} using a Taylor series, this will confirm that $e^{-t} = 1 - t + t^2/2! + \dots$

For $t + t^3/6 \dots$ note that the expected answer is:

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

Expand each exponential term using the known series for $e^t = 1 + t + t^2/2! + \dots$ and $e^{-t} = 1 - t + t^2/2! + \dots$, and subtract one from the other. This will yield $t + \frac{t^3}{6}$ which confirms the expected result. You will have to use 4 terms in the expansion to confirm the result.

7. Given the following system:

$$\begin{aligned}\frac{dx_1}{dt} &= x_1 \\ \frac{dx_2}{dt} &= x_1 - x_2\end{aligned}$$

with initial conditions, $x_1(0) = 1$ and $x_2(0) = 1$. Determine the $x_1(t)$ and $x_2(t)$ solutions using the eigenvalue/eigenvector method. Also state the stability of the system.

10

Laplace Transforms

“Don’t Panic”

– Douglas Adams, The Hitchhiker’s Guide to the Galaxy

10.1 Introduction

As we saw from the last chapter, solving linear differential equations can be a little tedious, although with modern software such as Mathematica it can now be done relatively easily. However a classical approach to solving linear differential equations and which has considerable other advantages, is the **Laplace transform**. The technique was developed by the 19th century French mathematician and astronomer, Pierre Simon de Laplace. In this chapter we will describe the use of the Laplace transform in solving linear differential equations. The method is analogous to using logarithms to make multiplication easier. When using logarithms we take the log of the two numbers we wish to multiply, add the resulting values, then take the antilog to obtain the product. The argument is that simple addition is much easier compared to multiplication. The same strategy can be used to solve linear differential equations. In this case we take the Laplace transform of the differential equations which turns them into algebraic equations, we solve the algebraic equations for the dependent variables and then take the inverse Laplace transform to recover the solution.

The Laplace transform converts *integral* and *differential* equations into *algebraic* equations.

10.2 Definition

The unilateral Laplace transform is defined as follows¹ Let $f(t)$ be a function that is defined for all positive t . If we multiply $f(t)$ by e^{-st} and integrate with respect to t from zero to infinity and the resulting integral *converges* to a new function $F(s)$, then we call $F(s)$ the Laplace transform of $f(t)$:

$$\mathcal{L}[f(t)] = \int_0^{\infty} e^{-st} f(t) dt = F(s)$$

The symbol $\mathcal{L}[f(t)]$ is read as the Laplace transform of $f(t)$. As written above the Laplace transform is also referred to as the unilateral Laplace transform to distinguish it from the bilateral form where the integration goes from minus infinity to plus infinity. The variable s is a complex variable of the form $s = \sigma + j\omega$.

Notation:

By convention the Laplace transform of a function is often indicated with a capital letter.

For example, $F(s)$, $H(s)$ or $X(s)$

The original function is represented by lower case letters.

For example, $f(t)$, $h(t)$ or $x(t)$.

In some cases the Laplace transform of a variable $f(t)$ is also denoted by the symbol $\mathcal{L}[f(t)]$.

Inverse Transform

If we determine the Laplace transform for a function $f(t)$ to be $F(s)$ then we can also define the inverse transform of $F(s)$ to be $f(t)$. The **inverse transform** is often denoted by:

$$f(t) = \mathcal{L}^{-1}[F(s)]$$

Convergence

The Laplace transform is a definite integral computed between zero and infinity. This raises the question of convergence and whether the integral, given a particular $f(t)$, will converge to a value less than infinity? If, for a given function $f(t)$, the transform fails to converge then the Laplace transform for $f(t)$ does not exist. For example the functions, $1/t$ and t^{t^2} do not have a Laplace transform.

¹Chapter 12. will revisit the Laplace transform to provide a more initiative understanding of the transform.

There are two *sufficient* conditions for a function to have a Laplace transform. The first is exponential order. For convergence to occur $f(t)$ must increase at a slower rate than the rate of decay of e^{-st} . This means that at some point the value of e^{+st} must exceed $f(t)$. Note that the rate of decay of e^{-st} is determined by the real part, σ in s . For some functions such as t^t there is no value for σ than can decay fast enough to bound t^t . Therefore functions such as t^t have no Laplace transform. In contrast a function such as $t^2 + 1$ is of exponential order because at $t = 0$, e^{2t} begins to exceed $t^2 + 1$. A function that is exponential order is one where there exists a constant, C and a such that:

$$|f(t)| \leq Ce^{at} \quad t > T$$

that is $f(t)$ should always be less than Ce^{at} (Figure 10.1). For example e^{t^2} is not exponential order because $e^{t^2} > e^t$ for $t > 1$.

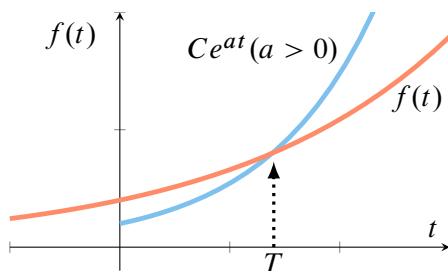


Figure 10.1 A function, $f(t)$, is exponential order because there are values for C and a where Ce^{at} exceeds $f(t)$ at a point T .

An equivalent way to look at convergence is that a function $f(t)$ has a Laplace transform if the product, $f(t)e^{-st}$ converges to zero:

$$\lim_{t \rightarrow \infty} f(t)e^{-st} = 0$$

A second condition that is satisfied for functions having a Laplace transform is if the function is a piecewise continuous function (Figure 10.3). A piecewise continuous function is one where a function can be described as a series of intervals where the function is continuous on each interval *and* at the end and start points of the intervals, the limits are finite. This is particularly useful in engineering where signals can be discontinuous, for example step inputs or waveforms such as a sawtooth. The reason why Laplace transforms can be found for piecewise continuous functions is that the transform can be written as a sum of integrals on each interval. For example given a piecewise function with two intervals, 0 to x and x to ∞ , so long as the end and start points for the intervals are finite the individual integrals will exist, the Laplace transform can therefore be written as:

$$\int_0^\infty e^{-st} f(t) dt = \int_0^x e^{-st} f(t) dt + \int_x^\infty e^{-st} f(t) dt$$

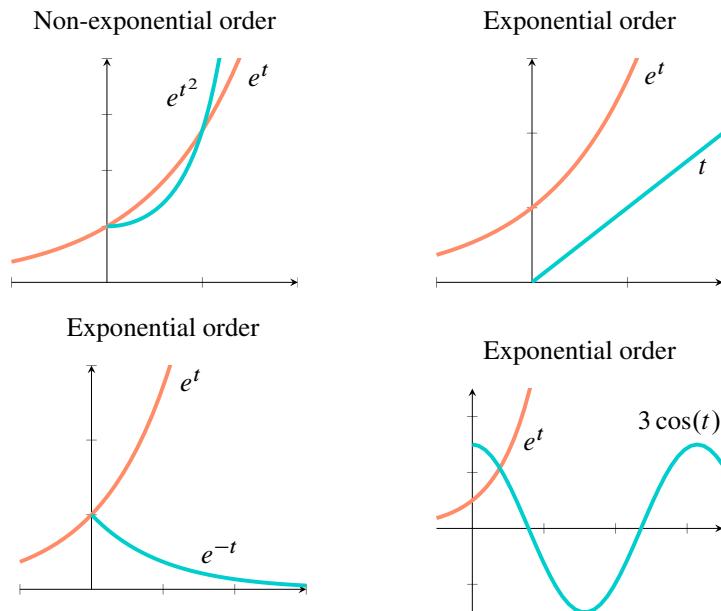


Figure 10.2 Exponential and non-exponential order functions. Adapted from Zill and Cullen (1986)

The two sufficient conditions, exponential order and piecewise continuous, ensure that, if true, a function will have a Laplace transform. The converse is not true however, that a function that has a Laplace transform means that the function is exponential order and piecewise²

Note these are sufficient, not necessary conditions, meaning that it is possible to find functions that may possess these conditions but still have a Laplace transform. For example $t^{1/2}$ is not piecewise at $t = 0$ because at this point it approaches infinity, but nevertheless has a Laplace Transform of $\sqrt{\pi}/s$. Fortunately when studying physical systems, we seldom encounter functions where the Laplace transform fails to converge.

10.3 Examples of Laplace Transforms

Variable

We will define the Laplace transform of the variable $x(t)$ as:

$$\mathcal{L}[x(t)] = X(s)$$

²A sufficient condition implies a truth but not necessarily the opposite. For example, ‘Albert is a bachelor’ implies that Albert is male. But someone being male does not imply they are a bachelor.

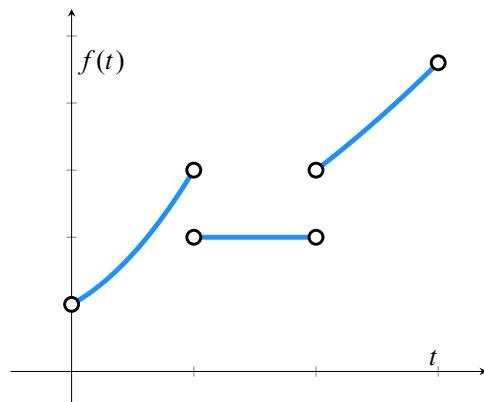


Figure 10.3 A piecewise continuous function, $f(t)$.

Unit Step Function

Let's look at a very simple function and compute its Laplace transform. The simplest function is $f(t) = 1$ for $t \geq 0$. That is at $t < 0$ $f(t) = 0$. This represents the **step function**, $u(t)$ (Figure 10.4) of magnitude one. The Laplace transform can be derived as follows (See Table 10.1):

$$F(s) = \int_0^{\infty} e^{-st} 1 \, dt = \left[-\frac{1}{s} e^{-st} \right]_{t=0}^{t=\infty} = \frac{1}{s}$$

$$\mathcal{L}[u(t)] = \frac{1}{s}$$

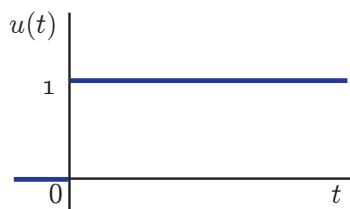


Figure 10.4 Step Function, $u(t)$

Ramp Function

Derive the Laplace transform for the Ramp function:

$$f(t) = at$$

with slope a . Substituting at into the Laplace transform:

$$\mathcal{L}[at] = \int_0^\infty e^{-st} at \, dt$$

Integrating by parts: $\int u dv = uv - \int v du$ where $du = dt$, $u = t$, $v = -1/s e^{-st}$ and $dv = e^{-st}$ will yield:

$$\begin{aligned}\mathcal{L}[at] &= a \int_0^\infty e^{-st} t \, dt \\ &= a \left[-\frac{te^{-st}}{s} \right]_0^\infty - \int_0^\infty \frac{e^{-st}}{-s} dt \\ &= a \left[-\frac{te^{-st}}{s} \right]_0^\infty - \left[-a \frac{e^{-st}}{s^2} \right]_0^\infty \\ &= a \left[\left(-\frac{\infty e^{-s \infty}}{s} \right) - \left(-\frac{0 e^{-s 0}}{s} \right) \right] - \left[-a \frac{e^{-st}}{s^2} \right]_0^\infty\end{aligned}$$

Let us consider the first term:

$$a \left[\left(-\frac{\infty e^{-s \infty}}{s} \right) - \left(-\frac{0 e^{-s 0}}{s} \right) \right]$$

The second part of this term is zero, the first part however is indeterminate because as one term (∞) goes to infinity the other goes to zero ($e^{-t} \infty$). To resolve the limit, we use L'Hospital's rule by first rearranging the expression to be a quotient. Considering only the problematic te^{-st} term:

$$\lim_{t \rightarrow \infty} te^{-st} = \lim_{t \rightarrow \infty} \frac{t}{e^{st}}$$

Applying L'Hospital's rule: $\lim f'(x)/g'(x)$

$$\lim_{t \rightarrow \infty} \frac{1}{se^{st}} = \lim_{t \rightarrow \infty} se^{-st} = 0$$

Therefore the first term is also zero in the limit. This leads us to:

$$\begin{aligned}\mathcal{L}[at] &= a(0 - 0) + \frac{a}{s^2} \\ &= \frac{a}{s^2}\end{aligned}$$

Exponential Function

Derive the Laplace transform of the exponential function:

$$f(t) = e^{-kt}$$

Substituting e^{-kt} into the Laplace transform:

$$\begin{aligned}\mathcal{L}[f(t)] &= \int_0^\infty e^{-kt} e^{-st} dt = \int_0^\infty e^{-(k+s)t} dt \\ \mathcal{L}[f(t)] &= \left[\frac{1}{s+k} e^{-(s+k)t} dt \right]_0^\infty\end{aligned}$$

and applying the integral limits we obtain:

$$\mathcal{L}[e^{-kt}] = \frac{1}{s+k}$$

Impulse Function

Derive the Laplace transform of the impulse function:

$$f(t) = \delta(t)$$

Inserting $\delta(t)$ into the Laplace function:

$$\mathcal{L}[f(t)] = \int_0^\infty \delta(t) e^{-st} dt$$

Because the impulse function only has a value at $t = 0$, the above equation becomes:

$$\mathcal{L}[f(t)] = \int_0^\infty \delta(t) 1 dt = \int_0^\infty \delta(t) dt = 1$$

That is:

$$\mathcal{L}[\delta(t)] = 1$$

For the shifted impulse, $\delta(t - a)$ is it possible to show that:

$$\mathcal{L}[\delta(t - a)] = e^{-as}$$

which clearly reverts to one when $a = 0$.

Derivative

Let us consider the Laplace transform for the derivative. In the context of solving differential equations this is probably the most important transform to look at. Consider the first order derivative:

$$f(t) = \frac{df}{dt}$$

The Laplace transform for the derivative is given by:

$$\mathcal{L}[f(t)] = \int_0^\infty e^{-st} \frac{df}{dt} dt$$

Using integration by parts: $\int u dv = uv - \int v du$ where:

$$u = e^{-st} \quad v = f(t) \quad du = -se^{-st} dt \quad dv = \frac{df}{dt} dt$$

gives:

$$\begin{aligned} \int_0^\infty e^{-st} \frac{df}{dt} dt &= [e^{-st} f(t)]_0^\infty - \int_0^\infty f(t)(-se^{-st} dt) \\ &= [e^{-st} f(t)]_0^\infty + s \int_0^\infty f(t)(e^{-st} dt) \\ &= [e^{-st} f(t)]_0^\infty + sF(s) \\ &= \left[\frac{f(\infty)}{e^\infty} - \frac{f(0)}{e^0} \right] + sF(s) \end{aligned}$$

That is:

$$\int_0^\infty e^{-st} \frac{df}{dt} dt = -f(0) + sF(s) = sF(s) - f(0)$$

$$\boxed{\mathcal{L}\left[\frac{df(t)}{dt}\right] = sF(s) - f(0)}$$

Integral

The final Laplace transform worth introducing is the transform of the integral. Without providing the proof (but which can be found in other standard texts), the Laplace transform of the integral is given by:

$$\mathcal{L} \left\{ \int_0^t f(\tau) d\tau \right\} = \frac{F(s)}{s}$$

Both the transform for the derivative and integral take on important roles in developing robust negative feedback designs which we will cover in a later chapter. It is worth pointing out that these transforms are often written simply in form as:

$$\text{Derivative: } s \quad \text{Integral: } \frac{1}{s}$$

Example 10.1

Find the Laplace transform for the following equation:

$$\begin{aligned} \frac{dx}{dt} &= -x \\ sX(s) - x(0) &= -X(s) \end{aligned}$$

Tables of Transforms

Most of the time there is no need to derive the transforms directly because there are published tables that list common functions and the corresponding Laplace transform (See Table 10.1 and Appendix H). In addition the Laplace transform is a linear operation which means there are many useful properties of the transform which can also greatly help when deriving the transform for a particular function.

10.4 Properties of the Laplace Transform

Linearity and Scaling

The Laplace transform is linear.

Scaling Consider a function $f(t)$ scaled by a factor a , that is $af(t)$. The Laplace transform of the scaled function is:

$$\mathcal{L}[af(t)] = a\mathcal{L}[f(t)]$$

Function: $f(t)$	Laplace Function: $\mathcal{L}[f(t)]$
$\delta(t)$	1
$au(t)$	$\frac{a}{s}$
$u(t - a)$	$\frac{1}{s}e^{-as}$
$tu(t)$	$\frac{1}{s^2}$
ae^{-kt}	$\frac{a}{s + k}$
t	$\frac{1}{s^2}$
t^n	$\frac{n!}{s^{n+1}}$
$\frac{a}{b}(1 - e^{-bt})$	$\frac{a}{s(s + b)}$
$\frac{df(t)}{dt}$	$sF(s) - f(0)$
$\sin(at)$	$\frac{a}{s^2 + a^2}$
$\cos(at)$	$\frac{s}{s^2 + b^2}$
e^{-at}	$\frac{1}{s + a}$
te^{-at}	$\frac{1}{(s + a)^2}$
$af_1(t) + bf_2(t)$	$aF_1(s) + bF_2(s)$

Table 10.1 Short Table of Laplace Transforms

For example, if:

$$\mathcal{L}\left[\frac{1}{b}(1 - e^{-bt})\right] = \frac{1}{s(s - b)}$$

then

$$\mathcal{L}\left[\frac{a}{b}(1 - e^{-bt})\right] = \frac{a}{s(s - b)}$$

Summation Rule If $f(t)$ and $g(t)$ are two functions then the Laplace transform of their sum $f(t) + g(t)$ is;

$$\mathcal{L}[f(t) + g(t)] = \mathcal{L}[f(t)] + \mathcal{L}[g(t)]$$

More generally

$$\mathcal{L}[af(t) + bg(t)] = a\mathcal{L}[f(t)] + b\mathcal{L}[g(t)]$$

Likewise we can also state the inverse:

$$\mathcal{L}^{-1}[aF(s) + bG(s)] = af(t) + bg(t)$$

Example 10.2

Find the Laplace transform of: $3t + 8t^2$

$$\mathcal{L}[3t + 8t^2] = 3\mathcal{L}[t] + 8\mathcal{L}[t^2]$$

If

$$F(s) = \frac{3}{s} - \frac{1}{s-2} + \frac{4}{s-3}$$

Then the inverse transform is:

$$\begin{aligned}\mathcal{L}^{-1}[F(s)] &= 3\mathcal{L}^{-1}\left[\frac{1}{s}\right] - \mathcal{L}^{-1}\left[\frac{1}{s-2}\right] + 4\mathcal{L}^{-1}\left[\frac{1}{s-3}\right] \\ &= 3 - e^{2t} - 4e^{3t}\end{aligned}$$

Change of Scale

If the time scale of the time domain t is changed, that is t is scaled by a , then the Laplace transform is modified as follows:

$$\mathcal{L}[f(at)] = \frac{1}{a}F\left(\frac{s}{a}\right) \quad (10.1)$$

The proof can be found in the chapter appendix. For example, find the Laplace transform of $\sin(3t)$. The transform of $\sin(t)$ is:

$$F(s) = \frac{1}{1+s^2}$$

We now substitute all occurrences of s with s/a and multiply by $1/a$. Hence the Laplace transform of $\sin(3t)$ is:

$$\mathcal{L}[\sin(3t)] = \frac{1}{3} \frac{1}{1+\left(\frac{s}{3}\right)^2} = \frac{3}{9+s^2}$$

First Time Shifting Theorem

The first time shifting theorem tells us that in the time domain, if we multiply a function by e^{at} then it results in a shift in the Laplace domain by a units.

$$\mathcal{L}[e^{-at} f(t)] = F(s+a)$$

Example 10.3

If:

$$f(t) = \cos(\omega t)$$

the Laplace transform is given by:

$$F(s) = \frac{s}{s^2 + \omega^2}$$

Now consider multiplying $f(t)$ by e^{-at} , that is:

$$f(t) = e^{-at} \cos(\omega t)$$

The transform of this equation can be determined by the first shifting theorem. To use it, we replace every occurrence of s with $s+a$, so that:

$$F(s) = \frac{s+a}{(s+a)^2 + \omega^2}$$

To illustrate the theorem with another example, find the Laplace transform of $e^{at}t^3$. First we find the transform of t^3 which is $6/s^4$. We then form $F(s-a)$ by replacing each s in $6/s^4$ with the value $(s-a)$ so that:

$$\mathcal{L}(e^{at}t^3) = \frac{6}{(s-a)^4}$$

Second Time Shifting Theorem

The second time shifting theorem can be used to find the transform of a shifted signal, Figure 10.5.

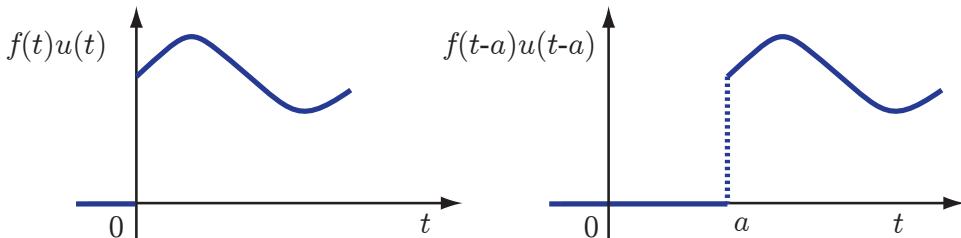


Figure 10.5 Shifting a function, $f(t)$.

The theorem states:

If $f(t)$ has the transform $F(s)$ then:

$$\mathcal{L}[f(t-a)u(t-a)] = F(s)e^{-as}$$

The time shifting theorem states that delaying a signal by a time units results in multiplying the signal's transform by e^{-as} . This is somewhat subtle and may require additional explanation. First note the initial ‘if’ statement in the theorem. What the theorem says is if we have a function $f(t)$ and we want to know what the shifted version of the transform is then we can use this theorem.

For example, consider $(t - 120)u(t - 120)$. In this case $f(t) = t$, therefore $F(s) = 1/s^2$. The theorem lets us answer the question what would be the transform if we shifted the function by 120 time units? Using the second time shifting theorem we can rewrite:

$$\mathcal{L}[(t - 120)u(t - 120)] = e^{-120s}\mathcal{L}[t]$$

$$= e^{-120s} \left(\frac{1}{s^2} \right)$$

Example 10.4

- What is the transform of:

$$tu(t)$$

In this case $f(t) = t$ and we are not shifting t , therefore the second shift theorem does not apply. Instead so long as $t > 0$, the transform is given by:

$$F(s) = \frac{1}{s^2}$$

Note that $u(t)$ makes no significant contribution to the function because its value is 1

2. What is the transform of:

$$(t - a) u(t - a)$$

In this case $f(t) = t$, therefore $F(s) = 1/s^2$ and the transform is then given by:

$$\frac{1}{s^2} e^{-as}$$

3. What is the transform of:

$$\sin(a(t - 3)) u(t - 3)$$

The function is in the appropriate form to use the second shifting theorem where we identify $f(t) = \sin(at)$. Therefore:

$$F(s) = \frac{a}{s^2 + a^2}$$

The transform can then be written as:

$$e^{-3s} \frac{a}{s^2 + a^2}$$

The shift theorem can also be modified as follows. Let $g(t) = f(t + a)$, then $g(t - a) = f(t)$. Using the second time shifting theorem we can write:

$$\begin{aligned}\mathcal{L}[u(t - a)f(t)] &= \mathcal{L}[u(t - a)g(t - a)] \\ &= e^{-as}\mathcal{L}[g(t)] \\ &= e^{-as}\mathcal{L}[f(t + a)]\end{aligned}$$

10.5 Inverse Transforms

A key element in the application of the Laplace transform is the ability to evaluate the inverse transform. Sometimes this can be done simply by looking up the transform in a table and locating the inverse. For example, find the inverse transform for the following expression:

$$Y(s) = \frac{5}{s - 2}$$

Using the third row in Table 10.1 we can see that the inverse transform is given by:

$$y = 5e^{2t}$$

More often however finding the inverse transform directly from a table is not possible. The problem arises because transforms of differential equations will often be a quotient of two polynomials:

$$Y(s) = \frac{F(s)}{G(s)}$$

In order to find the inverse transform we must break the expression, $Y(s)$, into simpler terms using partial fractions.

Partial Fractions

Computing partial fractions is a way of representing a complicated fraction as a sum of simpler terms. It is relatively easy to sum a series of fractions into a more complex fraction, for example:

$$\frac{8}{x+2} + \frac{4}{x-5} = \frac{12x-32}{(x+2)(x-5)}$$

The question here is how we can decompose a complex fraction into simpler terms. There are a number of different ways to determine partial fractions but only one will be described here. The first important rule is that the degree of the numerator must be less than the degree of the denominator. If not, then the numerator must be divided by the denominator using polynomial long division. For example, in the following, the numerator has a higher degree (3) than the denominator (2):

$$y = \frac{x^3}{x^2 - 1}$$

Polynomial division yields x remainder x . So that:

$$\frac{x^3}{x^2 - 1} = \frac{x}{x^2 - 1} + x$$

Once the expression has the right degree structure we can examine each factor in the denominator. Each factor will result in a term in the partial fraction. Table 10.2 shows a list of rules for dealing with factors in the denominator.

Factor in Denominator	Term in Partial Fraction
$ax + b$	$\frac{A}{ax + b}$
$(ax + b)^n$	$\frac{A_1}{ax + b} + \frac{A_2}{(ax + b)^2} + \dots + \frac{A_n}{(ax + b)^n}$
$ax^2 + bx + c$	$\frac{Ax + B}{ax^2 + bx + c}$
$(ax^2 + bx + c)^n$	$\frac{A_1x + B_1}{ax^2 + bx + c} + \frac{A_2x + B_2}{(ax^2 + bx + c)^2} + \dots + \dots + \frac{A_nx + B_n}{(ax^2 + bx + c)^n}$

Table 10.2 Rules for Determining Partial Fractions

Once the rules in the table have been applied it then remains to evaluate the numerator factors, A , B , etc. Let's look at some examples.

1) Determine the partial fraction for the expression:

$$\frac{8x - 42}{x^2 + 3x - 18}$$

First we factor the denominator:

$$\frac{8x - 42}{x^2 + 3x - 18} = \frac{8x - 42}{(x + 6)(x - 3)}$$

Using the rule from the first row of Table 10.2 we can decompose the expression into:

$$\frac{8x - 42}{(x + 6)(x - 3)} = \frac{A}{x + 6} + \frac{B}{x - 3}$$

To determine the values for A and B , we first eliminate the denominator by multiplying both sides by the denominator. This yields:

$$8x - 12 = A(x - 3) + B(x + 6)$$

The key here is to realize that this expression is true whatever the value of x . Therefore we can set values to x to selectively eliminate terms. For example, set $x = 3$. This will eliminate the A factor ($3 - 3 = 0$):

$$8(3) - 12 = B(3 + 6)$$

Solving for B gives:

$$B = -2$$

We can do the same thing to compute A by setting $x = -6$:

$$8(-6) - 12 = A(-6 - 3)$$

This yields:

$$A = 10$$

The final partial fraction is therefore:

$$\frac{8x - 42}{x^2 + 3x - 18} = \frac{10}{x + 6} - \frac{2}{x - 3}$$

2) Determine the partial fraction for the expression:

$$\frac{9 - 9x}{2x^2 + 7x - 4} = \frac{9 - 9x}{(2x - 1)(x + 4)}$$

As before we will factor the denominator:

$$\frac{9 - 9x}{2x^2 + 7x - 4} = \frac{A}{2x - 1} + \frac{B}{x + 4}$$

Eliminating the denominator gives:

$$9 - 9x = A(x + 4) + B(2x - 1)$$

We can now set $x = -4$ and $x = 1/2$ to determine A and B :

$$\frac{9 - 9x}{2x^2 + 7x - 4} = \frac{1}{2x - 1} - \frac{5}{x + 4}$$

3) Determine the partial fraction for the expression:

$$\frac{4x^2}{(x - 1)(x - 2)^2}$$

Notice that the denominator contains a square factor. The second row in the factor Table 10.2 shows us how to write out the individual terms in the partial fractions, namely:

$$\frac{4x^2}{(x - 1)(x - 2)^2} = \frac{A}{x - 1} + \frac{B}{x - 2} + \frac{C}{(x - 2)^2}$$

Again we will multiply top and bottom by the denominator from the original equation, $(x - 1)(x - 2)^2$, to yield:

$$4x^2 = A(x - 2)^2 + B(x - 1)(x - 2) + C(x - 1)$$

As before we can eliminate the B and C terms by setting $x = 1$ and the A and B terms by setting $x = 2$. This allows us to determine the values for A and C respectively:

$$A = 4 \quad C = 16$$

The problem now is how to determine B since there is no value for x that will simultaneously eliminate the A and C terms. In this case a different strategy is used. Since we know the values of A and C , all we need to do is pick a value for x . This means that all terms are set except for B . An easy value to assign to x is zero. This allows us to write:

$$\begin{aligned} 4(0)^2 &= (4)(-2)^2 + B(-1)(-2) + 16(-1) \\ &= 16 + 2B - 16 \\ 0 &= B \end{aligned}$$

That is there is no B term. We can now write the final partial fraction as:

$$\frac{4x^2}{(x - 1)(x - 2)^2} = \frac{4}{x - 1} + \frac{16}{(x - 2)^2}$$

4) Determine the partial fraction for the expression:

$$\frac{4}{x^2(x+k)}$$

Rewrite the equation to make it clearer how to use Table 10.2:

$$\frac{4}{(x-0)^2(x+k)}$$

We obtain one fraction from $(x+k)$ and two fractions from $(x-0)^2$ so that:

$$\frac{4}{(x-0)^2(x+k)} = \frac{A}{x} + \frac{B}{x^2} + \frac{C}{x+k}$$

Multiply both sides by $(x-0)^2(x+k)$:

$$4 = Ax(x+k) + B(x+k) + Cx^2$$

Now collect like terms in x :

$$4 = (A+C)x^2 + (Ak+B)x + Bk$$

Match equivalent terms (x^2 , x , constant) on the left and right side. Since there is no x^2 or x term on the left we can state:

$$A + C = 0$$

$$Ak + B = 0$$

$$Bk = 4$$

Solving for A , B and C yields:

$$A = -\frac{4}{k^2}$$

$$B = \frac{4}{k}$$

$$C = \frac{4}{k^2}$$

Finally inserting these coefficients back into the original decomposition gives:

$$\frac{4}{x^2(x+k)} = \frac{4}{kx^2} - \frac{4}{k^2x} + \frac{4}{k^2(x+k)}$$

10.6 Solving Differential Equations

The Laplace transform is a very useful tool for solving linear differential equations. The procedure is in three steps (Figure 10.6):

1. Convert the differential equations into the Laplace Domain, for example $x(t)$ terms become $X(s)$.
2. Solve the set of algebraic equations for $X(s)$.
3. Carry out the reverse Laplace transform to recover the solution, $x(t)$.

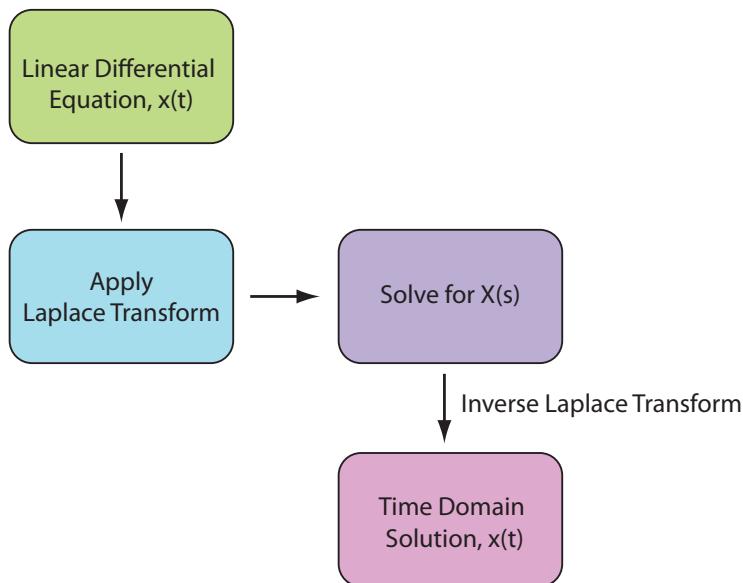


Figure 10.6 Solving Differential Equations using Laplace Transforms

We can illustrate this procedure with the following very simple example. Consider:

$$\frac{dx}{dt} = -kx$$

Taking the Laplace transform on both sides (using the Laplace transform table) yields:

$$sX(s) - x(0) = -kX(s)$$

Solving for $X(s)$:

$$X(s) = \frac{x(0)}{s + k}$$

Taking the inverse Laplace transform (again using the Laplace transform table) yields the solution:

$$x(t) = x(0)e^{-kt}$$

Let us apply the Laplace transform method to a more complex problem. Consider the two step pathway:

$$x_1 \xrightarrow{v_1} x_2 \xrightarrow{v_2}$$

The system has two state variables, x_1 and x_2 with x_2 draining into the environment. Assuming first-order kinetics, the system of differential equations for this model is:

$$\begin{aligned}\frac{dx_1}{dt} &= -k_1 x_1 \\ \frac{dx_2}{dt} &= k_1 x_1 - k_2 x_2\end{aligned}$$

Applying the Laplace rules from Table 10.1, we obtain:

$$\begin{aligned}sX_1(s) - x_1(0) &= -k_1 X_1(s) \\ sX_2(s) - x_2(0) &= k_1 X_1(s) - k_2 X_2(s)\end{aligned}$$

This set of equations represent a set of linear algebraic equations in $X_1(s)$ and $X_2(s)$. $x_1(0)$ and $x_2(0)$ represent the initial conditions for x_1 and x_2 at time zero. If we assume that $x_1(0) = a$ and $x_2(0) = 0$ at time zero, then:

$$\begin{aligned}sX_1(s) - a &= -k_1 X_1(s) \\ sX_2(s) &= k_1 X_1(s) - k_2 X_2(s)\end{aligned}$$

Solving for $X_1(s)$ and $X_2(s)$ yields:

$$\begin{aligned}X_1(s) &= \frac{a}{s + k_1} \\ X_2(s) &= \frac{k_1 a}{(s + k_1)(s + k_2)}\end{aligned}$$

Now that we have solved for $X_1(s)$ and $X_2(s)$ we must apply the inverse Laplace transform to recover the time domain solution. The inverse transform can be obtained by using the Laplace transform table in reverse. However we do not have entries in the table that resemble the right-hand side of $X_2(s)$ (A more complete table is given in Appendix H). In order to proceed we must split the equation into simpler terms. The hope is that the simpler terms will be found in a table of Laplace transforms. To

do this we use partial fractions. Let us first find the inverse transform of $X_1(s)$, this is straight forward because the term $a/(s + k_1)$ is the 3rd entry in the table, that is:

$$x_1 = a e^{-k_1 r}$$

For the second equation we will use partial fractions.

$$X_2(s) = \frac{k_1 a}{(s + k_1)(s + k_2)} \quad (10.2)$$

Let us separate $X_2(s)$ as follows:

$$\frac{k_1 a}{(s + k_1)(s + k_2)} = \frac{A}{s + k_1} + \frac{B}{s + k_2}$$

Multiply both sides by the left-side denominator to yield:

$$k_1 a = A(s + k_2) + B(s + k_1)$$

This equation must be true for all s , therefore let us set $s = -k_2$. This allows us to eliminate the A term giving us:

$$k_1 a = B(-k_2 + k_1)$$

So that:

$$B = \frac{k_1 a}{k_1 - k_2}$$

A can be determined in a similar way by setting $s = -k_1$ which yields:

$$A = -\frac{k_1 a}{k_1 - k_2}$$

Inserting A and B yields:

$$X_2(s) = \frac{k_1 a}{(k_1 - k_2)} \frac{1}{s + k_2} - \frac{k_1 a}{(k_1 - k_2)} \frac{1}{s + k_1}$$

We can now use the reverse transform of $a/(s + k)$ to yield an equation for $x_2(t)$:

$$x_2(t) = \frac{k_1 a}{(k_1 - k_2)} \left[e^{-k_2 t} - e^{-k_1 t} \right]$$

Recall that $a = x_1(0)$. The use of the inverse transform to derive differential equations is perhaps not so important today given the availability of symbolic applications such as Mathematica or the open source tools such as Sage (<http://www.sagemath.org/>) or wxMaxima (<https://github.com/wxmaxima/> and Maxima <http://maxima.sourceforge.net/>). However, the ability to form the transforms is still very important because the elimination of time and its replacement with s leads to a host of new analysis methods.

10.7 Laplace Transforms of Common Signals

In this section we'll look at some Laplace transforms of common signals. We have already seen from the table of transforms that the unit step and impulse signal have the transforms:

$$\mathcal{L}[u(t)] = \frac{1}{s}$$

$$\mathcal{L}[\delta(t)] = 1$$

Ramp The Laplace transform for the ramp function, at is given by (solution by integration by parts):

$$F(s) = a \int_0^\infty t e^{-st} dt = \frac{a}{s^2}$$

Exponential signal The Laplace transform for the exponential function, Ae^{-at} is given by:

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

Delayed signal The Laplace transform for a delayed signal $x(t - a)$ is given by:

$$\mathcal{L}(x(t - a)) = e^{-sa} X(s)$$

This is the second shifting theorem (Section 10.4). For example the Laplace transform of a delayed unit step response, $u(t - a)$ is given by:

$$\mathcal{L}[u(t - a)] = \frac{e^{-at}}{s}$$

The Laplace transform for a pulse, $u(t) - u(t - 2)$ can be evaluated using the linearity property:

$$\begin{aligned} F(s) &= \mathcal{L}[u(t) - u(t - 2)] = \mathcal{L}[u(t)] - \mathcal{L}[u(t - 2)] \\ &= \frac{1}{s} - \frac{e^{-2s}}{s} = \frac{1 - e^{-2s}}{s} \end{aligned}$$

Sine Wave The Laplace transform for a sinusoidal wave, $A \sin(\omega t)$ is given by

$$F(s) = \frac{A\omega}{(s^2 + \omega^2)}$$

10.8 Additional Examples

Example 10.5

Solve the following differential equation:

$$\frac{dx}{dt} = v - kx$$

with initial condition $x(0) = 0$. Apply the Laplace transform to both sides of the equation:

$$sX(s) = \frac{v}{s} - kX(s)$$

Solve for $X(s)$ gives:

$$X(s) = \frac{v}{s(s + k)}$$

Taking the inverse Laplace transform yields the final solution:

$$x = \frac{v}{k} \left(1 - e^{-kt} \right)$$

Example 10.6

Solve the following differential equation:

$$\frac{dx}{dt} = e^{2t} - 3x$$

with initial condition $x(0) = 6$. Apply the Laplace transform to both sides of the equation:

$$sX(s) - 6 = \frac{1}{s-2} - 3X(s)$$

Solve for $X(s)$ gives:

$$X(s) = \frac{s-6-11}{(s-2)(s+3)}$$

Decomposing into partial fractions:

$$X(s) = \frac{1}{5} \frac{1}{(s-2)} + \frac{1}{5} \frac{29}{(s+3)}$$

Taking the inverse Laplace transform yields the final solution:

$$x = \frac{1}{5} e^{-3t} (e^{5t} + 29)$$

Example 10.7

Solve the following system of differential equations:

$$\frac{dx}{dt} = x + 5y \quad x(0) = 0$$

$$\frac{dy}{dt} = -x - y \quad y(0) = 1$$

Apply the Laplace transform to both equations:

$$\begin{aligned}sX(s) - x(0) &= X(s) + 5Y(s) \\ sY(s) - y(0) &= -X(s) - Y(s)\end{aligned}$$

Solving for $X(s)$ and $Y(s)$ gives:

$$\begin{aligned}X(s) &= \frac{5}{s^2 + 4} = \frac{5}{2} \frac{2}{s^2 + 2^2} \\ Y(s) &= \frac{s - 1}{s^2 + 4}\end{aligned}$$

Taking the inverse Laplace transform yields the final solution:

$$\begin{aligned}x &= \frac{5}{2} \sin(2t) \\ y &= \frac{1}{2} (2 \cos(2t) - \sin(2t))\end{aligned}$$

Example 10.8

Solve the following second-order differential equation:

$$\frac{d^2x}{dt^2} + \frac{dx}{dt} - 2x = 4 \quad x(0) = 2 \quad \frac{dx}{dt}(0) = 1$$

Apply the Laplace transform:

$$(s^2 X(s) - 2s - 1) + (sX(s) - 2) - 2X(s) = \frac{4}{s}$$

Combine terms:

$$(s^2 + s - 2)X(s) = \frac{4}{s} + 2s + 3$$

$$X(s) = \frac{2s^2 + 3s + 4}{s(s - 1)(s + 2)}$$

Before we can evaluate the inverse Laplace transform we must first decompose the equation into partial fractions:

$$\frac{2s^2 + 3s + 4}{s(s - 1)(s + 2)} = \frac{A}{s} + \frac{B}{s - 1} + \frac{C}{s + 2}$$

$$2s^2 + 3s + 4 = A(s - 1)(s + 2) + Bs(s + 2) + Cs(s - 1)$$

From the last equation we can determine the unknown coefficient, A , B and C .

Setting $s = 0$ eliminates B and C terms so that $A = -2$.

Setting $s = 1$ eliminates A and C terms so that $B = 3$.

Setting $s = -2$ eliminates A and B terms so that $C = 1$.

This gives us:

$$X(s) = \frac{-2}{s} + \frac{3}{s-1} + \frac{1}{s+2}$$

Taking the inverse Laplace transform for each term yields the final solution:

$$y(t) = 3e^t + e^{-2t} - 2$$

Exercises

1. Using tables, find the Laplace transform for the following functions:

- (a) pulse, $u(t-2) - u(t-4)$
- (b) $t^2 - 3t + 4$
- (c) $e^{-t}u(t-\tau)$

2. Decompose the following equations into their partial fractions:

a)

$$\frac{s+4}{(s+1)(s+2)(s+3)}$$

b)

$$\frac{3s+1}{s(s+1)(s+2)}$$

c)

$$\frac{3s+1}{s(s^2+4s+4)}$$

d)

$$\frac{3s^2+10s+10}{s^3+4s^2+6s+4}$$

Hint: $s^2 + 2s + 2 = (s+1)^2 + 1$ $2s + 4 = 2(s+1) + 2$

$$\frac{2(s+2)}{s^2+2s+2} + \frac{1}{s+2}$$

3. Find the inverse Laplace transform of:

a) $F(s) = \frac{5}{s+3} - \frac{1}{2s-4} + \frac{7}{s^2}$

b) $F(s) = \frac{2s}{s^2+25} + \frac{3}{s^2+16}$

c) $F(s) = \frac{s+2}{(s+3)(s-4)}$

4. Solve the following differential equation systems using Laplace transforms.

a)

$$\frac{dx}{dt} = 3 - 2t \quad x(0) = 0$$

b)

$$\frac{dx}{dt} = e^{5t} + 5x \quad x(0) = 0$$

5. Given the following system:

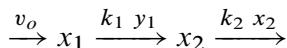
$$\frac{dx_1}{dt} = x_1$$

$$\frac{dx_2}{dt} = x_1 + x_2$$

with initial conditions, $x_1(0) = 1$ and $x_2(0) = 1$, solve for x_1 and x_2 using the Laplace transform method. Confirm that the solution is:

$$x_1 = e^t \quad \text{and} \quad x_2 = e^t(t+1)$$

6. Consider the following two species biochemical pathway:



with ODEs:

$$\frac{dx_1}{dt} = v_o - k_1 x_1$$

$$\frac{dx_2}{dt} = k_1 x_1 - k_2 x_2$$

Derive the free, forced and full time-domain solutions for this system for both variables, x_1 and x_2 in response to a step function in v_o .

10.9 Appendix

Proof for Equation 10.1. Find the Laplace transform of $f(at)$.

$$\mathcal{L}(f(at)) = \int_0^\infty e^{-st} f(at) dt$$

Let $x = at$ so that $t = x/a$ and:

$$dt = \frac{dx}{a}$$

When $t = 0, z = 0$ and when $t = \infty, x = \infty$. Substituting t :

$$\mathcal{L}(f(at)) = \int_0^\infty e^{-(s/a)x} f(x) \frac{dx}{a}$$

$$\mathcal{L}(f(at)) = \frac{1}{a} \int_0^\infty e^{-(s/a)x} f(x) dx$$

Therefore:

$$\mathcal{L}(f(at)) = \frac{1}{a} F\left(\frac{s}{a}\right)$$

11

Zero-State Response

“An empty vessel makes the loudest sound.”

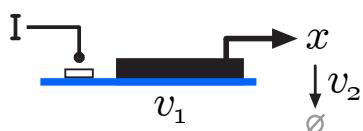
— William Shakespeare, 1564-1616

11.1 Introduction

In an earlier chapter (Chapter 9) we introduced the notion of the zero-input and zero-state response. In this chapter we will look in more detail at the zero-state response. To remind ourselves, the zero-state response is the response of a system to an input when the initial conditions are zero.

11.2 Zero-State Response to an Impulse

The response of a system to an impulse is one of the most important to understand. Consider the single gene circuit shown below:



where I is the input and x the single state variable. We will assume that $v_2 = k_2x$. The differential equation for this system is given by:

$$\frac{dx}{dt} = u(t) - k_2x$$

where $u(t)$ is an input signal that determines v_1 . To compute the zero-state response we set $x(0) = 0$ and apply Laplace transforms on both sides to yield:

$$sX(s) = U(s) - k_2X(s)$$

In the transformed equation we represent the input signal by the transform, $U(s)$. Solving for $X(s)$ yields:

$$X(s) = U(s) \frac{1}{s + k_2}$$

To obtain the response of the system to an arbitrary input we need only replace $U(s)$ with the appropriate transform. Of particular importance is the impulse response because its transform is $\mathcal{L}(\delta(t)) = 1$ so that $U(s) = 1$. The impulse response is therefore given by:

$$X(s) = \frac{1}{s + k_2}$$

The time domain solution to this response is given by:

$$x(t) = e^{k_2 t}$$

that is a simple exponential decay.

If we applied a different kind of input, for example a unit step response, $u(t)$, then $U(s) = 1/s$, so that the response of the system is now:

$$X(s) = \frac{1}{s} \left(\frac{1}{s + k_2} \right)$$

Here the time domain response is given by:

$$x(t) = \frac{1}{k_2} \left(1 - e^{-k_2 t} \right)$$

which describes a monotonic increase which plateaus at the steady state value, $1/k_2$. Given that the application of an impulse means that $U(s) = 1$, the response of a system to an impulse is of particular importance. If we know the response of a system to an impulse we can use this information to discover the response to any input simply by multiplying the impulse response by the new input.

The impulse response of a system therefore fully characterizes the internal dynamics of a system. The impulse response uncovers what is described as the **transfer function** of the system, in this case $1/(s + k_2)$. As we will see, the transfer function is of central importance to control theory. In the control literature the transfer function is often represented by the symbol $H(s)$.

The **impulse response** of a system is the output of the system when we apply an impulse, $\delta(t)$, to the system with all initial conditions **set to zero**.

The response of a system to an impulse in zero state is called the impulse response and is described by the **transfer function**, $H(s)$.

11.3 Convolution

In the last section we saw that if an impulse, $\delta(t)$ is applied to a system with all initial conditions set to zero the output is called the **impulse response**. In the Laplace domain this response is often represented by the symbol $H(s)$. Here we wish to consider the same concept but now in the time domain where the impulse response is represented by the symbol $h(t)$.

We begin by approximating a function $q(t)$ by a series of pulses shown in Figure 11.2. To do this we first define a pulse of unit height and width Δt by the symbol (See Figure 11.1):

$$p(t - n\Delta)$$

Note that Δt has been shorted to Δ . n refers to the n^{th} pulse. This means that our pulse occurs at $n\Delta t$ time (or $n\Delta$ using the short notation).

Let us now use a series of pulses to approximate a function $q(t)$. Referring to Figure 11.2, the height of a pulse at $n\Delta$ is $q(n\Delta)$. We can describe $q(t)$ approximately using the relation:

$$q(t) \simeq \sum_{-\infty}^{\infty} q(n\Delta) p(t - n\Delta)$$

where we sum over all pulses. The thinner the pulses the better the approximation. In the limit as Δ tends to zero, we can write the sum as a limit, that is:

$$q(t) = \lim_{\Delta \rightarrow 0} \sum_{-\infty}^{\infty} q(n\Delta) p(t - n\Delta)$$

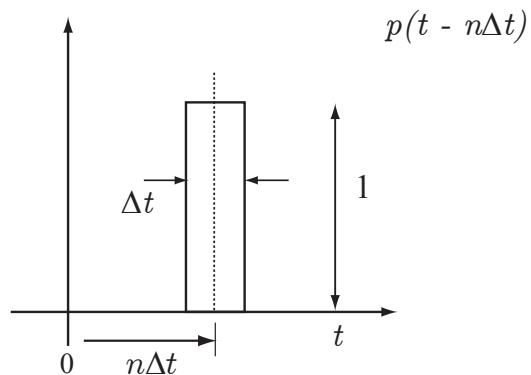


Figure 11.1 Unit height pulse of width Δt , denoted $p(t - n\Delta t)$ or $p(t - n\Delta)$ using the short notation..

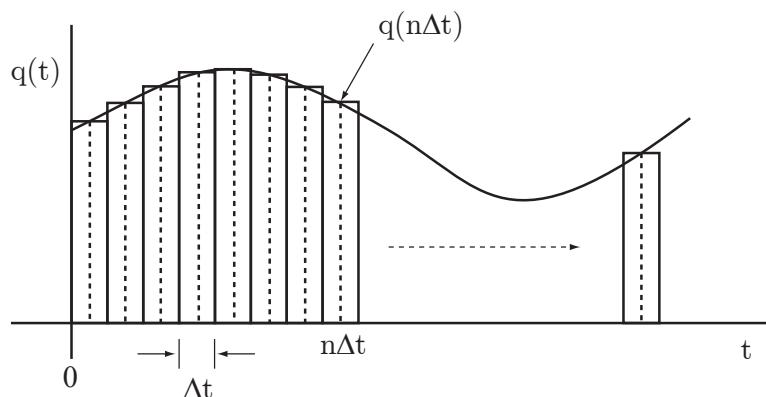


Figure 11.2 Approximation of a function $q(t)$ using a series of pulses of width Δt . The height of the n th pulse is $q(n\Delta t)$.

We now make one important change, we multiply top and bottom by Δ to obtain:

$$q(t) = \lim_{\Delta \rightarrow 0} \sum_{-\infty}^{\infty} q(n\Delta) \left[\frac{1}{\Delta} p(t - n\Delta) \right] \Delta \quad (11.1)$$

Recall that $p(t - n\Delta)$ has unit height. If we divide the pulse by Δ , the height of the pulse becomes $1/\Delta$. Since the width of the pulse is Δ , the area of $p(t - n\Delta)/\Delta$ must be $1/\Delta \times \Delta$, which is *one* and *unitless*. This is an important change because in the limit as $\Delta \rightarrow 0$, $p(t - n\Delta)/\Delta$ will converge to an impulse $\delta(t - \tau)$ where we have replaced $n\Delta$ with τ :

$$\lim_{\Delta \rightarrow 0} \frac{1}{\Delta} p(t - n\Delta)$$

becomes an impulse:

$$\delta(t - \tau) = \lim_{\Delta \rightarrow 0} \frac{1}{\Delta} p(t - n\Delta)$$

Taking the limit of equation (11.1) we note that the sum becomes an integral, such that the input function $q(t)$ can be written as:

$$q(t) = \int_{-\infty}^{\infty} q(\tau) \delta(t - \tau) d\tau \quad (11.2)$$

In the limit, $n\Delta$ is replaced with τ and Δ becomes $d\tau$. The function $q(t)$ has been expressed as a sequence of scaled impulses. With this result the impulse response in the time domain can be investigated.

Let us define the impulse response, $h(t)$ as the output resulting from the application of an impulse, $\delta(t)$, that is:

$$\delta(t) \rightarrow h(t)$$

Assuming the system is time invariant, this means applying an impulse at a delayed time τ will yield the same impulse response but delayed:

$$\delta(t - \tau) \rightarrow h(t - \tau)$$

If we assume the system is linear and we apply an input of magnitude $q(\tau)$ at τ , then the output will be $q(\tau)h(t - \tau)$:

$$q(\tau)\delta(t - \tau) \rightarrow q(\tau)h(t - \tau)$$

Integrating over all times τ leads to:

$$\int_{-\infty}^{\infty} q(\tau)\delta(t - \tau) d\tau \rightarrow \int_{-\infty}^{\infty} q(\tau)h(t - \tau) d\tau$$

However we know from equation (11.2) that:

$$\int_{-\infty}^{\infty} q(\tau) \delta(t - \tau) d\tau = q(t)$$

That is the application of an input $q(t)$ yields the output response:

$$q(t) \rightarrow \int_{-\infty}^{\infty} q(\tau) h(t - \tau) d\tau$$

Since the right-hand side is the output, $y(t)$, we can write:

$$y(t) = \int_{-\infty}^{\infty} q(\tau) h(t - \tau) d\tau \quad (11.3)$$

This integral is called the **convolution integral** and is often represented more simply using the notation:

$$y(t) = q(t) * h(t) \quad (11.4)$$

Without proof we will also state the following useful properties:

$$[af(t) * g(t)] = a [f(t) * g(t)] = [g(t) * ag(t)]$$

$$[f(t) * g(t)] = [g(t) * f(t)]$$

$$[f(t) * g(t)] * h(t) = f(t) * [g(t) * h(t)]$$

Convolution means “a twisting together” and describes the result of applying an input signal, $q(t)$, to a system.

The time domain output of an LTI system is given by the convolution of the impulse response with the input signal. Intuitively, convolution is the sum of many impulse responses, each one with a different time delay and scaled according to the value of the input signal.

The importance of convolution in this context is that if the impulse response of a system, $h(t)$, is known, then the response to any input can be determined by replacing $q(t)$ and recomputing the convolution. In other words, the system is fully characterized by the impulse response. The impulse response is all we need to know in order to compute the response of the system to any other input. We have already encountered this concept in the last section when we discussed the impulse in the

Laplace domain. However, unlike the Laplace domain where computing the system response involves multiplying the input function by the transfer function, in the time domain the same operation involves evaluating an integral. Equation (11.4) is the equivalent impulse response we saw in the Laplace domain, where the response of a system was given by $Y(s) = Q(s)H(s)$ (Substituting the same notation to make the comparison clearer).

A system can be fully characterized by measuring the response it has to a single impulse.

Without providing the proof we will state a very important theorem that relates the time-domain convolution to the Laplace domain:

$$\mathcal{L}[f(t) * g(t)] = F(s) G(s) \quad (11.5)$$

$$f(t) * g(t) = \mathcal{L}^{-1}[F(s) G(s)] \quad (11.6)$$

In the Laplace domain, convolution is equivalent to multiplication which is much simpler to carry out than attempting to find the convolution integral in the time domain itself. To determine the response of a system to an arbitrary input function it can be easier to work in the Laplace domain.

Example 11.1

Derive the convolution for the system with input signal $q(t) = t$ and transfer function, $h(t) = \sin(t)$. To derive the convolution, it is sometimes easier to first move the problem into the Laplace domain and use equation (11.6) to compute the convolution. We first take the Laplace transform of each function, that is:

$$Q(s) = \frac{1}{s^2} \quad H(s) = \frac{1}{1 + s^2}$$

The product of the Laplace transforms is:

$$Q(s)H(s) = \frac{1}{s^2(1 + s^2)} = \frac{1}{s^2} - \frac{1}{1 + s^2}$$

The inverse transform, which is the convolution, is then:

$$\int_{-\infty}^{\infty} q(\tau)h(t - \tau) = t - \sin(t)$$

Graphical Understanding of Convolution

At first sight convolution taken from a purely mathematical perspective may seem difficult to understand. However, operationally convolution is quite straight forward. A convolution is a superposition

of impulse responses. This involves the repeated application of an impulse where each impulse is delayed and scaled by the input function, $q(t)$ at that delayed time. Each time we apply the impulse, say at delayed time $(t - \tau)$ we record the response, $h(t - \tau)$. We then take all the recorded responses, $h(t - \tau)$ and sum them. This yields the convolution, that is $y(t)$, the response of the system to the input function, $q(t)$ (See Figure 11.3).

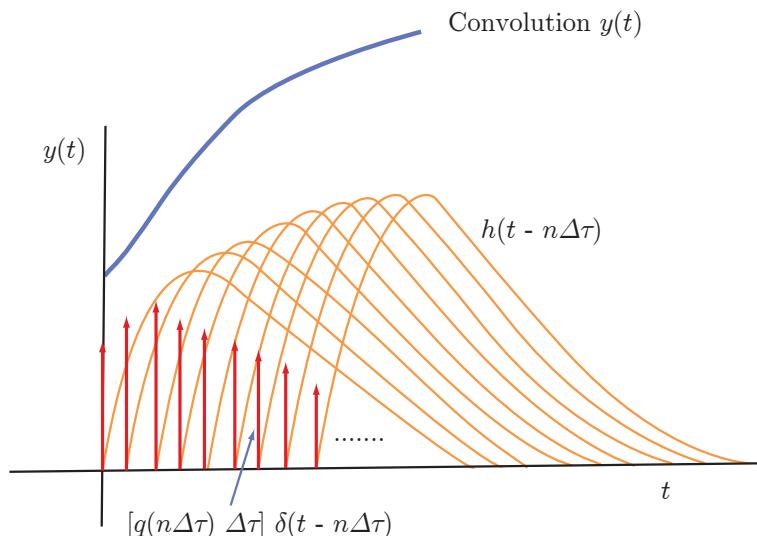


Figure 11.3 Graphical Explanation for Convolution. A sequence of impulses (Vertical arrows) whose areas are scaled by the input function, $q(t)$ is applied to the system. In each case a response, $h(t)$ is recorded. The sum of the responses yields the output $y(t)$, upper curve.

11.4 Relationship to State Equations

Let us investigate the zero-state response from the perspective of the state equation (Chapter 8). Let us first apply the Laplace transform method to the homogeneous equation:

$$\frac{d\mathbf{x}}{dt} = \mathbf{A}\mathbf{x}$$

Taking the Laplace transform on both sides yields:

$$\mathcal{L}\left(\frac{d\mathbf{x}}{dt}\right) = \mathcal{L}(\mathbf{A}\mathbf{x})$$

$$s\mathbf{X}(s) - \mathbf{x}_o = \mathbf{A}\mathbf{X}(s)$$

where \mathbf{x}_o is the vector of initial conditions. Collecting the term $X(s)$ yields:

$$(s\mathbf{I} - \mathbf{A})X(s) = \mathbf{x}_o$$

$$X(s) = (s\mathbf{I} - \mathbf{A})^{-1}\mathbf{x}_o \quad (11.7)$$

Taking the inverse transform on both sides (note that \mathbf{x}_o is a constant) yields the solution $\mathbf{x}(t)$:

$$\mathbf{x}(t) = \mathcal{L}^{-1}[(s\mathbf{I} - \mathbf{A})^{-1}]\mathbf{x}_o \quad (11.8)$$

The expression, $(s\mathbf{I} - \mathbf{A})^{-1}$ is often called the **resolvent** while $\mathcal{L}^{-1}[(s\mathbf{I} - \mathbf{A})^{-1}]$ is called the **state-transition matrix**, denoted by $\phi(t)$ (See equation 9.11):

$$\phi(t) = e^{\mathbf{A}t} = \mathcal{L}^{-1}[(s\mathbf{I} - \mathbf{A})^{-1}] \quad (11.9)$$

Equation (11.8) is often written in the shortened version:

$$\mathbf{x}(t) = \phi(t)\mathbf{x}_o \quad (11.10)$$

Note that from equation (9.10) the state-transition matrix is also the matrix exponential,

$$\phi(t) = e^{\mathbf{A}t} \quad (11.11)$$

Another way show that inverse transform of the resolvent is the matrix exponential is as follows. The standard power series expansion for $1/(s - x)$ in x is given by:

$$(s - x)^{-1} = \sum_{i=0}^{\infty} s^{-(i+1)}x^i$$

Without proof, this can be extended to the matrix form:

$$(s\mathbf{I} - \mathbf{A})^{-1} = \sum_{i=0}^{\infty} s^{-(i+1)}\mathbf{A}^i = \frac{\mathbf{I}}{s} + \frac{\mathbf{A}}{s^2} + \frac{\mathbf{A}^2}{s^3} + \dots$$

If we take the inverse transform of this series we obtain:

$$\mathcal{L}^{-1}[(s\mathbf{I} - \mathbf{A})^{-1}] = \mathbf{I} + \mathbf{A}t + \frac{\mathbf{A}^2t^2}{2!} + \frac{\mathbf{A}^3t^3}{3!} + \dots = e^{\mathbf{A}t}$$

This will be recognized as the matrix exponential (See equation 9.8).

Example 11.2

Apply (11.8) to the homogenous example given Chapter 9, example 1. In this example:

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

Applying equation (11.8) we obtain:

$$\mathbf{x}(t) = \mathcal{L}^{-1} \left(\left(s \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} - \begin{bmatrix} -2 & 0 \\ 1 & -1 \end{bmatrix} \right)^{-1} \mathbf{x}_o \right)$$

Inverting the matrix, yields:

$$\mathbf{x}(t) = \mathcal{L}^{-1} \left(\frac{1}{(s+2)(s+1)} \begin{bmatrix} s+1 & 0 \\ 1 & (s+2) \end{bmatrix} \mathbf{x}_o \right)$$

Multiplying out each row and noting the initial conditions of $x_1(0) = 2, x_2(0) = 3$, gives:

$$x_1 = \mathcal{L}^{-1} \left(\frac{2}{s+2} \right)$$

$$x_2 = \mathcal{L}^{-1} \left(\frac{3s+8}{(s+1)(s+2)} \right) = \mathcal{L}^{-1} \left(\frac{5}{(s+1)} - \frac{2}{s+2} \right)$$

The inverse transforms for these expressions can be easily determined from standard Laplace tables to yield:

$$\begin{aligned} x_1 &= 2e^{-t} \\ x_2 &= 5e^{-t} + 2e^{-2t} \end{aligned}$$

These solutions correspond to those obtained using the matrix exponential approach, equation (9.13).

Example 11.3

Use Laplace transforms to find $e^{\mathbf{A}t}$ for a system with the following \mathbf{A} matrix:

$$\begin{bmatrix} 0 & 1 \\ -6 & -5 \end{bmatrix}$$

We note from equation (11.9) that $e^{\mathbf{A}t}$ is given by the inverse transform of the resolvent. First we compute $(s\mathbf{I} - \mathbf{A})$:

$$(s\mathbf{I} - \mathbf{A}) = \begin{bmatrix} s & -1 \\ 6 & s+5 \end{bmatrix}$$

Next take the inverse to compute $(sI - A)^{-1}$:

$$(sI - A)^{-1} = \begin{bmatrix} \frac{s+5}{s^2 + 5s + 6} & \frac{1}{s^2 + 5s + 6} \\ \frac{-6}{s^2 + 5s + 6} & \frac{s}{s^2 + 5s + 6} \end{bmatrix}$$

Applying partial fractions:

$$\begin{bmatrix} \frac{3}{s+2} - \frac{2}{s+3} & \frac{1}{s+2} - \frac{1}{s+3} \\ \frac{6}{s+3} - \frac{6}{s+2} & \frac{3}{s+3} - \frac{2}{s+2} \end{bmatrix}$$

Taking the inverse transform for each elements yields the matrix exponential:

$$e^{At} = \begin{bmatrix} 3e^{-2t} - 2e^{-3t} & e^{-2t} - e^{-3t} \\ 6e^{-3t} - 6e^{-2t} & 3e^{-3t} - 2e^{-2t} \end{bmatrix}$$

Relationship to Impulse Response

In a previous chapter we saw that the solution (9.16) to a non-homogeneous equation was given by:

$$\mathbf{x}(t) = e^{At}\mathbf{x}(0) + \int_0^t e^{A(t-\tau)}\mathbf{B}\mathbf{u}(\tau)d\tau \quad (11.12)$$

Let us take the Laplace transform for the full state equation for \mathbf{x} . That is:

$$\mathcal{L}\left(\frac{d\mathbf{x}}{dt}\right) = \mathcal{L}(\mathbf{Ax} + \mathbf{Bu})$$

$$sX(s) - \mathbf{x}_o = \mathbf{AX}(s) + \mathbf{BU}(s)$$

where \mathbf{x}_o is the vector of initial conditions. Collecting the term $X(s)$ yields:

$$(sI - A)X(s) = \mathbf{x}_o + \mathbf{BU}(s)$$

$$X(s) = (sI - A)^{-1}\mathbf{x}_o + (sI - A)^{-1}\mathbf{BU}(s) \quad (11.13)$$

We now take the inverse Laplace transform on both sides.

$$\mathbf{x}(t) = \mathcal{L}^{-1}((sI - A)^{-1})\mathbf{x}_o + \mathcal{L}^{-1}((sI - A)^{-1}\mathbf{BU}(s))$$

Denoting the resolvent by $\Phi(s)$, we obtain:

$$\mathbf{x}(t) = \phi(t)\mathbf{x}_o + \mathcal{L}^{-1}(\Phi(s)\mathbf{B}\mathbf{U}(s))$$

Using the theorem (11.6):

$$f(t) * g(t) = \mathcal{L}^{-1}[F(s) G(s)]$$

we can write $\mathbf{x}(t)$ as:

$$\mathbf{x}(t) = \phi(t)\mathbf{x}_o + \phi(t) * \mathbf{B}\mathbf{u}(t))$$

Writing the convolution in integral form gives:

$$\mathbf{x}(t) = \phi(t)\mathbf{x}_o + \int_0^t \phi(t-\tau)\mathbf{B}\mathbf{u}(\tau)d\tau$$

Recalling that:

$$\phi(t) = \mathcal{L}^{-1}((s\mathbf{I} - \mathbf{A})^{-1}) = e^{\mathbf{A}t}$$

We can also substitute the state transition matrices with the exponential matrices to finally give:

$$\mathbf{x}(t) = e^{\mathbf{A}t}\mathbf{x}_o + \int_0^t e^{\mathbf{A}(t-\tau)}\mathbf{B}\mathbf{u}(\tau)d\tau$$

We can now recognize the second term in equation (9.16) as a convolution. As an example we can compute the impulse response by setting $u(\tau)$ to $\delta(\tau)$ and $\mathbf{x}(0) = 0$ so that:

$$\mathbf{x}(t) = \int_0^t e^{\mathbf{A}(t-\tau)}\mathbf{B}\delta(\tau)d\tau$$

We can use the sifting property (6.2)¹ to evaluate the integral, where $f(t)$ in the sifting theorem is $e^{\mathbf{A}(t-\tau)}\mathbf{B}$ and a in the term $\delta(t-a)$ is zero:

$$\mathbf{x}(t) = e^{\mathbf{A}t}\mathbf{B}$$

This is how the system responds to an impulse at time zero, when initial conditions are zero.

Initial Conditions Set to Zero

If we set the initial conditions, \mathbf{x}_o to zero then from equation (11.13) we can write:

$$\mathbf{X}(s) = [(s\mathbf{I} - \mathbf{A})^{-1}\mathbf{B}] \mathbf{U}(s) \quad (11.14)$$

¹ $\int_{-\infty}^{\infty} f(t) \delta(t-a)dt = f(a)$

The term $(sI - A)^{-1}B$ is called the **transfer function matrix** or more simply the **transfer function**, sometimes indicated by $H(s)$:

$$X(s) = H(s)U(s)$$

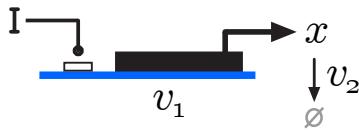
where

$$H(s) = (sI - A)^{-1}B \quad (11.15)$$

This expression corresponds to the forced component of the system and splits the response into two parts, $H(s)$ which is an intrinsic property of the system and $U(s)$ which is the perturbation we apply to the system. Both together give us the effect a perturbation has on the output of the system, $X(s)$.

Example 11.4

Derive the transfer function, $H(s)$, for the following system:



where I is the input and x the single state variable. We will assume that $v_2 = k_2x$. The differential equation for this system is given by:

$$\frac{dx}{dt} = u(t) - k_2x$$

where $u(t)$ is an input signal. In order to derive $H(s)$ we must set the initial conditions, $x(0)$ to zero. Taking the Laplace transform on both sides yields:

$$sX(s) = U(s) - k_2X(s)$$

At this stage we are not interested in any specific input signal, we therefore represent the input signal by the symbol, $U(s)$. Solving for $X(s)$ gives:

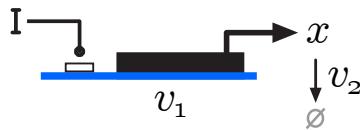
$$X(s) = U(s) \frac{1}{s + k_2}$$

The transfer function is therefore:

$$H(s) = \frac{1}{s + k_2}$$

Example 11.5

Derive the transfer function for the following system using the state space representation:



with ODE:

$$\frac{dx}{dt} = k_1 I - k_2 x$$

We can cast the ODE in the state space representation:

$$\frac{dx}{dt} = [-k_2]x + [k_1]I$$

The \mathbf{A} and \mathbf{B} state variable matrices are given by:

$$\mathbf{A} = [-k_2] \quad \mathbf{B} = [k_1]$$

Inserting these into equation (11.15) gives:

$$\mathbf{X}(s) = (s + k_2)^{-1} [k_1] \mathbf{U}(s)$$

The transfer function is therefore:

$$\mathbf{H}(s) = \frac{k_1}{s + k_2}$$

Example 11.6

Find the step response to I for the system that has the transfer function (see previous example):

$$H(s) = \frac{k_1}{s + k_2}$$

The Laplace transform of the step input, I is given by I/s , therefore the step response, $X(s)$ is given by:

$$X(s) = \frac{I}{s} \frac{k_1}{s + k_2}$$

The inverse transform is then given by:

$$x(t) = \frac{Ik_1}{k_2} \left(1 - e^{-k_2 t}\right)$$

11.5 Transfer Function

In this section we will consider the transfer function in more detail. Consider the Laplace transform for the following LTI system:

$$\mathcal{L}\left(\frac{dx}{dt}\right) = \mathcal{L}(Ax + Bu)$$

If we assume zero initial conditions (See equation 11.14) we can write:

$$X(s) = [(sI - A)^{-1}B]U(s)$$

As indicated in the last section the term $(sI - A)^{-1}B$ is called the **transfer function matrix**, $H(s)$:

$$X(s) = H(s)U(s)$$

There will be $n \times p$ elements in $H(s)$ where n is the number of state variables and p the number of inputs (See Figure 8.1). That is each element of $H(s)$ represents a transfer relationship between a given input and state variable. For a particular element in $H(s)$, we can define the transfer function as:

The **transfer function** for a given state variable and given output is defined as the ratio of the output, $X(s)$ to the input, $U(s)$, of the system in the Laplace domain. All initial conditions are assumed to be **zero**.

$$H(s) = \frac{X(s)}{U(s)}$$

A system which has a single state variable and a single input is called a single input single output system (**SISO** for short). A system which has multiple state variables and inputs is called a multiple input multiple output system (**MIMO** for short).

From differential equation theory we know that the generalized input-output relation for a dynamical system can be written as:

$$\begin{aligned} a_n \frac{d^m x(t)}{dt^n} + a_{n-1} \frac{d^{n-1} x(t)}{dt^{n-1}} + \dots + a_1 \frac{dx(t)}{dt} + a_0 x(t) \\ = b_m \frac{d^m u(t)}{dt^m} + b_{m-1} \frac{d^{m-1} u(t)}{dt^{m-1}} + \dots + b_1 \frac{du(t)}{dt} + b_0 u(t) \end{aligned}$$

where a and b are constants for an LTI system. Taking Laplace transforms on both sides and assuming initial conditions are zero:

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

Rearranging yields the transfer function:

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

From this we see that the transfer function is a ratio of two polynomials in s . The powers of s will all be integral powers and a condition satisfied by physical systems (causality condition) is that $m \leq n$. In general, the transfer function is given by the generalized form:

$$H(s) = K \frac{(s - z_1)(s - z_2) \dots (s - z_{m-1})(s - z_m)}{(s - p_1)(s - p_2) \dots (s - p_{n-1})(s - p_n)} = \frac{b(s)}{a(s)} \quad (11.16)$$

where K is called the **gain factor**. The denominator, $a(s)$, of the transfer function is often called the **characteristic polynomial**. If we isolate the characteristic polynomial and set it to zero:

$$a(s) = 0$$

the equation is called the **characteristic equation**. In engineering, the roots of the characteristic equation are called the **poles**. That is values for p_i that set the characteristic equation to zero. The roots of the equation in the numerator, $b(s)$, are called the **zeros**. That is values of z_i that set the numerator to zero. For example, if p_2 is 5, then one of the poles must be $s = -5$ since $s - 5 = 0$.

The denominator, $a(s)$, of the transfer function is called the characteristic polynomial.

The equation, $a(s) = 0$ is called the **characteristic equation**.

The roots of the polynomial in the denominator of the transfer function are called the **poles**.

The roots of the equation in the numerator of the transfer function are called the **zeros**.

Since the Laplace transform of the impulse response $\delta(t)$ is 1, we can also state that the output function, $X(s)$ is equal to the transfer function when we apply an impulse function to the system:

$$X(s) = H(s) \quad \text{when the input is an impulse function}$$

The **order** of a transfer function is the order of the characteristic polynomial. That is the highest power of s in the denominator.

Let us examine the denominator of the transfer function more closely. Let us write the transfer function in the following form that assumes the input is an impulse:

$$H(s) = \frac{N(s)}{(s - p_1)(s - p_2) \dots (s - p_n)}$$

Carrying out a partial fraction expansion and assuming unique poles, yields:

$$H(s) = \frac{\alpha_1}{s - p_1} + \frac{\alpha_2}{s - p_2} + \dots + \frac{\alpha_n}{s - p_n}$$

Taking the inverse Laplace transform gives us:

$$h(t) = \alpha_1 e^{p_1 t} + \alpha_2 e^{p_2 t} + \dots + \alpha_n e^{p_n t}$$

We can see that the long term dynamics of the system is determined by the exponents in each of the terms. These exponents are the poles. If all poles are negative then the exponents will be negative and all terms decay to zero. If a single pole is positive then that term will dominate the dynamics of the system. Such systems are termed unstable. We will discuss stability in more detail in a later chapter but we can see already that the stability of a system is related to the signs of the poles.

For example, the system, where $b > 0$:

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

has a pole of $-b$. This means that the impulse response in the time domain has the form:

$$\alpha_1 e^{-bt}$$

that is the system decays in response to an impulse. It is very common to see the poles and zeros of a system plotted on a so-called **pole-zero plot**. It is common to plot the poles and zeros on a two dimensional graph where the x axis is the real part and the y axis the imaginary part. On the plot, poles are often indicated using crosses, \times , and zeros using small circles, \circ .

Standard Form

Engineers will often write the transfer function (11.16) in a standard form:

$$H(s) = K \frac{(1 + s/z_1)(1 + s/z_2) \dots}{(1 + s/p_1)(1 + s/p_2) \dots}$$

The reasons for this are two-fold. It is slightly easier to read off the zeros and poles since one doesn't have to do a mental sign change when reading them off in a traditional transfer function (11.16). More importantly in a later chapter (13) we will discuss Bode plots which describe the change in amplitude and phase as a function of frequency. Manually drawing the Bode plots is much easier if the transfer function is put into the standard form first.

Example 11.7

Put the following transfer function into its standard form:

$$H(s) = \frac{2s + 1}{s^2 + 5s + 6}$$

$$H(s) = \frac{2s + 1}{(2 + s)(3 + s)}$$

$$H(s) = \frac{2(s + 1/2)}{(2 + s)(3 + s)} = \frac{(1 + s/0.5)}{2(1 + s/2)3(1 + s/3)} = \frac{1}{6} \frac{\left(1 + \frac{s}{0.5}\right)}{\left(1 + \frac{s}{2}\right)\left(1 + \frac{s}{3}\right)}$$

Example 11.8

1. What is the order of the transfer function:

$$\frac{a}{s + b}$$

The order of the characteristic polynomial (denominator) is one.

2. What is the order for the transfer function:

$$H(s) = \frac{2s + 1}{s^2 + 5s + 6}$$

The order for this transfer function is two.

Example 11.9

1. Identify the gain, poles and zeroes for the first order transfer function:

$$\frac{a}{s + b}$$

The system has no zeros but has one pole at $s = -b$. The gain of the system is a .

2. What are the poles and zeros for the transfer function:

$$H(s) = \frac{2s + 1}{s^2 + 5s + 6}$$

The first thing is factor the transfer function to:

$$H(s) = \frac{1}{2} \frac{s + 2}{(s + 3)(s + 2)}$$

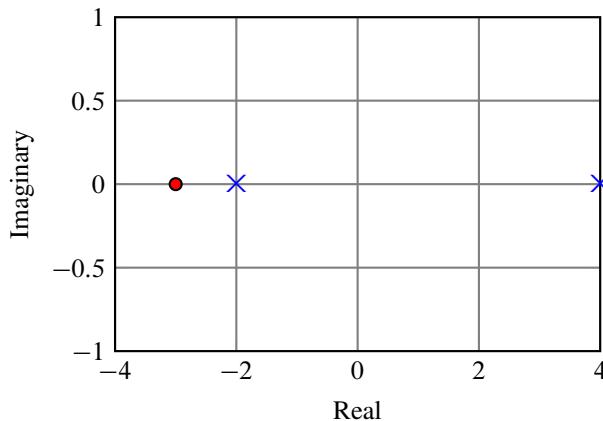
There is a single zero with a value $s = -2$ and two poles with values, $s = -3$ and $s = -2$.

Example 11.10

Plot the poles and zeros for the following transfer function:

$$H(s) = \frac{s + 3}{(s + 2)(s - 4)}$$

The system has one zero at -3 and two poles at -2 and 4 . A plot is shown below.

**Example 11.11**

Plot the poles and zeros for the following transfer function:

$$H(s) = \frac{(s^2 + 4s + 3)(s^2 - 2s + 2)}{(s^2 + 6s + 8)(s + 3)(s^2 + 9)}$$

First factorize the numerator and denominator to yield:

$$H(s) = \frac{(s + 1)(s + 3)(s - (1 + j))(s - (1 - j))}{(s + 2)(s + 4)(s + 3)(s + 3j)(s - 3j)}$$

Note that there are complex roots in both the numerator and denominator.

The system has zeros at $-1, -3, 1 + j, 1 - j$, and poles at $-2, -3, -4, -3j, 3j$. A plot is shown in Figure 11.4. Observe that the complex poles occur in conjugate pairs.

11.6 Steady State in the Laplace Domain

The Laplace transform offers a convenient way to compute the steady state using the final limit theorem,

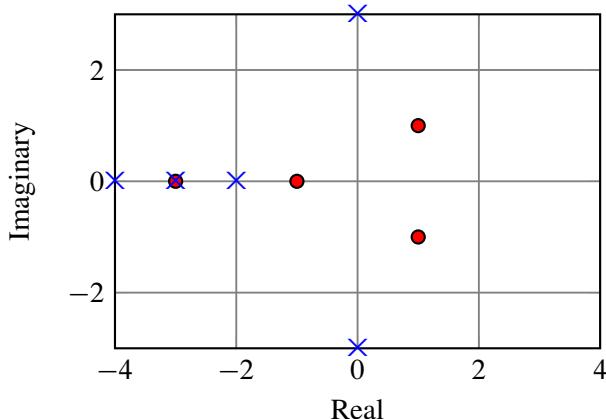


Figure 11.4 Pole-Zero Plot showing conjugate pair of zeros. Solid circle is a zero and cross is a pole.

Final-Value Theorem:

The final-value theorem is very useful for computing the behavior of the model at infinite time when the system might be at thermodynamic equilibrium or steady state. The theorem only applies to systems that are stable, that is where the poles are on the left-side of the pole-zero plot. The theorem is defined by:

$$\lim_{t \rightarrow \infty} f(t) = \lim_{s \rightarrow 0} sF(s)$$

Consider the system:

$$x_1 \xrightarrow{v_2} x_2 \xrightarrow{v_3}$$

where we assume that x_1 is transformed into x_2 and drains into the environment. The system therefore has two state variables, x_1 and x_2 . The system of differential equations for this model is assumed to be:

$$\begin{aligned}\frac{dx_1}{dt} &= -k_1 x_1 \\ \frac{dx_2}{dt} &= k_1 x_1 - k_2 x_2\end{aligned}$$

Taking the Laplace transforms of this yields the two equations:

$$F_1(s) = \frac{a}{s + k_1}$$

$$F_2(s) = \frac{k_1 a}{(s + k_1)(s + k_2)}$$

If we apply the final-value theorem we find that both x_1 and x_2 tend to zero. For example:

$$\lim_{t \rightarrow \infty} x(t) = \lim_{s \rightarrow 0} s \frac{a}{s + k_1} = 0$$

This is as expected, because all the mass will drain from the pathway leaving nothing in x_1 and x_2 ,

The following example illustrates how one must be careful in interpreting the result of the single-value theorem.

Example 11.12

Use the final value theorem to obtain the steady state value for the system:

$$X(s) = \frac{5}{s(s - 3)}$$

$$\lim_{t \rightarrow \infty} x(t) = \lim_{s \rightarrow 0} s \frac{5}{s(s - 3)} = -\frac{5}{3}$$

However the poles for $X(s)$ are 0 and 4 respectively indicating that the system will continue to expand in response to an impulse, i.e there is no steady final value. The final-value theorem is not applicable if one of the poles is positive.

Example 11.13

Use the final-value theorem to find the steady-state value for the system:

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

Note that the order of the denominator is larger than the order in the numerator, this means we can apply the final-value theorem. Is the system stable? Factor the denominator to:

$$s^2 + 3s^2 + 2s = s(s^2 + 3s + 2) = s(s + 2)(s + 1) = 0$$

All roots are negative or zero. The system is therefore stable.

$$\lim_{t \rightarrow \infty} x(t) = \lim_{s \rightarrow 0} s \frac{s^2 + 2s + 4}{s^3 + 3s^2 + 2s} = \lim_{s \rightarrow 0} s \frac{s^2 + 2s + 4}{s^2 + 3s + 2} = \frac{4}{2} = 2$$

Application

Let us look at the zero-state response of a first-order system and the application of the final-value theorem. A more detailed treatment of first-order systems will be given in Chapter 15 but consider the following differential equation that describes a single gene expressing a protein, x :

$$\frac{dx}{dt} = X_o k_1 - k_2 x$$

where X_o is the concentration of inducer, k_1 the rate constant for induction, and k_2 the degradation rate constant for the protein. We've seen this system before (See example 11.6) and derived the transfer function:

$$H(s) = \frac{k_1}{s + k_2} \quad (11.17)$$

We can consider the input X_o to be a step function in the sense that at time=0, we apply X_o . The response, $X(s)$ is then given by:

$$X(s) = U(s)H(s) = \frac{X_o}{s} \left(\frac{k_1}{s + k_2} \right)$$

The steady state for this system can be determined from the final-value theorem:

$$y_{ss} = \lim_{s \rightarrow 0} sX(s) = \lim_{s \rightarrow 0} \left(\frac{sX_o k_1}{s(s + k_2)} \right) = \frac{X_o k_1}{k_2}$$

The inverse Laplace transform of $X(s)$ is given by:

$$x(t) = \frac{X_o k_1}{k_2} (1 - e^{-k_2 t}) \quad (11.18)$$

We can confirm the steady state value of x by considering $t \rightarrow \infty$.

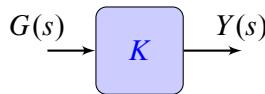
11.7 Block Diagrams

Large systems will often consist of many subcomponents interconnected in various ways. In control theory and particularly when systems are represented in the Laplace domain, there is a standard way to depict systems visually using block diagrams.

Block diagrams rely on the linearity rules that are obeyed by Laplace transforms and the assumption that the various subcomponents are only connected through their inputs and outputs and the input to one subcomponent does not affect the performance of the output it is connected to. From these rules, subsystems can be connected using four elementary rules. These include, gain, cascade (or series), parallel, and feedback. We will look at feedback more closely in a later chapter. Table 11.1 summarizes the most important of these rules.

The most basic block is the gain operation where a variable is scaled by a scalar quantity:

Type	s Domain
Gain	$X(s) = KG(s)$
Parallel	$X(s) = G_1(s) + G_2(s)$
Cascade (Series)	$X(s) = G_1(s)G_2(s)$
Feedback	$Y(s)/U(s) = \frac{G(s)}{1 \pm G(s)H(s)}$

Table 11.1 Table of Common Laplace Block Diagram Rules

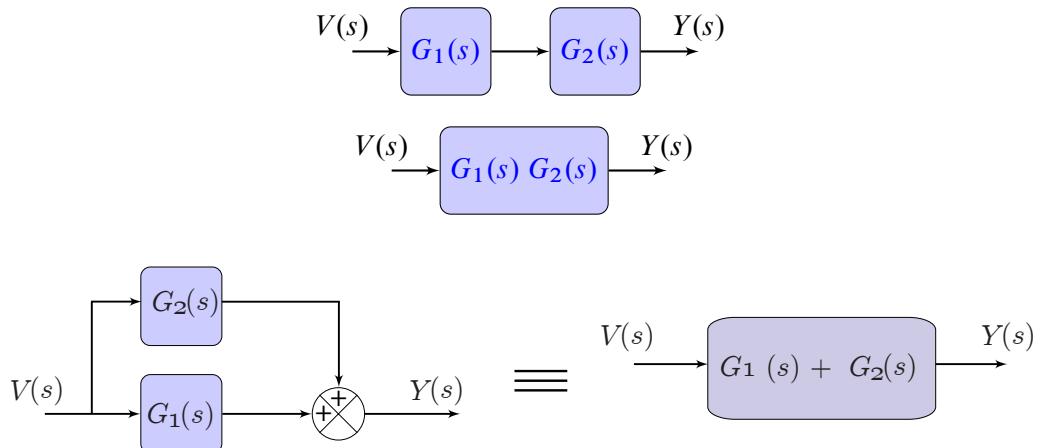
That is $Y(s) = KG(s)$. For two transfer functions in series, the overall transfer function is the product of the two. For example:

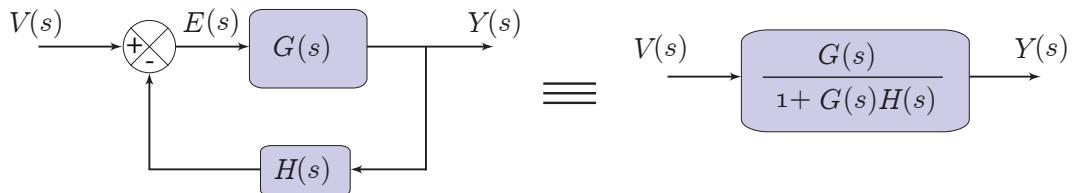
$$G_1(s) = \frac{1}{s+1} \quad \text{and} \quad G_2(s) = \frac{s+1}{s+2}$$

and $G_1(s)$ and $G_2(s)$ are in series, then

$$G(s) = G_1(s)G_2(s) = \frac{1}{s+2}$$

Combining blocks in series does assume that the output of the first subsystem is not affected by input to the second subsystem. This is a question of loading and in electronic systems is taken care by use of op amps that have high input and low output impedances.





One of the advantages of block diagrams is that it can give an engineer a quick impression of the overall system. This requires that the engineer is familiar with some of the common operations expressed in Laplace form. Three common transfer functions include the integrator, first-order and second-order functions. These are shown in the blocks below.

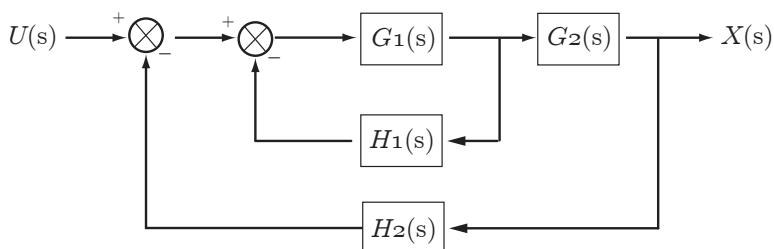
$$\begin{array}{c} \text{Block 1: } \frac{1}{s} \\ \text{Block 2: } \frac{K}{\tau s + 1} \\ \text{Block 3: } \frac{b_0}{s^2 + a_1 s + a_0} \end{array}$$

An engineer should be able to inspect a block diagram and identify the main components. In this book we will not use block diagrams to a great extent but can offer an alternative and obviously more visual way to represent transfer functions.

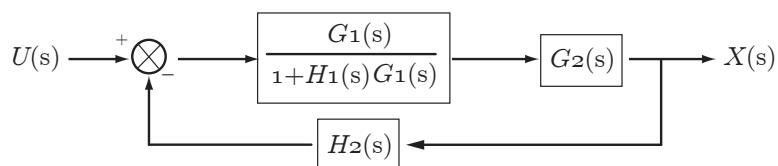
Example 11.14

Example 11.15

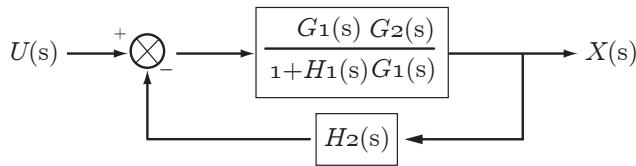
Reduce the following block diagram to a single block.



We can first simplify the inner loop using the closed loop feedback function (Figure 11.1 Feedback).



We combine the blocks $G_1(s)$ and $G_2(s)$ using the cascade rule:



We finally reduce the outer feedback loop:

$$U(s) \longrightarrow \boxed{\frac{G_1(s) G_2(s)}{1+H_1(s)G_1(s) + H_2(s) G_1(s) G_2(s)}} \longrightarrow X(s)$$

11.8 Summary of Transfer Functions

1. The transfer function is the ratio of the Laplace transforms of the output to the input. When defined this way the initial conditions *must* be at zero.
2. Using block diagram, transfer functions can be depicted graphically.
3. For a complex system the individual transfer functions can be combined using set rules to derive the overall transfer function. This assumes no loading issues between components.
4. Poles and zeros from the transfer function can be used to indicate overall behavior (see subsequent chapters)

Exercises

1. Define the impulse response for a system.
2. Define the transfer function for a system.
3. What is the relationship between the impulse response and the transfer function?
4. Find the convolution of the functions $u(t)$ and $u(t)$.
5. What is the equivalent mathematically of convolution in the Laplace domain?

6. Write down the general form the transfer function.
7. What are the poles and zeros of the transfer function?
8. State whether the following transfer functions are first or second order:

$$H(s) = \frac{1}{s + a}$$

$$H(s) = \frac{1}{s^2 + sa + b}$$

9. Using the final-value theorem determine the steady state values for x :

$$X(s) = \frac{4}{s(2s + 1)}$$

$$X(s) = \frac{6(s + 1)}{s(9s + 0.25)}$$

10. Give the system:

$$H(s) = \frac{2}{4s + 1}$$

what is the gain K and the time constant τ for this system?

11. What does the time constant represent in a first-order system?
12. It has been determined that the transfer function that determines the concentration of acetaminophen in patient's liver as a function of a constant infusion of acetaminophen is given by:

$$H(s) = \frac{0.03}{s + 0.06}$$

If the patient is suddenly given an infusion of 240 mg/l of acetaminophen what will be the steady state concentration of acetaminophen in the patient's liver? Compute the steady state concentration in two ways, by solving for the drug concentration at infinite time **and** by using the final-value theorem, show your working on both cases.

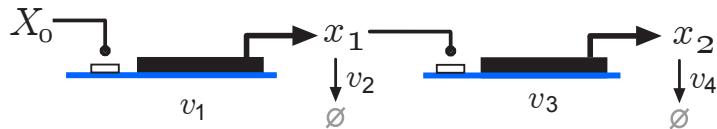
13. The above patient is now subjected to a new drug regime. This time the patient is administered acetaminophen by a slow infusion of increasing concentration over 120 seconds. At $t = 0$ the infusion concentration is zero. By 120 seconds the concentration of acetaminophen has reached 240 mg at which point the infusion concentration is fixed at 240 mg. Determine the time course of acetaminophen in the liver of the patient. Hint: Look up the Time Shifting property of the Laplace transform that is $\mathcal{L}[f(t - a)u(t - a)] = e^{-as}F(s)$

14. Determine $e^{\mathbf{A}t}$ for the system that has the following \mathbf{A} matrix:

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

Hint: Compute the inverse of the resolvent

15. Consider the following two gene cascade:



To make the calculations simpler we will assume simple mass-action kinetics for all steps. That is:

$$v_1 = k_1 X_o \quad v_2 = k_2 x_1 \quad v_3 = k_3 x_1 \quad v_4 = k_4 x_2$$

The input signal is supplied to inducer X_o .

- a) Find the transfer function $\mathbf{H}(s)$ for this system. Note there will be two transfer functions, one relating the input X_o to x_1 and another to x_2 .
- b) Obtain the time domain solutions to $x_1(t)$ and $x_2(t)$ assuming a simple step input in X_o ,
- c) Use the final-value theorem to determine the steady-state level of x_1 and x_2 .

12

Fourier Series

'There cannot be a language more universal and more simple, more free from errors and obscurities...more worthy to express the invariable relations of all natural things [than mathematics].'

— Joseph Fourier; *The Analytical Theory of Heat*, 1878

12.1 Introduction

In this chapter we will introduce the Fourier series in preparation for the next chapter on frequency response. Before proceeding let us state what the Fourier series is:

The Fourier series allows any periodic signal to be represented by a sum of sinusoids where the sinusoids are the same or multiples of the signal frequency.

The purpose of this chapter is not to provide an exhaustive or even moderately comprehensive review of the Fourier series but to provide sufficient details of the topic to enable the reader to gain a more intuitive understanding of the Laplace transform and a deeper appreciation of the frequency response which we will cover in the next chapter.

Let us review the main points regarding periodic functions that were introduced in Chapter 6. A continuous time dependent signal, $f(t)$ is called periodic with period T if:

$$f(t + T) = f(t)$$

This simply says that the signal $f(t)$ repeats itself every T units (Figure 12.1). For example, $\sin x$ is

periodic because $\sin(x + 2\pi) = \sin x$. The frequency of the signal, that is the number of complete cycles per unit time, is given by¹:

$$\omega = \frac{1}{T}$$

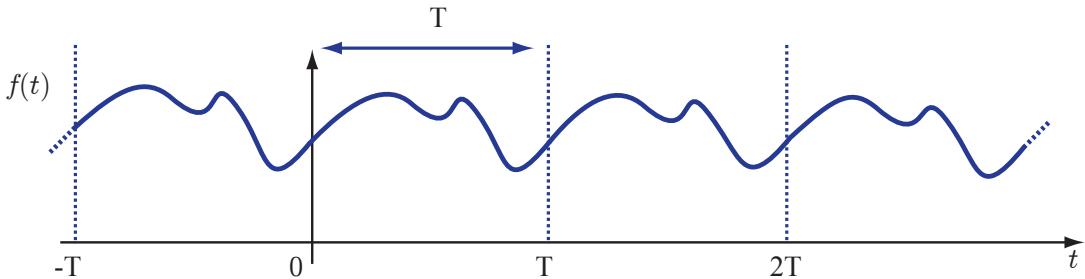


Figure 12.1 General periodic signal where $f(t) = f(t + T)$.

A particularly important class of periodic functions are the sinusinals:

$$x(t) = A \sin(\omega t + \theta)$$

where A is the amplitude, ω the angular frequency (radians per second) and θ the phase. If the frequency is expressed in Hertz, f , then $\omega = 2\pi f$.

Also of particular interest are the harmonics (See section 6.6). Any integer multiple of a frequency, ω is called a harmonic. For example, if the frequency of a periodic signal is ω , then its harmonics are $2\omega, 3\omega, \dots$, or in general $n\omega$. The base frequency, ω , is called the **fundamental frequency**. Terms such as $\cos(n\omega t)$ and $\sin(n\omega t)$ are called **harmonic components**.

The Fourier series is defined as the summation of harmonic components. There are different ways to represent this series but a common form is given below:

$$x(t) = \frac{a_0}{2} + a_1 \cos t + a_2 \cos 2t + a_3 \cos 3t + \dots \quad (12.1)$$

$$+ b_1 \sin t + b_2 \sin 2t + b_3 \sin 3t + \dots \quad (12.2)$$

This series is called the **Fourier Series** and the a_n and b_n terms the **Fourier Coefficients**. What is remarkable about this series is that we can use the Fourier series to approximate a huge variety of periodic functions, even periodic wave forms such as square or sawtooth waves.

The terms, a_i and b_i in the series determine the amplitude and phase of the harmonic terms. We can see this more clearly if we write out an individual summation term in the following way:

$$a_n \cos(n\omega t) + b_n \sin(n\omega t) = c_n \cos(n\omega t + \theta_n)$$

¹Angular frequency (ω): radians per second; Hertz (cycles per second) equals $\omega/(2\pi)$

where $c_n = \sqrt{a_n^2 + b_n^2}$ is the amplitude of the n^{th} component and $\theta_n = \tan^{-1}(b_n/a_n)$ the phase angle. With this change, the Fourier series can be reexpressed as ($c_o = a_o$):

$$f(t) = \frac{c_o}{2} + \sum_{n=1}^{\infty} c_n \cos(n\omega t + \theta_n)$$

This shows us more clearly that the Fourier series describes a signal via the summation of sinusoids with varying amplitudes and phases at different harmonics. An example of a Fourier series with only three terms is given by:

$$x(t) = \sin(\omega t) - \frac{1}{2} \sin(2\omega t) + \frac{1}{3} \sin(3\omega t)$$

where $b_1 = 1$, $b_2 = -1/2$ and $b_3 = 1/3$. One of the key questions in applying the Fourier series is to determine the values for a_n and b_n . Without going into the derivation (which can be found in most standard texts), it can be shown that:

$$a_n = \frac{2}{T} \int_{-T/2}^{T/2} f(t) \cos(n\omega t) dt, \quad n = 0, 1, 2, \dots$$

and for b_n :

$$b_n = \frac{2}{T} \int_{-T/2}^{T/2} f(t) \sin(n\omega t) dt, \quad n = 1, 2, \dots$$

where the integral is over a single cycle $-T/2$ to $T/2$ but any interval that spans the cycle would be appropriate. If the function, $f(t)$ is discontinuous with a series of continuous segments then the coefficients must be evaluated by summing the integrals of each continuous segment. This will be made clearer in the example below. Given a periodic function $f(t)$, the a_n and b_n coefficients can be determined using these formula and the Fourier series is completely specified for $f(t)$.

a_o may need some explanation. From the a_n integral at $n = 0$ we have:

$$a_o = \frac{2}{T} \int_{-T/2}^{T/2} f(t) \cos(0 \omega t) dt$$

However $\cos(0) = 1$, therefore:

$$a_o = \frac{2}{T} \int_{-T/2}^{T/2} f(t) dt$$

Seen this way, a_o is the average of $f(t)$ evaluated over one complete period, T . Dividing both sides by two yields the $a_o/2$ term in the Fourier series, equation (12.2).

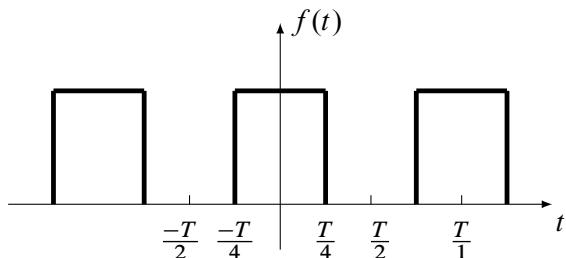


Figure 12.2 Square Waveform, with amplitude A and period T .

Example: Rectangular Wave

As an example let us consider the rectangular wave shown in Figure 12.2 which has a period of T .

The choice of what interval over which to integrate can make the coefficients easier to compute. For example, we could evaluate the Fourier coefficients by integrating between 0 and T . However note that over this period, the square wave has three continuous segments, one of which has a zero value. In contrast if we integrate between the interval $-T/2$ and $T/2$ then we only need to integrate a single segment between $-T/4$ and $T/4$ whose value is non-zero. For the other two segments, $-T/2$ to $-T/4$ and $T/4$ to $T/2$ the value of the function is zero. Why is this useful, because the integral over a segment whose functional value is zero is also zero. For this waveform we will determine the Fourier coefficients, a_n and b_n by integrating between $-T/2$ and $T/2$.

$$\begin{aligned} a_n &= \frac{2}{T} \int_{-T/2}^{T/2} f(t) \cos(n\omega t) dt \\ &= \frac{2}{T} \int_{-T/2}^{-T/4} f(t) \cos(n\omega t) dt + \frac{2}{T} \int_{-T/4}^{T/4} f(t) \cos(n\omega t) dt + \frac{2}{T} \int_{T/4}^{T/2} f(t) \cos(n\omega t) dt \end{aligned}$$

Note that the first and last integrals evaluate to zero, leaving:

$$\begin{aligned} &= \frac{2}{T} \int_{-T/4}^{T/4} A \cos(n\omega t) dt \\ &= \frac{2A}{n\omega T} \left[\sin(n\omega t) \right]_{-T/4}^{T/4} = \frac{2A}{n\pi} \sin\left(\frac{n\pi}{2}\right), n = 0, 1, 2, \dots \end{aligned}$$

This gives:

$$a_1 = \frac{2A}{\pi}, a_2 = -\frac{2A}{3\pi}, a_4 = 0, a_5 = \frac{2A}{5\pi}, \dots$$

For a_o , we have (note $\cos(0) = 1$):

$$a_o = \frac{2}{T} \int_{-T/2}^{T/2} f(t) \cos(0\omega t) dt = \frac{2}{T} \int_{-T/2}^{T/2} f(t) dt$$

The last term, except for the factor two, represents the average value for $f(t)$ over the given interval. For a square wave the average value is $A/2$. Therefore:

$$a_o = 2 \frac{A}{2} = A$$

For the b_n terms we have:

$$\begin{aligned} b_n &= \frac{2}{T} \int_{-T/2}^{T/2} f(t) \sin(n\omega t) dt \\ &= \frac{2}{T} \int_{-T/4}^{T/4} A \sin(n\omega t) dt = 0 \end{aligned}$$

That is all the b_n terms are zero. The Fourier series can now be written out as:

$$x(t) = \frac{A}{2} + \frac{2A}{\pi} \cos(\omega t) - \frac{2A}{3\pi} \cos(3\omega t) + \frac{2A}{5\pi} \cos(5\omega t) - \dots \quad (12.3)$$

Notice that the Fourier series is only made up of cosine terms with odd harmonics. If we were to shift the waveform so that the leading edge starts at zero (Figure 12.3) the Fourier series changes to:

$$x(t) = \frac{A}{2} + \frac{2A}{\pi} \sin(\omega t) - \frac{2A}{3\pi} \sin(3\omega t) + \frac{2A}{5\pi} \sin(5\omega t) - \dots$$

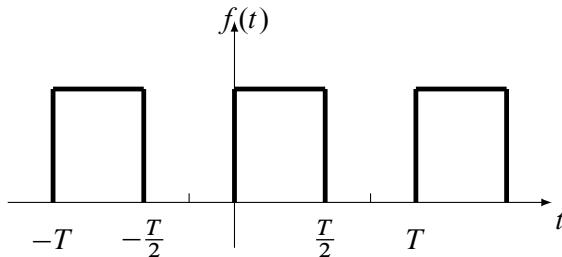


Figure 12.3 Square Waveform, with amplitude A and period T . Leading edge starts at zero.

This time the series is made up of only sine terms but the average term, $a_o/2$ and harmonics remain the same. Figure 12.4 plots four approximations to a saw tooth waveform and Figure 12.5 shows how well the sawtooth is approximated using 75 terms in the Fourier series.

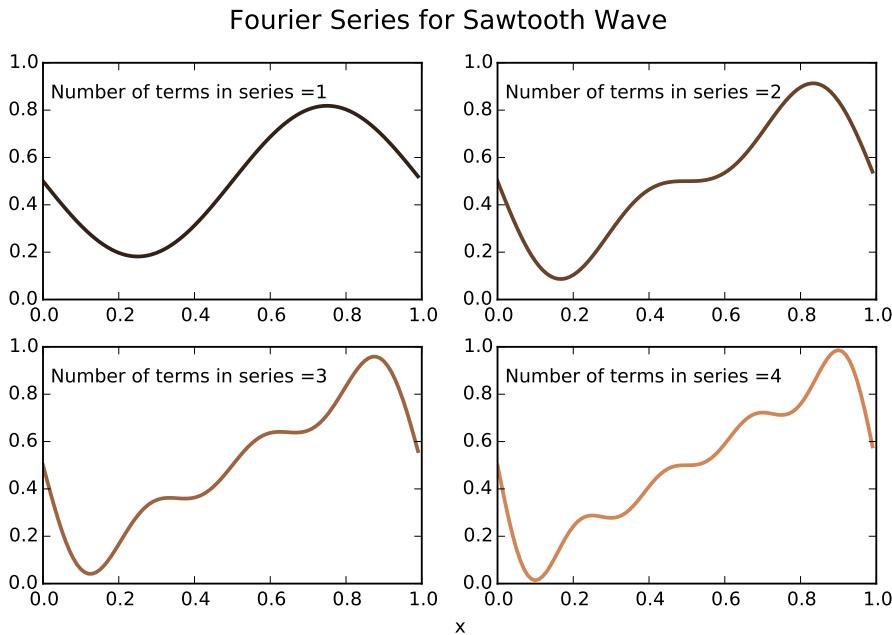


Figure 12.4 Fourier Series for approximating a saw tooth wave illustrating the first four terms in the series.

Odd and even functions

The presence or absence of sine or cosine terms in a Fourier series is related to the symmetry of the signal about the zero axis. A waveform is symmetrical about the y axis if $f(t) = f(-t)$ for all t . That is $f(-t)$ is an exact mirror image of $f(t)$. Such functions are called **even** functions. The Fourier series for even functions will also contain even functions. Cosine is an even function. Since the waveform in Figure 12.2 is even, the Fourier series only contains cosine terms.

Functions which are asymmetrical with respect to the y axis have the property $-f(t) = f(-t)$ for all time and are called **odd** functions. The Fourier series for odd functions will contain odd functions. Sine is an odd function. Since the waveform in Figure 12.3 is even, the Fourier series only contains sine terms.

Knowing whether a periodic function is even or odd can help double-check that we have the correct Fourier series.

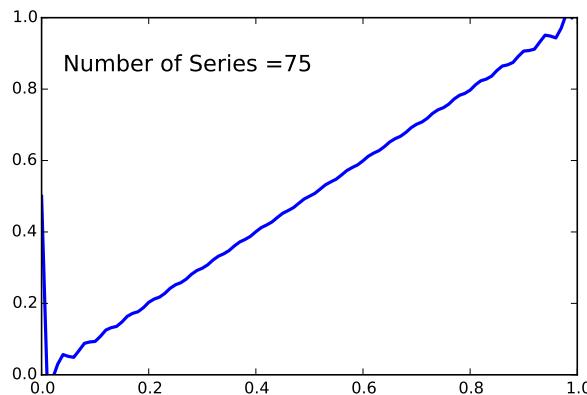


Figure 12.5 Saw tooth approximation using the first 75 terms in the Fourier Series.

12.2 Dirichlet Conditions

One thing we haven't mentioned up to now is what kinds of functions have a Fourier series? The conditions that a function $f(t)$ must meet so that it can be described using a Fourier series are known as the Dirichlet conditions. First we define a periodic interval $-T/2$ to $T/2$ of period T and then list the conditions below:

1.

$$\int_{-T/2}^{T/2} |f(t)| dt < \infty$$

This ensures that the function is integrable over the given interval and in turn ensures that the Fourier coefficients are also finite. Another way to look at it is that the function has a finite average value over the interval.

2. $f(t)$ has a finite number of discontinuities over the interval.
3. $f(t)$ has a finite number of positive and negative maxima and minima

These conditions have more practical value when we start to consider non-periodic functions.

12.3 Exponential Form

The components of the Fourier series can also be expressed in exponential form using the standard relations:

$$\sin(n\omega t) = \frac{1}{2i} (e^{in\omega t} - e^{-in\omega t})$$

$$\cos(n\omega t) = \frac{1}{2} (e^{in\omega t} + e^{-in\omega t})$$

Using complex exponentials the Fourier series can be written in the form:

$$f(t) = \sum_{n=-\infty}^{\infty} \alpha_n e^{in\omega t}$$

where the α_n terms are given by:

$$\alpha_n = \frac{1}{T} \int_{-T/2}^{T/2} f(t) e^{-in\omega t} dt$$

12.4 Frequency Spectrum

One of the most important aspects of the Fourier series is the power spectrum. Previously the Fourier series was expressed in the form:

$$f(t) = c_o + \sum_{n=1}^{\infty} c_n \cos(n\omega t + \theta_n)$$

where $c_o = a_o/2$. A plot of c_n for the different harmonic frequencies $n\omega$ is called the **amplitude spectrum** of $f(t)$. We can also plot the **phase spectrum** by plotting θ_n versus $n\omega$. The two plots taken together are called the **Fourier Spectrum**. Figure 12.6 shows the amplitude spectrum for the square wave. Each line corresponds to a component in the Fourier series, the length of which gives the relative strength that each term contributes. A periodic signal can therefore be described in terms of a set of harmonic frequencies of given strengths.

12.5 APeriodic Signals: Fourier Integral and Transform

Up to now we have focused on the ability of the Fourier series to describe periodic signals. Is it possible to generalize the Fourier series to both periodic and non-periodic signals?

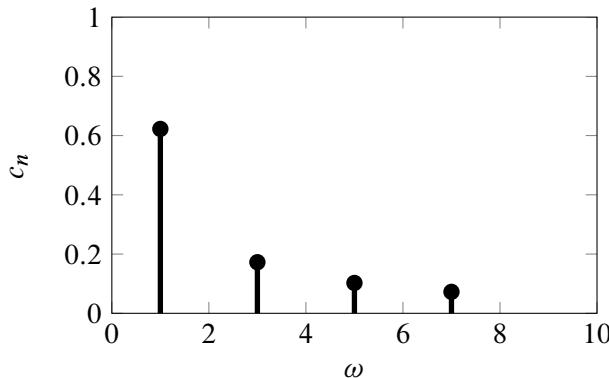


Figure 12.6 Line Spectrum for the rectangular wave 12.3 at $\omega, 3\omega, 5\omega$, etc.

In the line spectrum in Figure 12.6 we note that each line is constructed at $n\omega$ where n is the frequency at n times the fundamental frequency so that the spacing between lines is ω wide. If we were to increase T , the period of the wave form, the frequency ω would naturally decrease and the lines in the line spectrum would become closer. For example if we doubled the period to $2T$ the fundamental frequency is halved to $\omega/2$. If we continue to increase the period indefinitely, ω will tend to approach zero and the lines become so dense that the spectrum approaches a smooth curve, resulting in a continuous spectrum. As T tends to infinity we are left with a single cycle of the waveform spanning the whole of time, that is the signal is *no longer periodic*. To accommodate these changes let us modify the notation we use. In particular:

Discrete	Continuous
$n\omega$	$\rightarrow \omega$
ω	$\rightarrow \Delta\omega$
Period, T	$\rightarrow 2\pi/\Delta\omega$

$n\omega$ can take on any frequency in the continuous spectrum and we replace $n\omega$ with a simple ω . The original spacing between the lines, ω now becomes a small value $\Delta\omega$ with the period, T changing accordingly. Recall that the exponential form of the Fourier series was given by (updated to accommodate the new notation):

$$f(t) = \sum_{\omega=-\infty}^{\infty} \alpha_n e^{i\omega t}$$

where α_n is:

$$\alpha_n = \frac{\Delta\omega}{2\pi} \int_{-T/2}^{T/2} f(t) e^{-i\omega t} dt$$

Substituting α_n into $f(t)$ yields:

$$f(t) = \frac{1}{2\pi} \left[\sum_{\omega=-\infty}^{\infty} \int_{-T/2}^{T/2} f(t) e^{-i\omega t} dt \right] e^{i\omega t} \Delta\omega$$

As $T \rightarrow \infty$, $\Delta\omega \rightarrow d\omega$ and $\sum \rightarrow \int$, the above equation becomes:

$$f(t) = \frac{1}{2\pi} \int_{-\infty}^{\infty} \left[\int_{-\infty}^{\infty} f(t) e^{-i\omega t} dt \right] e^{i\omega t} d\omega$$

The quantity in the square brackets is a function of ω . Let us call this function $F(\omega)$. With this change in notation the above equation can be rewritten as:

Fourier Integral:

$$f(t) = \frac{1}{2\pi} \int_{-\infty}^{\infty} F(\omega) e^{-i\omega t} d\omega$$

Given our definition for $F(\omega)$ we can write this as:

Fourier Transform:

$$F(\omega) = \int_{-\infty}^{\infty} f(t) e^{-i\omega t} dt \quad (12.4)$$

The Fourier transform is often represented symbolically as:

$$F(\omega) = \mathcal{F}[f(t)]$$

and its inverse by:

$$f(t) = \mathcal{F}^{-1}[F(\omega)]$$

Both the time domain description, $f(t)$ and the frequency description $F(\omega)$ are equivalent in the sense that they contain the same information about a signal but using alternative methods.

It should be noted that $F(\omega)$ is a complex function of ω . As a result $F(\omega)$ contains both amplitude and phase information about the original signal. It is possible and often desirable to plot both the amplitude and phase of $F(\omega)$ as a function of frequency. Together the two plots describe the *frequency spectrum* of the signal. One unusual aspect about these plots is that the Fourier transform integral goes from $-\infty$ to $+\infty$, that is a full spectrum plot includes both negative and positive frequency which begs the question what are negative frequencies? The Fourier Transform given previously is called

the two-sided spectrum and is generally symmetric about zero particularly for real-world signals such as audio signals. The concept of negative frequencies are therefore not of great concern.

As with the Fourier series there are conditions on whether the function, $f(t)$ can be integrated. These conditions are similar to the Dirichlet conditions but where the range of integration is now infinite. We can summarize these conditions below:

1. It must be true that:

$$\int_{-\infty}^{\infty} |f(t)| dt < \infty$$

2. It must have at most a finite number of discontinuities in the range $-\infty < t < \infty$.
3. It must have at most a finite number of maxima and minima in the range $-\infty < t < \infty$.

The first condition is significant and has practical implications for the types of signal that have a Fourier transform. In particular signals such as steps and ramps which have positive values for all time so that the integral will not be $< \infty$ do not have Fourier transforms. This limits the applicability of the Fourier transform and as we'll see in the next section, opens the way forward to the development of the less restrictive Laplace transform.

In summary we can treat the Fourier transform as something that maps a time domain function into a series of frequencies that includes both their amplitudes and phases. The inverse transforms maps the frequency information back into the time domain.

12.6 Return to the Laplace Transform

In Chapter 10 we introduced the Laplace transform as a means to solve ordinary differential equations. Here we will revisit the Laplace transform from the perspective of the Fourier transform. It should be recalled that according to the Dirichlet conditions, the Fourier transform is limited to a subset of functions, particularly those functions who values never reach zero. Such functions includes steps and ramps. The first thing we could do to improve the Fourier transform is to ensure that it converges in these situations. The easiest way to do this is to replace $f(t)$ with a new signal such as:

$$F(\omega) = \int_{-\infty}^{\infty} g(t)e^{-i\omega t}$$

where $g(t)$ is

$$g(t) = f(t)e^{-\sigma t}$$

The idea is that as $f(t)$ is integrated, the term $e^{-\sigma t}$ will limit the value of the integral so that it remains finite. We can combine the two exponentials to give:

$$G(\omega) = \int_{-\infty}^{\infty} f(t) e^{-(\sigma+i\omega)t} dt$$

Let us replace the term $(\sigma + i\omega)$ with s so that:

$$G(s) = \int_{-\infty}^{\infty} f(t) e^{-st} dt$$

This final expression is of course the *bilateral Laplace transform*, see section 10.2. The one issue with this definition is that for negative t the convergence factor $e^{-\sigma t}$ will have the opposite effect and could cause the integral to diverge without limit as $t \rightarrow -\infty$. To avoid this we add the restriction that either $f(t)$ must not be time-extended to the left of the origin or we simply set the lower limit of the integration to zero. With this last change we arrive as the standard unilateral Laplace transform:

$$F(s) = \int_0^{\infty} f(t) e^{-st} dt$$

where $s = \sigma + i\omega$.

12.7 Fourier Transform of Discrete Data

There are many applications in engineering where there is the need to decompose signals from experiments into their component frequencies and amplitudes. A particular common application is the need to remove either contaminating signals, such as the 60 Hz signal that corresponds to the AC frequency or simply noise which is present at high frequencies. As we have seen the Fourier transform describes a signal in terms of its component amplitude and phase sinusoids. However the Fourier transform applies to continuous signals when in practice most signals we measure are sampled at discrete intervals. We therefore don't have an analytical function, $f(t)$, that we can integrate to obtain the Fourier transform 12.4, instead we have raw sampled data points. The answer to obtaining the Fourier transform for real sampled data is to use a discrete approximation of the integral 12.4, called the Discrete Fourier Transform or DFT. Recall that the Fourier transform of a continuous signal, $x(t)$ is given by:

$$F(s) = \int_0^{\infty} f(t) e^{-st} dt$$

This equation can be approximated using a discrete sum called the discrete Fourier transform:

$$X_k = \sum_{n=0}^{N-1} x_n e^{-j2\pi k n / N}, \quad k = 0, 1, \dots, N-1$$

where N is the total number of samples taken between 0 and T . As with the continuous transform, the discrete transform computes a set of complex numbers which correspond to the amplitude and phase components present in the original sampled signal. In practice the use of the discrete form is not very efficient and instead the fast Fourier transform (FFT) method developed by Cooley and Tukey in 1965 is used. The development of fast algorithms for computing Fourier transforms revolutionized signal processing especially when coupled with dedicated digital signaling processors (DSP) or cost effective standard microprocessors. Software for computing the Fourier transform via FFT is now widespread. To give an idea of the efficacy gains we can compare the number of multiplications necessary when using the standard DFT and the FFT. For 512 sampled points ($N = 512$), a DFT analysis requires 1,046,528 while the FFT requires only 9,216 multiplications.

Python provides FFT support via the `scipy` library. Two examples are given in listings 12.1 and 12.2. In the first data is sampled from a sum of two sinusoids with different amplitudes and phases. In the second, the same data is sampled except that now each data point has Gaussian noise added with a mean of zero and standard deviation of one. The two figures `x` and `y` illustrate the amplitudes when we apply the FFT to the data. In the first case (ref), the FFT clearly identifies two peaks corresponding to the two sinusoids at 50 and 80 Hz. What is more remarkable is the FFT still manages to pick out the underlying sinusoids even in the presence of significant noise. In such cases filters can be applied to remove all components other than 50 and 80 Hz in order to ‘clean’ up the data.

```
from scipy.fftpack import fft
import matplotlib.pyplot as plt
import numpy as np
import random

# Number of sample points
N = 600
# sample spacing
T = 1.0 / 800.0
x = np.linspace(0.0, N*T, N)
y = np.sin(50.0 * 2.0*np.pi*x) + 0.5*np.sin(80.0 * 2.0*np.pi*x)
plt.plot (x, y)
plt.show()

yf = fft(y)
xf = np.linspace(0.0, 1.0/(2.0*T), N/2)
plt.plot(xf, 2.0/N * np.abs(yf[0:N/2]))
plt.xlabel('Freq (Hz)')
plt.ylabel('Magnitude')
plt.grid()
plt.show()
```

Listing 12.1 Applying FFT in Python using data generated from the sum of two sinusoids.

```

from scipy.fftpack import fft
import matplotlib.pyplot as plt
import numpy as np
import random

# Number of sample points
N = 600
# sample spacing
T = 1.0 / 800.0
x = np.linspace(0.0, N*T, N)
y = np.sin(50.0 * 2.0*np.pi*x) + 0.5*np.sin(80.0 * 2.0*np.pi*x)
# Add some noise to the data
y = y + np.random.normal (0.0, 1, N)
plt.plot (x, y)
plt.show()

yf = fft(y)
xf = np.linspace(0.0, 1.0/(2.0*T), N/2)
plt.plot(xf, 2.0/N * np.abs(yf[0:N/2]))
plt.xlabel('Freq (Hz)')
plt.ylabel('Magnitude')
plt.grid()
plt.show()

```

Listing 12.2 Using FFT in Python to identify the underlying signal with Noisy Data.

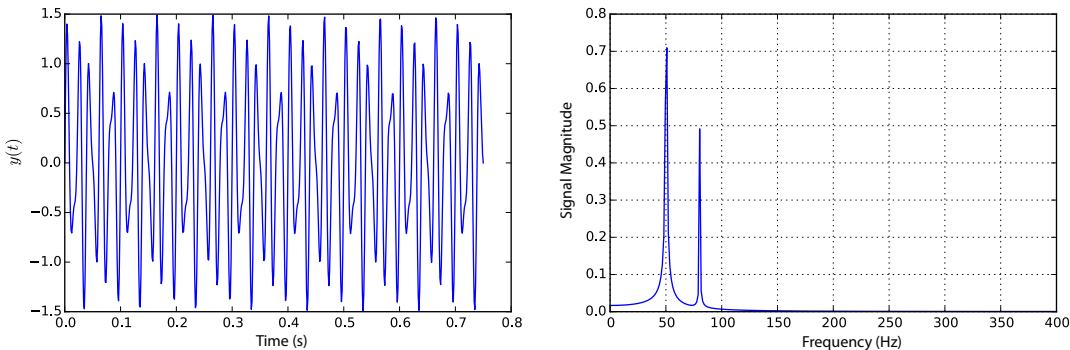


Figure 12.7 FFT applied with sampled noise free signal. Left panel: Sampled data; Right panel: FFT showing component signals at 50 and 80 Hz.

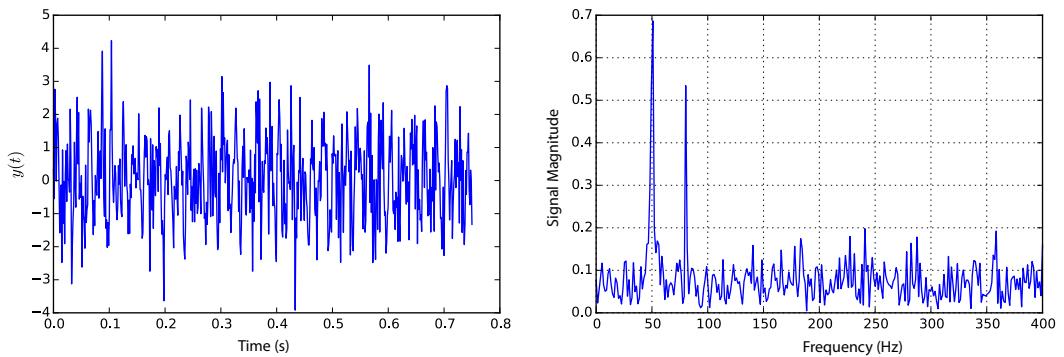


Figure 12.8 FFT applied with sampled noisy signal. Left panel: Noisy sampled data; Right panel: Even with significant noise FFT can still pick the signals at 50 and 80 Hz

13

Frequency Response

“Sines and cosines, how with thou respond?”

– *Anonymous*

13.1 Introduction

In a previous chapter we saw that a general sinusoidal signal is given by the relation:

$$y(t) = A \sin(\omega t + \theta)$$

where A is the amplitude, ω the frequency and θ the phase shift of the sinusoidal signal. Of particular interest is that multiplication of a sinusoidal by a constant will only change its amplitude, addition of two sinusoids, assuming they have the same frequency, will only change the amplitude and phase. Likewise integration only changes the phase and amplitude. None of these operations change the frequency component. The common factor is that these are linear operations. This means that if we input a sinusoidal signal into a *linear system*, the sinusoidal signal that is output will only see changes to its amplitude and phase. This is a key point in the analysis of linear systems. Of particular interest is how the steady state responds to a sinusoidal input, termed the **sinusoidal steady state response**.

13.2 Linear Systems and Sinusoids

Let us illustrate the effect of a linear system on a sinusoidal signal by way of an example. Consider the first-order transfer function:

$$H(s) = \frac{k_1}{s + k_2}$$

Let us apply a sinusoidal signal, $\cos(\omega t)$ to the system.¹ To obtain the output response we multiply the Laplace transform of $\cos(\omega t)$, by the transfer function $H(s)$:

$$X(s) = \frac{k_1}{s + k_2} \left(\frac{s}{s^2 + \omega^2} \right)$$

Multiplying this out yields:

$$X(s) = \frac{k_1 s}{(k_2 + s)(s^2 + \omega^2)}$$

The inverse Laplace transform for this is (Computed using Mathematica):

$$x(t) = \frac{k_1}{(k_1^2 + \omega^2)} (k_2 \cos(t\omega) + \omega \sin(t\omega) - e^{-k_2 t} k_2)$$

If we look at the system after the transients have decayed ($t \rightarrow \infty$) we obtain:

$$x(t) = \frac{k_1}{(k_1^2 + \omega^2)} (\omega \sin(t\omega) + k_2 \cos(t\omega))$$

The sum of cosine and sine terms can be reexpressed as a sine and a phase change using the trigonometry identity:

$$a \cos(x) + b \sin(x) = \sqrt{a^2 + b^2} \sin(x - \tan^{-1}\left(\frac{b}{a}\right))$$

Using this identity we can reexpress the response, $x(t)$ as:

$$x(t) = \frac{k_1}{(k_1^2 + \omega^2)} \sqrt{k_1^2 + \omega^2} \sin\left(\omega t - \tan^{-1}\left(\frac{\omega}{k_2}\right)\right)$$

Note that $\sqrt{x}/x = 1/\sqrt{x}$, therefore:

$$x(t) = \frac{k_1}{\sqrt{k_1^2 + \omega^2}} \sin\left(\omega t - \tan^{-1}\left(\frac{\omega}{k_2}\right)\right)$$

¹We chose a cosine input rather than a sine input because the algebra is slightly simpler.

From this we conclude that the amplitude changes to:

$$A = \frac{k_1}{\sqrt{k_1^2 + \omega^2}}$$

and the phase by:

$$\varphi = -\tan^{-1} \frac{\omega}{k_2}$$

These results correspond exactly to the results (13.13 and 13.12) we obtained from doing the same analysis on the first-order differential equation. However both approaches are a little tedious to use. Fortunately there is a much simpler way to obtain the amplitude and phase change for a given transfer function which we will consider in the next section.

Generalizing

For this proof we will need the following results:

Expressing a complex number $G(jw)$ in polar form (See F.17):

$$G(jw) = |G(jw)| e^{j\omega}$$

The complex exponential in terms of phase and frequency:

$$\cos(\omega t + \theta) = \frac{e^{j(\omega t + \theta)} + e^{-j(\omega t + \theta)}}{2}$$

Finally (See F.16):

$$\arg(G(-jw)) = -\arg(G(jw))$$

We can generalize the previous result as follows. Assume a system has a transfer function $H(s)$. Let us apply a sinusoidal input to the system, that is²:

$$r(t) = A \cos \omega t$$

The transform of this is:

$$R(s) = \frac{As}{s^2 + \omega^2}$$

The output, $X(s)$, of this system is then given by:

$$X(s) = H(s)R(s) = H(s) \frac{As}{(s - j\omega)(s + j\omega)}$$

²We chose a cosine input rather than a sine input because the algebra is slightly simpler.

Expanding the relationship using partial fractions:

$$X(s) = \frac{k_1}{s - j\omega} + \frac{k_2}{s + j\omega} + X_d(s)$$

$X_d(s)$ is the collection of all terms that originate in the denominator $H(s)$. If we assume that the system is stable, the transient terms in $X_d(s)$ will decay to zero as the system reaches steady state. This leaves the input terms in k_1 and k_2 . At steady state k_1 and k_2 can be evaluated as follows:

$$G(s) \frac{As}{(s - j\omega)(s + j\omega)} = \frac{k_1}{s - j\omega} + \frac{k_2}{s + j\omega}$$

Multiplying both sides by $(s - j\omega)(s + j\omega)$ yields:

$$G(s)As = k_1(s + j\omega) + k_2(s - j\omega)$$

If we set $s = j\omega$, this eliminates the k_2 term. We then solve for k_1 :

$$k_1 = \frac{G(j\omega)Aj\omega}{2j\omega} = \frac{1}{2}AG(j\omega)$$

We can do the same for k_2 to obtain:

$$k_2 = \frac{1}{2}AG(-j\omega)$$

Taking the inverse transform of $X(s)$ at steady state yields:

$$x(t) = k_1 e^{-j\omega t} + k_2 e^{j\omega t}$$

Referring to (F.17) we can represent a complex number such as $G(j\omega)$ in polar form using:

$$\begin{aligned} G(j\omega) &= |G(j\omega)|e^{j\theta} \\ G(-j\omega) &= |G(j\omega)|e^{-j\theta} \end{aligned}$$

Substituting k_1 and k_2 :

$$\begin{aligned} k_1 &= \frac{1}{2}A |G(-j\omega)| e^{-j\theta} \\ k_2 &= \frac{1}{2}A |G(j\omega)| e^{j\theta} \end{aligned}$$

Substituting these into the time domain solution, $x(t)$:

$$\begin{aligned} x(t) &= -\frac{1}{2}A|G(-j\omega)|e^{-j\theta}e^{-j\omega t} + \frac{1}{2}A|G(j\theta)|e^{j\theta}e^{j\omega t} \\ &= A|G(j\omega)| \frac{e^{j\theta}e^{j\omega t} + e^{-j\theta}e^{-j\omega t}}{2} \\ &= A|G(j\omega)| \frac{e^{j(\theta+\omega t)} + e^{-j(\theta+\omega t)}}{2} \\ &= A|G(j\omega)| \cos(\omega t + \theta) \end{aligned}$$

This result shows that a sinusoidal input yields an output that is also sinusoidal. The output sinusoidal has the same frequency as the input signal but with different phase, θ , and amplitude, $A|G(j\omega)|$.

13.3 Frequency Response

A sinusoidal signal that is applied to some specific input of a system will propagate itself throughout the system to varying degrees. For example we could apply a sinusoidal input to the input concentration, X_o of a simple linear chain of reactions. We could then examine the response in each of the species, S_1, S_2, \dots . Upon application of a sinusoidal signal, the system would initially go through a transition as it accommodates the input, eventually reaching some steady periodic behavior. Most likely the pathway species would display some kind of sinusoidal behavior in response to the varying input. Those species nearest the source, X_o , are likely to respond more fully to the input, while species further away are likely to be affected to a lesser degree. The frequency response of a system is the steady state response of a system to a sinusoidal input. In practice a range of frequencies is applied to the system and the response recorded.

Of interest is what can happen to a sinusoidal system as it traverses a pathway. In the most general case, anything. For example the signal could change amplitude, phase or frequency and it could even change its sinusoidal shape. However if our system is linear, then only the phase and amplitude will change, the frequency will be unaffected. This makes the analysis much simpler but also forces us to either study pure linear systems or turn nonlinear systems into linear ones.

In summary, the **frequency response** of a system concerns itself with how the **amplitude** and **phase** of an input signal of a given frequency propagates to all the state variables in a **linear system**. Often a graph is made over a range of frequencies, such a plot is called a **Bode Plot**.

If a system is linear, time invariant and stable, the response of the system to a sinusoidal input is a sinusoidal output with the same frequency as the input but differing in amplitude and phase.



Figure 13.1 Effect of a linear system on a sinusoidal signal.

We will discover in the next few sections and chapters that the frequency response is of considerable importance in understanding how systems operate. In particular, the frequency response has the following useful applications:

- Provides an indication of how close to instability we might be and how to move closer or further away.
- Provides an approach for measuring the degree of modularity in a system.
- Allows us to explain the onset of oscillations in a feedback circuit.
- Allows us to understand the properties of negative feedback in more detail.
- Gives us the machinery to reconstruct the internal structure of a system from the input/output response.
- Relates the DC component of the frequency domain to the existing field of metabolic control analysis.

13.4 Laplace and Fourier Transforms

In the previous sections we used two different approaches to determine how a sinusoidal signal is changed by a linear system. For larger systems, these approaches becomes too unwieldy. Instead, we can get the same information by using the unilateral Fourier Transform (See section 12.5). This transform takes a sinusoidal input signal, applies it to a linear system and computes the resulting phase and amplitude change at the given frequency of the sinusoidal input. The transform can be applied at all frequencies so that the complete frequency response can be computed indicating how the system alters sinusoidal signals at different frequencies. Analytically, the unilateral Fourier Transform is given by:

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

If we compare this to the Laplace transform:

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

we see that they are very similar. s in the Laplace transform is usually a complex number $\sigma + j\omega$. In the case of the unilateral Fourier transform, $s = j\omega$. To compute the Fourier transform we can therefore take the Laplace transform and substitute s with $j\omega$. In Fourier analysis, harmonic sine and cosines are multiplied into the system function, $f(t)$ and then integrated. The act of integration picks out the strength of the response to a given frequency.

Let us use the Fourier transform to obtain the amplitude and phase change for a general linear first-order differential equation:

$$\frac{dx}{dt} + ax = f(t)$$

Taking Laplace transforms on both sides yields:

$$sX(s) + aX(s) = \mathcal{L}(f(t))$$

so that

$$X(s) = \frac{\mathcal{L}(f(t))}{s + a}$$

The transfer function of the system, $H(s)$, is however the ratio of $\mathcal{L}(\text{output}/(\text{input}))$ so that

$$H(s) = \frac{X(s)}{\mathcal{L}(f(t))} = \frac{1}{s + a}$$

To obtain the frequency response we set $s = j\omega$:

$$H(j\omega) = \frac{1}{j\omega + a}$$

Given the complex number in standard form: $a + jb$. The amplitude of a complex number is:

$$A = \sqrt{a^2 + b^2}$$

The phase of a complex number is:

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

See Appendix F.6 for details.

Because $H(j\omega)$ is a complex number we can compute both the amplitude and phase shift. First we must put the equation into the standard form:

$$H(j\omega) = \frac{1}{a + j\omega} \frac{(a - j\omega)}{(a - j\omega)} = \frac{a}{a^2 + \omega^2} - \frac{j\omega}{a^2 + \omega^2}$$

From this we can compute the amplitude change:

$$A = \sqrt{\frac{a^2}{(a^2 + \omega^2)^2} + \frac{\omega^2}{(a^2 + \omega^2)^2}} = \sqrt{\frac{1}{a^2 + \omega^2}} = \frac{1}{\sqrt{a^2 + \omega^2}}$$

The phase shift can be computed using $\tan^{-1}(b/a)$ so that

$$\alpha = \tan^{-1}\left(\frac{-\omega}{a}\right) = -\tan^{-1}\left(\frac{\omega}{a}\right)$$

Note the sign change, this is because the vector that represents the complex number extend below the x axis so that the resulting angle is negative. These results are identical to the results obtained in the previous section when the differential equation was integrated directly. It is common to use the following symbols for the amplitude and phase:

Amplitude:	$ H(j\omega) $
Phase:	$\angle H(j\omega)$

13.5 Bode Plots

We saw in the previous sections that the amplitude and phase of a sinusoidal signal change when propagating through a linear system while the frequency remains unaltered. Studying how the phase and amplitude change can be very useful. We can plot the amplitude and phase as a function of the frequency of the sinusoidal signal. Such plots are called **Bode plots**.

A Bode plot is the frequency response of the system. It indicates how the phase and amplitude of a state variable relative to an input sine wave of a linear system varies as a function of frequency.

To determine the frequency response of a transfer function $H(s)$ we replace s with $j\omega$ and compute the magnitude, $|H(j\omega)|$ and phase as a function of ω . The computed amplitude from the transfer function will represent a gain, that is a ratio of output to input, for example a voltage gain, V_2/V_1 or protein gain, P_2/P_1 . For convenience gains are best expressed using decibels.

Decibels

Decibels (dB) are commonly used to measure the ratio of two intensities on a logarithmic scale. Thus the gain of a system, as described above, is conveniently expressed in decibels. The use of logarithms enables one to express very small and very large numbers using a single measure. We can define the gain in decibels as:

$$H_{\text{dB}}(j\omega) = 20 \log_{10} |H(j\omega)|$$

The decibel (dB) is the magnitude in log space of the ratio of two measurements. In practice the measurements are often electrical power but can be others such as sound. In terms of power the decibel is defined by:

$$\text{dB} = 10 \log_{10} \left(\frac{P_1}{P_o} \right)$$

when referring to other quantities such as voltage it is common to use the fact that power is proportional to the square of the voltage such that:

$$\text{dB} = 10 \log_{10} \left(\frac{V_1^2}{V_o^2} \right) = 20 \log_{10} \left(\frac{V_1}{V_o} \right)$$

In this book we will use the later definition where we multiply by 20 rather than 10. Because a decibel is related to a ratio, if a given decibel is negative then the signal is attenuated; if the ratio is positive then the signal is amplified. If the db is one, then there is no change between the input and output signal.

Common Values:

$$20 \log_{10} \sqrt{2} = 3 \text{ dB}$$

$$20 \log_{10} \frac{1}{\sqrt{2}} = -3 \text{ dB}$$

$$20 \log_{10} \sqrt{10} = 10 \text{ dB}$$

$$20 \log_{10} 10 = 20 \text{ dB}$$

$$20 \log_{10} 100 = 40 \text{ dB}$$

Rules of thumb:

1. As a number doubles, the decibel value increases by 6 dB
2. As a number increases by a factor of 10, the decibel value increases by 20 dB.

As mentioned already, one reason for using dB is that we can express small and large numbers using one measure. But also there is the convenience that if multiple systems are connected together each with its own gain, the overall gain of the system is the sum of the dB gains. For example if an amplifier is made up of three subamplifiers with gains of 16 dB, 24 dB and 45 dB respectively, then the overall gain is 85 dB. This property becomes very useful when we discuss Bode plots.

Example 13.1

An operational amplifier has a gain of 50 dB. What is the voltage gain, $|V_2/V_1|$? If $H_{dB} = 20 \log_{10}$ gain then

$$\text{gain} = 10^{0.05H_{dB}}$$

Therefore gain = 316. That means if the input voltage to the op amp is 1 mV, then the output voltage will be 316 mV.

The Bode Plot

The Bode plot is made up of two separate plots. One plot displays how the amplitude or gain changes as a function of frequency and the second how phase changes as a function of frequency. The x -axis in both plots marks the input frequency on a log scale which can either be expressed in radians/sec or in Hz. The amplitude plot is plotted using decibels (dB) - essentially a log scale - and the phase is plotted in degrees on a linear scale.

Given the transfer function $H(j\omega)$ we can in principle express it in terms of its real and imaginary parts:

$$H(j\omega) = \text{Re}(\omega) + j\text{Im}(\omega)$$

The amplitude for the Bode plot can then be computed using the following relation, see (F.14):

$$20 \log |H(j\omega)| = 20 \log \sqrt{\text{Re}[H(j\omega)]^2 + \text{Im}[H(j\omega)]^2} \quad (13.1)$$

and the phase from the relation, see (F.15):

$$\text{Arg}(H(j\omega)) = \tan^{-1} \frac{\text{Im}[H(j\omega)]}{\text{Re}[H(j\omega)]} \quad (13.2)$$

First-Order System

Although equations (13.1) and (F.15) are on the whole impractical, we will illustrate their use in a simple example. Let us consider the first-order system shown in Figure 13.2.

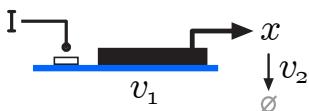


Figure 13.2 One Gene Network

As before we will assume that the expression rate for x is controlled by a factor X_o and that the step is governed by the rate law $v_1 = k_1 X_o$. We will assume the degradation step is governed by $v_2 = k_2 x$.

The input to the system will be via X_o and we will ask the question what is the frequency response of x with respect to a sinusoidal input on X_o . As shown before the transfer function, $H(s)$ for this combination of input and output is given by:

$$H(s) = \frac{k_1}{s + k_2} \quad (13.3)$$

To carry out a frequency response we first exchange s with $j\omega$ to give:

$$H(j\omega) = \frac{k_1}{j\omega + k_2}$$

To obtain the amplitude and phase shift we convert the transfer function into the standard form to separate the real and imaginary parts by multiplying top and bottom by the conjugate complex number:

$$\frac{k_1}{j\omega + k_2} \frac{k_2 - j\omega}{k_2 - j\omega} = \frac{k_1 k_2 - k_1 j\omega}{k_2^2 + \omega^2} = \frac{k_1 k_2}{k_2^2 + \omega^2} - \frac{k_1 j\omega}{k_2^2 + \omega^2}$$

From this we can determine the amplitude ($\sqrt{a^2 + b^2}$) given by:

$$\text{Amplitude} = |H(j\omega)| = \sqrt{\frac{k_1^2}{k_2^2 + \omega^2}} = \frac{k_1}{\sqrt{k_2^2 + \omega^2}} \quad (13.4)$$

Likewise the phase change can be computed from $\tan^{-1}(b/a)$:

$$\text{Phase} = \tan^{-1}\left(\frac{-\omega}{k_2}\right) = -\tan^{-1}\left(\frac{\omega}{k_2}\right)$$

Amplitude

Figure 15.7 shows a plot of the amplitude as a function of the frequency. Although it looks as if the plot starts at zero amplitude, it doesn't. At zero frequency ($\omega = 0$) the y -axis intercept is k_1/k_2 (13.4). At high frequency the amplitude tends to zero. The other thing to note is that the units of frequency are in rad/sec. If frequency should be expressed in Hertz the simple conversion is: 1 Hz and 2π radians both represent a full cycle, that is:

$$1 \text{ Hertz} \equiv 2\pi \text{ radians per second}$$

Therefore, as we've seen before, f Hertz in radians is:

$$\text{radians} = 2\pi f$$

To express the amplitude plot using Hertz we only need to express the radian frequency in terms of Hertz. This can be done by substituting $2\pi f$, where f is the frequency in units of Hertz wherever ω appears in the equation. For example:

$$\text{Amplitude} = |H(j\omega)| = \sqrt{\frac{k_1^2}{k_2^2 + \omega^2}} = \frac{k_1}{\sqrt{k_2^2 + (2\pi f)^2}}$$

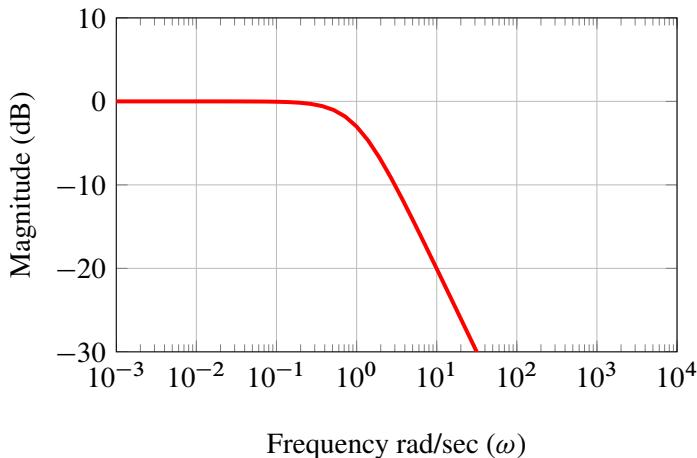


Figure 13.3 Bode Plot: Magnitude (dB) as a function of radians per second

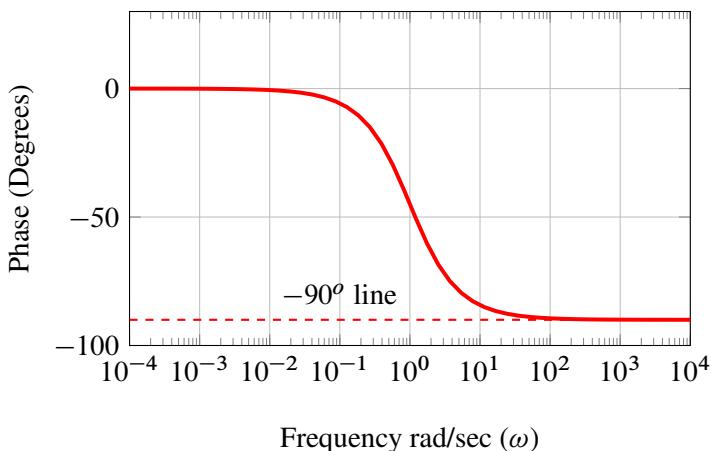


Figure 13.4 Bode Plot: Phase Shift

For example assume $k_1 = 0.1$ and $k_2 = 0.01$. At a frequency of 0.05 rad/sec, the amplitude in dB will equal:

$$\text{Gain} = \left(\frac{0.1}{\sqrt{0.01^2 + 0.05^2}} \right) = 1.961$$

$$\text{db Gain} = 20 \log_{10} (1.961) = 5.85$$

This means that if the input amplitude = 1.0, then the amplitude of the sine wave at the state variable will be 1.961 bigger.

To confirm this prediction the following script (Listing 13.1) can be run and the amplitude of the input and output curves compared with the Bode plot prediction. For example at 0.05 rad/sec we expect roughly a ratio of 2 to 1 for the output (x) to input (X_o), corresponding to roughly 6 decibels.

```
import math
import tellurium as te, roadrunner
from scipy import signal
import numpy as np, matplotlib.pyplot as plt

r = te.loada("""
    Xo := sin (0.05*time) + 2 # Inject sine wave into system at 0.05 rad/sec
    XoShift := Xo + 18; # Make Xo and x comparable when plotting
    $Xo -> x; k1*Xo;
    x ->; k2*x;

    x = 18; k1 = 0.1;k2 = 0.01
""")

m = r.simulate (0, 400, 900, ['time', 'XoShift', 'x'])

# Plot Bode plot, specify transfer function here
num = [r.k1]
den = [1, r.k2]
s1 = signal.lti(num, den)
w, mag, phase = signal.bode(s1, np.arange(0.0001, 1, 0.001).tolist())
plt.figure(figsize=(10,5))
plt.semilogx (w, mag, color="blue", linewidth="1")
plt.xlabel ("Frequency")
plt.ylabel ("Magnitude")
```

Listing 13.1 Amplitude Frequency Response for a First-Order System.

Phase Shift

The change in phase shift is shown in Figure 18.1 and starts at zero and goes negative, meaning the phase lags. At high frequencies the phase shift tends to -90° . The way to understand this is to look at the phase expression and realize that the smaller k_2 the more likely the phase shift will be -90° . If k_2 is very small then we can assume that the rate of degradation is very small, this means that the change in x is dominated by the input sine wave. The maximum rate of increase in x is when the sine wave is at its maximum peak. As the input sine wave decreases the rate of increase in x slows until the input sine wave crosses the steady state level of x . Once the input sine wave reaches the steady state level, the level of x also peaks. Thus the input sine wave and the concentration of x will be -90° out of phase with the concentration of x lagging. The frequency point at which the phase reaches -90° will depend on the k_2 value. Figure 13.5 illustrates the phase shift argument.

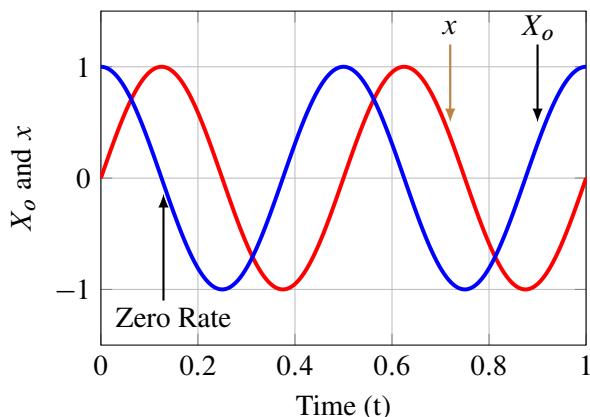


Figure 13.5 -90° phase shift at high frequencies or low degradation rates for the model shown in Figure 13.2. Note that the concentration of x peaks when the input rate is at zero.

Bandwidth

In electrical engineering, a common measure of an amplifier's performance is its bandwidth. This is the point in the frequency response where the power output from the amplifier has halved. In terms of voltage this represents a drop to 70.7% of the peak value. For example if the peak voltage is 5 volts and it drops to 70.7% of its value we get a voltage of 3.535 volts. Recall that power is given by $P = V^2/R$. Assuming a constant resistance of 1 ohm, the power output at 5 volts will be 25 watts, whereas at 3.535 volts, the power output is 12.5 watts, that is the power has exactly halved. In terms

of decibels, a reduction of an output to 70.7% of its peak value is given by:

$$\begin{aligned}\text{change} &= 20 \log_{10} \left[\frac{0.707}{1} \right] \text{dB} \\ &= -20 \log_{10}(1.414) \\ &= -3 \text{ dB}\end{aligned}$$

The last calculation allows us to express the bandwidth in terms of the drop in decibels, namely 3 dB. That is the frequency at which the amplitude has dropped by 3 dB is the bandwidth.

The **bandwidth** of a system is the frequency at which the gain drops below the 3 dB peak. 3 dB is when the signal is $1/\sqrt{2}$ of the maximum signal amplitude (about 70% of the signal strength).

Generalized Amplitude and Phase Solutions

In Chapter 11 a generalized transfer function was given, see equation 11.16, and repeated here:

$$H(jw) = K \frac{(s - z_1)(s - z_2) \dots (s - z_{m-1})(s - z_m)}{(s - p_1)(s - p_2) \dots (s - p_{n-1})(s - p_n)} \quad (13.5)$$

We can rewrite this equation more succinctly as:

$$H(j\omega) = K \frac{\prod_{i=1}^m (w + z_i)}{\prod_{i=1}^n (w + p_i)}$$

The amplitude, $|H(j\omega)|$, can be written as:

$$|H(j\omega)| = K \frac{\prod_{i=1}^m |\omega + z_i|}{\prod_{i=1}^n |\omega + p_i|}$$

which in turn as can be written as:

$$|H(j\omega)| = K \frac{\prod_{i=1}^m \sqrt{w^2 + z_i^2}}{\prod_{i=1}^n \sqrt{w^2 + p_i^2}}$$

The phase can be similarly generalized:

$$\angle H(j\omega) = \sum_{i=1}^m \angle(j\omega - z_i) - \sum_{i=1}^n \angle(j\omega - p_i) = \sum_{i=1}^m \tan^{-1} \left(\frac{\omega}{z_i} \right) - \sum_{i=1}^n \tan^{-1} \left(\frac{\omega}{p_i} \right)$$

These equations can be illustrated with some examples.

First-Order:

$$H(s) = \frac{K}{s + a}$$

$$|H(jw)| = \frac{K}{\sqrt{\omega^2 + a^2}} \quad \angle H(jw) = -\tan^{-1} \left(\frac{\omega}{a} \right)$$

Second order:

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

$$|H(jw)| = \frac{K}{\sqrt{\omega^2 + a^2} \sqrt{\omega^2 + b^2}} \quad \angle H(jw) = -\tan^{-1} \left(\frac{\omega}{a} \right) - \tan^{-1} \left(\frac{\omega}{b} \right)$$

Drawing Bode Plots

The use of the equations (13.1) and (13.2) is not very practical for plotting Bode plots. Instead there are two other approaches, one is to use software, this makes it very easy to draw Bode plots. The second and more traditional approach is to draw them by hand using a series of straight-line approximations. One of the reasons for the popularity of Bode plots was that once mastered, it was relatively easy to manually draw the frequency response. In this section will be briefly cover the art of manually drawing Bode plots and in a later section show how modern computer software makes the task much quicker and easier.

The key to manually drawing Bode plots is to recognize that transfer functions are a ratio of product terms. In log space these turn into a series of additions and subtractions³. For example consider the transfer function:

$$H(s) = K \frac{s}{1 + s\tau}$$

We first replace s with $j\omega$:

$$H(j\omega) = K \frac{j\omega}{1 + j\omega\tau}$$

Taking the magnitude on both sides:

$$|H(j\omega)| = |K| \frac{|j\omega|}{|1 + j\omega\tau|}$$

³Recall: $\log(AB) = \log(A) + \log(B)$ and $\log(A/B) = \log(A) - \log(B)$

and applying $20 \log_{10}$ yields

$$20 \log_{10}(|H(i\omega)|) = 20 \log_{10}|K| + 20 \log_{10}|j\omega| - 20 \log_{10}(|1 + j\omega\tau|) \quad (13.6)$$

Looked at in this way we can break a transfer function down into a series of more manageable building blocks. In a previous section 11.16 we introduced the generalized transfer function, namely:

$$G(s) = K_1 \frac{(s - z_1)(s - z_2)\dots}{(s - p_1)(s - p_2)\dots}$$

However in practice a better generalized version is one that includes pure integrators in the equation but ensuring that one or more poles are zero, that is:

$$G(s) = K_1 \frac{(s + z_1)(s + z_2)\dots}{s^r(s + p_1)(s + p_2)\dots}$$

Some authors go further and put the transfer function into a standard form (substituting s with $j\omega$):

$$G(j\omega) = K \frac{(1 + j\omega/z_1)(1 + j\omega/z_2)\dots}{(j\omega)^r(1 + j\omega/p_1)(1 + j\omega/p_2)\dots}$$

Note the $(j\omega)^r$ in the denominator which accounts for r possible integrators in the system. If we express this form in decibels we obtain (dropping the log10 subscript):

$$\begin{aligned} 20 \log |G(j\omega)| &= 20 \log |K| + 20 \log |1 + j\omega/z_1| + \dots \\ &\quad - 20 \log |(j\omega)^r| - 20 \log |1 + j\omega/p_1| \dots \end{aligned}$$

If we know how each of these building blocks contributes to the Bode plot we can construct the Bode plot by summing up each term. Five common building blocks are shown in Table 13.1 together with the corresponding amplitude and phase.

$H(j\omega)$	Building Block	Amplitude	Phase
K		$20 \log_{10} K $	$\tan^{-1}(0/K) = 0^\circ, -180^\circ$ if $K < 0$
$j\omega\tau$		$20 \log_{10} \omega\tau$	90°
$(j\omega\tau)^{-1}$		$-20 \log_{10} \omega\tau$	-90°
$1 + j\omega\tau$		$20 \log_{10} \sqrt{1 + (\omega\tau)^2}$	
$(1 + j\omega\tau)^{-1}$		$20 \log_{10} \left(\sqrt{1 + (\omega\tau)^2} \right)^{-1}$	

Table 13.1 Basic building blocks for constructing a Bode plot.

Each of these building block displays a particular line pattern in a Bode plot. For example, the simplest rule is the gain building block, K . Every transfer function has a gain so every Bode plot must have a gain component equal to $20 \log_{10} |K|$, that is a horizontal line.

Returning to the example in equation (13.6) we can see that this transfer function has three building blocks, a gain term K , a $j\omega$ term and a $1 + j\omega\tau$ term. The strategy is to take the curve described by each building block and sum them to obtain the final Bode plot.

We know that the gain term is a horizontal line, what about the other building blocks? Let us consider each in turn.

$j\omega\tau$ - derivative

The amplitude for $j\omega\tau$ is

$$|H(j\omega)| = 20 \log_{10} \omega\tau = 20 \log_{10} \omega + 20 \log_{10} \tau$$

This equation describes a straight line (Figure 13.6) that climbs at a rate of 20 db per decade. It intercepts the y axis at zero when:

$$0 = 20 \log_{10} \omega + 20 \log_{10} \tau$$

$$\log_{10} \omega = -\log_{10} \tau$$

which is the same as saying:

$$\omega = \frac{1}{\tau}$$

That is the line crosses the x axis at $1/\tau$. At the crossing point we will designate the frequency as ω_c . Therefore if $\tau = 1$, then the line crosses at 0. If $\tau = 10$, then the line crosses at 0.1. Its useful to express this relationship in the form:

$$\tau = \frac{1}{\omega_c}$$

This will be important when we consider the term $(1 + j\omega\tau)$ when it is written in the standard form $1 + j\omega/p$, where $1/p = \omega_c$.

The phase can be determined by realizing that $j\omega\tau$ has no real part hence on an argand diagram, $j\omega\tau$ points directly north, meaning that the angle to the x axis is a constant 90° .

$1/(j\omega\tau)$ - integrator

This is the same as $j\omega\tau$ but where the sign is swapped, $|H(j\omega)| = -20 \log_{10} \omega\tau$. This means the slope is $-20 \log_{10} \omega\tau$. The x axis intercept is still τ . The phase is still a constant but now negative at -90° .

In general, for factors such as $(j\omega)^{\pm m}$ the slope is $\pm 20m$ dB per decade and a constant phase shift of $\pm m 90^\circ$.

$(1 - j\omega\tau)^{\pm m}$

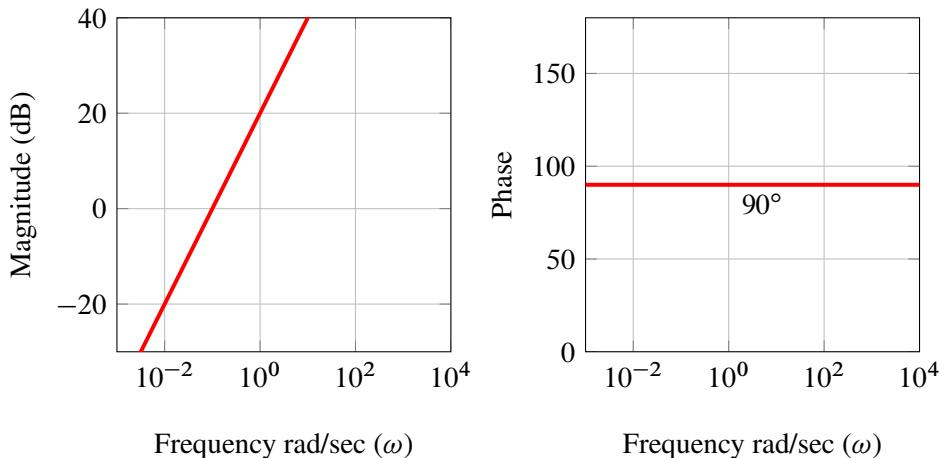


Figure 13.6 Bode Plot of $H(j\omega) = 20 \log_{10} j\omega\tau$

Probably the most common term we find in transfer functions is of the form $(1 + j\omega\tau)^{-1}$. The log magnitude for this is given by:

$$20 \log_{10} |1 + j\omega\tau|^{-1} = -20 \log_{10} \sqrt{1 + \omega^2\tau^2}$$

For very small values of ω such that $\omega\tau \ll 2$, we have:

$$-20 \log_{10} \sqrt{1 + \omega^2\tau^2} = -20 \log_{10} 1 = 0 \text{ dB}$$

This means that at small frequencies the Bode plot will be on the 0 dB line. For large values of $\omega\tau$, that is $\omega\tau \gg 1$:

$$20 \log_{10} |1 + j\omega\tau|^{-1} \approx -20 \log_{10} \omega\tau$$

This corresponds to the pure integrator we looked at previously and represents a line with a slope of $-20 \log_{10} \omega\tau$ and a x intercept at $1/\tau$. Together we have two straight lines which meet at $1/\tau$. Note the first line at $\omega\tau \ll 1$ is on the x axis. The meeting point, $1/\tau$, is called the **corner frequency**, ω_c , see Figure 13.7.

The corner frequency, also known as the cutoff frequency, break frequency or bandwidth is given by:

$$\omega_c = \frac{1}{\tau}$$

The term $(1 + j\omega\tau)$ has a similar profile that includes the same corner frequency, except that the slope for the second asymptote is positive rather than negative.

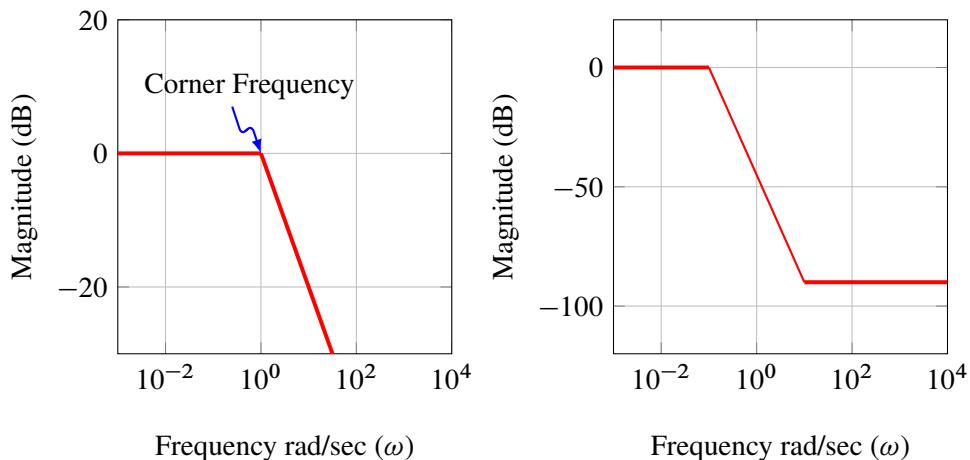


Figure 13.7 Approximate Bode Plot of $H(j\omega) = (1 + j\omega\tau)^{-1}$. The plot assumes τ equals one, hence the corner frequency, 1τ is at one.

The phase shift for $(1 - j\omega\tau)^{\pm m}$ also changes as a function of frequency.

$$\text{Phase} = \tan^{-1}\left(\frac{-\omega\tau}{1}\right) = -\tan^{-1}\left(\frac{\omega\tau}{1}\right)$$

At low $j\omega\tau$, the phase shift approaches 0° since:

$$\text{Phase} = -\tan^{-1}(\ll 1) \approx 0$$

When $\omega\tau = 1$ the phase shift becomes:

$$\text{Phase} = -\tan^{-1}(1) = -45^\circ$$

which corresponds to the corner frequency, ω_c . Finally when $\omega\tau \gg 1$:

$$\text{Phase} = -\tan^{-1}(\gg 1) \approx -90^\circ$$

A rule of thumb is that the two asymptotes will meet at 0.1τ and 10τ .

Example 13.2

Draw the Bode plot for the following transfer function:

$$G(s) = \frac{10^3(s + 20)}{(s + 20)(s + 200)} \quad (13.7)$$

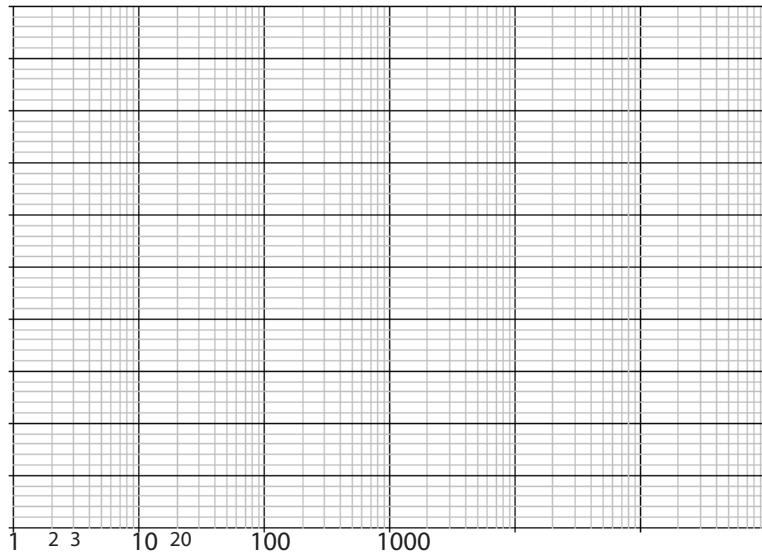


Figure 13.8 Semilog Graph Paper.

The first task is to put the transfer function into its standard form:

$$G(s) = 10^3 \frac{20 \left(1 + \frac{s}{20}\right)}{10 \left(1 + \frac{s}{10}\right) 200 \left(1 + \frac{s}{100}\right)} = \frac{10 \left(1 + \frac{s}{20}\right)}{\left(1 + \frac{s}{10}\right) \left(1 + \frac{s}{200}\right)}$$

Substituting $j\omega$:

$$G(j\omega) = \frac{10 \left(1 + \frac{j\omega}{20}\right)}{\left(1 + \frac{j\omega}{10}\right) \left(1 + \frac{j\omega}{200}\right)}$$

When manually drawing a Bode plot it is very common to use semilog graph paper 13.8, that is the x axis of the graph paper scales with the log.

Inspection of the transfer function reveals a gain and three terms of the form $(1 + j\omega\tau)^{\pm 1}$. Recall that for a term such as $(1 + j\omega\tau)^{\pm 1}$, the meeting points of the two asymptotes is called the corner frequency, ω_c and equals $1/\tau$. Hence a term such as:

$$\left(1 + \frac{j\omega}{10}\right)$$

where $\tau = 1/10$, the corner frequency is therefore $\omega_c = 1/\tau = 1/(1/10) = 10$. This explains why engineers like transfer functions expressed in the standard form because the corner frequencies can be read directly from the equation. We can summarize the various terms as follows:

- A constant gain $K = 10$

- A single zero with a corner frequency of $\omega_c = 20$
- Two poles with corner frequencies at $\omega_c = 10$ and $\omega_c = 200$

Magnitude Plot

To plot the magnitude plot we proceed as follows. For the gain of 10, the units in dB will be $20 \log 10 = 20\text{dB}$. We therefore draw a horizontal line at 20 dB.

For the single corner frequency in the numerator, we draw a 20 dB/decade line at $\omega_c = 20$. Finally for the two corner frequencies in the numerator, we draw two lines with slope -20 dB/decade and drawn at $\omega_c = 10$ and $\omega_c = 200$. These lines are drawn in Figure 13.9. Once the lines are drawn it is then a matter to sum the lines together. The fact that these lines have slopes of 20 dB makes it easier to sum the slopes. Thus the two lines at the zero $\omega_c = 10$ and pole $\omega_c = 10$ have opposite slopes and cancel each other out, resulting in part of the Bode plot being a horizontal line. One point to bear in mind, these are approximate plots, exact plots can be obtained using software.

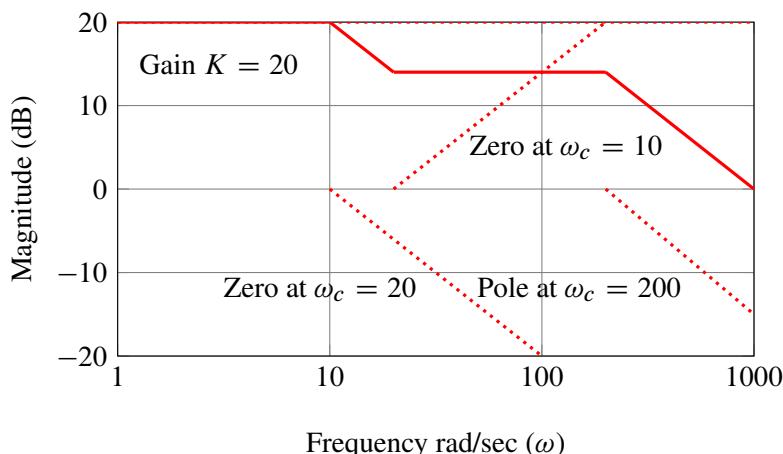


Figure 13.9 Manual drawing of a Magnitude Bode Plot. Dotted lines mark individual terms, solid lines represent the summation of the dotted lines.

Phase Plot

Let us now consider the phase plot. For the gain of 10, the phase will be 0° so we draw a horizontal line. For the remaining terms we use the rule of thumb that lines join at 0.1τ and 10τ where τ is the corner frequency.

For example, the corner frequency in the zero is 20. Since $\omega\tau > 1$ this yields a 90° phase shift. We therefore join a line from 0° at $\omega = 0.1 \times 20 = 2$ to $\omega = 10 \times 20 = 200$ which will be 200° .

For the first pole, again $\omega\tau > 1$ so that the phase shift is -90° , negative because the pole term is $1 = \omega\tau)^{-1}$. We join a line from 0° between $\omega = 0.1 \times 10 = 1$ and $\omega = 10 \times 10 = 100$ reaching -90° .

For the second pole, again $\omega\tau > 1$ and the phase shift is -90° . We join a line from 0° between $\omega = 0.1 \times 200 = 20$ and $\omega = 10 \times 200 = 2000$ reaching -90° .

We draw each of these lines on a semilog graph, see Figure 13.10.

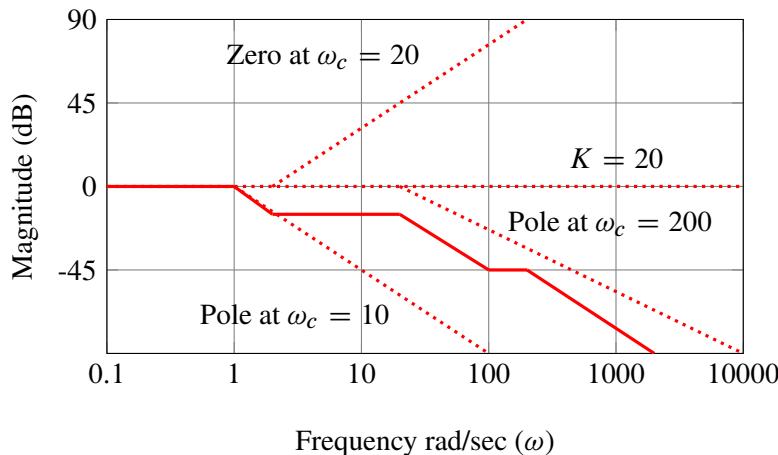


Figure 13.10 Manual drawing of a Phase Bode Plot. Dotted lines mark individual terms, solid lines represent the summation of the dotted lines.

We will not pursue the drawing of Bode plots manually further because most plots today are done using software. However it is useful to know how Bode plots come about based on the transfer function.

13.6 Bode Plots using Python

There are a number of software applications that support the plotting of Bode plots. Here we will use Python. There are at least two generic control theory packages for Python, these include a Matlab compatibility library (<http://python-control.sourceforge.net/manual/index.html>) and the signals library that is part of SciPy (<http://www.scipy.org/>). Of the two, the SciPy library is the easiest to install novices and also comes ready installed with many Python distributions such as Python(x,y) (<https://code.google.com/p/pythonxy/>), WinPython (<http://winpython.github.io/>) or Tellurium (<http://tellurium.analogmachine.org/>).

SciPy is a large collection of numerical analysis routines that uses the NumPy array library. Within SciPi we will focus on the Signal Processing library, `scipy.signal`. Assuming SciPy has been installed, the signal library can be installed using the command:

```
import scipy.signal as signal
```

Listing 13.2 Loading the signal library.

The first thing we must learn is how to create a LTI model using the signal library. There are a number of ways to do this.

Specifying the Transfer Function Coefficients

The first is to specify the numerator and denominator of the transfer function. For example to specify the transfer function:

$$H(s) = \frac{3s - 2}{4s^2 + 2s + 7}$$

we would first write out the numerator and denominator and then pass these to the `lti` method.

```
import scipy.signal as signal

num = [3, -2]
den = [4, 2, 7]
tf = signal.lti (num, den)
```

Listing 13.3 Defining a Transfer Function.

Note that the coefficients in the numerator, $[3, -2]$ corresponds to the coefficients in numerator equation, $3s - 2$. The same applies to the denominator. If there are terms missing, then zeros are used as place holders. For example, if the denominator is $s^2 + 3$, then the coefficients will be specified as $[1, 0, 3]$.

To plot the Bode curves we would use the `bode` method. The `bode` method takes as an argument the variable returned by the `lti` method. To illustrate this let us plot the Bode response for a simple first-order system:

$$H(s) = \frac{K}{s + a} \tag{13.8}$$

```
from scipy import signal
import matplotlib.pyplot as plt
K = 1 # gain
a = 1
# Create the LTI system
sys = signal.lti([K], [1, a])
# Bode returns the frequency axis and magnitude and phase values
w, mag, phase = signal.bode(sys)

# Plot the magnitude
plt.subplot (2, 1, 1)
plt.xlabel ('Frequency, rad/s')
plt.ylabel ('Amplitude, dB')
```

```

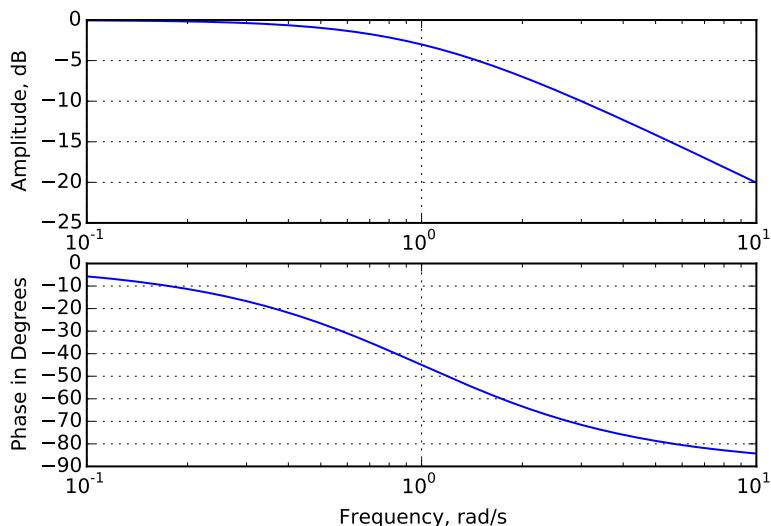
plt.semilogx(w, mag)
plt.grid(True)

# Plot the phase
plt.subplot (2, 1, 2)
plt.xlabel ('Frequency, rad/s')
plt.ylabel ('Phase in Degrees')
plt.semilogx(w, phase)
plt.grid(True)
# Export as a pdf file for publication quality
plot.savefig ('firstOrder.pdf')
plt.show()

```

Listing 13.4 Bode Plot for a First-Order System.

Figure 13.11 shows the output from the Bode plot.

**Figure 13.11** Bode plot from Python scipy first order system (13.8).

Let us return to the example we used to draw the Bode manually in (13.7). The transfer function was given by:

$$H(s) = \frac{10^3(s + 20)}{(s + 10)(s + 200)} = \frac{s + 20000}{s^2 + 210s + 2000}$$

The only line we need to change in the Python code we used previously is:

```
sys = signal.lti([1000, 20000], [1, 210, 2000])
```

Running this program will give the Bode plots shown in Figure 13.12. Notice the similarity with the manual effort in Figure 13.9.

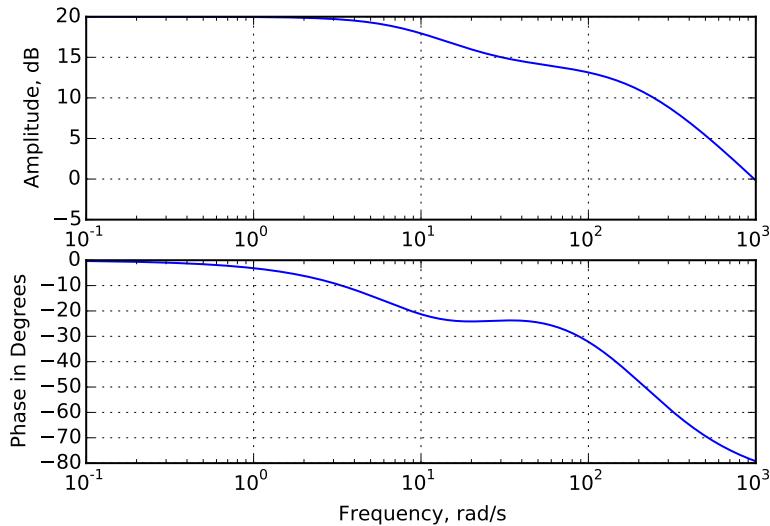


Figure 13.12 Bode plot for the example computed manually (13.7) but here computed by software.

Specifying the Zeros and Poles

It is also possible to specify a LTI system by the zeros and poles of the transfer function. Consider the transfer function:

$$H(s) = \frac{4}{(s + 4)(s - 6)} \quad (13.9)$$

This transfer function has no zeros and two poles, -4, 6. The `lti` method can take three arguments that are interpreted as the zeroes, poles and the gain of the system. The script shown in 13.5 shows the system defined in terms of its zeros, poles and gain. Note that the entry for the zeros is empty because there are no zeros for this system (The numerator is just 4).

```
from scipy import signal
import matplotlib.pyplot as plt
K = 4
sys = signal.lti([], [-4, 6], K)
w, mag, phase = signal.bode(sys)

plt.subplot (2, 1, 1)
```

```

plt.semilogx(w, mag)
plt.grid(True)

plt.subplot(2, 1, 2)
plt.semilogx(w, phase)
plt.grid(True)
plt.show()

```

Listing 13.5 Bode Plot for a First-Order System.

Specifying the State space Matrices

Another way to describe a system is by specifying the matrices of the state space model. This can be more useful if we don't have access to the transfer function. If we consider the following system:

$$\frac{dx_1}{dt} = v_o - k_1 x_1$$

$$\frac{dx_2}{dt} = k_1 x_1 - k_2 x_2$$

The state space matrices are given by:

$$A = \begin{bmatrix} -k_1 & 0 \\ k_1 & k_2 \end{bmatrix} \quad B = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

We will pass the state variables values straight through to the output without modification and assume that the D matrix is zero:

$$C = [1 \ 0] \quad D = [0]$$

With these in mind we can write the script shown in script 13.6.

```

from scipy import signal
import matplotlib.pyplot as plt
import numpy as np

k1 = 0.1; k2 = 0.34; vo = 4

A = np.array([[-k1, 0], [k1, -k2]])
B = np.array([[1],[0]])
# Generate data for x1 with respect to input
C = np.array([1, 0])

# Uncomment to generate data for x2 with respect to input

```

$$y(t) = [1 \ 0 \ 0] \begin{bmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \end{bmatrix}$$

Figure 13.13 When using the Python lti method, the C matrix must be a single row. Use ones and zero to pick out particular state variables of interest, in this example, $x_1(t)$.

```
#C = np.array([0, 1])
D = np.array([0])
sys = signal.lti(A, B, C, D)
w, mag, phase = signal.bode(sys)

plt.subplot (2,1,1)
plt.xlabel ('Frequency, rad/s')
plt.ylabel ('Amplitude, dB')
plt.semilogx(w, mag)
plt.grid(True)

plt.subplot (2,1, 2)
plt.xlabel ('Frequency, rad/s')
plt.ylabel ('Phase in Degrees')
plt.semilogx(w, phase)
plt.grid(True)
plt.show()
```

Listing 13.6 Bode Plot Using State Space Approach.

Normally C is a matrix of dimensions $m \times n$ where m is the number of outputs and n the number of state variables. If the outputs and state variables are the same then the C is set to the $m \times n$ identity matrix. This yields a $m \times 1$ vector of outputs. However it appears that the lti method in Python can only analyse one output at a time so that the C matrix has to be a $1 \times n$ vector. By setting one element in C vector to one and all other elements to zero it is possible to pick out one variable at a time and plot its frequency response (Figure 13.13).

By default lti returns the frequency in rad/sec. To convert this to Hertz, multiply the frequency vector that is returned by Bode by $1/2\pi$.

To generate publication quality plots, export the plot to pdf format: `plt.savefig('myplot.pdf')`

13.7 Zero Frequency Response

The zero frequency response is the response when the frequency, ω , is zero. That is $H(0)$. The zero frequency response is important because it has a clear physical meaning when the input is a step input and can shed light on the operation of a system. Assume that the input signal is a step response. The step signal will be relative to the prevailing steady state so that the change in X_o will be δX_o , that is $U(s) = \delta X_o / s$ then by the final-value theorem:

$$\lim_{s \rightarrow 0} s \frac{\delta X_o}{s} = \delta X_o$$

A δX_o input will result in a δx_{ss} change. This change can also be calculated using the final-limit theorem:

$$\begin{aligned}\delta x_{ss} &= \lim_{s \rightarrow 0} sX(s) = \lim_{s \rightarrow 0} sH(s)U(s) \\ &= \lim_{s \rightarrow 0} sH(s) \frac{\delta X_o}{s} = H(0) \delta X_o\end{aligned}$$

That is the steady state level of the output, δx_{ss} is given by the ratio:

$$\frac{\delta x_{ss}}{\delta X_o} = H(0)$$

That is, the gain at zero frequency is the ratio of the change in the steady state output to the change in the input signal. As the change in the input signal gets smaller, the ratio of the input to output converges to the derivative of the state variable with respect to the parameter undergoing a step response:

$$H(0) = \frac{dx}{dp} \quad (13.10)$$

where we use p to designate the step input. An example will help illustrate this result.

Example 13.3

Consider the simple gene regulatory network with a single transcription factor, x :

We will assume that the expression rate for x is controlled by a factor X_o . Let us examine the frequency response of this system to a sinusoidal input at X_o . We will also assume that the first step is governed by the rate law $v_1 = k_1 X_o$ and the degradation step by $v_2 = k_2 x$.

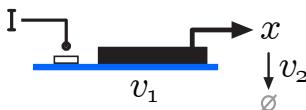


Figure 13.14 Single Gene Network

The transfer function for this system was determined in example 17.2 to be:

$$H_x(s) = \frac{k_1}{s + k_2}$$

The zero frequency response is determined by setting $s = 0$ so that the amplitude component is given by:

$$|H_x(0)| = \frac{k_1}{k_2}$$

Clearly the amplitude change is not zero, so what is it? If we look at the steady state solution, the answer will become clear. At steady state, $v_1 - v_2 = 0$, that is:

$$k_1 X_o = k_2 x$$

In other words the steady state concentration of x is:

$$x_{ss} = \frac{k_1 X_o}{k_2}$$

The steady state sensitivity of x with respect to X_o is given by:

$$\frac{dx}{dX_o} = \frac{k_1}{k_2} \quad (13.11)$$

which is the amplitude change we observed at zero frequency. Therefore the amplitude change at zero frequency is the sensitivity of the particular state variable (species concentration) with respect to the input signal.

We can use the final-value theorem to show that we can get the same result if we compute $H_x(s)$. Consider the same transfer function again:

$$H_x(s) = \frac{k_1}{s + k_2}$$

The output $X(s)$ as a result of an input $U(s)$ is:

$$X(s) = H_x(s) U(s)$$

Let's suppose that the input is a step function relative to the prevailing steady state, that is a constant signal, δX_o . The transform, $U(s)$, of δX_o is equal to $\delta X_o/s$, that is:

$$X(s) = H_x(s) \frac{\delta X_o}{s}$$

We now use the final-value theorem to find the value, δ_{ss} , at $t \rightarrow \infty$, that is to find the limit to:

$$\delta x_{ss} = \lim_{s \rightarrow 0} sH_x(s) \frac{\delta X_o}{s} = \lim_{s \rightarrow 0} s \frac{k_1}{s + k_2} \frac{\delta X_o}{s} = \frac{k_1}{k_2} \delta X_o$$

That is δx_{ss} , the steady state change in the output is $\delta X_o k_1 / k_2$. Dividing both sides by δX_o we obtain the gain which is that same as the gain we observed in the zero frequency (13.11).

$$\text{gain} = \frac{\delta x_{ss}}{\delta X_o}$$

Exercises

1. Which of the following statements is true:
 - (a) The effect of a linear system on a sinusoidal input is to change the amplitude and frequency.
 - (b) The effect of a linear system on a sinusoidal input is to change the frequency and phase.
 - (c) The effect of a linear system on a sinusoidal input is to change the amplitude and phase.
 - (d) The effect of a linear system on a sinusoidal input is to change only the amplitude.
2. For a complex number $a + jb$, write out the amplitude and phase.
3. Given the transfer function $H(s) = b/(s + a)$, derive the amplitude and phase response for this system.
4. An amplifier has a peak power output of 50 mW but drops to 25 mW when the input signal frequency reaches 20 kHz. What is the change in dB between the two power levels?
5. Draw a Bode plot for the transfer function $H(s) = b/(s + a)$.
6. Investigate how one might draw a Bode plot for the first-order system $1/(s + 1)$ using Matlab or a Matlab like application such as Octave or Scilab?
7. Define the bandwidth.
8. For the first-order system, $H(s) = 2/(s + 2)$, estimate the bandwidth.
9. Manually draw the Bode plots for the following transfer functions:

$$H(s) = \frac{20}{s(1 + 0.1s)}$$

$$H(s) = \frac{64(s + 2)}{s(1 + 0.5)}$$

10. Confirm the manual drawing of the Bode plots in the previous section by using a suitable software package to draw the Bode plots for you.
11. Show that the zero frequency response $H(0)$ is given by: dx/dp where x is a state variable and p an input.

13.8 Appendix

Solution of a linear systems with a sinusoidal inputs using the transitional approach.

$$\frac{dx}{dt} + ax = b \sin(\omega t)$$

where the input to the equation is a sinusoidal equation, $\sin(\omega t)$. This equation is in the standard linear form:

$$\frac{dx}{dt} + P(t)x = Q(t)$$

We can therefore use the integrating factor technique to solve this equation. The integrating factor is given by

$$\rho = e^{\int P(t)dt} = e^{\int adt} = e^{at}$$

Multiplying both sides by ρ and noting that

$$\frac{d}{dt} \left(\int P(t)dt \right) = P(t)$$

we obtain

$$\frac{d}{dt} (xe^{at}) = e^{at}b \sin(\omega t)$$

(Note that $\frac{d}{dt} (xe^{at}) = \frac{dx}{dt}e^{at} + xae^{at}$). Assuming an initial condition of $x(0) = 0$ and integrating both sides gives:

$$x = b \frac{\omega e^{-at} + a \sin(\omega t) - \omega \cos(\omega t)}{a^2 + \omega^2}$$

At this point we only want to consider the steady state sinusoidal response, hence as $t \rightarrow \infty$, then

$$x = \frac{b}{\sqrt{a^2 + \omega^2}} \frac{a \sin(\omega t) - \omega \cos(\omega t)}{\sqrt{a^2 + \omega^2}}$$

To show that the frequency of the input signal is unaffected, we proceed as follows. We start with the well known trigonometric identity:

$$\begin{aligned} A \sin(\beta - \alpha) &= A \cos(\alpha) \sin(\beta) - A \sin(\alpha) \cos(\beta) \\ &= a \sin(\beta) - \omega \cos(\beta) \end{aligned}$$

where $a = A \cos(\alpha)$ and $\omega = A \sin(\alpha)$. If we sum the squares of a and ω we obtain $a^2 + \omega^2 = A^2(\sin^2(\alpha) + \cos^2(\alpha)) = A^2$. That is:

$$A = \sqrt{a^2 + \omega^2}$$

Similarly, $\omega/a = A \sin(\alpha)/(A \cos(\alpha)) = \tan(\alpha)$. That it is:

$$\alpha = \tan^{-1}\left(\frac{\omega}{a}\right)$$

Since $a \sin(\beta) - \omega \cos(\beta) = A \sin(\beta - \alpha)$ where $\beta = \omega t$ then

$$x = \frac{b}{\sqrt{a^2 + \omega^2}} \sin(\omega t - \alpha)$$

This final result shows us that the frequency, ω remains unchanged but the amplitude is scaled by $\sqrt{a^2 + \omega^2}$ and the phase shifted by α . In summary the amplitude change is given by:

$$\frac{A_{out}}{A_{in}} = \frac{1}{\sqrt{a^2 + \omega^2}} \quad (13.12)$$

and the phase shift by:

$$\alpha = -\tan^{-1}\left(\frac{\omega}{a}\right) \quad (13.13)$$

14

Stability

“Q: Explain the concept of homeostasis?

A: It is when you stay at home all day and don’t go out.”

– Richard Benson, F in Exams: The Best Test Paper Blunders, 2008

14.1 Introduction

Biological systems are continually subject to external and internal momentary perturbations. Unless adequately checked, such perturbation could be highly disruptive and prevent normal functioning of the cell. Of particular interest is whether a given momentary perturbation leads to the system diverging from the steady state or whether the system relaxes back. In this chapter we will look more closely at the stability of systems.

14.2 Internal Stability

We will define the internal stability of a system as the tendency for the system to return to a steady state after one or more of the state variables have been perturbed. We have seen this already in the simulations shown in Figure 5.6.

The time domain solution to the first-order system:

$$\frac{dx_1}{dt} = k_1 X_o - k_2 x$$

is given by the total response equation:

$$x(t) = \frac{X_0 k_1}{k_2} \left(1 - e^{-k_2 t}\right) + x(0)e^{-k_2 t} \quad (14.1)$$

As described before the first part of the equation is the steady state term and the second part the transient term. The dynamics of the system is therefore determined by the second part of the equation. This part corresponds to the solution of the homogeneous system, i.e. when there is no input to the system. That is the solution to:

$$\frac{d\mathbf{x}}{dt} = \mathbf{A}\mathbf{x}$$

We know that the general solution to the homogeneous system is a weighted sum of exponentials, $c_1 e^{\lambda_i t}$, where λ_i are eigenvalues of the \mathbf{A} or Jacobian matrix. Recall that the eigenvalues of a matrix \mathbf{A} are defined by:

$$\mathbf{A}\mathbf{v} = \lambda\mathbf{v}$$

where \mathbf{v} is an eigenvector. The above definition can be rearranged to give a set of linear homogeneous equations:

$$(\mathbf{A} - \lambda\mathbf{I})\mathbf{v} = 0$$

From linear algebra we know that a non-trivial solution to a homogeneous system only exists if the determinant of $(\mathbf{A} - \lambda\mathbf{I})$ is zero. Therefore the eigenvalues of \mathbf{A} are the solutions to the equation:

$$\det(\mathbf{A} - \lambda\mathbf{I}) = 0$$

which when expanded yields a polynomial in the eigenvalues, λ . For example if the \mathbf{A} matrix is given by:

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

Then $\mathbf{A} - \lambda\mathbf{I}$ is given by:

$$\begin{bmatrix} 2 - \lambda & 1 \\ 0 & -(1 + \lambda) \end{bmatrix}$$

from which the determinant can be written out as :

$$-(2 - \lambda)(1 + \lambda) = 0$$

From which we conclude that there are two eigenvalues with $\lambda = -1$ and 2 .

It will be recalled that the Laplace transform with non-zero initial conditions is given by:

$$X(\mathbf{s}) = (s\mathbf{I} - \mathbf{A})^{-1}\mathbf{x}_0 + (s\mathbf{I} - \mathbf{A})^{-1}\mathbf{B}\mathbf{U}(s))$$

The first part represents the transient response under non-zero initial conditions and the second part the zero-state response. Other than a change of sign, the term $(sI - A)$, is the same as the expression for computing the eigenvalues for the homogeneous system. We therefore conclude that λ terms for the homogeneous solution can be obtained by looking at polynomial that results from expanding the determinant of $(sI - A)$. When written out in the transfer form, the expansion of the determinant appears as the polynomial in the denominator of the transfer function. We conclude that the dynamics of a system can be determined by examining the solutions to the polynomial in the denominator of the transfer function, $H(s)$.

Given the importance of this result, control engineers have named the various components of the transfer function. As noted in previous chapters the denominator is often called the **characteristic equation**. Likewise the equation formed by setting the numerator to zero is often called the **characteristic polynomial**. The roots to these polynomials also have terms attached to them as we've seen before. The roots of the characteristics equation (denominator) are called the **poles**, while the roots of the numerator are called the **zeros**.

Consider a system with the transfer function:

$$H(s) = \frac{s}{s^2 + 6s + 8}$$

The characteristic equation is given by:

$$s^2 + 6s + 8 = 0$$

The poles (solutions) to this equation are then:

$$s_1 = -4 \quad s_2 = -2$$

These values correspond to the λ terms in the solution to the homogeneous system. That is the homogeneous solution must be of the form:

$$C_1 e^{-4t} + C_2 e^{-2t}$$

Under a linearized system, the homogeneous solution corresponds to solutions to the equation:

$$\frac{d\delta\mathbf{x}}{dt} = A\delta\mathbf{x}$$

Given that both exponentials are negative, we conclude that an initial condition, $\delta\mathbf{x}$ will return back to the steady state solution. Our criterion for looking at the internal stability is to examine how internal perturbations in $\delta\mathbf{x}$ relax or expand.

The steady state of a system is stable if all the eigenvalues of the Jacobian matrix have negative real parts. The steady state is unstable if at least one of the eigenvalues has a positive real part.

Complex Eigenvalues

In general the eigenvalues computed from the Jacobian matrix can be complex, that is $\lambda = a + ib$. Since entries in the jacobian are always real, any complex eigenvalue will be automatically accompanied by its complex conjugate. For example:

$$\begin{bmatrix} 3 & -2 \\ 4 & -1 \end{bmatrix}$$

has the characteristic equation: $\lambda^2 - 2\lambda + 5 = 0$. Since one of the roots is the complex number $\lambda = 1 + 2i$, the second root must be the complex conjugate: $\lambda = 1 - 2i$. If the $n \times n$ matrix A has real entries, its complex eigenvalues will always occur in complex conjugate pairs. Why is this important? Consider a solution to a homogeneous system:

$$x(t) = c_1 z_1 e^{(\lambda+i\mu)t} + c_2 z_2 e^{(\lambda-i\mu)t}$$

Using Euler's formula, $e^{i\mu} = \cos(\mu) + i \sin(\mu)$ we obtain:

$$\begin{aligned} x(t) &= c_1 z_1 e^{\lambda t} (\cos(\mu t) + i \sin(\mu t)) \\ &\quad + c_2 z_2 e^{\lambda t} (\cos(\mu t) + i \sin(\mu t)) \end{aligned}$$

Writing $z_1 = a + bi$ and $z_2 = a - bi$, we get:

$$\begin{aligned} x(t) &= c_1(a + bi)e^{\lambda t} (\cos(\mu t) + i \sin(\mu t)) \\ &\quad + (a - bi)e^{\lambda t} (\cos(\mu t) + i \sin(\mu t)) \end{aligned}$$

Multiply out and separate the real and imaginary parts yields:

$$\begin{aligned} x(t) &= e^{\lambda t} [c_1(a \cos(\mu t) - b \sin(\mu t) + i(a \sin(\mu t) + b \cos(\mu t))) \\ &\quad + c_2(a \cos(\mu t) - b \sin(\mu t) - i(a \sin(\mu t) + b \cos(\mu t)))] \end{aligned}$$

The complex terms cancel leaving only the real parts. If we set $c_1 + c_2 = 2k_1$ and $(c_1 - c_2)i = 2k_2$ then:

$$\begin{aligned} x(t) &= e^{\lambda t} [k_1(a \cos(\mu t) - b \sin(\mu t)) \\ &\quad k_2(a \sin(\mu t) + b \cos(\mu t))] \end{aligned}$$

This result shows that the appearance of complex numbers in the eigenvalues results in periodic solutions.

Example 14.1

The following system:

$$\rightarrow x_1 \rightarrow x_2 \rightarrow$$

is governed by the set of differential equations:

$$\frac{dx_1}{dt} = 3 - 2x_1$$

$$\frac{dx_2}{dt} = 2x_1 - 4x_2$$

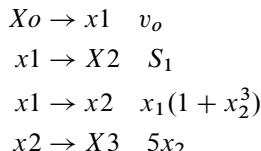
The Jacobian matrix is computed by differentiating the equations with respect to the steady state values of x_1 and x_2 :

$$\mathbf{J} = \begin{bmatrix} -2 & 0 \\ 2 & -4 \end{bmatrix}$$

The eigenvalues for this matrix are: -2 and -4 , respectively. Since both eigenvalues are negative, the system is stable to small perturbations in x_1 and x_2 .

Example 14.2

Consider the system:



where X_o , X_2 and X_3 are fixed species. At steady state $x_1 = 2.295$ and $x_2 = 1.14$ with parameter values $v_o = 8$. Determine whether this steady state is stable or not. The differential equations for the system are given by:

$$\frac{dx_1}{dt} = v_o - x_1 - x_1(1 + x_2^3)$$

$$\frac{dx_2}{dt} = x_1(1 + x_2^q) - 5x_2$$

The Jacobian matrix is computed by differentiating the equations with respect to the steady state values of S_1 and S_2 :

$$\mathbf{J} = \begin{bmatrix} -2 - x_2^3 & -3x_1x_2^2 \\ 1 + x_2^3 & -5 + 3x_1x_2^2 \end{bmatrix} = \begin{bmatrix} -3.4815 & -8.948 \\ 2.482 & 3.948 \end{bmatrix}$$

The eigenvalues for this matrix are: $0.2333 + 2.9i$ and $0.2332 - 2.9i$, respectively. Since the real parts of the eigenvalues are positive, the system is unstable to small perturbations in x_1 and x_2 .

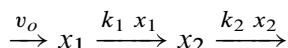
14.3 External Stability

There is one other type of stability that is useful with respect to biochemical systems, namely **external stability**. This refers to the idea that if a system is externally stable, then a finite change to an input of the system should elicit a finite change to the internal state of the system. In control theory this is called BIBO, or Bounded Input Bounded Output stability. It is very important to bear in mind that the finite change in the internal state refers to a linearized system. When the system is nonlinear, the output may be bounded by physical constraints.

A system that is internally unstable will also be unstable to changes in the systems inputs. In biochemical systems such inputs could be the boundary species that feed a pathway, a drug intervention, or the total mass of a conserved cycle. External stability can be determined using the same criteria used for internal stability, that is the real parts of eigenvalues of the Jacobian matrix should all be negative.

14.4 Phase Plots

The word **phase space** refers to a space where all possible states are presented. For example, in a biochemical pathway with two species, x_1 and x_2 , the phase space consists of all possible trajectories of x_1 and x_2 in time. For two dimensional systems the phase space can be very conveniently display on an x/y graph where each axis represents one of the state variables. A visual representation of the phase space is often called a **phase portrait** or **phase plane**. To illustrate a phase portrait consider the following simple reaction network:



with ODEs:

$$\frac{dx_1}{dt} = v_o - k_1 x_1$$

$$\frac{dx_2}{dt} = k_1 x_1 - k_2 x_2$$

We can assign particular values to the parameters, set up some initial conditions and plot in phase space the evolution of x_1 and x_2 . If we replot the solution using many different initial conditions we get something that looks like the plots shown in Figure 14.3 to 14.8.

We can assign particular values to the parameters, set up some initial conditions, and plot the evolution of x_1 and x_2 in phase space. If we replot the solution using many different initial conditions, we get something that looks like the plots shown in Figures 14.2 to 14.8.

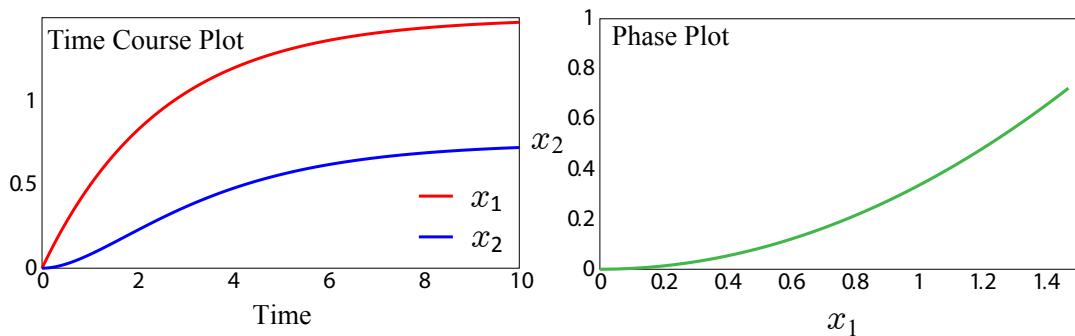


Figure 14.1 Time course simulation plot and corresponding phase plot.

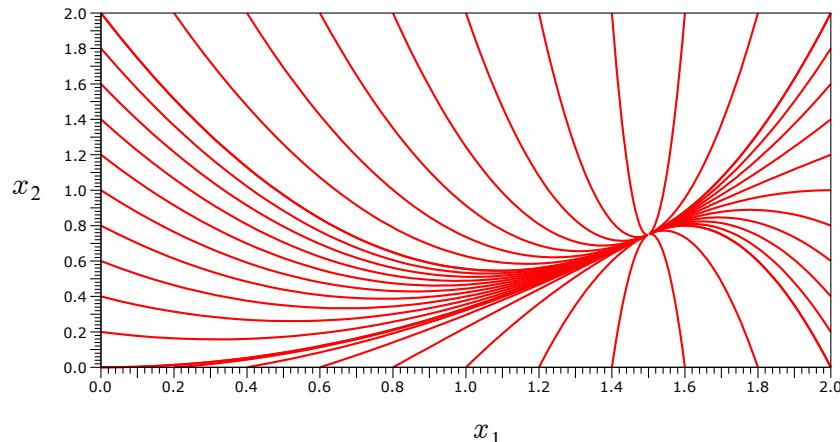


Figure 14.2 Multiple trajectories plotted on the phase plot, carried out by the Python listing: 14.4.

The plots illustrate a variety of transient behaviors around the steady state. These particular transient behaviors apply specifically to linear differential equations. If we have a nonlinear system and we linearize the system around the steady state, the linearized system will also behave in a way suggested by these plots.

Consider the general two dimensional linear set of differential equations:

$$\frac{dx_1}{dt} = a_{11}x_1 + a_{12}x_2$$

$$\frac{dx_2}{dt} = a_{21}x_1 + a_{22}x_2$$

As we've seen already, such a two dimensional linear system of differential equations has solutions

of the form:

$$\begin{aligned}x_1 &= c_1 k_1 e^{\lambda_1 t} + c_2 k_2 e^{\lambda_2 t} \\x_2 &= c_3 k_3 e^{\lambda_3 t} + c_4 k_4 e^{\lambda_4 t}\end{aligned}$$

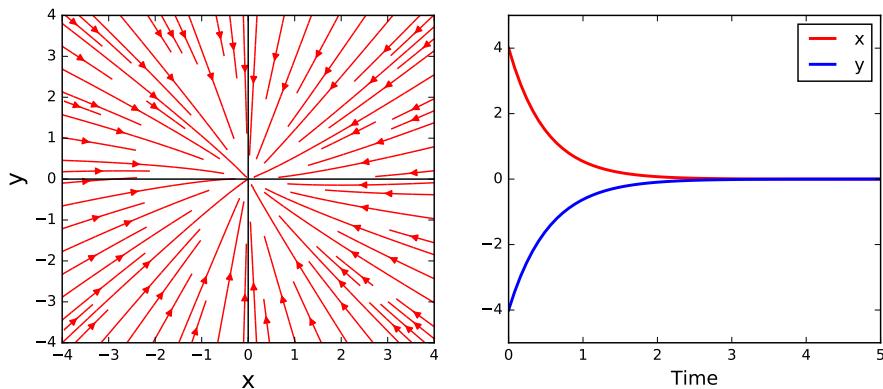


Figure 14.3 Trajectories for a two species reaction network. On the left is the phase plot and on the right, a single transient as a function of time. This system illustrates a stable node corresponding to *Negative Eigenvalues* in the Jacobian. Matrix A : $a_{11} = -2, a_{12} = 0, a_{21} = -0.15, a_{22} = -2$. Corresponding eigenvalues: $\lambda_1 = -2, \lambda_2 = -2$. The symmetry in the trajectories is due to eigenvalues of the same magnitude. See code listing 14.6

That is, a sum of exponential terms. The c_i and k_i terms are constants related to the initial conditions and eigenvectors, respectively, but the λ_i terms or eigenvalues determine the qualitative pattern that a given behavior might have. It should be noted that the eigenvalues can be complex or real numbers. In applied mathematics, e raised to a complex number immediately suggests some kind of periodic behavior. Let us consider different possibilities for the various eigenvalues.

- **Both Eigenvalues have the same sign, different magnitude but are real.** If both eigenvalues are negative, the equations describe a system known as a **stable node**. All trajectories move towards the steady state point. If the eigenvalues have the same magnitude and the c_i terms have the same magnitude, the trajectories move to the steady state in a symmetric manner as shown in Figure 14.3. If the k_i values differ, the trajectories will tend to twist as seen in Figure 14.4. They move towards the steady state rather than away.

If the two eigenvalues are both positive, the trajectories move out from the steady state reflecting the fact that the system is unstable. Such a point is called an **unstable node**. If the two eigenvalues have different magnitudes but are still positive, the trajectories twist as shown in Figure 14.4.

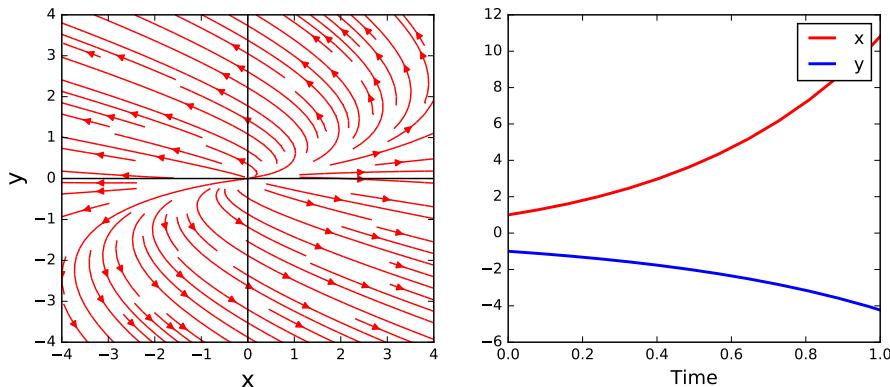


Figure 14.4 Trajectories for a two species reaction network. On the left is the phase plot and on the right a single transient as a function of time. This system illustrates a unstable node, also called an improper node corresponding to *Positive Eigenvalues*. Matrix A : $a_{11} = 1.2, a_{12} = -2, a_{21} = -0.05, a_{22} = 1.35$. Corresponding eigenvalues: $\lambda_1 = 1.6, \lambda_2 = 0.95$. See code listing 14.7

- **Real Eigenvalues but of opposite sign.** If the two eigenvalues are real but of opposite sign, we see behavior called a **saddle-node** shown in Figure 14.5. This is where the trajectories move towards the steady state in one direction, called the stable manifold, and form a stable ridge. In all other directions trajectories move away, resulting in an unstable manifold. Since trajectories can only move towards the steady state if they are exactly on the stable ridge, the saddle nodes are generally considered unstable.

Description	Eigenvalues	Behavior
Both Positive	$r_1 > r_2 > 0$	Unstable
Both Negative	$r_1 < r_2 < 0$	Stable
Positive and Negative	$r_1 < 0 < r_2$	Saddle point
Complex Conjugate	$r_1 > r_2 > 0$	Unstable spiral
Complex Conjugate	$r_1 < r_2 < 0$	Stable spiral
Pure Imaginary	$r_1 = r_2 = 0$	Center

Table 14.1 Summary of Steady State Behaviors. r_i is the real part of the complex number.

- **Complex Eigenvalues.** Sometimes the eigenvalues can be complex, that is of the form $a + ib$ where i is the imaginary number. It may seem strange that the solution to a differential equation that describes a physical system can admit complex eigenvalues. To understand what this means we must recall Euler's formula:

$$e^{i\theta} = \cos(\theta) + i \sin(\theta)$$

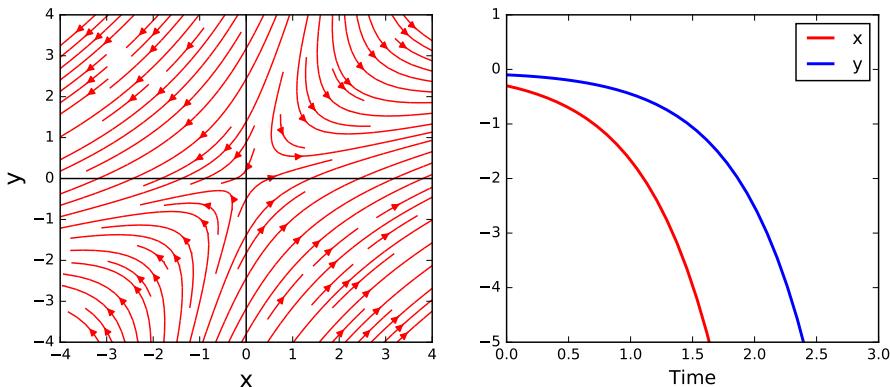


Figure 14.5 Trajectories for a two species reaction network. On the left is the phase plot and on the right a single transient as a function of time. This system illustrates a saddle node corresponding to *One Positive and One Negative Eigenvalue*. Matrix A : $a_{11} = 2, a_{12} = -1, a_{21} = 1, a_{22} = -2$. Corresponding eigenvalues: $\lambda_1 = -1.73, \lambda_2 = 1.73$. See code listing 14.8

Extended to:

$$e^{(a+bi)t} = e^{at} \cos(bt) + i e^{bt} \sin(bt) \quad (14.2)$$

When the solutions are expressed in sums of sine and cosine terms, the imaginary parts cancel out, leaving just trigonometric terms with real parts (The proof is provided at the end of the chapter in an appendix). This means that systems with complex eigenvalues show periodic behavior.

Figures 14.6, 14.7, and 14.8 show typical trajectories when the system admits complex eigenvalues. If the real parts are positive, the spiral trajectories move outwards away from the steady state. Such systems are unstable. In a pure linear system, the trajectories will expand out forever. They will only stop and converge to a stable oscillation if the system has nonlinear elements which limits the expansion. In these cases we observe limit cycle behavior.

If the real parts of the eigenvalues are negative, the spiral trajectory moves into the steady state and is therefore considered stable.

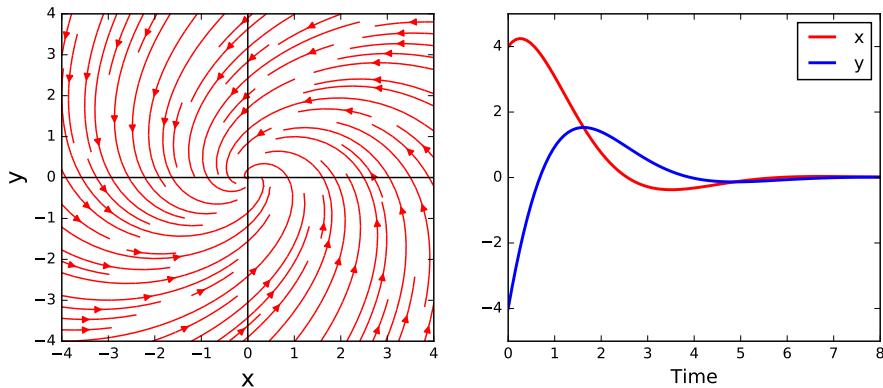


Figure 14.6 Trajectories for a two species reaction network. On the left is the phase plot and on the right, a single transient as a function of time. This system illustrates a stable spiral node corresponding to *Negative Complex Eigenvalues*. Matrix A : $a_{11} = -0.5, a_{12} = -1, a_{21} = 1, a_{22} = -1$. Corresponding eigenvalues: $\lambda_1 = -0.75 + 0.97i, \lambda_2 = -0.75 - 0.97i$. See code listing 14.5

Conjugate Pair

A complex conjugate pair is a complex number of the form: $a \pm bi$. The eigenvalues for a two variable linear system with matrix A can be computed directly using the relation:

$$\lambda = \frac{\text{tr}(A) \pm \sqrt{\text{tr}^2(A) - 4 \det(A)}}{2}$$

where $\text{tr}(A) = a + d$, and $\det(A) = ad - bc$. If the term in the square root is negative, the eigenvalues will always come out as a conjugate pair owing to the \pm term. If $\text{tr}^2(A) - 4 \det(A) < 0$, then the solution will be the conjugate pair:

$$\lambda = \frac{\text{tr}(A)}{2} \pm \frac{\sqrt{\text{tr}^2(A) - 4 \det(A)}}{2}$$

Therefore a complex eigenvalue will always be accompanied by its conjugate partner.

14.5 Routh-Hurwitz Method

In the last section we saw that the stability of a system could be determined by solving the characteristic equation and examining the poles. For simple systems this is not too difficult, however the method becomes more unwieldy and often impossible for larger systems. As a result various methods

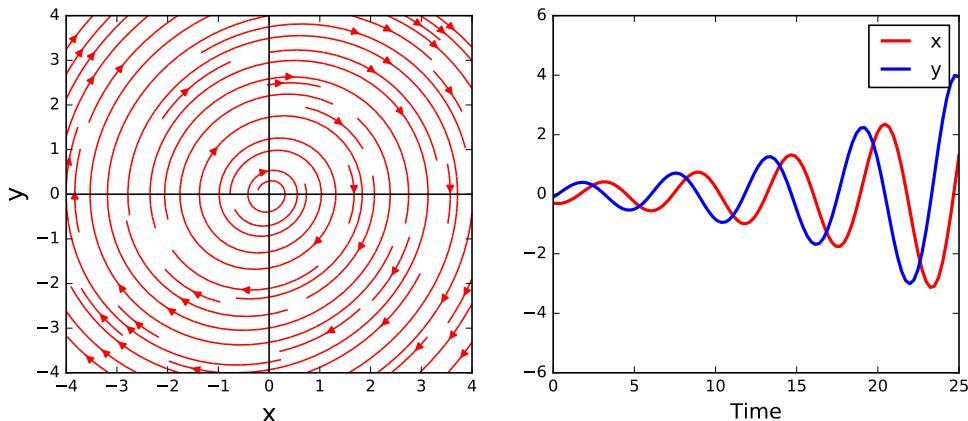


Figure 14.7 Phase portrait for the two species reaction network. Unstable spiral node. **Positive Complex Eigenvalues.** Matrix A : $a_{11} = 0, a_{12} = 1.0, a_{21} = -1.2, a_{22} = 0.2$. Corresponding eigenvalues: $\lambda_1 = 0.1 + 1.09i, \lambda_2 = 0.1 - 1.09i$. See code listing 14.9

have been introduced to make the task easier. A popular method is the Routh-Hurwitz procedure. This doesn't explicitly compute the solutions but does tell us the number of roots with positive real parts. The reader is referred to Appendix F where a discussion of general properties of polynomials is given.

Approach

The first point to make is that for a polynomial such as¹:

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

if any of the coefficients is zero or negative then there is at least one root with nonnegative real part. The exception is a_n , if that is zero we can divide through out by s to recover the required structure.

The Routh-Hurwitz criteria is therefore focused on finding negative roots for those polynomials where the coefficients have the same sign and where all coefficients are present. The Routh-Hurwitz method is based on building a table from which the number of positive roots to the characteristic equation is equal to the number of sign changes in the first column of the table. Before the table can be constructed we emphasize again the following requirements:

1. All the coefficients, a_i , must be present (non-zero) in the polynomial.

¹Many textbooks will order the coefficients a_n to a_0 , here we order them a_0 to a_n . The reason for this change is that it simplifies the notation later on which can otherwise become cluttered, see Ogata, Modern Control Engineering.

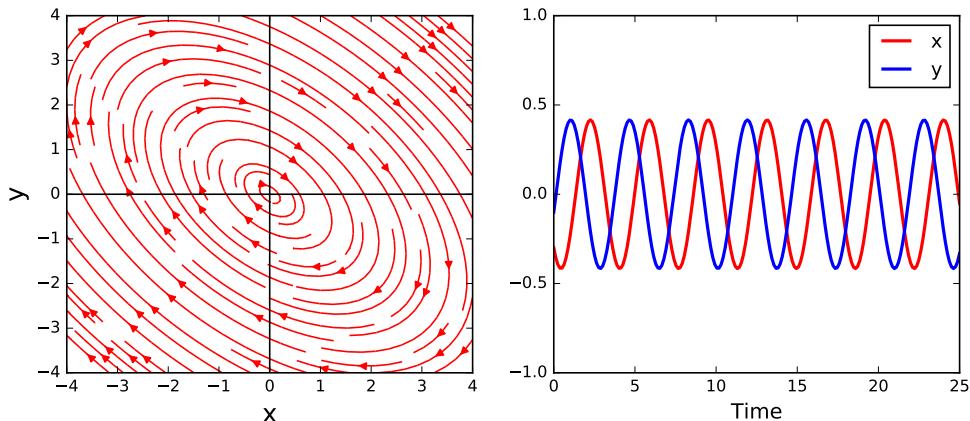


Figure 14.8 Phase portrait for the two species reaction network. Center node. **Complex Eigenvalues, Zero Real Part.** Matrix A : $a_{11} = 1, a_{12} = 2.0, a_{21} = -2, a_{22} = -1$. Corresponding eigenvalues: $\lambda_1 = 0 + 1.76i, \lambda_2 = 0 - 1.76i$. See code listing 14.10

2. All coefficients, a_i , must be positive

If these criterion are satisfied then a Routh table can be constructed. Alternatively if any of the coefficients are zero or negative in the presence of at least one positive coefficient, then there are one or more roots that are purely imaginary or with roots that have positive real parts. In other words the system will be unstable, the method stops there.

If the order of the polynomial is at least two and any coefficient is zero or negative, then the polynomial has at least one root with a non-negative real part. That is the system will be unstable.

If the coefficients are all positive and present, the procedure involves the construction of a “Routh Array” or table shown in Table 14.2.

The first two rows of the table are constructed from the coefficients of the polynomial. The remaining rows are derived from the two rows immediately above. For example, the third and fourth rows of the table are computed using the relations:

$$\begin{aligned} b_1 &= \frac{a_1 a_2 - a_0 a_3}{a_1} & b_2 &= \frac{a_1 a_4 - a_0 a_5}{a_1} & b_3 &= \frac{a_1 a_6 - a_0 a_7}{a_1} \text{ etc.} \\ c_1 &= \frac{b_1 a_3 - b_2 a_1}{b_1} & c_2 &= \frac{b_1 a_5 - b_3 a_1}{b_1} & c_3 &= \frac{b_1 a_7 - a_1 b_4}{b_1} \text{ etc.} \end{aligned}$$

Subsequent rows are computed in a similar manner from the immediate rows above. The process is continued until the n^{th} row is completed. Stability is then determined by the number of sign changes

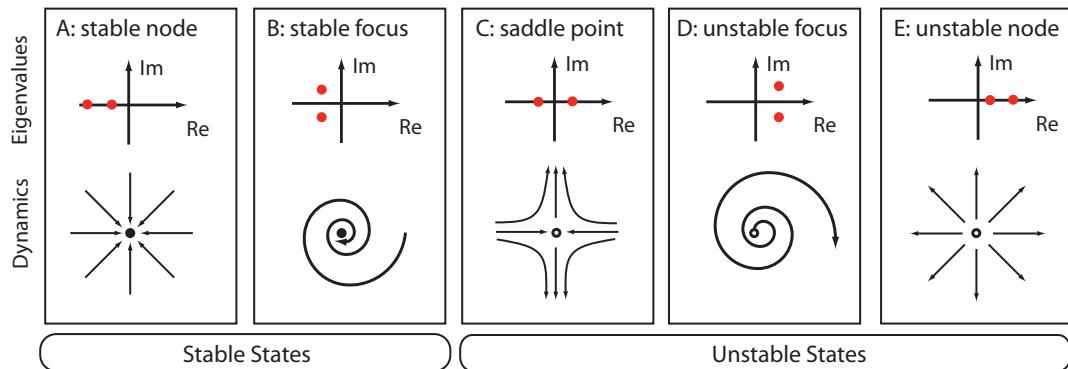


Figure 14.9 Summary of behaviors including dynamics and associated eigenvalues for a two dimensional linear system. Adapted from “Computational Models of Metabolism: Stability and Regulation in Metabolic Networks”, Adv in Chem Phys, Vol 142, Steuer and Junker.

a_0	a_2	a_4	\dots
a_1	a_3	a_5	\dots
b_1	b_2	b_3	\dots
c_1	c_2	c_3	\dots
etc.			

Table 14.2 Routh-Hurwitz Table

in the **1st** column where the number of sign changes equal to the number of poles with real parts greater than zero.

A system is stable if and only if all the entries of the first column of the Routh table are of the same sign.

Quadratic Polynomial

Consider a quadratic polynomial, $ax^2 + bx + c = 0$.

Table (14.3) shows the Routh table for the simple quadratic equation $s^2 + 3s + 1 = 0$. Note, there is no a_3 so terms containing a_3 are simplified:

In the case of a second order polynomial, we see that Routh's stability criterion reduces to the condi-

$$\begin{array}{cc|c} a_0 & a_2 \\ a_1 & 0 \\ \hline (a_1a_2 - a_0a_3)/a_1 & \end{array} \Rightarrow \begin{array}{cc|c} a_0 & a_2 \\ a_1 & 0 \\ \hline (a_1a_2)/a_1 & \end{array} \Rightarrow \begin{array}{cc|c} a_0 & a_2 \\ a_1 & 0 \\ a_2 & \end{array}$$

Table 14.3 Routh-Hurwitz Table for a simple quadratic polynomial

tion that all coefficients should be positive. In the specific case $s^2 + 3s + 1 = 0$ we have:

$$\begin{array}{cc|c} 1 & 1 \\ 3 & 0 \\ \hline 1 & \end{array}$$

Since there are no sign changes, all the roots must have negative real parts, therefore the system is stable.

Cubic Polynomial

Let us now consider a cubic polynomial. In general the Routh table for a cubic polynomial will look like:

$$\begin{array}{cc|c} a_0 & a_2 \\ a_1 & a_3 \\ \hline a_1a_2 - a_0a_3 & \\ a_1 & \\ \hline a_3 & \end{array}$$

The only way to ensure no sign changes in the first column is for the term $\frac{a_1a_2 - a_0a_3}{a_1}$ to be positive and this term can only be positive if:

$$a_1a_2 > a_0a_3$$

This gives us a clear set of rules for deciding on the stability of a third order system. The coefficients have to be all positive and the condition, $a_1a_2 > a_0a_3$ must be true.

We can illustrate this as follows. Table (14.4) shows the Routh table for the characteristic equation:

$$s^3 + 4s^2 + 1s + 16 = 0$$

From Table 14.4 we see **two sign changes** between the second and third row and the third and forth row. This tells us that there must be two positive roots. Therefore the system is unstable.

a_0	a_2	
a_1	a_3	
$\frac{a_1 a_2 - a_0 a_3}{a_1}$	$\frac{a_1 a_4 - a_0 a_5}{a_1}$	\Rightarrow
$b_1 a_3 - b_2 a_1$		
b_1		

Table 14.4 Routh-Hurwitz Table for the cubic, $s^3 + 4s^2 + 1s + 16 = 0$

Higher Order Polynomials

Consider the quartic equation:

$$s^4 + 6s^3 + 2s^2 + s + 3 = 0$$

We note that there are no zero entries and all coefficients are positive, therefore we will construct a Routh table as shown in Table 14.5. Given that there are two sign changes, there must be two positive

1	2	3
6	1	
11/6	3	
-97/11		
3		

Table 14.5 Routh-Hurwitz Table for the quartic, $s^4 + 6s^3 + 2s^2 + s + 3 = 0$

roots, therefore the systems must be unstable.

Special Case: Zeros in the first column

There is one special case to consider when zeros appear in the first column but not in other columns. If this happens there are a number of methods that can be used to prevent the zero from appearing. One method is to multiply the original polynomial by $(s + 1)$. This introduces a negative root and prevents zero appearing in the first column. Since we are only interested in positive roots this makes no difference to any conclusions we might draw. For example:

$$s^4 + s^3 + 2s^2 + 2s + 5 = 0$$

results in the partial Routh table:

1	2	5
1	2	
0	5	

Because the third row first column contains a zero we are unable to complete the subsequent columns. To get round this we multiply the polynomial by $(s + 1)$ to obtain:

$$(s + 1)(s^4 + s^3 + 2s^2 + 2s + 5) = s^5 + 2s^4 + 3s^3 + 4s^2 + 7s + 5 = 0$$

The Routh table now becomes:

1	3	7
2	4	5
2	9	
-10	10	
11		
10		

We now see there are two sign changes and therefore two positive roots, the system is therefore unstable.

An alternative strategy is to replace and first column zero entry with a very small number ϵ . The subsequent rows are constructed as before though the elements will contain the ϵ factor. Once the table has been constructed, ϵ is taken to the zero limit and this yields the final values from which the sign changes can be determined.

Special Case: Zero Row

If one of the rows is entirely zero, it means that there is a pair of roots of equal magnitude and opposite in sign. These could be two real roots with equal magnitudes and opposite signs or two conjugate imaginary roots. The polynomial immediately above the zero row is called the auxiliary polynomial. The auxiliary polynomial will only contain even powers. If m is the

The roots of the auxiliary polynomial are also solutions to the original polynomial. In order to continue the following procedures must be followed:

1. Form the auxiliary polynomial, $A(s) = 0$
2. Take the derivative of the auxiliary equation with respect to a :

$$\frac{dA(s)}{ds} = 0$$

3. Replace the row of zeros with the coefficients of $dA(s)/dt$.

4. Continue building the Routh table as before

Forming the auxiliary polynomial is best done by first labeling the rows of the Routh table, thus:

s^n	a_0	a_2	a_4	\dots
s^{n-1}	a_1	a_3	a_5	\dots
s^{n-2}	b_1	b_2	b_3	\dots
s^{n-3}	c_1	c_2	c_3	\dots
\vdots				
k	\dots			\dots

If the row for s^m is zero, the auxiliary equation is formed from the terms in the s^{m+1} row. Assuming these terms are k_1, k_2, \dots , then the auxiliary equation is given by:

$$k_1 s^{m+1} + k_2 s^{m-1} + k_3 s^{m-3} + \dots + (k_j s \text{ or } k_j) = 0$$

Note that the exponents change in steps of two. The last term depends on whether $m + 1$ is odd or even. We can illustrate this with an example, consider the polynomial:

$$s^5 + 4s^4 + 8s^3 + 8s^2 + 7s + 4 = 0$$

The Routh table is:

s^5	1	8	7
s^4	4	8	4
s^3	6	6	0
s^2	4	4	
s^1	0	0	

The 5th row is all zeros. In this situation we form the auxiliary polynomial from the row above. In this case $m = 1$, therefore $m + 1 = 2$ and $m - 1 = 0$, therefore the auxiliary equation is

$$A(s) = 4s^2 + 4 = 0$$

From this we form the derivative:

$$\frac{dA(s)}{ds} = 8s$$

Replace the zero row with the coefficients from this polynomial and continue the table:

8	0
4	

We now see there is no sign change so that we conclude that the system is stable.

Conclusion

One advantage of using the Routh-Hurwitz table (other than been straight forward to construct) is that entries in the table can be composed from elasticity coefficients. Thus sign changes (and hence stability) can be traced to particular constraints on the elasticity coefficients. Examples of this will be given in the following chapters.

14.6 Numerical Determination of Roots

The Routh-Hurwitz method described in the previous section is an excellent way to determine the stability of a system using just pencil and paper. As we've seen it can also be used to determine the threshold at which a system shifts from unstable to stable.

In addition to pencil and paper approaches there are many desktop computer applications available that can be used to numerically find all the roots of a polynomial system. Thus not only do we get the signs but also the values. Here we describe how to use Python to estimate the roots of a polynomial, but equivalent methods exist in tools such as MATLAB, Scilab, Mathematica, R and Julia.

Python offers a number of ways to compute the roots of a polynomial equation. Here will will focus on the NumPy package which is a basic and commonly used packages for scientific computing in Python. NumPy offers a useful N-dimensional array type (matrices) for Python as well as a number of useful linear algebra, Fourier transforms, random number and other functions. Here we are interested in the `root` method provided by NumPy. The root method takes a single argument which is a one dimensional array of the coefficients of the polynomial of interest. For example, let's start with a simple quadratic problem:

$$2s^2 + 5s - 3 = 0$$

To solve this using Python we would enter:

```
import numpy as np
p = [2, 5, -3]
print np.roots (p)
[-3.  0.5]
```

Listing 14.1 Solving $2s^2 + 5s - 3 = 0$

The array `p` holds the values for the coefficients starting at the highest coefficient. If a coefficient is absent, for example:

$$x^2 + 9 = 0$$

the position in the one dimension array is set to zero, thus:

```
import numpy as np

p = [1, 0, 9]
print np.roots (p)
[-0.+3.j  0.-3.j]
```

Listing 14.2 Solving $x^2 + 9 = 0$

The `root` method works equally well for larger polynomials:

$$2s^7 + 6s^6 + 3s^5 + 8s^4 + 2s^3 + 3s^2 + 3s - 6 = 0$$

where the Python code is:

```
import numpy as np

p = [2, 6, 3, 8, 2, 3, 3, -6]
print np.roots (p)
[-2.92359818+0.j
 -0.97076051+0.j
 -0.32627831+1.14744923j
 -0.32627831-1.14744923j
 0.45025943+0.9728138j
 0.45025943-0.9728138j
 0.64639646+0.j ]
```

Listing 14.3 Solving $2s^7 + 6s^6 + 3s^5 + 8s^4 + 2s^3 + 3s^2 + 3s - 6 = 0$

14.7 Stability from Frequency Response

Examining the roots of the characteristic equation will give a yes/no answer to whether a system is stable or not. Another approach that can be used is to examine the frequency response using Bode plots. This can be used to investigate stability as well as giving an indication of the relative stability of the system. For example, how far is the point of instability from the current state. The following applies to systems with a negative feedback loop and may not necessarily apply to other systems such as those where instability is due to positive feedback (Chapter 22).

We can gauge the relative stability of a system in terms of either the **gain margin** or the phase margin. We will investigate more fully the reasons for instability in negative feedback systems but suffice to say here that instability requires two criterion to be satisfied:

1. The phase shift for at least one of the state variables must be able to exceed $|180|$ degrees.
2. The loop gain when the phase shifts exactly -180 degrees must be greater than or equal to one.

The first criterion ensures that there is a frequency at which the negative feedback becomes reinforcing rather than reducing the effect of perturbations. This is the result of the -180 degree shift combining with the -180 degrees from the negative feedback action itself, giving a total -360 degree shift. This ensures that the negative feedback loop now acts as a positive feedback, this is a destabilising situation. However to ensure that the destabilization can propagate, the loop gain must be equal to or greater than one. A loop gain less than one will result in the destabilising action becoming attenuated, meaning the system is stable. Both criterion can be inspected using a Bode plot. Most systems will experience noise which has a wide spectrum of frequencies thus ensuring that frequencies that correspond to -180° will be present. Note that without noise, even an unstable system can sit precariously at the unstable focus.

The gain margin is the amount of gain increase required to make the loop gain unity at the frequency where the phase angle is -180° .

The phase margin is the difference between the phase of the response and -180° when the loop gain is 1.0, see Figure 14.10. Both terms can be used to measure the relative stability of a negative feedback system. Both however must be measured with respect to the loop gain and not the closed loop gain (See Figure 3.5).

To reduce the chance of unsustained oscillations a rule of thumb is that there should be a 45° to 60° phase margin. When the loop gain is one (crosses the y axis at zero), the phase must be less than 180° .

The **phase margin** is the amount the phase angle exceeds -180 degrees when the loop gain is one.

The **gain margin** is the size of the loop gain when the phase angle is -180 degrees.

Let us define the **phase crossover frequency**, ω_c , as the frequency at which the phase angle reaches -180° . The gain margin is the amplitude at the phase crossover frequency. Let a be the factor that the gain must be changed so that the gain is 1.0, that is:

$$|G(j\omega_c)|a = 1$$

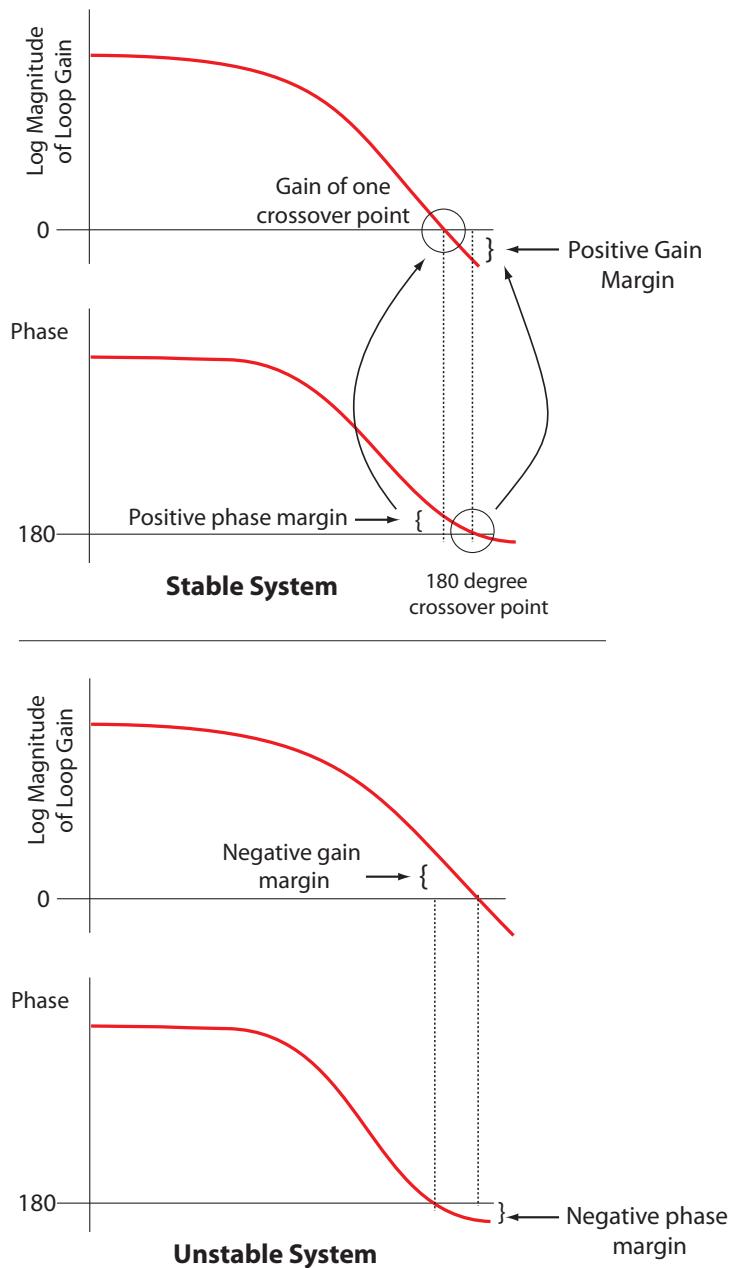


Figure 14.10 Gain and Phase Margins

In other words:

$$a = \frac{1}{|G(j\omega_c)|}$$

Expressed in decibels, we obtain:

$$a\text{dB} = 20 \log a = 20 \log \frac{1}{|G(j\omega_c)|} = -20 \log |G(j\omega_c)|$$

Expressed in decibels, the gain margin will be negative if the system is unstable, a negative gain margin. If the margin is positive then the system will be stable, a positive gain margin. From this analysis it is straight forward to compute the gain margin. First ω_c is computed when the phase plot crosses the -180° axis. Using ω_c , the gain margin in dB can be computed using $20 \log(1/G(j\omega_c))$.

Example 14.3

Consider a system where the loop gain is given by:

$$G(j\omega)H(j\omega) = \frac{1}{(s-1)^3}$$

The amplitude as a function of frequency will be given by:

$$|G(j\omega)H(j\omega)| = \frac{1}{\sqrt{1 + \omega^2}}$$

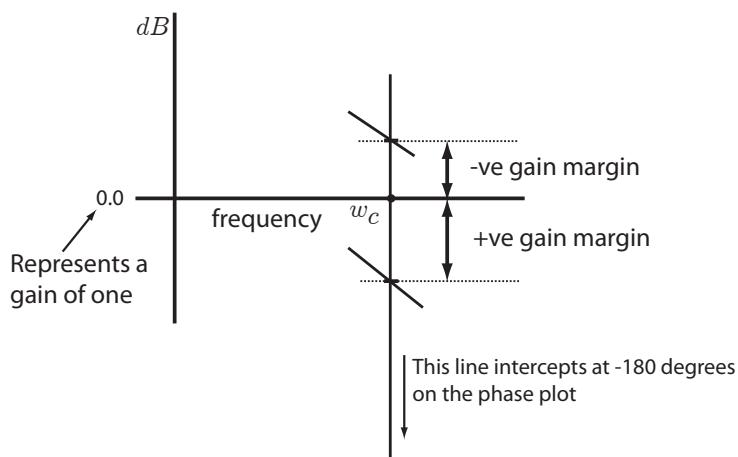


Figure 14.11 Gain and Phase Margins

Further Reading

1. Edelstein-Keshet L (2005) Mathematical Model sin Biology. SIAM Classical In Applied Mathematics. ISBN-10: 0-89871-554-7
2. Fall CP, Marland ES, Wagner JM, Tyson JJ (2000) Computational Cell Biology. Springer: Interdisciplinary Applied Mathematics. ISBN 0-387-95369-8
3. Savageau, M. A. 1976. Biochemical systems analysis: a study of function and design in molecular biology. Addison-Wesley, Reading, Mass. Reprinted in 2010 and available from Amazon.
4. Steuer R and Junker BH (2009). Computational models of metabolism: stability and regulation in metabolic networks. Advances in chemical physics, 142, 105.

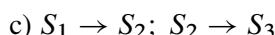
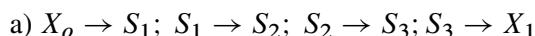
Exercises

1. Determine the Jacobian matrix for the following systems:

$$\text{a) } \frac{dx}{dt} = x^2 - y^2 \quad \frac{dy}{dt} = x(1 - y)$$

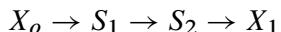
$$\text{b) } \frac{dx}{dt} = y - xy \quad \frac{dy}{dt} = xy$$

2. Compute the steady state solutions to the two systems in the previous question.
3. Determine the stability of the solutions from the previous question.
4. Determine the Jacobian in terms of the elasticities and stoichiometry matrix for the following systems:



Assume all reactions are product insensitive, X_i species are fixed, and in c) S_3 regulates the first step.

5. Show that the following system is stable to perturbations in S_1 and S_2 by computing the eigenvalues at steady state:



The three rate laws are given by:

$$v_1 = \frac{V_{m1} X_o}{Km_1 + X_o + S_1/K_i}$$

$$v_2 = \frac{V_{m2} S_2}{Km_2 + S_1 + S_2/K_j}$$

$$v_2 = \frac{V_{m3} S_3}{Km_3 + S_2}$$

Assign the following values to the parameters: $X_o = 1; X_1 = 0; V_{m1} = 1.5; V_{m2} = 2.3; V_{m3} = 1.9; K_{m1} = 0.5; K_{m2} = 0.6; K_{m3} = 0.45; K_i = 0.1; K_j = 0.2$.

6. Show that the following system is unstable. What kind of unstable dynamics does it have?

```
import tellurium as te

rr = te.loada ('''
J0: $X0 -> S1; VM1*(X0-S1/Keq1)/(1+X0+S1+pow(S4,h));
J1: S1 -> S2; (10*S1-2*S2)/(1+S1+S2);
J2: S2 -> S3; (10*S2-2*S3)/(1+S2+S3);
J3: S3 -> S4; (10*S3-2*S4)/(1+S3+S4);
J4: S4 -> $X1; Vm4*S4/(KS4+S4);

X0 = 10;           X1 = 0;
S1 = 0.973182;   S2 = 1.15274;
S3 = 1.22721;    S4 = 1.5635;
VM1 = 10;          Keq1 = 10;
h = 10;            Vm4 = 2.5;
KS4 = 0.5;
'''')
```

7. Show that the following system is unstable. What kind of unstable dynamics does it have?

```
import tellurium as te

rr = te.loada (''
J1: $src -> X;      k1*S;
```

```

J2:    X -> R;      (kop + ko*EP)*X;
J3:    R -> $waste; k2*R;
J4:    E -> EP;     Vmax_1*R*E/(Km_1 + E);
J5:    EP -> E;     Vmax_2*EP/(Km_2 + EP);

src = 0;          kop = 0.01;
ko = 0.4;         k1 = 1;
k2 = 1;          R = 1;
EP = 1;          S = 0.2;
Km_1 = 0.05;     Km_2 = 0.05;
Vmax_2 = 0.3;    Vmax_1 = 1;
KS4 = 0.5;
'''')

```

8. Using Figure 22.13, show graphically that the toggle switch model has two stable and one unstable steady states.

Appendix

See <http://tellurium.analogmachine.org> for more details of Tellurium.

```

import tellurium as te
import matplotlib.pyplot as plt

rr = te.loada '''
$Xo -> S1;  k1*Xo;
S1 -> S2;  k2*S1;
S2 -> $X1;  k3*S2;

k1 = 0.6; Xo = 1;
k2 = 0.4; k3 = 0.8;
'''')

S1Start = 0
S2Start = 0

te.setHold (True)

for i in range(1, 11):
    rr.S1 = S1Start
    rr.S2 = S2Start
    m = rr.simulate(0, 10, 120, ["S1", "S2"], integrator = "cvode")
    p = te.plotArray(m)

```

```
plt.setp (p, color='r')
S1Start = S1Start + 0.2

S1Start = 0
S2Start = 0
for i in range(1, 11):
    rr.S1 = S1Start
    rr.S2 = S2Start
    m = rr.simulate(0, 10, 120, ["S1", "S2"], integrator = "cvode")
    p = te.plotArray(m)
    plt.setp (p, color='r')
    S2Start = S2Start + 0.2

S1Start = 2
S2Start = 0
for i in range(1, 11):
    rr.S1 = S1Start
    rr.S2 = S2Start
    m = rr.simulate(0, 10, 120, ["S1", "S2"], integrator = "cvode")
    p = te.plotArray(m)
    plt.setp (p, color='r')
    S2Start = S2Start + 0.2

S1Start = 0
S2Start = 2
for i in range(1, 11):
    rr.S1 = S1Start
    rr.S2 = S2Start
    m = rr.simulate(0, 10, 120, ["S1", "S2"], integrator = "cvode")
    p = te.plotArray(m)
    plt.setp (p, color='r')
    S1Start = S1Start + 0.2
```

Listing 14.4 Script for Figure 14.2.

```
import numpy as np
import matplotlib.pyplot as plt
import tellurium as te

Y, X = np.mgrid[-4:4:100j, -4:4:100j]

r = te.loada('''
    x' = -0.5*x - y;
    y' = x - y;
```

```
x = 4; y = -4;
''')

m = r.simulate (0, 8, 100)

U = -0.5*X - Y
V = X - Y

plt.subplots(1,2, figsize=(10,4))
plt.subplot(121)
plt.xlabel('x', fontsize='16')
plt.ylabel('y', fontsize='16')
plt.streamplot(X, Y, U, V, density=[1, 1])

plt.ylim((-4,4))
plt.xlim((-4,4))

plt.axhline(0, color='black')
plt.axvline(0, color='black')

plt.subplot(122)

plt.ylim((-5,5))
plt.xlim((0,8))
plt.xlabel('Time', fontsize='13')

plt.plot (m[:,0], m[:,1], color='r', linewidth=2, label='x')
plt.plot (m[:,0], m[:,2], color='b', linewidth=2, label='y')
plt.legend()
plt.show()
```

Listing 14.5 Script for Figure 14.6.

```
import numpy as np
import matplotlib.pyplot as plt
import tellurium as te

Y, X = np.mgrid[-4:4:100j, -4:4:100j]

r = te.loada('''
    x' = -2*x - 0*y;
    y' = -0.15*x - 2*y;
```

```
x = 4; y = -4;
''')

m = r.simulate (0, 8, 100)

U = -2*X - 0*Y
V = -0.15*X - 2*Y

plt.subplots(1,2, figsize=(10,4))
plt.subplot(121)
plt.xlabel('x', fontsize='16')
plt.ylabel('y', fontsize='16')
plt.streamplot(X, Y, U, V, density=[1, 1])

plt.ylim((-4,4))
plt.xlim((-4,4))

plt.axhline(0, color='black')
plt.axvline(0, color='black')

plt.subplot(122)

plt.ylim((-5,5))
plt.xlim((0,5))
plt.xlabel('Time', fontsize='13')

plt.plot (m[:,0], m[:,1], color='r', linewidth=2, label='x')
plt.plot (m[:,0], m[:,2], color='b', linewidth=2, label='y')
plt.legend()
plt.show()
```

Listing 14.6 Script for Figure 14.3.

```
import numpy as np
import matplotlib.pyplot as plt
import tellurium as te

Y, X = np.mgrid[-4:4:100j, -4:4:100j]

r = te.loada('''
    x' = 1.2*x - 2*y;
    y' = -0.05*x + 1.35*y;

    x = 1; y = -1;
```

```
''')

m = r.simulate (0, 8, 100)

U = 1.2*X - 2*Y
V = -0.05*X + 1.35*Y

plt.subplots(1,2, figsize=(10,4))
plt.subplot(121)
plt.xlabel('x', fontsize=16)
plt.ylabel('y', fontsize=16)
plt.streamplot(X, Y, U, V, density=[1, 1])

plt.ylim((-4,4))
plt.xlim((-4,4))

plt.axhline(0, color='black')
plt.axvline(0, color='black')

plt.subplot(122)

plt.ylim((-6,12))
plt.xlim((0,1))
plt.xlabel('Time', fontsize=13)

plt.plot (m[:,0], m[:,1], color='r', linewidth=2, label='x')
plt.plot (m[:,0], m[:,2], color='b', linewidth=2, label='y')
plt.legend()
plt.show()
```

Listing 14.7 Script for Figure 14.4.

```
import numpy as np
import matplotlib.pyplot as plt
import tellurium as te

Y, X = np.mgrid[-4:4:100j, -4:4:100j]

r = te.loada('''
    x' = 2*x - 1*y;
    y' = 1*x - 2*y;

    x = -0.3; y = -0.1;
''')
```

```
m = r.simulate (0, 8, 100)

U = 2*X - 1*Y
V = 1*X -2*Y

plt.subplots(1,2, figsize=(10,4))
plt.subplot(121)
plt.xlabel('x', fontsize='16')
plt.ylabel('y', fontsize='16')
plt.streamplot(X, Y, U, V, density=[1, 1])

plt.ylim((-4,4))
plt.xlim((-4,4))

plt.axhline(0, color='black')
plt.axvline(0, color='black')

plt.subplot(122)

plt.ylim((-5,1))
plt.xlim((0,3))
plt.xlabel('Time', fontsize='13')

plt.plot (m[:,0], m[:,1], color='r', linewidth=2, label='x')
plt.plot (m[:,0], m[:,2], color='b', linewidth=2, label='y')
plt.legend()
plt.show()
```

Listing 14.8 Script for Figure 14.5.

```
import numpy as np
import matplotlib.pyplot as plt
import tellurium as te

Y, X = np.mgrid[-4:4:100j, -4:4:100j]

r = te.loada'''
    x' = 0*x + 1*y;
    y' = -1.2*x + 0.2*y;

    x = -0.3; y = -0.1;
'''
```

```
m = r.simulate (0, 25, 100)

U = 0*X + 1*Y
V = -1.2*X + 0.2*Y

plt.subplots(1,2, figsize=(10,4))
plt.subplot(121)
plt.xlabel('x', fontsize='16')
plt.ylabel('y', fontsize='16')
plt.streamplot(X, Y, U, V, density=[1, 1])

plt.ylim((-4,4))
plt.xlim((-4,4))

plt.axhline(0, color='black')
plt.axvline(0, color='black')

plt.subplot(122)

plt.ylim((-6,6))
plt.xlim((0,25))
plt.xlabel('Time', fontsize='13')

plt.plot (m[:,0], m[:,1], color='r', linewidth=2, label='x')
plt.plot (m[:,0], m[:,2], color='b', linewidth=2, label='y')
plt.legend()
plt.show()
```

Listing 14.9 Script for Figure 14.7.

```
import numpy as np
import matplotlib.pyplot as plt
import tellurium as te

Y, X = np.mgrid[-4:4:100j, -4:4:100j]

r = te.loada('''
    x' = 1*x + 2*y;
    y' = -2*x - 1*y;

    x = -0.3; y = -0.1;
''')

m = r.simulate (0, 25, 200)
```

```
U = 1*X + 2*Y
V = -2*X - 1*Y

plt.subplots(1,2, figsize=(10,4))
plt.subplot(121)
plt.xlabel('x', fontsize='16')
plt.ylabel('y', fontsize='16')
plt.streamplot(X, Y, U, V, density=[1, 1])

plt.ylim((-4,4))
plt.xlim((-4,4))

plt.axhline(0, color='black')
plt.axvline(0, color='black')

plt.subplot(122)

plt.ylim((-1,1))
plt.xlim((0,25))
plt.xlabel('Time', fontsize='13')

plt.plot (m[:,0], m[:,1], color='r', linewidth=2, label='x')
plt.plot (m[:,0], m[:,2], color='b', linewidth=2, label='y')
plt.legend()
plt.show()
```

Listing 14.10 Script for Figure 14.8.

15

First-Order Systems

“What’s in behind here? Anything dangerous?”

– Sladden in *Quatermass and the Pit*, 1958

15.1 Introduction

We now have all the machinery in place to make a thorough study of systems. One of the simplest systems is the first-order system which we will look at in detail in this chapter.

There are different ways to express the first-order transfer function, we have already seen the following form in Chapter 11, equation 11.17:

$$H(s) = \frac{k_1}{s + k_2} \quad (15.1)$$

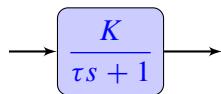
Dividing top and bottom by k_2 and defining $\tau = 1/k_2$ we can also obtain the form:

$$H(s) = \frac{K}{\tau s + 1} \quad (15.2)$$

where the symbol τ is called the **time constant** and tells us something about how responsive the system is. K is given by $k_1\tau$ and is the gain of the system. This form is called the standard form. Finally, dividing top and bottom by τ yields a third form:

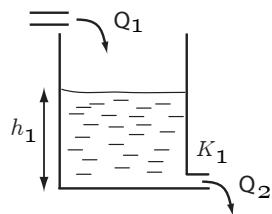
$$H(s) = \frac{k_1}{s + 1/\tau} \quad (15.3)$$

Note that these are all functionally identical. They arise as a result of specific application problems, for convenience or because they highlight particular properties that are of interest. Some of these will be illustrated in this chapter. The block diagram for a first-order system is shown below:

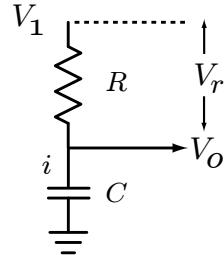


These systems, such as (15.1), are called first-order because the highest power of s in the denominator, the characteristic polynomial, is *one*. A first-order system has a single pole, see Figure 15.2. In this case the pole is negative which means that his system is stable. Figure 15.1 shows three typical first-order systems that includes a simple water tank, an RC circuit, and a gene expression cassette.

a) Simple Water Tank



b) RC Circuit



c) Gene Expression Cassette

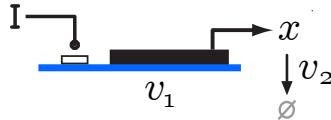


Figure 15.1 Three examples that show first-order dynamics.

Simple Tank Model

We have already seen the tank model from Chapter 1. The differential equation that describes the change in volume of the tank is given by:

$$\frac{dV}{dt} = Q_1 - Q_2 = Q_1 - (K/A)V$$

For the sake of illustration we will assume that the flow out of the tank is proportional to the volume of water in the tank. This is a first-order differential equation in V .

RC Circuit

Using Kirchhoff's voltage law, we know that in Figure 15.1b:

$$V_1 = V_r + V_o$$

where V_r is the voltage across the resistor R . Assuming that little or no current is drawn from the output V_o , the current flowing through the resistor and capacitor will be the same. From the capacitor law, we know that:

$$i = C \frac{dV_o}{dt}$$

and since $V_r = iR$, we can deduce by substitution that:

$$V_1 = C \frac{dV_o}{dt} R + V_o$$

Rearranging yields:

$$\frac{dV_o}{dt} = \frac{V_1}{RC} - \frac{V_o}{RC}$$

This is a first-order differential equation in V_o .

Simple Gene Cassette

A model for a simple gene cassette was described in Chapter 11 where the rate of change of protein, x , was given by the differential equation:

$$\frac{dx}{dt} = u(t) - kx$$

This is a first-order differential equation in V .

What is common with the three examples is that the differential equation describing their dynamics is a first-order linear differential equation:

$$\frac{dx}{dt} = u(t) - kx$$

or in standard form:

$$\frac{dx}{dt} + P(t)x = f(t)$$

Taking Laplace transforms on both sides, setting $x(0) = 0$, and assuming the input $u(t)$ is a unit impulse we obtain the first-order transfer function:

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

This is slightly different from equation 15.1 where we also include the input gain K from the input in the numerator. In this case the gain is one. We will now summarize the dynamical properties of a first-order system. All the properties we describe apply equally to all three systems in Figure 15.1.

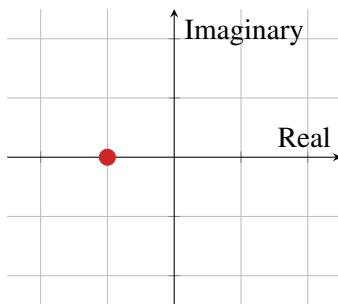


Figure 15.2 Location of the single pole ($p = -2$) in the first-order system: $K/(0.5 s + 1)$

15.2 Zero-Input Response

The zero input response represents the case when the input, $u(t)$ is zero, so that the differential equation reduces to $dx/dt = -kx$ and the solution becomes:

$$x(t) = x(0)e^{-kt}$$

The term $1/k$ is known as the time constant, τ , so that we can express the solution as:

$$x(t) = x(0)e^{-t/\tau}$$

Operationally τ is the time it takes for the system to decay to 36.8% of its original value. Therefore if τ equals 2 minutes and $x(0)$ is 10 mM, then after 2 minutes, only 3.68 mM will be left. Another way to look at this is that the time constant is related to the half-life by way of:

$$t_{1/2} = \tau \ln 2$$

15.3 Impulse Response

If the input to a first-order system is an impulse, $\delta(t)$, of strength K , with $x(0) = 0$ (zero state response), then we can write the Laplace response equation as:

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

The inverse transform is given by:

$$x(t) = Ke^{-t/\tau}$$

We see that the impulse response is almost identical to the free response but with a different scaling factor.

15.4 Unit Response

Let us apply a unit step input to a first-order system. Recall that the Laplace transform for a unit step is $1/s$, therefore, given a unit step strength of K and initial conditions $x(0) = 0$, the response equation is:

$$X(s) = K \frac{1}{s} \frac{1}{s + 1/\tau} = \frac{K}{s} - \frac{K}{s + 1/\tau}$$

The inverse transform yields:

$$x(t) = K(1 - e^{-t/\tau})$$

This result tells us that initially $x(t)$ is zero but over time $x(t)$ tends to K . The time constant, τ , tells us that at time τ , the value of $x(t)$ is 63.2% of its maximum value. Another useful property is that at $t = 0$, the slope of the tangent line at $t = 0$ is given by $1/\tau$.

Unit Step Response:

At $t = 0$ the slope = $1/\tau$

At time τ $x(t)$ has reached 63.2% of its maximum

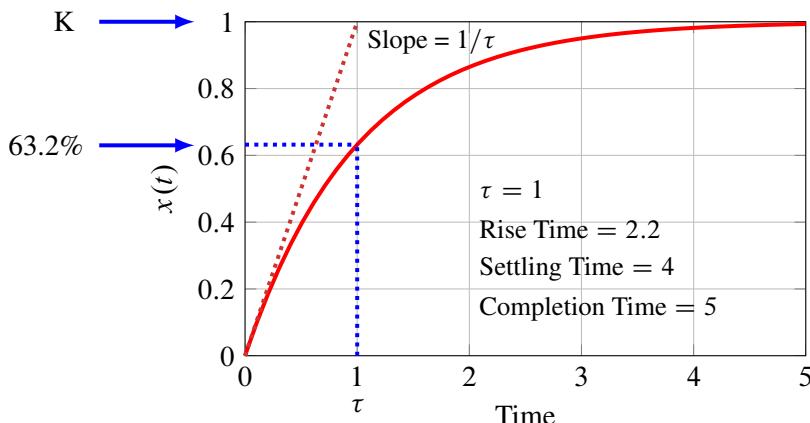


Figure 15.3 Transient response of a first-order system when a unit-step is applied and $\tau = 1$.

The transient step response for a first-order system is shown in Figure 15.3. There are a number of transient metrics that are commonly used in the literature to characterize such a response. We have already encountered one which is the time constant. Two others are also frequently used:

Rise Time, T_r . The raise time is defined as the time for the response to go from 10% to 90% of its final value. If we set the transient equation to 0.1 and 0.9 and compare the difference, the raise time

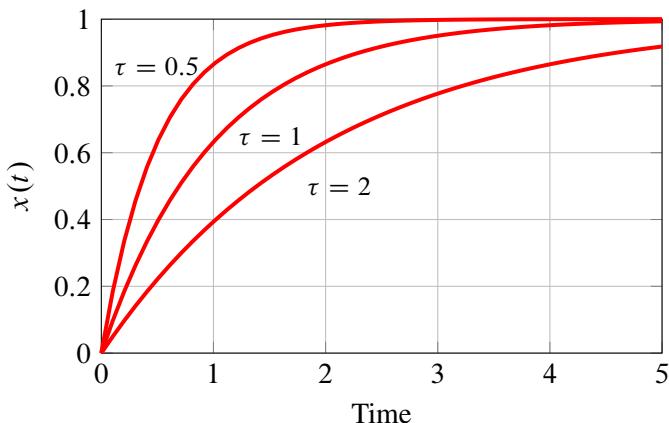


Figure 15.4 Transient response as a function of τ

can be computed from:

$$T_r = 2.2\tau$$

For a system with a time constant of 2 minutes, this means it will take 4.4 minutes for the system to go from 0.1% to 0.9% of its maximum value.

Settling Time, T_s The settling time is defined as the time for the response to reach within 2% of the final value. By looking at the transient equation and setting the final value to 0.98 it is possible to show that the setting time is given by:

$$T_s = 4\tau$$

For a system with a time constant of 2 minutes, this means it will take 8 minutes for the system to reach within 2% of the final value. Often it is considered that the system has completed its response when $t \geq 5\tau$. In our example, the response will have completed after 10 minutes.

Example

Consider the simple reaction system governed by the differential equation:

$$\frac{dx}{dt} = X_o k_1 - k_2 x$$

where the input X_o is the concentration of external metabolite, k_1 the rate constant for the first reaction, and k_2 the degradation rate constant for the metabolite, x . We've seen this system before (See example 11.6) and derived the transfer function:

$$H(s) = \frac{k_1}{s + k_2}$$

An alternative way to look at the transfer function is to divide top and bottom by k_2 , this yields:

$$H(s) = \frac{K}{\tau s + 1}$$

where $K = k_1/k_2$ and $\tau = 1/k_2$. The reason for this change is that the new constants, K and τ have a useful physical interpretation. Using this notation, and applying a unit step input to X_o , the time domain solution is given by:

$$x(t) = X_o K (1 - e^{-t/\tau}) \quad (15.4)$$

Comparing this to equation 11.18 we can see that the term $X_o K$ is the steady state solution, that is:

$$\lim_{t \rightarrow \infty} x(t) = X_o K$$

At $t = 0$ we also note that rate of change of $x(t)$ is:

$$\frac{dx}{dt} = \frac{d(X_o K (1 - e^{-t/\tau}))}{dt} = \frac{X_o K}{\tau} e^{-t/\tau} \Big|_{t=0} = \frac{X_o K}{\tau} \quad (15.5)$$

That is, τ , the time constant, indicates how fast the system can relax.

Consider an electrical circuit that comprises of a resistor, R , in series with a capacitor, C , and a voltage source, V . By considering the resistance and capacitance laws together with Kirchhoff's voltage law we can derive the following differential equation:

$$V = v_c + RC \frac{dv_c}{dt}$$

In this equation v_c the voltage across the capacitor is the state variable. This is a first-order differential equation where the transfer function, $H(s)$ is given by:

$$H(s) = \frac{1/RC}{s + 1/RC} = \frac{1}{RCs + 1}$$

In this case time constant for the system is the product, RC .

Example 15.1

A **unit step** is applied at $t = 0$ to a first-order system. After 10 minutes the response of the system has a value of 1.264 units and at steady state the system settles to a value of 2.0. What is the transfer function of this system?

The standard first-order transfer system is given by:

$$H(s) = \frac{K}{\tau s + 1}$$

For such a system the unit step time-domain response is given by - See equation (15.4):

$$x(t) = K(1 - e^{-t/\tau})$$

where $K = 2$, the final steady state value because we applied a **unit** step input. We also know that at 600 seconds, $x(t) = 1.264$:

$$1.264 = 2 \left(1 - e^{-600/\tau}\right)$$

Solving for τ gives $\tau = 600$. Therefore the transfer function is:

$$H(s) = \frac{2}{1 + 600s}$$

15.5 Ramp Response

Left as an exercise, see the end of Chapter exercises.

15.6 Frequency Response

We have already encountered the frequency response for a first-order system in Chapter 13, equation (13.4). Here we will repeat the main results of that discussion. First we will list the three first-order transfer function forms and their corresponding amplitude and phase (note that $1/k_2 = \tau$):

First-Order Form	Amplitude	Phase
$\frac{k_1}{s + k_2}$	$\frac{k_1}{\sqrt{\omega^2 + k_2^2}}$	$-\tan^{-1}\left(\frac{\omega}{k_2}\right)$
$\frac{K}{\tau s + 1}$	$\frac{K}{\sqrt{\tau^2 \omega^2 + 1}}$	$-\tan^{-1}(\tau\omega)$
$\frac{k_1}{s + 1/\tau}$	$\frac{k_1}{\sqrt{\omega^2 + 1/\tau^2}}$	$-\tan^{-1}(\tau\omega)$

Like the transient behavior, there are a number of useful metrics that are used to characterize the frequency response based on Bode plots. Plots 15.5 and 15.6 show typical amplitude and phase plots for a first-order system.

Amplitude With respect to the amplitude plot at zero frequency ($\omega = 0$) the *y*-axis intercept is k_1/k_2 . This is also called the **DC gain**. For a first-order response the amplitude remains relatively constant until at some points it begins to rapidly decline. A useful measure that gives an indication of when the amplitude has fallen off to a significant degree is the bandwidth. To be specific the **bandwidth** of a system is the frequency at which the gain drops below the 3 dB peak. 3 dB is when the signal is

$1/\sqrt{2}$ of the maximum signal amplitude, about 70% of the signal strength (This is 70% of the non-dB signal). The bandwidth is also called the cut-off frequency or the corner frequency.

The frequency at which the gain of the system falls to $1/\sqrt{2}$ of the peak gain is called the **bandwidth**. This is roughly 70% of the signal strength.

Example 15.2

Show that the bandwidth is equal to $1/\tau$ for a first-order system where τ is the time constant.

The transfer function is given by:

$$H(s) = \frac{1}{s + 1/\tau}$$

Previously we showed that the frequency amplitude for a first-order system is given by:

$$\text{Amplitude} = \frac{1}{\sqrt{(1/\tau)^2 + \omega^2}}$$

Note that at zero frequency, $\omega = 0$, Amplitude(0) = τ .

We now wish to solve for ω in terms of $1/\tau$ at the bandwidth point, that is $1/\sqrt{2}$ of the DC level. It is important to recall that the DC level is given by the transfer function at zero frequency. If we set $\omega = 0$ in the above equation then the amplitude is τ . Therefore we must solve the following equation in terms of τ :

$$\frac{\tau}{\sqrt{2}} = \frac{1}{\sqrt{(1/\tau)^2 + \omega^2}}$$

Inverting and squaring both sides yields:

$$\frac{2}{\tau^2} = \frac{1}{\tau^2} + \omega^2$$

From which we conclude that:

$$\omega = \frac{1}{\tau}$$

If the gain, K , is non-unity then the bandwidth is at:

$$\omega = \frac{\sqrt{2K^2 - 1}}{\tau}$$

For a first-order system with unity gain, the bandwidth is given by:

$$\omega = \frac{1}{\tau}$$

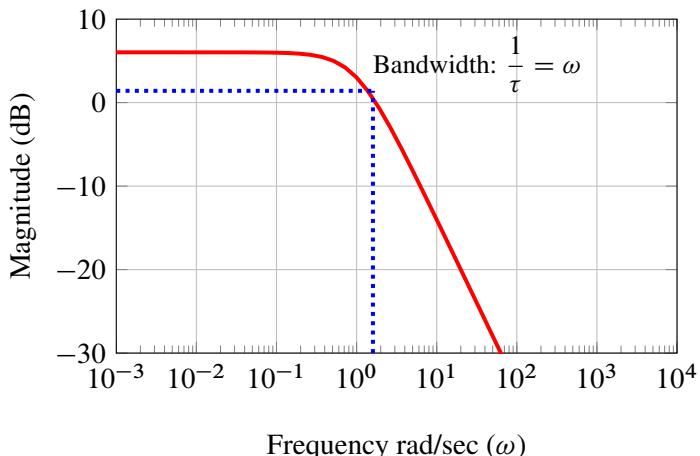


Figure 15.5 Bode Plot: Magnitude (dB). $k_1 = 2$ and $k_2 = 1$, $\tau = 2$. DC point ($\omega = 0$) is at $20 \log_{10}(2)$. The bandwidth is also called the corner frequency.

Zero frequency ($\omega = 0$)	k_1/k_2
Cut-off or bandwidth	$1/\tau$
Amplitude loss beyond cut-off	-20 dB/Decade

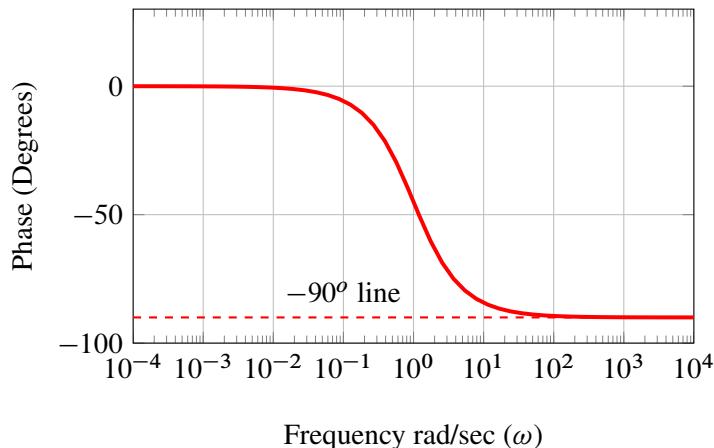
Table 15.1 Summary of Amplitude Properties

Phase Plot As we've seen before the phase shift (Figure 15.6) for a first-order system will lags by up to -90° at high frequencies.

$$\text{Phase} = \tan^{-1} \left(\frac{-\omega}{k_2} \right) = -\tan^{-1} \left(\frac{\omega}{k_2} \right)$$

Effect of Changes to Rates Constants

Let us examine the frequency response of the first-order gene expression cassette (Figure 15.1) with respect to the values of the rate constants, k_1 and k_2 . Figure 15.7 shows the amplitude Bode plot for the system where the input is the inducer molecule and the output the expressed protein. The plots show the amplitude change for two situations. In one case, solid line, the values for the rates constants are $k_1 = 0.4$ and $k_2 = 0.2$. Note that the bandwidth is approximately at 0.1 rad/sec. In the second case we make both rate constants 100 times bigger, that is $k_1 = 400$ and $k_2 = 20$. The amplitude change is shown by the dotted line where the bandwidth has increased to approximately 10 rad/sec. In

**Figure 15.6** Bode Plot: Phase Shift

comparing the two lines two things stand out. The first is that both lines appear to begin at the same point at low frequencies. We can see this more clearly if we look at the amplitude expression:

$$\text{Amplitude} = \frac{k_1}{\sqrt{\omega^2 + k_2^2}}$$

As the frequency tends to zero the amplitude tends to:

$$\text{Amplitude}_{\omega \rightarrow 0} = \frac{k_1}{k_2}$$

In other words the point at zero frequency is dependent on the ratio of the rate constants. The second and more dramatic change is the change in the bandwidth. Increasing the values of the rate constants has increased the responsiveness of the system.

Exercises

1. Show that for a unit step response applied to a first-order system, the slope at $t = 0$ is given by $1/\tau$.
2. Show that the raise time can be computed from the expression $T_r = 2.2\tau$.
3. Show that the settling time can be computed from the expression $T_s = 4\tau$.
4. Let the transfer function for a first-order system be $1/(\tau s + 1)$. The application of a unit step input takes 10 seconds to reach 50% of the step height. Compute the time constant, τ .

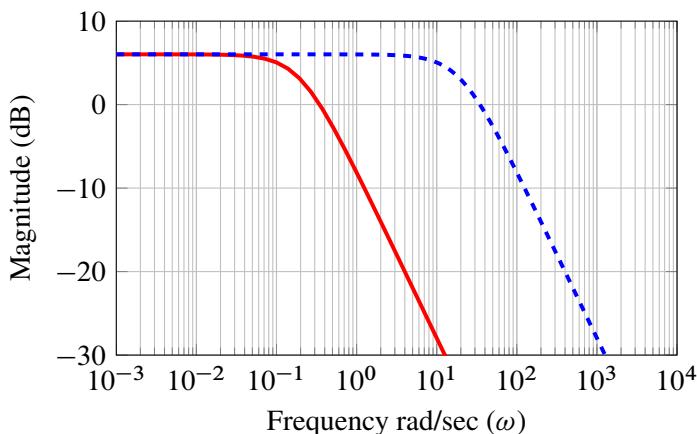


Figure 15.7 Bode Plot: Magnitude (dB). The left most plot has $k_1 = 0.4$ and $k_2 = 0.2$. The right most plot, the rate constants are 100 times bigger. This shows us that at higher rates, the system is able respond at much higher frequencies.

5. Derive the transfer function of a first-order system in response to a unit-ramp.
6. Derive the time dependent equation for the response of a first-order system to a unit ramp input (Assume initial conditions are zero).
7. At infinite time what is the difference between the value of the ramp and the state variable for a first-order system? This question addresses how well a first-order system can track a ramp input.

16

Second-Order Systems

“You’re all scared of the wrong thing.”

– Riddick in *Riddick*, 2013

16.1 Introduction

In the last chapter we looked at first order systems characterized by the transfer function:

$$H(s) = \frac{K}{\tau s + 1} \quad (16.1)$$

Recall that the order of a system is defined by the order of the characteristic polynomial. That is, the highest power of s in the denominator.

Compared to the relatively simple first-order system, second-order systems exhibit much more interesting behavior. Not only can we change the speed of response as with first-order systems, we can also change the qualitative form of the response. Whereas a first-order system could be described using a single state variable, a second-order system requires a minimum of two state variables. The general form for a second order system is:

$$H(s) = \frac{b}{s^2 + as + b} \quad (16.2)$$

However, in the literature we will often find the second-order transfer function expressed in the **standard form: second-order system**. This originates from a study of second order mechanical and

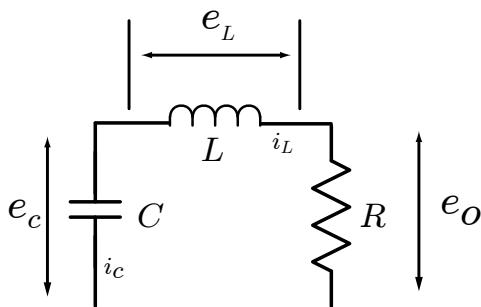


Figure 16.1 Electrical circuit with a capacitor (C), inductor (L) and resistor (R). i is the current and e the voltage.

electrical systems. Consider the RCL circuit in Figure 16.1 where the capacitor is initially charged up at $t = 0$, then allowed to discharge through the inductor and resistor.

Using Kirchhoff's voltage law for a cycle, we can state that $e_L + e_0 + e_c = 0$. The voltage across the inductor, e_L is given by $L \frac{di_L}{dt}$, and the voltage across the resistor by Ri . Substituting these into the voltage law equation gives:

$$L \frac{di_L}{dt} + Ri + e_c = 0$$

Since the current through the inductor and resistor are equal (neither can accumulate charge) and is discharged from the capacitor, the common current in the circuit is given by $i = Cde_c/dt$. Substituting i with this relationship yields:

$$LC \frac{d^2e_c}{dt^2} + RC \frac{de_c}{dt} + e_c = 0$$

Finally dividing by LC :

$$\frac{d^2e_c}{dt^2} + \frac{R}{L} \frac{de_c}{dt} + \frac{1}{LC} e_c = 0$$

If we define $\omega_n = 1/\sqrt{LC}$ and $R/L = 2\zeta\omega_n$, then we write the second-order differential equation in the form:

$$\frac{d^2e_c}{dt^2} + 2\zeta\omega_n \frac{de_c}{dt} + \omega_n^2 e_c = 0$$

In this equation ω_n is called the **natural frequency** and ζ^1 **damping ratio**. All second-order physical systems can be expressed in terms of these parameters. An example later in the chapter will show

¹The lower-case Greek letter ζ is called zeta.

how we can represent a two gene cascade as a second-order system. The equivalent transfer function of the standard form is given by:

$$H(s) = \frac{\omega_n^2}{s^2 + 2\xi\omega_n s + \omega_n^2} \quad (16.3)$$

In terms of equation (16.2), $b = \omega_n$ and $a = 2\xi\omega_n$. The transfer block for a second-order systems is represented by:

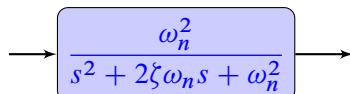
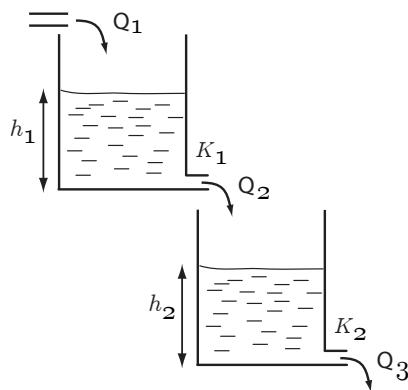
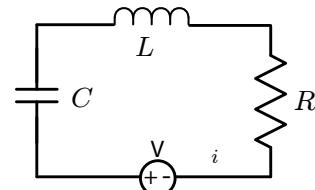


Figure 16.2 shows some examples of physical systems that can be described using second-order dynamics.

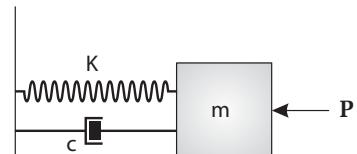
a) Water Tank Cascade



b) RCL Circuit



c) Mechanical System



d) Gene Expression Cascade

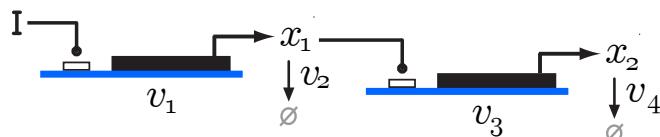


Figure 16.2 Four examples that show second-order dynamics. The mechanical system comprises a damper c and spring, K connected to a movable block of mass m .

Behavior

The behavior of a second-order systems is determined by the roots (i.e. the poles) of the characteristic equation in the transfer function (Equation (16.3)):

$$s^2 + 2\zeta\omega_n s + \omega_n^2 = 0$$

The two roots are given:

$$s_1 = \left(-\zeta + \sqrt{\zeta^2 - 1} \right) \omega_n, \quad s_2 = \left(-\zeta - \sqrt{\zeta^2 - 1} \right) \omega_n \quad (16.4)$$

Given these roots we can delineate certain cases which are of interest. The time-domain solution will be a sum of two exponentials. How these exponentials behave will depend on ζ and the magnitudes of the roots. Let us consider four cases.

16.2 Overdamped Case: $\zeta > 1$

If $\zeta > 1$ we can see that both poles, s_1 and s_2 must be real. For example if $\zeta = 1.5$ and $\omega_n = 3$, then the characteristic equation is $s^2 + 9s + 9$ and the poles are given by -7.854 and -1.146. If we assume the following transfer function:

$$H(s) = \frac{9}{s^2 + 9s + 9}$$

and apply a unit step input to the system, that is:

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

the resulting response in the time domain, assuming initial conditions equal zero, is given by:

$$x(t) = 1 + 0.171e^{-7.854t} - 1.171e^{-1.146t}$$

At infinite time the response settles to 1.0. One of the characteristics of an overdamped system is that the system tends to be slow in transitioning to the new steady state. In general the time domain solution for the overdamped system is given by:

$$x(t) = 1 + \frac{\zeta - \sqrt{\zeta^2 - 1}}{2\sqrt{\zeta^2 - 1}} e^{-\zeta - \sqrt{\zeta^2 - 1}\omega_n t} - \frac{\zeta + \sqrt{\zeta^2 - 1}}{2\sqrt{\zeta^2 - 1}} e^{-\zeta + \sqrt{\zeta^2 - 1}\omega_n t}$$

The second exponential in this expression will have increasingly smaller exponents as the damping factor increases, meaning that the decay rate will progressively get longer and longer. Hence the phrase overdamped.

16.3 Critically Damped Case: $\zeta = 1$

If $\zeta = 1$ then the poles of the characteristic equation will be equal and negative, that is $s_1 = -\omega_n, s_2 = -\omega_n$, repeated roots. The negative sign ensures the system is stable. Using the same transfer function as before the time domain solution is given by:

$$x(t) = 1 + e^{-\omega_n t} (1 + \omega_n t)$$

This is a much simpler sum of exponentials, note however that the second exponential is multiplied by time. The net result is that the response is much faster compared to the overdamped case. Also like the overdamped case, there is no overshoot as it approaches the steady state (see the case $0 < \zeta < 1$).

16.4 Underdamped Case: $0 < \zeta < 1$

If $\zeta < 1$ we have what is termed an underdamped system. The first thing to note in equation (16.4) is that exponents are now complex, this immediately tells us that the behavior of the system will be oscillatory (See Table: 16.1). We can write out the time-domain solution as:

$$x(t) = 1 - e^{-\zeta \omega_n t} \left(\frac{\zeta}{\sqrt{1 - \zeta^2}} \sin(\omega_d t) + \cos(\omega_d t) \right)$$

where $\omega_d = \omega_n \sqrt{1 - \zeta^2}$. If we use the trigonometric identity $\sin(\alpha + \beta) = \sin \alpha \cos \beta + \cos \alpha \sin \beta$ (See end of Chapter) we can also express the solution as:

$$x(t) = 1 - \frac{e^{-\zeta \omega_n t}}{\sqrt{1 - \zeta^2}} \sin \left(\omega_n \sqrt{1 - \zeta^2} t + \tan^{-1} \frac{\sqrt{1 - \zeta^2}}{\zeta} \right) \quad (16.5)$$

The term $\omega_n \sqrt{1 - \zeta^2}$ is sometimes abbreviated to ω_d and is called the **damped natural frequency**. The importance of the last modification is that the tan component is a constant and not a function of time. We are thus left with a term $\sin(\omega_d t + \theta)$ which is a classic sine wave with frequency and phase. Finally, we should also note that the sine function itself is modified by a decaying exponential. It is this that ensures that the sine wave is in fact damped and eventually disappears completely at large times.

The underdamped case is characterized by a damped oscillatory behavior as the system approaches its equilibrium point. The response is faster than the critically damped case but suffers from **overshoot**.

This equation also tells us one more thing. At $\zeta = 0$ (no damping), the exponential term reduces to one. This means there is no longer a decaying component applied to the sine wave. Therefore when $\zeta = 0$ we will observe sustained oscillations. This can happen in a mechanical system that employs a spring and a weight where there is no friction, not a particularly realistic situation.

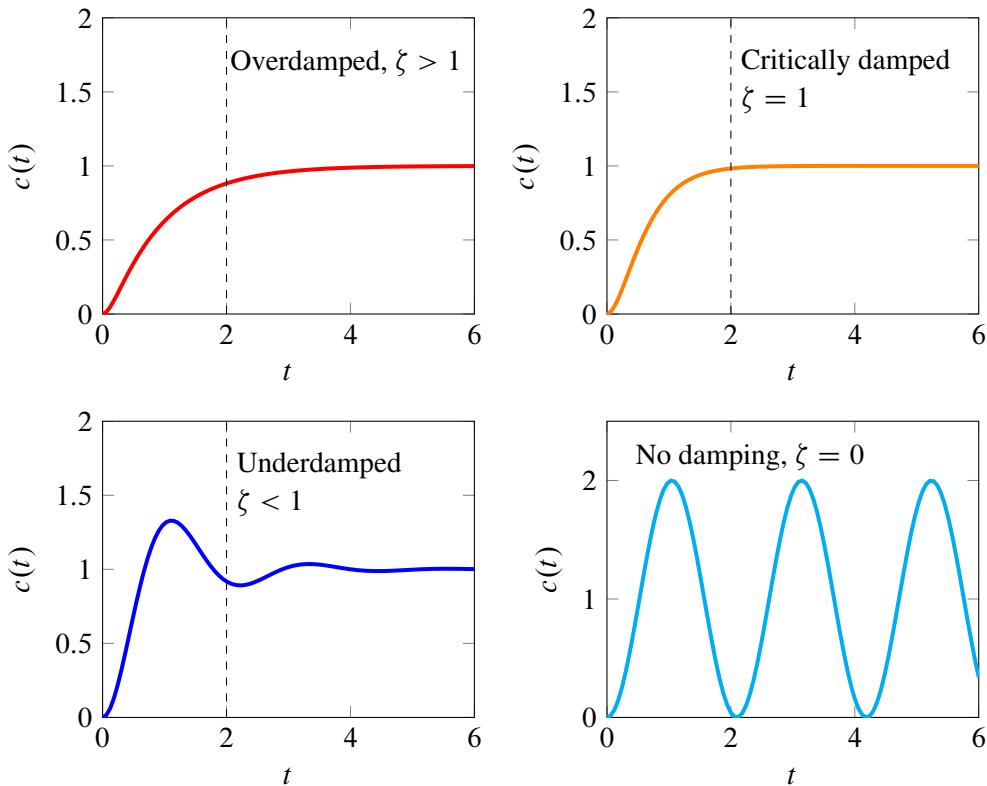


Figure 16.3 Four different behaviors exhibited by a second-order system in response to a step input. ζ is the damping factor. Note the initial response is quicker at the beginning as the damping factor decreases.

16.5 No damping: $\zeta = 0$

For completeness we describe, as already briefly mentioned, one other case when $\zeta = 0$, that is no damping at all. With no damping, damped oscillations no longer occur, instead the oscillations are sustained. These are harmonic oscillations and are not often found in biochemical systems if at all. They are unusual in that they loose no energy, require no energy source to sustain themselves and the amplitude of the oscillation depends on the initial conditions. The equation for the non-damped case is:

$$x(t) = 1 - \cos(\omega_n t)$$

This explains why ω_n is called the natural frequency, if there were no damping, this is the frequency of the oscillation.

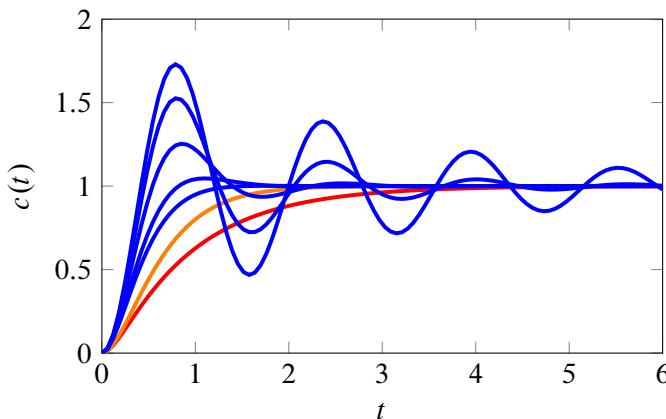


Figure 16.4 Effect of changing the value of ζ . These are similar to the curves shown in Figure 16.3 but overlaid. The most oscillatory curve was computed at $\zeta = 0.1$.

ζ	Roots (Poles)	Behavior
$\zeta > 1$	Two distinct real roots	Overdamped: Sluggish, no oscillations
$\zeta = 1$	Two equal real roots	Critically damped: Faster response, no oscillations
$0 < \zeta < 1$	Two complex conjugate roots	Underdamped: Fast, but damped oscillations occur

Table 16.1 Summary of behaviors for a second-order system.

Table ?? summarizes the various modes we expect from a second order system.

16.6 Transient Response Metrics

As indicated in the previous sections, the transient response of a second-order system is characterized by the values for ζ and ω_n . Figure 16.3 shows a variety of typical second-order responses. Although ζ and ω_n are in theory sufficient to specify the curve, from a practical point of view it would be useful to define response metrics that have a more operational and useful meanings. In the following section we will describe some of the metrics, Figure 16.5 shows how these measures relate to the response.

Rise Time: T_r

The rise time, T_r , is the time taken for the system to rise from 10% to 90% of its final value in response to a step input. Obtaining analytical solutions for the rise time is difficult and the following

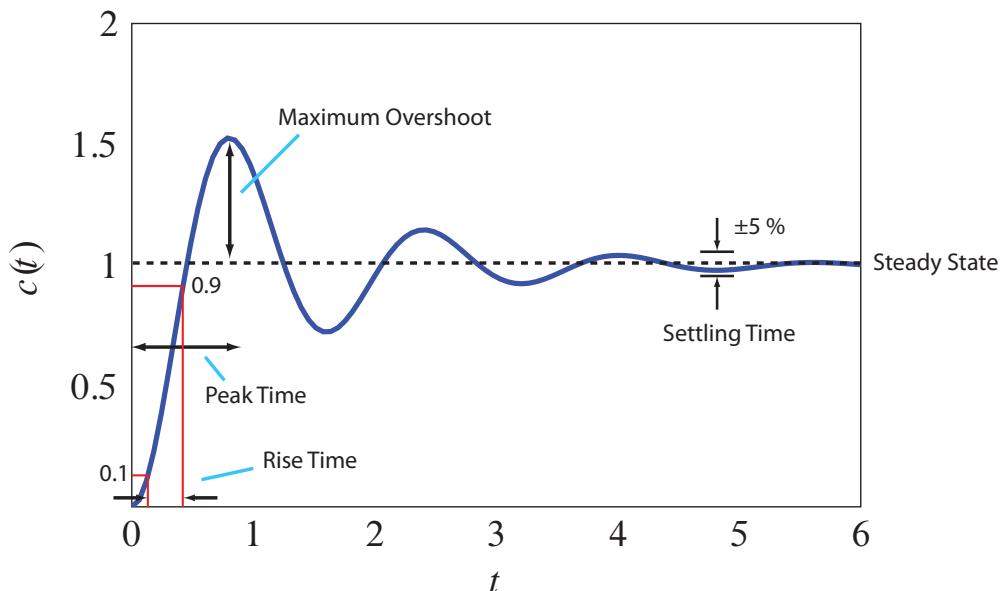


Figure 16.5 Metrics for characterizing a second-order response.

relationships are approximate:

$$10\% \text{ to } 90\% \text{ rise time } T_r = \frac{0.8 + 2.5\zeta}{\omega_n}$$

If we take an average $\zeta = 0.5$, then the rise time is given approximately by:

$$T_r = \frac{2}{\omega_n}$$

For underdamped systems ($\zeta < 1$), engineers will often use the time it takes to go from 0% to 100%, that is when the curve first crosses the final value. Note that given the unit step input, the final value is 1.0. We can therefore define the rise time as the time it takes for the underdamped system to cross the final value line. Looking at equation (16.5) and setting $x(t) = 1$ it must be true that:

$$\sin\left(\omega_n \sqrt{1 - \zeta^2} T_r + \theta\right) = \sin(n\pi)$$

where we have substituted time for T_r . However there are multiple places where the curve crosses the 1.0 value which is the reason why the right-hand side is $\sin(n\pi)$ where n corresponds to the peaks in the response. We are interested in the first crossing point when $n = 1$. Taking \sin^{-1} on both sides yields:

$$\omega_n \sqrt{1 - \zeta^2} T_r + \theta = \pi$$

Solving for T_r gives:

$$100\% \text{ rise time } T_r = \frac{\pi - \tan^{-1} \sqrt{(1 - \zeta^2)/\zeta}}{\omega_d}$$

The rise time is inversely proportional to the natural frequency. Therefore shorter rise times require higher natural frequencies.

Peak Time: T_p

The peak time is the time required to reach the peak value of the step response. It can be estimated by setting the derivative of $x(t)$ to zero. After some algebra it is possible to show that:

$$T_p = \frac{\pi}{\omega_d}$$

As expected, the higher the natural frequency the short time it takes to reach the peak value.

Percentage Overshoot: M_p

The percentage overshoot is the percentage by which the first peak (at T_p) exceeds the steady state value. Substituting T_p into the equation for $x(t)$ we can compute the peak value and knowing the steady state solution we obtain:

$$M_p = 100 e^{-\frac{\zeta\pi}{\sqrt{1-\zeta^2}}}$$

Note that as ζ approaches one (at which point the overshoot should vanish), M_p approaches zero.

Settling Time: T_s

The settling time, T_s , is the time it takes for the response to enter the $\pm 5\%$ band around the steady state such that it stays within that band. The value can be computed by considering the exponential envelope that bounds the damped oscillation. By computing when the exponential decay comes within 5% of the steady state we can compute the settling time. For a 5% bound the setting time is given by:

$$T_s = \frac{3}{\zeta\omega_n}$$

and for a 2% bound:

$$T_s = \frac{4}{\zeta\omega_n}$$

Example 16.1

A second-order system has a transfer function of:

$$h(s) = \frac{9}{s^2 + 3s + 9}$$

what is the natural frequency, ω_n and the damping ratio, ζ ? since $\omega_n^2 = 9$, then $\omega = 3$. $2\zeta\omega_n = 3$, therefore $\zeta = 3/(2\omega_n) = 0.5$

Example 16.2

A second-order system has a transfer function of:

$$H(s) = \frac{25}{s^2 + 4s + 25}$$

Find the rise time, peak time, settling time and maximum overshoot. Since $\omega_n^2 = 25$, then $\omega = 5$. $2\zeta\omega_n = 4$, therefore $\zeta = 3/(2\omega_n) = 0.4$

The rise time is: $\frac{\pi - \tan^{-1} \sqrt{(1-\zeta^2)/\zeta}}{\omega_d}$ so that:

$$T_r = \frac{\pi - \tan^{-1} \sqrt{(1 - 0.4^2)/0.4}}{5\sqrt{1 - 0.4^2}} = 0.475s$$

$$T_p = \frac{\pi}{\omega_n \sqrt{1 - \zeta^2}} = \frac{\pi}{5\sqrt{1 - 0.16}} = 0.686s$$

$$T_s = \frac{4}{\zeta\omega_n} = \frac{4}{(0.4)(0.5)} = 2s$$

$$M_p = 100 e^{-\frac{\pi\zeta}{\sqrt{1-\zeta^2}}} = 100 e^{-\frac{0.4\pi}{\sqrt{1-0.16}}} = 25.4\%$$

16.7 Frequency Response

The frequency response for a second-order system is much more interesting compared to a first-order system. The plots shown in Figure 16.6 and 16.7 show the magnitude and phase shift as a function of frequency. The first observation to note is that the phase change now shifts up to 180 degrees. This compares to 90 degrees for a first-order system. Each increase in order adds 90 degrees to the phase shift. Therefore a third order system can shift the signal up to 270 degrees.

The second and more obvious difference is in the magnitude or amplitude plot. We can see that the amplitude response shows a pronounced resonance peak. This peak gets stronger as ζ gets smaller. The frequency at which the peak occurs is called the **resonant frequency**. It is possible to show that the resonant frequency, ω_r occurs at:

$$\omega_r = \omega_n \sqrt{1 - 2\zeta^2}$$

If we look carefully at the amplitude plot, we can see the peak frequency drifts lower as ζ increases. In the limit as ζ approaches zero the peak approaches ω_n . Note that the equation can only be used for $\zeta < 0.707$. This is because the root term goes negative for $\zeta > 0.707$. For $\zeta > 0.707$ the response shows no peak in the amplitude.

The amplitude of the peak can be shown to equal:

$$M_r = \frac{1}{2} \frac{1}{\zeta \sqrt{1 - \zeta^2}}$$

Note that for $\zeta > 0.707$, $M_r = 1$. As ζ approaches zero, M_r approaches infinity.

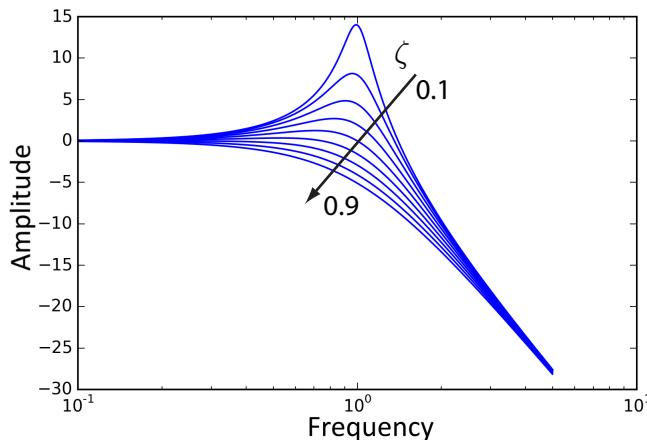


Figure 16.6 Amplitude plot for a second-order system with varying ζ : $0.1 < \zeta \leq 0.9$. See Python script 16.1

16.8 Example

In a later chapter we will look at the system shown in Figure 16.8 and derive the following transfer functions:

$$\mathbf{H}(j\omega) = \begin{pmatrix} \frac{k_1}{k_2 + jw} \\ \frac{k_1 k_3}{(jw + k_2)(jw + k_4)} \end{pmatrix} \quad (16.6)$$

We note that the transfer function with respect to x_2 is second-order. If we look at the characteristic equation:

$$(s + k_2)(s + k_4) = 0$$

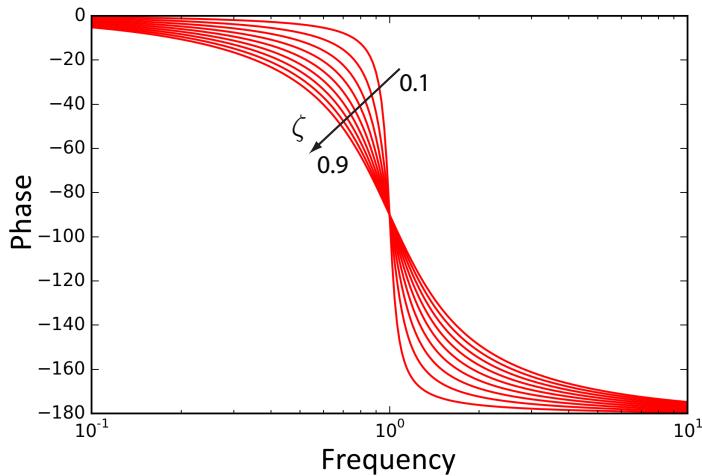


Figure 16.7 Phase plot for a second-order system with varying ζ : $0.1 < \zeta \leq 0.9$. See Python script 16.1

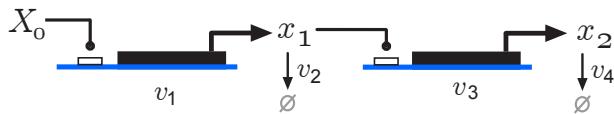


Figure 16.8 Two Gene Network Cascade

We see immediately that the poles are given by:

$$s_1 = -k_2 \quad s_2 = -k_4$$

These are real distinct poles assuming $k_2 \neq k_4$. Referring to the properties of a second-order system we see that this corresponds (16.5) to an overdamped system. If both rate constants are equal then the system will be a critically damped system. In either case we do *not* see any oscillations. We therefore conclude that a two step gene regulatory network cannot display oscillatory behavior. What is the relationship to the damping ratio and natural frequency? Beginning with:

$$H(s) = \frac{k_1 k_3}{s^2 + (k_2 + k_4)s + k_2 k_4}$$

Divide top and bottom by $k_2 k_4$:

$$H(s) = \frac{k_1 k_3 / (k_2 k_4)}{\frac{1}{k_2 k_4} + \frac{k_2 + k_4}{k_2 k_4} s + 1}$$

where we will replace the term $k_1k_3/(k_2k_4)$ by K . Let us set the term $1/(k_2k_4) = \omega_n^2$ such that $1/\omega_n = \sqrt{k_2k_4}$. We can write the transfer function as:

$$H(s) = \frac{K}{\frac{s^2}{\omega_n^2} + \frac{1}{\omega_n^2}(k_2 + k_4)s + 1}$$

We will define $k_2 + k_4 = 2\xi\omega_n$, therefore:

$$H(s) = \frac{K}{\frac{s^2}{\omega_n^2} + \frac{2\xi\omega_n}{\omega_n^2}s + 1}$$

Multiply top and bottom by ω_n^2 yields:

$$H(s) = K \frac{\omega_n^2}{s^2 + 2\xi\omega_n s + \omega_n^2}$$

This final result corresponds to the standard form given at the start of the chapter (Equation 16.2) but with a scaling factor K . To go further, since we've already stated that $k_2 + k_4 = 2\xi\omega_n$ and $\omega_n = \sqrt{k_2k_4}$, we can write:

$$\xi = \frac{1}{2} \frac{k_2 + k_4}{\sqrt{k_2k_4}} \quad (16.7)$$

In order to show that the system cannot oscillate we need to prove that $\xi \geq 1$. Let us note the following expression:

$$(\sqrt{x} - \sqrt{y})^2 \geq 0$$

That is:

$$x + y - 2\sqrt{xy} \geq 0$$

is true. By the rules of inequalities:

$$x + y \geq 2\sqrt{xy}$$

Now divide both sides by $2\sqrt{xy}$ to yield:

$$\frac{1}{2} \frac{x + y}{\sqrt{xy}} \geq 1$$

This last result has the same structure as equation 16.7. Therefore we conclude that:

$$\xi = \frac{1}{2} \frac{k_2 + k_4}{\sqrt{k_2k_4}} \geq 1$$

Since ξ must be greater than one (no matter what positive values are assigned to k_2 and k_4), the two stage genetic circuit shown in Figure 16.8 cannot oscillate in any form and can only admit critically damped or overdamped behavior.

Further Reading

1. Tripathi A N (1987) Linear Systems Theory. John Wiley & Sons ISBN: 0-470-20354-4
2. Houpis C, Sheldon S N and D'Azzo J J (2003) Linear Control System Analysis and Design. CRC Press 5th Edition ISBN-10: 0824740386

Exercises

1. Write down the general form of a transfer function that describes a second-order system. Name each of the terms in the transfer function.
2. Determine the roots for the characteristic equation of a second-order transfer function.
3. Describe the different behaviors that a second order system can display.
4. What is the difference between an overdamped and critically damped.
5. Given the following transfer functions, indicate the type of behavior you might expect.

$$1) H(s) = \frac{12}{s^2 + 8s + 12}$$

$$2) H(s) = \frac{16}{s^2 + 8s + 16}$$

$$2) H(s) = \frac{20}{s^2 + 8s + 20}$$

6. Consider a gene network in Figure 16.8, if the rate constants are set to $k_2 = 0.1, k_4 = 0.3$, estimate the rise time for the system. How would you reduce the rise time?

16.9 Appendix

Show that

$$x(t) = 1 - e^{-\xi\omega_n t} \left(\frac{\xi}{\sqrt{1-\xi^2}} \sin(\omega_d t) + \cos(\omega_d t) \right)$$

is equal to:

$$x(t) = 1 - \frac{e^{-\xi\omega_n t}}{\sqrt{1-\xi^2}} \sin \left(\omega_n \sqrt{1-\xi^2} t + \tan^{-1} \frac{\sqrt{1-\xi^2}}{\xi} \right)$$

The proof relies on the trigonometric identity:

$$\sin(\alpha + \beta) = \sin \alpha \cos \beta + \cos \alpha \sin \beta$$

Note that the term $\tan^{-1} \frac{\sqrt{1-\xi^2}}{\xi}$ (equivalent to β in the trigonometric identity) is not a function of ω_n and hence can be treated as a constant. Consider first the equation:

$$A \sin \alpha + B \cos \beta$$

Multiplying both sides of the trigonometric identity by a factor F and rearranging:

$$F \sin(\alpha + \beta) = F \cos \beta \sin \alpha + F \sin \beta \cos \alpha$$

Since $F \cos \beta$ and $F \sin \beta$ are constants, comparing this equation to $A \sin \alpha + B \cos \beta$ we can state that:

$$\begin{aligned} B &= F \sin \beta \\ A &= F \cos \beta \end{aligned}$$

Dividing the first equation by the second yields:

$$\frac{B}{A} = \frac{F \sin \beta}{F \cos \beta} = \tan \beta$$

Therefore:

$$\beta = \tan^{-1} \frac{B}{A}$$

Next, sum the squares of both terms:

$$A^2 + B^2 = F^2 \sin^2 \beta + F^2 \cos^2 \beta = F^2 (\sin^2 \beta + \cos^2) = F^2$$

Therefore:

$$F = \sqrt{A^2 + B^2}$$

Given:

$$x(t) = 1 - e^{-\xi \omega_n t} \left(\frac{\xi}{\sqrt{1 - \xi^2}} \sin(\omega_d t) + \cos(\omega_d t) \right)$$

and isolating the term:

$$\left(\frac{\xi}{\sqrt{1 - \xi^2}} \sin(\omega_d t) + \cos(\omega_d t) \right)$$

We see this matches the expression: $A \sin \alpha + B \cos \beta$. Therefore the equivalents term for A and B are:

$$A = \frac{\xi}{\sqrt{1 - \xi^2}} \quad B = 1$$

The β term in $F \sin(\alpha + \beta)$ is equivalent to the $\cos(\omega_d t)$ term where β is given by:

$$\beta = \tan^{-1} \frac{1}{\frac{\zeta}{\sqrt{1-\zeta^2}}} = \tan^{-1} \frac{\sqrt{1-\zeta^2}}{\zeta}$$

The α term is equal to $\omega_d t$ where $\omega_d = \omega_a \sqrt{1 - \zeta^2}$. Finally the F factor is equal to

$$F = \sqrt{A^2 + B^2} = \sqrt{\frac{\zeta^2}{1 - \zeta^2} + 1} = \frac{1}{\sqrt{1 - \zeta^2}}$$

16.10 Appendix

```

import numpy as np
from scipy import signal
from matplotlib import pyplot as plt

# Coefficients in numerator of transfer function
num = [1]
# Coefficients in denominator of transfer function
# High order to low order, eg 1*s^2 + 0.1*s + 1
den = [1, 0.1, 1]

# Scan over zeta, a parameter for a second-order system
zetaRange = np.arange(0.1, 1.1, 0.1)

f1 = plt.figure()
for i in range(0,9):
    den = [1, 2*zetaRange[i], 1]
    print den
    s1 = signal.lti(num, den)
    % Specify our own frequency range: np.arange(0.1, 5, 0.01)
    w, mag, phase = signal.bode(s1, np.arange(0.1, 5, 0.01).tolist())
    plt.semilogx (w, mag, color="blue", linewidth="1")
plt.xlabel ("Frequency")
plt.ylabel ("Magnitude")
plt.savefig ("c:\\mag.png", dpi=300, format="png")

plt.figure()

for i in range(0,9):
    den = [1, zetaRange[i], 1]

```

```
s1 = signal.lti(num, den)
w, mag, phase = signal.bode(s1, np.arange(0.1, 10, 0.02).tolist())
plt.semilogx (w, phase, color="red", linewidth="1.1")
plt.xlabel ("Frequency")
plt.ylabel ("Phase")
plt.savefig ("c:\\phase.png", dpi=300, format="png")
```

Listing 16.1 Code used to generate Figures 16.6 16.7

17

Cellular Networks

“Machines take me by surprise with great frequency.”

– Alan Turing, Computing machinery and intelligence. *Mind*, 59, 433-460, 1950.

17.1 Systems Equation

As described in Chapter 4, equation that describes cellular networks (4.28) is given by:

$$\frac{d\mathbf{x}}{dt} = \mathbf{Nv}(\mathbf{x}, \mathbf{p}) \quad (17.1)$$

where \mathbf{x} is a vector of species, \mathbf{N} the stoichiometry matrix, \mathbf{v} the vector of rates and \mathbf{p} a vector of parameters or inputs. This system is often nonlinear which means in general there are no closed solutions. When engineers are confronted with an intractable set of equations, they will **linearize**.

In previous chapters we saw that linearization of a model yielded the state space representation, that is:

$$\begin{aligned}\frac{d\mathbf{x}}{dt} &= \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{p}(t) \\ \mathbf{y}(t) &= \mathbf{C}\mathbf{x}(t) + \mathbf{D}\mathbf{p}(t)\end{aligned}$$

where $\mathbf{x}(t)$ represents the internal state variables and $\mathbf{y}(t)$ the output variables. \mathbf{p} represent the parameters and inputs to the system. As a reminder (See Chapter 8), The \mathbf{A} matrix is called the state matrix (also called the Jacobian) of size $m \times m$ where m is the number of state variables. The \mathbf{B} matrix is

called the control matrix of size $m \times q$ where q is the number of parameters or inputs. The \mathbf{C} matrix is called the output matrix of size $r \times m$ where r is the number of output variables and the \mathbf{D} matrix is called the feed-forward matrix of size $r \times q$. The question to address in this chapter is how we can use the state space representation to describe cellular models that are in the form 17.1. Let us first linearize equation (17.1).

A note on Matrix and Vector Calculus

$$\begin{aligned}\frac{\partial A\mathbf{x}}{\partial \mathbf{x}} &= A \\ \frac{\partial A\mathbf{f}(\mathbf{x})}{\partial \mathbf{x}} &= A \frac{\partial \mathbf{f}}{\partial \mathbf{x}} \\ \frac{\partial(\mathbf{u} + \mathbf{v})}{d\mathbf{x}} &= \frac{\partial \mathbf{u}}{\partial \mathbf{x}} + \frac{\partial \mathbf{v}}{\partial \mathbf{x}} \\ \frac{\partial A\mathbf{f}(\mathbf{x}, \mathbf{y})}{\partial \mathbf{x}} &= A \frac{\partial \mathbf{f}}{\partial \mathbf{x}} d\mathbf{x} + A \frac{\partial \mathbf{f}}{\partial \mathbf{y}} d\mathbf{y}\end{aligned}$$

Linearizing Cellular State Space Equation

When we linearize we must linearize around a specific operating point of interest. The most common point of interest is the steady state. Let us therefore linearize the system equation around the steady state solution given by:

$$\mathbf{N}\mathbf{v}(\mathbf{x}_{ss}, \mathbf{p}_{ss}) = 0$$

where \mathbf{x}_{ss} is the vector of state variables at steady state and \mathbf{p}_{ss} the set of parameters and inputs that determine the given steady state. Taking note that the stoichiometry matrix \mathbf{N} is a constant and using the linearization equation (7.8) we obtain:

$$\frac{d\delta\mathbf{x}}{dt} = \mathbf{N} \frac{\partial \mathbf{v}(\mathbf{x}_{ss}, \mathbf{p}_{ss})}{\partial \mathbf{x}} \delta\mathbf{x} + \mathbf{N} \frac{\partial \mathbf{v}(\mathbf{x}_{ss}, \mathbf{p}_{ss})}{\partial \mathbf{p}} \delta\mathbf{p} \quad (17.2)$$

If m is the number of species, n the number of reactions and p the number of parameter inputs, then the dimensions the various matrices will be:

$$\frac{d\delta\mathbf{x}}{dt} \Rightarrow m \times 1$$

$$\mathbf{N} \Rightarrow n \times m$$

$$\frac{\partial \mathbf{v}(\mathbf{x}_{ss}, \mathbf{p}_{ss})}{\partial \mathbf{x}} \Rightarrow n \times m$$

$$\frac{\partial \mathbf{v}(\mathbf{x}_{ss}, \mathbf{p}_{ss})}{\partial \mathbf{p}} \Rightarrow n \times p$$

As with any linearization, the result is only approximate for small changes in $\delta\mathbf{x}$ and $\delta\mathbf{p}$. Note that the equation includes the stoichiometry matrix and two partial derivatives and represents a set of linear ordinary differential equations. Such equations can be solved where the initial conditions are given by $\delta\mathbf{x}(0) = \delta\mathbf{x}_o$. The set of parameters and inputs will be $\delta\mathbf{p} = \delta\mathbf{p}_o$.

From the linearized equation 17.2, it is possible to define the \mathbf{A} and \mathbf{B} matrices for a cellular model in the state space representation. Given the linearized equation:

$$\frac{d\delta\mathbf{x}}{dt} = \mathbf{N} \frac{\partial \mathbf{v}(\mathbf{x}_{ss}, \mathbf{p}_{ss})}{\partial \mathbf{x}} \delta\mathbf{x} + \mathbf{N} \frac{\partial \mathbf{v}(\mathbf{x}_{ss}, \mathbf{p}_{ss})}{\partial \mathbf{p}} \delta\mathbf{p}$$

we see that the \mathbf{A} matrix or Jacobian, is given by:

$$\mathbf{A} = \mathbf{N} \frac{\partial \mathbf{v}(\mathbf{x}_{ss}, \mathbf{p}_{ss})}{\partial \mathbf{x}}$$

where the term:

$$\frac{\partial \mathbf{v}(\mathbf{x}_{ss}, \mathbf{p}_{ss})}{\partial \mathbf{x}}$$

is the matrix of unscaled elasticities:

$$\frac{\partial \mathbf{v}}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial v_1}{\partial x_1} & \frac{\partial v_1}{\partial x_2} & \cdots & \frac{\partial v_1}{\partial x_m} \\ \frac{\partial v_2}{\partial x_1} & \frac{\partial v_2}{\partial x_2} & \cdots & \frac{\partial v_2}{\partial x_m} \\ \vdots & & & \\ \frac{\partial v_n}{\partial x_1} & \frac{\partial v_n}{\partial x_2} & \cdots & \frac{\partial v_n}{\partial x_m} \end{bmatrix}$$

The \mathbf{B} matrix is:

$$\mathbf{B} = \mathbf{N} \frac{\partial \mathbf{v}(\mathbf{x}_{ss}, \mathbf{p}_{ss})}{\partial \mathbf{p}}$$

We can now state that the state space representation for a cellular network is given by the equation:

$$\frac{d\delta \mathbf{x}}{dt} = \mathbf{N} \frac{\partial \mathbf{v}}{\partial \mathbf{x}} \delta \mathbf{x}(t) + \mathbf{N} \frac{\partial \mathbf{v}}{\partial \mathbf{p}} \delta \mathbf{p}(t)$$

Taking the Laplace transform of this equation will give us the *transfer function* for this system as:

$$\mathbf{H}(s) = \left(s\mathbf{I} - \mathbf{N} \frac{\partial \mathbf{v}}{\partial \mathbf{x}} \right)^{-1} \mathbf{N} \frac{\partial \mathbf{v}}{\partial \mathbf{p}} \quad (17.3)$$

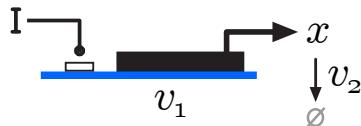
or as we've seen in a previous chapter (11.15).

$$\mathbf{H}(s) = (s\mathbf{I} - \mathbf{A})^{-1} \mathbf{B} \quad (17.4)$$

From now on we will use the symbol $\mathbf{H}_x(s)$ to designate the transfer function with respect to the state variables \mathbf{x} .

Example 17.1

Consider the simple gene regulatory network with a single transcription factor, x :



We will assume that the expression rate for x is controlled by an inducer I . We will also assume that the first step is governed by the rate law $v_1 = k_1 I$ and the degradation step by $v_2 = k_2 x$.

We will compute the transfer function given by:

$$\mathbf{H}_x(s) = \left(s\mathbf{I} - \mathbf{N} \frac{\partial \mathbf{v}}{\partial \mathbf{x}} \right)^{-1} \mathbf{N} \frac{\partial \mathbf{v}}{\partial \mathbf{p}}$$

First we need to collect the three matrix terms, \mathbf{N} , $\partial \mathbf{v}/\partial \mathbf{x}$ and $\partial \mathbf{v}/\partial \mathbf{p}$.

$$\mathbf{N} = [1 \ -1]$$

$$\frac{\partial \mathbf{v}}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial v_1}{\partial x} \\ \frac{\partial v_2}{\partial x} \end{bmatrix} = \begin{bmatrix} 0 \\ k_2 \end{bmatrix}$$

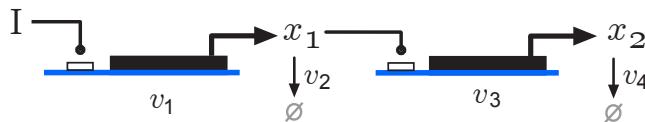
$$\frac{\partial \mathbf{v}}{\partial \mathbf{p}} = \begin{bmatrix} \frac{\partial v_1}{\partial I} \\ \frac{\partial v_2}{\partial I} \end{bmatrix} = \begin{bmatrix} k_1 \\ 0 \end{bmatrix}$$

Inserting these into the transfer function yields:

$$\begin{aligned} \mathbf{H}_x(s) &= \left(s - [1 - 1] \begin{bmatrix} 0 \\ k_2 \end{bmatrix} \right)^{-1} [1 - 1] \begin{bmatrix} k_1 \\ 0 \end{bmatrix} \\ &= (s + k_2)^{-1} k_1 = \frac{k_1}{s + k_2} \end{aligned}$$

Example 17.2

Consider the two gene cascade regulatory network with transcription factors, x_1 and x_2 :



v_1 represent the expression rate for protein x_1 . v_2 represents the degradation rate of x_1 . v_3 is the expression rate of protein x_2 . v_4 is the degradation rate of x_2 . I is the input inducer to the cascade. We will assume that there are two inputs that can perturb the activity of v_1 and v_3 . We will compute the transfer function given by:

$$\mathbf{H}_x(s) = \left(sI - \mathbf{N} \frac{\partial \mathbf{v}}{\partial \mathbf{x}} \right)^{-1} \mathbf{N} \frac{\partial \mathbf{v}}{\partial \mathbf{p}}$$

First we need to collect the three matrix terms, \mathbf{N} , $\partial \mathbf{v}/\partial \mathbf{x}$ and $\partial \mathbf{v}/\partial \mathbf{p}$.

$$\mathbf{N} = \begin{bmatrix} 1 & -1 & 0 & 0 \\ 0 & 0 & 1 & -1 \end{bmatrix}$$

$$\frac{\partial \mathbf{v}}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial v_1}{\partial x_1} & \frac{\partial v_1}{\partial x_2} \\ \frac{\partial v_2}{\partial x_1} & \frac{\partial v_2}{\partial x_2} \\ \frac{\partial v_3}{\partial x_1} & \frac{\partial v_3}{\partial x_2} \\ \frac{\partial v_4}{\partial x_1} & \frac{\partial v_4}{\partial x_2} \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ \frac{\partial v_2}{\partial x_1} & 0 \\ \frac{\partial v_3}{\partial x_1} & 0 \\ 0 & \frac{\partial v_4}{\partial x_2} \end{bmatrix}$$

In the $\partial \mathbf{v} / \partial \mathbf{p}$ the input parameter only has an effect on v_1 , therefore the matrix has only a single non-zero entry:

$$\frac{\partial \mathbf{v}}{\partial \mathbf{p}} = \begin{bmatrix} \frac{\partial v_1}{\partial p_1} & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$$

Inserting these into the transfer function yields:

$$\begin{aligned} D &= \frac{\partial v_2}{\partial x_1} \frac{\partial v_4}{\partial x_2} + \frac{\partial v_2}{\partial x_1} s + \frac{\partial v_4}{\partial x_1} s + s^2 \\ \mathbf{H}(s) &= \begin{bmatrix} s + \frac{\partial v_2}{\partial x_1} & 0 \\ -\frac{\partial v_3}{\partial x_1} & s + \frac{\partial v_4}{\partial x_2} \end{bmatrix}^{-1} \begin{bmatrix} \frac{\partial v_1}{\partial p_1} \end{bmatrix} \\ &= \frac{1}{D} \begin{bmatrix} s + \frac{\partial v_4}{\partial x_2} & 0 \\ -\frac{\partial v_3}{\partial x_1} & s + \frac{\partial v_2}{\partial x_1} \end{bmatrix} \begin{bmatrix} \frac{\partial v_1}{\partial p_1} \end{bmatrix} \\ &= \frac{1}{D} \begin{bmatrix} \left(s + \frac{\partial v_4}{\partial x_2} \right) \frac{\partial v_1}{\partial p_1} \\ -\frac{\partial v_3}{\partial x_1} \frac{\partial v_1}{\partial p_1} \end{bmatrix} \end{aligned}$$

17.2 Output Equations

So far we have only considered the state space equation related to the state variables, \mathbf{x} . Here we will consider the output component, \mathbf{y} , that is:

$$\mathbf{y}(t) = \mathbf{C}\mathbf{x}(t) + \mathbf{D}\mathbf{p}(t)$$

There many possibilities for selecting a possible output. Such choices depend on the application in mind. Here we will consider the reaction rates, or fluxes, as one possible set of output measures. The reaction rate for a given step is given by:

$$\mathbf{v} = \mathbf{v}(\mathbf{x}, \mathbf{p})$$

If we linearize this equation we obtain:

$$\mathbf{v} = \frac{\partial \mathbf{v}}{\partial \mathbf{x}} \delta \mathbf{x} + \frac{\partial \mathbf{v}}{\partial \mathbf{p}} \delta \mathbf{p}$$

Taking the Laplace transform gives:

$$\mathcal{L}(\mathbf{v}) = V(s) = \mathbf{C}X(s) + \mathbf{D}\mathbf{P}(s)$$

where $\mathbf{C} = \partial \mathbf{v} / \partial \mathbf{x}$ and $\mathbf{D} = \partial \mathbf{v} / \partial \mathbf{p}$. But from equation 11.14 $X(s) = (s\mathbf{I} - \mathbf{A})^{-1} \mathbf{B} \mathbf{P}(s)$ so that:

$$V(s) = [\mathbf{C}(s\mathbf{I} - \mathbf{A})^{-1} \mathbf{B} + \mathbf{D}] \mathbf{P}(s)$$

Hence the transfer function, $\mathbf{H}_J(s)$ is given by:

$$\mathbf{H}_J(s) = \mathbf{C}(s\mathbf{I} - \mathbf{A})^{-1} \mathbf{B} + \mathbf{D}$$

or (See equation 17.4):

$$\mathbf{H}_J(s) = \mathbf{C} \mathbf{H}_x(s) + \mathbf{D}$$

Substituting the various state space terms with the network terms yields:

$$\mathbf{H}_J(s) = \frac{\partial \mathbf{v}}{\partial \mathbf{x}} \left(s\mathbf{I} - \mathbf{N} \frac{\partial \mathbf{v}}{\partial \mathbf{x}} \right)^{-1} \mathbf{N} \frac{\partial \mathbf{v}}{\partial \mathbf{p}} + \frac{\partial \mathbf{v}}{\partial \mathbf{p}} \quad (17.5)$$

A subscript, J , has been added to the transfer function to indicate that this function refers to the flux rather than the individual state variables. This is to distinguish it from the state variable transfer function, $\mathbf{H}_x(s)$. Finally we can substitute $\mathbf{H}_x(s)$ in to the above equation to yield:

$$\mathbf{H}_J(s) = \frac{\partial \mathbf{v}}{\partial \mathbf{x}} \mathbf{H}_x(s) + \frac{\partial \mathbf{v}}{\partial \mathbf{p}} \quad (17.6)$$

A discussion in Chapter 13 highlighted the importance of the transfer functions at zero frequency. This was shown to be the derivative of the steady state response when the input was a step input (Equation 13.7).

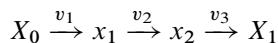
$$\mathbf{H}_x(0) = \frac{d\mathbf{x}}{d\mathbf{p}}$$

A similar interpretation can be made for flux response so that:

$$\mathbf{H}_J(0) = \frac{d\mathbf{J}}{d\mathbf{p}}$$

Example 17.3

The use of the flux transfer function, $\mathbf{H}_J(s)$ makes most sense when applied to metabolic pathways. Consider the following metabolic sequence of enzyme catalyzed reactions:



We will assume that the metabolites, X_o and X_1 are fixed and that v_1 to v_3 are governed by the rate laws:

$$v_1 = E_1(k_1 X_o - k_2 x_1)$$

$$v_2 = E_2(k_3 S_1 - k_4 x_2)$$

$$v_3 = E_3 k_5 x_2$$

where E_i is the concentration of the i^{th} enzyme and k_i are the kinetic rate constants. m the number of state variable is two and n the number of reactions is three. We will assign three inputs to the model, corresponding to step functions in E_1 , E_2 and E_3 . The number of inputs, q , is therefore three.

First we write down the stoichiometry matrix for this system. This will be a two by three matrix. Note that since X_o and X_1 are fixed they will not appear in the stoichiometry matrix:

$$\mathbf{N} = \begin{bmatrix} 1 & -1 & 0 \\ 0 & 1 & -1 \end{bmatrix}$$

We must also write down the $\partial \mathbf{v} / \partial \mathbf{x}$ which will be a n by m matrix:

$$\frac{\partial \mathbf{v}}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial v_1}{\partial x_1} & \frac{\partial v_1}{\partial x_2} \\ \frac{\partial v_2}{\partial x_1} & \frac{\partial v_2}{\partial x_2} \\ \frac{\partial v_3}{\partial x_1} & \frac{\partial v_3}{\partial x_2} \end{bmatrix}$$

Two of the terms in this matrix are zero so that:

$$\frac{\partial \mathbf{v}}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial v_1}{\partial x_1} & 0 \\ \frac{\partial v_2}{\partial x_1} & \frac{\partial v_2}{\partial x_2} \\ 0 & \frac{\partial v_3}{\partial x_2} \end{bmatrix}$$

In addition $\partial \mathbf{v}/\partial \mathbf{p}$ will be a n by q matrix:

$$\frac{\partial \mathbf{v}}{\partial \mathbf{p}} = \begin{bmatrix} \frac{\partial v_1}{\partial E_1} & \frac{\partial v_1}{\partial E_2} & \frac{\partial v_1}{\partial E_3} \\ \frac{\partial v_2}{\partial E_1} & \frac{\partial v_2}{\partial E_2} & \frac{\partial v_2}{\partial E_3} \\ \frac{\partial v_3}{\partial E_1} & \frac{\partial v_3}{\partial E_2} & \frac{\partial v_3}{\partial E_3} \end{bmatrix}$$

Six of the terms in the $\partial \mathbf{v}/\partial \mathbf{p}$ matrix are however zero, so that:

$$\frac{\partial \mathbf{v}}{\partial \mathbf{p}} = \begin{bmatrix} \frac{\partial v_1}{\partial E_1} & 0 & 0 \\ 0 & \frac{\partial v_2}{\partial E_2} & 0 \\ 0 & 0 & \frac{\partial v_3}{\partial E_3} \end{bmatrix}$$

Next we construct the $\mathbf{H}_x(s)$ transfer function (17.4), given by $\mathbf{H}_x(s) = (s\mathbf{I} - \mathbf{A})^{-1}\mathbf{B}$, using the relations:

$$\mathbf{A} = \mathbf{N} \frac{\partial \mathbf{v}}{\partial s}$$

$$\mathbf{B} = \mathbf{N} \frac{\partial \mathbf{v}}{\partial \mathbf{p}}$$

$$D = \frac{dv_2}{dx_1} \frac{dv_2}{dx_2} + \left(\frac{dv_2}{dx_1} - \frac{dv_1}{dx_1} + s \right) \left(\frac{dv_3}{dx_2} - \frac{dv_2}{dx_2} + s \right)$$

$$\mathbf{H}_x(s) = \frac{1}{D} \times$$

$$\begin{bmatrix} \frac{dv_1}{dE_1} \left(\frac{dv_3}{dx_2} - \frac{dv_2}{dx_2} + s \right), & -\left(\frac{dv_3}{dx_2} + s \right) \frac{dv_2}{dE_2}, & \frac{dv_2}{dx_2} \frac{dv_3}{dE_3} \\ \frac{dv_2}{dx_1} \frac{dv_1}{dE_1}, & \frac{dv_2}{dE_2} \left(s - \frac{dv_1}{dx_1} \right), & -\frac{dv_3}{dE_3} \left(\frac{dv_2}{dx_1} - \frac{dv_1}{dx_1} + s \right) \end{bmatrix}$$

The flux transfer matrix, $\mathbf{H}_J(s)$ can be obtained by inserting $\mathbf{H}_x(s)$ into $\mathbf{C} \mathbf{H}_x(s) + \mathbf{D}$ where

$$\mathbf{C} = \frac{\partial \mathbf{v}}{\partial \mathbf{x}}$$

$$\mathbf{D} = \frac{\partial \mathbf{v}}{\partial \mathbf{p}}$$

These equations are unwieldy and we will show in the next couple of chapters how their structure can be greatly simplified to reveal useful information.

17.3 Canonical Transfer Functions

The $\mathbf{H}_x(s)$ and $\mathbf{H}_J(s)$ transfer functions are dependent on the particular parameter that is used to perturb the system. If we eliminate the parameter term we can define canonical transfer functions. These are defined as:

$$\mathcal{C}_x = \left(s\mathbf{I} - \mathbf{N} \frac{\partial \mathbf{v}}{\partial \mathbf{x}} \right)^{-1} \mathbf{N} \quad (17.7)$$

$$\mathcal{C}_J = \frac{\partial \mathbf{v}}{\partial \mathbf{x}} \mathcal{C}_x + \mathbf{I} \quad (17.8)$$

In the biology community, these canonical transfer functions are also referred to as the unscaled **concentration and flux control coefficients** when s is set to zero. We will return to this point in a later chapter.

Canonical transfer functions are parameter independent in that the particular parameter used to perturb the system is not explicitly specified. All that is assumed is that a reaction is perturbed, but by what is not indicated.

Example 17.4

Frequency Response

Let us examine the frequency response for the two variable system in Figure 17.1.

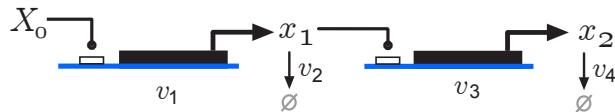


Figure 17.1 Two Gene Network Cascade

For the sake of argument, let us again assume simple kinetics at each step, thus $v_1 = k_1 X_0$, $v_2 = k_2 x_1$, $v_3 = k_3 x_1$ and $v_4 = k_4 x_2$.

Again we need to compute:

$$\mathbf{H}(j\omega) = \left(j\omega \mathbf{I} - \mathbf{N} \frac{\partial \mathbf{v}}{\partial \mathbf{x}} \right)^{-1} \mathbf{N} \frac{\partial \mathbf{v}}{\partial \mathbf{p}}$$

First we will collect the three matrix terms, \mathbf{N} , $\partial \mathbf{v}/\partial \mathbf{x}$ and $\partial \mathbf{v}/\partial \mathbf{p}$.

$$\mathbf{N} = \begin{bmatrix} 1 & -1 & 0 & 0 \\ 0 & 0 & 1 & -1 \end{bmatrix}$$

$$\frac{\partial \mathbf{v}}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial v_1}{\partial x_1} & \frac{\partial v_1}{\partial x_2} \\ \frac{\partial v_2}{\partial x_1} & \frac{\partial v_2}{\partial x_2} \\ \frac{\partial v_3}{\partial x_1} & \frac{\partial v_3}{\partial x_2} \\ \frac{\partial v_4}{\partial x_1} & \frac{\partial v_4}{\partial x_2} \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ k_2 & 0 \\ k_3 & 0 \\ 0 & k_4 \end{bmatrix}$$

$$\frac{\partial \mathbf{v}}{\partial \mathbf{p}} = \begin{bmatrix} \frac{\partial v_1}{\partial X_o} \\ \frac{\partial v_2}{\partial X_o} \\ \frac{\partial v_3}{\partial X_o} \\ \frac{\partial v_4}{\partial X_o} \end{bmatrix} = \begin{bmatrix} k_1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

and insert the terms into the transfer function to yield:

$$\mathbf{H}(j\omega) = \begin{pmatrix} \frac{k_1}{k_2 + jw} \\ \frac{k_1 k_3}{(jw + k_2)(jw + k_4)} \end{pmatrix} \quad (17.9)$$

It is far easier however to compute the amplitude and phase numerically. Many computer languages provide a function called ArcTan2, or more commonly, atan2 that will compute the arcTangent of a value while taking into account the four quadrants, ie they will return values in the range $-\pi$ to π . These functions take two arguments, the first is the imaginary part and the second argument the real part. For example, to compute the magnitude and angle of a complex number (and converting radians to degrees) the following code can be used in MATLAB:

```
% MATLAB Code to compute the angle and magnitude of a complex number
z = 4 + 3 i
theta = atan2 (imag (z), real (z))*180/3.1415
magnitude = abs (z)
```

36.87

5.0

If we are using Python, the cmath package has a method called phase which can be used to compute the angle of a complex number. In turn, phase calls atan2, however phase saves one from splitting the complex number in the real and imaginary parts. In the code below the math method, degrees is used to convert radians into degrees.

```
# Equivalent Python code
import cmath
```

```

z = 4 + 3j
theta = math.degrees (cmath.phase (z))
magnitude = abs (z)

```

36.87

5.0

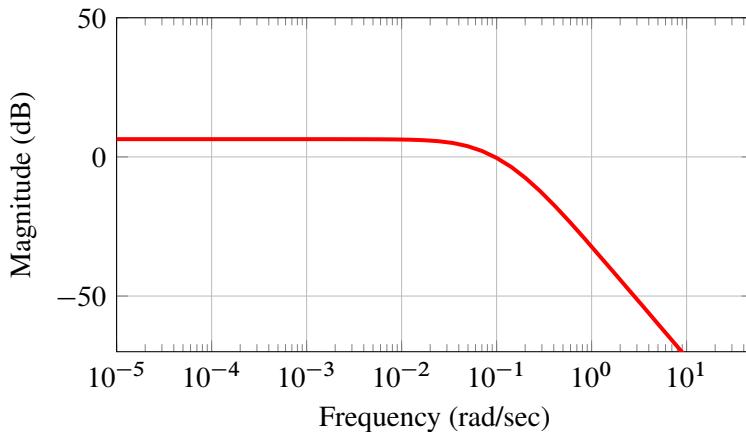


Figure 17.2 Bode Plot for a two gene cascade: Magnitude Plot. All rate laws are simple irreversible first-order with values: $v_1 : k_1 = 0.1, v_2 : k_2 = 0.2, v_3 : k_3 = 0.25, v_4 : k_4 = 0.06, X_o = 1$, see Figure 17.1

Flux Frequency Response

Since $\mathbf{v} = \mathbf{v}(\mathbf{x}, \mathbf{p})$ we can linearize this to give:

$$\mathbf{v} = \frac{\partial \mathbf{v}}{\partial \mathbf{x}} \delta \mathbf{x} + \frac{\partial \mathbf{v}}{\partial \mathbf{p}} \delta \mathbf{p}$$

Taking the Laplace Transform of this gives:

$$\mathcal{L}(\mathbf{v}) = \mathbf{C} \mathbf{X}(s) + \mathbf{D} \mathbf{P}(s)$$

But $\mathbf{X}(s) = (s\mathbf{I} - \mathbf{A})^{-1} \mathbf{B} \mathbf{P}(s)$ so that:

$$\mathbf{V}(s) = [\mathbf{C}(s\mathbf{I} - \mathbf{A})^{-1} \mathbf{B} + \mathbf{D}] \mathbf{P}(s)$$

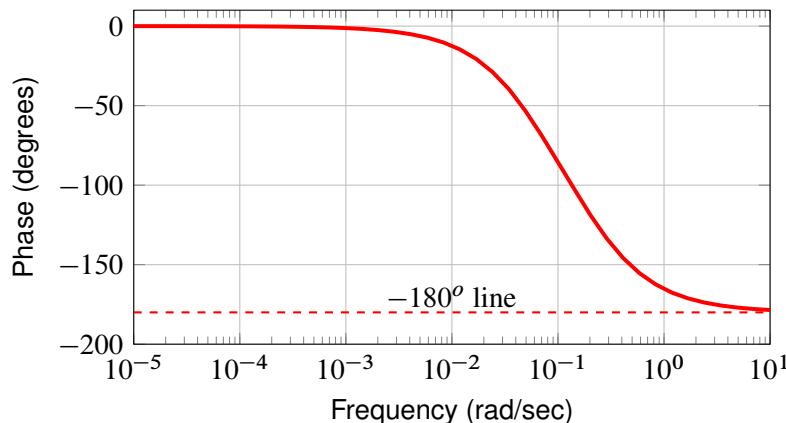


Figure 17.3 Bode Phase Plot: See Figure 17.2

The transfer function, $\mathbf{H}_J(s)$ is given by $\mathbf{V}(s)\mathbf{P}(s)^{-1}$:

$$\mathbf{H}_J(s) = \mathbf{C}(s\mathbf{I} - \mathbf{A})^{-1}\mathbf{B} + \mathbf{D}$$

Substituting the various state space terms with the network terms and setting $s = j\omega$, finally yields:

$$\mathbf{H}_J(j\omega) = \frac{\partial \mathbf{v}}{\partial \mathbf{x}} \left(j\omega \mathbf{I} - \mathbf{N} \frac{\partial \mathbf{v}}{\partial \mathbf{x}} \right)^{-1} \mathbf{N} \frac{\partial \mathbf{v}}{\partial \mathbf{p}} + \frac{\partial \mathbf{v}}{\partial \mathbf{p}}$$

18

Scaled Transfer Functions

On two occasions I have been asked, "Pray, Mr. Babbage, if you put into the machine wrong figures, will the right answers come out?" ... I am not able rightly to apprehend the kind of confusion of ideas that could provoke such a question.

– Charles Babbage, *Passages from the Life of a Philosopher*, 1864

18.1 Introduction

Example 17.3 showed how it was possible to derive the state variable and flux transfer functions for a small metabolic pathway. The result was a somewhat unwieldy set of matrix expressions which could be difficult to interpret. An alternative method and one that makes interpretation much simpler is the use of scaled transfer functions. In biology it is often more convenient to work with relative than absolute changes. This eliminates the need to be concerned with units and also makes it easier to compare measurements across different laboratories. We saw in Chapter 4 the introduction of the elasticity (4.14) which represents a scaled partial derivative. Since the transfer functions include partial derivatives such as $\partial v / \partial x$ and $\partial v / \partial p$ it should be possible to replace these with the scaled versions. In the process the transfer function becomes a scaled transfer function.

18.2 Scaled Transfer Functions

Let us first define some diagonal matrices:

$$\text{diag}(\mathbf{x}_o) = \begin{bmatrix} x_1 & 0 & \dots & 0 \\ 0 & x_2 & \dots & 0 \\ \vdots & & & \\ 0 & 0 & \dots & x_m \end{bmatrix} \quad \text{diag}(\mathbf{p}_o) = \begin{bmatrix} p_1 & 0 & \dots & 0 \\ 0 & p_2 & \dots & 0 \\ \vdots & & & \\ 0 & 0 & \dots & p_q \end{bmatrix}$$

$$\text{diag}(\mathbf{v}(\mathbf{x}_o, \mathbf{p}_o)) = \text{diag}(\mathbf{v}) = \begin{bmatrix} v_1 & 0 & \dots & 0 \\ 0 & v_2 & \dots & 0 \\ \vdots & & & \\ 0 & 0 & \dots & v_n \end{bmatrix}$$

where n is the number of reactions, m the number of molecular species and q the number of parameters (or inputs). \mathbf{x}_o and \mathbf{p}_o refer to the steady state values for the state variables and inputs. To reduce clutter, whenever we refer to \mathbf{x} or \mathbf{p} in the following equations, we are in fact referring to \mathbf{x}_o and \mathbf{p}_o . From these diagonal matrices we can define the scaled transfer functions as:

$$\tilde{\mathbf{H}}_x(s) = \text{diag}(\mathbf{x})^{-1} \mathbf{H}_x(s) \text{diag}(\mathbf{p})$$

$$\tilde{\mathbf{H}}_J(s) = \text{diag}(\mathbf{v})^{-1} \mathbf{H}_J(s) \text{diag}(\mathbf{p})$$

where the \tilde{H} indicates a scaled transfer function. We will also define the matrix of scaled elasticities as follows:

$$\boldsymbol{\varepsilon}_x = \text{diag}(\mathbf{v})^{-1} \frac{\partial \mathbf{v}}{\partial \mathbf{x}} \text{diag}(\mathbf{x})$$

$$\boldsymbol{\varepsilon}_p = \text{diag}(\mathbf{v})^{-1} \frac{\partial \mathbf{v}}{\partial \mathbf{p}} \text{diag}(\mathbf{p})$$

Scaled $H_x(0)$ Transfer Function

Let us now consider the scaled transfer function at zero frequency. At zero frequency we know (13.7) that $H(0) = d\mathbf{x}/d\mathbf{p}$, so that.

$$\frac{d\mathbf{x}}{d\mathbf{p}} = - \left(\mathbf{N} \frac{\partial \mathbf{v}}{\partial \mathbf{x}} \right)^{-1} \mathbf{N} \frac{\partial \mathbf{v}}{\partial \mathbf{p}}$$

We can pre-multiply by $\text{diag}(\mathbf{x})^{-1}$ and post-multiply by $\text{diag}(\mathbf{p})$ to yield:

$$\text{diag}(\mathbf{x})^{-1} \frac{d\mathbf{x}}{d\mathbf{p}} \text{diag}(\mathbf{p}) = -\text{diag}(\mathbf{x})^{-1} \left(\mathbf{N} \frac{\partial \mathbf{v}}{\partial \mathbf{x}} \right)^{-1} \mathbf{N} \frac{\partial \mathbf{v}}{\partial \mathbf{p}} \text{diag}(\mathbf{p})$$

The term on the left-side is the scaled transfer function, $\tilde{\mathbf{H}}_x(0)$. Multiply both sides by $\text{diag}(\mathbf{x})$ to yield:

$$\text{diag}(\mathbf{x}) \tilde{\mathbf{H}}_x(0) = - \left(\mathbf{N} \frac{\partial \mathbf{v}}{\partial \mathbf{x}} \right)^{-1} \mathbf{N} \frac{\partial \mathbf{v}}{\partial \mathbf{p}} \text{diag}(\mathbf{p})$$

Multiply both sides by:

$$\left(\mathbf{N} \text{diag}(\mathbf{v}) \text{diag}(\mathbf{v})^{-1} \frac{\partial \mathbf{v}}{\partial \mathbf{x}} \right)$$

to obtain:

$$\begin{aligned} \left(\mathbf{N} \text{diag}(\mathbf{v}) \text{diag}(\mathbf{v})^{-1} \frac{\partial \mathbf{v}}{\partial \mathbf{x}} \right) \text{diag}(\mathbf{x}) \tilde{\mathbf{H}}_x(0) &= -\mathbf{N} \frac{\partial \mathbf{v}}{\partial \mathbf{p}} \text{diag}(\mathbf{p}) \\ (\mathbf{N} \text{diag}(\mathbf{v}) \boldsymbol{\epsilon}_x) \tilde{\mathbf{H}}_x(0) &= -\mathbf{N} \text{diag}(\mathbf{v}) \text{diag}(\mathbf{v})^{-1} \frac{\partial \mathbf{v}}{\partial \mathbf{p}} \text{diag}(\mathbf{p}) \end{aligned}$$

where $\boldsymbol{\epsilon}_x$ is the matrix of scaled species elasticities and $\boldsymbol{\epsilon}_p$ is the scaled parameter elasticities. Finally multiplying both sides by $(\mathbf{N} \text{diag}(\mathbf{v}) \boldsymbol{\epsilon}_x)^{-1}$:

$$\tilde{\mathbf{H}}_x(0) = -(\mathbf{N} \text{diag}(\mathbf{v}) \boldsymbol{\epsilon}_x)^{-1} \mathbf{N} \text{diag}(\mathbf{v}) \boldsymbol{\epsilon}_p \quad (18.1)$$

From this we will define the scaled canonical transfer function (See Equation 17.8) by dropping $\boldsymbol{\epsilon}_p$:

$$\mathbf{C}_x(0) = -(\mathbf{N} \text{diag}(\mathbf{v}) \boldsymbol{\epsilon}_x)^{-1} \mathbf{N} \text{diag}(\mathbf{v}) \quad (18.2)$$

Scaled $H_J(0)$ Transfer Function

Starting with equation 17.6 we know that at $H_J(0) = d\mathbf{J}/d\mathbf{p}$ so that:

$$\frac{d\mathbf{J}}{d\mathbf{p}} = \frac{\partial \mathbf{v}}{\partial \mathbf{x}} \frac{d\mathbf{x}}{d\mathbf{p}} + \frac{\partial \mathbf{v}}{\partial \mathbf{p}}$$

Multiplying both sides by $\text{diag}(\mathbf{v})^{-1}$ and $\text{diag}(\mathbf{p})$ and inserting $\text{diag}(\mathbf{x})^{-1}\text{diag}(\mathbf{x})$ between the terms $\partial \mathbf{v}/\partial \mathbf{x}$ and $d\mathbf{x}/d\mathbf{p}$ yields after simplification:

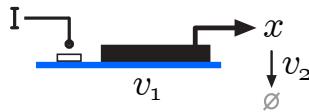
$$\tilde{\mathbf{H}}_J(0) = \boldsymbol{\epsilon}_x \tilde{\mathbf{H}}_x(0) + \boldsymbol{\epsilon}_p$$

Dropping the $\boldsymbol{\epsilon}_p$ gives us the scaled canonical transfer function for $\mathbf{C}_J(0)$ (Note $\mathbf{C}_x(0) = \mathbf{H}_x(0)\boldsymbol{\epsilon}_p$):

$$\mathbf{C}_J(0) = \boldsymbol{\epsilon}_x \mathbf{C}_x(0) + \mathbf{I}$$

Example 18.1

Compute the scaled frequency response for the single gene circuit:



We will assume that the expression rate for x is controlled by a inducer I at a rate $v_1 = f_1(I)$ and the degradation step by the rate $v_2 = f_2(x)$. Taking our cue from example 17.2 in the previous chapter, the transfer function for this system with respect to the input I can be shown to be:

$$\mathbf{H}(s) = \frac{\frac{\partial v_1}{\partial I}}{s + \frac{\partial v_2}{\partial x}} \quad (18.3)$$

Note that instead of k_1 and k_2 that was shown in example 17.2, we must use the unscaled elasticities explicitly since v_1 and v_2 are provided as functions $f_1()$ and $f_2()$. Multiplying both sides by I and dividing both sides by x yields:

$$\mathbf{H}(s) \frac{I}{x} = \frac{v_1 \frac{\partial v_1}{\partial I} \frac{I}{v_1}}{sx + v_2 \frac{\partial v_2}{\partial x} \frac{x}{v_2}} = \frac{v_1 \varepsilon_I^{v_1}}{sx + v_2 \varepsilon_x^{v_2}}$$

The left-hand side is the scaled transfer function, so that:

$$\tilde{\mathbf{H}}_x(s) = \frac{(v_1/x)\varepsilon_I^{v_1}}{s + (v_2/x)\varepsilon_x^{v_2}}$$

Figure 18.1 shows the Bode plot of the amplitude using equation (18.3). The dashed line shows what happens when the degradation rate constant is increased. This results in a much reduced gain at low frequencies but a slightly extended bandwidth.

Effect of Degradation Rate Constant

The previous example showed that the scaled transfer function for the transcription factor, x with respect to the inducer, I , was given by:

$$\tilde{\mathbf{H}}_x(s) = \frac{(v_1/x)\varepsilon_I^{v_1}}{s + (v_2/x)\varepsilon_x^{v_2}}$$

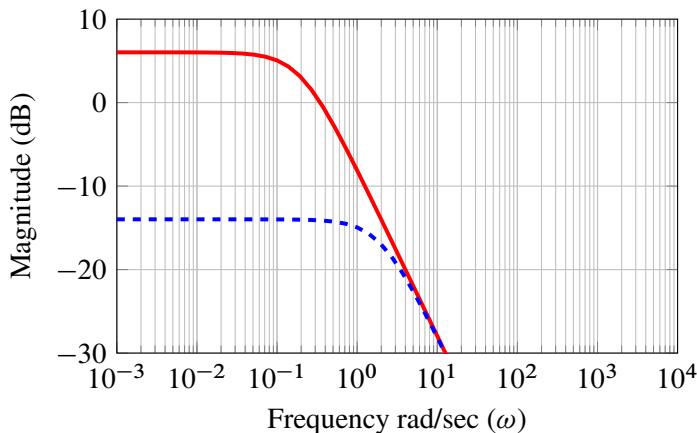


Figure 18.1 Bode Plot: Magnitude (dB). Effect (dashed line) of a higher degradation rate constant, ε_x^2 . Results in a reduced gain at low frequencies and slight increase in the bandwidth.

The zero frequency response can be obtained by setting $s = 0$ and knowing that $v_1 = v_2$ at steady state, we obtain:

$$\tilde{H}_x(0) = \frac{\varepsilon_I^{v_1}}{\varepsilon_x^{v_2}} = \frac{dx}{dI} \frac{I}{x}$$

This tells us the relative sensitivity of the steady state to changes in inducer. If the degradation reaction, v_2 is first-order, the elasticity, $\varepsilon_x^{v_2} = 1$, then the sensitivity is equal to:

$$\tilde{H}_x(0) = \varepsilon_I^{v_1}$$

That is the sensitivity of x is equal to the response of gene expression to inducer. By way of example, if the effect of inducer on v_1 is governed by a Hill equation with Hill coefficient, n , we know that the elasticity is given by:

$$\varepsilon_I^{v_1} = n(1 - Y)$$

where Y is the degree of saturation by inducer. If the Hill coefficient, $n = 4$ and the promoter is half saturated, the elasticity will equal:

$$\varepsilon_I^{v_1} = 2$$

This means that a 1% increase in inducer will result in a 2% increase in x .

19

Structural Constraints

Life is really simple, but we insist on making it complicated.

– Confucius

19.1 Introduction

In Chapter 17 a brief mention was made of the canonical transfer functions. These were defined as:

$$\mathcal{C}_x = \left(s\mathbf{I} - \mathbf{N} \frac{\partial \mathbf{v}}{\partial \mathbf{x}} \right)^{-1} \mathbf{N}$$

$$\mathcal{C}_J = \frac{\partial \mathbf{v}}{\partial \mathbf{x}} \mathcal{C}_x + \mathbf{I}$$

In the biology community, these canonical transfer functions are also referred to as the unscaled **concentration and flux control coefficients** when s is set to zero. Canonical transfer functions are parameter independent in that the particular parameter used to perturb the system is not explicitly specified. All that is assumed is that a reaction is perturbed, but by what is not indicated. Because we can define transfer functions in terms of the topological structure of biochemical networks, there are a range of interesting constraints, called structural constraints that emerge from the canonical transfer functions.

19.2 Structural Constraints

Summation Constraints

Let the basis for the null space of \mathbf{N} be given by \mathbf{K} , that is:

$$\mathbf{N}\mathbf{K} = \mathbf{0}$$

Post multiplying the canonical equation, \mathcal{C}_x by \mathbf{K} gives:

$$\mathcal{C}_x(j\omega)\mathbf{K} = \left(j\omega\mathbf{I} - \mathbf{N}\frac{\partial\mathbf{v}}{\partial\mathbf{x}} \right)^{-1} \mathbf{N}\mathbf{K}$$

yielding:

$$\mathcal{C}_x(j\omega)\mathbf{K} = \mathbf{0} \quad (19.1)$$

Post multiplying the canonical equation, \mathcal{C}_J by \mathbf{K} gives:

$$\mathcal{C}_J(j\omega)\mathbf{K} = \frac{\partial\mathbf{v}}{\partial\mathbf{x}}\mathcal{C}_x\mathbf{K} + \mathbf{K}$$

so that:

$$\mathcal{C}_J(j\omega)\mathbf{K} = \mathbf{K} \quad (19.2)$$

Connectivity Constraints

Given the canonical concentration transfer function:

$$\mathcal{C}_x(j\omega) = \left(j\omega\mathbf{I} - \mathbf{N}\frac{\partial\mathbf{v}}{\partial\mathbf{x}} \right)^{-1} \mathbf{N}$$

we can write the inverse term in the above equation as:

$$\left(j\omega\mathbf{I} - \mathbf{N}\frac{\partial\mathbf{v}}{\partial\mathbf{x}} \right)^{-1} \left(j\omega\mathbf{I} - \mathbf{N}\frac{\partial\mathbf{v}}{\partial\mathbf{x}} \right) = \mathbf{I}$$

or

$$\left(j\omega\mathbf{I} - \mathbf{N}\frac{\partial\mathbf{v}}{\partial\mathbf{x}} \right)^{-1} j\omega\mathbf{I} - \left(j\omega\mathbf{I} - \mathbf{N}\frac{\partial\mathbf{v}}{\partial\mathbf{x}} \right)^{-1} \mathbf{N}\frac{\partial\mathbf{v}}{\partial\mathbf{x}} = \mathbf{I}$$

Rearranging:

$$\left(j\omega \mathbf{I} - \mathbf{N} \frac{\partial \mathbf{v}}{\partial \mathbf{x}} \right)^{-1} \mathbf{N} \frac{\partial \mathbf{v}}{\partial \mathbf{x}} = \left(j\omega \mathbf{I} - \mathbf{N} \frac{\partial \mathbf{v}}{\partial \mathbf{x}} \right)^{-1} j\omega \mathbf{I} - \mathbf{I}$$

This can we rewritten as

$$\mathcal{C}_x(j\omega) \frac{\partial \mathbf{v}}{\partial \mathbf{x}} = \left(j\omega \mathbf{I} - \mathbf{N} \frac{\partial \mathbf{v}}{\partial \mathbf{x}} \right)^{-1} j\omega \mathbf{I} - \mathbf{I}$$

so that at $\omega = 0$ we obtain:

$$\mathcal{C}_x(0) \frac{\partial \mathbf{v}}{\partial \mathbf{x}} = -\mathbf{I} \quad (19.3)$$

Similarly we can derive a connectivity theorem for the fluxes. If:

$$\mathcal{C}_J(j\omega) = \frac{\partial \mathbf{v}}{\partial \mathbf{x}} \mathcal{C}_x + \mathbf{I}$$

then we can post multiply by $\frac{\partial \mathbf{v}}{\partial \mathbf{x}}$:

$$\mathcal{C}_J(j\omega) \frac{\partial \mathbf{v}}{\partial \mathbf{x}} = \frac{\partial \mathbf{v}}{\partial \mathbf{x}} \mathcal{C}_x \frac{\partial \mathbf{v}}{\partial \mathbf{x}} + \frac{\partial \mathbf{v}}{\partial \mathbf{x}}$$

But at $\omega = 0$, $\mathcal{C}_x(0) \frac{\partial \mathbf{v}}{\partial \mathbf{x}} = -\mathbf{I}$

Therefore:

$$\mathcal{C}_J(0) \frac{\partial \mathbf{v}}{\partial \mathbf{x}} = 0 \quad (19.4)$$

Scaled Constraints

In Chapter 18 the scaled canonical transfer function with respect to the species was defined by:

$$\mathbf{C}_x(0) = -(\mathbf{N} \operatorname{diag}(\mathbf{v}) \varepsilon_x)^{-1} \mathbf{N} \operatorname{diag}(\mathbf{v})$$

For clarity we will drop the (0) notation. Let us multiply both sides of above equation by the one vector, $[1, 1, \dots]^T$, so that:

$$\mathbf{C}_x \mathbf{1} = -(\mathbf{N} \operatorname{diag}(\mathbf{v}) \varepsilon_x)^{-1} \mathbf{N} \operatorname{diag}(\mathbf{v}) \mathbf{1}$$

$$\mathbf{C}_x \mathbf{1} = -(\mathbf{N} \operatorname{diag}(\mathbf{v}) \varepsilon_x)^{-1} \mathbf{N} \mathbf{v}$$

But at steady state, $\mathbf{N} \mathbf{v} = 0$, therefore:

$$\mathbf{C}_x \mathbf{1} = 0 \quad (19.5)$$

Similarly for the flux constraints:

$$\mathbf{C}_J = \varepsilon_x \mathbf{C}_x + \mathbf{I}$$

Inserting \mathbf{C}_s into the above equation yields:

$$\mathbf{C}_J = -\varepsilon_x (\mathbf{N} \operatorname{diag}(\mathbf{v}) \varepsilon_x)^{-1} \mathbf{N} \operatorname{diag}(\mathbf{v}) + \mathbf{I}$$

Multiplying both sides by $\mathbf{1} = [1, 1, \dots]^T$ yields:

$$\mathbf{C}_J \mathbf{1} = -\varepsilon_x (\mathbf{N} \operatorname{diag}(\mathbf{v}) \varepsilon_x)^{-1} \mathbf{N} \operatorname{diag}(\mathbf{v}) \mathbf{1} + \mathbf{1}$$

using the same arguments for \mathbf{C}_x reduces the equation to:

$$\mathbf{C}_J \mathbf{1} = \mathbf{1} \quad (19.6)$$

In scalar form these relations are given by:

$$\sum C_i^x = 0$$

$$\sum C_i^J = 1$$

where the summation is over all reaction steps in the pathway. These are called the flux and concentration summation theorems. Operationally they have an interesting interpretation. The way to understand the canonical transfer functions is to think of C_i^x as:

$$C_i^x = \frac{dx}{dv_i} \frac{v_i}{x} \simeq \frac{\delta x}{\delta v_i} \frac{v_i}{x}$$

where v_i is called the local reaction rate of the i^{th} reaction. A local reaction rate has the following operational interpretation. For the i^{th} reaction, let us clamp (fix) all state variable molecular species that might affect the reaction rate. We now find a suitable parameter, for example the enzyme concentration, and increase this by some small factor. This will cause the reaction rate to increase by some value δv_i . We now release the clamps and let the system evolve. With a higher reaction rate, substrate

will be consumed at a higher rate and product produced at a higher rate. Once the system reaches its new steady state, the reaction velocity v_i has settles to a new rate with changes in concentration x by δx . To compute the canonical transfer function, we take the ratio of the δx to δv_i . We stress again that the δv_i is the change in reaction rate we impose before the system is allowed to evolve to the new steady state. We do not specify how the v_i rate change is brought about.

If we assume that local reaction rates are proportional to enzyme concentration, that is:

$$v_i = k_i E_i$$

Then:

$$\frac{dx}{d(k_i E_i)} \frac{k_i E_i}{x} = \frac{dx}{dE_i} \frac{E_i}{x}$$

We can therefore write the concentration theorem as follows:

$$\sum \frac{dx}{dE_i} \frac{E_i}{x} = \frac{dx/x}{dE_i/E_i} = 0$$

We can interpret this as follows. Increasing the concentration of every enzyme in the pathway by a given relative amount, dE_i/E_i results in no change to the species concentrations. Likewise the flux theorem can be interpreted to mean that increasing the concentration of every enzyme in the pathway by a given relative amount, results in the flux changes by the same relative amount.

Scaled Connectivity Constraints

20

Linear Pathways

“Begin at the beginning,” the King said, very gravely, “and go on till you come to the end: then stop”.

– Lewis Carroll, *Alice in Wonderland*

20.1 Introduction

Linear or straight chain pathways represent the simplest network and are a good starting point to begin to understand the behavior of cellular networks. A typical linear pathway is depicted as follows:



This pathway has m floating species and n reactions ($n = m + 1$). X_o and X_1 are fixed species representing the source and sink pools respectively. Let us focus first on a simple two step pathway:



$$\frac{dx}{dt} = v_1(E_1, x) - v_2(E_2, x)$$

Here we assume that the first step, v_1 is both a function of the enzyme level E_1 and the intermediate x and the second step is a function of E_2 and x . We assume that X_1 has zero concentration, perhaps because v_2 discharges into a large volume so that the concentration of X_1 hardly changes. We assume X_o is fixed and therefore isn't considered an input in this example. The two inputs are E_1 and E_2

but we will generalize this as inputs to v_1 and v_2 . We start with the generalized network transfer equations for but the species and fluxes:

$$\mathbf{H}_x(s) = \left(s\mathbf{I} - \mathbf{N} \frac{\partial \mathbf{v}}{\partial \mathbf{x}} \right)^{-1} \mathbf{N} \frac{\partial \mathbf{v}}{\partial \mathbf{p}}$$

and

$$\mathbf{H}_J(s) = \frac{\partial \mathbf{v}}{\partial \mathbf{x}} \mathbf{H}_x(s) + \frac{\partial \mathbf{v}}{\partial \mathbf{p}}$$

Because we are not being specific in stating out outputs other than stating that inputs will be applied at v_1 and v_1 , we can drop the $\partial v / \partial p$ terms and compute the canonical transfer functions (See also example 17.2). This gives us two transfer functions for x and four transfer functions for the two fluxes, J_1 and J_2 .

$$\begin{aligned} \mathcal{C}_x(s) &= \begin{bmatrix} \frac{1}{s-\varepsilon_x^1+\varepsilon_x^2} & -\frac{1}{s-\varepsilon_x^1+\varepsilon_x^2} \end{bmatrix} \\ \mathcal{C}_J(s) &= \begin{bmatrix} \frac{s+\varepsilon_1^2}{s-\varepsilon_x^1+\varepsilon_x^2} & -\frac{\varepsilon_1^1}{s-\varepsilon_x^1+\varepsilon_x^2} \\ \frac{\varepsilon_1^2}{s-\varepsilon_x^1+\varepsilon_x^2} & \frac{s-\varepsilon_1^1}{s-\varepsilon_x^1+\varepsilon_x^2} \end{bmatrix} \end{aligned}$$

where ε_j^i are the unscaled elasticities, $\partial v_i / \partial x_j$. The scaled canonical transfer functions are given by:

$$\begin{aligned} C_x(s) &= \begin{bmatrix} \frac{1}{s \frac{x}{v_1} - \varepsilon_x^1 + \varepsilon_x^2} & -\frac{1}{s \frac{x}{v_2} - \varepsilon_x^1 + \varepsilon_x^2} \end{bmatrix} \\ C_J(s) &= \begin{bmatrix} \frac{s \frac{x}{v} + \varepsilon_1^2}{s \frac{x}{v} - \varepsilon_x^1 + \varepsilon_x^2} & -\frac{\varepsilon_1^1}{s \frac{x}{v} - \varepsilon_x^1 + \varepsilon_x^2} \\ \frac{\varepsilon_1^2}{s \frac{x}{v} - \varepsilon_x^1 + \varepsilon_x^2} & \frac{s \frac{x}{v} - \varepsilon_1^1}{s \frac{x}{v} - \varepsilon_x^1 + \varepsilon_x^2} \end{bmatrix} \end{aligned}$$

where ε_j^i are the scaled elasticities, $(\partial v_i / \partial x_j) x_j / v_i$. One way to interpret these equations is to look at the transfer functions zero frequency where they represent the sensitivities with respect to changes in the inputs v_1 and v_2 .

$$\begin{aligned} C_x(0) &= \frac{dx}{dv_i} \frac{v_i}{x} = \begin{bmatrix} \frac{1}{-\varepsilon_x^1 + \varepsilon_x^2} & -\frac{1}{-\varepsilon_x^1 + \varepsilon_x^2} \end{bmatrix} \\ C_J(0) &= \frac{dJ_i}{dv_j} \frac{v_j}{x} = \begin{bmatrix} \frac{\varepsilon_1^2}{-\varepsilon_x^1 + \varepsilon_x^2} & -\frac{\varepsilon_1^1}{-\varepsilon_x^1 + \varepsilon_x^2} \\ \frac{\varepsilon_1^2}{-\varepsilon_x^1 + \varepsilon_x^2} & \frac{-\varepsilon_1^1}{-\varepsilon_x^1 + \varepsilon_x^2} \end{bmatrix} \end{aligned}$$

Note that for a given column in $C_J(0)$, the row entries are the same, this is because at steady state, the fluxes J_1 and J_2 are equal to each other.

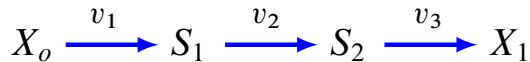
20.2 Front Loading

In a linear pathway with linear reversible kinetics on each step, given two adjacent flux control coefficients, the upstream coefficient will always be equal or larger than the downstream coefficient, that is for the i^{th} step the following is true:

$$C_i^J \geq C_{i+1}^J$$

This means that in a linear pathway control will be concentrated upstream. To understand why this should be the case we must consider the elasticities and control equations for a linear pathway.

Using the flux summation and connectivity theorems it is straight forward to derive the flux control equations. For example for the three step pathway:



one can derive the following flux control coefficient equations:

$$C_{E_1}^J = \varepsilon_1^2 \varepsilon_2^3 / D$$

$$C_{E_2}^J = -\varepsilon_1^1 \varepsilon_2^3 / D$$

$$C_{E_3}^J = \varepsilon_1^1 \varepsilon_2^2 / D$$

where D the denominator is given by:

$$D = \varepsilon_1^2 \varepsilon_2^3 - \varepsilon_1^1 \varepsilon_2^3 + \varepsilon_1^1 \varepsilon_2^2$$

It is possible to do this for pathways with additional steps from which a clear pattern emerges in the equations. For a pathway with n steps where n is even, we have the following equations:

$$C_1^J = \varepsilon_1^2 \varepsilon_2^3 \varepsilon_3^4 \varepsilon_4^5 \cdots \varepsilon_n^{n+1} / D$$

⋮

$$C_m^J = \prod_{k=m}^n \varepsilon_k^{k+1} \prod_{k=m-1}^1 \varepsilon_k^k / D$$

⋮

$$C_n^J = \varepsilon_1^1 \varepsilon_2^2 \varepsilon_3^3 \varepsilon_4^4 \cdots \varepsilon_{n+1}^{n+1} / D$$

If we look carefully at C_1^J we see that the numerator is the product of all the substrate elasticities. This implies that a perturbation in E_1 ‘hops’ from one enzyme to the next until it reaches the end of the pathway. Conversely, the control coefficient of the last enzyme includes all the product elasticities, that is the perturbation ‘hops’ from one enzyme to the next until it reaches the beginning of the pathway.

If we looked at any intermediate enzyme step we would find two groups of elasticities, one group representing the perturbation traveling downstream via the substrate elasticities and the other representing the perturbation traveling upstream via product elasticities.

We must now recall that given a reversible mass-action rate law, such as $k_1S - k_2P$, the elasticities are given by:

$$\varepsilon_S^v = \frac{1}{1-\rho}$$

$$\varepsilon_P^v = -\frac{\rho}{1-\rho}$$

From these equations it follows that $\varepsilon_S^v + \varepsilon_P^v = 1$, that is:

$$\| \varepsilon_S^v \| \geq \| \varepsilon_P^v \|$$

That is the absolute value of the substrate elasticity is always greater than the product elasticity. Given that an upstream enzyme will have more substrate elasticities than product elasticities, it follows that the numerator will be larger when compared to an enzyme further downstream which will have more of the small value product elasticities. What this means is that perturbations at a downstream enzyme will be attenuated compared to a similar perturbation at an upstream step. Hence the control coefficients upstream will on average be larger.

The origins of the asymmetry between the substrate and product elasticities is a thermodynamic one. If the thermodynamic gradient were to be reversed so that the pathway flux traveled ‘upstream’, the elasticity values exchange so that now the front loading occurs downstream, although ‘downstream’ is now ‘upstream’ because the flux has reversed.

In a linear pathway governed by linear kinetics and without regulation, flux control is biased towards the start of the pathway, an effect called **front loading**.

21

Negative Feedback

“The living being is stable. It must be so in order not to be destroyed, dissolved or disintegrated by the colossal forces, often adverse, which surround it.”

– Charles Richet, 1900.

21.1 Introduction

Biological networks are riddled with regulatory interactions. These include positive and negative, forward and feedback. Chapter 3 described some of the basic properties of negative feedback. Properties such as disturbance rejection and improvements to performances. In this chapter we will consider more closely the properties of systems with negative feedback regulation. Negative feedback can occur in many different ways in biochemical networks. Some is quite overt, for example an allosteric regulator. Others can be more subtle, for example by simple sequestration in a signaling network. Some of the more important problems in the analysis of negative feedback include:

- The Transient Response
- Stability
- Accuracy
- Sensitivity
- Linearization

If we were to summarize in one sentence the role of negative feedback, it would be:

Feedback is used to combat the effects of uncertainty.

21.2 Types of Feedback

In Chapter 3 we considered a type of negative feedback called **proportional control** as shown in Figure 21.1. The error, $e(t)$ that arises from comparing the set point and the feedback is feed into a controller, K_p . We assume that the output from the controller is proportional to the error, eK_p . That is if the error doubles, the output of the controller will also double. This type of feedback control is called proportional control. We can express this type of control with the equation:

$$u = K_p e(t)$$

There are however other ways to respond to the error signal. Two of the most common are called **integral control** and **derivative control**. In practice these are often combined to form a **PID controller**, where PID refers to proportional-integral-derivative. PID control is by far the most common form of control in man made devices and there is even evidence to suggest that similar control systems are found in biology [81].

Before we consider the different types of feedback let us look more closely at the properties of a feedback system that uses proportional control.

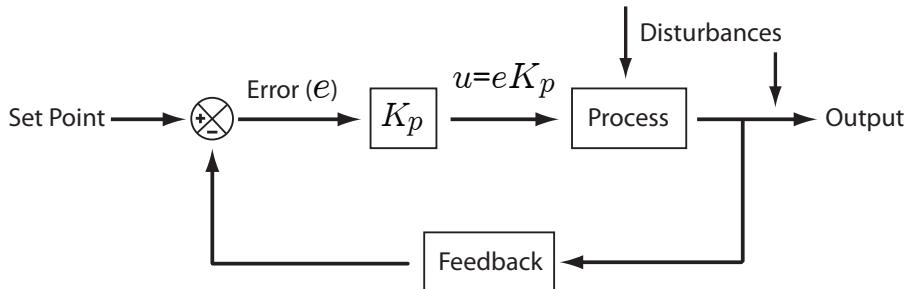


Figure 21.1 Negative feedback system using proportional control.

21.3 Proportional Control

Transient Response

Consider the simple genetic circuit in Figure 21.2: If we assume first-order kinetics on v_2 and an

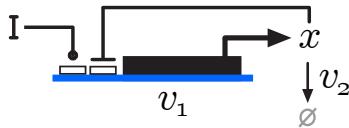


Figure 21.2 Simple Negative Feedback System.

unknown function that represents the effect of the input I and feedback x , we can write the differential equation for this system as:

$$\frac{dx}{dt} = v_1(I, x) - k_2 x$$

where the function v_1 indicates that the rate is a function of x as well as the input, I . We can linearize this equation using (17.2) to obtain:

$$\frac{d\delta x}{dt} = \left(\frac{\partial v_1}{\partial x} - \frac{\partial v_2}{\partial x} \right) \delta x + \frac{\partial v_1}{\partial I} \delta I$$

The solution to this linear differential equation is:

$$x(t) = \delta x_o e^{\alpha t} + \frac{\partial v_1}{\partial I} \frac{\delta I}{\alpha} (e^{\alpha t} - 1)$$

where:

$$\alpha = \frac{\partial v_1}{\partial x} - \frac{\partial v_2}{\partial x} < 0 \quad \text{and} \quad \delta x_o = \text{initial condition}$$

Note that α is net negative because $\partial v_1 / \partial x$, the feedback term, is always negative. The exponential terms represent the transient response and depend on the feedback term, $\partial v_1 / \partial x$. The larger the absolute value of $\partial v_1 / \partial x$ the faster the decay rate. The conclusion therefore is that negative feedback can increase the responsiveness of the system.

Negative feedback can increase the responsiveness of the system.

Improved Frequency Response

If negative feedback can increase the responsiveness of the system then we would also expect the frequency response to reflect this. Let us redo the previous analysis this time using transfer functions. Consider again the negative feedback system shown in Figure 21.2.

We can compute the transfer function from the equation (Chapter 17):

$$\mathbf{H}(s) = \left(s\mathbf{I} - \mathbf{N} \frac{\partial \mathbf{v}}{\partial \mathbf{x}} \right)^{-1} \mathbf{N} \frac{\partial \mathbf{v}}{\partial \mathbf{p}}$$

The stoichiometry matrix for this system is: $[1 - 1]$ and the unscaled elasticity matrix, $\partial \mathbf{v} / \partial \mathbf{x}$ is a two by one matrix:

$$\frac{\partial \mathbf{v}}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial v_1}{\partial x} \\ \frac{\partial v_2}{\partial x} \end{bmatrix}$$

If we assume that the input, I only affects v_1 then the $\frac{\partial \mathbf{v}}{\partial \mathbf{p}}$ matrix is given by:

$$\frac{\partial \mathbf{v}}{\partial \mathbf{p}} = \begin{bmatrix} \frac{\partial v_1}{\partial I} \\ 0 \end{bmatrix}$$

Entering these terms into the transfer function equation gives us:

$$H(s) = \frac{\frac{\partial v_1}{\partial I}}{s + \left(\frac{\partial v_2}{\partial x} - \frac{\partial v_1}{\partial x} \right)} \quad (21.1)$$

This equation has the same structure as Equation 13.3. Equation 13.4 is the amplitude frequency response. In this case the k_2 term in equation 13.4 is now the sum of two unscaled elasticities. Since $\partial v_1 / \partial x$ is negative the sum is overall positive.

$$\text{Amplitude} = |H(j\omega)| = \frac{\frac{\partial v_1}{\partial I}}{\sqrt{\left(\frac{\partial v_2}{\partial x} - \frac{\partial v_1}{\partial x} \right)^2 + \omega^2}} \quad (21.2)$$

This equation is plotted as a Bode plot in Figure 21.3. Two curves are shown, the continuous line represents the system with weak negative feedback and the dotted line is the same system with strong negative feedback. The input signal has been adjusted in each case so that both systems have the same gain at zero frequency. If this weren't done, the strong negative feedback line would be lower due to attenuation and it would be more difficult to see the extension in the bandwidth. The result however is that with negative feedback, the frequency response, or bandwidth, **improves**. This is consistent with the previous result which showed an increase in the transient response.

Negative feedback improves the frequency response of the system (but at the expense of gain).

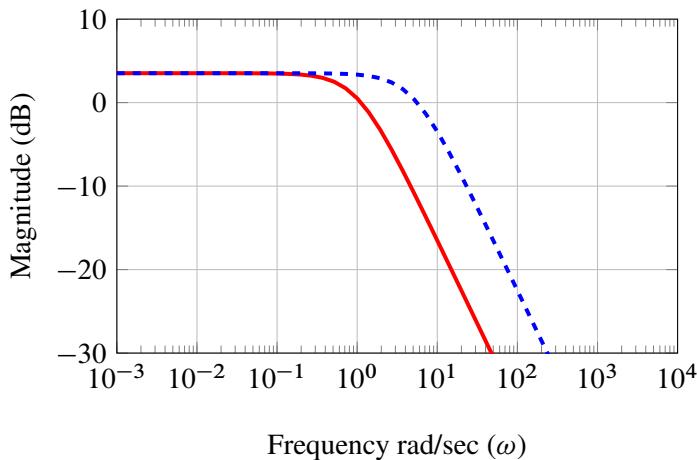


Figure 21.3 Bode Plot: Magnitude (dB). Dotted line is generated from a system with stronger negative feedback. Note that the bandwidth with stronger feedback is higher. The gain has been increased to compensate for the loss of gain due to the negative feedback, hence both plots start from the same y-axis intercept.

21.4 Reduced Gain

Equation 21.2 also shows us that the gain of the system is reduced in the presence of negative feedback. This can be more easily seen if we look at the gain at zero frequency, $H(0)$ which we know from the final-value theorem (Equation 13.7) to be equal to the unscaled sensitivity of the state variable with respect to a step input, p :

$$H(0) = \frac{dx}{dp}$$

Setting ω to zero in equation 21.2 we obtain:

$$H(0) = \frac{dx}{dI} = \frac{\frac{\partial v_1}{\partial I}}{\frac{\partial v_2}{\partial x} - \frac{\partial v_1}{\partial x}}$$

The term in the denominator is positive so that increases in the feedback strength, $\partial v_1 / \partial x$ will reduce the gain. This is expected, because as the feedback strengthens the ability of the system to resist changes in the input I , improves.

Feedback reduces the gain of the system.

21.5 Accuracy

The accuracy of a feedback system is the difference between the set point and output of the system at steady state. The closer the output is to the set point the more accurate the system is. Figure 21.4 shows a block diagram of a generic feedback system with components expressed as transforms.

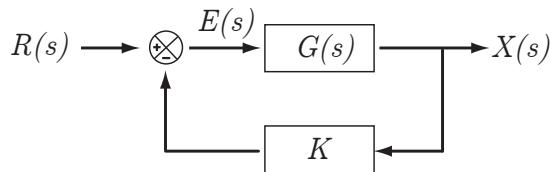


Figure 21.4 Negative feedback block diagram

The closed loop transfer function is given by:

$$\frac{X(s)}{R(s)} = \frac{G(s)}{1 + KG(s)} \quad (21.3)$$

This can be rewritten as:

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

Since the error term, $E(s)$ is given by:

$$E(s) = R(s) - KX(s)$$

Combining the two equations gives us:

$$E(s) = R(s) \left[1 - \frac{KG(s)}{1 + KG(s)} \right] = R(s) \frac{1}{1 + KG(s)} \quad (21.4)$$

Assuming the system is stable, the degree of error in the non-Laplace domain can be obtained using the final-value theorem:

$$e_{ss} = \lim_{t \rightarrow \infty} e(t) = \lim_{s \rightarrow 0} sE(s) = \lim_{s \rightarrow 0} \frac{sR(s)}{1 + KG(s)}$$

In order to evaluate e_{ss} we must decide on an input function, $R(s)$. The simplest is a step function so that $R(s) = 1/s$. Inserting this into the equation gives:

$$e_{ss} = \lim_{s \rightarrow 0} \frac{s/s}{1 + KG(s)} = \frac{1}{1 + KG(0)} \quad (21.5)$$

The term, $KG(0)$ is the *loop gain* (See Figure 3.5) and is a constant which we designate, K_p , so that:

$$e_{ss} = \frac{1}{1 + K_p} \quad (21.6)$$

K_p is often called the **position error constant**. This result shows that the error can be made smaller by increasing the loop gain although the error can only be eliminated if the gain is infinite. The higher the gain the smaller the error. However as we will see later in the chapter, the stronger the feedback the more likely the system will become unstable if $G(s)$ has an order of two or more.. There is therefore a trade off between accuracy and stability when designing feedback systems. Unless the loop gain is very large, there will always be an offset between the set point and the outputs.

Increasing the loop gain improves the accuracy of the feedback response.

Example 21.1

The loop gain, $KG(s)$ for a system is given by:

$$KG(s) = \frac{90}{s + 10}$$

what is the steady state error when a unit step input is applied to the system? Using equation (21.5) we can write (Using $R(s) = 1/s$):

$$\begin{aligned} e_{ss} &= \lim_{s \rightarrow 0} s \frac{R(s)}{1 + 90/(s + 10)} \\ &= \lim_{s \rightarrow 0} s \frac{R(s)(s + 10)}{s + 10 + 90} \\ &= \lim_{s \rightarrow 0} s \frac{1(s + 10)}{s(s + 100)} \\ &= \lim_{s \rightarrow 0} \frac{s + 10}{s + 100} \\ e_{ss} &= 0.1 \end{aligned}$$

In other words, the steady state-state error will be 10 percent of the set point (this is so because the change in the step point was a unit step input).

Response to Different Inputs

It is a common practice to classify the response of a negative feedback system according to its ability to follow step, ramp and parabolic inputs. To investigate these responses it is also common to consider the unit feedback as shown in Figure 21.5

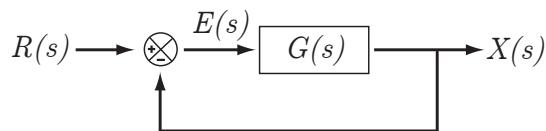


Figure 21.5 Unity Negative feedback block diagram

To analyses this system we will introduce a type for the system which will be based on the number of zero poles (also equivalent to the number of integrators). The addition of the integrator terms, s^n further generalizes the transfer function.

We will define the **type** of a system as the number of poles at $s = 0$

$$\text{Type 0: } G(s) = \frac{K(s + z_1) \dots}{(s + p_1) \dots}$$

$$\text{Type 1: } G(s) = \frac{K(s + z_1) \dots}{s(s + p_1) \dots}$$

$$\text{Type 2: } G(s) = \frac{K(s + z_1) \dots}{s^2(s + p_1) \dots}$$

...

Example 21.2

Determine the type for the following open-loop transfer function (Assuming unity feedback):

$$G(s) = \frac{40s(1 + 6s)}{(20s^2 + 5s)(s^2 + 2s + 7)s^2}$$

Grouping the pure s terms yields:

$$G(s) = \frac{40s(1 + 6s)}{s^3(20s + 5)(s^2 + 2s + 7)}$$

Cancelling s terms in the numerator and denominator gives:

$$G(s) = \frac{40(1 + 6s)}{s^2(20s + 5)(s^2 + 2s + 7)}$$

The transfer function is therefore of type 2.

Type 0 Steady State Errors

A type 0 system is defined using the open loop transfer function:

$$G(s) = \frac{K(s + z_1) \dots}{(s + p_1) \dots}$$

Step Input: Applying a step function, $R(s) = A/s$ and using equation (21.5) we obtain:

$$e_{ss} = \frac{A}{1 + K_p}$$

which we've seen before in equation (21.6).

Ramp: For a ramp input: $R(s) = A/s^2$ and using equation (21.5) we obtain:

$$e_{ss} = \frac{A}{0}$$

that is the steady state error becomes infinity. A type 0 system cannot track a ramp.

Parabolic: For a parabolic input: $R(s) = A/s^3$ and using equation (21.5) we obtain:

$$e_{ss} = \frac{A}{0}$$

Again we see that a type 0 system cannot track a parabolic input.

Type 1 Steady State Errors

A type 1 system is defined using the open loop transfer function:

$$G(s) = \frac{K(s + z_1) \dots}{s(s + p_1) \dots}$$

Note the single integrator term in the denominator.

Step Input: For a step input, the error is given by:

$$e_{ss} = \frac{A}{\infty}$$

That is, other than transient terms, a type 1 system can track a step input exactly. This observation becomes crucial when we discuss integral control later in the chapter.

Ramp: For a ramp input, the error is given by:

$$e_{ss} = \frac{A}{K_v}$$

Type (n)	Step (A/s)	Ramp (A/s^2)	Parabolic (A/s^3)
0	$e_{ss} = \frac{A}{1 + K_p}$	Infinite	Infinite
1	$e_{ss} = 0$	$\frac{A}{K_v}$	Infinite
2	$e_{ss} = 0$	0	$\frac{A}{K_a}$

Table 21.1 Summary of behaviors for different systems types.

That is a type 1 system has a steady-state error of K_v , also known as the **velocity error constant**. As the ramp increases the error is a constant displacement from the steady-state.

Parabolic: For a parabolic input the error is given by:

$$e_{ss} = \frac{A}{0}$$

that is a type 1 system cannot follow a parabolic input. Type 2 responses are left as an exercise for the reader except to say that the constant associated with the type 2 response is called the **acceleration error constant**, K_a . Table 21.1 summarises these results.

21.6 Linearization

We will consider only the steady-state behavior of the system. We take the input r , the output y , and the error e to be constant scalars. Assume (for now), that both the amplifier A and the feedback K act by multiplication (take A and K as non-negative scalars). Then without feedback (i.e. $K = 0$), the system behavior is described by $y = Ar$, which is an amplifier (with gain A) provided that $A > 1$. Assuming for now that the disturbance d is zero, and if we now include feedback in our analysis, the behavior of the system is as follows. From the diagram, we have

$$y = Ae \quad e = r - Ky.$$

Eliminating e , we find

$$y = \frac{Ar}{1 + AK} \quad \text{or simply } y = Gr,$$

where $G = \frac{A}{1 + AK}$ is the *system (or closed loop) gain*. Comparing G with A , it is immediate that the feedback does indeed reduce the gain of the amplifier. Further, if the *loop gain* AK is large ($AK \gg 1$), then

$$G \approx \frac{A}{AK} = \frac{1}{K}.$$

That is, as the gain AK increases, the system behavior becomes more dependent on the feedback loop and less dependent on the rest of the system. We next indicate three specific consequences of this key insight.

Improved fidelity of response Consider now the case where the process block A is nonlinear. We can write down the following relationships:

$$A(e) = y \quad e = r - Ky = r - KA(e).$$

where k is the feedback term, e the error term, r the set point, and y the output. Combining both equations to eliminate e , yields:

$$y = A(r - Ky)$$

Taking the inverse function of A on both sides:

$$A(y)^{-1} = r - Ky$$

Therefore:

$$y = \frac{r}{K} - \frac{A(y)^{-1}}{K}$$

So long as K is larger than $A(y)^{-1}$, then we can write:

$$y = \frac{r}{K}$$

In other words y is a linear function of r . The feedback compensates for the nonlinearities $A(e)$ and the system response is not distorted.

A natural objection to the implementation of feedback as described above is that the system sensitivity is not actually reduced, but rather is shifted so that the response is more sensitive to the feedback K and less sensitive to the amplifier A . However, in each of the cases described above, we see that it is the nature of the loop gain AK (and not just the feedback K) which determines the extent to which the feedback affects the nature of the system. This suggests an obvious strategy. By designing a system which has a small “clean” feedback gain and a large “sloppy” amplifier, one ensures that the loop gain is large and the behavior of the system is satisfactory. Engineers employ precisely this strategy in the design of electrical feedback amplifiers, regularly making use of amplifiers with gains several orders of magnitude larger than the feedback gain (and the gain of the resulting system).

21.7 Sensitivity

Resistance to internal parameter variation In all real amplifiers, both man-made and natural, there will be variation in the amplifier (A) characteristics, either as a result of the manufacturing

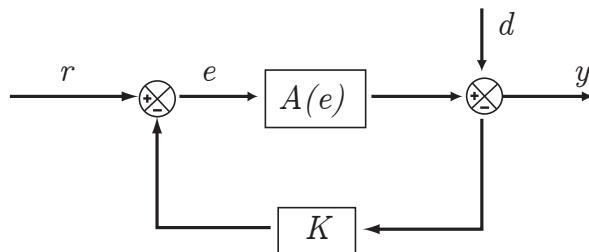


Figure 21.6 Linear Feedback System

process or internally generated thermal noise. We can study the effect of variation in the amplifier characteristics by investigating how A causes variation in the gain G .

Considering the sensitivity of the system gain G to variation in the parameter A , we find

$$\frac{\partial G}{\partial A} = \frac{\partial}{\partial A} \frac{A}{1 + AF} = \frac{1}{(1 + AF)^2}.$$

Clearly, this sensitivity decreases as AF increases. It may be more telling to consider the relative sensitivity, in which case we find

$$\frac{\partial G}{\partial A} \frac{A}{G} = \frac{1}{1 + AF},$$

so that for a small change ΔA in the gain of the amplifier, we find the resulting change ΔG in the system gain satisfies

$$\frac{\Delta G}{G} \approx \frac{1}{1 + AF} \frac{\Delta A}{A}.$$

As the strength of the feedback (K) increases the influence of variation in A decreases.

Resistance to disturbances in the output Suppose now that a nonzero disturbance d affects the output as in Figure 21.6. The system behavior is then described by

$$y = Ae - d \quad e = r - Ky.$$

Eliminating e , we find

$$y = \frac{Ar - d}{1 + AK}.$$

The sensitivity of the output to the disturbance is then

$$\frac{\partial y}{\partial d} = -\frac{1}{1 + AK}.$$

Again, we see that the sensitivity decreases as the loop gain AK is increased. In practical terms, this means that the imposition of a load on the output, for example a current drain in an electronic circuit or protein sequestration on a signaling network, will have less of an effect on the amplifier as the feedback strength increases. In electronics this property essentially modularizes the network into functional modules.

21.8 Sensitivity at Zero Frequency

Figure 21.7 shows the simplest possible negative feedback system, two reactions where the first reaction is inhibited by x . Let's assume that there exist two drugs, p_1 and p_2 which act on the reaction rates, v_1 and v_2 respectively. The drugs can be used to reduce the flux through the pathway and assuming only one of the drugs can be used at any one time, which drug should be used, p_1 that targets v_1 or p_2 that targets v_2 ?

This question can be answered by looking at the flux transfer function at zero frequency in response to a step input.

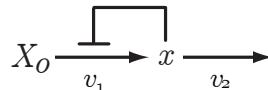


Figure 21.7 Two steps with negative feedback.

The flux transfer function is given by equation 17.5 and is repeated here for convenience:

$$\mathbf{H}_J(s) = \frac{\partial \mathbf{v}}{\partial \mathbf{x}} \left(s\mathbf{I} - \mathbf{N} \frac{\partial \mathbf{v}}{\partial \mathbf{x}} \right)^{-1} \mathbf{N} \frac{\partial \mathbf{v}}{\partial \mathbf{p}} + \frac{\partial \mathbf{v}}{\partial \mathbf{p}} \quad (21.7)$$

Given Figure 21.7 we can construct the required matrices:

$$\mathbf{N} = \begin{bmatrix} 1 & -1 \end{bmatrix}, \quad \frac{\partial \mathbf{v}}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial v_1}{\partial x} \\ \frac{\partial v_2}{\partial x} \end{bmatrix}, \quad \frac{\partial \mathbf{v}}{\partial \mathbf{p}} = \begin{bmatrix} \frac{\partial v_1}{\partial p_1} & 0 \\ 0 & \frac{\partial v_2}{\partial p_2} \end{bmatrix}$$

Note that because there are two possible inputs, p_1 and p_2 at two possible sites, v_1 and v_2 , the $\partial \mathbf{v} / \partial \mathbf{p}$

must be a 2 by 2 matrix. If we enter these matrices into equation 21.7 we obtain:

$$H_J(0) = \begin{bmatrix} \frac{dJ_1}{dp_1} & \frac{dJ_1}{dp_2} \\ \frac{dJ_2}{dp_1} & \frac{dJ_2}{dp_2} \end{bmatrix} = \begin{bmatrix} \frac{\partial v_2}{\partial x} \frac{\partial v_1}{\partial p_1} & \frac{\partial v_1}{\partial x} \frac{\partial v_2}{\partial p_2} \\ \frac{\partial v_2}{\partial x} - \frac{\partial v_1}{\partial x} & \frac{\partial v_2}{\partial x} - \frac{\partial v_1}{\partial x} \\ \frac{\partial v_2}{\partial x} \frac{\partial v_1}{\partial p_1} & \frac{\partial v_1}{\partial x} \frac{\partial v_2}{\partial p_2} \\ \frac{\partial v_2}{\partial x} - \frac{\partial v_1}{\partial x} & \frac{\partial v_2}{\partial x} - \frac{\partial v_1}{\partial x} \end{bmatrix}$$

We can make one final transformation which is to convert the unscaled terms into scaled ones. Because v_1 and v_2 are equal at steady state, the $H_J(0)$ matrix contains two identical columns. We can scale the elasticities by multiplying the top and bottom by x , p_1 , p_2 and v_1 as appropriate (on the understanding that $v_1 = J_1 = v_2 = J_2$). We can pick out the top row to obtain the following transfer function vector at zero frequency:

$$[C_{p_1}^J \quad C_{p_2}^J] = \left[\frac{\varepsilon_x^2}{\varepsilon_x^2 - \varepsilon_x^1} \quad -\frac{\varepsilon_x^1}{\varepsilon_x^2 - \varepsilon_x^1} \right]$$

The scaled derivatives dJ/dp have been replaced with the symbol C_p^J , and the scaled derivatives $\partial v/\partial x$ with ε_x^v . To make the analysis easier we assume that the elasticities $\varepsilon_{p_1}^{v_1}$ and $\varepsilon_{p_2}^{v_2}$ both equal one. We can use these results to determine which is the best target for the drug. We first assume that the negative feedback is strong, this means that ε_x^1 is large and negative. Given this the two sensitivities reduce to:

$$[C_{p_1}^J \quad C_{p_2}^J] = [0 \quad 1]$$

This result indicates that the best target to change the flux is v_2 . Note that in the absence of feedback the opposite is true, that is the best target to change the flux is v_1 .

$$[C_{p_1}^J \quad C_{p_2}^J] = [1 \quad 0]$$

Three Step Pathway

A similar analysis can be done for a three step pathway (Figure 21.8). In this case we will ask which enzyme catalyzed reaction has the largest influence on the steady state flux. Given there are three reactions there are three enzymes, E_i , and therefore three inputs to the system. The resulting matrices

we need to construct the transfer functions are:

$$\mathbf{N} = \begin{bmatrix} 1 & -1 & 0 \\ 0 & 1 & -1 \end{bmatrix}, \quad \frac{\partial \mathbf{v}}{\partial \mathbf{x}} = \begin{bmatrix} 0 & \frac{\partial v_1}{\partial x_2} \\ \frac{\partial v_2}{\partial x_1} & 0 \\ 0 & \frac{\partial v_3}{\partial x_2} \end{bmatrix}, \quad \frac{\partial \mathbf{v}}{\partial \mathbf{p}} = \begin{bmatrix} \frac{\partial v_1}{\partial E_1} & 0 & 0 \\ 0 & \frac{\partial v_2}{\partial E_2} & 0 \\ 0 & 0 & \frac{\partial v_3}{\partial E_3} \end{bmatrix}$$

We can now form both transfer functions, $H_x(s)$ and $H_J(s)$ for this system where there are three inputs corresponding to the three enzymes. Scaling all terms yields the following transfer function terms at zero frequency.

$$C_{E_1}^J = \frac{\varepsilon_1^2 \varepsilon_2^3}{\varepsilon_1^2 \varepsilon_2^3 - \varepsilon_1^2 \varepsilon_2^1}$$

$$C_{E_2}^J = 0$$

$$C_{E_3}^J = \frac{-\varepsilon_1^2 \varepsilon_2^1}{\varepsilon_1^2 \varepsilon_2^3 - \varepsilon_1^2 \varepsilon_2^1}$$

$$C_{E_1}^{S_2} = \frac{\varepsilon_1^2}{\varepsilon_1^2 \varepsilon_2^3 - \varepsilon_1^2 \varepsilon_2^1}$$

$$C_{E_2}^{S_2} = 0$$

$$C_{E_3}^{S_2} = \frac{-\varepsilon_1^2}{\varepsilon_1^2 \varepsilon_2^3 - \varepsilon_1^2 \varepsilon_2^1}$$

We have assumed that the reactions catalyzed by E_1 and E_2 are product insensitive so that their elasticities are zero. From these equations we can make the following observations. Let us make the feedback strength, ε_2^1 strong, that is $\ll 0$. In this situation the transfer functions tend to the following limits:

$$C_{E_1}^J \rightarrow \frac{-\varepsilon_2^3}{\varepsilon_2^1}$$

$$C_{E_2}^J = 0$$

$$C_{E_3}^J \rightarrow 1$$

$$C_{E_1}^{S_2} \rightarrow \frac{-1}{\varepsilon_2^1}$$

$$C_{E_2}^{S_2} = 0$$

$$C_{E_3}^{S_2} \rightarrow \frac{1}{\varepsilon_2^1}$$

21.9 Implications for Drug Targeting

The analysis of feedback illustrates an important principle for those engaged in finding new drug targets. The aim of a drug is to cause a disruption to the network in such a way that it restores the

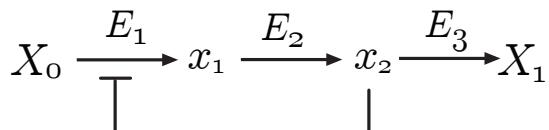


Figure 21.8 Three step pathway with negative feedback, E_i represent enzymes.

network to its ‘healthy’ wild-type state. Clearly targets must be susceptible to disruption for the drug to have any effect. The analysis of feedback suggests that targets inside the feedback loop are not suitable because any attempt to disturb these targets will be resisted by the feedback loop. Conversely, targets upstream and particularly downstream are very susceptible to disturbance. Figure 21.9 illustrates the effect of a 20 fold decrease in enzyme activity at two points in a simple reaction chain. In the first case both disruption at the center or end of the network has a significant effect on the concentration of the last species, S_3 . In the second panel, the same pathway is shown but with a negative feedback loop from S_3 to the first enzyme. The same activity reductions are also shown, but this time note the almost insignificant effect that modulating a step inside the loop has compared to modulating a step outside the loop.

Thus the take-home message to pharmaceutical companies who are looking for suitable targets is to avoid targeting reaction steps inside feedback loops!

21.10 Stability

Another key property of negative feedback systems is their propensity to become unstable. In terms of the frequency response it is straight forward to understand the origins of this instability. In a negative feedback system, most disturbances are damped due to the action of the feedback. However what if the feedback mechanism takes too long to respond so that by the time the negative feedback acts, the disturbance has already abated. In such a situation, the feedback would now attempt to restore a disturbance that is no longer present resulting in an incorrect action. Imagine that the feedback system acts in the opposite direction to the disturbance because of the delay, this would cause the disturbance to grow rather than fall. Imagine also that there is sufficient gain in the feedback loop to amplify or at least maintain this disturbance, the result would be a growing signal. If the loop gain amplifies then the disturbance will grow until it reaches the physical limits of the system at which point the loop gain is likely to fall and the disturbance fall. The result is a continuous growth and decline in the original disturbance, that is a sustained oscillation.

The key elements for sustaining an oscillation is a sufficient loop gain (at least 1.0) and a delay in the system of exactly -360° . As we have seen, specific phase shifts only occur at a particular frequency, however random disturbances in the system will occur at all frequencies, therefore disturbances in a system that has sufficient loop gain will quickly locate the point where the phase shift is at -180° .

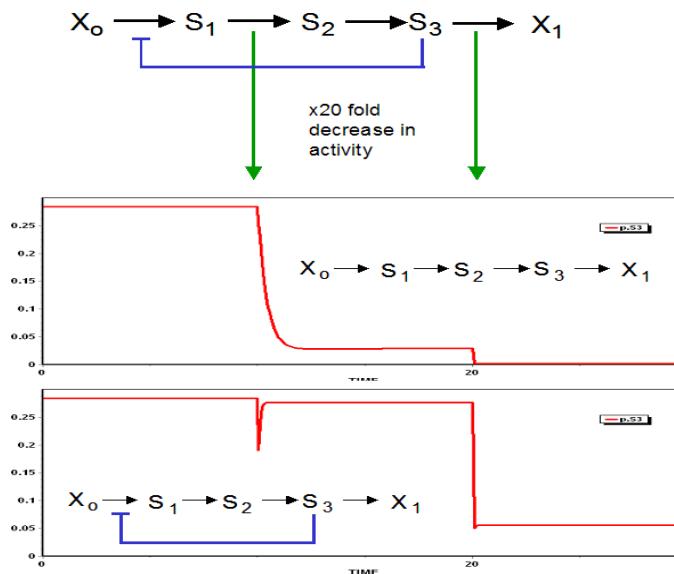


Figure 21.9 Negative Feedback and Drug Targets

resulting in a rapid destabilization of the system.

The requirement for a -180° shift plus the -180° brought about by the inverting feedback (giving -360° in total) means that at this point the negative feedback is effectively behaving as a positive feedback. That is an increase at the input means that the input is first inverted 180 degrees by the phase shift, then another 180 degree by the negative feedback, meaning that the input instead of being diminished now increases in value. Positive feedbacks are normally destabilizing influences.

Mention gain margin etc here

Stability from the Transfer Function

In this section we will look at the stability of negative feedback systems from the perspective of the transfer function, $H(s)$. Consider first a simple gene cascade with negative feedback. This system (Figure 21.10) has just two steps within the feedback loop. x_2 is the feedback signal that inhibits v_1 .

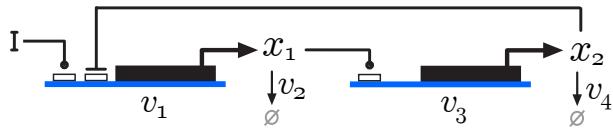


Figure 21.10 Negative Feedback in a Two Gene Cascade

We can compute the transfer function from the usual equation:

$$\mathbf{H}(s) = \left(s\mathbf{I} - \mathbf{N} \frac{\partial \mathbf{v}}{\partial \mathbf{x}} \right)^{-1} \mathbf{N} \frac{\partial \mathbf{v}}{\partial \mathbf{p}}$$

To determine the stability of the system we only need to look at the characteristic equation which can be computed from the determinant of:

$$\left(s\mathbf{I} - \mathbf{N} \frac{\partial \mathbf{v}}{\partial \mathbf{x}} \right)$$

We will assume that the degradation steps, \$v_2\$ and \$v_3\$, are first order. In addition because the pathway is not branched, the unscaled elasticities translate directly to the scaled elasticities. This means that the degradation scaled elasticities are one. We can write down \$\mathbf{N}\$ and the elasticity matrix as:

$$\mathbf{N} = \begin{bmatrix} 1 & -1 & 0 & 0 \\ 0 & 0 & 1 & -1 \end{bmatrix}$$

Note that entry \$(1, 3)\$ is zero because \$v_2\$ does not consume \$x_1\$. \$\partial \mathbf{v} / \partial \mathbf{x}\$ is a \$(4, 1)\$ matrix corresponding to 4 reaction rates and 2 species, \$x_1\$ and \$x_2\$. We will also use the symbol \$\mathcal{E}_j^i\$ to denote the unscaled elasticity for reaction \$i\$ with respect to species \$j\$.

In future examples \$\mathcal{E}_j^i\$ will be used to denote the unscaled elasticity for reaction \$i\$ with respect to species \$j\$, that is:

$$\mathcal{E}_j^i = \frac{\partial v_i}{\partial x_j}$$

For example $\varepsilon_{x_2}^{v_1}$ is the feedback elasticity.

$$\frac{\partial \mathbf{v}}{\partial \mathbf{x}} = \begin{bmatrix} 0 & \varepsilon_{x_2}^{v_1} \\ \varepsilon_{x_1}^{v_2} & 0 \\ \varepsilon_{x_1}^{v_3} & 0 \\ 0 & \varepsilon_{x_2}^{v_4} \end{bmatrix}$$

We will now replace each unscaled elasticity, ε with the equivalent term involving the scaled elasticity, ε . We do this by substituting each unscaled elasticity with the term:

$$\varepsilon_j^i = \varepsilon_j^i \frac{v_i}{x_j}$$

If we assume that the following reactions are first-order, v_2 , v_3 and v_4 then the corresponding elasticities, ε_1^2 , ε_1^3 , and ε_2^4 can be set to one. This means that the $\frac{\partial \mathbf{v}}{\partial \mathbf{x}}$ matrix can be written as:

$$\frac{\partial \mathbf{v}}{\partial \mathbf{x}} = \begin{bmatrix} 0 & \varepsilon_{x_2}^{v_1} F_1 \\ F_2 & 0 \\ F_3 & 0 \\ 0 & F_4 \end{bmatrix}$$

where F_i are the scaling factors, v_i/x_j . To compute the determinant and hence the characteristic equation we form the matrix:

$$s\mathbf{I} - \mathbf{N}\frac{\partial \mathbf{v}}{\partial \mathbf{x}} = \begin{bmatrix} s + F_2 & -\varepsilon_{x_2}^{v_1} F_1 \\ -F_3 & s + F_4 \end{bmatrix}$$

from which the characteristic equation can be shown to be $(a_{11}a_{22} - a_{21}a_{12})$:

$$s^2 + s(F_2 + F_4) + F_2F_4 - F_1F_3\varepsilon_{x_2}^{v_1} = 0$$

The criterion for negative roots (and hence stability) in a quadratic equation is that all the coefficients have the same sign. We should note that all F_i factors are positive therefore it should be clear that the first two coefficients (attached to s^2 and s) are positive. The right most coefficient is $-F_1F_3\varepsilon_{x_2}^{v_1}$ where the elasticity $\varepsilon_{x_2}^{v_1}$ is negative because it represents the negative feedback. The expression, $-F_1F_3\varepsilon_{x_2}^{v_1}$ is therefore positive overall. Since all the coefficients are positive and referring to the rule for quadratic equations (Appendix F) the system must be stable. In fact, unless the negative feedback changes to a positive feedback loop, there is no possibility of this system **ever** showing instability.

Three Gene Pathway

Now let us look at a slightly modified system shown in Figure 23.2 where we have added a second step in the feedback loop.

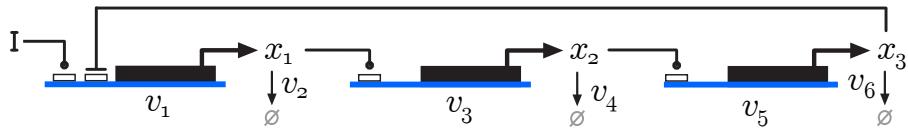


Figure 21.11 Negative Feedback in a Three Gene Cascade

The stoichiometry matrix and elasticity matrix are now given by:

$$\mathbf{N} = \begin{bmatrix} 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 \end{bmatrix}$$

$$\frac{\partial \mathbf{v}}{\partial \mathbf{x}} = \begin{bmatrix} 0 & 0 & \varepsilon_{x_3}^{v_1} F_1 \\ \varepsilon_{x_1}^{v_2} F_2 & 0 & 0 \\ \varepsilon_{x_1}^{v_3} F_3 & 0 & 0 \\ 0 & \varepsilon_{x_2}^{v_4} F_4 & 0 \\ 0 & \varepsilon_{x_2}^{v_5} F_5 & 0 \\ 0 & 0 & \varepsilon_{x_3}^{v_6} F_6 \end{bmatrix}$$

Replacing the unscaled elasticities, \mathcal{E} as before using the expression:

$$\mathcal{E}_j^i = \varepsilon_j^i \frac{v_i}{x_j}$$

and multiplying out the matrices we can obtain:

$$s\mathbf{I} - \mathbf{N} \frac{\partial \mathbf{v}}{\partial \mathbf{x}} = \begin{bmatrix} s - \varepsilon_{x_1}^{v_2} F_2 & 0 & \varepsilon_{x_3}^{v_1} F_1 \\ \varepsilon_{x_1}^{v_3} F_3 & s - \varepsilon_{x_2}^{v_4} F_4 & 0 \\ 0 & \varepsilon_{x_2}^{v_5} F_5 & s - \varepsilon_{x_3}^{v_6} F_6 \end{bmatrix}$$

Assuming first-order kinetics except for the feedback mechanism (meaning all elasticities other than the feedback elasticity equal one) yields:

$$sI - \mathbf{N} \frac{\partial \mathbf{v}}{\partial \mathbf{x}} = \begin{bmatrix} s - F_2 & 0 & \varepsilon_{x_3}^{v_1} F_1 \\ F_3 & s - F_4 & 0 \\ 0 & F_5 & s - F_6 \end{bmatrix}$$

The determinant of this matrix gives use the following characteristic equation:

$$\begin{aligned} s^3 + s^2(F_2 + F_4 + F_6) + s(F_2F_4 + F_2F_6 + F_4F_6) \\ + F_2F_4F_6 - F_1F_3F_5\varepsilon_{x_3}^{v_1} = 0 \end{aligned}$$

Let us now make the following assumption. We will assume that at steady state all the reaction rates, v_1, v_2, v_3, v_4, v_5 and v_6 are equal. This allows us to make some useful simplifications where we can now state that $F_1 = F_6$, $F_3 = F_2$ and $F_5 = F_4$. This changes the characteristic equation to:

$$\begin{aligned} s^3 + s^2(F_2 + F_4 + F_6) + s(F_2F_4 + F_2F_6 + F_4F_6) \\ + F_2F_4F_6(1 - \varepsilon_{x_3}^{v_1}) = 0 \end{aligned}$$

As expected this is a cubic equation. For a general cubic equation such as:

$$a_0x^3 + a_1x^2 + a_2x + a_3 = 0$$

the condition for all negative roots is that:

$$a_0 > 0, a_1 > 0, a_3 > 0 \quad \text{and} \quad a_1a_2 > a_0a_3$$

Negative roots of course indicate stability. We note that the first three conditions are satisfied because the first two coefficients are positive (All F_i terms are positive) and the expression $(1 - \varepsilon_{x_3}^{v_1})$ is net positive (Note $\varepsilon_{x_3}^{v_1} < 0$). With these conditions satisfied the requirement for stability ($a_1a_2 > a_0a_3$) becomes:

$$(F_2 + F_4 + F_6)(F_2F_4 + F_2F_6 + F_4F_6) > F_2F_4F_6(1 - \varepsilon_{x_3}^{v_1})$$

Dividing both sides by $F_2F_4F_6$ yields a new inequality:

$$\frac{F_2}{F_4} + \frac{F_2}{F_6} + \frac{F_4}{F_2} + \frac{F_4}{F_6} + \frac{F_6}{F_2} + \frac{F_6}{F_4} + 3 > 1 - \varepsilon_{x_3}^{v_1}$$

Subtracting one from both sides yields:

$$\frac{F_2}{F_4} + \frac{F_4}{F_2} + \frac{F_6}{F_4} + \frac{F_4}{F_6} + \frac{F_6}{F_2} + \frac{F_2}{F_6} + 2 > -\varepsilon_{x_3}^{v_1} \quad (21.8)$$

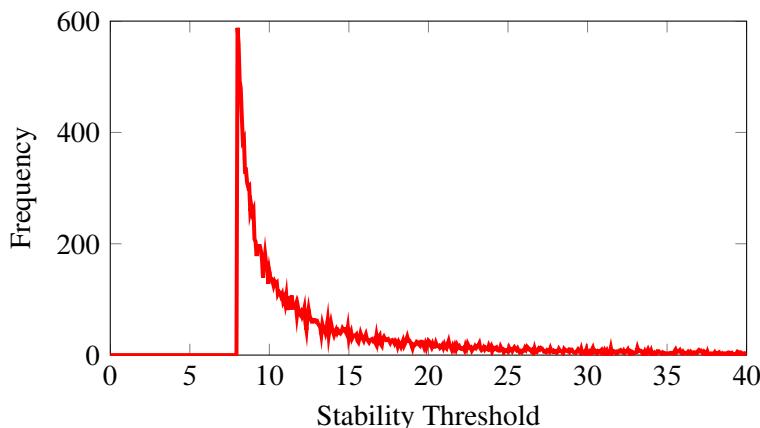


Figure 21.12 Stability Threshold for equation 21.8. Only feedback elasticities greater than eight result in instability.

This is the condition for stability. So long as the left hand side is greater than the feedback elasticity, $\varepsilon_{x_3}^{v_1}$, the system will be stable. The question now is what values can the left-hand side have? One way to pin down the inequality further is to try all combinations of F_i to discover what range of values the left-hand side can yield. Figure 21.12 shows combinations of F_i terms and what is clear from the graph is that no combination falls below 8. Eight is therefore a lower bound for the left-hand term. This means that the feedback elasticity must be greater than -8 in order to achieve stability. Values less than -8 will cause instability.

Another way to look at equation 21.8 is to recall the relation:

$$\frac{1}{a} + a \geq 2$$

Looking more closely at 21.8 we see that each pair of terms is in the form $(1/a) + a$. Therefore each pair of terms must be ≥ 2 . The lowest value each term can have is 2, therefore the minimum value for the sum of three of the paired terms must be 6. With the addition of the remaining 2 in the equation means that the minimum value for the left-hand side of equation (21.8) is 8. This matches the result from the numerical study 21.12. We can therefore summarize that if

$$-\varepsilon_{x_3}^{v_1} < 8 \tag{21.9}$$

then the system is stable. For example if $\varepsilon_{x_3}^{v_1} = -2.0$ then the system will be stable. If the strength of the negative feedback is more than -8 then the system will be unstable. For example if $\varepsilon_{x_3}^{v_1} = -10$ then the system will be unstable. This means that instability in this system requires a fairly strong degree of feedback. A similar analysis can be done on pathways of other lengths, see next section.

From this analysis we can also make a few more observations. We have assumed that the elasticities for the effect of x_1 on v_3 and x_2 on v_5 was first order. It is more likely however that these values are, if anything, greater than one since we know that many transcription factors show cooperativity with respect to gene expression. If we reintroduce these factors we find that the stability expression becomes:

$$8 > -\varepsilon_{x_3}^{v_1} \varepsilon_{x_1}^{v_3} \varepsilon_{x_2}^{v_5}$$

If all three elasticities are greater than one, then the threshold for instability is in fact lower. For example, if $\varepsilon_{x_1}^{v_3}$ and $\varepsilon_{x_2}^{v_5}$ have values of two each, then the threshold equation becomes:

$$8 > -4\varepsilon_{x_3}^{v_1}$$

That is:

$$\begin{aligned} -8 &< 4\varepsilon_{x_3}^{v_1} \\ -2 &< \varepsilon_{x_3}^{v_1} \end{aligned}$$

The threshold for instability is now much lower. The opposite is also true if the transcription factor binding is non-cooperative (elasticities < 1) or that the steps within the transcriptional and translational machinery results in the elasticities having values less than one.

In designing a stable negative feedback system we must ensure two things:

- The feedback elasticity should not be too strong (Threshold dependent on the length of the pathway)
- The number of steps between the feedback signal and its site of action should be as small as is possible

Effect of Pathway Length on Stability

Consider the following pathway with negative feedback which has n species and $n + 1$ steps:

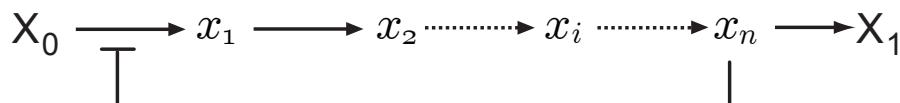


Figure 21.13 Negative feedback system with n species.

The stoichiometry and elasticity matrices have the general form:

$$\mathbf{N} = \begin{bmatrix} 1 & -1 & 0 & 0 & \dots \\ 0 & 1 & -1 & 0 & \dots \\ 0 & 0 & 1 & -1 & \dots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix} \quad \frac{\partial \mathbf{v}}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial v_1}{\partial x_1} & 0 & 0 & 0 & \dots & \frac{\partial v_1}{\partial x_n} \\ \frac{\partial v_2}{\partial x_1} & \frac{\partial v_2}{\partial x_2} & 0 & 0 & 0 & \dots \\ 0 & \frac{\partial v_3}{\partial x_2} & \frac{\partial v_3}{\partial x_3} & 0 & 0 & \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix}$$

Replacing $\partial v / \partial x$ with \mathcal{E} and assuming all reactions are product insensitive, we obtain:

$$\frac{\partial \mathbf{v}}{\partial \mathbf{x}} = \begin{bmatrix} 0 & 0 & 0 & 0 & \dots & \mathcal{E}_n^1 \\ \mathcal{E}_1^2 & 0 & 0 & 0 & 0 & \dots \\ 0 & \mathcal{E}_2^3 & 0 & 0 & 0 & \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix}$$

The Jacobian, $\mathbf{N} \frac{\partial \mathbf{v}}{\partial \mathbf{x}}$ is given by:

$$\mathbf{N} \frac{\partial \mathbf{v}}{\partial \mathbf{x}} = \begin{bmatrix} -\mathcal{E}_1^2 & 0 & 0 & 0 & \dots & \mathcal{E}_n^1 \\ \mathcal{E}_1^2 & -\mathcal{E}_2^3 & 0 & 0 & \dots & \\ 0 & \mathcal{E}_2^3 & -\mathcal{E}_2^4 & 0 & \dots & \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix}$$

We now replace terms \mathcal{E}_j^i with $\varepsilon_j^i F_j$ and compute $s\mathbf{I} - \mathbf{N} \frac{\partial \mathbf{v}}{\partial \mathbf{x}}$ from which we can derive the characteristic equation (in the dynamical systems community the matrix is computed using $(A - \lambda I)$ so that the signs will be reversed, however the result is identical):

$$s\mathbf{I} - \mathbf{N} \frac{\partial \mathbf{v}}{\partial \mathbf{x}} = \begin{bmatrix} s + \varepsilon_1^2 F_1 & 0 & 0 & 0 & \dots & -\varepsilon_n^1 F_n \\ -\varepsilon_1^2 F_1 & s + \varepsilon_2^3 F_2 & 0 & 0 & \dots & 0 \\ 0 & -\varepsilon_2^3 F_2 & s + \varepsilon_3^4 F_3 & 0 & \dots & 0 \\ 0 & 0 & -\varepsilon_3^4 F_3 & s + \varepsilon_4^5 F_4 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix}$$

To make matters simpler, let us assume that all reactions other than the feedback step are first-order with equal rate constants. In such a situation, the substrate elasticities, $\varepsilon_j^i = 1$ and all species concentrations, x_j are the same. This means that $\varepsilon_1^2 = \varepsilon_2^3 = \varepsilon_3^4 = \varepsilon_j^i = 1$ and $F_1 = F_2 = F_3 = F_n = F$. The matrix can be rewritten as:

$$\begin{bmatrix} s + F & 0 & 0 & 0 & \dots & 0 & -\varepsilon_n^1 F \\ -F & s + F & 0 & 0 & \dots & 0 & 0 \\ 0 & -F & s + F & 0 & \dots & 0 & 0 \\ 0 & 0 & -F & s + F & \dots & 0 & 0 \\ \dots & & & & & & \end{bmatrix}$$

As before the characteristic equation is obtained from the determinant which in this case has a simple form:

$$(s + F)^n - \varepsilon_n^1 F^n = 0$$

Following Savageau [55] we now solve for s using de Moivre's theorem. We first express the above equation by rearranging and taking the n^{th} root on both sides. Because ε_n^1 is negative, when taking the n^{th} root we must take into account that there are n possible roots. To include this fact in the expression, we add a negative sign to ε_n^1 and insert $(-1)^{1/n}$ to compensate. The multiple roots are now found in the term $(-1)^{1/n}$:

$$s = [(-1)^{1/n} (-\varepsilon_n^1)^{1/n} - 1]F$$

The n roots of $(-1)^{1/n}$ can be obtained using de Moivre's theorem (See Appendix F.6):

$$s_m = \left[(-\varepsilon_n^1)^{1/n} \left(\cos \left(\frac{2m+1}{n}\pi \right) + i \sin \left(\frac{2m+1}{n}\pi \right) \right) - 1 \right] F \quad \text{for } m = 0, 1, 2, \dots, n$$

For the system to be stable the real part of the s_m must be negative. Since F is positive, for stability it must be true that:

$$(-\varepsilon_n^1)^{1/n} \cos \left(\frac{2m+1}{n}\pi \right) - 1 < 0$$

The transition to *instability* will occur when the largest s_m becomes positive, this will happen at $m = 0$ since this maximizes the cosine term. Setting $m = 0$ yields:

$$(-\varepsilon_n^1)^{1/n} \cos \left(\frac{\pi}{n} \right) - 1 < 0$$

The left-hand side must be greater than +1 to make the overall expression greater than 0, that is:

$$(-\varepsilon_n^1)^{1/n} \cos \left(\frac{\pi}{n} \right) > 1 \tag{21.10}$$

Length of Gene Cascade	Instability $-\varepsilon_{\text{feedback}}$	Threshold
1	stable	
2	stable	
3	8.0	
4	4.0	
5	2.9	
6	2.4	
7	2.1	
:	:	
∞	1.0	

Table 21.2 Relationship between pathway length and the degree of feedback inhibition on the threshold for stability (See Figure 21.13). $-\varepsilon_{\text{feedback}}$ is the elasticity of the feedback inhibition. The analysis assumes first order kinetics for all reactions other than the feedback reaction. Additionally it also assumes that the rate constants for the first-order reactions all have the same value. This ensures that the species levels are the same.

We rearrange this inequality noting the sign change rules for inequalities to obtain:

$$-\varepsilon_n^1 < \frac{1}{\cos^n(\pi/n)} = \sec^n(\pi/n)$$

We therefore arrive at the central result [25, 54, 62, 59]:

$$-\varepsilon_n^1 < \sec^n(\pi/n)$$

Let us consider the cases $n = 1$ and $n = 2$. For both $n = 1$ and $n = 2$ the left-hand side of equation (21.10) can never exceed one, in both cases the left-hand side is negative. When $n = 3$, $\sec^3(\pi/3)$ equals 8, that is:

$$-\varepsilon_3^1 < 8$$

This confirms the result we obtained previously in equation (21.9). To give another example, if the length of the pathway is 7 steps, then $\sec^7(\pi/7) = 2.08$. That is the feedback elasticity must be less than -2.08 for the system to become unstable.

Table 21.2 summarizes some of the threshold points for a negative feedback system with varying pathway lengths.

21.11 Biological Interpretation of the Transfer Function

The transfer function $H_x(s)$ for the simple feedback system shown in Figure 21.2 is given by:

$$H_x(s) = \frac{1}{s + \left(\frac{\partial v_2}{\partial x} - \frac{\partial v_1}{\partial x} \right)} = \frac{1}{\left(s + \frac{\partial v_2}{\partial x} \right) - \frac{\partial v_1}{\partial x}} \quad (21.11)$$

Dividing top and bottom by $(s + \partial v_2 / \partial x)$ we can write the transfer function into a more familiar form (See (3.2) and (21.3)):

$$H(s) = \frac{1/(s + \partial v_2 / \partial x)}{1 - \partial v_1 / \partial x / (s + \partial v_2 / \partial x)} = \frac{G(s)}{1 - \frac{\partial v_1}{\partial x} G(s)} = \frac{G(s)}{1 + KG(s)}$$

where:

$$K = -\frac{\partial v_1}{\partial x}, \quad \text{and} \quad G(s) = \frac{1}{(s + \partial v_2 / \partial x)}$$

Note that the term $\partial v_1 / \partial x$ is negative so that the signs match the signs in the standard form. The expression:

$$KG(s)$$

is the loop gain for the feedback system. From this comparison we can make the following deductions. The feedback elasticity $\partial v_1 / \partial x$ represents the strength of the feedback; the first-order term $G(s)$ is the open loop gain and the set point is given by the input elasticity and step input, $\delta I \partial v_1 / \partial I$. We can also state that the feedback system shown in Figure 21.2 is using a proportional controller to set the response.

We can illustrate this with a more concrete example. Consider the nonlinear model which describes the rate of change of x , where X_o is the input species and x represses the first reaction v_1 :

$$\frac{dx}{dt} = v_1 - v_2 = \frac{X_o}{1 + x^n/k_1} - k_2 x$$

Linearizing the model yields:

$$\frac{d\delta x}{dt} = \frac{\partial v_1}{\partial x} \delta x + \frac{\partial v_2}{\partial x} \delta x + \frac{\partial v_1}{\partial X_o} \delta X_o$$

$$\frac{d\delta x}{dt} = \left(\frac{\partial v_1}{\partial x} + \frac{\partial v_2}{\partial x} \right) \delta x + \frac{\partial v_1}{\partial X_o} \delta X_o$$

$$\text{Let } A = \frac{\partial v_2}{\partial x} - \frac{\partial v_1}{\partial x} \quad \text{and} \quad B = \frac{\frac{\partial v_1}{\partial X_o}}{\frac{\partial v_2}{\partial x} - \frac{\partial v_1}{\partial x}}$$

So that:

$$\frac{d\delta x}{dt} = A(x - Bx_o)$$

At steady state

$$x = BX_o$$

Define the following terms:

$$G = \frac{1}{\partial v_2 / \partial x} \quad \text{and} \quad K = -\frac{\partial v_1}{\partial x}$$

Dividing top and bottom by $\partial v_2 / \partial x$ gives us:

$$x = \frac{\frac{\partial v_1 / \partial X_o}{\partial v_2 / \partial x}}{1 + \frac{K}{\partial v_2 / \partial x}} X_o$$

$$x = \frac{\frac{\partial v_1}{\partial X_o} G}{1 + KG} X_o$$

21.12 PID Controller

Integral Control

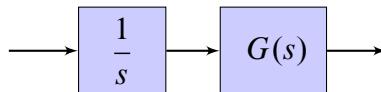
In integral control the signal that feeds the process block, A , is the integral of the error signal, e . In practice this means is that a running total of the error signal is maintained and it is this total that is fed to the process block. Mathematically we can express this as:

$$u = K_I \int e(t) dt$$

Integrating the error in the negative feedback will ensure that the error between the set point and output is zero. To show this recall that the Laplace transform of an integral is $F(s)/s$:

$$\mathcal{L} \left[\int_0^t f(\tau) d\tau \right] = \frac{F(s)}{s}$$

It is therefore easy to add an integrator to $G(s)$, by including a $1/s$ component in $G(s)$:



The error, $E(s)$ (Equation 21.4) is given by:

$$E(s) = R(s) \frac{1}{1 + K(1/s)G(s)} = R(s) \frac{s}{s + KG(s)}$$

Using the final-value theorem as before, where we set $R(s) = 1/s$:

$$e_{ss} = \lim_{s \rightarrow 0} s \frac{s/s}{s + KG(s)} = 0$$

That is the presence of the integrator reduces the error to zero. Integral control is a very powerful control strategy for removing any difference between the set point and the output. So long as there is some difference between the set point and the output, the integral control will accumulate the error forcing the process, A to match the set point. Once the set point and output are equal, the error goes to zero and the integration stops. This behavior is in contrast to proportional control where there will always be a small difference between the output and the set point. It is possible to close the gap by increasing the K_p constant but, as we will see, at the expense of instability.

Derivative Control

The third type of control is called derivative control. What this does is feed that rate of change of the error into the process block A . In practice this can be simply achieved by sampling the error at two different times and passing on the difference to the process block. Mathematically we can express this using the equation:

$$u = K_D \frac{de(t)}{dt}$$

The advantage of using derivative control is that it can slow down fast changes and so help prevent overshoot and undershoot in the output. For example, assume the set point is set to 5 units. If the feedback returns 10 units indicating that the output is above the set point, then the error will be $5 - 10 = -5$, ie the output should be reduced. Assume that a moment later, the output rises to 25 units, as a result the error changes to $5 - 25 = -20$. The change in the error per unit time is therefore $-20 - (-5) = -15$ units per time. The negative sign indicates that the process should slow down the rate at which the output is rising. One disadvantage of derivative control is that computing derivatives tends to amplify any noise in the signals so that is the noise and gain of the derivative control is sufficiently high, the output of the system can be swamped by noise and in the worse case become unstable. Another disadvantage is that a derivative controller puts a damper on the response time so that the system will respond slower to changes.

Collectively one can think of the three modes of feedback in the following way:

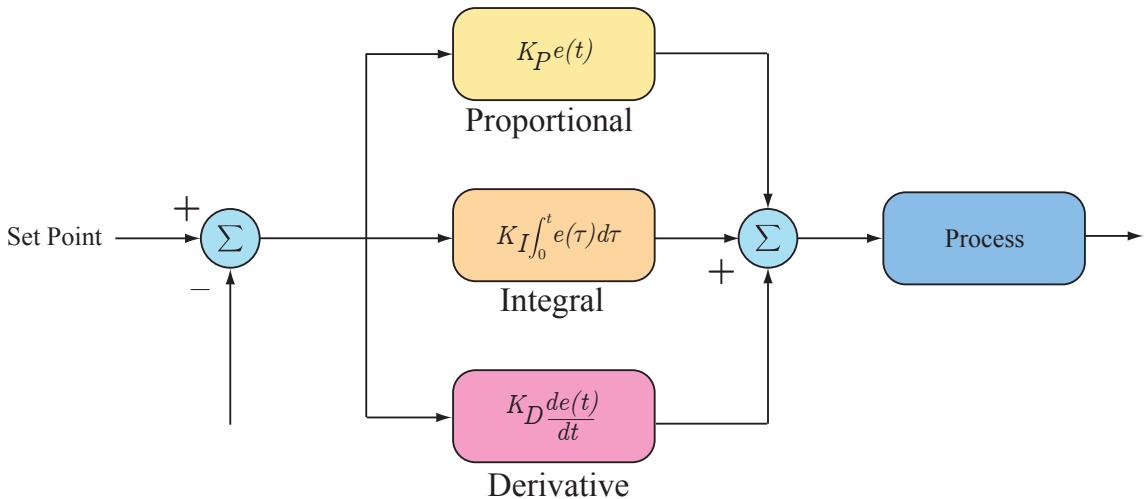


Figure 21.14 PID Control.

- Proportional Control - responds to the present
- Integral Control - responds to the past
- Derivative Control - responds to the future

Given the properties of the different ways to handle error, a PID controller ensures stability, accuracy and prevents overshoot and undershoot in transient changes. Mathematically a PID controller is represented as:

$$u = K_P e(t) + K_I \int e(t) dt + K_D \frac{de(t)}{dt}$$

Derivative control: Improves overshoot and transient response but does nothing to improve the steady state error.

Integral control: Improves the steady state error but makes the system less stable in general.

In practice PID controllers can be made by using three op amp circuits in parallel. A typical circuit is shown in Figure 21.15.

Exercises

1. Determine whether the following two feedback systems are stable or unstable in response to a unit step input in $R(s)$.
 - a)

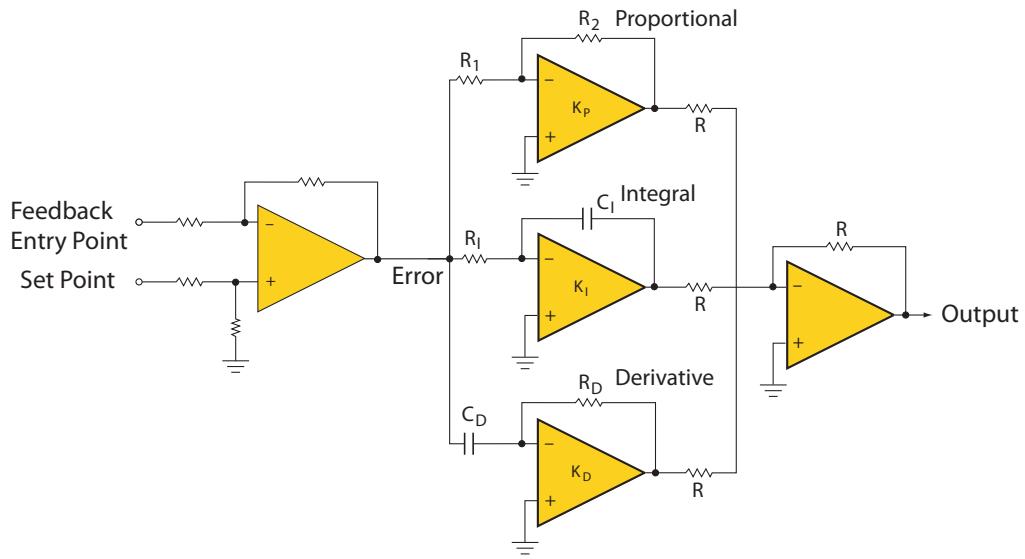
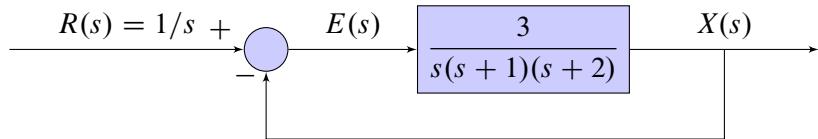
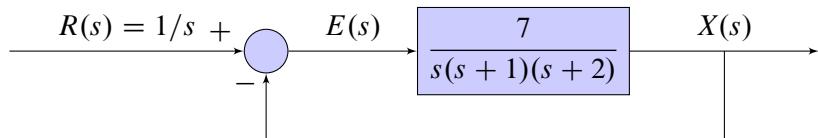


Figure 21.15 Op Amp based PID Controller.



b)



2. A controller has the PI function given by:

$$c(t) = K_p e(t) + K_p \int_0^t e(\tau) d\tau$$

where $e(t)$ is the error between the set point and the feedback:

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

What is the transfer function for the PI controller?

3. What do we mean by the type for an open loop transfer function?
4. Derive the step, ramp and parabolic error response for a type 1 system.
5. Derive the step, ramp and parabolic error response for a type 2 system.

22

Positive Feedback

“At my signal... unleash hell!”

– Maximus in *Gladiator*, 2000

22.1 Basic Properties

Many biological systems show positive feedback regulation. Figure 22.1 shows a simple gene regulatory network where the gene product activates its own production. As the transcription factor x accumulates, it binds to an operator site upstream of the gene which increases its synthesis. The more transcription factor made, the higher the rate of expression.

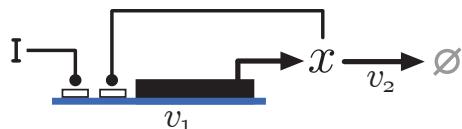


Figure 22.1 System with Positive Feedback

At first glance this would seem to be a very unstable situation. One might imagine that the transcription factor concentration would continue to increase without limit. However, the physical limits, in this case the saturation of the translation, transcription, and degradation machinery, ultimately limits the upper value for the concentration of transcription factor. To investigate the properties of this net-

work we will construct a simple model. This model uses the following kinetic laws for the synthesis and degradation steps:

$$v_1 = b + k_1 \frac{x^4}{k_2 + x^4}$$

$$v_2 = k_3 x$$

b is the basal expression rate of v_1 , that is the rate of expression can never go below this rate. The second term is a Hill like equation where the variable in the Hill equation is x itself, so that as x increases the value of the Hill term also increases. Let us now run a series of time course simulations at many different starting points for x . Figure 22.2 shows the plots generated using the script in Listing 22.1. The plots show two steady states, a high state at around 40, and a low state at around 3. Depending on the initial condition we can either evolve to the upper or lower state.

```
import tellurium as te
import numpy as np

rr = te.loada ('''
J1: $Xo -> x; 0.1 + k1*x^4/(k2+x^4);
x -> $w; k3*x;

k1 = 0.9;
k2 = 0.3;
k3 = 0.7;
''')

rr.x = 0.05;
m1 = rr.simulate (0, 15, 30, ['time', 'x']);

for i in range(1, 20):
    rr.x = i*0.2;
    m2 = rr.simulate (0, 15, 30, ["x"]);
    m1 = np.column_stack ((m1, m2))
te.plotArray (m1)
```

Listing 22.1 Python script used to generate Figure 22.2.

To understand what is going on we need to look more closely at the individual rate laws. The first equation, v_1 , mimics a reaction that is activated by its product. In this case the activation shows a degree of sigmoidicity in the response. The second equation, v_2 is a simple first-order rate law. The steady state of this simple model can be computed at

$$\frac{dx}{dt} = v_1 - v_2 = 0 \text{ that is } v_1 = v_2$$

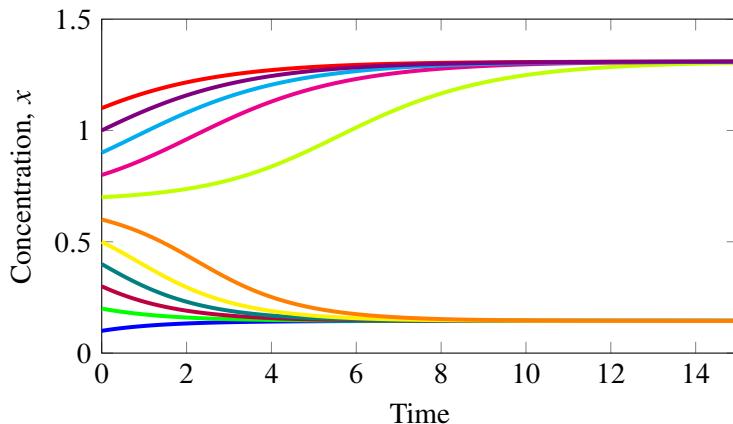


Figure 22.2 Time course data generated from Python model 22.1. Each line represents a different initial concentration for x . Some trajectories transition to the low state while others to the upper state.

If v_1 and v_2 are plotted against x we obtained Figure 22.3. The intersection points where v_1 and v_2 are equal are marked with filled circles and by definition indicate the steady state solutions because at these points, $v_1 = v_2$. This simple plot shows us that there must be three possible steady states in the system.

We can investigate the properties of the three steady states by varying the slope of v_2 . This can be done by changing k_3 . This is shown in Figure 22.4. At a high k_3 value, only one intersection point remains (Panel c), the low intersection point. If the value of k_3 is low, only the high intersection point remains (Panel a). However with the right set of parameter values, we can make a system with three steady state values (Panel b).

We can determine the three different steady state stabilities by doing a simple graphical analysis on Figure 22.3. Figure 22.5 shows the same plot but with perturbations added to the image.

Starting with the first steady state in the low left corner of Figure 22.5, consider a perturbation made in x , δx . This means that both v_1 and v_2 increase, however $v_2 > v_1$ meaning that after the perturbation, the rate of change in x is *negative*. Since it is negative, this restores the perturbation back to the steady state. The same logic applies to the upper steady state. This tells us that the lower and upper steady states are both stable.

What about the middle steady state? Consider again a perturbation, δx . This time $v_1 > v_2$ which means that the rate of change of x is *positive*. Since it is positive, the perturbation, instead of falling back, continues to grow until x reaches the upper steady state. We conclude that the middle steady state is unstable. This system possess three steady states, one unstable and two stable. Such a system is known as a **bistable system** because it can rest in one of two stable states but not the third.

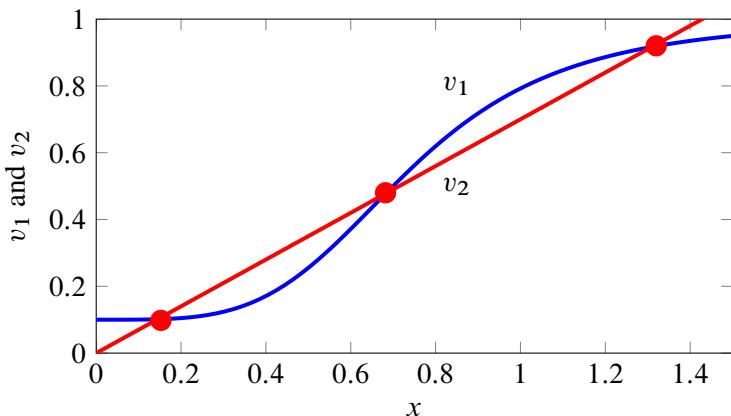


Figure 22.3 Reaction velocities, v_1 and v_2 , as a function of x for the system in the Python script 22.1. The intersection points marked by full circles indicate possible steady states where $v_1 = v_2$. Computed using the SBW rate law plotter. $k_1 = 0.9$; $k_2 = 0.3$; $k_3 = 0.7$; $b = 0.1$.

We can get an estimate for the values of all three steady states from Figure 22.3. Reading directly from the graph we find x values at 0.145, 0.683, and 1.309. It is also possible to use the steady state solver from TELLURIUM to locate the steady states. Listing 22.2 shows a simple script to compute them. By setting an appropriate initial condition, we can use Tellurium to pin point all three steady states. For example, if we use an initial value of x at 0.43, the steady state solver will locate the third steady state at 0.683. Steady state solvers such as the one included with Tellurium can be used to find unstable states, providing the initial starting point is close enough.

```
import tellurium as te

rr = te.loada('''
$Xo -> x; 0.1 + k1*x^4/(k2+x^4);
x -> $w; k3*x;

// Initialization here
k1 = 0.9; k2 = 0.3;
k3 = 0.7; x = 0;
''')

# Compute steady state
rr.steadyState();
```

Listing 22.2 Basic bistable model for computing the steady state value of x .

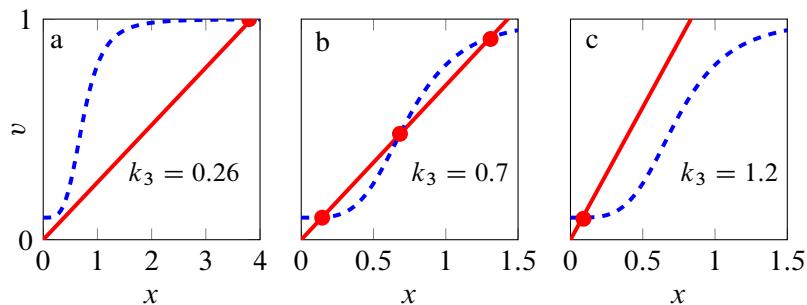


Figure 22.4 v_1 and v_2 plotted against x concentration. Intersection points on the curves mark the steady state points. Panel a) One intersection point at a high steady state; b) Three steady states; c) One low steady state.

22.2 Stability of Positive Feedback

What determines the stability of a positive feedback system? Let us consider the same genetic network with positive feedback as before (Figure 22.1). The differential equation for this system is:

$$\frac{dx}{dt} = v_1(x) - v_2(x)$$

where we have explicitly shown that each reaction rate is a function of x . To determine whether the system is stable to small perturbations we can differentiate the equation with respect to x to form the Jacobian. Notice there is only one element in the Jacobian because we only have one state variable:

$$\frac{dx/dt}{dx} = \frac{\partial v_1}{\partial x} - \frac{\partial v_2}{\partial x}$$

The terms on the right are unscaled elasticities (4.15). If the expression is positive, the system is unstable because it means that dx/dt is increasing if we increase x . We can scale both sides to yield:

$$J_s = \varepsilon_x^1 - \varepsilon_x^2$$

where the right-hand term now includes the scaled elasticities (4.14). The criteria for stability is again that $\varepsilon_x^1 > \varepsilon_x^2$. Therefore, if the positive feedback is stronger than the effect of x on the degradation step v_2 , the system will be unstable.

Recall that the elasticities are a measure of the kinetic order of the reaction. Thus an elasticity of one means the reaction is first-order. A saturable irreversible Michaelis-Menten reaction will have a variable kinetic order between one and zero (near saturation). A Hill equation can, depending on the Hill coefficient, have kinetic orders greater than one (Table 22.1). Knowing this information, there are at least two ways to make sure that the elasticity for the feedback elasticity, v_1 , is greater than the elasticity for the degradation step, v_2 :

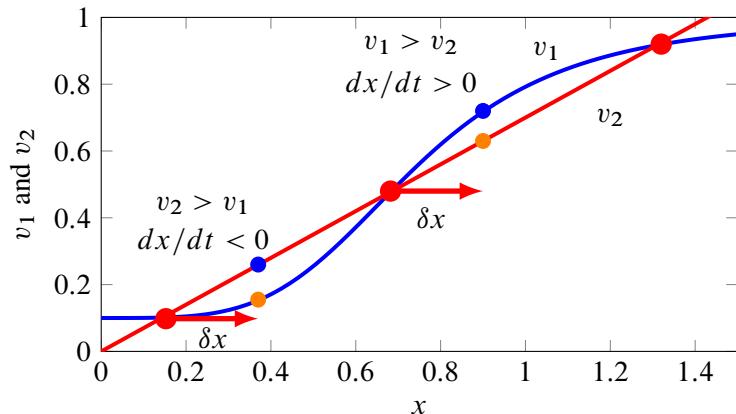


Figure 22.5 A graphical understanding of the stability of the steady states. See text for details. Computed using the SBW rate law plotter. $k_1 = 0.9$; $k_2 = 0.3$; $k_3 = 0.7$; $b = 0.1$.

Kinetic Order	Elasticity
First-Order	1.0
Zero-Order	0.0
Sigmoidal	> 1.0

Table 22.1

1. v_1 is modeled using a Hill equation with a Hill coefficient > 1 and v_2 is first-order or less.
2. A Hill coefficient = 1 on v_1 , with Michaelis-Menten saturable kinetics on v_2 to ensure less than first-order kinetics on v_2 .

By substituting the three possible steady state values for x into the equation for dx/dt , we can compute the value for the Jacobian element in each case (Table 22.2).

Steady State x	Jacobian: $(dx/dt)/dx$	Elasticity, $\varepsilon_x^{v_1}$
0.145	-0.664	0.052
0.683	0.585	1.835
1.309	-0.47	0.33

Table 22.2 Table of steady state values of x and corresponding values for the Jacobian element. Negative Jacobian values indicate a stable steady state, positive elements indicate an unstable steady state. The table shows one stable and two unstable steady states.

The unstable steady state at $x = 0.683$ has an elasticity for v_1 of 1.835. Note this value is greater than the elasticity of the first-order degradation reaction, v_2 , which equals one. Therefore this state is unstable. A more formal approach is given below.

The stoichiometry matrix and elasticity matrix are given by:

$$\mathbf{N} = [1 \quad -1]$$

$$\frac{\partial \mathbf{v}}{\partial \mathbf{x}} = \begin{bmatrix} \varepsilon_x^{v_1} \\ \varepsilon_x^{v_2} \end{bmatrix}$$

Inserting these terms into:

$$\left(s\mathbf{I} - \mathbf{N} \frac{\partial \mathbf{v}}{\partial \mathbf{x}} \right)^{-1}$$

and computing the determinant we can obtain the characteristic equation:

$$s + \varepsilon_x^{v_2} - \varepsilon_x^{v_1} = 0$$

In order for the system to be stable the root, s , must be negative. Let us first convert the unscaled elasticities (ε), into scaled versions. We do this by swapping each unscaled elasticity with the term:

$$\mathcal{E}_j^i = \varepsilon_j^i \frac{v_i}{x_j}$$

This gives us:

$$s + \varepsilon_x^{v_2} \frac{v_2}{x} - \varepsilon_x^{v_1} \frac{v_1}{x} = 0$$

Since at steady state $v_1 = v_2$ we can replace the terms $v_1/x, v_2/x$ with the same factor, F and the root is then (note the change in signs):

$$s = F (\varepsilon_x^{v_1} - \varepsilon_x^{v_2})$$

For stability the root must be negative. Since $F > 0$, this is equivalent to stating that:

$$\varepsilon_x^{v_1} - \varepsilon_x^{v_2} < 0$$

for stability. Noting that $\varepsilon_x^{v_1}$ is positive, it must be that case that

$$\varepsilon_x^{v_1} < \varepsilon_x^{v_2}$$

for the system to be stable. Conversely, if the positive feedback elasticity is larger than the degradation elasticity then the system will be unstable. If we assume that the degradation step, v_2 is first-order then $\varepsilon_x^{v_2} = 1$ and the condition for instability will simply be:

Condition for instability (assuming first-order kinetics for v_2): $\varepsilon_x^{v_1} > 1$

This is easily achieved if the positive feedback displays cooperativity. Table 22.2 shows the elasticity for each of the steady states. Note that for the unstable state, the elasticity is 1.68, a value greater than one.

Bifurcation Plot

A common way to depict multiply steady states is system is to use a bifurcation plot. In its simplest form, a bifurcation plot is just a plot of the steady state value of a system variable, such as a concentration or flux versus a parameter of the system. The bistable system we could plot the steady state value of x versus the value of the parameter k_3 . Figure 22.6 shows a plot where we track the value of x as a function of k_3 .

Figure 22.6 shows that at some value of the parameter k_3 , the system has three possible steady states, outside this range only a single steady state persists. Bifurcation diagrams are extremely useful for uncovering and displaying such information. Drawing bifurcation diagrams is not easy, however, and there are some software tools that can help. Figure 22.6 for example was generated using the SBW

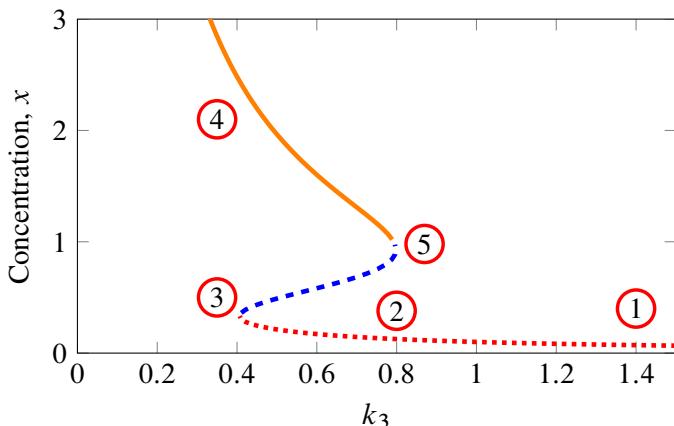


Figure 22.6 Plotting intersection points from Figure ?? as a function of k_3 . Dotted line marks the lower intersection point, dashed line the middle intersection points, and solid line the upper intersection point. Computed using the SBW AUTO C# Tool.

Auto C# tool¹. Another useful tool for drawing bifurcation diagrams is Oscill8². Both tools can read SBML. Figure 22.6 was generated first by entering the model into Tellurium (Shown in listing 22.2) to generate the SBML. The model was then passed to Auto C# to produce the bifurcation diagram.

The bifurcation plot shows how the steady state changes as a function of a parameter, in this case k_3 . Of interest is the following observation. If we start k_3 at a high value of 1.4 (Marker 1), we see that there is only one low steady state. As k_3 is lowered, we pass the point at approximately $k_3 = 0.8$ (Marker 2) where three steady states emerge. We continue lowering k_3 , and see that the concentration of x rises very slowly until about 0.4 (marker 3). At this point the system jumps to a single steady state, but now at a high level (Marker 4). The interesting observation is that if we now increase the value of k_3 , we do not traverse the same path. As we increase k_3 beyond 0.4, we do not drop back to the low state, but continue along the high state until we reach $k_3 = 0.8$ (Marker 5), at which point we jump down to the low state (Marker 2). The direction in which we traverse the parameter k_3 affects the type of behavior we observe. This special phenomena is called **hysteresis**, Figure 22.7.

Hysteresis is where the behavior of a system depends on its past history.

¹http://jdesigner.sourceforge.net/Site/Auto_C.html

²<http://oscill8.sourceforge.net/>

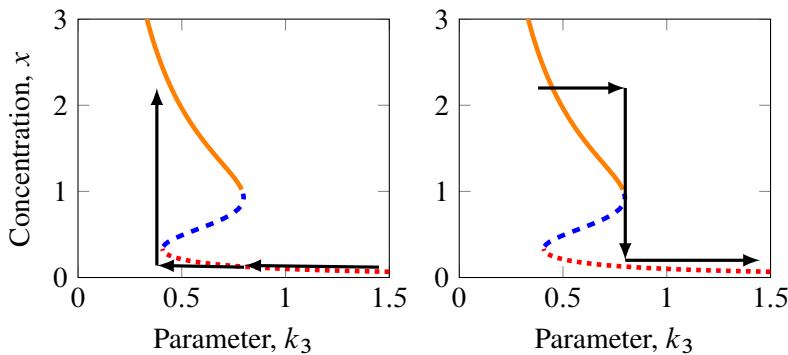


Figure 22.7 Depending on whether we increase or decrease k_3 , the steady state path we traverse will be different. This is a characteristic of hysteresis.

Irreversible Bistability

It is possible to design an irreversible bistable switch. Figure 22.9 shows the bifurcation plot for such a system. This is modified from the ‘Mutual activation’ model in the review by Tyson [64], Figure 1e. In this example increasing the signal results in the system switching to the high state at around 2.0. If we reduce the signal from a high level, we traverse the top arc. If we assume the signal can never be negative, we will remain at the high steady state even if the signal is reduced to zero. The bifurcation plot in the negative quadrant of the graph is physically inaccessible. This means it is not possible to transition to the low steady state by decreasing signal. As a result, the bistable system is **irreversible**, that is, once it is switched on, it will always remain on.

```
import tellurium as te

r = te.loada '''
$X -> R1;  k1*EP + k2*Signal;
R1 -> $w;  k3*R1;
EP -> E;    Vm1*EP/(Km + EP);
E -> EP;   ((Vm2+R1)*E)/(Km + E);

Vm1 = 12;  Vm2 = 6;
Km = 0.6;
k1 = 1.6; k2 = 4;
E = 5;     EP = 15;
k3 = 3;    Signal = 0.1;
'''
```

Listing 22.3 Script for Figure 22.9.

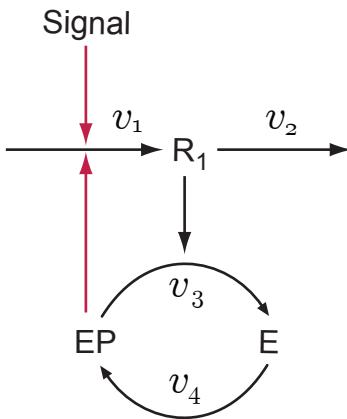


Figure 22.8 System with Positive Feedback using a covalent modification cycle, E , EP .

The Python script for the model is shown in listing 22.3. To create Figure 22.9, first install Oscill8³, then launch Jarnac. Load the script into Jarnac. Run the script (press green button in toolbar) to put the model into memory. Go to the SBW menu and select Oscill8. Once in Oscill8, select Run; 1 Parameter. In the new dialog box select continuation and the parameter “Signal”. Then select run to view the bifurcation plot.

Toggle Switch

The final example we will consider is the toggle switch. This example illustrates the idea of an attractor basin in a phase plot. The system in question is shown in Figure 22.10 and is comprised of just two nodes, S_1 and S_2 . Each node inhibits the other. This is a high level diagram, but mechanistic realizations can be made using protein or gene regulatory networks. In fact, one of the first synthetic biology constructs was the toggle switch made from an engineered gene regulatory network [19]. Intuitively, one can imagine the type of behavior this system might exhibit. For example, if S_1 has a high concentration then this will repress S_2 . Since S_2 is now low, repression of S_1 is negligible. This state appears stable. Alternatively we could imagine that S_2 is at a high concentration. This will repress S_1 and thus we have another state that appears stable.

We can better study the behavior of the toggle switch by building a computer simulation. Using the

³<http://oscill8.sourceforge.net/>

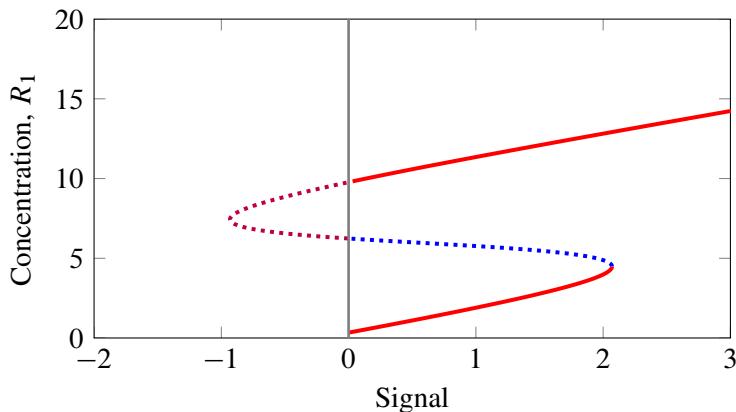


Figure 22.9 Bifurcation diagram for species R_1 with respect to the signal. Signal from the model shown in Tellurium/Python script 22.3. The continuous line represents stable steady state points, the dotted line the unstable steady states. Plotted using Oscil8 <http://oscill8.sourceforge.net/>.



Figure 22.10 The toggle switch. Two mutually inhibited nodes, S_1 and S_2 .

set of differential equations shown in equations (22.1), we describe the model as:

$$\begin{aligned} \frac{dS_1}{dt} &= \frac{k_1}{1 + S_2^{n_1}} - k_2 S_1 \\ \frac{dS_2}{dt} &= \frac{k_3}{1 + S_1^{n_2}} - k_4 S_2 \end{aligned} \tag{22.1}$$

A suitable set of parameter values are given by $k_1 = 6$; $k_3 = 6$; $k_2 = 2$; $k_4 = 2$. Figure 22.12 shows the phase plot for this system. The three points marked by round circles represent the steady state locations. Note that there is a high S_1 /low S_2 state, and a low S_1 /high S_2 state. These correspond to the states we intuitively described before. There is also a third point which represents a medium level of S_1 and S_2 . The arrows in the diagram represent the direction of change of S_1 and S_2 in time. The diagram has been subdivided into four **basins of attraction**. For example, the phase plot highlights one of the direction arrows in the basin labelled two. The arrows initially point right to left and down. This tells us that if we started a time course simulation at the first arrow, the concentration of S_1 and

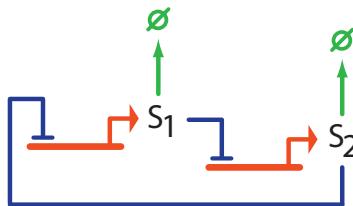


Figure 22.11 The toggle switch constructed from a gene regulatory network.

S_2 would both decline. Initially, the arrow points roughly in the direction of the unstable state but as it moves, its direction changes and eventually moves towards the upper stable state to the left.

The phase plot gives us a condensed view of how any initial condition will evolve. Points that start in basin one will converge to the third state, basin two points converge on the first state, basin three to the third state, and finally basin four to the first state.

Nullclines

The toggle switch model from the previous section also allows us to introduce the concept of nullclines. We've seen various mechanisms for visualizing a system, especially related to its steady state stability. For example, in Figure 22.5 we show how the stability of a one dimensional system can be visualized, the bifurcation plot in Figure 22.6, and the phase plot in Figure 22.12. There is yet another plot that can help us understand the instability of a two dimensional system and that has to do with plotting the nullclines. The nullcline is the solution to a differential equation when set to zero as a function of the two system variables. For example, the toggle switch differential equation for S_1 is:

$$\frac{dS_1}{dt} = \frac{k_1}{1 + S_2^{n_1}} - k_2 S_1$$

We can set this to zero and find all combinations of S_1 and S_2 that satisfy this equation. These points form a line on a two dimensional plane. Such a line is called the **nullcline**. Figure 22.13 shows two nullclines corresponding to the two differential equations for the toggle model. Note that where the nullclines intersect, we find the steady state because at these points, both equations yield the same values for S_1 and S_2 .

Although useful, many of the plots we have reviewed are limited to systems of two or three variables. The nullclines in an n -dimensional system cannot easily be visualized. Likewise, a two dimensional phase plot can only take a slice through the dynamics of a system with many variables. Nevertheless, these techniques, in particular bifurcation plots, are extremely useful in delineating the potential behavioral modes of networks. See [8, 9] for examples of bifurcation plots in studying protein signaling pathways.

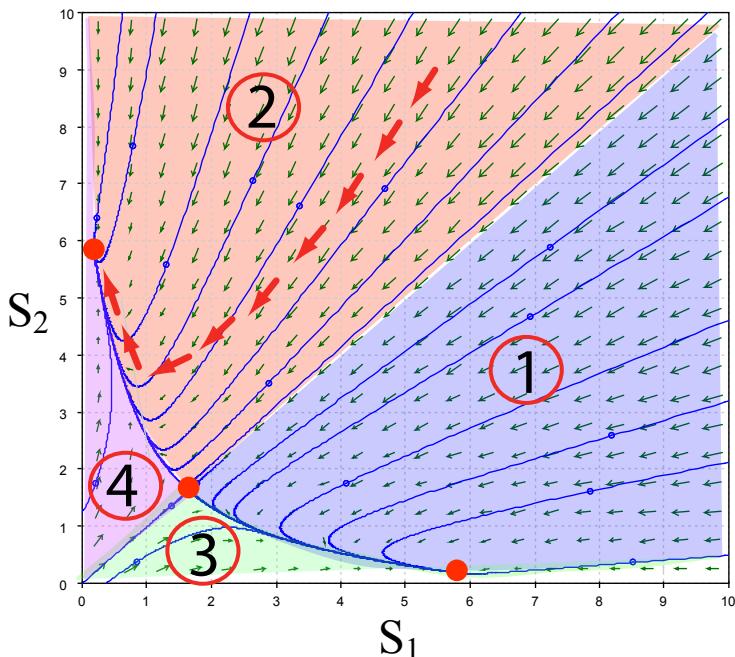


Figure 22.12 Phase portrait for a toggle switch, equation (22.1). The four numbered areas mark the four basins of attraction. The line of thicker arrows illustrates one possible time course trajectory given an initial starting point. Diagram generated by PPlane <http://math.rice.edu/~dfield/dpp.html>, $k_1 = 6; k_3 = 6; k_2 = 2; k_4 = 2$.

Further Reading

1. Laurent M and Kellershohn N (1999) Multistability: a major means of differentiation and evolution in biological systems. *Trends in Biochemical Sciences* 24, 418-422
2. Ferrell JE (2002) Self-perpetuating states in signal transduction: positive feedback, double-negative feedback and bistability. *Current Opinion in Chemical Biology* 6, 140–148
3. Tyson JJ, Chen KC, N B (2003) Sniffers, buzzers, toggles and blinks: dynamics of regulatory and signaling pathways in the cell. *Current Opinion in Cell Biology* 15, 221–231

Exercises

1. Answer this

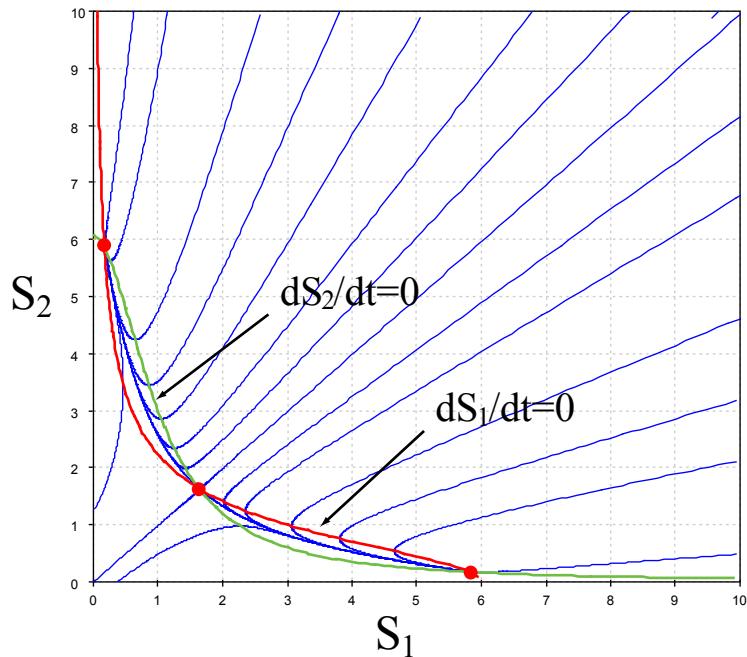


Figure 22.13 Phase portrait for a toggle switch. The two nullclines are superimposed on the phase plot and labelled with arrows. Diagram generated by PPlane <http://math.rice.edu/~dfield/dpp.html>. See Figure 22.12 for parameter values.

23

Oscillators

“No single thing abides; but all things flow. ”

– Titus Lucretius Carus (c.99-55 BCE)

23.1 Introduction

The study of oscillatory systems in biochemistry has a long history dating back to at least to the 1950’s. Until recently however, there was little interest in the topic from mainstream molecular biology. In fact, one suspects that the concept of oscillatory behavior in cellular networks was considered more a curiosity than anything serious. With the advent of new measurement technologies, particularly high quality microscopy, and the ability to monitor specific protein levels using GFP and other fluorescence techniques a whole new world has opened up to many experimentalists. Of particular note is the recent discovery of oscillatory dynamics in the p53/Mdm2 couple [34, 22] and Nf- κ B [27] signaling; thus rather than being a mere curiosity, oscillatory behavior is in fact an important, though largely unexplained, phenomenon in cells.

Basic Oscillatory Designs

There are two basic kinds of oscillatory designs, one based on negative feedback and a second based on a combination of negative and positive feedback. Both kinds of oscillatory design have been found in biological systems. An excellent review of these oscillators and specific biological examples can be found in [64, 14]. A more technical discussion can be found in [63, 62].

23.2 Negative Feedback Oscillator

Negative feedback oscillators are the simplest kind to understand and probably one of the first to be studied theoretically [25]. Savageau [55] in his book provides a detailed analysis and summary of the properties of feedback oscillators. Figure 23.1 shows a simple example of a system with a negative feedback loop. We can analyze this system by deriving the characteristic equations (the denominator of the frequency response) and constructing a Routh-Hurwitz table. Using this technique it can be easily shown that a pathway with only two intermediates in the feedback loop *cannot* oscillate. In general a two variable system with a negative feedback is stable under all parameter regimes. Once a third variable has been added the situation changes and the pathway shown in Figure 23.1, which has three variables, can admit oscillatory behavior. Figure 23.1 shows a metabolic network but feedback oscillators can also be made from genetic components as shown in Figure 23.2.

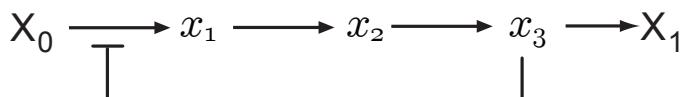


Figure 23.1 Simple negative feedback model with three variables, S_1 , S_2 and S_3 . This network can oscillate.

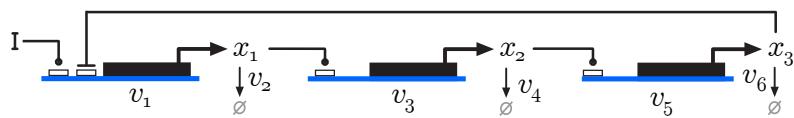


Figure 23.2 Simple negative feedback model constructed from three genes. This network can oscillate.

A critical factor that determines the onset of oscillations, apart from the number of variables is the strength of the feedback. Savageau [55] showed that if the substrate elasticities were equal (*e.g.* all first-order kinetics) then the ratio of the feedback elasticity (ε_{inh}) to the output elasticity (ε_{sub} , ε_3^4) determined the onset of oscillations (Table 23.1). Table 23.1 shows that as the pathway becomes longer less feedback inhibition is required to destabilize the pathway. This highlights the other factor that contributes to instability, the delay in routing the signal around the network. All feedback oscillators require some device to provide amplification of a signal combined with a suitable time delay so that the signal response can go out of phase. In metabolic pathways amplification is often provided by a cooperative enzyme while the delay is provided by the intermediate steps in the pathway. In signalling pathways, amplification can be generated by covalent modification cycles. Amplification can also be provided by another means. The criterion for instability is the ratio of the inhibition elasticity to the substrate elasticity. If the output reaction of the pathway is governed by a saturable enzyme

Length of Pathway	Instability Threshold $-\varepsilon_{inh}/\varepsilon_{sub}$
1	stable
2	stable
3	8.0
4	4.0
5	2.9
6	2.4
7	2.1
:	:
∞	1.0

Table 23.1 Relationship between the pathway length and the degree of feedback inhibition on the threshold for stability. ε_{inh} is the elasticity of the feedback inhibition and ε_{sub} is the elasticity of the distal step with respect to the signal.

then it is possible to have ε_{sub} less than unity. This means that it is possible to trade cooperativity at the inhibition site with saturation at the output reaction. The modified Goodwin model of Bliss [3] illustrates the model with no cooperativity at the inhibition site but with some saturation at the output reaction by using a simple Michaelis-Menten rate law.

A second property uncovered by BCT is that stability is enhanced if the kinetic parameters of the participating reactions are widely separated, that is a mixture of ‘fast’ and ‘slow’ reactions. The presence of ‘fast’ reactions effectively shortens the pathway and thus it requires higher feedback strength to destabilize the pathway since the delay is now less.

All feedback oscillators require some device to provide amplification of a signal combined with a suitable time delay so that the signal response can go out of phase. In metabolic pathways amplification is often provided by a cooperative enzyme while the delay is provided by the intermediate steps in the pathway. In signalling pathways, amplification can be generated by covalent modification cycles [24, 33] while in genetic networks, amplification can be provided by cooperative binding of the transcription factor to the promoter site. The criterion for instability is the ratio of the inhibition elasticity to the substrate elasticity. If the output reaction of the pathway is governed by a saturable enzyme then it is possible to have ε_{sub} less than unity. This means that it is possible to trade cooperativity at the inhibition site with saturation at the output reaction. The modified phase-shift Goodwin model of Bliss [3] illustrates a model with no cooperativity at the inhibition site but with a degree of saturation at the output reaction by using a simple Michaelis-Menten rate law.

One of the characteristics of negative feedback oscillators is that they tend to generate smooth oscil-

lations (Figure 23.3) and in man-made devices they are often used to generate simple trigonometric functions.

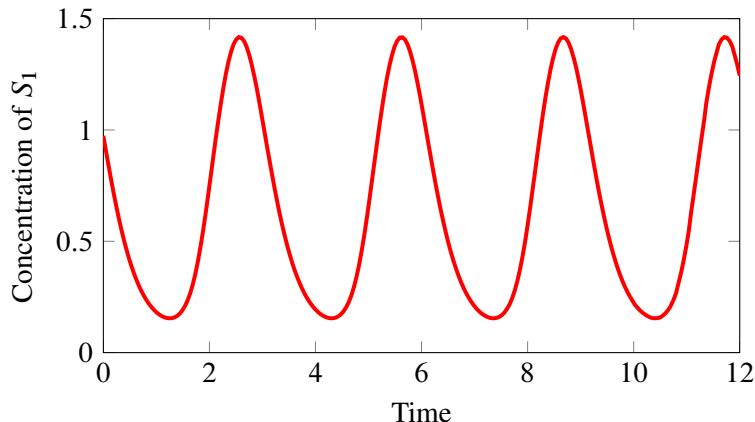


Figure 23.3 Plot of S_3 versus time for the model shown in Figure 23.1. Note that the profile of the oscillation is relatively smooth. See Table 23.2 for Python listing.

```
import tellurium as te

r = te.loada('''
J0: $X0 -> S1; VM1*(X0-S1/Keq1)/(1+X0+S1+pow(S4,h));
J1: S1 -> S2; (10*S1-2*S2)/(1+S1+S2);
J2: S2 -> S3; (10*S2-2*S3)/(1+S2+S3);
J3: S3 -> S4; (10*S3-2*S4)/(1+S3+S4);
J4: S4 -> $X1; J4_Vm4*S4/(KS4+S4);

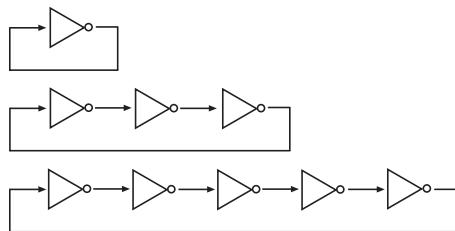
X0 = 10;           X1 = 0;
S1 = 0.973182;   S2 = 1.15274;
S3 = 1.22721;    S4 = 1.5635;

VM1 = 10; Keq1 = 10;
h = 10;   Vm4 = 2.5;
KS4 = 0.5;
''')

m = r.simulate (0, 12, 250, ["time", "S1"]);
te.plotwithLegend (r, m)
```

Listing 23.1 Python script used to generate Figure 23.3.

A related oscillator that operates in a similar way to the feedback oscillator is the ring oscillator (See Figure 23.4). This device is composed of an odd number of signal inverters connected into a closed chain. Instability requires sufficient amplification between each inverter so that the signal strength is maintained. A ring oscillator has been implemented experimentally in *E. coli* [13] where it was termed a repressilator. Ring oscillators with an even number of inverters can be used to form memory units or toggle switches. The even number of units means that the signal latches to either on or off, the final state depending on the initial conditions. Toggle circuits have also been implemented experimentally in *E. coli* [18].



causes the species to discharge. Once the species has discharged the bistable switch returns to the original state and the sequence begins again. Figure 23.5 shows a simple genetic relaxation oscillator with the bistable switch on the left (formed from P_1) and the negative feedback originating from the gene product P_2 .

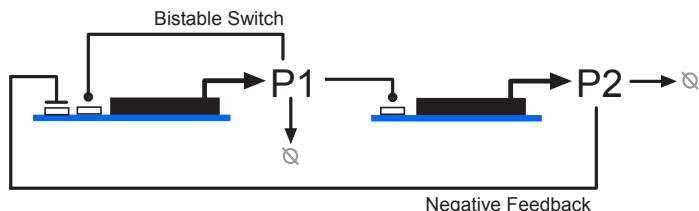


Figure 23.5 Relaxation oscillator made from genetic components.

One of the characteristics of a relaxation oscillator is the ‘spiky’ appearance of the oscillations. This is due to the rapid switching of the bistable circuit which is much faster compared to the operation of the negative feedback. Man-made devices that utilize relaxation oscillators are commonly used to generate saw-tooth signals. Figure 23.6 illustrates a plot from a hypothetical relaxation oscillator published by Tyson’s group [64].

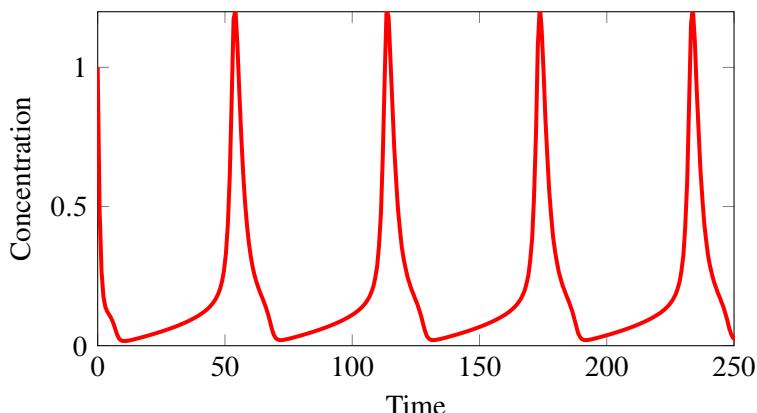


Figure 23.6 Typical spiky appearance of oscillatory behavior from a relaxation oscillator, from Tyson et. al, 2003, model 2(c). See Table 23.2 for Jarnac listing.

```

import tellurium as te

r = te.loada('''
J1: $src -> X;           k1*S;
  
```

```

J2:    X -> R;      (kop + ko*EP)*X;
J3:    R -> $waste; k2*R;
J4:    E -> EP;     Vmax_1*R*E/(Km_1 + E);
J5:    EP -> E;     Vmax_2*EP/(Km_2 + EP);

src = 0;      kop = 0.01;
ko = 0.4;    k1 = 1;
k2 = 1;      R = 1;
EP = 1;      S = 0.2;

Km_1 = 0.05; Km_2 = 0.05;
Vmax_2 = 0.3; Vmax_1 = 1;
'')
m = r.simulate (0, 300, 400, ["time", "R"]);
te.plotWithLegend (r, m)

```

Listing 23.2 Python script used to generate Figure 23.6. From Tyson et al [64], Substrate-depletion oscillator 2c

23.4 Oscillator Classification

As previously discussed, oscillators fall into two broad categories, feedback oscillators and relaxation oscillators. Within the relaxation oscillation group, some authors [64] have proposed to divide this group into two and possibly three additional subgroups, these include; substrate-depletion, activator-inhibitor and toggle based relaxation oscillators. The grouping is based on two variable oscillators and a comparison of the sign patterns in the Jacobian matrix. Although toggle based relaxation oscillations have the same Jacobian sign pattern as substrate-depletion based oscillations, the bistability is implemented differently.

Figure 23.7 shows examples of six different oscillators together with their classification and stylized forms.

Even though each mechanistic form (first column) in Figure 23.7 looks different, the stylized forms (second column) fall, into one of three types. The stylized forms reflect the structure of the Jacobian for each model. Only a limited number of sign patterns in the Jacobian can yield oscillators [26]. Using evolutionary algorithms [11, 43] many hundreds of related mechanisms can be generated, see the model repository at www.sys-bio.org for a large range of examples. Although many of these evolved oscillators look quite different, each one can be classified in a only a few basic configurations.

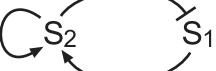
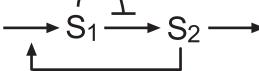
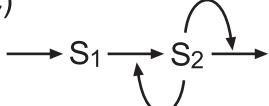
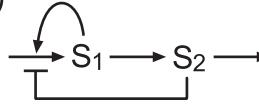
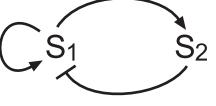
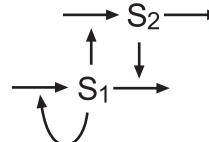
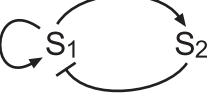
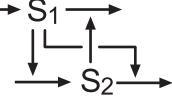
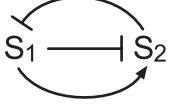
Mechanism	Stylized Form	Type
a) 		SD
b) 		SD
c) 		SD
d) 		AI
e) 		AI
f) 		SD/T

Figure 23.7 Classification of Relaxation Oscillators into substrate-depletion, activator-inhibitor and toggle based. Note that although the mechanistic networks are quite variable, the underlying operation is the same as shown in the stylized column. Type codes: SD = Substrate-Depletion; AI = Activator-Inhibitor; SD/T = Substrate-Depletion/Toggle. The stylized form is generated by computing the Jacobian matrix for each network. Elements in the Jacobian indicate how each species influences changes in another. Model a) corresponds to model c) in Figure 2 of [64] and model e) to model b) in Figure 2 of [64].

Appendices

A

Symbols

A.1 List Of Symbols

x	Species concentration or other state variable
y	Output variable
p	Parameter of a system
v_i	Reaction velocity
s	Laplace complex variable, $\sigma + j\omega$
$H(s)$	Transfer Function
ε_j^i	Scaled elasticity coefficient
\mathcal{E}_j^i	Unscaled elasticity coefficient
$\mathcal{L}()$	Laplace transform
A	State matrix or Jacobian
B	Control matrix
C	Output matrix
D	Feed-forward matrix
N	Stoichiometry matrix
k_i	Reaction rate constant

$\delta(t)$	Delta function
$u(t)$	Unit step function
ω	Frequency
θ	Phase angle

B

Numerical Methods

B.1 Introduction

In this chapter we will look at ways to solve the differential equations that we generate when we build a model of a system. For example, consider the simplest possible model, the first-order irreversible degradation of a molecular species, S into product P :



The differential equation for this simple reaction is given by the familiar form:

$$\frac{dS}{dt} = -k_1 S \quad (\text{B.1})$$

Our aim is to solve this equation so that we can describe how S changes in time. There are at least two ways to do this, we can either solve the equation mathematically or use a computer to obtain a numerical solution. Mathematically we can either solve the system using traditional methods from calculus or we can use the Laplace transform method.

To use the traditional method we move the dependent variables onto one side and in the independent variables onto the other, this is called the separation of variables. In the above equation one can easily do this by dividing both sides by S to give:

$$\frac{dS}{dt} \frac{1}{S} = -k_1 \quad (\text{B.2})$$

In differential calculus, the derivative of $\ln y$ with respect to t is

$$\frac{d \ln y}{dt} = \frac{dy}{dt} \frac{1}{y}$$

This means we can rewrite equation B.2 in the following form:

$$\frac{d \ln S}{dt} = -k_1$$

We now integrate both sides with respect to dt , that is:

$$\int \ln S dt = -k_1 \int dt$$

$$\ln S = -k_1 t + C$$

where C is the constant of integration. If we assume that at $t = 0$, $S = S_o$, then $\ln S_o = C$. Substituting this result into the solution gives:

$$\ln S = -k_1 t + \ln S_o$$

$$\ln \left(\frac{S}{S_o} \right) = -k_1 t$$

Taking anti-natural logarithms on both sides and multiply both sides by S_o gives:

$$S = S_o e^{-k_1 t}$$

For simple systems such as this, it is possible to obtain analytical solutions but very rapidly one is confronted with the fact that for 99.99% of problems, no mathematical solution exists. In such cases, we must carry out numerical simulations.

B.2 Numerical Solutions

In the last section we saw how it was possible to solve a differential equation mathematically. If the system of differential equation is linear there are systematic methods for deriving a solution. Most of the problems we encounter in biology however are non-linear and for such cases mathematical solutions rarely exist. Because of this, computer simulation is often used instead. Since the 1960s,

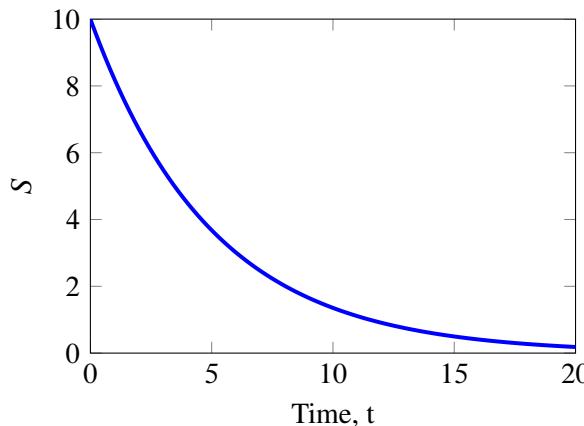


Figure B.1 Exponential decay from the equation: $S = S_o e^{k_1 t}$ where $S_o = 10$, $k_1 = 0.2$.

almost all simulations have been carried out using digital computers. Before the advent of digital computers, analog computer were frequently used where an analog of the system was built using either mechanical or more commonly, electrical analogs of concentrations. Here we will focus on methods used on digital computers.

The general approach to obtaining a solution by computer is as follows:

1. Construct the set of ordinary differential equations, with one differential equation for every molecular species in the model.
2. Assign values to all the various kinetic constants and boundary species.
3. Initialize all floating molecular species to their starting concentrations.
4. Apply an integration algorithm to the set of differential equations.
5. If required, compute the fluxes from the computed species concentrations
6. Plot the results of the simulation.

Step four is obviously the key to the procedure and there exist a great variety of integration algorithms. We will describe three common approaches to give a flavor of how they work. Other than for educational purposes (which is significant), it is rare now for a modeler to write their own integration computer code because many libraries and applications now exist that incorporate excellent integration methods.

An integration algorithm approximates the behavior of what is, strictly speaking, a continuous system, on a digital computer. Since digital computers can only operate in discrete time, the algorithms convert the continuous system into a discrete time system. This is the reason why digital computers can only generate approximations. In practice a particular discrete step size, h , is chosen, and solution points are generated at the discrete points up to some upper limit. As we will discover, the approximation generated by the simplest methods is dependent on the size of the step size and in general the smaller the step size the more accurate the solution. However since computers can only represent numbers to a given precision (usually 15 digits on modern computers), it is not possible to continually reduce the step size in the hope of increasing the accuracy of the solution. For one thing, the algorithm will soon reach the limits of the precision of the computer and secondly, the smaller the step size the longer it will take to compute the solution. There is therefore often a tradeoff that is made between accuracy and computation time.

Let us first consider the simplest method, the Euler method, where the tradeoff between accuracy and computer time can be clearly seen.

B.2.1 Euler Method

The Euler method is the simplest possible way to solve a set of ordinary differential equations. The idea is very simple. Consider the following differential equation that describes the degradation rate of a species, S :

$$\frac{dS}{dt} = -k_1 S$$

The Euler method uses the rate of change of S to predict the concentration at some future point in time. Figure B.2 describes the method in detail. At time t_1 , the rate of change in S is computed from the differential equation using the known concentration of S at t_1 . The rate of change is used to compute the change in S over a time interval h , using the relation, $h \frac{dS}{dt}$. The current time, t_1 is incremented by the time step, h and the procedure repeated again, this time starting at t_2 . The method can be summarized by the following two equations which represent one step in an iteration that repeats until the final time point is reached:

$$y(t + h) = y(t) + h \frac{dy(t)}{dt}$$

$$t_{n+1} = t_n + h \quad (B.3)$$

Figure B.2 also highlights a problem with the Euler method. At every iteration, there will be an error between the change in S we predict and what the change in S should have been. This error is called the **truncation error** and will accumulate at each iteration. If the step size is too large, this error can make the method numerically unstable resulting in wild swings in the solution.

Figure B.2 also suggests that the larger the step size the larger the truncation error. This would seem to suggest that the smaller the step size the more accurate the solution will be. This is indeed the

case, up to a point. If the step size becomes too small there is the risk that roundoff error which will propagate at each step into the solution. In addition, if the step size is too small it will require a large number of iterations to simulate even a small time period. The final choice for the step size is therefore a compromise between accuracy and effort. A theoretical analysis of error propagation in the Euler method indicates that the error accumulated over the entire integration period (called the **global error**) is proportional to the step size. Therefore halving the step size will reduce the global error by half. This means that to achieve even modest accuracy, small step sizes are necessary. As a result, the method is rarely used in practice. The advantage of the Euler method is that it is very easy to implement in computer code or even on a spreadsheet.

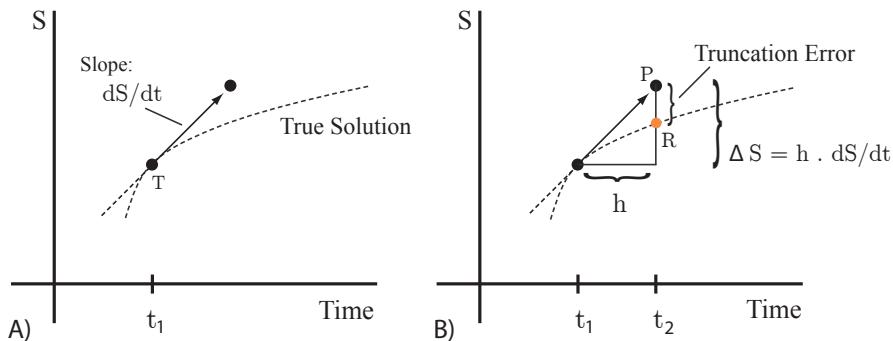


Figure B.2 Euler Method. Starting at t_1 , the slope dS/dt at t_1 is computed (Panel A). The slope is used to project forward to the next solution in time step, h , to t_2 (Panel B). The new solution at t_2 is indicated by P . However the solution is given by point R , located on the solution curve at t_2 . Reducing the step size h will reduce the error between the exact and the projected solution, but will simultaneously increase the number of slope projections necessary to compute the solution over a given time period.

The Euler method can also be used to solve systems of differential equations. In this case all the rates of change are computed first followed by the application of the Euler equation B.3. As in all numerical integration methods, the computation must start with an initial condition for the state variables at time zero. The algorithm is described using pseudo-code in Algorithm 1.

Example 2

Solve the decay differential equation B.1 using the Euler method. Assume $k_1 = 0.2$ and the concentration of S_o and P are time = 0 is 10 and 0 respectively. Assume a step size, h , of 0.4. Form a table of four columns, write out the solution to two three decimal places. The 4th column should include the exact solution for comparison.

n = Number of state variables

$y_i = i^{\text{th}}$ variable

Set timeEnd

currentTime = 0

h = stepSize

Initialize all y_i at currentTime

while currentTime < timeEnd **do**

for $i = 1$ to n **do**

$dy_i = f_i(y)$

for $i = 1$ to n **do**

$y_i(t + h) = y_i(t) + h \ dy_i$

 currentTime = currentTime + h

end while

Algorithm 1: Euler Integration Method, $f_i(y)$ represents the i^{th} differential equation from the system of ordinary differential equations.

Time	Solution (S)	dS/dt	Exact Solution
0	10	2	10
0.4	9.2	1.84	9.23
0.8	8.464	1.6928	8.52
1.2	7.787	0.01	7.87
...			

Table B.1 Solution Table

Figure B.3 shows the effect of different steps sizes on the Euler method. Four cases are shown, in the worse case the solution is unbounded and the computer will eventually crash with an overflow error. The second case is where the result is bounded but the solution bears no resemblance at all to the actual solution. The third case shows that the solution is beginning to resemble the actual solution but irregularities appear near the beginning of the integration. The final case shows the actual solution generated from a specialized integrator.

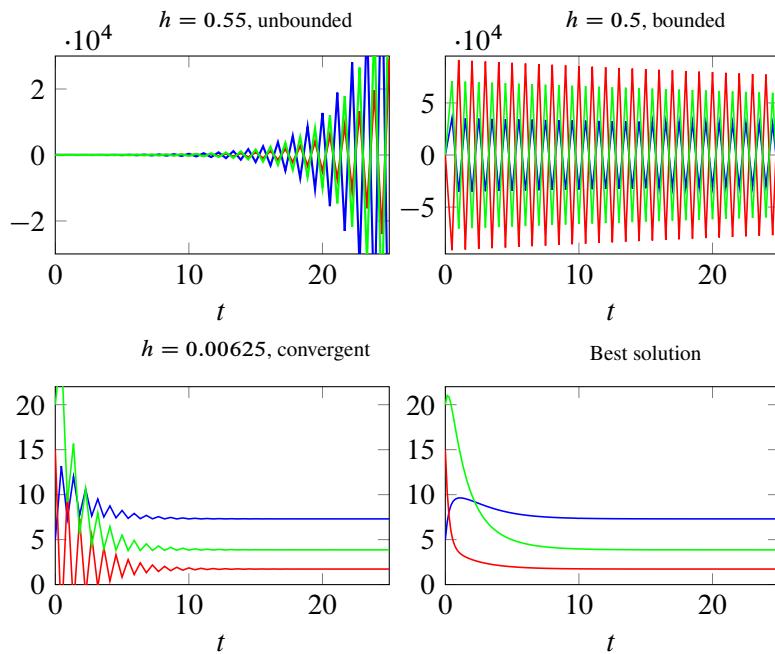


Figure B.3 Effect of different step sizes on the Euler method using a simple linear chain of reactions where each reaction follows reversible mass-action kinetics: $X_o \xrightleftharpoons[k_2]{k_1} S_1, S_1 \xrightleftharpoons[k_4]{k_3} S_2, S_2 \xrightleftharpoons[k_6]{k_5} S_3, S_3 \xrightleftharpoons[k_7]{k_7} X_1$ where $k_1 = 0.45, k_2 = 0.23, k_3 = 0.67, k_4 = 1.2, k_5 = 2.3, k_6 = 0.3, k_7 = 0.73, X_o = 10, X_1 = 0, S_1 = 5, S_2 = 15, S_3 = 20$.

B.2.2 Modified Euler or Heun Method

As indicated in the last section, the Euler method, though simple to implement, tends not to be used in practice because it requires small step sizes to achieve reasonable accuracy. In addition the small step size makes the Euler method computationally slow. A simple modification however can be made to the Euler method to significantly improve its performance. This approach can be found under a number of headings, including the modified Euler method, the Heun or the improved Euler method.

The modification involves improving the estimate of the slope by averaging two derivatives, one at the initial point and another at the end point. In order to calculate the derivative at the end point, the first derivative must be used to predict the end point which is then corrected by using the averaged slope (Figure B.4). This method is a very simple example of a predictor-corrector method. The method can be summarized by the following equations:

$$\text{Predictor: } y(t + h) = y(t) + h \frac{dy(t)}{dt} \quad (\text{B.4})$$

$$\text{Corrector: } y(t + h) = y(t) + \frac{h}{2} \left(\frac{dy(t)}{dt} + \frac{dy(t + h)}{dt} \right) \quad (\text{B.5})$$

$$t_{n+1} = t_n + h \quad (\text{B.6})$$

Figure B.4 describes the Heun method graphically.

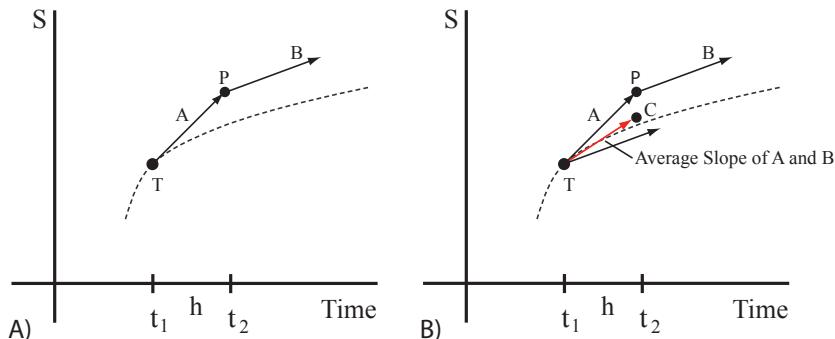


Figure B.4 Heun Method. Starting at t_1 , the slope A at T is computed. The slope is used to predict the solution at point P using the Euler method. From point P the new slope, B is computed (Panel A). Slopes A and B are now averaged to form a new slope C (Panel B). The averaged slope is used to compute the final prediction.

A theoretical analysis of error propagation in the Heun method shows that it is a second order method, that is if the step size is reduced by a factor of 2, the global error reduced by a factor of 4. However,

to achieve this improvement, two evaluations of the derivatives is required per iteration, compared to only one for the Euler method. Like the Euler method the Heun method is also quite easy to implement.

```

 $n = \text{Number of state variables}$ 
 $y_i = i^{\text{th}}$  variable
Set timeEnd
currentTime = 0
 $h = \text{stepSize}$ 
Initialize all  $y_i$  at currentTime

while currentTime < timeEnd do
  for  $i = 1$  to  $n$  do
     $a_i = f_i(y)$ 

    for  $i = 1$  to  $n$  do
       $b_i = f_i(y + h a)$ 

    for  $i = 1$  to  $n$  do
       $y_i(t + h) = y_i(t) + \frac{h}{2} (a_i + b_i)$ 

  currentTime = currentTime +  $h$ 
end while
```

Algorithm 2: Heun Integration Method. $f_i(y)$ is the i^{th} ordinary differential equation

B.2.3 Runge-Kutta

The Heun method described in the previous section is sometimes called the RK2 method where RK2 stands for 2nd order Runge-Kutta method. The Runge-Kutta methods are a family of methods developed around the 1900s by the German mathematicians, Runge and Kutta. In addition to the 2nd order Heun method there are also 3rd, 4th and even 5th order Runge-Kutta methods. For hand coded numerical methods, the 4th order Runge-Kutta algorithm (often called RK4) is probably the most popular among modelers. The algorithm is a little more complicated in that it involves the evaluation and weighted averaging of four slopes. In terms of global error, however, RK4 is considerably better than Euler or the Heun method and has a global error of the order of four. This means that halving the step size will reduce the global error by a factor of 1/16. Another way of looking at this is that the step size can be increased 16 fold over the Euler method and still have the same global error. The method can be summarized by the following equations which have been simplified by removing the dependence on time:

$$\begin{aligned}
 k_1 &= h f(y_n) \\
 k_2 &= h f\left(y_n + \frac{k_1}{2}\right) \\
 k_3 &= h f\left(y_n + \frac{k_2}{2}\right) \\
 k_4 &= h f\left(y_n + k_3\right) \\
 y(t+h) &= y(t) + \frac{1}{6} \left(k_1 + 2k_2 + 2k_3 + k_4 \right) \\
 t_{n+1} &= t_n + h
 \end{aligned}$$

Figure B.5 shows a comparison of the three methods, Euler, Heun and RK4 in solving the Van der Pol equations. The Van der Pol equations are a classic problem set that is often used when comparing numerical methods. The equations model an oscillating system, inspired originally from modeling vacuum tubes but also later formed the basis for developments in modeling action potentials in neurons. The Figure shows that the Heun and RK4 methods are very similar, at least for the Van der Pol equations, though this is not always be the case. For this particular model the solution generated by the RK4 method is very similar to the best possible solution that can be obtained by numerical solution.

B.2.4 Variable Step Size Methods

In the previous discussion of numerical methods for solving differential equations, the step size, h , was assumed to be fixed. This makes implementation quite straight forward but also make the methods inefficient. For example, if the solution is at a point where it changes very little then the method could probably increase the step size without loosing accuracy while at the same time achieve a considerable speedup in the time it takes to generate the solution. Likewise if at a certain point in the integration the solution starts to change rapidly, it would be prudent to lower the step size to increase accuracy. Such strategies are implemented in the **variable step size methods**.

The approach used to automatically adjust the steps size can vary from quite simple approaches to very sophisticated methods. The simplest approach is to carry out two integration trials, one at a step size of h and another trial using two steps size but at $h/2$. The software now compares the solution generated by the two trials. If the solutions are significantly different then the step size must be reduced. If the solutions are about the same then it might be possible to increase the step size. These tests are repeatedly carried out, adjusting the step size as necessary as the integration proceeds. This simple variable step size approach can be easily incorporated into some of the simpler algorithms particularly the fourth order Runge-Kutta where it is called the variable step-size Runge-

n = Number of state variables
 y_i = i^{th} variable
 $\text{timeEnd} = 10$
 $\text{currentTime} = 0$
 h = stepSize
 Initialize all y_i at currentTime

```

while currentTime < timeEnd do
  for  $i = 1$  to  $n$  do
     $k_{1i} = hf(y_i)$ 

  for  $i = 1$  to  $n$  do
     $k_{2i} = hf(y_i + k_{1i}/2)$ 

  for  $i = 1$  to  $n$  do
     $k_{3i} = hf(y_i + k_{2i}/2)$ 

  for  $i = 1$  to  $n$  do
     $k_{4i} = hf(y_i + k_{3i})$ 

  for  $i = 1$  to  $n$  do
     $y_i(t + h) = y_i(t) + \frac{h}{6} (k_{1i} + 2k_{2i} + 2k_{3i} + k_{4i})$ 

  currentTime = currentTime +  $h$ 
end while
  
```

Algorithm 3: 4th Order Runge-Kutta Integration Method

Kutta. Another approach to adjusting the step size is called the Dormand-Prince method. This method carries out two trials based on the fourth and fifth order Runge-Kutta. Any difference between the two trials is used to adjust the step size. Matlab's ode45 implements the Dormand-Prince method. Similar methods to Dormand-Prince include the Fehlberg and more recently the Cash-Karp method.

Many of these simple adjustable step size solvers are quite effective. Sometimes they can be slow however especially for the kinds of problem we find in biochemical models. In particular there is a class of problem called stiff problems which generally plagues the biochemical modeling community. Stiff models require highly specialized solvers which have been developed in the last four decades.

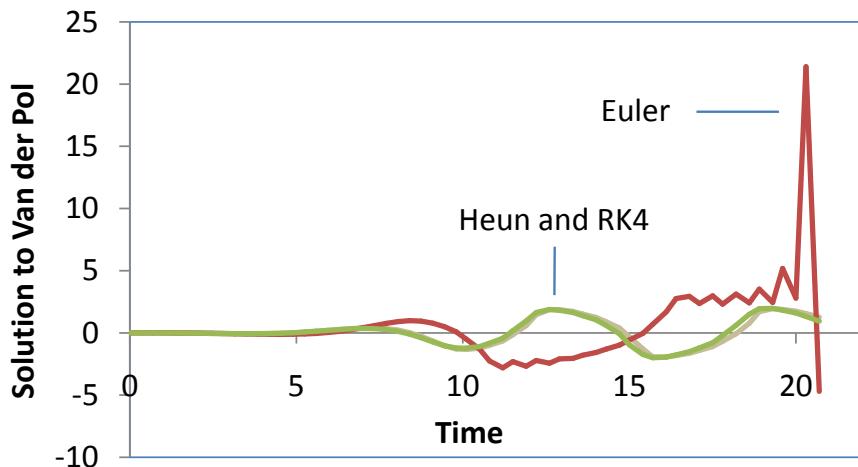


Figure B.5 Comparison of Euler, Heun and RK4 numerical methods at integrating the Van der Pol dynamic system: $dy_1/dt = y_2$ and $dy_2/dt = -y_1 + (1 - y_1 y_1)y_2$. The plots show the evolution of y_1 in time. The RK4 solution is almost indistinguishable from solutions generated by much more sophisticated integrators. Step size was set to 0.35.

B.2.5 Stiff Models

Many differential equations we encounter in biochemical models are so-called **stiff** systems. The word stiff apparently comes from earlier studies on spring and mass systems where the springs had large spring constants and therefore difficult to stretch. A stiff system is often associated with widely different time scales in a system, for example when the rate constants are widely different in a biochemical model. Such systems may have molecular species whose decay rates are very fast compared to other components. This means that the step size has to be very small to accommodate the fast processes even though the rest of the system could be accurately solved using a much larger step size. The overall result is the need for very small steps sizes and therefore significant computational cost and the possibility of roundoff error which will tend to be amplified by the large time constants in the fast system. The net result are solutions which bear no resemblance to the true solution.

Most modern simulators will employ specific stiff algorithms for solving stiff differential equations. Of particular importance is the sundials suite and odepak. Sundials includes a number of very useful, well written and documented solvers. In particular the CVODE solver is very well suited to finding solutions to stiff differential equations. As a result sundials is widely used in the biochemical modeling community (for example by Jarnac and roadRunner). Before the advent of sundials, the main workhorse for solving stiff systems was the suite of routines in odepak. Of particular importance was LSODA which in the 1990s was very popular and is still a valuable set of software (currently

used in COPASI). The original stiff differential equation solver was developed by Gear in the 1970s and is still used in Matlab in the form of `ode15s`.

Further Reading

Unfortunately there are very few reasonably priced books on numerical analysis. The two most popular expensive books are by Press and Burden and are included here for reference. Both books can be bought second-hand at reasonable prices and the content has not changed significantly between editions. One could even argue that the code examples in the latest edition of Press are actually worse than the previous editions.

1. Press, Teulolsky, Vetterling and Flannery (2007) Numerical Recipes. Cambridge University Press, 3rd Edition ISBN-10: 0521880688
2. Burden and Faires (2010) Numerical Analysis. Brooks Cole, 9th Edition. ISBN-10: 0538733519

For the budget conscious buyer I can highly recommend the Dover edition:

1. Dahlquist and Björck (2003) Numerical Methods. Dover Publications ISBN-10: 0486428079

Exercises

1. Implement the Euler method in your favorite computer language and use the code to solve the following two problems. Set initial conditions, $S_1 = 10$, $S_2 = 0$. Set the rate constants to $k_1 = 0.1$; $k_2 = 0.25$. Investigate the effect of different steps sizes, h , on the simulation results.
 - a) $dS_1/dt = -k_1 S_1$
 - b) $dS_1/dt = -k_1 S_1$; $dS_2/dt = k_1 S_1 - k_2 S_2$
2. The following model shows oscillations at a step of $h = 0.3$ when using the Euler method to solve the differential equations. By using simulation, show that these oscillations are in fact an artifact of the simulation.
3. Solve the following problem using the Euler method. What step size do you need to secure a reliable solution? Run the same problem on a specialist application such as Matlab or SBW. Compare the time take to carry out both simulations.

C

Modeling with Python

In this appendix a brief description of the Python programming language will be given plus a brief introduction to the Antimony reaction network format and libRoadRunner.

Python Python is an easy to learn general purpose interactive programming language. It has similar usability characteristics to Matlab or Basic. As such it is a good language to use for doing pathway simulations and is easily learned by new users. In recent years Python has also become more widely used as a general purpose scientific programming language and now supports many useful libraries and tools for modelers. All the scripts we provide in this book are written in Python.

Antimony SBML has become a de facto standard for exchanging models of biological pathways. Any tool we use should therefore be able to support SBML. However SBML is a computer readable language and it is not easy for humans to read or write SBML. Instead more human readable formats have been developed. In this text book we will be using the Antimony pathway description language. Models can be described in Antimony then converted to SBML or vice versa.

libRoadRunner To support SBML from within Python we developed a C/C++ simulation library called libRoadRunner that can read and run models based on SBML. In order to use libRoadRunner within Python, we also provide a Python interface that makes it easy to carry out simulations with Python.

Spyder Integration of the various tools including Python is achieved by using spyder2 (<https://>

code.google.com/p/spyderlib/). Spyder2 offers a Matlab like experience in a friendly, cross-platform environment.

C.1 Introduction to Python

One great advantage of the Python language is that it runs on many computer platforms, most notably Windows, Mac and Linux and is freely downloadable from the Python web site. To execute Python code we will need access to what is often referred to as a Python IDE (Integrated Development Environment). In the Python world there are many IDEs to choose from, ranging from very simple consoles to sophisticated development systems that includes documentation, debuggers and other visual aids. In this book we use the cross-platform IDE called spyder2 (<https://code.google.com/p/spyderlib/>).

The best way to learn Python is to download a copy and start using it. We have prepared installers that install all the relevant components you need, these can be found at tellurium.analogmachine.org. The Tellurium distribution includes some additional helper routines which can make life easier for new users. The Tellurium version can be downloaded for Mac and Windows computers. We will use the Windows version here. To download the installer go to the web site tellurium.analogmachine.org, and click on the first link you see called *Download Windows version here*. Run the installer and follow the instructions.

Once Tellurium is installed go to the start menu, find Tellurium and select the application call Tellurium spyder. If successful you should see something like the screen shot in Figure C.1 but without the plotting window. The screen-shot shows three important elements, on the left we see an editor, this is where models can be edited. On the lower right is the Python console where Python commands can be entered. At the top right we show plotting window that illustrates some output from a simulation. For those familiar with IPython, the latest version of spyder2 supports the IPython console directly.

Once you have started the Tellurium IDE, let us focus on the Python console at the bottom right of the application. A screen-shot of the console is shown in Figure C.2.

The `>>>` symbol marks the place where you can type commands. The following examples are based on Python 2.7. To add two numbers, say $2 + 5$, we would type the following:

```
>>> print 2 + 5
7
>>>
```

Listing C.1 Simple Arithmetic

Just like Matlab or Basic we can assign values to variables and use those variables in other calculations:

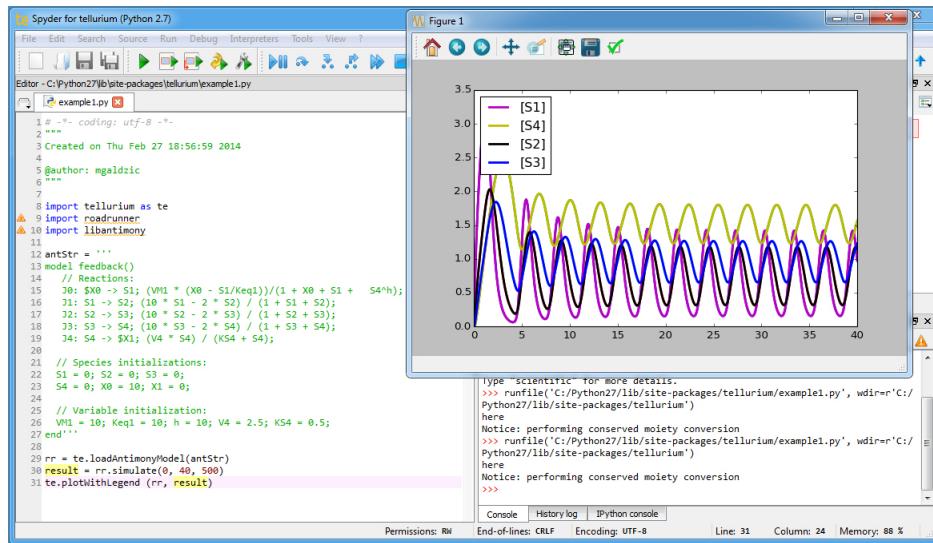


Figure C.1 Screen-shot of Tellurium, showing editor on the left, Python console bottom right and plotting window top-right.

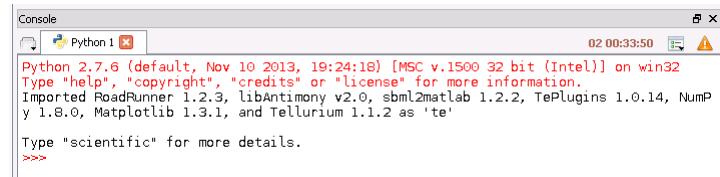


Figure C.2 Screen-shot of Tellurium, focusing on the Python console.

```
>>> a = 2
>>> b = 5
>>> c = a + b
>>> print c
7
>>>
```

Listing C.2 Assigning values to variables

The types of values we can assign to variables include values such as integers, floating point numbers, Booleans (True or False), strings and complex numbers.

```
>>> a = 2
```

```
>>> b = 3.1415
>>> c = False
>>> d = "Hello Python"
>>> e = 3 + 6j
>>>
```

Listing C.3 Different kinds of values

Many functions in Python are accessible via modules. For example to compute the sin of a number we can't simply type `sin (30)`. Instead we must first load the math module. We can then call the `sin` function:

```
>>> import math
>>> print sin (3.1415)
9.265358966049026e-05
>>>
```

Listing C.4 Importing modules (libraries) into Python

In Tellurium we preload some libraries including the math library.

Repeating Calculations

One of the commonest operations we do in computer programming is iteration. We can illustrate this with a simple example that loops ten times, each time printing out the loop index. This example will allow us to introduce the IDE editor. The editor is the panel on the left side of the IDE. In the editor we can type Python code, for example we could type:

```
a = 4.0
b = 8.0
c = a/b
print "The answer is:", c
```

Listing C.5 Writing a simple program in the IDE editor

When we've finished typing this in the editor window, we can save our little program to a file (Select Menu: File/Save As...) and run the program by clicking on the green arrow in the tool bar of the IDE (Figure C.3). If we run this program we will see:

```
The answer is: 0.5
>>>
```

Listing C.6 Writing a simple program in the IDE editor**Figure C.3** Screen-shot of Tellurium, focusing on the Toolbar with the run button circled.

The IDE allows a user to have as many program files open at once, each program file is given its own tab so that it is easy to move from one to the other. This is useful if one is working on multiple models at the same time.

We will now use the editor to write the simple program that loops ten times, this is shown below:

```
for i in range (10):
    print i,
```

Listing C.7 A simple loop in python

This will generate the sequence:

```
0 1 2 3 4 5 6 7 8 9
```

Listing C.8 Result from simple loop program

There are a number of new concepts introduced in this small looping program. The first line contains the `for` keyword can be translated into literal English as “for all elements in a list, do this”. The list is generated from the `range()` function and in this case generates a list of 10 numbers starting at 0. `i` is the loop index and within the loop, `i` can be used in other calculations. In this case we will just print the value of `i` to the console. Each time the program loops it extracts the next value from the list and assigns it to `i`.

Two things are important to note in the print line. The first and most important is that the line has been indented four spaces. This isn’t just for aesthetic reasons but is actually functional. It tells Python what code should be executed *within* the loop. To elaborate we could add more lines to the loop, such as:

```
for i in range (10):
    a = i
    b = a*2
    print b,
```

```
print "Finished Loop"
```

Listing C.9 A simple loop illustrating multiple statements

In this example there are three indented lines, this means that these three lines will be executed within the loop. The last line which prints a message, is not indented and therefore will not be executed within the loop. This means we only see the message appear once right at the end. The output for this little program is shown below.

```
0 2 4 6 8 10 12 14 16 18 Finished Loop
```

Another important point worth noting is the use of the , after the loop print statement. The comma is used to suppress a newline. This is why the output appears on one line only. If we had left out the comma each print statement would be on its own line.

A final word about range(). Range takes up to three arguments. In the example we only gave one argument, 10. A single argument means create a list starting at zero, incrementing one for each item until the incremented value reaches 10. A second argument such as range (5, 10) means start the list at 5 rather than zero. Finally, a third argument can be used to specify the increment size. For example the command range (1, 10, 2) will yield the list:

```
[1, 3, 5, 7, 9]
```

The easiest way to try out the various options in range is to type them at the console to get immediate feedback.

The use of variables, printing results, importing libraries and looping are probably the minimum concepts one needs to start using Python. However there are a huge range of resources online to help learn Python. Of particular interest is the codecademy web site (<http://www.codecademy.com/>). This site offers an interactive means to learn Python (including other programming languages).

C.2 Describing Reaction Networks using Antimony

The code shown in the panel below illustrates the description of a very simple model using the Antimony syntax followed by two lines of Python that uses libRoadRunner to run a simulation of the model. In this section we will briefly describe the Antimony syntax. A more detailed description of Antimony can be found at <http://antimony.sourceforge.net/index.html>.

```
import tellurium as te  
  
rr = te.loada ('''
```

```
S1 -> S2; k1*S1;  
S1 = 10; k1 = 0.1  
''')  
  
rr.simulate (0, 50, 100)  
rr.plot()
```

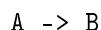
Listing C.10 Simple model Antimony and simulated using libRoadRunner

The main purpose of Antimony is to make it straight forward to specify complex reaction networks using a familiar chemical reaction notation.

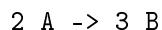
A chemical reaction can be an enzyme catalyzed reaction, a binding reaction, a phosphorylation, a gene expressing a protein or any chemical process that results in the conversion of one of more species (reactants) to a set of one or more other species (products). In Antimony, reactions are described using the notation:



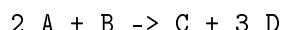
where the reactants are on the left side and products on the right side. The left and right are separated by the \rightarrow symbol. For example:



describes the conversion of reactant A into product B. In this case one molecule of A is converted to one molecule of B. The following example shows non-unity stoichiometry:



which means that two molecules of A react to form three molecules of B. Bimolecular and other combinations can be specified using the $+$ symbol, that is:



tells us that two molecules of A combine with one molecule of B to form one molecule of C and three molecules of D.

To specify species that do not change in time (boundary species), add a dollar character in front of the name, for example:



means that during a simulation A is fixed.

Reactions can be named using the syntax J1:, for example:



means the reaction has a name, J1. Named reaction are useful if you want to refer to the flux of the reaction; kinetic rate laws come immediately after the reaction specification. If only the stoichiometry matrix is required, it is not necessary to enter a full kinetic law, a simple $\dots \rightarrow S1; v;$ is

sufficient. Here is an example of a reaction that is governed by a Michaelis-Menten rate law:

$A \rightarrow B; V_m * A / (K_m + A);$

Note the semicolons. Here is a more complex example involving multiple reactions:

```
MainFeed: $X0 -> S1; Vm*X0/(Km + X0);
TopBranch: S1 -> $X1; Vm1*S1/(Km1 + S1);
BottomBranch: S1 -> $X2; Vm2*S1/(Km2 + S1);
```

There is no need to pre-declare the species names shown in the reactions or the parameters in the kinetic rate laws. Strictly speaking, declaring the names of the floating species is optional, however this feature is for more advanced users who wish to define the order of rows that will appear in the stoichiometry matrix. For normal use there is no need to pre-declare the species names. To pre-declare parameters and variables see the example below:

```
const Xo, X1, X2; // Boundary species
var S1;           // Floating species

MainFeed: $X0 -> S1; Vm*X0/(Km + X0);
TopBranch: S1 -> $X1; Vm1*S1/(Km1 + S1);
BottomBranch: S1 -> $X2; Vm2*S1/(Km2 + S1);
```

We can load an Antimony model into libRoadRunner using the short-cut command `loada`. For example:

```
rr = te.loada '''
const Xo, X1, X2; // Boundary species
var S1;           // Floating species

MainFeed: $X0 -> S1; Vm*X0/(Km + X0);
TopBranch: S1 -> $X1; Vm1*S1/(Km1 + S1);
BottomBranch: S1 -> $X2; Vm2*S1/(Km2 + S1);
'''
```

To reference model properties and methods, the property or method must be proceeded with the `roadrunner` variable. e.g. `rr.S1 = 2.3;`

When loaded into libRoadRunner the model will be converted into a set of differential equations. For example, consider the following model:

```
$Xo -> S1; v1;
S1 -> S2; v2;
S2 -> $X1; v3;
```

will be converted into:

$$\frac{dS_1}{dt} = v_1 - v_2$$

$$\frac{dS_2}{dt} = v_2 - v_1$$

Note that there are no differential equations for X_o and X_1 . This is because they are fixed and do not change in time. If the reactions have non-unity stoichiometry, this is taken into account when the differential equations are derived.

C.2.1 Initialization of Model Values

To initialize the concentrations and parameters in a model we can add assignments after the network is declared, for example:

```
MainFeed:      $X0 -> S1;  Vm*X0/(Km + X0);
TopBranch:    S1 -> $X1; Vm1*S1/(Km1 + S1);
BottomBranch: S1 -> $X2; Vm2*S1/(Km2 + S1);

X0 = 3.4;   X1 = 0.0;
S1 = 0.1;
Vm = 12;   p.Km = 0.1;
Vm1 = 14;  p.Km1 = 0.4;
Vm2 = 16;  p.Km2 = 3.4;
```

C.3 Using libRoadRunner in Python

libRoadRunner is a high performance simulator that can simulate models described using SBML. In order to use Antimony with libRoadRunner it is necessary to first convert an Antimony description into SBML and then load the SBML into libRoadRunner. Tellurium provides a handy routine called `loadAntimonyModel` to help with this task (The short-cut name is `loada`). To load an Antimony model we first assign an Antimony description to a string variable, for example:

```
model = '''
S1 -> S2; k1*S1;

S1 = 10; k1 = 0.1;
'''
```

We now use the `loadAntimonyModel` (`model`) or `loada` to load the model into libRoadRunner.

```
>>> rr = te.loadAntimonyModel (model)
```

Listing C.11 Loading an Antimony model

In this book we generally use the short-cut command as follows:

```
rr = te.loada ('''  
S1 -> S2; k1*S1;  
  
S1 = 10; k1 = 0.1;  
'''')  
>>>
```

Listing C.12 Loading an Antimony model using the short-cut command

Note that `loadAntimonyModel` and `loada` are part of the Tellurium Python package supplied with the Tellurium installer. If the Tellurium packages hasn't been loaded, use the following command to load the Tellurium package:

```
>>> import tellurium as te
```

Listing C.13 Importing the Tellurium Package

C.3.1 Time Course Simulation

Once a model has been loaded into libRoadRunner, performing a simulation is very straight forward. To simulate a model we use the libRoadRunner `simulate` method. This method has many options but for everyday use four options will suffice. The following panel illustrates a number examples of how to use `simulate`.

```
>>> result = rr.simulate ()  
>>> result = rr.simulate (0, 10)  
>>> result = rr.simulate (0, 10, 100)  
>>> result = rr.simulate (0, 10, 100, ['time', 'S1'])
```

Listing C.14 Calling the `simulate` method

Argument	Description
1st	Start Time
2nd	End Time
3rd	Number of Points
4th	Selection List

Let us focus on the forth version of the simulate method that takes four arguments. This call will run a time course simulation starting at time zero, ending at time 10 units, and generating 100 points. The results of the run are deposited in the matrix variable, `result`. At the end of the run, the `result` matrix will contain columns corresponding to the time column and all the species concentrations as specified by the forth argument. The forth argument can be used to change the columns that are returned from the simulate method. For example:

```
>>> result = rr.simulate (0, 10, 1000, ['S1'])}
```

will return a matrix 1,000 rows deep and one column wide that corresponds to the level of species S1. Note that the special variable Time is available and represents the independent time variable in the model.

To visualize the output in the form of a graph, one can pass the matrix of results to the plot command. In the following example we return one species level, S1 and three fluxes. Finally we plot the results.

```
result = rr.simulate (0, 10, 1000, ['Time', 'S1', 'J1', 'J2', 'J3']);
te.plotWithLegend (rr, result)
```

or if we are not interested in the result data itself we can use the libRoadRunner plot:

```
rr.simulate (0, 10, 1000, ['Time', 'S1', 'J1', 'J2', 'J3']);
rr.plot()
```

It is possible to set the output column selections separately using the command:

```
rr.selections = ['time', 'S1']
```

This can save some typing each time a simulation needs to be carried out. By default the selection is set to time as the first column followed by all molecular species concentrations. As such it is more common to simply enter the command:

```
>>> result = rr.simulate (0, 10, 50)
```

In fact even the start time and end time and number of points are optional and if missing, simulate

will revert to its defaults.

```
>>> result = rr.simulate()
```

C.3.2 Plotting Simulation Results

Tellurium comes with Matplotlib which is a common plotting package used by many Python users. To simplify its use we provide two simple plotting calls:

```
te.plot (array)
te.plotWithLegend (rr, array)
```

The first takes the resulting array generated by a call to `simulate` and uses the first column as the x axis and all subsequent columns as y axis data. The second call takes the roadrunner variable as well as the array and does the same kind of plot but this time adds a legend to the plot. We will use the first plotting command in the next section where we merge together multiple simulations.

C.3.3 Applying Perturbations to a Simulation

Often in a simulation we may wish to perturb a species or parameter at some point during the simulation and observe what happens. One way to do this in Tellurium is to carry out two separate simulations where a perturbation is made in between the two simulations. For example, let's say we wish to perturb the species concentration for a simple two step pathway and watch the perturbation decay. First, we simulate the model for 10 time units; this gives us a transient and then a steady state.

```
import numpy # Required for vstack
import tellurium as te

rr = te.loada ('''
$X_0 \rightarrow S_1; k_1*X_0;
S_1 \rightarrow X_1; k_2*S_1;

X_0 = 10; k_1 = 0.3; k_2 = 0.15;
''')

m1 = rr.simulate (0, 40, 50)
```

We then make a perturbation in S_1 as follows:

```
rr.S1 = rr.S1 * 1.6
```

which increases S1 by 60%. We next carry out a second simulation:

```
m2 = rr.simulate (40, 80, 50)
```

Note that we set the time start of the second simulation to the end time of the first simulation. Once we have the two simulations we can combine the matrices from both simulations using the Python command `vstack`

```
% Merge the two result array together  
m = numpy.vstack ((m1, m2))
```

Finally, we plot the results, screen-shot shown in Figure C.4.

```
te.plotArray (m)
```

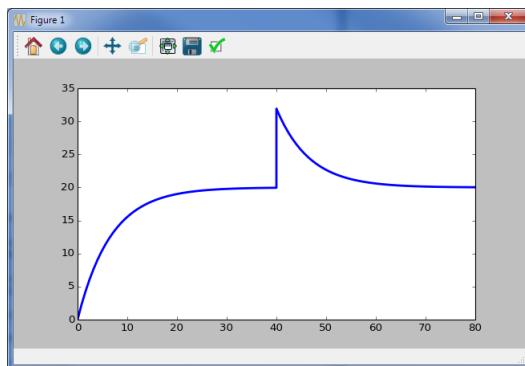


Figure C.4 Screen-shot from Matplotlib showing effect of perturbation in S1.

C.3.4 Steady State and Metabolic Control

To evaluate the steady-state first make sure the model values have been previously initialized, then enter the following statement at the console.

```
>>> rr.getSteadyState()
```

This statement will attempt to compute the steady state and return a value indicating how effective the computation was. It returns the norm of the rate of change vector (i.e. `sqrt (Sum of dydt)`). The closer this is to zero, the better the approximation to the steady state. Anything less than 10^{-4} usually indicates that a steady state has been found.

Once a steady state has been evaluated, the values of the metabolites will be at their steady state values, thus S1 will equal the steady state concentration of S1.

The fluxes through the individual reactions can be obtained by either referencing the name of the reaction (e.g. J1), or via the short-cut command rv. The advantage to looking at the reaction rate vector is that the individual reaction fluxes can be accessed by indexing the vector (see example below). **Note that indexing is from zero.**

```
>>> print rr.J1, rr.J2, rr.J3
3.4, ...
>>> for i in range (0, 2):
...     print rr.rv()[i]
3.4
etc
->
```

To compute control coefficients use the statement:

`getCC (Dependent Measure, Independent parameter)`

The dependent measure is an expression usually containing flux and metabolite references, for example, S1, J1. The independent parameter must be a simple parameter such as a Vmax, Km, ki, boundary metabolite (X0), or a conservation total such as cm_xxxx. Examples include:

```
rr.getCC ('J1', 'Vmax1')
rr.getCC ('J1', 'Vm1') + rr.getCC ('J1', 'Vm2')
rr.getCC ('J1', 'X0')
rr.getCC ('J1', 'cm_xxxx')
```

To compute elasticity coefficients use the statement:

`getEE (Reaction Name, Parameter Name)`

For example:

```
rr.getEE ('J1', 'X0')
rr.getEE ('J1', 'S1')
```

Since getCC and getEE are built-in functions, they can be used alone or as part of larger expressions. Thus, it is easy to show that the response coefficient is the product of a control coefficient and the adjacent elasticity by using:

```
R = rr.getCC ('J1', 'X0')
print R - rr.getCC ('J1', 'Vm') * rr.getEE ('J1', 'X0')
```

To obtain the conservation matrix for a model use the model method, `getConservationMatrix`. Note that in the Antimony text we use the `var` word to predeclare the species so that we can set up the rows of the stoichiometry matrix in a certain order if we wish. This allows us to obtain conservation matrices with only positive terms.

```
import tellurium as te

rr = te.loada ('''
var ES, S1, S2, E;

J1: E + S1 -> ES; v;
J2: ES -> E + S2; v;
J3: S2 -> S1; v;
''')

print rr.getConservationMatrix()
print rr.fs()

# Output
[[ 1.  1.  0.]
 [ 1.  0.  0.  1.]]
['ES', 'S1', 'S2', 'E']
```

The result given above indicates that the conservation relations, $ES + S1 + E$ and $E + ES$ exist in the model. As a result, Tellurium would generate two internal parameters of the form `cm` corresponding to the two relations.

C.3.5 Other Model Properties of Interest

There are a number of predefined objects associated with a reaction network model which might also be of interest. For example, the stoichiometry matrix, `sm`, the rate vector `rv`, the species levels vector and `dv` which returns the rates of change.

```
print rr.sm()
print rr.rv()
print rr.sv()
print rr.dv()
```

The names for the parameters and variables in a model can be obtained the short-cuts:

```
print rr.fs() # List of floating species names
print rr.bv() # List of boundary species names
```

```
print rr.ps() # List of parameter names  
print rr.rs() # List of reaction names  
print rr.vs() $ List of compartment names
```

The jacobian matrix can be returned using the command: `rr.getFullJacobian()`.

C.4 Generating SBML and Matlab Files

Tellurium can import and export standard SBML [29] as well as export Matlab scripts for the current model. To load a model in SBML, load it directly into libRoadRunner. For example:

```
>>> rr = roadrunner.RoadRunner ('mymodel.xml')  
>>> result = rr.simulate (0, 10, 100)
```

There are two ways to retrieve the SBML, one can either retrieve the original SBML loaded using `rr.getSBML()` or retrieve the *current* SBML using `rr.getCurrentSBML()`. Retrieving the current SBML can be useful if the model has been changed. To save the SBML to a file we can use the Tellurium helper function `saveToFile ()`, for example:

```
>>> te.saveToFile ('mySBMLModel.xml', rr.getCurrentSBML())
```

To convert an SBML file into Matlab, use the `getMatlab` method:

```
import tellurium as te  
  
rr = te.loada (''  
    S1 -> S2; k1*S1;  
    S2 -> S3; k2*S2;  
    S1 = 10; k1 = 0.1; k2 = 0.2;  
'')  
  
# Save the SBML  
te.saveToFile ('model.xml', rr.getSBML())  
  
# Save the Matlab  
te.saveToFile ('model.mat', rr.getMatlab())
```

C.5 Exercise

Figure C.5 shows a two gene circuit with a feedforward loop. Assume the following rate laws for the four reactions:

$$v_1 = k_1 X_o$$

$$v_2 = k_2 x_1$$

$$v_3 = k_3 X_o$$

$$v_4 = k_4 x_1 x_2$$

Assume that all rate constants are equal to one and that $X_o = 1$. Assume X_o is a fixed species.

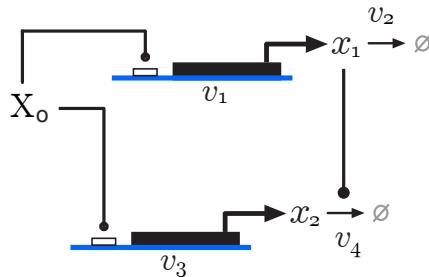


Figure C.5 Two gene circuit with feedforward loop.

1. Use Tellurium to model this system.
2. Run a simulation of the system from 0 to 10 time units.
3. Next, change the value of X_o to 2 (double it) and rerun the simulation for another 10 time units from where you left off in the last simulation. Combine both simulations and plot the result, that is time on the x-axis, and X_o and x_2 on the y-axis.
4. What do you see?
5. Write out the differential equations for x_1 and x_2 .
6. Show algebraically that the steady state level of x_2 is independent of X_o .

D

Enzyme Kinetics in a Nutshell

Enzymes

Enzymes are protein molecules that can accelerate a chemical reaction without changing the reaction equilibrium constant.

Enzyme Kinetics

Enzyme kinetics is a branch of science that deals with the many factors that can affect the rate of an enzyme-catalysed reaction. The most important factors include the concentration of enzyme, reactants, products, and the concentration of any modifiers such as specific activators, inhibitors, pH, ionic strength, and temperature. When the action of these factors is studied, we can deduce the kinetic mechanism of the reaction. That is, the order in which substrates and products bind and unbind, and the mechanism by which modifiers alter the reaction rate.

D.1 Michaelis-Menten Kinetics

The standard model for enzyme action describes the binding of free enzyme to the reactant forming an **enzyme-reactant complex**. This complex undergoes a transformation, releasing product and free

enzyme. The free enzyme is then available for another round of binding to new reactant.



where k_1 , k_{-1} and k_2 are rate constants, S is substrate, P is product, E is the free enzyme, and ES the enzyme-substrate complex.

By assuming a steady state condition on the enzyme substrate complex, we can derive the Briggs-Haldane equation relation (sometimes mistakenly called the Michaelis-Menten equation):

$$v = \frac{V_m S}{K_m + S} \quad (\text{D.2})$$

where V_m is the maximal velocity, and K_m the substrate concentration that yields half the maximum velocity.

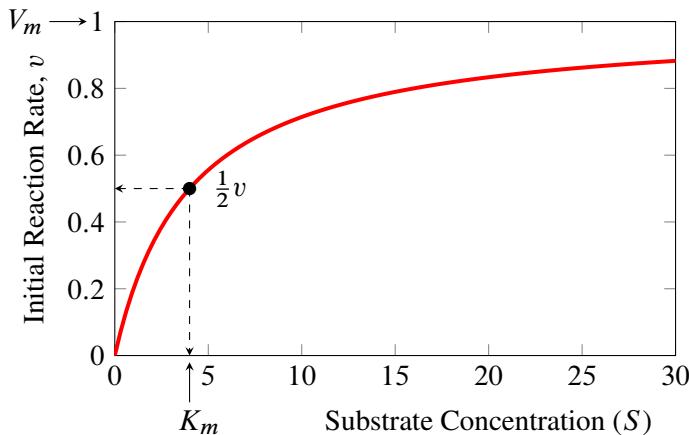


Figure D.1 Relationship between the initial rate of reaction and substrate concentration for a simple Michaelis-Menten rate law. The reaction rate reaches a limiting value called the V_m . K_m is set to 4.0 and V_m to 1.0. The K_m value is the substrate concentration that gives half the maximal rate.

D.2 Reversibility and Product Inhibition

In vivo it is unlikely that an enzyme reaction is completely irreversible. Even if an enzyme shows no reverse reaction rate from product to substrate, there is still likely to be some degree of product inhibition because the product can bind to the active site and compete with the substrate.

D.3 Reversible Rate laws

An alternative and more realistic model is the reversible form:



The aggregate rate law for the reversible form of the mechanism can also be derived and is given by:

$$v = \frac{V_f S/K_S - V_r P/K_P}{1 + S/K_S + P/K_P} \quad (\text{D.4})$$

D.4 Haldane Relationship

For the reversible enzyme kinetic law there is an important relationship:

$$K_{eq} = \frac{P_{eq}}{S_{eq}} = \frac{V_f K_P}{V_r K_S} \quad (\text{D.5})$$

This equation shows that the four kinetic constants, V_f , V_r , K_P and K_S are not independent. Haldane relationships can be used to eliminate one of the kinetic constants by substituting the equilibrium constant in its place. This is useful because equilibrium constants tend to be known compared to kinetic constants which may be unknown. By incorporating the Haldane relationship, we can eliminate the reverse maximal velocity (V_r) from D.4 to yield the equation:

$$v = \frac{V_f / K_S (S - P / K_{eq})}{1 + S / K_S + P / K_P} \quad (\text{D.6})$$

Separating out the terms makes it easier to see that the above equation can be partitioned into a number of distinct parts:

$$v = V_f \cdot (1 - \Gamma / K_{eq}) \cdot \frac{S / K_S}{1 + S / K_S + P / K_P} \quad (\text{D.7})$$

where $\Gamma = P / S$. The first term, V_f , is the maximal velocity; the second term, $(1 - \Gamma / K_{eq})$, indicates the direction of the reaction according to thermodynamic considerations. The last term refers to the fractional saturation with respect to substrate. Thus we have a maximal velocity, a thermodynamic and a saturation term.

D.5 Competitive Inhibition

There are many molecules capable of slowing down or speeding up the rate of enzyme catalyzed reactions. Such molecules are called enzyme inhibitors and activators. One common type of inhibition,

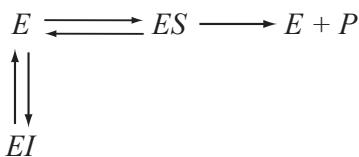
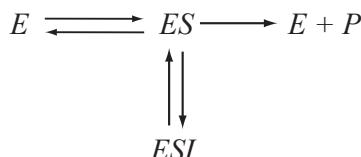
a) Competitive Inhibition**b) Uncompetitive Inhibition**

Figure D.2 Competitive and Uncompetitive Inhibition. P is the concentration of product, E is the free enzyme, ES the enzyme-substrate complex, and ESI the enzyme-substrate-inhibitor complex.

called **competitive inhibition**, occurs when the inhibitor is structurally similar to the substrate so that it competes for the active site by forming a dead-end complex.

The kinetic mechanism for a pure competitive inhibitor is shown in Figure D.2(a), where I is the inhibitor and EI the enzyme inhibitor complex. If the substrate concentration is increased, it is possible for the substrate to eventually out compete the inhibitor. For this reason the inhibitor alters the enzyme's apparent K_m , but not the V_m .

$$\begin{aligned} v &= \frac{V_m S}{S + K_m \left(1 + \frac{I}{K_i}\right)} \\ &= \frac{V_m S / K_m}{1 + S / K_m + I / K_i} \end{aligned} \tag{D.8}$$

At $I = 0$, the competitive inhibition equation reduces to the normal irreversible Michaelis-Menten equation. Note that the term $K_m(1 + I/K_i)$ in the first equation more clearly shows the impact of the inhibitor, I , on the K_m . The inhibitor has no effect on the V_m .

A reversible form of the competitive rate law can also be derived:

$$v = \frac{\frac{V_m}{K_s} \left(S - \frac{P}{K_{eq}}\right)}{1 + \frac{S}{K_s} + \frac{P}{K_p} + \frac{I}{K_i}} \tag{D.9}$$

where V_m is the forward maximal velocity, and K_s and K_p are the substrate and product half saturation constants.

Sometimes reactions appear irreversible, where no discernable reverse rate is detected, and yet the forward reaction is influenced by the accumulation of product. This effect is caused by the product competing with substrate for binding to the active site and is often called **product inhibition**. Given that product inhibition is a type of competitive inhibition, we will briefly discuss it. An important

industrial example of this is the conversion of lactose to galactose by the enzyme β -galactosidase where galactose competes with lactose, slowing the forward rate [21].

To describe simple product inhibition with rate irreversibility, we can set the P/K_{eq} term in the reversible Michaelis-Menten rate law (D.4) to zero. This yields:

$$v = \frac{V_m S}{S + K_m \left(1 + \frac{P}{K_p}\right)} \quad (\text{D.10})$$

It is not surprising to discover that equation (D.10) has exactly the same form as the equation for competitive inhibition (D.8). As the product increases, it out competes the substrate and therefore slows down the reaction rate.

We can also derive the equation by using the following mechanism and the rapid-equilibrium assumption:



where the reaction rate v is assumed to be proportional to ES .

D.6 Cooperativity

Many proteins are known to be oligomeric, meaning they are composed of more than one identical protein subunit where each subunit has one or more binding sites. Often the individual subunits are identical.

If the binding of a ligand (a small molecule that binds to a larger macromolecule) to one site alters the affinity at other sites on the same oligomer, it is called **cooperativity**. If ligand binding increases the affinity of subsequent binding events, it is termed **positive cooperativity** whereas if the affinity decreases, it is termed **negative cooperativity**. One characteristic of positive cooperativity is that it results in a sigmoidal response instead of the usual hyperbolic response.

The simplest equation that displays sigmoid like behavior is the Hill equation:

$$v = \frac{V_m S^n}{K_d + S^n} \quad (\text{D.12})$$

One striking feature of many oligomeric proteins is the way individual monomers are physically arranged. Often one will find at least one axis of symmetry. The individual protein monomers are not arranged in a haphazard fashion. This level of symmetry may imply that the gradual change in the binding constants as ligands bind, as suggested by the Adair model, might be physically implausible. Instead, one might envision transitions to an alternative binding state that occurs within the entire oligomer complex. This model was originally suggested by Monod, Wyman and Changeux [40], abbreviated as the MWC model. The original authors laid out the following criteria for the MWC model:

1. The protein is an oligomer.
2. Oligomers can exist in two states: R (relaxed) and T (tense). In each state, symmetry is preserved and all subunits must be in the same state for a given R or T state.
3. The R state has a higher ligand affinity than the T state.
4. The T state predominates in the absence of ligand S.
5. The ligand binding microscopic association constants are all identical. This is in complete contrast to the Adair model.

Given these criteria, the MWC model assumes that an oligomeric enzyme may exist in two conformations, designated T (tensed, square) and R (relaxed, circle). The equilibrium between the two states has an equilibrium constant $L = T/R$, which is also called the **allosteric constant**. If the binding constants of ligand to the two states are different, the distribution of the R and T forms can be displaced towards either one form or the other. By this mechanism, the enzyme displays sigmoid behavior. A minimal example of this model is shown in Figure D.3.

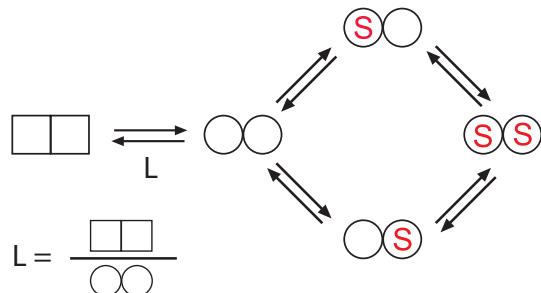


Figure D.3 A minimal MWC model, also known as the exclusive model, showing alternative microscopic states in the circle (relaxed) form. L is called the allosteric constant. The square form is called the tense state.

In the **exclusive model** (Figure D.3) the ligand can only bind to the relaxed form (circle). The mechanism that generates sigmoidicity in this model works as follows. When ligand binds to the relaxed form, it displaces the equilibrium from the tense form to the relaxed form. In doing so, additional ligand binding sites become available. Thus, one ligand binding may generate four or more new binding sites. Eventually there are no more tense states remaining, at which point the system is saturated with ligand. The overall binding curve will therefore be sigmoidal and will show positive cooperativity. Given the nature of this model, it is not possible to generate negative cooperativity. By assuming equilibrium between the various states, it is possible to derive an aggregate equation for the dimer

case of the exclusive MWC model:

$$v = V_m \frac{\frac{S}{k_R} \left(1 + \frac{S}{k_R}\right)}{\left(1 + \frac{S}{k_R}\right)^2 + L}$$

This also generalizes to n subunits as follows:

$$Y = \frac{\frac{S}{k_R} \left(1 + \frac{S}{k_R}\right)^{n-1}}{\left(1 + \frac{S}{k_R}\right)^n + L} \quad (\text{D.13})$$

For more generalized reversible rate laws that exhibit sigmoid behavior, the reversible Hill equation is a good option. Invoking the rapid-equilibrium assumption, we can form a reversible rate law that shows cooperativity:

$$v = \frac{V_f \alpha (1 - \rho) (\alpha + \pi)}{1 + (\alpha + \pi)^2}$$

where $\rho = \Gamma/K_{eq}$ and α and π are the ratio of reactant and product to their respective equilibrium constant, α/K_S and π/K_P . For an enzyme with h (using the author's original notation) binding sites, the general form of the reversible Hill equation is given by:

$$v = \frac{V_f \alpha (1 - \rho) (\alpha + \pi)^{h-1}}{1 + (\alpha + \pi)^h} \quad (\text{D.14})$$

D.7 Allostery

An allosteric effect is where the activity of an enzyme or other protein is affected by the binding of an effector molecule at a site on the protein's surface, other than the active site. The MWC model described previously can be easily modified to accommodate allosteric action.

The key to including allosteric effectors is to influence the equilibrium between the tense (T) and relaxed (R) states (See Figure D.4). To influence the sigmoid curve, an allosteric effector need only displace the equilibrium between the tense and relaxed forms. For example, to behave as an activator, an allosteric effector needs to preferentially bind to the R form and shift the equilibrium away from the less active T form. An allosteric inhibitor would do the opposite, that is bind preferentially to the T form so that the equilibrium shifts towards the less active T form. In both cases the V_m of the enzyme is unaffected.

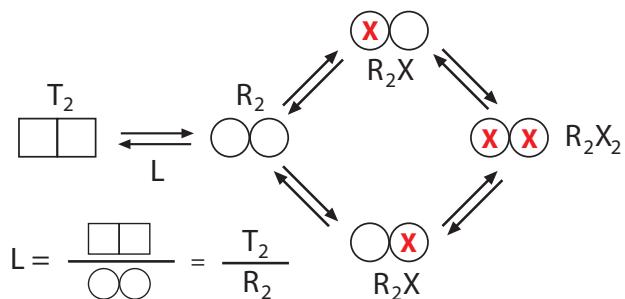


Figure D.4 Exclusive MWC model based on a dimer showing alternative microscopic states in the form of T and R states. The model is exclusive because the ligand, X , only binds to the R form.

The net result of this is to modify the normal MWC aggregate rate law to the following if the effector is an inhibitor:

$$v = V_m \frac{\alpha (1 + \alpha)^{n-1}}{(1 + \alpha)^n + L(1 + \beta)^n} \quad (\text{D.15})$$

where $\alpha = S/K_s$, $\beta = I/K_I$, and K_s and K_I are kinetic constants related to each ligand. A MWC model that is regulated by an inhibitor or an activator is described by the equation:

$$v = V_m \frac{\alpha (1 + \alpha)^{n-1}}{(1 + \alpha)^n + L \frac{(1 + \beta)^n}{(1 + \gamma)^n}}$$

There are also reversible forms of the allosteric MWC model but they are fairly complex. Instead, it is possible to modify the reversible Hill rate law to include allosteric ligands.

$$v = \frac{V_f \alpha \left(1 - \frac{\Gamma}{K_{eq}}\right) (\alpha + \pi)^{h-1}}{\frac{1 + \mu^h}{1 + \sigma \mu^h} + (\alpha + \pi)^h} \quad (\text{D.16})$$

where:

$$\begin{aligned} \sigma < 1 && \text{inhibitor} \\ \sigma > 1 && \text{activator} \end{aligned}$$

Simple Hill Equations

When modeling gene regulatory networks, we often need simple activation and repression rate laws. It is common to use the following Hill like equations to model activation and repression, respectively.

The third equation shows one example of how we can model dual repression and activation, where S_1 acts as the activator and S_2 the inhibitor. n_1 and n_2 are Hill like coefficients which may be used to alter the responsiveness of each factor.

$$\text{Activation: } v = \frac{V_m S^n}{K + S^n}$$

$$\text{Repression: } v = \frac{V_m}{K + S^n}$$

$$\text{Dual: } v = \frac{V_m S_1^{n_1}}{1 + K_1 S_1^{n_1} + K_2 S_2^{n_2} + K_3 S_1^{n_1} S_2^{n_2}}$$

Further Reading

1. Sauro HM (2012) Enzyme Kinetics for Systems Biology. 2nd Edition, Ambrosius Publishing
ISBN: 978-0982477335

E

Visualizing Simulations

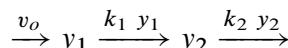
E.1 Time Course Simulation

The simplest way to visualize a simulation is to plot the time courses that describe the evolution of the state variables. Although useful, such plots are not the only way to display a behavior of a model. In this chapter we will briefly look at three different ways to visualize a model.

E.2 NullClines

Nullclines are very useful for looking at systems of two variables. If we have a variable x_i with differential equation df_i/dt , then the nullcline is the line that satisfies, $df_i/dt = 0$. If two nullclines are plotted, then their intersection makes the steady state points.

For example, the system:



with ODEs:

$$\frac{dy_1}{dt} = v_o - k_1 y_1$$

$$\frac{dy_2}{dt} = k_1 y_1 - k_2 y_2$$

has two nullclines. Assuming y_1 is plotted on the x axis and y_2 on the y axis, then the first nullcline is simply $y_1 = v_o/k_1$, that is a vertical straight line at y_1 . The second equation is given by:

$$y_2 = k_1 y_1 / k_2$$

and represents a straight line with slope k_1/k_2 running through the zero point.

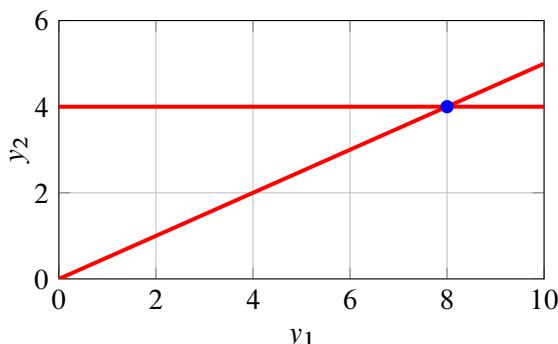


Figure E.1 Nullclines for the system: $dy_1/dt = v_o - k_1 y_1$; $dy_2/dt = k_1 y_1 - k_2 y_2$ where $v_o = 6$, $k_1 = 0.15$, $k_2 = 3$. The intersection point marks the steady state when both equations equal zero.

E.3 Bifurcation Plots

Phase portraits and nullcline visualize time course trajectories. A bifurcation plot is quite different. In a bifurcation plot we visualize how the **steady state solution** changes as a function of a model parameter. So simple models, a bifurcation plot can be quite simple. For example, consider the simple model:



The differential equation for this is:

$$\frac{dS_1}{dt} = v_o - kS_1$$

The steady state solution can be found by setting the differential equation to zero and solving for S_1 . If we do this we obtain:

$$S_1 = \frac{v_o}{k}$$

To plot a bifurcation plot we must choose a parameter, for example k and plot the steady state solution with respect to k .

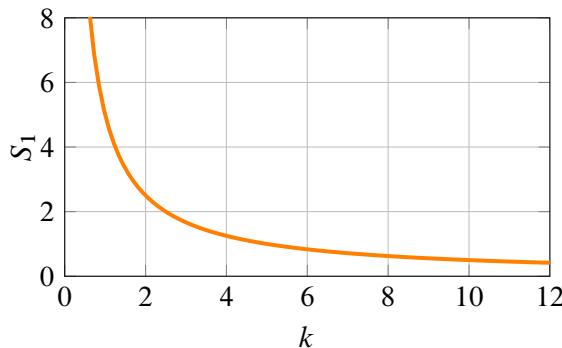


Figure E.2 Very simple bifurcation plot system $\xrightarrow{v_o} S_1 \xrightarrow{kS_1}$ with respect to k . $v_o = 5$. The graph shows how the steady state value of S_1 changes as a function of k .

Not all bifurcation plots are so simple and later on we will see what bifurcation plots can be so useful when we consider more complex behavior as well as the numerical issues that present themselves in more complex cases.

F

Math Notes

F.1 Polynomials

An equation of the form:

$$a_n s^n + a_{n-1} s^{n-1} + \dots + a_1 s + a_o$$

is called a polynomial. Such equations involve only addition, subtraction, multiplication and variables raised to positive integer powers. Given this definition, the equations, $3s^{-2}$ and $2/(s+2)$ are not polynomials because the exponents are negative. The degree of a polynomial is the highest integer power, for example the degree of the polynomial $7s^4 + 3s^2 + 2s$ is four. Of particular interest in many applications are the roots of the polynomial function:

$$F(s) = 0 = a_n s^n + a_{n-1} s^{n-1} + \dots + a_1 s + a_o$$

An important theorem related to polynomials is the factor theorem. This states that given a polynomial $F(s)$, then $(s-p)$ is a factor of $F(s)$ if and only if $F(p) = 0$. That is p is a root of $F(s)$. For example, if $F(s) = s^3 - 8$ then $s-2$ is a factor because 2 is a root of $F(s)$. In general we can factor a polynomial $F(s)$ of n^{th} degree into n linear factors:

$$F(s) = A(s - p_n)(s - p_{n-1}) \dots (s - p_o) = 0$$

Consider the second-order polynomial where we factor and collect terms:

$$Q_2(s) = s^2 + a_1 s + a_o = (s - p_1)(s - p_2) = s^2 - (p_1 + p_2)s + p_1 p_2 \quad (\text{F.1})$$

Likewise we can do the same with a third-order polynomial:

$$\begin{aligned} Q_3(s) &= s^3 + a_2 s^2 + a_1 s + a_0 = (s - p_1)(s - p_2)(s - p_3) \\ &= s^3 - (p_1 + p_2 + p_3)s^2 + (p_1 p_2 + p_1 p_3 + p_2 p_3)s - p_1 p_2 p_3 \end{aligned} \quad (\text{F.2})$$

By induction we can extend this to an n th-order polynomial:

$$Q_n(s) = s^n + a_{n-1} s^{n-1} + \dots + a_1 s + a_0 \quad (\text{F.3})$$

so that the coefficients are given by:

a_{n-1} = the negative of the sum of all roots

a_{n-2} = the sum of products of all possible combinations taken two at a time

a_{n-3} = the negative sum of products of all possible combinations taken three at a time

⋮

$a_0 = (-1)^n$ multiplied by the product of all roots

From this analysis we can discern a useful pattern. First we make sure that the polynomial must be in the form F.3, that is all coefficients are non-zero. The one exception is if a_0 is zero, we can divide throughout by s and this we yield the form F.3.

Looking at the second-order polynomial, we can see that if the roots p_i are negative then the terms $-(p_1 + p_2)$ and $p_1 p_2$ must be positive. For the general case, $Q_n(s)$ we can make the same assertion, however note that the reverse statement, if all the coefficients are positive then all the roots will be negative is **not** necessarily true. This rule applies to the general case $Q_n(s)$ and can be stated as.

A **necessary** but not sufficient condition for negative roots is that all coefficients must be positive. This includes the case when the roots are complex where the real parts will be negative.

Note that for all roots to be negative no coefficient can be zero.

1. If any coefficient a_i is equal to zero then one or more roots will be positive
2. If any coefficient a_i is negative, then at least one root is positive
3. If all coefficients are positive then roots may be negative or positive

Note that for a polynomial higher than second order, the rules provide only a necessary condition that all roots will be negative. For higher order polynomials, additional rules must be invoked such as constructing the Routh table, see section 14.5. Using the Routh method the following rules can be derived for 2nd 3rd and 4th order polynomials.

Quadratic: $a_2s^2 + a_1s + a_o = 0$

All roots are negative if

$$a_2 > 0, a_1 > 0, a_o > 0$$

Cubic: $a_3s^3 + a_2s^2 + a_1s + a_o = 0$

All roots are negative if:

$$a_3 > 0, a_2 > 0, a_1 > 0, a_o > 0$$

and

$$a_1a_2 > a_0a_3$$

Quartic: $a_4s^4 + a_3s^3 + a_2s^2 + a_1s + a_o = 0$

All roots are negative if:

$$a_4 > 0, a_3 > 0, a_2 > 0, a_1 > 0, a_o > 0$$

and

$$a_1a_2 - a_0a_3 > 0$$

and

$$a_1a_2a_3 - a_1^2a_4 - a_0a_3^2 > 0$$

F.2 Absolute Values

$$|-a| = |a|$$

$$|ab| = |a||b|$$

$$|a/b| = |a|/|b|$$

F.3 Radian Measure

The radian is the standard angular unit and is widely used in mathematics and engineering. Figure F.1 explain what a radian is graphically. Given the radius, r of a circle, we sweep out an arc on the circle of the **same** length. The angle suspended between the radius and the arc is defined to be 1 radian. Given that we know that the length of the circumference of a circle is $2\pi r$, we can state that if our circle has radius 1, then the fraction of the circle taken up by one radian is:

$$\frac{\text{radius}}{\text{circumference}} = \frac{1}{2\pi} \approx 16\%$$

The other way to think about this is that since a full circle represents 360° , one radian must represent:

$$\frac{\text{radius}}{\text{circumference}} = \frac{\text{angle}}{360^\circ} = \frac{1}{2\pi}$$

Therefore:

$$\text{angle of 1 radian in degrees} = \left(\frac{360}{2\pi}\right)^\circ \approx 57.30^\circ$$

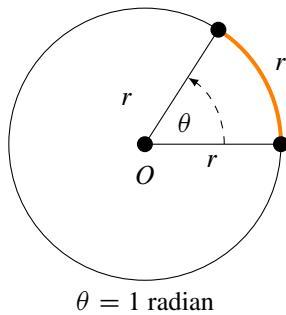


Figure F.1 Radians

This angle is independent of the value of the radius itself. Therefore it is convenient to define a radian in terms of a unit circle. Conversion formula:

$$\text{Degrees to radians: } x \text{ degrees} = \left(\frac{\pi}{180} x\right) \text{ radians}$$

$$\text{Radians to degrees: } x \text{ radians} = \left(\frac{180}{\pi} x\right) \text{ degrees}$$

Here are some useful numbers to memorize:

Degrees	Radians
0	0
90	$\frac{\pi}{2}$
180	π
270	$\frac{3\pi}{2}$
360	2π

F.4 Common Integrals

$$\int k \, dx = kx \quad (\text{F.4})$$

$$\int x \, dx = \frac{1}{2}x^2 \quad (\text{F.5})$$

$$\int \frac{1}{x^2} \, dx = -\frac{1}{x} \quad (\text{F.6})$$

$$\int \frac{1}{x} \, dx = \ln|x| \quad (\text{F.7})$$

$$\int e^{ax} \, dx = \frac{1}{a}e^{ax} \quad (\text{F.8})$$

$$\int \sin(ax) \, dx = -\frac{1}{a} \cos(ax) \quad (\text{F.9})$$

Definite Integrals

If a function f is continuous on the interval a to b , then the **definite integral** of f is given by:

$$\int_a^b f(x) \, dx = \lim_{n \rightarrow \infty} \sum_{i=1}^n f(x_i) \Delta x$$

We can compute an indefinite integral using:

$$\int_a^b f(x) \, dx = F(b) - F(a)$$

where $F()$ is the antiderivative of $f(x)$. For example:

$$\int_1^2 4t^3 \, dt = t^4 \Big|_1^2 = 2^4 - 1^4 = 15$$

Improper Integrals

An improper integral is one where either one of the limits is infinity or the integral fails to converge in the interval of integration. For example the following is an improper integral:

$$\int_0^\infty e^{-3x} dx$$

For improper integrals, what is important is that the integral converges. For example consider the following improper integral:

$$\int_1^\infty \frac{dx}{x^2}$$

Let us first integrate assuming the upper limit is t :

$$\int_1^t \frac{dx}{x^2} = -\frac{1}{x} \Big|_1^t = 1 - \frac{1}{t}$$

We can now look at the integral in the limit as t approaches infinity::

$$\lim_{t \rightarrow \infty} \left(1 - \frac{1}{t} \right) = 1$$

We note that the integral converges to one. In contrast the following integral does not diverge:

$$\int_1^\infty \frac{dx}{x}$$

As before let us first consider that the upper limit is t :

$$\int_1^t \frac{dx}{x} = \ln(x) = \ln(t) - \ln(1) = \ln(t)$$

In the limit as t approaches infinity we have:

$$\lim_{t \rightarrow \infty} \ln(t) = \infty$$

Therefore this integral does not converge but diverges. The convergence of improper integrals is particularly important when dealing with the Laplace transform. For example the Laplace transform where $f(t) = 1$ is given by:

$$\begin{aligned} \int_0^\infty e^{-st} dt &= \left[-\frac{1}{s} e^{-st} \right] \\ &= \left[-\frac{1}{s} e^0 \right] - \left[-\frac{1}{s} e^{-\infty} \right] = \frac{1}{s} \end{aligned}$$

and as we can see it converges.

F.5 Taylor Series

Expressions like $1 + 2x + 6x^2$ and $2 + 4x + x^2 - 3x^3$ that contain the sum of a number of terms raised to a positive power are called polynomials. The only operations allowed in a polynomial are addition, subtraction, multiplication and non-negative integer powers. One of the simplest polynomials is the straight line, $y = a + bx$, termed a polynomial of first degree. The coefficients, a and b , can be chosen so that the line will pass through any two points. As such, we can express any straight line using $y = a + bx$. Similarly for a polynomial of second degree, $y = a + bx + cx^2$, a parabola, we can choose the constants, a , b , and c so that the curve passes through any three points.

It follows that we can find a polynomial equation of n^{th} degree that will pass through any $n + 1$ points. If the polynomial has an infinite number of terms, we imagine it could be made to follow any function, $f(x)$, by suitable adjustment of the polynomial coefficients. Although this statement may not always be true, in many cases it is, which makes the polynomial series very useful.

A polynomial of infinite degree is called a polynomial series:

$$f(x) = c_0 + c_1x + c_2x^2 + c_3x^3 + \dots$$

The question is, how can we find the polynomial series that will represent a given function, for example $\sin(x)$? To answer this we have to determine the constants, c_0, c_1 , etc. in the polynomial equation. Let us assume that we wish to know the value of $\sin(x)$ at $x = 0$ using a polynomial series. At $x = 0$, all terms vanish except for c_0 , therefore at $x = 0$:

$$f(0) = c_0$$

We can therefore interpret the first constant, c_0 , as the value of the function at $x = 0$. What about c_1 ? Let us take the derivative of the series, that is:

$$f'(x) = c_1 + 2c_2x + 3c_3x^2 + \dots$$

If we set $x = 0$, we find that:

$$f'(0) = c_1$$

The second constant, c_1 , in the polynomial series is the first derivative of the function. If we take the second derivative we can also show at $x = 0$, $f''(x_0) = 2c_2$, that is $c_2 = f''(0)/2$. For the third derivative we can show $f'''(0) = 3(2)c_3$, that is $c_3 = f'''(0)/(3!)$. This pattern continues for the remaining terms in the polynomial so we can write:

$$f(x) = f(0) + f'(0)x + \frac{f''(0)}{2!}x^2 + \frac{f'''(0)}{3!}x^3 + \dots$$

This series is called the **Maclaurin series** for the function, $f(x)$. It approximates the function around the specific value of $x = 0$. To illustrate the use of the Maclaurin series, consider expanding $\sin(x)$

Function	First-Order	Second-order
$\frac{1}{1+x}$	$1 + x$	$1 + x + x^2$
$\sqrt{1+x}$	$1 + \frac{x}{2}$	$1 + \frac{x}{2} + \frac{x^2}{8}$
$\sin(x)$	x	$x - \frac{x^3}{3!}$

Table F.1 Examples of common approximations centered around $x = 0$.

around $x = 0$. $f(0)$ will equal $\sin(0) = 0$. $f'(0) = \cos(0) = 1$ and so on. We can therefore write the series as:

$$\begin{aligned}\sin(x) &= 0 + 1x + 0 - \frac{1}{3!}x^3 + 0 + \frac{1}{5!}x^5 - \dots \\ \sin(x) &= x - \frac{x^3}{3!} + \frac{x^5}{5!} - \dots\end{aligned}$$

What if we wanted to approximate a function about an arbitrary value of x ? To do this we would use the Taylor series which is a generalization of the Maclaurin series. The **Taylor series** is defined by:

$$f(x) = f(x_o) + \frac{\partial f}{\partial x_o}(x - x_o) + \frac{1}{2!} \frac{\partial^2 f}{\partial x_o^2}(x - x_o)^2 + \dots + \frac{1}{n!} \frac{\partial^n f}{\partial x_o^n}(x - x_o)^n + \dots \quad (\text{F.10})$$

where the approximation is now centered on x_o . If we set x_o equal to zero, we will obtain the Maclaurin series.

If we keep three terms, we obtain another very important approximation called the **quadratic approximation**:

$$f(x) = f(x_0) + \delta x \frac{df}{dt} + \frac{1}{2}(\delta x)^2 \frac{d^2 f}{dt^2} \quad (\text{F.11})$$

In optimization strategies near the optimum, we can often approximate the fitness surface using a quadratic function.

Taylor Series in Two or More Dimensions

It is possible to derive a Taylor series for equations with multiple variables, for example $f(x, y)$. In this case the expansion is a little bit more complicated. A Taylor series expansion around x_o and y_o

for the function $f(x, y)$ is given by:

$$\begin{aligned} f(x, y) \approx f(x_o, y_o) + \frac{\partial f}{\partial x_o}(x - x_o) + \frac{\partial f}{\partial y_o}(y - y_o) + \\ \frac{1}{2!} \left[(x - x_o)^2 \frac{\partial^2 f}{\partial x_o^2} + 2(x - x_o)(y - y_o) \frac{\partial^2 f}{\partial x_o \partial y_o} + (y - y_o)^2 \frac{\partial^2 f}{\partial y_o^2} \right] \\ + \dots \end{aligned}$$

where all derivatives are evaluated at the operating point, x_o, y_o . There is a very nice compact form for this equation which generalizes to any number of dimensions which can be written in terms of vectors and a matrix:

$$f(\mathbf{x}_o) = f(\mathbf{x}_o) + (\mathbf{x} - \mathbf{x}_o)^T \nabla f + \frac{1}{2!} (\mathbf{x} - \mathbf{x}_o)^T \mathbf{H} (\mathbf{x} - \mathbf{x}_o) + \dots \quad (\text{F.12})$$

where ∇f is called the gradient (or grad f), and \mathbf{H} the Hessian. Note that the vector notation, \mathbf{v}^T means that the vector is in row form because by convention a vector is often depicted as a column. Given a function, $f(x, y, \dots)$, ∇f is just the vector of partial derivatives:

$$\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \dots \right]^T$$

The following equivalent notation in terms of the unit vectors is also frequently found in the literature for ∇f :

$$\nabla f = \frac{\partial f}{\partial x} \mathbf{i} + \frac{\partial f}{\partial y} \mathbf{j} + \dots$$

The Hessian matrix, \mathbf{H} , for the function $f(x, y)$ is given by:

$$\mathbf{H} = \begin{bmatrix} \frac{\partial^2 f}{\partial x \partial x} & \frac{\partial^2 f}{\partial x \partial y} \\ \frac{\partial^2 f}{\partial y \partial x} & \frac{\partial^2 f}{\partial y \partial y} \end{bmatrix} = \begin{bmatrix} \frac{\partial^2 f}{\partial x_i \partial x_j} \end{bmatrix}$$

which can be naturally extended to functions with any number of variables.

F.6 Complex Numbers

Imaginary numbers are solutions to equations such as $\sqrt{-x}$. However there is no real numbers that satisfy this square root. Instead let us write the equation in the following form: $\sqrt{(x)(-1)} =$

$\sqrt{x}\sqrt{-1}$. The problem now reduces to defining the square root of -1. We will define $\sqrt{-1}$ as the imaginary unit, denoted i . We can therefore represent the solution to $\sqrt{-x}$ as $i\sqrt{x}$. Thus the solution for $\sqrt{-9}$ is $3i$. Given these definitions we can also state the following identities:

$$\begin{aligned} i^2 &= -1 \\ i^3 &= i^2i = -i \\ i^4 &= i^2i^2 = 1 \\ i^5 &= i^4i = i \\ &\vdots \end{aligned}$$

Although i is often used to represent the imaginary unit number, in engineering j is used instead to avoid confusion with electrical current, i . From now on we will use the symbol j .

Imaginary numbers can also be paired up with real numbers to form **complex numbers**. Such numbers have the form:

$$a + bj$$

where a represents the **real part** and b the **imaginary part**. This notation can be considered a shorthand for the more general statement:

$$(a, 0j) + (0, bj)$$

For convenience the 0 values are omitted and the notation shortened to $a + bj$. It is therefore important to realize that the ‘+’ notation doesn’t mean that a is added to b , this is not possible because a and b are quite different objects. The rules for the addition of complex numbers is different and will be described later.

Conjugate Pairs

A conjugate complex pair is given by the pair of complex numbers:

$$a + bj \quad a - bj$$

That is both numbers have the same real part, a , but the imaginary part, b , has the same magnitude but opposite sign.

Polar form

We can express a complex number on a two dimension plane where the horizontal axis represents the real part and the vertical axis the imaginary part. A complex number can therefore represented as a point on the plane.

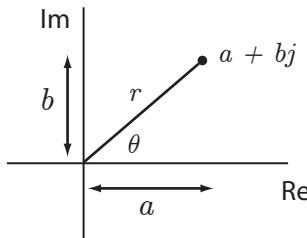


Figure F.2 Argand plane

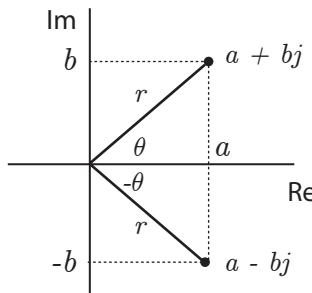


Figure F.3 Argand plane showing a conjugate pair.

The conjugate complex number can be visualized in polar form as shown in Figure F.3.

We can also express a complex number in terms of the distance the point is away from the origin and the angle it has with respect to the horizontal axis. In this way we can express the real and imaginary parts using trigonometric functions:

$$b = r \sin \theta \quad a = r \cos \theta$$

where r is the length of the line from the origin to the point and θ the angle. The following two representations are therefore equivalent.

$$a + bj = r(\cos \theta + j \sin \theta) \quad (\text{F.13})$$

When written like this, r is also known as the magnitude or modulus of the complex number, z , that is:

$$|z| = r = \sqrt{a^2 + b^2} \quad (\text{F.14})$$

The notation $|z|$ is often used to denote the magnitude of a complex number. Examples:

$$|3 - 4i| = \sqrt{3^2 + (-4)^2} = \sqrt{25} = 5$$

$$|-4i| = \sqrt{(-4)^2} = \sqrt{16} = 4$$

The angle, θ is known as the argument or phase and is given by:

$$\theta = \tan^{-1} \left(\frac{b}{a} \right) \quad (\text{F.15})$$

The argument of the complex number, z , is sometimes denoted by:

$$\arg(z)$$

In calculating the angle we must be careful about the sign of b/a . Figure F.4 illustrates the four possible situations. If the point is in the second quadrant, 180° should be added to the \tan^{-1} result. If the point is in the third quadrant, then 180° should be subtracted from the \tan^{-1} result. Angles computed in the 4th quadrant will be negative. The 1st quadrant needs no adjustment.

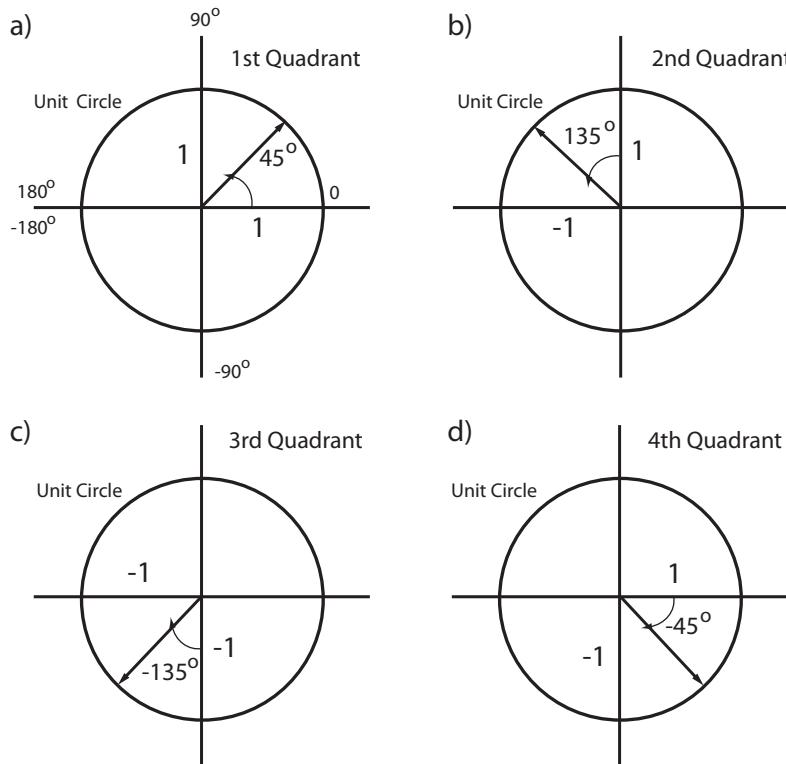


Figure F.4 a) Both axes are positive, $\arctan(1/1) = 45^\circ$; b) Horizontal axis is negative -1, $\arctan(1/-1) = +135^\circ$; c) Vertical axis is negative, $\arctan(-1/1) = -45^\circ$; d) Both axes are negative, $\arctan(-1/-1) = -135^\circ$

The rules for computing the angle are summarized in the list below. The atan2 function often found in software such as Matlab will usually automatically take into consideration the signs.

$$\text{atan2}(y, x) = \begin{cases} \tan^{-1}\left(\frac{y}{x}\right) & x > 0, y > 0 \\ -\tan^{-1}\left(\frac{y}{x}\right) & x > 0, y < 0 \\ \pi + \tan^{-1}\left(\frac{y}{x}\right) & y \geq 0, x < 0 \\ -\pi + \tan^{-1}\left(\frac{y}{x}\right) & y < 0, x < 0 \\ \frac{\pi}{2} & y > 0, x = 0 \\ -\frac{\pi}{2} & y < 0, x = 0 \\ \text{undefined} & y = 0, x = 0 \end{cases} \quad (\text{F.16})$$

Examples:

$$\text{atan2}(2/5) = 21.8^\circ$$

$$\text{atan2}(-2/5) = -21.8^\circ$$

$$\text{atan2}(-2/-5) = -158.18^\circ$$

$$\text{atan2}(2/-5) = 158.18^\circ$$

Example F.1

- a) If $\mathbf{a} = [4 \ 3]$, the length of the vector from the origin to the point $(4, 3)$ is:

$$l = \sqrt{4^2 + 3^2} = 5 \text{ units}$$

The angle of the vector relative to the x axis is:

$$\alpha = \tan^{-1}(3/4) = 36.87 \text{ degrees}$$

- b) If $\mathbf{a} = [-1, 2]$, find the angle relative to the x-axis.

Figure F.4 in the Appendix shows the various quadrants for computing angles. Given that the vector is $[-1, 2]$, this indicates that the vector is in the 2nd quadrant. The angle is therefore computed using:

$$180^\circ + \tan^{-1}(y/x) = 180^\circ + (-63.44) = 116.56^\circ$$

- c) If $\mathbf{a} = [-1, -3]$, find the angle relative to the x-axis.

Figure F.4 in the Appendix shows the various quadrants for computing angles. Given that the vector is $[-1, -3]$, this indicates that the vector is in the 3rd quadrant. The angle is therefore computed using:

$$-180^\circ + \tan^{-1}(y/x) = -180^\circ + 71.57 = 108.43^\circ$$

Basic Complex Arithmetic

Let $a + bj$ and $c + dj$ be complex numbers. Then:

1. $a + bj = c + dj$ if and only if $a = c$ and $b = d$ (i.e. the real parts are equal and the imaginary parts are equal)
2. $(a + bj) + (c + dj) = (a + c) + (b + d)j$ (i.e. add the real parts together and add the imaginary parts together)
3. $(a + bj) - (c + dj) = (a - c) + (b - d)j$
4. $(a + bj)(c + dj) = (ac - bd) + (ad + bc)j$
5. $(a + bj)(a - bj) = a^2 + b^2$
6. $\frac{a + bj}{c + dj} = \frac{(ac + bd) + (bc - ad)j}{c^2 + d^2}$

Additional Comments on Multiplication

As noted above multiplication of two complex numbers is given by:

$$(a + bj)(c + dj) = (ac - bd) + (ad + bc)j$$

We can compute the length of the product as:

$$\text{length} = \sqrt{(ac - bd)^2 + (ad + bc)^2}$$

Simplifying yields:

$$\text{length} = \sqrt{a^2 + b^2} \sqrt{c^2 + d^2}$$

That is the length of the product is the length of the two individual lengths multiplied together. Another way to see this is to multiply two complex numbers that are in polar form. For example, consider the two complex numbers:

$$\begin{aligned} z_1 &= r_1(\cos \theta_1 + j \sin \theta_1) \\ z_2 &= r_2(\cos \theta_2 + j \sin \theta_2) \end{aligned}$$

If we multiply these together we obtain initially:

$$z_1 z_2 = r_1 r_2 (\cos \theta_1 + j \sin \theta_1)(\cos \theta_2 + j \sin \theta_2)$$

Multiplying out the bracketed terms we obtain:

$$z_1 z_2 = r_1 r_2 ((\cos \theta_1 \cos \theta_2 - \sin \theta_1 \sin \theta_2) + j(\cos \theta_1 \sin \theta_2 + \cos \theta_2 \sin \theta_1))$$

By the summation trigonometry identity, these expressions can be simplified to:

$$z_1 z_2 = r_1 r_2 (\cos(\theta_1 + \theta_2) + j(\sin(\theta_1 + \theta_2)))$$

This expression is also in polar form where we see that the lengths have been multiplied $r_1 r_2$ and the angles have been added, $\theta_1 + \theta_2$. We can summaries these results using the expressions:

$$\begin{aligned} |zw| &= |z| |w| \\ \arg(zw) &= \arg(z) + \arg(w) \end{aligned}$$

Division of Complex Numbers

Division is accomplished by multiplying the top and bottom by the conjugate. Note that the product of a complex number and its conjugate gives a real number, this allows us to eliminate the imaginary part from the denominator.

$$\begin{aligned} \frac{a + bj}{c + dj} &= \frac{a + bj}{c + dj} \cdot \frac{c - dj}{c - dj} \\ &= \frac{(ac - b(-d)) + (a(-d) + bc)j}{c^2 + d^2} \\ &= \frac{(ac + bd) + (bc - ad)j}{c^2 + d^2} \\ &= \frac{ac + bd}{c^2 + d^2} + \frac{bc - ad}{c^2 + d^2}j \end{aligned}$$

In terms of the polar form division of two complex number is given by:

$$\frac{1}{z_2} = \frac{r_1}{r_2} (\cos(\theta_1 - \theta_2) + j(\sin(\theta_1 - \theta_2)))$$

Here we see that the final length is the division of the two original lengths and the new angle is the difference between the two original angles.

Exponential Form

By Euler's formula:

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

we can substitute the sine/cosine terms in the polar representation to give:

$$a + bj = r e^{j\theta}$$

This gives us three equivalent ways to represent a complex number:

$$a + bj = r(\cos(\theta) + j \sin(\theta)) = re^{j\theta}$$

This means that any complex number, z , can be expressed as a as length and an exponential term:

$$z = |z|e^{j\theta} \quad (\text{F.17})$$

de Moivre's Theorem

Equation F.13 showed how a complex number could be expressed in terms of sine and cosines (polar form):

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

The square of z is given by:

$$z^2 = r^2(\cos \theta + j \sin \theta)^2 = r^2(\cos^2 \theta - \sin^2 \theta + 2j \cos \theta \sin \theta) = r^2(\cos 2\theta + \sin 2\theta)$$

This result can be generalized to any integer n such that:

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

The above equation is called de Moivre's Theorem. The theorem has a number of applications, for example we can use de Moivre's theorem to evaluate $(1+j)^{16}$. First we determine the magnitude and angle for the complex number $(1+j)$. The magnitude is $\sqrt{1+1} = \sqrt{2}$. The angle is $\tan^{-1} 1/1 = \pi/4$. We can therefore write:

$$(1+j)^{16} = r^2(\cos(n\theta) + j \sin(n\theta)) = (\sqrt{2})^{16}(\cos(16\pi/4) + j \sin(16\pi/4)) = 256(1+0) = 256$$

Finding Roots

Of more interest to the work in this book is using de Moivre's theorem to find the roots of complex numbers. For example, let us find the roots to $\sqrt[3]{27j}$. First we write $27j$ in polar form. The magnitude $r = \sqrt{0^2 + (27)^2} = 27$. The angle is given by $\tan^{-1} 27/1 = \pi/2$. We can therefore write $27j$ as:

$$27j = 27 \left(\cos \frac{\pi}{2} + j \sin \frac{\pi}{2} \right)$$

Since we are looking for $\sqrt[3]{27j}$ we can also look at this term in the following equivalent form: $z^3 = 27j$. By de Moire's theorem:

$$r^3(\cos 3\theta + j 3\theta) = 27j = 27 \left(\cos \frac{\pi}{2} + j \sin \frac{\pi}{2} \right)$$

What possible values are there for θ ? It is true that:

$$\cos 3\theta = \cos \frac{\pi}{2} \quad \text{and} \quad \sin 3\theta = \sin \frac{\pi}{2}$$

$\cos 3\theta$ will repeat itself every 2π . Therefore it must be true that:

$$3\theta = \frac{\pi}{2} + 2\pi k$$

where k is an integer. Since there are three possible roots we can set k to 0, 1, and 2 and evaluate the roots in turn.

$k = 0$, $3\theta = \pi/2$, therefore $\theta = \pi/6$

$$\begin{aligned} z_1 &= 3\left(\cos \frac{\pi}{6} + j \sin \frac{\pi}{6}\right) \\ &= 3\left(\frac{\sqrt{3}}{2} + j \frac{1}{2}\right) \\ &= \frac{3\sqrt{3}}{2} + j \frac{3}{2} \end{aligned}$$

$k = 1$, $3\theta = \pi/2 + 2\pi(1)$, therefore $\theta = 5\pi/6$

$$\begin{aligned} z_1 &= 3\left(\cos \frac{5\pi}{6} + j \sin \frac{5\pi}{6}\right) \\ &= 3\left(\frac{-\sqrt{3}}{2} + j \frac{1}{2}\right) \\ &= -\frac{3\sqrt{3}}{2} + j \frac{3}{2} \end{aligned}$$

$k = 2$, $3\theta = \pi/2 + 2\pi(2)$, therefore $\theta = 9\pi/6 = 3\pi/2$

$$\begin{aligned} z_1 &= 3\left(\cos \frac{3\pi}{2} + j \sin \frac{3\pi}{2}\right) \\ &= 3(0 + j(-1)) \\ &= -3j \end{aligned}$$

Setting $k = 3$ gives the same result as $k = 0$, setting $k = 4$ gives the same result as $k = 1$ and the therefore the three unique cube roots of $-27j$ are:

$$z_1 = \frac{3\sqrt{3}}{2} + j \frac{3}{2}$$

$$z_2 = -\frac{3\sqrt{3}}{2} + j \frac{3}{2}$$

$$z_3 = -3j$$

In general the n^{th} root of z is given by:

$$\sqrt[n]{z} = \sqrt[n]{r} \left(\cos\left(\frac{\theta + 2\pi k}{n}\right) + j \sin\left(\frac{\theta + 2\pi k}{n}\right) \right)$$

for $k = 0, 1, 2, \dots, n - 1$

F.7 Eigenvalues and Eigenvectors

Eigenvalues and eigenvector play a prominent role when solving differential equations and determining the stability and dynamics of linear dynamical systems. As such they constitute an important part of the linear algebra toolkit in systems biology. In this chapter we will briefly review what they are and describe some of their properties.

Definition

A square matrix such as A can be used to transform a given vector, v in specific ways. For example, if the matrix A is:

$$\begin{bmatrix} 2 & 0 \\ 0 & 4 \end{bmatrix}$$

then the result of multiplying A into v will yield a vector that is similar to v but where the first element is scaled by 2 and the second element by 4.

For an arbitrary square matrix, if it is possible to find a vector v such that when we multiply the vector by A we get a scaled version of v , then we call the vector v the **eigenvector** of A and the scaling value, the **eigenvalue** of A . In other literature eigenvalues are also called latent roots, characteristic roots or proper values. For a matrix of dimension n , there will be at most n eigenvalues and n eigenvectors. In the case of the simple example above the eigenvalues must be 2 and 4 respectively while the two eigenvectors will be:

$$\begin{bmatrix} \alpha \\ 0 \end{bmatrix} \quad \begin{bmatrix} 0 \\ \alpha \end{bmatrix}$$

Definition:

For an $n \times n$ matrix A , a scalar λ is called an **eigenvalue** of A if there is a non-zero vector, v , called an **eigenvector** such that:

$$A v = \lambda v$$

Computing Eigenvalues and Eigenvectors

A method for computing eigenvalues and eigenvectors can be devised if we rearrange equation $bA\mathbf{v} = \lambda I\mathbf{v}$ as follows:

$$\begin{aligned} A\mathbf{v} &= \lambda I\mathbf{v} \\ A\mathbf{v} - \lambda I\mathbf{v} &= 0 \\ (A - \lambda I)\mathbf{v} &= 0 \end{aligned}$$

From linear algebra we know that there will be non-zero solutions to $(A - \lambda I)\mathbf{v} = 0$ if $\det(A - \lambda I) = 0$. We can use this observation to compute the eigenvalues and eigenvectors of a matrix. For example consider the matrix:

$$\begin{bmatrix} 3 & 6 \\ 1 & 4 \end{bmatrix}$$

Computing $A - \lambda I$ yields:

$$\begin{aligned} A - \lambda I &= \begin{bmatrix} 3 - \lambda & 6 \\ 1 & 4 - \lambda \end{bmatrix} \\ \det(A - \lambda I) &= (3 - \lambda)(4 - \lambda) - 6 \\ &= \lambda^2 - 7\lambda + 6 \\ &= (\lambda - 6)(\lambda - 1) \end{aligned}$$

The eigenvalues are therefore 6 and 1. With two eigenvalues there will be two eigenvectors. First we consider $\lambda = 6$.

$$\begin{aligned} (A - \lambda I)\mathbf{v} &= 0 \\ \left(\begin{bmatrix} 3 & 6 \\ 1 & 4 \end{bmatrix} - \begin{bmatrix} 6 & 0 \\ 0 & 6 \end{bmatrix} \right) \mathbf{v} &= 0 \\ \begin{bmatrix} -3 & 6 \\ 1 & -2 \end{bmatrix} \mathbf{v} &= 0 \end{aligned}$$

By inspection we can see that the eigenvector is:

$$\begin{bmatrix} 2 \\ 1 \end{bmatrix}$$

satisfied this equation. Likewise we can do the same for the other eigenvalue, $\lambda = 1$ where the corresponding eigenvector is found to be:

$$\begin{bmatrix} -3 \\ 1 \end{bmatrix}$$

G

Common Integrals

$$\begin{aligned}\int adx &= ax \\ \int (u + v)dx &= \int udx + \int vdx \\ \int udv &= uv - \int vdu\end{aligned}$$

Computing a definite integral:

$$\int_a^b f(x) dx = F(b) - F(a)$$

Example:

$$\begin{aligned}\int_0^\infty e^{-st} dt &= \left[-\frac{1}{s}e^{-st} \right]_0^\infty \\ &= \left[-\frac{1}{s}e^{-\infty} \right] - \left[-\frac{1}{s}e^0 \right] = \frac{1}{s}\end{aligned}$$

$$\int k \, dx = kx \quad (\text{G.1})$$

$$\int x \, dx = \frac{1}{2}x^2 \quad (\text{G.2})$$

$$\int \frac{1}{x^2} \, dx = -\frac{1}{x} \quad (\text{G.3})$$

$$\int x^n \, dx = -\frac{1}{n+1}x^{n+1} \quad n \neq -1 \quad (\text{G.4})$$

$$\int \frac{1}{x} \, dx = \ln|x| \quad (\text{G.5})$$

$$\int e^{ax} \, dx = \frac{1}{a}e^{ax} \quad (\text{G.6})$$

$$\int \frac{1}{(x+a)^2} \, dx = -\frac{1}{x+a} \quad (\text{G.7})$$

$$\int \frac{1}{1+x^2} \, dx = \tan^{-1} x \quad (\text{G.8})$$

$$\int \sin(ax) \, dx = -\frac{1}{a}\cos(ax) \quad (\text{G.9})$$

$$\int \cos(ax) \, dx = \frac{1}{a}\sin(ax) \quad (\text{G.10})$$

H

Table of Laplace Transforms

<hr/> <hr/>		
	Function: $f(t)$	Laplace Function: $F(s)$
Scaling	$af(t)$	$aF(s)$
Linearity	$a_1 f_1(t) \pm a_2 f_2(t)$	$a_1 F_1(s) \pm a_2 F_2(s)$
Unit Step	1	$\frac{1}{s}$
Delay	$u(t - a)$	$\frac{e^{-as}}{s}$
Ramp	t	$\frac{1}{s^2}$
Impulse	$\delta(t)$	1
Delayed Impulse	$\delta(t - t_0)$	e^{-st_0}
Exponential	$e^{at} f(t)$	$F(s - a)$

Basic Operations:

	Function: $f(t)$	Laplace Transform: $F(s)$	Remarks
1.	$f(t)$	$\int_0^\infty e^{-st} f(t) dt$	Transform
2.	$a f(t)$	$a F(s)$	Scaling
3.	$a_1 f_1(t) \pm a_2 f_2(t)$	$a_1 F_1(s) \pm a_2 F_2(s)$	Linearity
4.	$f'(t)$	$sF(s) - f(0)$	Derivative
5.	$f''(t)$	$s^2 F(s) - sf(0) - f'(0)$	Second derivative
6.	$f^n(t)$	$s^n F(s) - s^{(n-1)} f(0) - \cdots - f^{(n-1)}(0)$	n^{th} Derivative
7.	$\int_0^t f(\tau) d\tau$	$\frac{F(s)}{s}$	Integral
8.	$\int_0^t f(x)g(t-x) dx$	$F(s)G(s)$	Convolution
9.	$t^n f(t)$	$(-1)^n \frac{d^n F(s)}{ds^n}$	Power
10.	$f(at)$	$\frac{1}{a} f\left(\frac{s}{a}\right)$	Time scaling
11.	$e^{at} f(t)$	$F(s-a)$	1st shifting theorem
12.	$f(t-a)u(t-a)$	$e^{-as} F(s)$	2nd shifting theorem
13.		$\lim_{t \rightarrow 0} f(t) = \lim_{s \rightarrow \infty} sF(s)$	Initial value theorem
14.		$\lim_{t \rightarrow \infty} f(t) = \lim_{s \rightarrow 0} sF(s)$	Final value theorem

Common Transforms:

	Function: $f(t)$	Laplace Transform: $F(s)$	Remarks
15.	1	$\frac{1}{s}$	Unit Step
16.	$u(t - a)$	$\frac{e^{-as}}{s}$	Delay
17.	at	$\frac{a}{s^2}$	Ramp
18.	t^2	$\frac{2}{s^3}$	Parabola
19.	t^n	$\frac{n!}{s^{n+1}}$	Power (n is an integer)
20.	$\frac{1}{(n-1)!}t^{n-1}$	$\frac{1}{s^n}$	Power (n is an integer)
21.	$\frac{t^n}{n!}$	$\frac{1}{s^{n+1}}$	
22.	$\delta(t)$	1	Impulse
23.	$\delta(t - t_o)$	e^{-st_o}	Delayed Impulse
24.	e^{at}	$\frac{1}{s - a}$	Exponential
25.	e^{-at}	$\frac{1}{s + a}$	
26.	te^{at}	$\frac{1}{(s - a)^2}$	
27.	$t^n e^{at}$	$\frac{n!}{(s - a)^{n+1}}$	

	Function: $f(t)$	Laplace Function: $F(s)$	Remarks
28.	$\sin at$	$\frac{a}{s^2 + a^2}$	Trigonometric
29.	$\sin(at + \theta)$	$\frac{s \sin \theta + a \cos \theta}{s^2 + a^2}$	
30.	$\cos at$	$\frac{s}{s^2 + a^2}$	
31.	$\sinh at$	$\frac{a}{s^2 - a^2}$	
32.	$\cosh at$	$\frac{s}{s^2 - a^2}$	
33.	$t \sin at$	$\frac{2as}{(s^2 + a^2)^2}$	
34.	$t \cos at$	$\frac{s^2 - a^2}{(s^2 + a^2)^2}$	
35.	$t \sinh at$	$\frac{2as}{(s^2 - a^2)^2}$	
36.	$t \cosh at$	$\frac{s^2 + a^2}{(s^2 - a^2)^2}$	
37.	$e^{at} \sin bt$	$\frac{a}{(s - a)^2 + b^2}$	
38.	$e^{at} \cos bt$	$\frac{s - a}{(s - a)^2 + b^2}$	
39.	$e^{at} t^n$	$\frac{n!}{(s - a)^{n+1}}$	
40.	$e^{bt} \cos at$	$\frac{s - b}{(s - b)^2 - a^2}$	
41.	$\frac{1}{a}(a \cos(at) - \sin(at))$	$\frac{s - 1}{s^2 + a^2}$	

	Function: $f(t)$	Laplace Function: $F(s)$	Remarks
42.	$\frac{a}{b}(1 - e^{-bt})$	$\frac{a}{s(s + b)}$	
43.	$\frac{e^{at} - e^{bt}}{a - b}$	$\frac{1}{(s - a)(s - b)}$	
44.	$\frac{ae^{at} - be^{bt}}{a - b}$	$\frac{s}{(s - a)(s - b)}$	
45.	$\frac{1}{b}e^{-at} \sin(bt)$	$\frac{1}{(s + a)^2 + b^2}$	
46.	$\frac{1}{\sqrt{\pi t}}$	$\frac{1}{\sqrt{s}}$	
47.	$\frac{1}{\sqrt{\pi t}}e^{-a^2/4t}$	$\frac{e^{-a\sqrt{s}}}{\sqrt{s}}$	
48.	$\frac{a}{2\sqrt{\pi t^3}}e^{-a^2/4t}$	$e^{-a\sqrt{s}}$	
49.	$\operatorname{erfc}\left(\frac{a}{2\sqrt{t}}\right)$	$\frac{e^{-a\sqrt{s}}}{s}$	
50. Saw tooth	$[u(t) - u(t - a)]\frac{t}{a}$	$\frac{1}{as^2} - \frac{e^{-as}}{s(1 - e^{-as})}$	
51. Triangle Wave	$\frac{1}{a}[u(t)t - 2u(t - a)(t - a) + u(t - 2a)(t - 2a)]$	$\frac{1}{as^2} \tanh \frac{as}{2}$	
52. Square Wave	$u(t) - 2u(t - a) + u(t - 2a)$	$\frac{1}{s} \tanh \frac{as}{2}$	

References

- [1] Aris, R. 1965. Arch. Rational Mech. Anal **19**:81–99.
- [2] Black, H. 1977. IEEE spectrum **14** (12):55–61.
- [3] Bliss, R. D., P. R. Painter, and A. G. Marr. 1982. J Theor Biol **97** (2):177–193.
- [4] Boahen, K. 2005. Scientific American **292** (5):56–63.
- [5] Bode, A. M., and Zigang Dong. 2004. Nat Rev Cancer **4** (10):793–805.
- [6] Brilli, M., Marco Fondi, Renato Fani, Alessio Mengoni, Lorenzo Ferri, Marco Bazzicalupo, and Emanuele Biondi. 2010. BMC Systems Biology **4** (1):52.
- [7] Chen, K. C., L Calzone, A Csikasz-Nagy, F R Cross, B Novak, and J J Tyson. 2004. Mol Biol Cell **15** (8):3841–3862.
- [8] Chickarmane, V., B. N. Kholodenko, and H. M. Sauro. 2007. J Theor Biol **244** (1):68–76.
- [9] Chickarmane, V., C. Troein, U. A. Nuber, H. M. Sauro, and C. Peterson. 2006. PLoS Comput Biol **2** (9).
- [10] Cohen, P. 2000. Trends in Biochemical Sciences **25** (12):596–601.
- [11] Deckard, A., and H. M. Sauro. 2004. Chembiochem **5**:1423–31.
- [12] Dormand, J. R., and Peter J Prince. 1980. Journal of computational and applied mathematics **6** (1):19–26.
- [13] Elowitz, M. B., and S Leibler. 2000. Nature **403**:335–338.
- [14] Fall, C. P., E. S. Marland, J. M. Wagner, and J. J. Tyson. 2002. *Computational Cell Biology*. Springer-Verlag, New York.
- [15] Field, R. J., and R. M. Noyes. 1974. J. Chemical Physics **60**(5):1877–1884.
- [16] FitzHugh, R. 1955. Bull. Math. Biophysics **17**:257–278.

- [17] Gama-Castro, S., Verónica Jiménez-Jacinto, Martín Peralta-Gil, Alberto Santos-Zavaleta, Mónica I Peñaloza-Spinola, Bruno Contreras-Moreira, Juan Segura-Salazar, Luis Muñiz-Rascado, Irma Martínez-Flores, Heladia Salgado, César Bonavides-Martínez, Cei Abreu-Goodger, Carlos Rodríguez-Penagos, Juan Miranda-Ríos, Enrique Morett, Enrique Merino, Araceli M Huerta, Luis Treviño-Quintanilla, and Julio Collado-Vides. 2008. Nucleic Acids Res **36** (Database issue):D120–D124.
- [18] Gardner, T. S., C. R. Cantor, and J. J. Collins. 2000. Nature **403**:339–342.
- [19] Gardner, T. S., and J. J. Collins. 2000. Nature **405**:520–521.
- [20] Gear, C. W. 1971. *Numerical Initial Value Problems in Ordinary Differential Equations*. Prentice-Hall, Inc. Englewood Cliffs, New Jersey.
- [21] Gekas, V., and M. Lopez-Leiva. 1985. Process biochemistry **20** (1):2–12.
- [22] Geva-Zatorsky, N., N Rosenfeld, S Itzkovitz, R Milo, A Sigal, E Dekel, T Yarnitzky, Y Liron, P Polak, G Lahav, and U Alon. 2006. Mol Syst Biol **2**:2006–2006.
- [23] Goldbeter, A. 1997. *Biochemical Oscillations and Cellular Rhythms: The Molecular Bases of Periodic and Chaotic Behaviour*. Cambridge University Press.
- [24] Goldbeter, A., and D. E. Koshland. 1984. J. Biol. Chem. **259**:14441–7.
- [25] Goodwin, B. 1965. Advances in Enzyme Regulation **3**:425–438.
- [26] Higgins, J. 1967. Ind. Eng. Chem. **59**(5):18–62.
- [27] Hoffmann, A., A. Levchenko, M. L. Scott, and D. Baltimore. 2002. Science **298**:1241–1245.
- [28] Hofmeyr, J.-H. 1995. J Bioenerg Biomembr **27** (5):479–490.
- [29] Hucka, M., A. Finney, H. M. Sauro, H. Bolouri, J. C. Doyle, H. Kitano, A. P. Arkin, B. J. Bornstein, D. Bray, A. Cornish-Bowden, A. A. Cuellar, S. Dronov, E. D. Gilles, M. Ginkel, V. Gor, I. I. Goryanin, W. J. Hedley, T. C. Hodgman, J. H. Hofmeyr, P. J. Hunter, N. S. Juty, J. L. Kasberger, A. Kremling, U. Kummer, N. Le NovÃÆŠÃÂtre, L. M. Loew, D. Lucio, P. Mendes, E. D. Mjolsness, Y. Nakayama, M. R. Nelson, P. F. Nielsen, T. Sakurada, J. C. Schaff, B. E. Shapiro, T. S. Shimizu, H. D. Spence, J. Stelling, K. Takahashi, M. Tomita, J. Wagner, and J. Wang. 2003. Bioinformatics **19**:524–531.
- [30] Huerta, A. M., H. Salgado, D. Thieffry, and J. Collado-Vides. 1998. Nucleic Acids Res **26** (1):55–59.
- [31] Jones, M. H. 1977. *A practical introduction to electronic circuits*. Cambridge University Press, Cambridge.

- [32] Kanwal, R. P. 2011. *Generalized functions: theory and applications.* Springer Science & Business Media.
- [33] Kholodenko, B. N. 2000. Eur. J. Biochem **267**:1583–1588.
- [34] Lahav, G., N. Rosenfeld, A. Sigal, N. Geva-Zatorsky, A. J. Levine, M. B. Elowitz, and U. Alon. 2004. Nature, Genetics **36**(2):147–150.
- [35] Lotka, A. J. 1920. J. Am. Chem. Soc. **42**:1595–1599.
- [36] Macek, B., F. Gnad, B. Soufi, C. Kumar, J.V. Olsen, I. Mijakovic, and M. Mann. 2008. Molecular & Cellular Proteomics **7** (2):299.
- [37] Manning, G., D. B. Whyte, R. Martinez, T. Hunter, and S. Sudarsanam. 2000. Science **298**:1912–1934.
- [38] Mendes, P. 1993. Comput. Applic. Biosci. **9**:563–571.
- [39] Mindell, D. 2000. Technology and Culture **41** (3):405–434.
- [40] Monod, J., J Wyman, and J. P. Changeux. 1965. J Mol Biol **12**:88–118.
- [41] Nicolis, G. 1971. Adv. Chem. Phys **19**:209–324.
- [42] Olivier, B. G., J. M. Rohwer, and J. H. Hofmeyr. 2005. Bioinformatics **21**:560–1.
- [43] Paladugu, S., V. Chickarmane, A. Deckard, J.P. Frumkin, M. McCormack, and H.M. Sauro. 2006. IEE Proceedings - Systems Biology **153**:223–235.
- [44] Pomerening, J. R., E D Sontag, and J E Ferrell. 2003. Nat Cell Biol **5** (4):346–351.
- [45] Ptacek, J., G. Devgan, G. Michaud, H. Zhu, X. Zhu, J. Fasolo, H. Guo, G. Jona, A. Breitkreutz, R. Sopko, *et al.*. 2005. Nature **438** (7068):679–684.
- [46] Ptacek, J., and M. Snyder. 2006. Trends in Genetics **22** (10):545–554.
- [47] R, J. F., E. Koros, and R. M. Noyes. 1972. Journal of the American Chemical Society **94**:8649—8664.
- [48] Sauro, H. M. 1993. Comput Appl Biosci **9** (4):441–450.
- [49] Sauro, H. M. 2000. In: Hofmeyr, J.-H. S., J. M. Rohwer, and J. L. Snoep. , (ed.), *Animating the Cellular Map: Proceedings of the 9th International Meeting on BioThermoKinetics*, . Stellenbosch University Press.
- [50] Sauro, H. M. 2011. *Enzyme Kinetics for Systems Biology*. Ambrosius Publishing. First Edition.

- [51] Sauro, H. M. 2014. *Systems Biology: An Introduction to Pathway Modeling*. Ambrosius Publishing.
- [52] Sauro, H. M., Totte T Karlsson, Maciej Swat, Michal Galdzicki, and Andy Somogyi. 2013. bioRxiv .
- [53] Sauro, H. M., and B. N. Kholodenko. 2004. Prog Biophys Mol Biol. **86**:5–43.
- [54] Savageau, M. A. 1975. J Mol Evol **5** (3):199–222.
- [55] Savageau, M. A. 1976. *Biochemical systems analysis: a study of function and design in molecular biology*. Addison-Wesley, Reading, Mass.
- [56] Seshasayee, A. S. N., P. Bertone, G. M. Fraser, and N.M. Luscombe. 2006. Current Opinion in Microbiology **9** (5):511–519.
- [57] Shen-Orr, S. S., R. Milo, S. Mangan, and U. Alon. 2002. Nature Genetics **31**:64–68.
- [58] Somogyi, E. T., Jean-Marie Bouteiller, James A. Glazier, Matthias König, J. Kyle Medley, Maciej H. Swat, and Herbert M. Sauro. 2015. Bioinformatics (Oxford, England) .
- [59] Thron, C. 1991. Bulletin of mathematical biology **53** (3):403–424.
- [60] Toledo, F., and Geoffrey M Wahl. 2006. Nat Rev Cancer **6** (12):909–923.
- [61] Trafton, A. 2012. <http://www.mit.edu/newsoffice/2011/brain-chip-1115.html>.
- [62] Tyson, J., and H. G. Othmer. 1978. Progress in Theoretical Biology (R. Rosen & F.M. Snell, Eds.) **5**:1–62.
- [63] Tyson, J. J. 1975. J. Chem. Phys **62**:1010–1015.
- [64] Tyson, J. J., K. C. Chen, and B. Novak. 2003. Current Opinion in Cell Biology **15**:221–231.
- [65] van der Pol, B., and J. van der Mark. 1928. Philosophical Magazine Supplement **6**:763–775.
- [66] Wikipedia 2013a. http://en.wikipedia.org/wiki/Analog_computer. [Online; accessed 4-January-2013].
- [67] Wikipedia 2013b. http://en.wikipedia.org/wiki/Antikythera_mechanism. [Online; accessed 4-January-2013].
- [68] Wikipedia 2013c. http://en.wikipedia.org/wiki/Cellular_automaton. [Online; accessed 4-January-2013].
- [69] Wikipedia 2013d. http://en.wikipedia.org/wiki/Difference_engine. [Online; accessed 4-January-2013].

- [70] Wikipedia 2013e. http://en.wikipedia.org/wiki/Differential_analyser. [Online; accessed 4-January-2013].
- [71] Wikipedia 2013f. <http://en.wikipedia.org/wiki/Emergence>. [Online; accessed 4-January-2013].
- [72] Wikipedia 2013g. <http://en.wikipedia.org/wiki/Fractal>. [Online; accessed 4-January-2013].
- [73] Wikipedia 2013h. http://en.wikipedia.org/wiki/MONIAC_Computer. [Online; accessed 4-January-2013].
- [74] Wikipedia 2013i. http://en.wikipedia.org/wiki/Neural_networks. [Online; accessed 4-January-2013].
- [75] Wikipedia 2013j. <http://en.wikipedia.org/wiki/Rangekeeper>. [Online; accessed 4-January-2013].
- [76] Wikipedia 2013k. http://en.wikipedia.org/wiki/Slide_rule. [Online; accessed 4-January-2013].
- [77] Wikipedia 2013l. <http://en.wikipedia.org/wiki/SPICE>. [Online; accessed 7-January-2013].
- [78] Wikipedia 2013m. http://en.wikipedia.org/wiki/Tide_predicting_machine. [Online; accessed 4-January-2013].
- [79] Wikipedia 2013n. <http://en.wikipedia.org/wiki/V-2a>. [Online; accessed 4-January-2013].
- [80] Woods, J. H., and H. M. Sauro. 1997. *Comput Appl Biosci* **13** (2):123–130.
- [81] Yi, T., Y. Huang, M. I. Simon, and J. Doyle. 2000. *Proc. Natl. Acad. Sci. USA*. **97**:4649—4653.

History

See corrections at front of book.

Index**Symbols**

K_a	77
K_d	77
K_m	500
V_m	500
\mathbf{p}	9
$\mathbf{u}(t)$	10
$\mathbf{x}(t)$	11
$\mathbf{y}(t)$	12
ζ	360

A

acceleration error constant	416
accuracy	412
addition of signals	140
additivity	162
adenine nucleotides	69
allosteric constant	504
allostery	505
amplitude	149, 289
analytical solutions	29
angular frequency	149
antimony	44
apparent kinetic order	80
approximations	32
association constant	77
average: Fourier series	267

B

bandwidth	292
behavior	33
bifurcation plot	446
biochemical model	186
biochemical systems theory	80

Biomodels	46
bistable	441
block diagrams	27, 256
Bode plot	251
Bode Plot: drawing	294
Bode plots	286
boundary variables	10
Briggs-Haldane	500
buffer circuit	21

C

cAMP	63
canonical transfer function	386
CAP	63
causal models	16
chaos	42
characteristic equation	250, 362
characteristic polynomial	250
chemical equilibrium	76
clamp	10
classification of models	15
closed loop gain	53
closed system	5
common integrals	517
competitive	502
competitive inhibition	501
complete solution	191
complex exponential	155
complex exponentials	270
complex number	323
complex numbers	521
computer exchange format	71
conjugate Pair	323
conjugate pairs	522

constants	9	enzyme action	499		
continuous variables	14	enzyme elasticity	84		
control matrix	179	enzyme inhibitor complex	502		
control theory, what is	47	enzyme kinetics	499		
convergence, Laplace	208	enzyme-reactant complex	499		
convolution	237	equilibrium constant	76, 77		
convolution, explanation	241	Euler's theorem	155		
cooperativity	503	Euler, software	46		
critically damped	363	even functions	268		
Ctesibius	48	exclusive model	504		
cut-off point	292	exponential	153		
CVODE	41	exponential order	209		
D					
damped natural frequency	363	exponential series	198		
damped oscillations	363	extensive property	74		
damping ratio	360	extent of reaction	73		
decibels	286	F			
definite integral	517	fast Fourier transform	275		
delayed signals	137	feed-forward matrix	179		
dependent variable	10	FFT	275		
derivative control	435	fidelity	55		
deterministic model	14	final-value theorem	254		
dimensional analysis	31	First Time Shifting Theorem	218		
dimensions and units	31	first-order system	191		
Dirichlet conditions	269	first-order system: Bode plot	288		
discrete Fourier transform	274	first-order systems	256, 288		
discrete variables	14	Fourier Coefficients	264		
disequilibrium ratio	77	Fourier series	263		
dissociation constant	77	Fourier series, aperiodic	270		
Dormand-Prince method	40	Fourier series, exponential form	270		
dynamic models	15	Fourier Transform	284		
E					
eigenvalues	530	Fourier transform, discrete data	274		
eigenvectors	530	fractional amounts	72		
elasticity	445	frequency response	279, 283		
elasticity coefficient	80	frequency spectrum	270		
elasticity rules	85	functional module	54		
electrical model	184	functional parts	79		
electrical models	16	fundamental frequency	264		
G					
gain	53				

gain factor	250
gain margin	333
gain, feedback.....	411
Gear.....	41
gene expression	63
general solutions	195
generic feedback.....	51
global error	471
governor.....	49
GPL.....	42
GSL.....	42

H

Haldane relationship	501
harmonic components	264
harmonics	151, 264
Harold Black	50
heaviside function	136
Heron	49
high input impedance	24, 56
Hill equations	506
homogeneity	162
homogenous system	195

I

improper integral	518
impulse	144
independent variable	11
initial conditions	11
initial rate	500
inputs	10, 135
integral control	434
intensive property	74
inverse transform	208
inverse transforms	220
irreversible bistability	448
isolated system	5

J

Jacobian	179, 180
James Watt	49

Jarnac	46
Julia	41

K

Kacser	63
kinetic mechanism	499
kinetic order	79
Kirchhoff's voltage law	353

Ktesibios

L

lac repressor	63
Laplace Transform Table	215
Laplace transform: Fourier series	273
Laplace transforms	207
Laplace, change of scale	217
Laplace, common signals	228
Laplace, exponential	213
Laplace, impulse	213
Laplace, inverse	220
Laplace, linearity	215
Laplace, ramp	212
Laplace, scaling	215
libRoadRunner	44
line spectrum	271
linear systems	161, 162
linear systems and sinusoids	280
Linear Time-Invariant System	164
linear/nonlinear models	16
linearization	55, 166
linearize ODE	170
linearize system of ODEs	171
loading	20
log form	81
logarithmic scales	82
loop gain	53
Lorenz attractor	42
Low output impedance	56
low output impedance	23
LTI system	164
lumped models	16

M

- mass-action kinetics 75, 82
 mass-action ratio 77
 Mathematica 41, 87
 Matlab 40
 Matplotlib 44
 matrix exponential 196, 243
 maximal velocity 500
 mechanical model 182
 metabolic pathways 64
 Michaelis-Menten kinetics 499
 model variables 8
 model, definition 5
 MWC model 504

N

- natural frequency 360
 negative cooperativity 503
 negative feedback, op amps 25
 Newton algorithm 116
 no damping 364
 non-Homogeneous system 199
 non-inverting Op Amp 25
 Numerical Recipes 46
 NumPy 43

O

- odd and even functions 268
 odd harmonics 267
 ODE, linearize 170
 ode15s 40
 ode45 40
 op amp, gain 25
 op amps 21
 open loop gain 53
 open system 5
 open-loop 414
 operational definition 80
 operator site 63, 439
 order 250
 Oscill8 449

- output matrix 179
 outputs 11
 overdamped 362

P

- parameters 9
 partial fractions 221
 peak time 367
 percentage overshoot 367
 period 149
 periodic 150, 263
 periodic behavior 320
 periodic solutions 322
 pharmokinetic 185
 phase 149
 phase margin 333
 phase shift 292
 Philon 49
 PID controller 408, 434
 Pneumatica 49
 polar form 281, 522
 pole-zero plot 251, 254
 poles 250
 polynomials 513
 position error constant 413
 positive cooperativity 503
 positive feedback loop 439
 positive feedback: stability 443
 power spectrum 270
 product inhibition 500, 502
 proportional change 82
 proportional control 408
 protein networks 65
 pulse train 141
 pulses 140
 pure competitive inhibitor 502
 Python 41, 43

Q

- quadratic function 520
 quasi-equilibrium 107

R

R state 504
radians 515
ramp 142
rate constant 76
rate of change 72
rate of reaction 74
rational fractions 72
reaction kinetics 72
reaction order 79
reaction rate 73
rectangular wave 266
regulator 51
RegulonDB 64
relaxed 504
resolvent 243
resonance peak 368
resonant frequency 368
response metrics 365
response of linear systems 189
reversible 76
reversible rate law 501
rise time 365
robustness 56
roots, polynomial 513
Routh-Hurwitz 514
Routh-Hurwitz criterion 323
Runge-Kutta method 40

S

saddle node 321
Sage 41
sawtooth wave 267
SBML 44, 71
scaled transfer functions 391
scaling, unit step 137
Scilab 41
SciPy 43
scipy: FFT 275
Second Time Shifting Theorem 219

second-order: freq response 368
second-order system 359
semilog graph paper 299
sensitivity 55
servomechanism 51
settling time 367
shifting 137
sifting 146
sigmoid response 503
signal processing 275
sinusoidal 149, 264
sinusoidal input 280
sinusoids 280
software: bifurcation 446
solving ODEs 225
spiral trajectories 322
stable node 320
standard form: complex number 285
standard form: first-order system 347
standard form: linear differential equation 349
standard form: second-order system 359
standard form: transfer function 251, 295, 299
state equation 242
state matrix 179
state space 177
state space equations 180
state space, linear 181
state transition matrix 197, 243
state variables 11
static models 15
steady state 33, 109
steady-state error 412
stochastic model 14
stoichiometric amount 69
stoichiometric amounts 69
stoichiometric coefficient 71
stoichiometric coefficients 70
stoichiometric networks 68
subelasticity terms 86
substrate elasticity 83

- sum of sinusoids 152
sundials 41
superposition 162, 193
system, definition 5

T

- T state 504
Taylor series 154, 519
Tellurium 44
tense 504
thermodynamic equilibrium 33, 109
Thomas Mead 49
time constant 153, 353
time invariance 164
time invariant models 16
Torrielli's Law 2
total response 200
total solution 191
transcription factor 439
transcription factors 63
transfer function 249
transient state 33
truncation error 470
two gene cascade 369
type of a system 414
Tyson 448

U

- uncompetitive inhibition 502
underdamped 363
unit step 136
unit step input 362
unstable node 320

V

- velocity error constant 416
volume 75

W

- water tank model 2

Y

- Yorick 46

Z

- zero frequency response 307
zero-input response 191, 200
zero-order 83
zero-state response 200, 235
zeros 250