Digital Control Systems

Christopher Nielsen
Control Systems Group
Department of Electrical & Computer Engineering

Acknowledgments

These notes were created for an introductory undergraduate courses on digital control systems at the University of Waterloo. They are based on various references cited within as well as course notes written by Prof. Daniel E. Davison, Prof. Daniel E. Miller at the University of Waterloo and Prof. Manfredi Maggiore, Prof. Bruce Francis at the University of Toronto. Rollen D'Souza kindly reviewed an earlier version of these notes and provided valuable feedback. All errors, typographical or otherwise, are solely mine.

Christopher Nielsen
University of Waterloo

Contents

Notation

$\mathbb{R}[s]$	Ring of polynomials in the complex variable s with real coefficients.	8
$:=$	Equal by definition.	8
$\delta(t)$	Continuous-time impulse.	7
$\overline{\mathbb{C}}^+$	Closed right half complex plane.	7
$\mathbb{R}(s)$	Rational functions in $s \in \mathbb{C}$ with coefficients in $\mathbb{R}$.	20
$\mathscr{Z}(x)$	z-transform of a signal $x[k]$.	117
$\operatorname{adj}(\cdot)$	Adjoint matrix.	132
$\operatorname{adj}(\cdot)$	Adjugate of a square matrix.	88
$\delta[k]$	Discrete-time impulse sequence.	116
$\det(\cdot)$	Determinant of a square matrix.	131
$\det(\cdot)$	Determinant of a square matrix.	88
$\eta(a, \omega)$	Describing function of a nonlinearity for the input $a\sin(\omega t)$.	106
$\|\|G\|\|_\infty$	The maximum magnitude of $G(j\omega)$.	36
$\|\|u\|\|_\infty$	Least upper bound of a signal $u(t)$.	21
$\mathbb{C}^+$	Open right half complex plane.	7
$\mathbb{C}^-$	Open left half complex plane.	7
$\mathbf{1}(t)$	Unit step function.	7
$\mathbf{1}[k]$	Discrete-time unit step function.	7
$\mathscr{F}(x)$	Fourier transform of a signal $x(t)$.	16
$\mathscr{L}(x)$	Laplace transform of a signal $x(t)$.	13
$j\mathbb{R}$	Imaginary axis.	7

Acronyms and Initialisms

B.I.B.O. Bounded-Input Bounded-Output. 21

C.R.H.P. Closed Right Half Complex Plane. 7

C.T. Continuous-Time. 12

ch.p. characteristic polynomial. 25

D.T. Discrete-Time. 12

F.I.R. Finite Impulse Response. 131

F.T. Fourier Transform. 16

F.V.T. Final-Value Theorem. 27

L.T. Laplace Transform. 13

L.T.I. Linear Time Invariant. 17

O.L.H.P. Open Left Half Complex Plane. 7

O.R.H.P. Open Right Half Complex Plane. 7

P.C. Personal Computer. 1

P.I. Proportional-Integral. 91

P.P.P. Pole Placement Problem. 54

R.O.C. Region of Convergence. 13

Chapter 1

Introduction

This course is about digital control systems. The controllers are implemented on digital computers.

Contents

1.1 Example

Consider a simple example; a cart with a motor drive as in Figure 1.1.

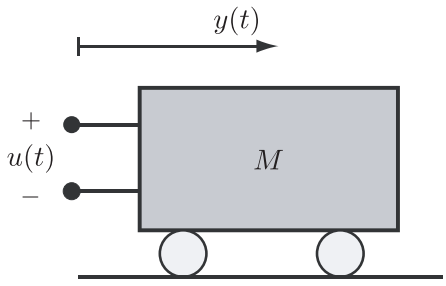


Figure 1.1: A simple cart.

The cart has an input $u(t)$ which is the voltage to the motor and an output $y(t)$ which is the position of the cart. Suppose the system is controlled by an embedded system or Personal Computer (P.C.) equipped with hardware that can do analog-to-digital conversion and digital-to-analog conversion, see Figure 1.2.

In this setup the position $y(t)$ is sampled and these sampled values are read in to the computer as a data stream. The computer does some appropriate discrete-time control computations. Then the discrete-time control values are output as a voltage $u(t)$. The reference signal $r(t)$ is shown as another continuous-time signal to the computer but it could also be generated in software.

So a digital control system involves both continuous-time and discrete-time signals. Let's see this in more detail by modeling the components of the PC from a control viewpoint as shown in Figure 1.3. Continuous-time

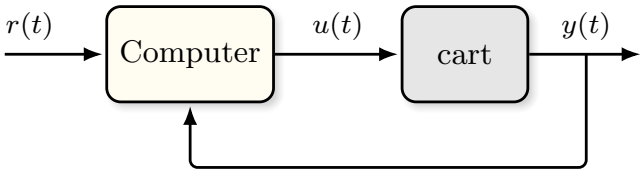


Figure 1.2: A simple cart with computer control.

signals are shown as solid lines and discrete-time signals are shown as dashed lines. The "μ" component denotes the actual processing unit of the embedded system. The simplest example would be this: The position $y(t)$ is

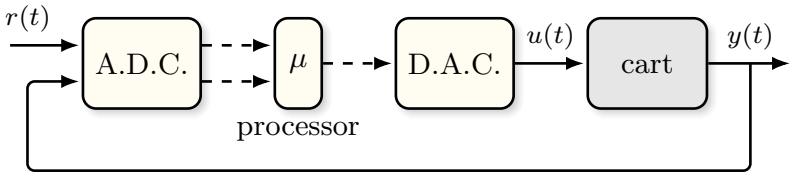


Figure 1.3: A simple cart with a more detailed model of the controller.

sampled to produce $y(kT)$, where T is the sampling period and k is an integer – the discrete-time independent variable. Likewise for $r(kT)$. Then proportional error feedback is computed

$$u(kT) = K_p\left(r(kT) - y(kT)\right).$$

Here, K_p is a controller gain. Then $u(kT)$ is converted to $u(t)$ by interpolation, the simplest method is to hold the sampled value for the period T:

$$u(t) = u(kT), \qquad kT \le t < (k+1)T.$$

Notice that the cart sees a continuous-time system, see Figure 1.4.

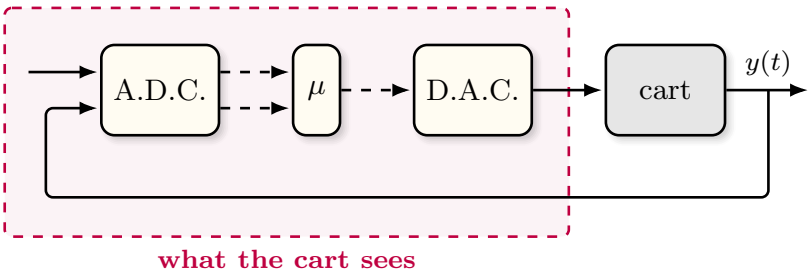


Figure 1.4: System from the viewpoint of the cart.

On the other hand, the processor sees a discrete-time system, see Figure 1.5. In this course we will be dealing with both continuous-time and discrete-time systems and their interconnections. Feedback systems that have a mixture of discrete-time and continuous-time elements are called **sampled-data systems**.

1.2 Some themes in the course

1. Discrete-time signals and system theory is similar to, and sometimes simpler than, the continuous-time theory.
   - Discrete-time has difference equations instead of differential equations.
   - Discrete-time has z-transforms instead of Laplace transforms.

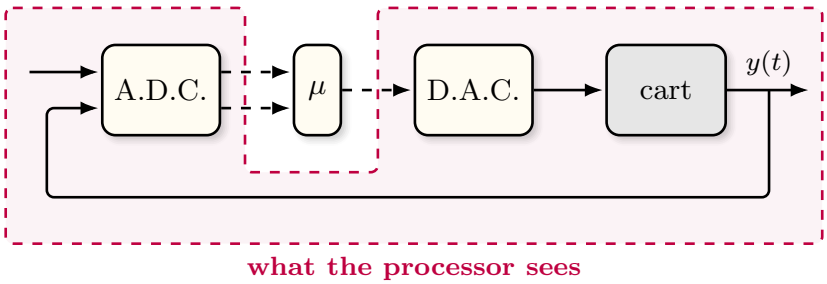


Figure 1.5: System from the viewpoint of the processor.

- Both domains have transfer function models and state-space models.

2. It's the interface elements A/D and D/A that complicate things. Sampled-data systems are periodically time-varying, not time-invariant.

3. Good block diagrams are very helpful. We'll use dashed arrows to indicate discrete-time signals and solid lines to indicate continuous-time signals.

4. Digital control design is *not* more difficult than analog design unless there are hardware limitations. Furthermore, digital controllers can achieve some specifications that cannot be achieved using analog control.

5. When the system to be controlled is physical in nature (like the cart), we will end up with a sampled-data system. However some important applications are naturally cast as pure discrete-time control systems.

   - There are various applications of control theory in the general area of computing [Hellerstein et al., 2004]. Many of these problems cannot be posed in terms of differential equations and hence, do not have an analog equivalent.
   - Control theory is used in online advertising [Karlsson, 2020] and inventory/supply-chain control [Lalwani et al.,
     These problems are also naturally posed in discrete-time.
   - The way that information spreads in a social network has been modelled as a discrete-time system [Egerstedt, 2011]. There is much recent research on how a group of people or robots can achieve "agreement," or "consensus." For an introduction to this vast subject, consult the survey papers [Ren and Beard, 2008] and [Garin and Schenato, 2010].
   - The economy is a large, dynamical system with many actors: governments, organizations, companies and individuals [Åström and Murray, 2019]. Governments control the economy through laws and taxes, the central banks by setting interest rates and companies by setting prices and making investments. Individuals control the economy through purchases, savings and investments. Many efforts have been made to model the system both at the macro level and at the micro level, but this modeling is difficult because the system is strongly influenced by the behaviours of the different actors in the system.

     Keynes [Keynes, 1936] developed a simple model to understand relations among gross national product, investment, consumption and government spending. One of Keynes' observations was that under certain conditions, e.g., during the 1930s depression, an increase in the investment of government spending could lead to a larger increase in the gross national product. This idea was used by several governments to try to alleviate the depression. Keynes' ideas can be captured by a simple discrete-time model.

1.2.1 The use of computers in system operation and control

In the past, control systems were implemented using analog devices and circuits. Some disadvantages of analog controllers: 1) Inflexible: Changes in control or system require rewiring; difficult to maintain. 2) Difficult to implement sophisticated controllers.

When computers became cheap enough, they replaced analog controllers. They were referred to as digital control. Nowadays, computer control refers to the use of computers in many aspects of system operation and control:

1. Sequencing: Most often in consumer electronics, e.g., your microwave oven. There is little dynamics or interaction with the environment.
2. Digital control: Carry out the actual control of the system, including control law implementation, control signal generation, signal acquisition, etc.
3. Distributed control: Implementation of microcontrollers at different locations as simple control loops to achieve an overall objective.
4. Supervision and control: Monitors the status of various processes (hardware and software) in addition to controlling the physical processes. In addition, the computer may also be used for other tasks not directly related to control, e.g., data logging and presentation, interaction with the human operator, etc.

1.3 Continuous-time control systems

The basic block diagram of a **unity feedback continuous-time system** is shown in Figure 1.6. This is the setup that you studied in your third year control course. The signals and systems in this figure are:

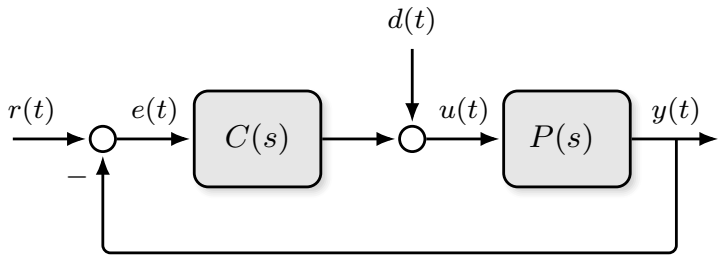


Figure 1.6: Continuous-time unity feedback control system.

systems	$P(s)$, plant transfer function
	$C(s)$, controller transfer function
signals	r, reference (or command) input
	e, tracking error
	d, disturbance
	u, plant input
	y, plant output.

The signals coming from the outside — r and d — are called **exogenous inputs**. The plant model $P(s)$ is a transfer function derived using physical laws and/or system identification techniques. The controller $C(s)$ is designed by the control engineer. You can think of this as a block diagram for the cart control system in Section 1.1 when taking the point of view of the cart. Typical control objectives are:

- closed-loop stability,

- adequate transient performance and robustness to model uncertainty and disturbances,
- tracking, i.e., $y(t) \to r(t)$ as $t \to \infty$ for a family of reference signals, e.g., steps.

Continuous-Time Control Design Problem. *Given a set of control objectives and a plant model $P(s)$, design, if possible, a control law $C(s)$ such that the closed-loop system achieves the objectives.*

1.4 Discrete-time control systems

Suppose that we don't have a plant that operates in continuous-time, rather in discrete-time. In other words the plant input and output are only defined at discrete-time instants. As you know from your signals and systems course we use the z-transform in discrete-time instead of the Laplace transform. The block diagram of a linear time-invariant discrete-time plant will be drawn as in Figure 1.7.

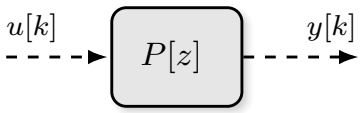


Figure 1.7: Discrete-time plant block diagram.

Here $P[z]$ is transfer function of the plant. It is obtained by taking the z-transform of a system described by a linear time-invariant difference equation and setting all initial conditions to zero. The signals $u[k]$ and $y[k]$ represent, respectively, a discrete-time input sequence and the corresponding discrete-time output sequence. We will always write the argument of a discrete-time signal using square brackets.

The discrete-time unity feedback control system is illustrated in Figure 1.8. You can think of this as a block diagram for the cart control system in Section 1.1 when taking the point of view of the embedded processor. The signals and systems in this figure are:

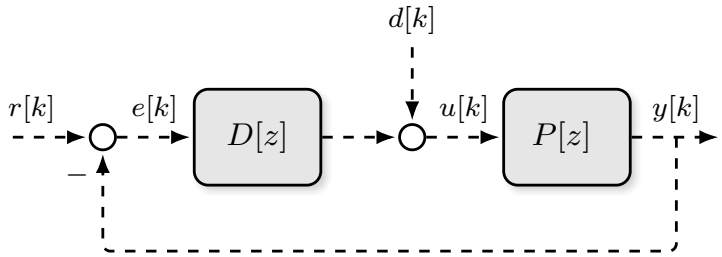


Figure 1.8: Discrete-time unity feedback control system.

systems
- $P[z]$, plant transfer function
- $D[z]$, controller transfer function

signals
- r, reference (or command) input
- e, tracking error
- d, disturbance
- u, plant input
- y, plant output.

In this course we will design feedback control laws for discrete-time systems like the one in Figure 1.8.

1.5 Sampled-data control systems

Systems that have a mixture of discrete-time and continuous-time elements are called sampled-data systems. Our setup is illustrated in Figure 1.9. You can think of this as a block diagram for the cart control system in

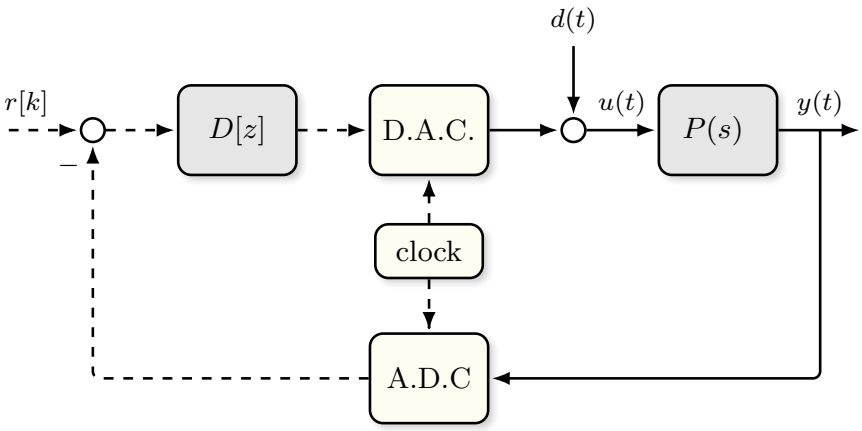


Figure 1.9: Sampled-data unity feedback control system.

Section 1.1 when taking a holistic view of the overall system. The main objective of this course is to analyze and design control laws for this class of system.

Sampled-Data Control Design Problem. *Given a set of continuous-time performance specifications and a plant model $P(s)$, design, if possible, a digital control law $D[z]$ such that the closed-loop system achieves the objectives.*

There are two approaches to solving this design problem: **emulation** and **direct design**.

1. Emulation:
   - Ignore the A/D and D/A elements and design a continuous-time controller $C(s)$.
   - Discretize the controller $C(s)$ to obtain $D[z]$ which is an approximation to $C(s)$.
   - Advantage: easy to motivate and uses your known design tools for continuous-time control laws.
   - Disadvantage: The sampling period of the hardware must be relatively small.
2. Direct Design:
   - Discretize the plant to get a discrete-time model $P[z]$ and treat the original sampled-data system as a discrete-time system.
   - Design $D[z]$ for the discretized plant $P[z]$.
   - Advantage: can handle large sampling periods.
   - Disadvantage: Analysis can be intricate.

1.6 Course objectives

This course is a follow-up to your introductory course on continuous-time control systems. From ECE484 it is hoped that you learn the following:

- Implementation issues associated with using computers in a modern control system.

- How to deal with common nonlinearities that come up in the implementation of control systems.
- The benefits and drawbacks of emulation and direct design.
- How to discretize continuous-time controllers and plants and why we use different methods for controllers and plants.
- How to test for stability of discrete-time systems and discrete-time feedback systems.
- How to design controllers using pole placement in continuous-time and discrete-time.
- How to design a simple feedback loop using frequency-domain methods in discrete-time.
- How to design tracking and regulation controllers for discrete-time systems using state-space models.

1.7 Very brief history

1. Pioneering period 1955-59: Owes much to chemical process industry. Computers at the time were very expensive. In 1959, the Texaco Company used the first digital computer based control system, called the Thompson-Ramo-Wooldridge 300. Too expensive for mass adoption, only made economic sense when the process was complex and very expensive to run. Mainly used for finding optimal operating conditions and changing setpoints on analog controllers. Computers were of course slow and unreliable (mean time between failures was hours or days). Supported by computer manufacturers who saw a potential market.
2. The first use of a digital computer for fully direct control of a process was initiated by Imperial Chemical Industries who began work in 1959 with the Ferranti Company on a Direct Digital Control (DDC) scheme for a soda ash plant at Fleetwood, Lancashire UK.
3. Transistors helped to bring in minicomputers 1967: Reduced costs for a small control system to $100,000. Allowed computer control to be implemented for less complex applications. Can be installed close to the process. Special process control computers were introduced. Considerable improvement in speed, data storage capacity, and reliability.
4. With the introduction of the PC in 1983, the design of modern control systems became possible for the individual engineer. Thereafter, a plethora of software control systems design packages was developed, including ORACLS, Program CC, Control-C, PC-Matlab, MATRIX_x, Easy5, SIMNON, and others.
5. 1980s - 1990s: General use in all areas of control. Microcontrollers used in consumer electronics (CD players, etc.)
6. 1990s - 2000s: Embedded systems and distributed control, control networks consisting of many computers and devices communicating to each other through a communication network and protocol (e.g., BMWs).
7. The theory of digital control and sampled-data systems was developed by many researchers. Since all computer-controlled systems rely on sampling, it is very important to know when a signal can be recovered from its samples. Nyquist showed that to recover a sinusoidal signal from its samples, it is necessary to sample at least twice per period. A complete solution to this problem was given in a landmark work by Claude Shannon [Shannon, 1949]. During the late 1940's and early 1950's, the theory of z-transforms was being developed by Hurewicz [Hurewicz, 1947], Tsypkin [Tsypkin, 1950], and Barker [Barker, 1952]. During the 1950's, the theory of sampled data systems was being developed at Columbia University by J.R. Ragazzini, G. Franklin, and L.A. Zadeh [Ragazzini and Franklin, 1958], [Ragazzini and Zadeh, 1952]; as well as by E.I. Jury [Jury, 1960], B.C. Kuo [Kuo, 1963], and others.

Refer to [Åström and Wittenmark, 1997] for more historical details. See [Williams, 1978] for an interesting account of the early attempts to use digital computers for industrial chemical plant control.

1.8 Notation

Generally, signals are written lower case: e.g., $x(t)$ or $x[k]$. Their transforms are capitalized: $X(s)$, $X(j\omega)$ or $X[z]$. The impulse is $\delta(t)$ or $\delta[k]$. In signals and systems the unit step is denoted $u(t)$, but in control $u(t)$ denotes a plant input so we'll denote the unit step by $\mathbf{1}(t)$ or $\mathbf{1}[k]$. We will use the following notation for parts of the complex plane

$$\begin{aligned}
\mathbb{C}^- &:= \{s \in \mathbb{C} : \mathrm{Re}\,(s) < 0\} && \text{(Open Left Half Complex Plane (O.L.H.P.))}\\
\mathbb{C}^+ &:= \{s \in \mathbb{C} : \mathrm{Re}\,(s) > 0\} && \text{(Open Right Half Complex Plane (O.R.H.P.))}\\
j\mathbb{R} &:= \{s \in \mathbb{C} : \mathrm{Re}\,(s) = 0\} && \text{(Imaginary axis)}\\
\overline{\mathbb{C}}^+ &:= \{s \in \mathbb{C} : \mathrm{Re}\,(s) \geq 0\} && \text{(Closed Right Half Complex Plane (C.R.H.P.))}.
\end{aligned}$$

We denote by $\mathbb{R}[s]$ the ring of polynomials in the (complex) variable s with coefficients in $\mathbb{R}$. The symbol $:=$ means equal by definition. It is convenient to write vectors sometimes as column vectors and sometimes as n-tuples, i.e., ordered lists. For example

$$x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, \qquad x = (x_1, x_2).$$

We'll use both.

1.A Appendix: Aliasing

Suppose that we sample[1] a signal $x(t)$ with sampling frequency $\omega_\mathrm{s} = 2\pi/T$.

Aliasing occurs when the continuous-time signal x contains frequency content higher than $\omega_\mathrm{s}/2$, i.e., the Nyquist frequency. When this happens, distinct continuous-time signals can become indistinguishable after sampling.

Example 1.A.1. On the one hand, sampling the signal $x_0(t) = \cos(\omega t)$ with sampling frequency $\omega_\mathrm{s} = 2\pi/T$ yields

$$\begin{aligned}
x_0[k] &= x_0(kT)\\
&= \cos(\omega k T)\\
&= \cos\left(2\pi \frac{\omega}{\omega_\mathrm{s}} k\right).
\end{aligned}$$

On the other hand, sampling the signal $x_1(t) = \cos((\omega + \omega_\mathrm{s})t)$ with the same sampling frequency yields

$$\begin{aligned}
x_1[k] &= \cos((\omega + \omega_\mathrm{s})kT)\\
&= \cos\left(2\pi \frac{\omega + \omega_\mathrm{s}}{\omega_\mathrm{s}} k\right)\\
&= \cos\left(2\pi \frac{\omega}{\omega_\mathrm{s}} k + 2\pi k\right)\\
&= \cos\left(2\pi \frac{\omega}{\omega_\mathrm{s}} k\right).
\end{aligned}$$

So $x_0[k] = x_1[k]$ even though $x_0(t) \neq x_1(t)$. ▲

[1] For simplicity we model the A.D.C. block using an ideal sampler, see Section 4.2.

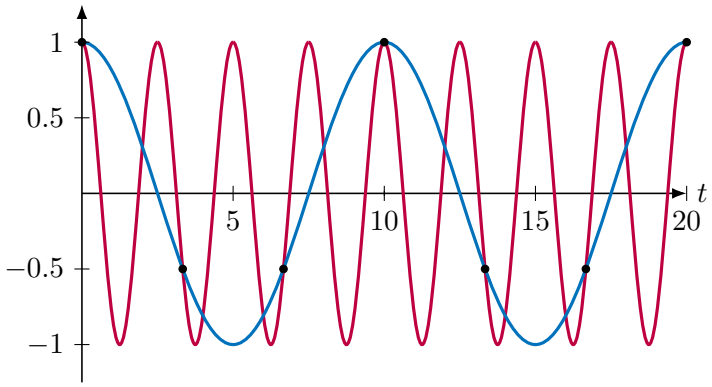


Figure 1.10: Aliasing in Example 1.A.1 with $\omega = \pi/5$ and sampling period 10/3 seconds.

Generalizing Example 1.A.1, we can say that any member of the family of continuous-time signals

$$x_n(t) = \cos\left((\omega + n\omega_s)t\right), \qquad n \in \mathbb{Z}$$

produces the same discrete-time signal

$$x[k] = \cos\left(2\pi \frac{\omega}{\omega_s} k\right)$$

when sampled at the frequency ω_s radians per second. To avoid aliasing, we must sample at least twice as fast as the highest frequency content in our measurement – including any corrupting noise.

In the context of digital control, aliased noise enters the controller and can generate spurious control actions. For example, 1kHz noise in a motor control system, aliased down to 10Hz, will generate observable 10Hz

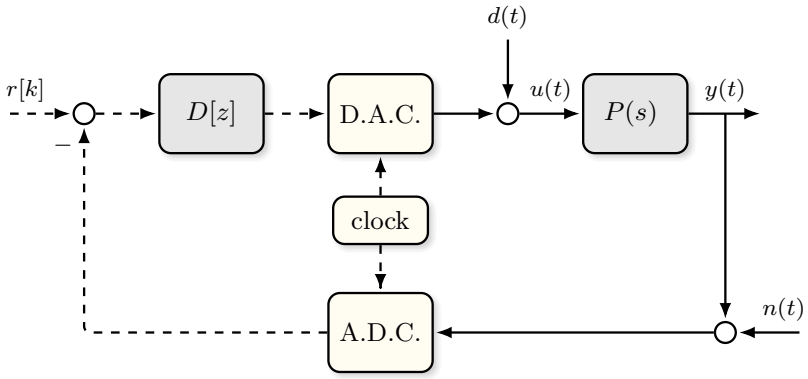


Figure 1.11: Measurement noise n entering the loop.

oscillations on the motor's shaft. There are two common strategies for dealing with aliasing.

1. Sample fast: select ω_s much larger than the bandwidth of the plant. Sometimes this is infeasible and furthermore sometimes unmodelled high frequency noise may still result in aliasing.

2. Filtering: include an analog low-pass filter $F(s)$ in the loop as in Figure 1.12. The filter reduces aliasing by (ideally) filtering out all frequencies greater than its cutoff frequency ω_{cutoff}. A rule of thumb is $\omega_{\text{cutoff}} \leq \omega_s/5$. The filter adds expense and introduces phase lag into the feedback loop which can hurt stability margins.

Example 1.A.2. (Data acquisition in a lab) In the lab we are acquiring data. The signal of interest has frequency 8π rad/s (4 Hz) and we are sampling at $\omega_s = 100\pi$ rad/s (50 Hz). Suppose that noise from the lights enters at 120π and 240π (60 Hz and 120 Hz).

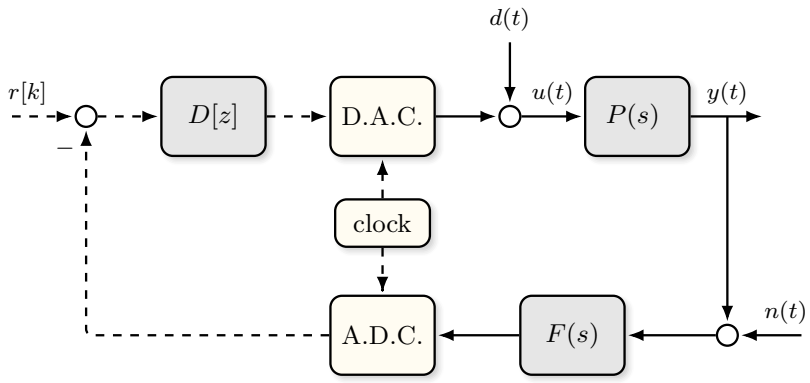
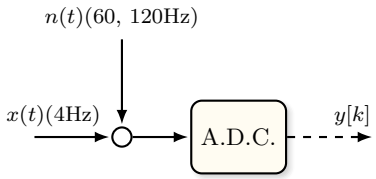


Figure 1.12: Anti-aliasing filter introduced at the input to the A.D.C.

(a) Without using an anti-aliasing filter, the setup is as below.

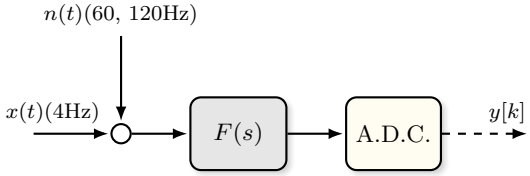


The 120π rad/s noise will be aliased to $\omega + n\omega_s = 120\pi + n100\pi$, $n \in \mathbb{Z}$. The 240π rad/s noise will be aliased to $\omega + n\omega_s = 240\pi + n100\pi$, $n \in \mathbb{Z}$. The sampled signal can only exhibit frequencies at or below the Nyquist frequency $\omega_s/2 = 50\pi$ rad/s (25 Hz). As a result of these facts, the sampled signal will have frequency content at

$$\{8\pi, 20\pi, 40\pi\}$$

radians per second where the latter two components are due to aliasing of the noise.

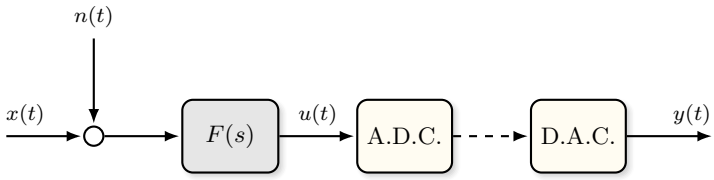
(b) If an (ideal) low-pass anti-aliasing filter $F(s)$ with cutoff frequency ω_{cutoff} is added then the setup is as below.



If ω_{cutoff} equals either 40π or 90π rad/s, then the noise signals will not appear at the A.D.C. and only the desired 8π rad/s content will appear in the sampled signal y. However, in this example $\omega_{\text{cutoff}} = 90\pi$ rad/s is a worse choice because it allows unmodelled noise in the range $[40\pi, 90\pi]$.

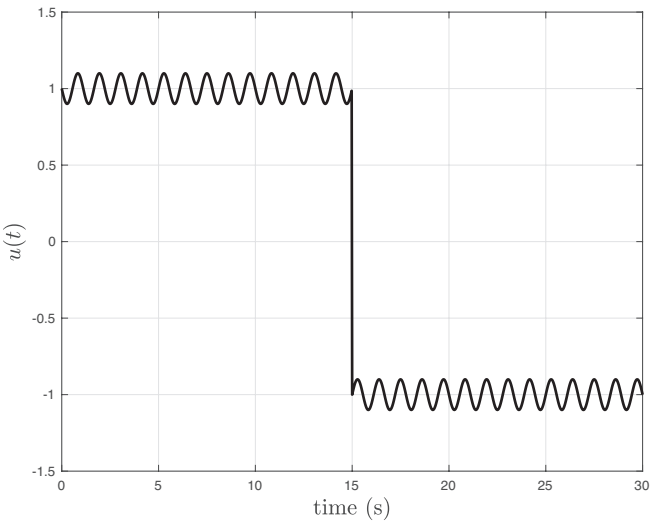
▲

Example 1.A.3. Consider the setup below

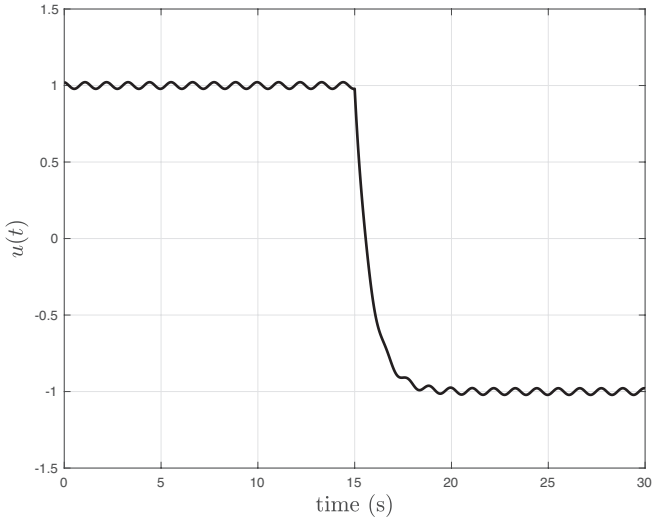


where x is a square wave with period 60 seconds and n is a sinusoid of frequency 0.9 Hz. The sampling rate is $\omega_s = 2\pi$ rad/s. Figure 1.13 illustrates the effect of anti-alias filtering.

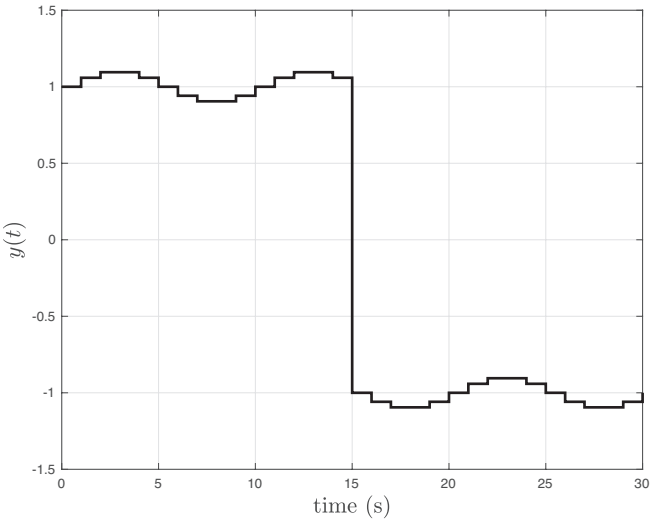
▲



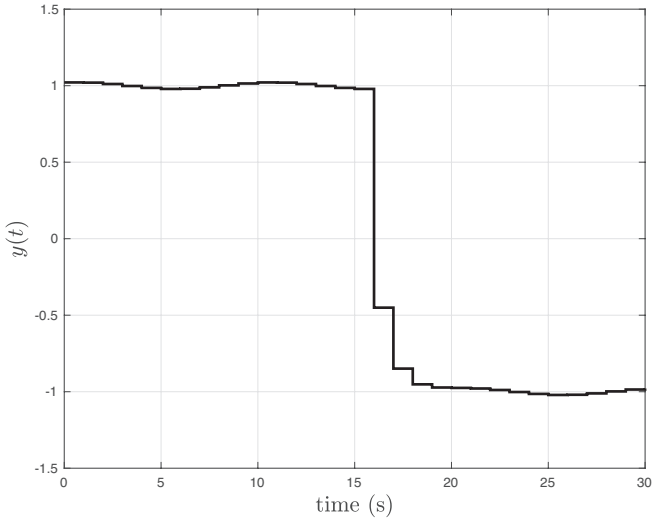
(a) Input to A.D.C. with $F(s) = 1$.



(b) Input to A.D.C. with $F(s) = \frac{\omega_{\text{cutoff}}}{s+\omega_{\text{cutoff}}}$, $\omega_{\text{cutoff}} = \omega_s/5$.



(c) Output of D.A.C. with $F(s) = 1$.



(d) Output of D.A.C. with $F(s) = \frac{\omega_{\text{cutoff}}}{s+\omega_{\text{cutoff}}}$, $\omega_{\text{cutoff}} = \omega_s/5$.

Figure 1.13: The effect of anti-aliasing filters in Example 1.A.3.

Chapter 2

Review of signals, systems and analog control

In this chapter we briefly review some basic concepts from continuous-time control of linear systems, linear signals and systems and linear algebra. The material should be familiar to you.

Contents

2.1 Signals and transforms

A Continuous-Time (C.T.) signal x is a real- or complex-valued function of a real variable t. Here t usually represents time, e.g., $x(t)$ could be the charge level in the battery of a mobile phone at time t. A Discrete-Time (D.T.) signal x is a real- or complex-valued function of an integer k, e.g., $x[k]$ could be your monthly bank statement or the value of a variable in a computer program. In this course discrete-time signals normally come from sampling continuous-time signals.

When we write $\sin(\theta)$ the angle is always in radians. So in the signal $\sin(\omega t)$, ωt has units of radians, and hence ω has units of radians per second, assuming that t is in seconds. We could also write $\omega = 2\pi f$, where f is in Hz. We usually write $\sin(\omega t)$ instead of $\sin(2\pi f t)$. Instead of using $\sin(\omega t)$ or $\cos(\omega t)$ we'll often use the complex sinusoid $e^{j\omega t}$. By Euler's formula

$$e^{j\omega t} = \cos(\omega t) + j \sin(\omega t)$$

and therefore

$$\cos(\omega t) = \mathrm{Re}\left(e^{j\omega t}\right) = \frac{1}{2}\left(e^{j\omega t} + e^{-j\omega t}\right) \qquad \sin(\omega t) = \mathrm{Im}\left(e^{j\omega t}\right) = \frac{1}{2j}\left(e^{j\omega t} - e^{-j\omega t}\right).$$

The continuous-time impulse $\delta(t)$ is not really a legitimate function because its "value" at $t = 0$ is not a real number. And you can't rigorously get δ as the limit of a sequence of ever-narrowing rectangles, because that sequence does not converge in any ordinary sense. In mathematics it is called a **distribution**.

The main idea is that $\delta(t)$ is not a function, but rather it is a way of defining the linear transformation $x \mapsto x(0)$ that maps a signal $x(t)$ to its value at $t = 0$. This linear transformation should properly be written as $\delta(x) = x(0)$ (i.e., δ maps x to $x(0)$) but historically it has been written as

$$\int_{-\infty}^{\infty} x(t)\delta(t)\mathrm{d}t = x(0).$$

You know this as the "sifting formula." Let us emphasize that the expression

$$\int_{-\infty}^{\infty} x(t)\delta(t)\mathrm{d}t \tag{2.1}$$

is not intended to mean integration of the product $x(t)\delta(t)$ of functions — δ isn't a function; rather, the expression is that of an operation on x whose value is defined to be $x(0)$. The expression is defined to be valid for all functions $x(t)$ that are smooth at $t = 0$; smooth means continuously differentiable up to every order. For example, using a change of variables $\lambda := t - \tau$, we get

$$\int_{-\infty}^{\infty} x(t)\delta(t-\tau)\mathrm{d}t = \int_{-\infty}^{\infty} x(\lambda+\tau)\delta(\lambda)\mathrm{d}\lambda = x(\tau).$$

2.1.1 The Laplace transform

Let $x(t)$ be a signal defined for all t or for just $t \geq 0$. The one-sided **Laplace Transform (L.T.)** of $x(t)$ is

$$X(s) = \int_{0}^{\infty} x(t)\mathrm{e}^{-st}\mathrm{d}t.$$

Here s is a complex variable. We use $\mathscr{L}(x)$ to denote the Laplace Transform of x. Normally, the integral converges for some values of s and not for others. That is, there is a **region of convergence**. It turns out that the Region of Convergence (R.O.C.) is always an open right half-plane, of the form $\{s \in \mathbb{C} : \mathrm{Re}\,(s) > a\}$. Within the ROC $X(s)$ has no poles.

Example 2.1.1. The unit step:

$$x(t) = \mathbf{1}(t) = \begin{cases} 1 & , \quad t \geq 0 \\ 0 & , \quad t < 0. \end{cases}$$

Actually, the precise value at $t = 0$ doesn't matter. The LT is

$$X(s) = \int_{0}^{\infty} \mathrm{e}^{-st}\mathrm{d}t = -\frac{\mathrm{e}^{-st}}{s}\Big|_{0}^{\infty} = \frac{1}{s}$$

and the ROC is

$$\{s \in \mathbb{C} : \mathrm{Re}\,(s) > 0\}.$$

The same $X(s)$ is obtained if $x(t) = 1$ for all t, even $t < 0$. This is because the one-sided LT is oblivious to negative time. Notice that $X(s)$ has a pole at $s = 0$ on the western boundary of the ROC. ▲

The LT exists provided $x(t)$ satisfies two conditions. The first is that it is piecewise continuous on $t \geq 0$. This means that, on any time interval (t_1, t_2), $x(t)$ has at most a finite number of jumps, and between these jumps $x(t)$ is continuous. A square wave has this property for example. The second condition is that it is of exponential order, meaning there exist constants M, c such that $|x(t)| \leq M\mathrm{e}^{ct}$ for all $t \geq 0$. This means that if $x(t)$ blows up, at least there is some exponential that blows up faster. For example, $\exp(t^2)$ blows up too fast.

Example 2.1.2. Some other examples: An exponential:

$$x(t) = \mathrm{e}^{at}, \qquad X(s) = \frac{1}{s-a}, \qquad \mathrm{ROC} : \{s \in \mathbb{C} : \mathrm{Re}\,(s) > a\}\,.$$

A sinusoid:

$$x(t) = \cos(\omega t) = \frac{1}{2}\left(\mathrm{e}^{j\omega t} + \mathrm{e}^{-j\omega t}\right), \qquad X(s) = \frac{s}{s^2+w^2}, \qquad \mathrm{ROC} : \{s \in \mathbb{C} : \mathrm{Re}\,(s) > 0\}\,.$$

▲

The LT thus maps a class of time-domain functions $x(t)$ into a class of complex-valued functions $X(s)$. The mapping $x(t) \mapsto X(s)$ is linear.

Example 2.1.3. Let's use linearity to find the LT of the signal in Figure 2.1.

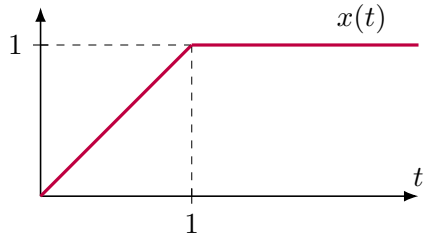


Figure 2.1: Time-domain signal for Example 2.1.3.

Thus $x = x_1 + x_2$, where $x_1(t) = t$ is the unit ramp starting at time 0 and x_2 the ramp of slope -1 starting at time 1. By linearity, $X(s) = X_1(s) + X_2(s)$. We compute that

$$X_1(s) = \frac{1}{s^2}, \qquad \{s \in \mathbb{C} : \mathrm{Re}\,(s) > 0\}\,,$$
$$X_2(s) = -\mathrm{e}^{-s}\frac{1}{s^2}, \qquad \{s \in \mathbb{C} : \mathrm{Re}\,(s) > 0\}\,.$$

Thus

$$X(s) = \frac{1-\mathrm{e}^{-s}}{s^2}, \qquad \{s \in \mathbb{C} : \mathrm{Re}\,(s) > 0\}\,.$$

▲

There are tables of LTs. So in practice, if you have $X(s)$, you can get $x(t)$ using a table like Table 2.1. If

$$X(s) = \frac{N(s)}{D(s)}$$

where N, $D \in \mathbb{R}[s]$ are polynomials and the degree of N is strictly less than the degree of D, then we can use partial fraction expansions and the linearity of the LT to obtain $x(t)$.

Example 2.1.4. Given $X(s) = \frac{3s+17}{s^2-4}$, let us find $x(t)$. We don't need the ROC to find $x(t)$, but actually we know what it is. Since we're using the one-sided LT the ROC must be a right half-plane, and because $X(s)$ must be analytic within its ROC, the ROC of $X(s)$ must be $\mathrm{Re}\,(s) > 2$. We have

$$X(s) = \frac{3s+17}{s^2-4} = \frac{c_1}{s-2} + \frac{c_2}{s+2}, \quad c_1 = \frac{23}{4}, \quad c_2 = -\frac{11}{4}$$

and therefore

$$x(t) = c_1\mathrm{e}^{2t} + c_2\mathrm{e}^{-2t}, \quad t \geq 0.$$

We do not know if $x(t)$ equals zero for $t < 0$. ▲

Table 2.1: Important (one-sided) Laplace transforms.

Description	Time domain $f(t)$	s-Domain $F(s)$
Unit step	$\mathbf{1}(t)$	$\frac{1}{s}$
Impulse	$\delta(t)$	1
Ramp	t	$\frac{1}{s^2}$
Exponential	e^{at}	$\frac{1}{s-a}$
Sine	$\sin(\omega t)$	$\frac{\omega}{s^2+\omega^2}$
Cosine	$\cos(\omega t)$	$\frac{s}{s^2+\omega^2}$
Generalized exponential	$t^n\mathrm{e}^{at}$	$\frac{n!}{(s-a)^{n+1}}$
Generalized sine	$\mathrm{e}^{at}\sin(\omega t)$	$\frac{\omega}{(s-a)^2+\omega^2}$
Generalized cosine	$\mathrm{e}^{at}\cos(\omega t)$	$\frac{s-a}{(s-a)^2+\omega^2}$
Sine with linear growth	$t\sin(\omega t)$	$\frac{2\omega s}{(s^2+\omega^2)^2}$
Cosine with linear growth	$t\cos(\omega t)$	$\frac{s^2-\omega^2}{(s^2+\omega^2)^2}$

Example 2.1.5. Given

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

let us find $x(t)$. Once again, we don't need the ROC to find $x(t)$, but we can deduce that it is the right half-plane $\mathrm{Re}\,(s) > 2$. We have

$$X(s) = \frac{s-2}{(s-1)(s+2)} = \frac{c_1}{s-1} + \frac{c_2}{s+2}, \quad c_1 = -\frac{1}{3}, \quad c_2 = \frac{4}{3}$$

and therefore

$$x(t) = -\frac{1}{3}\mathrm{e}^{t} + \frac{4}{3}\mathrm{e}^{-2t}, \quad t \geq 0.$$

▲

Example 2.1.6. Given

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

we write

$$X(s) = \frac{s+1}{s(s+2)^2} = \frac{c_1}{s} + \frac{c_2}{s+2} + \frac{c_3}{(s+2)^2}.$$

This implies

$$s+1 = c_1(s+s)^2 + c_2 s(s+2) + c_3 s.$$

Comparing coefficients we get three linear equations. You can verify that

$$c_1 = \frac{1}{4},\ c_2 = -\frac{1}{4},\ c_3 = \frac{1}{2}.$$

Therefore

$$x(t) = \frac{1}{4} - \frac{1}{4}\mathrm{e}^{-2t} + \frac{1}{2}t\mathrm{e}^{-2t}, \quad t \geq 0.$$

▲

The LT of the product $f(t)g(t)$ of two functions is *not* equal to $F(s)G(s)$, the product of the two transforms. Then what operation in the time-domain does correspond to multiplication of the transforms? The answer is **convolution**. Let $f(t), g(t)$ be defined on $t \geq 0$. Define a new function

$$h(t) = \int_0^t f(t-\tau)g(\tau)\mathrm{d}\tau, \ t \geq 0.$$

We say h is the convolution of f and g. Another equivalent way of writing h is

$$h(t) = \int_0^t f(\tau)g(t-\tau)\mathrm{d}\tau, \ t \geq 0.$$

We also frequently use the star notation $h = f * g$ or $h(t) = f(t) * g(t)$ to indicate convolution.

Properties of Laplace transforms

Let f and g be real-valued univariate functions, continuously differentiable at $t = 0$, and let a be a real constant.

(i) $\mathscr{L}\{f+g\} = \mathscr{L}\{f\} + \mathscr{L}\{g\}$

(ii) $\mathscr{L}\{af\} = a\mathscr{L}\{f\}$

(iii) $\mathscr{L}\left\{\frac{\mathrm{d}f}{\mathrm{d}t}\right\} = s\mathscr{L}\{f\} - f(0)$

(iv) $\mathscr{L}\{f * g\} = \mathscr{L}\{f\}\mathscr{L}\{g\}$

(v) $\mathscr{L}\left\{\int_0^t f(\tau)\mathrm{d}\tau\right\} = \frac{1}{s}\mathscr{L}\{f\}$

(vi) $\mathscr{L}\{f(t-T)\} = \mathrm{e}^{-sT}\mathscr{L}\{f\}, \qquad T \geq 0.$

2.1.2 The Fourier transform

The **Fourier transform** of $x(t)$ is

$$X(j\omega) = \int_{-\infty}^{\infty} x(t)\mathrm{e}^{-j\omega t}\mathrm{d}t.$$

We use $\mathscr{F}(x)$ to denote the Fourier Transform of x. The inversion formula is

$$x(t) = \frac{1}{2\pi}\int_{-\infty}^{\infty} X(j\omega)\mathrm{e}^{j\omega t}\mathrm{d}\omega.$$

If the absolute value of $x(t)$ is integrable, that is,

$$\int_{-\infty}^{\infty} |x(t)|\mathrm{d}t < \infty$$

then $X(j\omega)$ is a continuous function of ω. An example is $x(t) = \mathrm{e}^{-|t|}\cos(t)$. If the absolute value of $x(t)$ is only square integrable, that is,

$$\int_{-\infty}^{\infty} |x(t)|^2\mathrm{d}t < \infty,$$

then $X(j\omega)$ may not be a continuous function of ω. An example is $\sin(t)/t$.

The constant signal that equals one for all time is neither absolutely integrable nor square integrable. Its Fourier Transform (F.T.) is defined to be $2\pi\delta(\omega)$. We can convince ourselves of this by noting that the inversion formula is an instance of the sifting formula

$$\frac{1}{2\pi}\int_{-\infty}^{\infty} 2\pi\delta(\omega)\mathrm{e}^{j\omega t}\mathrm{d}\omega = 1.$$

The forward FT is defined to be

$$\int_{-\infty}^{\infty} \mathrm{e}^{-j\omega t}\mathrm{d}t = 2\pi\delta(\omega)$$

and it has no meaning in the sense of ordinary functions. Likewise, the sinusoidal signal $\mathrm{e}^{j\omega_0 t}$ is neither absolutely integrable nor square integrable. Its FT is defined to be $2\pi\delta(\omega-\omega_0)$. In general, FTs where either $x(t)$ or $X(j\omega)$ is not a function (i.e. has an impulse) must be treated with care to make sure that the result is correct.

2.2 Transfer functions

Linear time-invariant systems, and *only* Linear Time Invariant (L.T.I.) systems, have transfer functions. The **transfer function** of an LTI system is defined to be the ratio $Y(s)/U(s)$ where the LTs are taken with zero initial conditions.

Example 2.2.1. (Time-Varying System) A system governed by the differential equation

$$\dot{y} + (\sin t)y = u.$$

does not have a transfer function — it isn't time invariant. ▲

Example 2.2.2. (Low Pass Filter) Consider the RC low pass filter in Figure 2.2 with input voltage $u(t)$ and output voltage $y(t)$ — the voltage across the capacitor. This circuit only has one loop so that Kirchhoff's

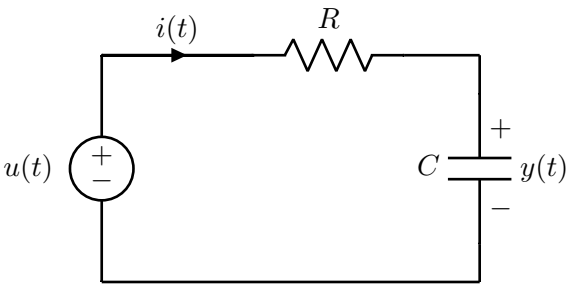


Figure 2.2: An RC filter.

voltage law gives

$$-u(t) + v_R(t) + y(t) = 0$$

where $v_R(t)$ is the voltage across the resistor. Substituting with the device equation for a resistor we get

$$-u(t) + Ri(t) + y(t) = 0.$$

From the device equation for capacitors we have $i(t) = C\dot{y}(t)$ and so the above can be written

$$RC\dot{y}(t) + y(t) = u(t).$$

Take Laplace transforms with zero initial conditions:

$$sRCY(s) + Y(s) = U(s).$$

Therefore the TF is

$$\frac{Y(s)}{U(s)} = \frac{1}{RCs + 1}.$$

This transfer function is **rational**, a ratio of polynomials. ▲

Exercise 2.1. Re-derive the TF from Example 2.2.2 using the voltage-divider rule and complex impedances.

Example 2.2.3. (Mass-Spring-Damper) In mechanics, one of the simplest differential equations is that of a mass-spring system with damping

$$M\ddot{q} + c(\dot{q}) + Kq = u. \tag{2.2}$$

Figure 2.3 illustrates this system. The variable $q(t) \in \mathbb{R}$ represents the position of the mass M with $q = 0$

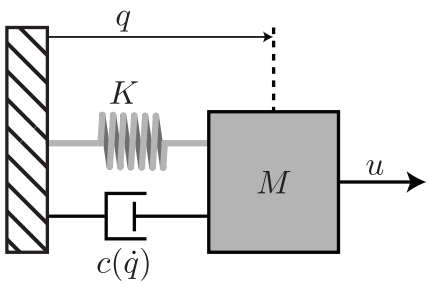


Figure 2.3: Mass-spring-damper system. The position of the mass M is q with $q = 0$ corresponding to the rest position of the spring. The forces acting on M are due to a linear spring with constant K, a damper whose force depends on velocity $\dot{q}$ and an applied force u.

corresponding to the position of the mass when the spring is unsprung. The spring is assumed to be linear, i.e., it satisfies Hooke's law, which says that the force exerted by the spring is proportional to the displacement of the mass. The friction element, called a damper, is assumed to exert a force $c(\dot{q})$ which is possibly nonlinear function of the velocity $\dot{q}$. It can model effects such as viscous drag or static friction. The force u is an externally applied force that we treat as the input. This system is **second order** because the highest derivative appearing in the differential equation (2.2) is the second derivative of q.

If the damper function is a linear function of $\dot{q}$, i.e., $c(\dot{q}) = b\dot{q}$ for some constant $b \in \mathbb{R}$, then the equation of motion becomes linear

$$M\ddot{q} = u - b\dot{q} - Kq.$$

This is a linear time-invariant system so taking LTs we get the TF

$$\frac{Q(s)}{U(s)} = \frac{1}{Ms^2 + bs + K}.$$

If the damper function is a nonlinear function of $\dot{q}$, then the differential equation becomes **nonlinear** and the system does not have a TF. ▲

Example 2.2.4. (RLC Circuit) Consider the RLC circuit in Figure 2.4. Apply KVL to this circuit to get

$$-u(t) + v_R(t) + v_C(t) + v_L(t) = 0$$

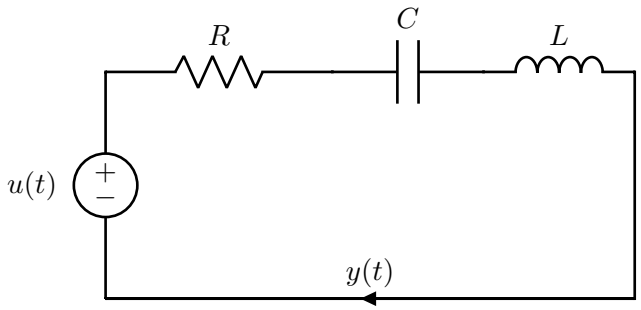


Figure 2.4: RLC circuit.

where $v_R(t)$, $v_C(t)$, $v_L(t)$ are the voltages across, respectively, the resistor, capacitor and inductor. We substitute the device equations and use the definition of $y(t)$ to obtain

$$-u(t) + Ry(t) + L\dot{y}(t) + \frac{1}{C}\int_0^t y(\tau)\mathrm{d}\tau = 0.$$

This is not a differential equation because of the integral. Take the time derivative of this equation to obtain an ODE model

$$-\dot{u} + R\dot{y} + L\ddot{y} + \frac{1}{C}y = 0.$$

Apply Laplace transforms with zero initial conditions:

$$-sU(s) + sRY(s) + s^2LY(s) + \frac{1}{C}Y(s) = 0.$$

The resulting TF is

$$\frac{Y(s)}{U(s)} = \frac{Cs}{LCs^2 + RCs + 1}.$$

▲

Example 2.2.5. (DC Motor) A common actuator used in control systems is the **permanent magnet DC motor** to provide rotary motion. Figure 2.5 shows a schematic diagram of a DC motor.

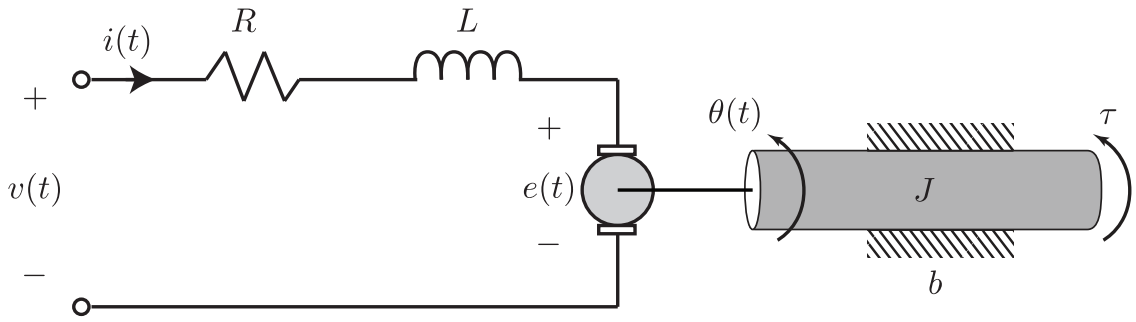


Figure 2.5: DC Motor : Circuit and free body diagram for the rotor/shaft.

It is common to assume that the "electrical time constant" L/R is small in comparison to the "mechanical time constant" J/b. This leads to the simplified model of a DC motor

$$J\ddot{\theta} = -\left(b + \frac{K_\tau K_e}{R}\right)\dot{\theta} + \frac{K_\tau}{R}v$$

where K_τ, K_e are real constants, J is the inertia of the shaft and b models viscous friction. If we now define

$$a_1 \coloneqq \frac{1}{J}\left(b + \frac{K_\tau K_e}{R}\right), \qquad b_0 \coloneqq \frac{K_\tau}{JR},$$

then the TF from the voltage v to the shaft angle θ is

$$\frac{\Theta(s)}{V(s)} = \frac{b_0}{s(s+a_1)}.$$

The TF from the voltage v to the shaft velocity $\omega = \mathrm{d}\theta/\mathrm{d}t$ is

$$\frac{\Omega(s)}{V(s)} = \frac{b_0}{s+a_1}.$$

▲

Table 2.2 lists other examples of transfer functions.

Table 2.2: Common Transfer Functions.

Description	Governing Equation	Transfer Function
Pure gain	$y(t) = u(t)$	1
Integrator	$\dot{y}(t) = u(t)$	$\frac{1}{s}$
Double integrator	$\ddot{y}(t) = u(t)$	$\frac{1}{s^2}$
Ideal differentiator	$y(t) = \dot{u}(t)$	s
Time delay	$y(t) = u(t-T),\ T > 0$	e^{-sT} (irrational)
Prototype second order system	$\ddot{y}(t) + 2\zeta\omega_n\dot{y}(t) + \omega_n^2 y(t) = K\omega_n^2 u(t)$	$\frac{K\omega_n^2}{s^2+2\zeta\omega_n s+\omega_n^2}$
Proportional-integral-derivative controller	$y(t) = K_p u(t) + K_i \int_0^t u(\tau)\mathrm{d}\tau + K_d\dot{u}(t)$	$K_p + \frac{K_i}{s} + K_d s$

Definition 2.2.1. (i) A transfer function $G(s)$ is **(real) rational** if it is the quotient of two polynomials

$$G(s) = \frac{b_m s^m + \cdots + b_1 s + b_0}{s^n + a_{n-1}s^{n-1} + \cdots + a_1 s + a_0}$$

where the coefficients a_i, b_i are real constants. The numbers m, n are the **degrees** of the numerator and denominator polynomials. Let $\mathbb{R}(s)$ denote the set of rational functions in $s \in \mathbb{C}$ with coefficients in $\mathbb{R}$.

(ii) A rational transfer function is **proper** if $n \geq m$. This is equivalent to the condition

$$\lim_{s\to\infty} G(s) \quad \text{exists in } \mathbb{C}.$$

(iii) A rational transfer function is **strictly proper** if $n > m$. This is equivalent to the condition

$$\lim_{s\to\infty} G(s) = 0.$$

(iv) A transfer function is **improper** if it is not proper.

It is understood, once and for all, that the ratio of polynomials

$$\frac{s+1}{s(s+2)}, \qquad \frac{(s+1)(s-1)}{s(s+2)(s-1)}$$

represent the *same* rational function. In the first function the numerator and denominator polynomials are coprime[1] while in the second function these polynomials are not coprime. They represent the same rational function in the same way that the expressions $\frac{5}{10}$ and $\frac{1}{2}$ represent the same rational number.

Definition 2.2.2. A complex number $p \in \mathbb{C}$ is a **pole** of a transfer function $G(s)$ if

$$\lim_{s \to p} |G(s)| = \infty.$$

A complex number $z \in \mathbb{C}$ is a **zero** of a transfer function $G(s)$ if

$$\lim_{s \to z} G(s) = 0.$$

If $G(s)$ is rational and proper and if its numerator and denominator polynomials are coprime, then the roots of the denominator are the poles of $G(s)$ and the roots of the numerator are the zeros of $G(s)$. For example the transfer functions

$$\frac{s+1}{s(s+2)}, \qquad \frac{(s+1)(s-1)}{s(s+2)(s-1)}$$

have poles at $s = 0$ and $s = -2$ and zeros at $s = -1$. In the second expression, $s = 1$ is a root of the denominator but isn't a pole since the numerator and denominator aren't coprime.

2.3 Bounded-input bounded-output stability

Consider an LTI system with a single input, a single output, and a strictly proper rational transfer function. The model is therefore $y = g * u$ in the time-domain, or $Y(s) = G(s)U(s)$ in the s-domain. We ask the question: Does a bounded input always produce a bounded output? First, let's define precisely what boundedness of a signal means.

Definition 2.3.1. Let $u(t)$ be a real-valued signal defined for $t \geq 0$. We say u is **bounded** if there exists a constant b such that, for all $t \geq 0$, $|u(t)| \leq b$.

Familiar bounded signals are steps $u(t) = \mathbf{1}(t)$ and sinusoids $u(t) = \sin(t)$, but not ramps $u(t) = t$ or growing exponentials $u(t) = \mathrm{e}^t$. The **least upper bound** of a signal u is denoted $\|u\|_\infty$. You can think of[2] $\|u\|_\infty$ as $\max_{t \geq 0} |u(t)|$.

Definition 2.3.2. A linear time-invariant system with input u and output y is **Bounded-Input Bounded-Output (B.I.B.O.)** stable if every bounded input u produces a bounded output y, i.e.,

$$\|u\|_\infty \text{ finite} \Rightarrow \|y\|_\infty \text{ finite}.$$

The system is **unstable** if it is not BIBO stable.

[1] Two polynomials are coprime if they have no common roots.
[2] To be precise, $\|u\|_\infty := \sup_{t \geq 0} |u(t)|$ where sup is the **supremum**.

The next theorem says that checking whether or not an LTI system is BIBO stable can be accomplished using either its impulse response or its transfer function.

Theorem 2.3.3. *Assume $G(s)$ is strictly proper, rational. Then the following three statements are equivalent:*

1. *The system is BIBO stable.*
2. *The impulse-response function $g(t)$ is absolutely integrable, i.e., $\int_0^\infty |g(t)|\mathrm{d}t < \infty$.*
3. *Every pole of the transfer function $G(s)$ has negative real part.*

Example 2.3.1. (Low Pass Filter) Consider the RC filter from Example 2.2.2

$$G(s) = \frac{1}{RCs+1}, \qquad g(t) = \frac{1}{RC}\mathrm{e}^{-t/RC}\mathbf{1}(t).$$

The transfer function $G(s)$ has one pole at $s = -\frac{1}{RC}$. Since R is a resistance and C is a capacitance, they are both positive and so the pole has negative real part. According to the theorem, every bounded u produces a bounded y. Let's verify

$$\begin{aligned}
|y(t)| &= \left| \int_0^t g(\tau)u(t-\tau)\mathrm{d}\tau \right| \\
&\leq \int_0^t |g(\tau)||u(t-\tau)|\mathrm{d}\tau \\
&\leq \int_0^t |g(\tau)|\mathrm{d}\tau \ \|u\|_\infty \\
&\leq \int_0^\infty |g(\tau)|\mathrm{d}\tau \ \ \|u\|_\infty \\
&= \int_0^\infty \frac{1}{RC}\mathrm{e}^{-\tau/RC}\mathrm{d}\tau \ \ \|u\|_\infty \\
&= \|u\|_\infty.
\end{aligned}$$

Thus $\|y\|_\infty \leq \|u\|_\infty$ which shows that every bounded input produces a bounded output. ▲

Example 2.3.2. (Integrator) The integrator has

$$G(s) = \frac{1}{s}, \qquad g(t) = \mathbf{1}(t).$$

According to the theorem, the system is not BIBO stable; there exists some bounded input that produces an unbounded output. For example the unit step $u(t) = \mathbf{1}(t)$ is bounded, but it produces the output $y(t) = t\mathbf{1}(t)$, which is an unbounded ramp. It is not true that every bounded input produces an unbounded output, only that some bounded input does. For example, if the input is $u(t) = (\sin t)\mathbf{1}(t)$, then the output is bounded. ▲

Using Theorem 2.3.3 we conclude that $\frac{1}{s+1}$, $\frac{1}{(s+1)^2}$, $\frac{1}{s^2+2s+1}$ are BIBO stable, but not $\frac{1}{s^2}$ nor $\frac{1}{s-1}$.

The theorem can be extended to the case where $G(s) \in \mathbb{R}(s)$ is only proper (and not strictly proper). Then, using long division, write

$$G(s) = G_1(s) + c, \ G_1(s) \text{ strictly proper}, \ c \in \mathbb{R}.$$

The impulse response has the form

$$g(t) = g_1(t) + c\delta(t).$$

Theorem 2.3.3 remains true with the second statement changed to say that $g_1(t)$ is absolutely integrable. On the other hand, the next result shows that improper systems are never BIBO stable.

Theorem 2.3.4. *If $G(s) \in \mathbb{R}(s)$ is improper, then $G(s)$ is not BIBO stable.*

Proof. Write the transfer function $G(s)$ as

$$G(s) = G_1(s) + G_2(s)$$

where $G_1(s) \in \mathbb{R}(s)$ is strictly proper and $G_2(s) \in \mathbb{R}[s]$ has degree at least 1. For any input u the output y is, by linearity, $y = y_1 + y_2$ where

$$Y_1(s) = G_1(s)U(s), \qquad Y_2(s) = G_2(s)U(s).$$

If $G_1(s)$ has poles with non-negative real parts, then the result follows from Theorem 2.3.3. If $G_1(s)$ has all its poles in $\mathbb{C}^-$, consider the bounded input $u(t) = \sin(t^2)$. Any derivative of u will involve terms polynomial in t and such terms will not be bounded as $t \to \infty$. But y_2 is a linear combination of u and its derivatives so the result follows. ■

Theorem 2.3.4 shows that the differentiator system $G(s) = s$ is not BIBO stable.

2.4 Stability of feedback systems

We turn to feedback systems. We make the following standing assumption throughout this section to ensure

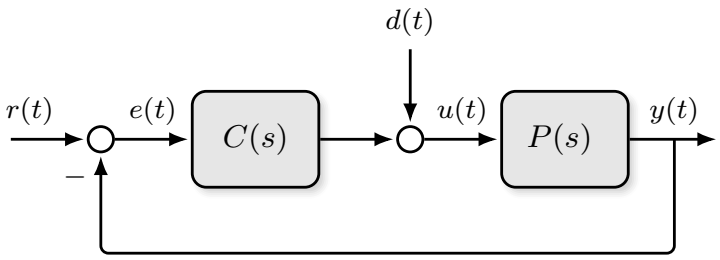


Figure 2.6: Unity feedback system.

that the feedback system in Figure 2.6 is **well-posed**[3].

Assumption 2.4.1. The plant and controller TFs are rational, $C(s)$ is proper, and $P(s)$ is strictly proper. ◀

Our objective is to define what it means for the system in Figure 2.6 to be stable.

[3]Being well-posed means, roughly, that every set of independent signals (r, d) produces a unique set of dependent signals (e, u, y) and that the relationship between the independent and dependent signals is causal.

Definition 2.4.2. The feedback system in Figure 2.6 is **input-output stable** provided e, u, and y are bounded signals whenever r and d are bounded signals; briefly, the system from (r, d) to (e, u, y) is BIBO stable.

Input-output stability is equivalent to saying that the 6 transfer functions from (r, d) to (e, u, y) are BIBO stable, in the sense that all poles are in $\mathrm{Re}\,(s) < 0$. Let's find these transfer functions. First, we write the equations for the outputs of the summing junctions:

$$E = R - PU$$
$$U = D + CE.$$

Assemble into a vector equation:

$$\begin{bmatrix} 1 & P \\ -C & 1 \end{bmatrix} \begin{bmatrix} E \\ U \end{bmatrix} = \begin{bmatrix} R \\ D \end{bmatrix}.$$

In view of our standing assumption (P strictly proper, C proper), the determinant of

$$\begin{bmatrix} 1 & P \\ -C & 1 \end{bmatrix}$$

is not identically zero.

Exercise 2.2. Show that Assumption 2.4.1 implies that $1 + PC$ is not identically zero.

Thus we can solve for E, U:

$$\begin{bmatrix} E \\ U \end{bmatrix} = \begin{bmatrix} 1 & P \\ -C & 1 \end{bmatrix}^{-1} \begin{bmatrix} R \\ D \end{bmatrix} = \begin{bmatrix} \frac{1}{1+PC} & \frac{-P}{1+PC} \\ \frac{C}{1+PC} & \frac{1}{1+PC} \end{bmatrix} \begin{bmatrix} R \\ D \end{bmatrix}.$$

The output is given by

$$Y = PU = \frac{PC}{1+PC} R + \frac{P}{1+PC} D.$$

We just derived the following closed-loop transfer functions:

$$R \text{ to } E : \frac{1}{1+PC}, \qquad R \text{ to } U : \frac{C}{1+PC}, \qquad R \text{ to } Y : \frac{PC}{1+PC},$$
$$D \text{ to } E : \frac{-P}{1+PC}, \qquad D \text{ to } U : \frac{1}{1+PC}, \qquad D \text{ to } Y : \frac{P}{1+PC}.$$

Since, whenever r and e are bounded, so is $y = r - e$, it suffices to look at the 4 transfer functions from (r, d) to (e, u), namely,

$$\begin{bmatrix} \frac{1}{1+PC} & \frac{-P}{1+PC} \\ \frac{C}{1+PC} & \frac{1}{1+PC} \end{bmatrix}.$$

Remark 2.4.3. An alternative definition for input-output stability of an interconnected system is: a feedback system is input-output stable if and only if *every* possible transfer function in the system (imagine introducing noise everywhere) is BIBO stable. ♦

Example 2.4.1.

$$P(s) = \frac{1}{s^2-1}, \qquad C(s) = \frac{s-1}{s+1}.$$

The 4 transfer functions are

$$\begin{bmatrix} E \\ U \end{bmatrix} = \begin{bmatrix} \dfrac{(s+1)^2}{s^2+2s+2} & \dfrac{s+1}{(s-1)(s^2+2s+2)} \\ \dfrac{(s+1)(s-1)}{s^2+2s+2} & \dfrac{(s+1)^2}{s^2+2s+2} \end{bmatrix} \begin{bmatrix} R \\ D \end{bmatrix}.$$

Three of these are stable; the one from D to E is not. Consequently, the feedback system is not input-output stable. This is in spite of the fact that a bounded r produces a bounded y. The problem here is that C cancels an unstable pole of P. That isn't allowed. ▲

Example 2.4.2.

$$P(s) = \frac{1}{s-1}, \qquad C(s) = K$$

The feedback system is input-output stable if and only if $K > 1$ (check). This shows that a feedback system can be stable even if the individual components are unstable. ▲

We now provide a way to test for input-output stability. Write

$$P = \frac{N_p}{D_p}, \qquad C = \frac{N_c}{D_c}.$$

We assume the pair $(N_p,\ D_p)$ is coprime, i.e., have no common roots, and $(N_c,\ D_c)$ is coprime too.

Definition 2.4.4. The **characteristic polynomial** of the system in Figure 2.6 is $\pi(s) \coloneqq D_pD_c + N_pN_c$.

The characteristic polynomial (ch.p.) of a feedback system is the least common multiple of the denominators in the four transfer functions from (r, d) to (e, u), i.e.,

$$\begin{bmatrix} \frac{1}{1+PC} & \frac{-P}{1+PC} \\ \frac{C}{1+PC} & \frac{1}{1+PC} \end{bmatrix} = \frac{1}{D_pD_c + N_pN_c} \begin{bmatrix} D_pD_c & -N_pD_c \\ D_pN_c & D_pD_c \end{bmatrix}. \tag{2.3}$$

Theorem 2.4.5. *The feedback system is input-output stable if, and only if, the characteristic polynomial has no roots with* $\mathrm{Re}(s) \geq 0$.

Example 2.4.3. (Unstable Pole-Zero Cancellation)

$$P(s) = \frac{1}{s^2-1}, \qquad C(s) = \frac{s-1}{s+1}.$$

We have seen that with this plant and controller combination the system is not input-output stable because the transfer function from d to e is unstable. Notice the unstable pole-zero cancellation. The characteristic polynomial is

$$\pi(s) = s - 1 + (s^2-1)(s+1) = (s-1)(s^2+2s+2).$$

This has a right half-plane root – precisely the root that the controller cancels. ▲

Definition 2.4.6. The plant $P(s)$ and controller $C(s)$ have a **pole-zero cancellation** if there exists a $\lambda \in \mathbb{C}$ such that

$$N_p(\lambda) = D_c(\lambda) = 0 \qquad \text{(controller pole cancels a plant zero)}$$
$$D_p(\lambda) = N_c(\lambda) = 0 \qquad \text{(controller zero cancels a plant pole).}$$

It is called an **unstable pole-zero cancellation** if $\operatorname{Re}(\lambda) \geq 0$.

Corollary 2.4.7. *If there is an unstable pole-zero cancellation, then the feedback system is not input-output stable.*

Proof. If there is an unstable pole-zero cancellation at $\lambda \in \overline{\mathbb{C}}^+$, then

$$\pi(\lambda) = D_p(\lambda)D_c(\lambda) + N_p(\lambda)N_c(\lambda) = 0 + 0.$$

Thus π has a root in $\overline{\mathbb{C}}^+$ so that by Theorem 6.7.4 the system is not input-output stable. ■

Remark 2.4.8. In practice, it is not possible to exactly cancel a plant zero or pole because of modelling errors. In Example 2.4.1 therefore the transfer function from r to y will also be unstable. However, it is important to stress that even in the ideal case with a perfect pole-zero cancellation, we still get an unstable system. ♦

A second way to test feedback stability is as follows.

Theorem 2.4.9. *The feedback system is input-output stable if, and only if,*

1. *The transfer function $1 + PC$ has no zeros in $\operatorname{Re}(s) \geq 0$, and*
2. *the product PC has no unstable pole-zero cancellations.*

Exercise 2.3. Prove Theorem 2.4.9.

Example 2.4.4.

$$P(s) = \frac{1}{s^2 - 1}, \qquad C(s) = \frac{s - 1}{s + 1}$$

Check that 1) holds but 2) does not. ▲

Remark 2.4.10. Occasionally we will encounter non-unity feedback systems as shown in Figure 2.7. This is

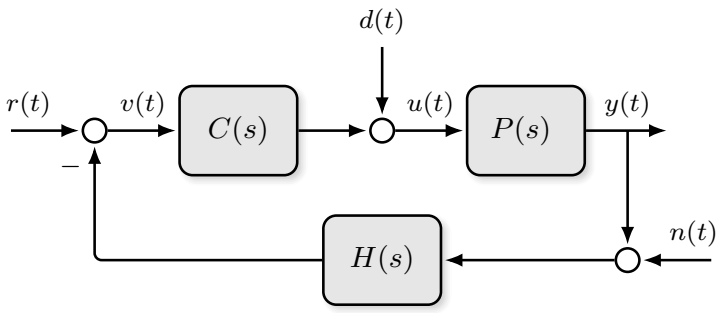


Figure 2.7: Non-unity feedback system.

the case when we model the dynamics of the sensor that provides the feedback as a transfer function $H(s)$. The

exogenous signal n models sensor noise. If $P(s)$, $C(s)$ and $H(s)$ all belong to $\mathbb{R}(s)$, i.e., they're real rational, then we can write

$$P = \frac{N_p}{D_p},\ C = \frac{N_c}{D_c},\quad H = \frac{N_h}{D_h}$$

and you can check that the characteristic polynomial becomes

$$\pi(s) = D_c D_p D_h + N_c N_p N_h.$$

In this case we have feedback stability if, and only if, all the roots of $\pi(s)$ are in $\mathbb{C}^-$. ♦

Exercise 2.4. Find the 9 transfer functions from (r, d, n) to (v, u, y).

2.5 Time-domain response

In practice we are usually interested in evaluating the output response of a system in the time-domain. In the analysis problem, an input signal is applied to the system, and the performance of the system is evaluated by studying the system response in the time-domain. The setup we have in mind is shown in Figure 2.8. Typical input signals for the time response are steps and ramps.

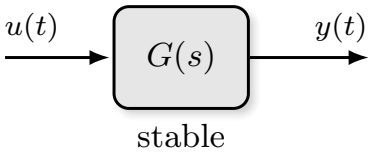


Figure 2.8: Block diagram for evaluating system time response.

The time response $y(t)$ of a BIBO stable linear time-invariant control system can be divided into two parts: the **transient response** $y_t(t)$ and the **steady-state response** $y_{ss}(t)$. The transient response is the part of the response that goes to zero as time becomes large. The steady-state response is the part of the total response that persists after the transient has died out.

2.5.1 Steady-state response and the final-value theorem

The Final-Value Theorem (F.V.T.) can sometimes be used to find the steady-state values of signals.

Theorem 2.5.1 (Final-value theorem). *Let $f(t)$ be a signal defined for $t \geq 0$ and let its Laplace transform $F(s)$ be rational and proper.*

(a) If $F(s)$ has all its poles in $\mathbb{C}^-$, then $f(t)$ converges to zero as $t \to \infty$.

(b) If $F(s)$ has all its poles in $\mathbb{C}^-$ except for a simple pole at $s = 0$, then

$$\lim_{t\to\infty} f(t) = \lim_{s\to 0} sF(s). \tag{2.4}$$

(c) In all other cases, $f(t)$ does not approach a constant as $t \to \infty$.

Remember, you have to *know* that $f(t)$ has a final value, by examining the poles of $F(s)$, before you can apply the final value theorem. Equation (2.4) is valid if, and only if, $sF(s)$ has no poles in the closed right half complex plane $\overline{\mathbb{C}}^+$.

Example 2.5.1. Consider a system modeled by a transfer function

$$Y(s) = G(s)U(s), \qquad G(s) = \frac{s+80}{(s+4)(s+10)}.$$

The system is BIBO stable by Theorem 2.3.3 since all of its poles are in $\mathbb{C}^-$. If we apply a unit step input then $U(s) = 1/s$ and $y(t)$ is bounded. Since all the poles of $sY(s)$ are in $\mathbb{C}^-$ we are in case (b) of the final value theorem, whence

$$\begin{aligned}\lim_{t\to\infty} y(t) &= \lim_{s\to 0} sY(s)\\ &= \lim_{s\to 0} sG(s)U(s)\\ &= \lim_{s\to 0} G(s)\\ &= G(0)\\ &= 2.\end{aligned}$$

▲

As an application of the FVT, consider the usual feedback system with a given reference signal $r(t)$ which represents the desired value of the system output. The **tracking error** is defined as $e(t) \coloneqq r(t) - y(t)$. The **steady-state tracking error** equals $e_{\text{ss}} \coloneqq \lim_{t\to\infty} e(t)$. The final value theorem can often be used to find e_{ss}.

Another application is the following: the FVT allows us to determine BIBO stability of an LTI system by taking a single measurement.

Corollary 2.5.2. *A linear time-invariant system with a proper and rational transfer function is BIBO stable if, and only if, its unit step response approaches a constant value.*

Exercise 2.5. Use the Final-Value Theorem to prove Corollary 2.5.2.

2.5.2 Transient response due to step inputs

The key transient characteristics of a step response are illustrated in Figure 2.9.

Prototype second order system

While the quantities in Figure 2.9 can be experimentally found for a system of any order, in the case of the prototype second order system we can obtain *closed-form* expressions for the various characteristics. These closed-form expressions motivate many of our design equations.

Definition 2.5.3. The **prototype second order system** with input u and output y is governed by the differential equation

$$\ddot{y} + 2\zeta\omega_n\dot{y} + \omega_n^2 y = K\omega_n^2 u, \qquad K, \zeta, \omega_n \in \mathbb{R}. \tag{2.5}$$

The associated transfer function is

$$\frac{Y(s)}{U(s)} = \frac{K\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2}. \tag{2.6}$$

The prototype second order system is important because it's easy to analyse and has far richer dynamics than a first order system. This makes it a workhorse for obtaining simple models of many engineering systems.

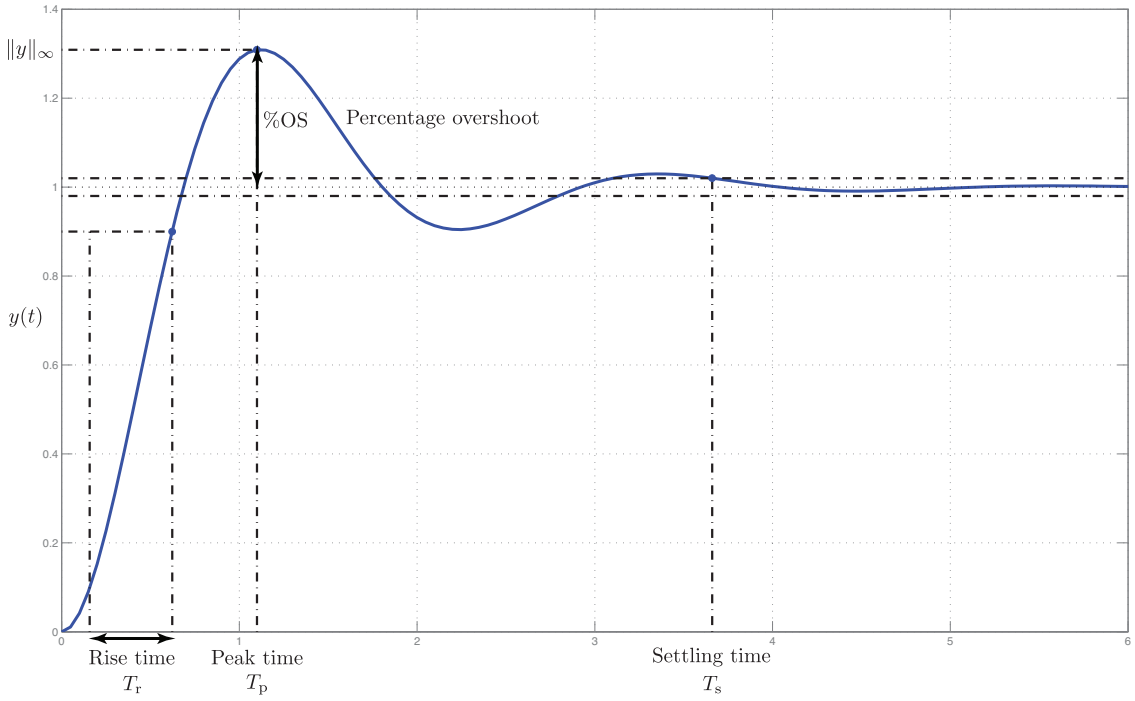


Figure 2.9: Measures of step response performance.

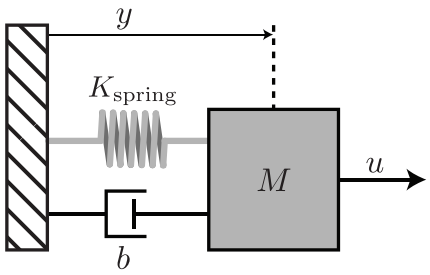


Figure 2.10: Mass-spring-damper system.

Example 2.5.2. (Mass-Spring-Damper) The mass spring damper in Figure 2.10 from Example 2.2.3 has transfer function

$$\frac{Y(s)}{U(s)} = \frac{1}{Ms^2 + bs + K_{\text{spring}}}.$$

Comparing this transfer function to the prototype second order system (2.6) we deduce

$$\omega_n = \sqrt{\frac{K_{\text{spring}}}{M}}, \qquad K = \frac{1}{K_{\text{spring}}}, \qquad \zeta = \frac{b}{2\sqrt{K_{\text{spring}} M}}.$$

▲

Example 2.5.3. (RLC Circuit) Consider the series RLC circuit in Figure 2.11 (cf. Example 2.2.4). The system TF is (verify!)

$$\frac{Y(s)}{U(s)} = G(s) = \frac{1}{s^2 LC + sRC + 1} = \frac{\frac{1}{LC}}{s^2 + \frac{R}{L}s + \frac{1}{LC}}.$$

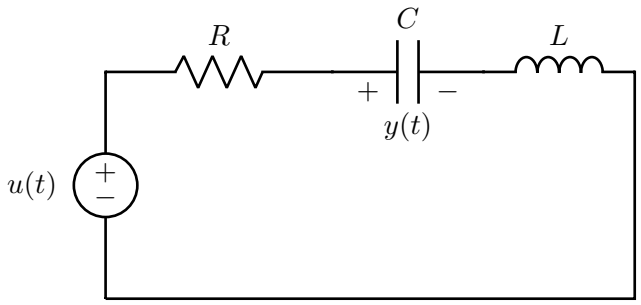


Figure 2.11: RLC circuit with voltage across capacitor taken as output.

Comparing this transfer function to the prototype second order system (2.6) we deduce

$$\omega_n = \frac{1}{\sqrt{LC}}, \qquad K = 1, \qquad \zeta = \frac{R}{2}\sqrt{\frac{C}{L}}.$$

▲

The parameter ζ is a useful way to categorize the prototype second order system and has its own name.

Definition 2.5.4. The **damping ratio** of the prototype second order system is the parameter ζ in (2.6). The **natural undamped frequency** of the prototype second order system is the parameter ω_n in (2.6). The prototype second or system is called

- **undamped** if $\zeta = 0$,
- **underdamped** if $0 < \zeta < 1$,
- **critically damped** if $\zeta = 1$,
- **overdamped** if $\zeta > 1$.

The step responses for an underdamped, critically damped and overdamped prototype second order system are shown in Figure 2.12. Table 2.3 lists key properties of the prototype second order system (2.6).

Exercise 2.6. Suppose that $L = 10\,\mu\text{H}$ and $C = 160\,\mu\text{F}$ in the RLC circuit from Example 2.5.3. Find the range of resistance values so that the circuit is underdamped.

Table 2.3: Properties of prototype second order system (2.6).

Name	Poles	Zeros	Steady-state gain	Step response
Underdamped	$-\zeta\omega_n \pm j\omega_n\sqrt{1-\zeta^2}$	None	K	Oscillatory
Critically damped	$-\omega_n$ (repeated)	None	K	Not oscillatory
Overdamped	$-\zeta\omega_n \pm \omega_n\sqrt{\zeta^2-1}$	None	K	Not oscillatory

Figure 2.13 shows the pole locations of a stable underdamped second order system and how they depend on the parameters ζ, ω_n. The percentage overshoot in the step response of an underdamped prototype second order system (2.6) is given by the formula

$$\%\text{OS} = \text{e}^{\frac{-\zeta\pi}{\sqrt{1-\zeta^2}}} \quad \text{(underdamped prototype second order system).} \tag{2.7}$$

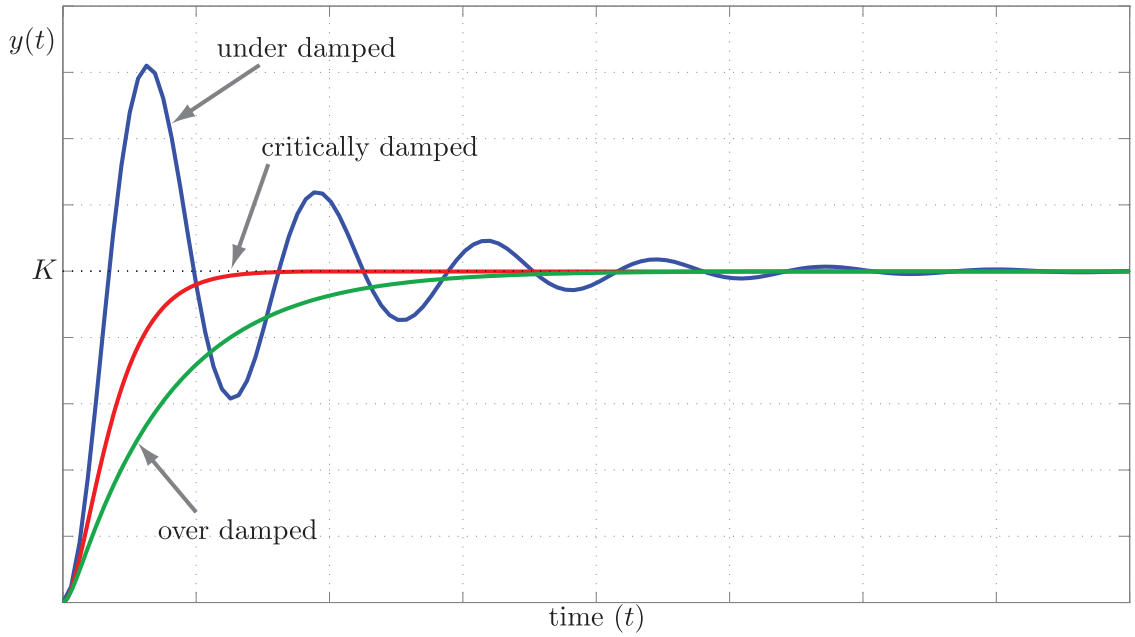


Figure 2.12: Representative step responses of the prototype second order system for different values of ζ and fixed ω_n.

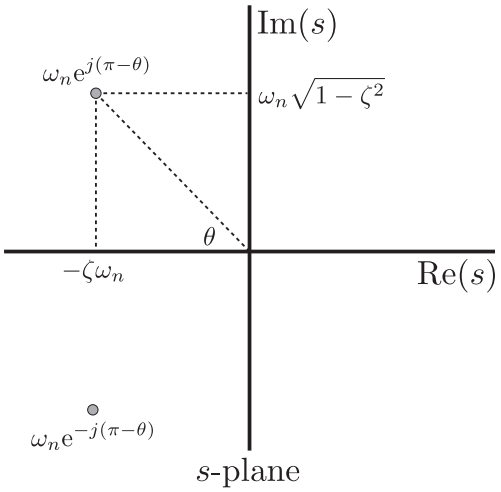


Figure 2.13: Pole location of an underdamped second order system. In this figure $\theta = \arccos(\zeta)$.

Notice that the percentage overshoot only depends on the damping ratio ζ and not on ω_n. We rearrange (2.7) to obtain the percentage overshoot as a function of the damping ratio

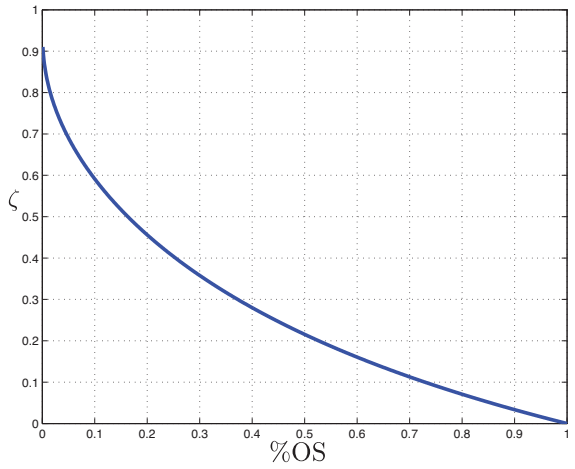


Figure 2.14: Damping ratio as a function of %OS (2.8).

$$\zeta = -\frac{\ln(\%\mathrm{OS})}{\sqrt{\pi^2 + (\ln(\%\mathrm{OS}))^2}}. \tag{2.8}$$

Observe that the more damping (larger ζ) the less overshoot. We can use this observation to help us impose constraints on a system's damping ratio as follows. Let $\%\mathrm{OS}^{\max}$ denote the maximum allowable overshoot. Then using (2.8) the specification $\%\mathrm{OS}^{\max}$ is translated into a specification on the damping ratio

$$\%\mathrm{OS} \leq \%\mathrm{OS}^{\max} \iff \zeta \geq -\frac{\ln(\%\mathrm{OS}^{\max})}{\sqrt{\pi^2 + (\ln(\%\mathrm{OS}^{\max}))^2}}.$$

Lastly, since $\zeta = \cos(\theta)$ where θ is the angle in Figure 2.13,

$$\%\mathrm{OS} \leq \%\mathrm{OS}^{\max} \iff \theta \leq \arccos\left(-\frac{\ln(\%\mathrm{OS}^{\max})}{\sqrt{\pi^2 + (\ln(\%\mathrm{OS}^{\max}))^2}}\right) =: \theta^{\max}. \tag{2.9}$$

Therefore, in order to meet an overshoot specification, the second order system must have all its poles in a "cone" in the left half of the s-plane

$$\{s \in \mathbb{C} : |\arg(s)| \geq \pi - \theta^{\max}\}$$

where $\arg(s) \in (-\pi, \pi]$ returns the principle argument of a complex number.

Example 2.5.4. Suppose that $L = 10\,\mu\mathrm{H}$ and $C = 160\,\mu\mathrm{F}$ in the RLC circuit from Example 2.5.3. Find the range of resistor values $R > 0$ so that the system is (i) stable (ii) underdamped and (iii) its step response satisfies $\%\mathrm{OS} \leq 0.05$.

From Exercise 2.6 we know that (i) and (ii) are satisfied if $R < 0.5$. In this example $\%\mathrm{OS}^{\max} = 0.05$ so that (iii) is satisfied if

$$\zeta \geq -\frac{\ln(0.05)}{\sqrt{\pi^2 + (\ln(0.05))^2}} = 0.69.$$

Using the expression for ζ from Example 2.5.3 we obtain the constraint

$$\zeta = \frac{R}{2}\sqrt{\frac{C}{L}} = 2R \geq 0.69 \iff R \geq 0.345.$$

In summary, the design specifications are met if $0.345 \leq R < 0.5$. ▲

Exercise 2.7. Draw the region in the s-plane in which the poles of an underdamped second order system must lie so that $\%\mathrm{OS} < 0.05$.

For an underdamped second order system we can crudely estimate the settling time with[4]

$$T_{\mathrm{s}} \approx \frac{4}{\zeta \omega_n} \quad \text{(underdamped prototype second order system).} \tag{2.10}$$

From this relationship we see that the larger the ω_n, the faster the response. Note that T_{s} only depends on the real part of the poles of $G(s)$.

Suppose our design specification is that the settling time must be less than $T_{\mathrm{s}}^{\max}$. This means that

$$T_{\mathrm{s}} \leq T_{\mathrm{s}}^{\max} \;\Leftrightarrow\; \zeta \omega_n \geq \frac{4}{T_{\mathrm{s}}^{\max}}.$$

The poles of a stable underdamped second order system have real part $-\zeta\omega_n$ (Figure 2.13, Table 2.3). Therefore, in order to meet a settling time specification, the second order system should have all its poles sufficiently far to the left of the s-plane

$$\left\{ s \in \mathbb{C} : \mathrm{Re}\,(s) \leq -\frac{4}{T_{\mathrm{s}}^{\max}} \right\}.$$

Exercise 2.8. Find the settling time for the system from Example 2.5.4 when $R = 0.4\,\Omega$.

The time to peak for an underdamped prototype second order system is given by

$$T_{\mathrm{p}} = \frac{\pi}{\omega_n \sqrt{1-\zeta^2}} \quad \text{(underdamped prototype second order system).} \tag{2.11}$$

Note that T_{p} only depends on the imaginary part of the poles for an underdamped second order system. Also observe that the larger the bandwidth ($\omega_{\mathrm{BW}} \approx \omega_n$), the smaller T_{p} is and the faster the response becomes.

Exercise 2.9. Draw the region in the s-plane in which the poles of an underdamped second order system must lie so that $T_{\mathrm{p}} < 3$ seconds.

2.6 Frequency response

If you apply a sinusoidal input to a BIBO stable LTI system, then, in steady-state, the output is also a sinusoid of the same frequency with a possible change in magnitude and phase. This is illustrated in Figure 2.15.

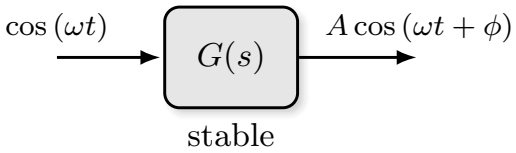


Figure 2.15: Steady-state output of a BIBO stable LTI system excited by a sinusoidal input.

[4] It can be shown that the 2% settling time for a critically damped system is approximately given by $T_{\mathrm{s}} \approx \frac{5.8}{\omega_n}$. Unless explicitly stated, we will use (2.10) for computing the settling time of second order systems.

Theorem 2.6.1. *Assume $G(s)$ is rational, proper, and has all its poles in $\mathbb{C}^-$. Then the steady-state response to the input $u(t) = ae^{j\omega t}$ is*

$$y(t) = aG(j\omega)e^{j\omega t}. \tag{2.12}$$

In particular

- *The steady-state response to the input $u(t) = a\cos(\omega t)$ is*

$$y(t) = a\,|G(j\omega)|\cos(\omega t + \measuredangle G(j\omega)). \tag{2.13}$$

- *The steady-state response to the input $u(t) = a\sin(\omega t)$ is*

$$y(t) = a\,|G(j\omega)|\sin(\omega t + \measuredangle G(j\omega)). \tag{2.14}$$

- *The steady-state response to the input $u(t) = a\mathbf{1}(t)$ is*

$$y(t) = aG(0). \tag{2.15}$$

Therefore the response of a BIBO stable LTI system to a continuous-time sinusoid input of frequency ω is also a continuous-time sinusoid of the same frequency. The amplitude of the output sinusoid is $|G(j\omega)|$ times the input amplitude, and the phase of the output is shifted by $\measuredangle G(j\omega)$ with respect to the input phase. This justifies the picture in Figure 2.15 and leads to the following definition.

Definition 2.6.2. Assume $G(s)$ is rational, proper, and has all its poles in $\mathbb{C}^-$.

(a) The function $\mathbb{R} \to \mathbb{C}$, $\omega \mapsto G(j\omega)$ is the **frequency response** of G.

(b) The function $\mathbb{R} \to \mathbb{R}$, $\omega \mapsto |G(j\omega)|$ is the **amplitude** or **magnitude response** of G.

(c) The function $\mathbb{R} \to (-\pi, \pi]$, $\omega \mapsto \measuredangle G(j\omega)$ is the **phase response** of G.

Hence, the frequency response of a BIBO stable continuous-time LTI system $G(s)$ is determined by setting $s = j\omega$ and varying ω. Here's the important point: Under the stated assumptions about the system ($G(s)$ is rational, proper, and all poles in $\mathbb{C}^-$), the region of convergence of the Laplace integral includes the imaginary axis. Therefore it is legitimate to set $s = j\omega$ in $G(s)$ to get $G(j\omega)$, which, by the way, equals the Fourier transform of the impulse response $g(t)$.

2.6.1 Graphical representations of the frequency response

One of the reasons why frequency response is a powerful tool is that it is possible to succinctly understand its primary features by means of plotting functions or parameterized curves. In this section we review this.

Example 2.6.1. Consider a system with transfer function

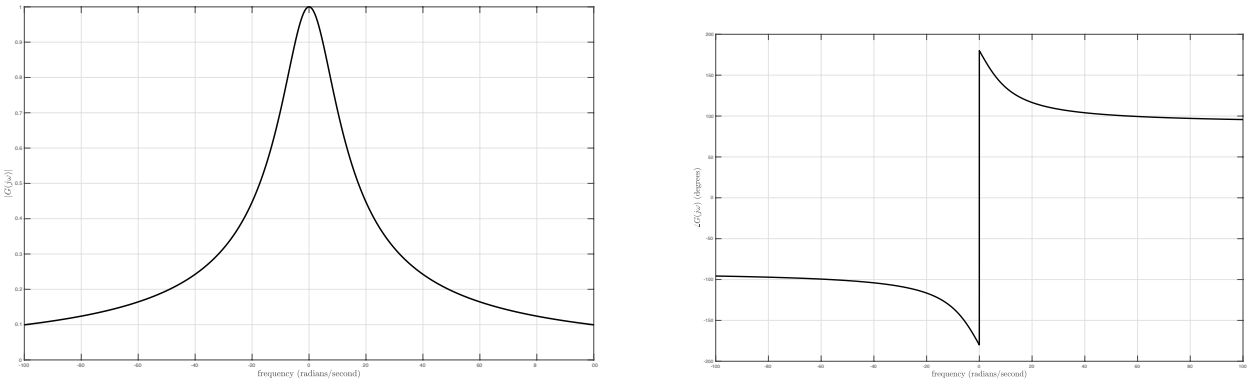
$$G(s) = \frac{-10}{s+10}.$$

Its frequency response is the function

$$G(j\omega) = \frac{-10}{j\omega+10}.$$

The magnitude and phase responses are plotted in Figure 2.16 for $\omega \in [-100, 100]$. The basic plots can be generated in MATLAB using the script below.

```
w  = -100:0.01:100;
s = 1i.*w;
G = -10./(s+10);
plot(w, abs(G), 'LineWidth', 2); grid on;
figure; plot(w, rad2deg(angle(G)), 'LineWidth', 2); grid on;
```



(a) Magnitude response. *(b) Phase response.*

Figure 2.16: A plot of the magnitude and phase response for Example 2.6.1.

▲

Remark 2.6.3. From now on, when we graphically represent the functions from Definition 2.6.2 we will only consider non-negative values of ω. One may think that we are losing information by not considering negative values of frequency. This is not the case because when a system's impulse response $g(t) = \mathscr{L}^{-1}(G(s))$ is real-valued, its magnitude response is an even function, i.e., $|G(j\omega)| = |G(-j\omega)|$ and its phase response is an odd function, i.e., $\measuredangle(G(j\omega)) = -\measuredangle(G(-j\omega))$, c.f., Figure 2.16. ♦

A Bode plot is simply a plot of the amplitude and phase response functions of G, see Definition 2.6.2. These plots however are done in a very particular way: we plot $20\log|G(j\omega)|$ vs ω and $\measuredangle G(j\omega)$ vs ω with ω plotted on log scales (all logarithms are base 10 in this course). The two plots (i) $20\log|G(j\omega)|$ vs $\log(\omega)$ and (ii) $\measuredangle G(j\omega)$ vs $\log(\omega)$ together are called the **Bode plot** of $G(s)$. The units of the plot of $20\log|G(j\omega)|$ are called **decibels** (dB).

A Bode plot tells us how the system $G(s)$ responds in steady-state to sinusoidal inputs. Since the Fourier series can be used to represent a large class of signals, Bode plots can be used to tell us about a system response to "almost all" inputs.

Example 2.6.2. (RLC Circuit) In this example we'll use software to create the Bode plot of the RLC circuit from Example 2.2.4. You should know how to sketch them by hand. The governing equation for the circuit is

$$-\dot{u} + R\dot{y} + L\ddot{y} + \frac{1}{C}y = 0.$$

The TF of this system is (verify)

$$\frac{Y(s)}{U(s)} = G(s) = \frac{s}{Ls^2 + Rs + \frac{1}{C}}$$

and the frequency response is

$$G(j\omega) = \frac{j\omega}{-jL\omega^2 + jR\omega + \frac{1}{C}}.$$

Figure 2.17 shows the Bode plot for this system and was generated using the following script in MATLAB

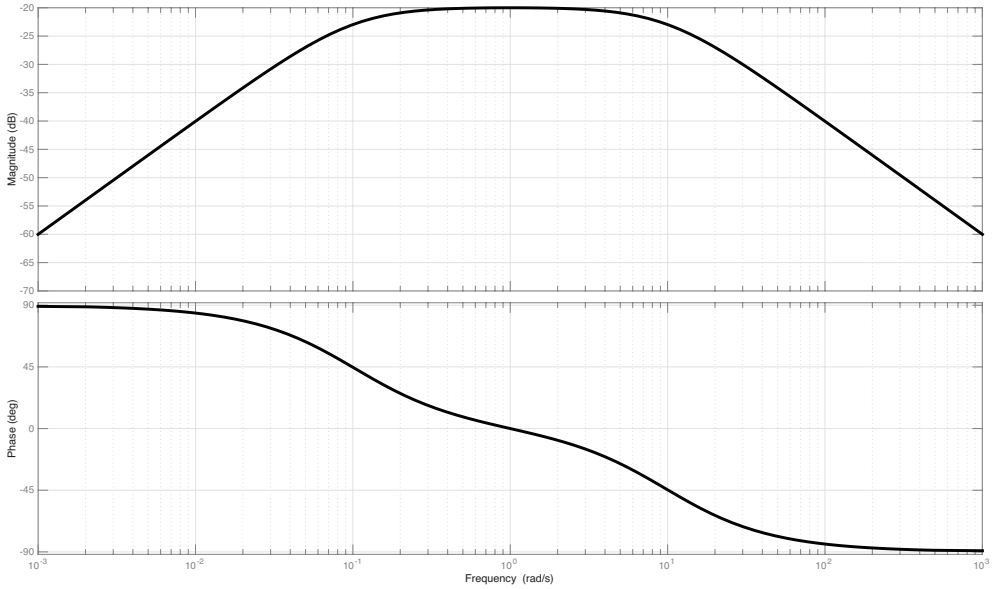


Figure 2.17: Bode plot of RLC circuit with $R = 10$, $L = 1$, $C = 1$.

```
s = tf('s');
R =10; L = 1; C = 1;
G = s/(L*s^2 + R*s + 1/C);
bode(G);
grid on;
```

▲

Example 2.6.3. (Low Pass Filter) Recall the TF of the so-called low pass RC filter from Example 2.2.2

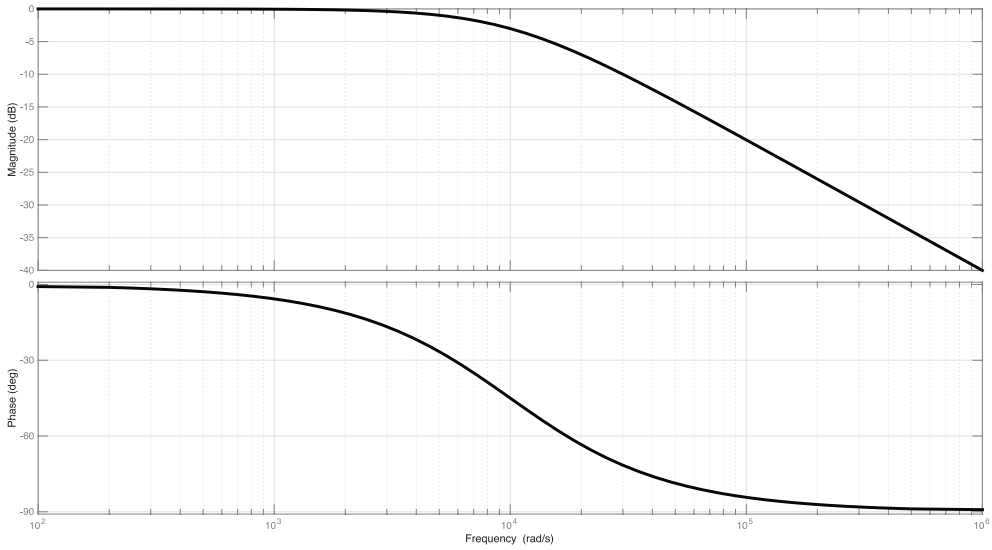


Figure 2.18: Bode plot of RC circuit with $RC = 10^{-4}$.

$$\frac{Y(s)}{U(s)} = G(s) = \frac{1}{RCs + 1}.$$

The associated frequency response is

$$G(j\omega) = \frac{1}{jRC\omega + 1}.$$

Figure 2.18 shows the Bode plot for this system with $RC = 10^{-4}$. The Bode plot provides justification for the name low pass filter. From the plot of the magnitude response we see that when we apply sinusoidal inputs with frequencies above 10^4 rad/s, the amplitude of the steady-state output signal is heavily attenuated. For instance, if the input has amplitude one and frequency $\omega = 10^5$, then $20\log|G(j\omega)| \approx -20$dB and therefore the amplitude of the steady-state output is $|G(j\omega)| \approx 10^{-1}$. ▲

The above example helps to motivate the notion of bandwidth for control systems. The term bandwidth is used by engineers in many different contexts like (i) bandwidth of filters (ii) bandwidth of communication channels (iii) bandwidth of control systems. In control systems, the higher the bandwidth the better because, generally speaking, it means a faster response.

Definition 2.6.4. Let $G(s)$ be a rational, proper transfer function with all its poles in $\mathbb{C}^-$. Let $\|G\|_\infty := \sup_{\omega\in[0,\infty)} |G(j\omega)|$ denote the maximum magnitude of $G(j\omega)$. The **bandwidth** of G is the width of the frequency range in $[0,\infty)$ in which, for every ω in this range

$$|G(j\omega)| \geq \frac{1}{\sqrt{2}}\|G\|_\infty.$$

In terms of decibels, for every ω in this frequency range, $20\log|G(j\omega)| - 20\log\|G\|_\infty \geq -3$dB.

Example 2.6.4. The Bode plot in Figure 2.17 of Example 2.6.2 shows that the RLC circuit has a bandwidth of 9.9 radians/second. Input signals with frequencies outside of the interval $[0.1, 10]$ are heavily attenuated. ▲

Example 2.6.5. The Bode plot in Figure 2.18 of Example 2.6.3 shows that the RC circuit has a bandwidth of 10^4 radians/second. Input signals with frequencies outside of the interval $[0, 10^4]$ are heavily attenuated. ▲

When, as in Example 2.6.5, the maximum magnitude of $G(j\omega)$ occurs at $\omega = 0$, we will often say that a system's bandwidth is simply ω_{BW} with the understanding that this refers to the width of the interval $[0, \omega_{\text{BW}}]$. Such systems allow input signals in the frequency range $[0, \omega_{\text{BW}}]$ to pass through without much attenuation. The bandwidth is easily obtained from the amplitude response plot of $G(s)$ as illustrated in Figure 2.19.

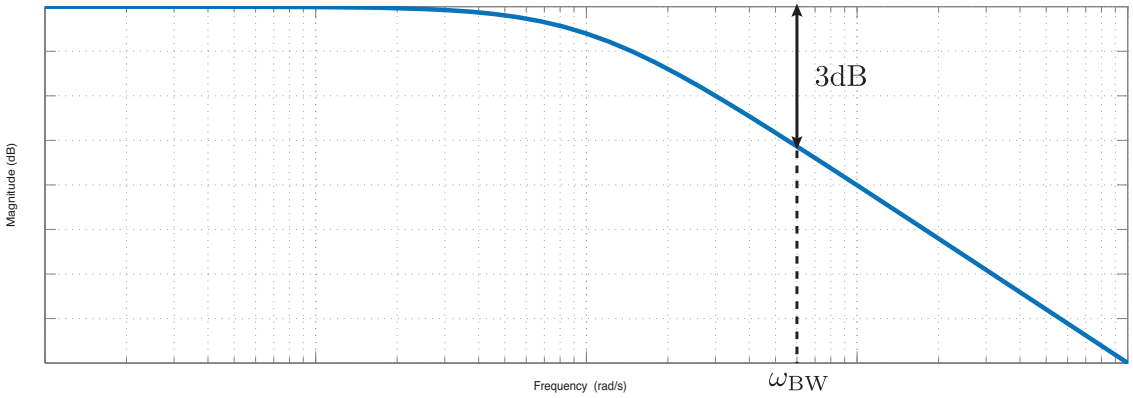


Figure 2.19: Obtaining the bandwidth of a system from its amplitude response.

Remark 2.6.5. The significance of the ratio $1/\sqrt{2}$ (-3dB) in Definition 2.6.4 has its origins in the notion of signal power. The instantaneous power in a signal $u(t)$ is proportional to $u^2(t)$. If we apply an input signal

of frequency ω_{BW} to a low pass system with $G(0) = 1$, then the output signal will have instantaneous power proportional to $(u(t)/\sqrt{2})^2 = u^2(t)/2$. In other words, the output will have *half* the instantaneous power of the input signal. ♦

2.7 Dominant poles and zeros

The formulas we derived in Section 2.5.2 apply to second order systems. In general, it is very difficult to obtain closed-form formulas to evaluate the performance of higher order systems. However, in many cases, there are natural separations among the system poles and zeros. For example, some poles and zeros are much closer to the imaginary axis than others as illustrated in Figure 2.20.

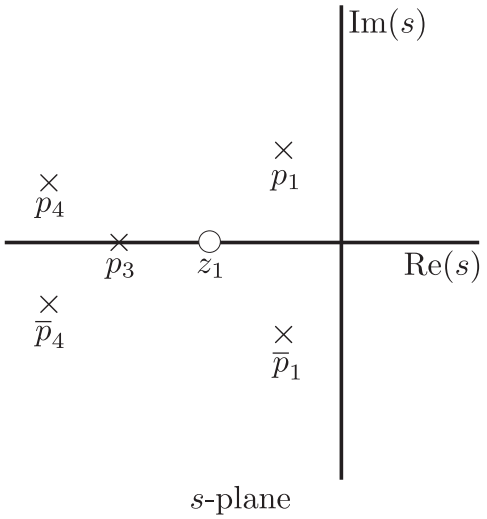


Figure 2.20: Pole-zero configuration with p_1, $\bar{p}_1$ as dominant poles.

Poles and zeros "far to the left" do not have a significant impact on the low frequency response[5] of a system. Hence good low-order approximations can be found by appropriately neglecting the less significant poles and zeros. Typically, we neglect poles and zeros that are at least 5 times further away from the imaginary axis. The poles and zeros close to the imaginary axis are called **dominant poles** and **dominant zeros**. The poles and zeros at least 5 times further away from the imaginary axis are called **non-dominant poles** and **non-dominant zeros**.

Example 2.7.1. (Model Reduction)

$$\begin{aligned} G(s) &= \frac{s+10}{(s+11)(s+12)(s^2+2s+2)} \\ &= \underbrace{\frac{s+10}{(s+11)(s+12)}}_{=:G_{\text{fast}}(s)} \underbrace{\frac{1}{s^2+2s+2}}_{=:G_{\text{slow}}(s)}. \end{aligned}$$

The portion of the step response due to $G_{\text{fast}}(s)$ decays to zero much faster than the portion of the time response due to $G_{\text{slow}}(s)$. This motivates us to approximate $G_{\text{fast}}(s)$ by its steady-state gain

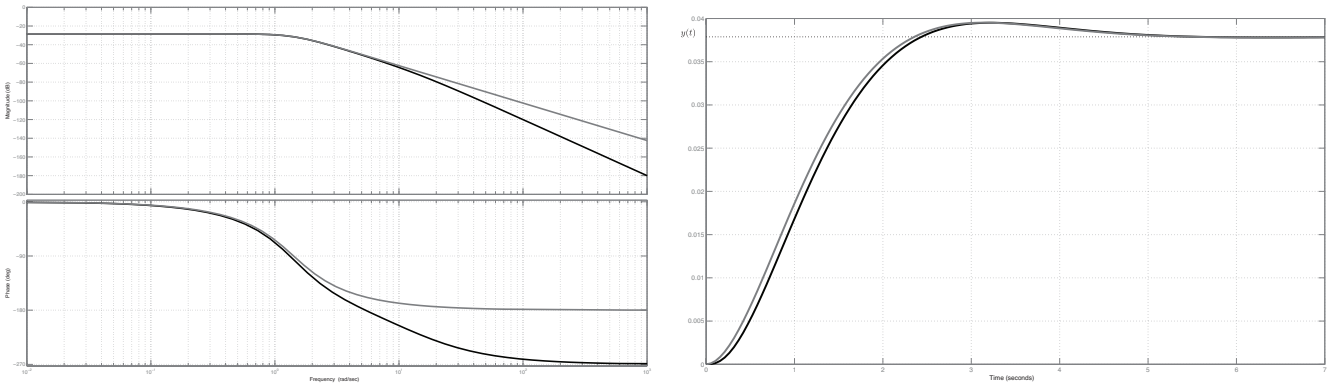
$$G_{\text{fast}}(0) = \frac{10}{132}$$

[5]The Bode plot is affected at high frequencies where the gain is typically small.

and hence

$$G(s) \approx \frac{10}{132}\frac{1}{s^2+2s+2} =: \hat{G}(s).$$

Figure 2.21a shows the frequency response of $G(s)$ and $\hat{G}(s)$. As expected, at low frequencies the approximation is reasonably good. The step responses for the actual and approximated systems are shown in Figure 2.21b.



(a) Bode plot. *(b) Step response.*

Figure 2.21: Bode and step responses of $G(s)$ (black) and $\hat{G}(s)$ (grey) from Example 2.7.1.

▲

When reducing the order of system model it is important to decide the frequency range in which the approximation should be valid.

Example 2.7.2. Consider the transfer function

$$G(s) = 100\frac{\left(1+\frac{1}{60}s\right)\left(1+\frac{1}{900}s\right)}{\left(1+\frac{1}{100}s\right)\left(1+\frac{1}{500}s\right)\left(1+\frac{1}{600}s\right)\left(1+\frac{1}{1000}s\right)}.$$

We want to obtain a model that describes the process well in the frequency range $\omega \leq 600$. In this case

$$G_{\text{fast}}(s) = \frac{1+\frac{1}{900}s}{1+\frac{1}{1000}s}$$

and the approximated transfer function is

$$\hat{G}(s) = 100\frac{\left(1+\frac{1}{60}s\right)}{\left(1+\frac{1}{100}s\right)\left(1+\frac{1}{500}s\right)\left(1+\frac{1}{600}s\right)}.$$

▲

2.8 Tracking reference signals

Cruise control in a car regulates the speed to a prescribed set point. What is the principle underlying its operation? The answer lies in the final value theorem (FVT).

Example 2.8.1. Consider the unity feedback system in Figure 2.22 with

$$P(s) = \frac{1}{s+1}, \qquad C(s) = \frac{1}{s}.$$

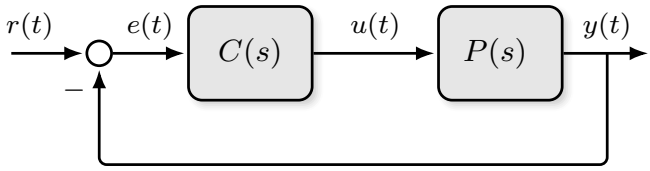


Figure 2.22: System for Example 2.8.1.

Let r be a constant, $r(t) = r_0\mathbf{1}(t)$, $r_0 \in \mathbb{R}$. Then we have

$$\begin{aligned} E(s) &= \frac{1}{1+P(s)C(s)}R(s) \\ &= \frac{s(s+1)}{s^2+s+1}\frac{r_0}{s} \\ &= \frac{s+1}{s^2+s+1}r_0 \end{aligned}$$

The FVT applies to $E(s)$, and $e(t) \to 0$ as $t \to \infty$. Thus the feedback system provides asymptotic tracking of step reference signals with zero steady-state error.

How it works: $C(s)$ contains an internal model of $R(s)$ (i.e., an integrator); closing the loop creates a zero in the TF from $R(s)$ to $E(s)$ exactly to cancel the unstable pole of $R(s)$. (This isn't an illegal pole-zero cancellation.) ▲

Remark 2.8.1. Tracking error is always defined by $e \coloneqq r - y$ where r is the command or reference input and y is the plant output. In a unity feedback control system the tracking error is fed back to the controller. If the sensor TF in the feedback path is not 1, as in Figure 2.7, then the signal coming out of the negative feedback summing junction is not the tracking error. ♦

Let's generalize the preceding example. Consider the unity feedback control system in Figure 2.6 and write

$$\frac{E(s)}{R(s)} = \frac{1}{1+CP} = \frac{D_pD_c}{D_pD_c + N_pN_c}. \tag{2.16}$$

Suppose that $R(s) = r_0/s$, i.e., $r(t) = r_0\mathbf{1}(t)$. If we *assume* that the controller $C(s)$ stabilizes the closed-loop system, then $\pi(s) = D_pD_c + N_pN_c$ has all its roots in $\mathbb{C}^-$, the TF (2.16) is BIBO stable, and the final-value theorem applies. Application of the FVT to the tracking error yields

$$e_{\text{ss}} = \lim_{t\to\infty} e(t) = \lim_{s\to 0} sE(s) = \lim_{s\to 0} s\frac{1}{1+C(s)P(s)}R(s) = \lim_{s\to 0}\frac{s}{1+C(s)P(s)}\frac{r_0}{s} = \lim_{s\to 0}\frac{1}{1+C(s)P(s)}.$$

Therefore $e_{\text{ss}} = 0$ if, and only if, $\lim_{s\to 0} P(s)C(s) = \infty$. This means that $P(s)C(s)$ has at least one pole at $s = 0$, i.e., at least one integrator.

If $P(s)$ does not have a pole at $s = 0$ and we want asymptotic step tracking, it is common to choose

$$C(s) = \frac{1}{s}C_1(s)$$

so that $C(s)P(s)$ has an integrator. Then $C_1(s)$ is designed to provide feedback stability. The integrator adds more phase lag though and this can make the overall system harder to stabilize.

Internal model principle

It is possible to generalize the above discussion to arbitrary reference signals.

Theorem 2.8.2 (Internal model principle)**.** *Assume that $P(s)$ is strictly proper, $C(s)$ is proper and the feedback system is stable. If $C(s)P(s)$ contains an internal model of the unstable part of $R(s)$, then perfect asymptotic tracking occurs.*

The internal model principle of Theorem 2.8.2 can be understood as follows. Suppose that the Laplace transform of the reference signal $r(t)$ is $R(s) = \mathscr{L}\{r(t)\}$. Write

$$R(s) = \frac{N_r(s)}{D_r(s)} = \frac{N_r(s)}{D_r^+(s)D_r^-(s)}$$

where the roots of the polynomial $D_r^-(s)$ are in $\mathbb{C}^-$ ($\mathrm{Re}\,(s) < 0$) while the roots of the polynomial $D_r^+(s)$ are in $\overline{\mathbb{C}}^+$ ($\mathrm{Re}\,(s) \geq 0$). The internal model principle says that in order for the plant output $y(t)$ to asymptotically converge to $r(t)$, the product $C(s)P(s)$ should have the form

$$C(s)P(s) = \frac{N(s)}{D(s)D_r^+(s)}$$

where $N(s), D(s) \in \mathbb{R}[s]$ are polynomials. That is, the product $C(s)P(s)$ must have a copy of $D_r^+(s)$ in its denominator. If this is the case and the closed-loop system is stable, then

$$sE(s) = s\frac{1}{1+C(s)P(s)}R(s) = s\frac{D(s)D_r^+(s)}{\pi(s)}\frac{N_r(s)}{D_r^+(s)D_r^-(s)} = s\frac{D(s)}{\pi(s)}\frac{N_r(s)}{D_r^-(s)}.$$

So $sE(s)$ has no poles with $\mathrm{Re}\,(s) \geq 0$ and by the FVT $e_{\mathrm{ss}} = 0$.

Example 2.8.2. Let

$$P(s) = \frac{1}{s+1}.$$

We want the closed-loop system to be stable and to track the reference signal

$$r(t) = r_0 \sin(t).$$

Then

$$R(s) = \frac{r_0}{s^2+1}.$$

In this case $N_r(s) = r_0$ and $D_r^+(s) = s^2+1$, $D_r^-(s) = 1$. Since the plant doesn't contain a copy of $D_r^+(s)$, the internal model principle suggests that we should pick a controller of the form

$$C(s) = \frac{1}{D_r^+(s)}C_1(s) = \frac{1}{s^2+1}C_1(s)$$

where $C_1(s) \in \mathbb{R}(s)$ is a TF yet to be determined. That is, we embed an internal model of the unstable part of $R(s)$ in $C(s)$ and allow an extra factor $C_1(s)$ to achieve feedback stability. You can check that $C_1(s) = s$ works. ▲

Remark 2.8.3. The internal model principle of Theorem 2.8.2 is only concerned with the unstable part of the Laplace transform $R(s)$ of the reference signal $r(t)$. For example, if $r(t) = \mathrm{e}^{-t} + \sin(t)$ then

$$R(s) = \frac{1}{s+1} + \frac{1}{s^2+1} = \frac{s^2+s+2}{(s+1)(s^2+1)}.$$

In this case $D_r^-(s) = s+1$ and $D_r^+(s) = s^2+1$. The internal model needs to include the poles at $s = \pm j$ (roots of $D_r^+(s)$) but not the pole at $s = -1$. This is because the part of $r(t)$ corresponding to stable pole decays to zero in steady-state and hence there is nothing to asymptotically track. ♦

To help understanding, it is useful to interpret the internal model principle in the frequency domain.

Example 2.8.3. Returning to Example 2.8.2 we have that the TF from r to e is

$$\frac{E(s)}{R(s)} = \frac{1}{1 + C(s)P(s)} = \frac{s^3 + s^2 + s + 1}{s^3 + s^2 + 2s + 1} =: S(s).$$

Figure 2.23 shows the Bode plot of the frequency response $S(j\omega)$. Notice that the system heavily attenuates

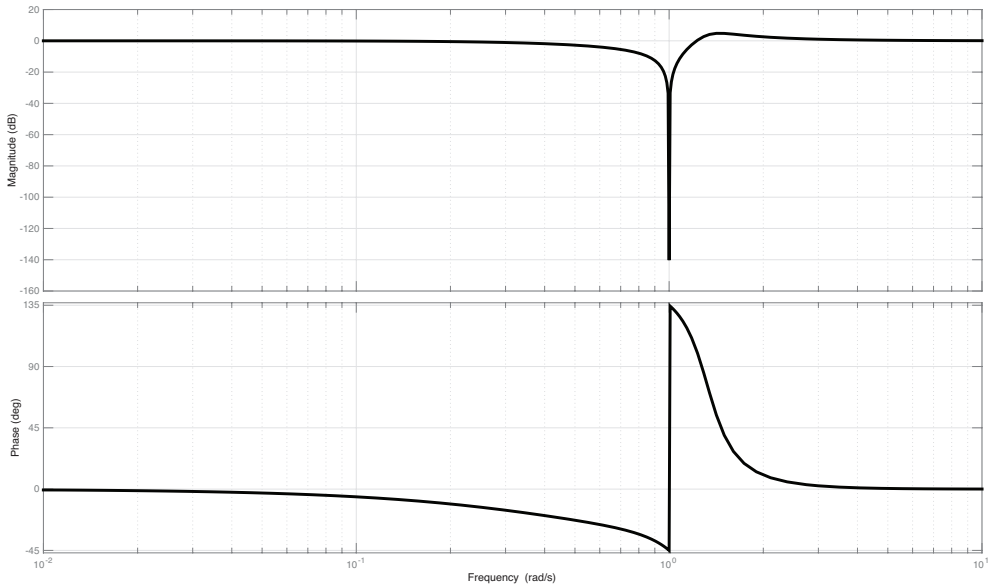


Figure 2.23: Bode plot of E/R from Example 2.8.2.

signals at the frequency $10^0 = 1$ rad/s. Indeed, the internal model introduces zeros at $s = \pm j$ in S which creates a notch filter. Any reference signals at this frequency result in zero steady-state error as we saw using the FVT in the previous example. ▲

Exercise 2.10. At what frequencies do you expect the Bode plot of E/R from Example 2.8.1 to heavily attenuate signals? Confirm your response by plotting the Bode plot.

In summary, we have the following facts:

- If the closed-loop system is unstable, then we can't even talk about steady-state tracking performance since there will likely not be a steady-state response.
- Even if the closed-loop system is stable, we have to be careful applying the FVT to compute the steady-state tracking error because, depending on the reference signal, $sE(s)$ may have poles in $\mathrm{Re}\,(s) \geq 0$.
- If the closed-loop system is stable and $P(s)C(s)$ contains an internal model of the unstable part of $R(s)$, then perfect asymptotic tracking occurs.
- It does not matter if $P(s)$ or $C(s)$ provide the internal model. Perfect asymptotic tracking occurs when the product $C(s)P(s)$ has an internal model of the unstable part of $R(s)$.
- If $C(s)P(s)$ does not contain an internal model, then increasing the gain $|C(j\omega)P(j\omega)|$ over the range of frequencies that $r(t)$ contains decreases the steady-state tracking error. This is consistent with the idea that high-gain leads to good performance.

2.A Design procedure for continuous-time lag controllers

The controller is

$$C(s) = KC_1(s) = K\frac{\alpha Ts+1}{Ts+1}, \qquad 0<\alpha<1,\ T>0,\ K>0, \qquad (\textbf{lag controller}). \tag{2.17}$$

Lag controllers are used for two distinct purposes:

1. To increase the phase margin. This is done indirectly by lowering the high frequency gain.
2. Boost low frequency gain to improve steady-state tracking and disturbance rejection without having too much effect on gain margin, phase margin nor high frequency behaviour.

Procedure for lag controller design

The given specifications are:

(a) Steady-state specification (determined by tracking or disturbance rejection requirement),

(b) Increase phase margin to be greater than or equal to $\Phi_{\text{pm}}^{\text{des}}$ (determined by damping ratio or robustness requirements).

1. Use FVT to fix the steady-state gain K of the controller.
2. Draw the Bode plot of $KP(j\omega)$. Check Φ_{pm}.
3. If Φ_{pm} specification is met, we're done (a proportional controller can do the job!). Otherwise, find $\omega^\star$ such that
$$\text{dist}\left(\measuredangle(KP(j\omega^\star)), 180^\circ\right) = \Phi_{\text{pm}}^{\text{des}} + \delta.$$
Here $\Phi_{\text{pm}}^{\text{des}}$ is the given specification and δ is a buffer to account for approximations, usually 5°.
4. Shift the gain down at the frequency $\omega^\star$ from step 3 to get a new gain crossover frequency
$$\alpha = \frac{1}{K|P(j\omega^\star)|}.$$
5. Ensure that the phase isn't affected near the frequency $\omega^\star$ from steps 3, 4
$$\frac{10}{\alpha T} \leq \omega^\star.$$
6. Simulate the closed-loop system and check the Bode/Nyquist plots of $C(j\omega)P(j\omega)$ to make sure all specifications are met.

2.B Design procedure for continuous-time lead controllers

The controller is

$$C(s) = KC_1(s) = K\frac{\alpha Ts+1}{Ts+1}, \qquad \alpha>1,\ T>0,\ K>0, \qquad (\textbf{lead controller}). \tag{2.18}$$

Lead controllers are used for two distinct purposes.

1. To increase the phase margin Φ_{pm} by adding phase at the appropriate frequency while also meeting a steady-state tracking requirement.

2. Increase phase margin while simultaneously increasing the closed-loop bandwidth to make the system response faster.

We need three formulas.

1. The frequency ω_m. This is the midpoint between $\frac{1}{\alpha T}$ and $\frac{1}{T}$ on the logarithmically scaled frequency axis. It equals
$$\omega_m = \frac{1}{T\sqrt{\alpha}}. \tag{2.19}$$
2. The magnitude of $C_1(j\omega)$ at ω_m.
$$|C_1(j\omega_m)| = \sqrt{\alpha}. \tag{2.20}$$
3. The angle $\phi_{\max}$. This is the angle of $C_1(j\omega_m)$
$$\phi_{\max} = \sin^{-1}\left(\frac{\alpha-1}{\alpha+1}\right), \qquad \alpha = \frac{1+\sin(\phi_{\max})}{1-\sin(\phi_{\max})}. \tag{2.21}$$

Procedure for lead controller design

The given specifications are:

(a) increase phase margin to be at least equal to $\Phi_{\text{pm}}^{\text{des}}$ (determined by damping ratio or robustness requirements).

(b) *One* of the following:

(i) steady-state specification (determined by tracking or disturbance rejection requirement), or

(ii) a desired closed-loop bandwidth (determined by time-domain specifications or noise rejection requirements).

1. Define $\hat{K} \coloneqq K\sqrt{\alpha}$.
   (i) Use FVT to select $\hat{K}$ such that $\hat{K}P(s)$ meets the steady-state specification. Purposefully boost the gain by 10dB to compensate for the distortion that comes from α.
   (ii) Pick $\hat{K}$ so that the gain crossover frequency of $\hat{K}P(j\omega)$ equals the desired bandwidth.
2. Draw the Bode plot of $\hat{K}P(j\omega)$.
3. Find ω_{gc} and Φ_{pm}; set $\omega_m = \omega_{\text{gc}}$.
4. Determine the amount of phase to add: $\phi_{\max} = \Phi_{\text{pm}}^{\text{des}} - \Phi_{\text{pm}}$.
5. Compute α using (2.21). Set $K = \hat{K}/\sqrt{\alpha}$.
6. Compute T using (2.19).
7. Simulate closed-loop system and check Bode plot of $KC_1(j\omega)P(j\omega)$ to make sure all specifications are met. If they aren't met, you can return to Step 1 and use a different $\hat{K}$.

2.C Design procedure for continuous-time lead-lag controllers

The controller is

$$C(s) = KC_1(s)C_2(s) = K\frac{\alpha_1 T_1 s+1}{T_1 s+1}\frac{\alpha_2 T_2 s+1}{T_2 s+1},\ \alpha_1>1,\ 0<\alpha_2<1,\ T_1,T_2>0,\ K>0,\ (\textbf{lead-lag controller}). \tag{2.22}$$

The lead-lag controller is the product of a lead controller $C_1(s)$ and a lag controller $C_2(s)$. It is closely related to a P.I.D. controller.

Procedure for lead-lag controller design

The given specifications are:

(a) Steady-state specification (determined by tracking or disturbance rejection requirement),

(b) Increase phase margin to be greater than or equal to $\Phi_{\text{pm}}^{\text{des}}$ (determined by damping ratio or robustness requirements).

1. Define $\hat{K} \coloneqq K\sqrt{\alpha_1}$. Use FVT to select $\hat{K}$ such that $\hat{K}P(s)$ meets the steady-state specification. Purposefully boost the gain by 10dB to compensate for the distortion that comes from α_1.
2. Draw the Bode plot of $\hat{K}P(j\omega)$.
3. If Φ_{pm} specification is met, we're done (a proportional controller can do the job!). Otherwise, divide $\Phi_{\text{pm}}^{\text{des}}$ into two roughly equal parts $\Phi_{\text{pm}}^{\text{des}} = \Phi_{\text{pm},1} + \Phi_{\text{pm},2}$.
4. Design the lag controller $C_2(s)$ to get $\Phi_{\text{pm}} = \Phi_{\text{pm},2}$.
5. Draw the Bode plot of $\hat{K}C_2(j\omega)P(j\omega)$.
6. Design the lead controller $C_1(s)$ to get $\Phi_{\text{pm}} = \Phi_{\text{pm}}^{\text{des}}$ for the partially compensated system. Set $K = \hat{K}/\sqrt{\alpha}_1$.
7. Simulate the closed-loop system and check Bode plot of $C(j\omega)P(j\omega)$ to make sure all specifications are met.

Remark 2.C.1. The reason that the lag controller is designed first in the above procedure is that a lead controller will tend to "flatten" the phase curve. This means that if we tried to then design a lag controller, the crossover frequency will be very small and we'll get a sluggish response. ⧫

Chapter 3

Pole placement using polynomials

In this chapter we introduce a design technique called pole placement. Given a plant $P(s)$ and set of design specifications, first convert the design specifications into a "good region" of the complex plane where the closed-loop poles must be located in order to meet the specifications, then design a control law $C(s)$ so that the poles of the closed-loop system are in the good region.

Contents

3.1 Converting design specifications into desired pole locations

Consider our usual continuous-time unity feedback control system in Figure 3.1. We make the following standing

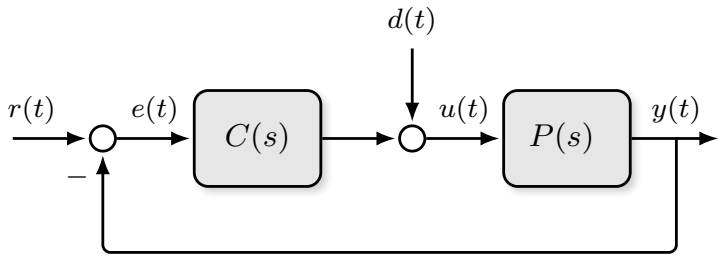


Figure 3.1: Continuous-time unity feedback system.

assumption throughout this chapter.

Assumption 3.1.1. The plant TF $P(s)$ is rational and proper. The numerator and denominator polynomials of the plant are coprime and the denominator polynomial is monic. ◀

As usual, write $P(s) = N_p(s)/D_p(s)$ and $C(s) = N_c(s)/D_c(s)$. Then the characteristic polynomial of the closed-loop system in Figure 3.1 is

$$\pi(s) = D_p(s)D_c(s) + N_p(s)N_c(s).$$

In this section our goal is to convert given time-domain and frequency-domain specifications for the closed-loop system into a "good region" $\mathbb{C}_g$ of the complex plane where the closed-loop poles (roots of the characteristic

polynomial) should be located in order to meet the specifications. Since we always want closed-loop stability, we will always require that $\mathbb{C}_g \subseteq \mathbb{C}^-$.

Our approach to selecting the "good region" is based on approximating the closed-loop transfer function from the reference to the output as an underdamped prototypical second order system

$$\frac{Y(s)}{R(s)} = \frac{C(s)P(s)}{1 + C(s)P(s)} = \frac{N_p(s)N_c(s)}{D_p(s)D_c(s) + N_p(s)N_c(s)} \approx \frac{K\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2}, \qquad 0 < \zeta < 1.$$

We will justify this approximation in our final design.

3.1.1 Settling time

Suppose our design specification is that the settling time due to a step reference signal must be less than a given value $T_{\text{s}}^{\max}$. In Section 2.5.2 we showed that in order to meet a settling time specification, the good region $\mathbb{C}_g$ must satisfy

$$\mathbb{C}_g \subseteq \left\{ s \in \mathbb{C} : \text{Re}\,(s) \leq -\frac{4}{T_{\text{s}}^{\max}} \right\} \qquad \text{(settling time)}.$$

3.1.2 Overshoot

Suppose our design specification is that the percentage overshoot due to a step reference signal must be less than $\%\text{OS}^{\max}$. In Section 2.5.2 we showed that in order to meet such an overshoot specification, the good region $\mathbb{C}_g$ must satisfy

$$\mathbb{C}_g \subseteq \{ s \in \mathbb{C} : |\arg(s)| \geq \pi - \theta^{\max} \} \qquad \text{(overshoot)}$$

where $\arg(s) \in (-\pi, \pi]$ returns the principle argument and

$$\theta^{\max} = \arccos\left(-\frac{\ln(\%\text{OS}^{\max})}{\sqrt{\pi^2 + (\ln(\%\text{OS}^{\max}))^2}} \right).$$

3.1.3 Phase margin

Consider the system in Figure 3.1 with

$$C(s)P(s) = \frac{\omega_n^2}{s(s + 2\zeta\omega_n)} \in \mathbb{R}(s). \tag{3.1}$$

Then the closed-loop transfer function from r to y is

$$\frac{Y(s)}{R(s)} = \frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2}$$

which is the prototype second order system with $K = 1$. To find the gain crossover frequencies, set $|C(j\omega)P(j\omega)| = 1$ and solve for ω to obtain

$$\omega_{\text{gc}} = \omega_n \sqrt{\sqrt{1 + 4\zeta^4} - 2\zeta^2}. \tag{3.2}$$

Using the definition of phase margin (and after some messy algebra) we get that the phase margin is

$$\Phi_{\text{pm}} = \arctan\left(\frac{2\zeta}{\sqrt{\sqrt{1 + 4\zeta^4} - 2\zeta^2}} \right). \tag{3.3}$$

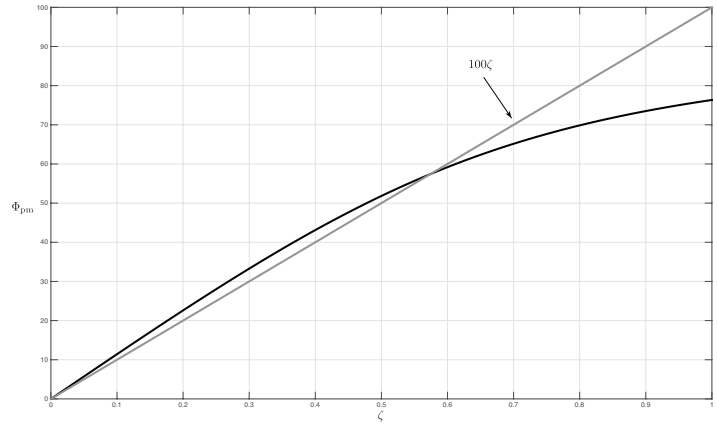


Figure 3.2: Phase margin Φ_{pm} as a function ζ for (3.1) (black) and its linear approximation (grey).

Observe that the phase margin only depends on the damping ratio ζ and not on ω_n. Figure 3.2 shows a graph of phase margin Φ_{pm} versus damping ratio ζ and also shows that it can be reasonably well approximated by a line for small values of ζ. We conclude that, for the system (3.1) we have

$$\text{If } 0 \leq \zeta \leq 0.7, \quad \text{then} \quad \Phi_{\text{pm}} \approx 100\zeta \quad \text{(expressed in degrees)}. \tag{3.4}$$

Alternatively, in terms of percentage overshoot

$$\Phi_{\text{pm}} \approx 60(1 - \%\text{OS}), \quad 0.4 \leq \%\text{OS} \leq 1, \quad \text{(expressed in degrees)}. \tag{3.5}$$

While the expression (3.4) was derived for the specific loop gain (3.1), we will use it as a general rule-of-thumb for converting overshoot specifications into phase margin specifications. In summary, if we are given a phase margin specification, we first convert it into a damping ratio/overshoot specification and then convert that into a good region of the complex plane.

3.2 Pole placement design

The idea of pole placement is to design the controller so that the poles of the closed-loop system, i.e., the roots of the closed-loop characteristic polynomial, are placed in the prescribed desirable location. Consider the unity feedback system in Figure 3.1 and let the plant be given by

$$P(s) = \frac{N_p(s)}{D_p(s)} = \frac{b_n s^n + b_{n-1}s^{n-1} + \cdots + b_1 s + b_0}{s^n + a_{n-1}s^{n-1} + \cdots + a_1 s + a_0}$$

where $N_p(s)$ and $D_p(s)$ are coprime. The controller is real rational and proper

$$C(s) = \frac{N_c(s)}{D_c(s)} = \frac{g_m s^m + g_{m-1}s^{m-1} + \cdots + g_1 s + g_0}{f_m s^m + f_{m-1}s^{m-1} + \cdots + f_1 s + f_0}.$$

We have $2m + 2$ gains f_i, g_i to "tweak" in our controller $C(s)$. The closed-loop characteristic polynomial

$$\pi(s) = D_p(s)D_c(s) + N_p(s)N_c(s) +$$

will have (at most) degree $n + m$.

Pole Placement Problem. *Given a plant $P(s) \in \mathbb{R}(s)$ and a symmetric set [1] of $n+m$ desired closed-loop pole locations $\{\lambda_1, \ldots, \lambda_{n+m}\} \subset \mathbb{C}_g$, find, if possible, a control law $C(s) \in \mathbb{R}(s)$ such that the poles of the closed-loop system equal $\{\lambda_1, \ldots, \lambda_{n+m}\}$.*

[1] A set $\Lambda \subset \mathbb{C}$ is symmetric if $\lambda \in \Lambda$ implies that $\overline{\lambda} \in \Lambda$ where $\overline{\lambda}$ is the complex conjugate of λ. This is also called conjugate symmetry.

It turns out that, under the assumption that the pair (N_p, D_p) is coprime, we can always solve this problem as long as we choose a controller of high enough order.

Theorem 3.2.1. *Suppose that Assumption 3.1.1 holds. There exists an mth order controller that solves the Pole Placement Problem for any symmetric set $\{\lambda_1, \ldots, \lambda_{n+m}\}$ if, and only if, $m \geq n-1$.*

The pole placement controller is unique if, and only if, $m = n-1$. If $m > n-1$ then pole placement controllers become non-unique for a given set of desired closed-loop poles. If the order of the controller is less than $n-1$, then there are no guarantees and in most cases the problem is not solvable.

Let's derive the design equations. Take $m = n-1$ as the order of our controller. Then the closed-loop characteristic polynomial will have degree $n+n-1 = 2n-1$. Choose $2n-1$ desired pole locations $\{\lambda_1, \ldots, \lambda_{2n-1}\} \subset \mathbb{C}_g$ where the good region $\mathbb{C}_g$ is obtained using the results of the previous section. The desired pole locations generate a desired closed-loop characteristic polynomial

$$\begin{aligned}\pi_{\text{des}}(s) &= \prod_{i=1}^{2n-1} (s-\lambda_i) \\ &=: s^{2n-1} + \alpha_{2n-2}s^{2n-2} + \cdots + \alpha_1 s + \alpha_0.\end{aligned}$$

So the problem is, given N_p, D_p, $\pi_{\text{des}} \in \mathbb{R}[s]$, find polynomials N_c, D_c such that

$$D_p(s)D_c(s) + N_p(s)N_c(s) = \pi_{\text{des}}(s).$$

This is called a polynomial Diophantine equation. Comparing coefficients between π and π_{des} yields the equation

$$\begin{bmatrix} 1 & 0 & \cdots & 0 & b_n & \cdots & \cdots & 0 \\ a_{n-1} & 1 & & \vdots & b_{n-1} & b_n & & \vdots \\ \vdots & \ddots & \ddots & \vdots & \vdots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & 1 & \vdots & & \ddots & b_n \\ a_0 & \cdots & \cdots & a_{n-1} & b_0 & \cdots & \cdots & b_{n-1} \\ 0 & \ddots & & \vdots & 0 & \ddots & & \vdots \\ \vdots & & \ddots & \vdots & \vdots & & \ddots & \vdots \\ 0 & \cdots & \cdots & a_0 & 0 & \cdots & \cdots & b_0 \end{bmatrix} \begin{bmatrix} f_{n-1} \\ \vdots \\ \vdots \\ f_0 \\ g_{n-1} \\ \vdots \\ \vdots \\ g_0 \end{bmatrix} = \begin{bmatrix} 1 \\ \alpha_{2n-2} \\ \vdots \\ \alpha_n \\ \alpha_{n-1} \\ \vdots \\ \alpha_0 \end{bmatrix}. \tag{3.6}$$

Remark 3.2.2.

- The $2n \times 2n$ matrix on the left-side of equation (3.6) has a nice regular structure. It is the Sylvester matrix $\mathbf{S}(D_p(s), N_p(s))$ discussed in Appendix 3.A.
- By Theorem 3.A.1, coprimeness of N_p and D_p implies that the Sylvester matrix in (3.6) is invertible, i.e., that a solution exists and is unique.
- We obtained the "good region" using the design equations of the prototype second order system. The actual closed-loop system is not, in general, second order. Therefore, when we pick our desired pole locations in $\mathbb{C}_g$, we should ensure that there are two dominant complex conjugate poles so that our approximation is valid. See Section 2.7.
- One significant limitation with pole placement is that we have no idea where the closed-loop zeros will be and consequently we have little idea what is happening in the frequency domain: the final controller may have poor sensitivity, undesirable bandwidth, etc. This can limit its use in practice. ⧫

Example 3.2.1. (Second Order Plant) Consider the unity-feedback system in Figure 3.1 with plant

$$P(s) = \frac{1}{s(s+1)} = \frac{1}{s^2+s+0} =: \frac{b_2 s^2 + b_1 s + b_0}{s^2 + a_1 s + a_0}.$$

The plant is second order, $n = 2$. The specifications are closed-loop stability and $T_s \leq 2$. To define the "good region" we assume that the closed-loop system is second order system and use the results of Section 3.1.1. This yields the good region of the complex plane $\mathbb{C}_g = \{s \in \mathbb{C} : \mathrm{Re}\,(s) \leq -2\}$. To guarantee that the poles of the closed-loop system can be placed in $\mathbb{C}_g$, we choose a controller of order $n-1 = 2-1 = 1$, i.e.,

$$C(s) = \frac{g_1 s + g_0}{f_1 s + f_0}.$$

Next we pick $2n-1 = 3$ desired pole locations in the good region. We'll place two dominant complex conjugate poles at $\lambda_1 = -3-j$, $\lambda_2 = -3+j$ and put the third pole further to the left so that our second order approximation is reasonable. Choose, say, $\lambda_3 = -10$. Then

$$\begin{aligned}\pi_{\mathrm{des}}(s) &= (s+3+j)(s+3-j)(s+10) \\ &= s^3 + 16s^2 + 70s + 100 \\ &=: s^3 + \alpha_2 s^2 + \alpha_1 s + \alpha_0.\end{aligned}$$

The resulting design equation (3.6) is

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} f_1 \\ f_0 \\ g_1 \\ g_0 \end{bmatrix} = \begin{bmatrix} 1 \\ 16 \\ 70 \\ 100 \end{bmatrix}.$$

This equation is small enough to solve by hand and obtain

$$f_1 = 1, \qquad f_0 = 15, \qquad g_1 = 55, \qquad g_0 = 100$$

and the control law is given by $C(s) = (55s+100)/(s+15)$. The closed-loop TF from r to y is

$$\frac{Y(s)}{R(s)} = \frac{C(s)P(s)}{1+C(s)P(s)} = \frac{55s+100}{s^3+16s^2+70s+100} = \frac{55s+100}{(s+3+j)(s+3-j)(s+10)}.$$

The closed-loop step response is shown in Figure 3.3. The settling time is 1.73 seconds and meets the specifications. ▲

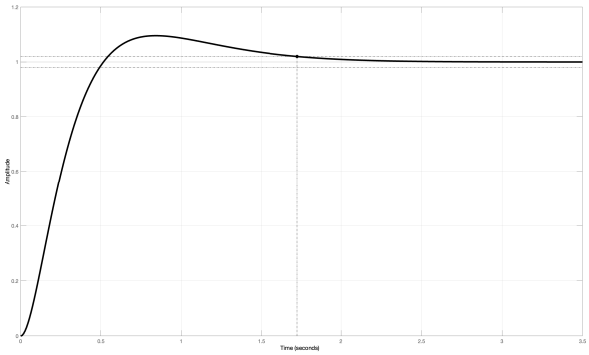


Figure 3.3: Step response for Example 3.2.1.

The MATLAB code below was used to design and simulate the controller from the above example.

```
Np = [0 0 1];
Dp = [1 1 0];
P = tf(Np, Dp); % plant
n = length(Dp)-1; %order of plant

S = zeros(2*n); % Sylvester matrix to be constructed

for i=1:n
    S(i:i+n, i) = Dp';
    S(i:i+n, i+n) = Np';
end

Lambda = [-3-1i -3+1i -10]; % desired pole locations

if (length(Lambda) ~= 2*n-1)
    disp('Incorrect number of desired pole locations selected');
    C = [];
else
    pides = poly(Lambda); % desired ch.p.

    % compute controller gains
    gains = S\pides;
    Dc = gains(1:n)';
    Nc = gains(n+1:2*n)';
    C = tf(Nc, Dc);% controller
end

% simulate closed-loop response
G = feedback(C*P, 1); % unity feedback system
step(G); %step response
```

In order to achieve *arbitrary* pole placement, the controller order has to be at least $n-1$ where n is the plant order. However we may be able to place the closed-loop poles to particular positions using controllers with order less than $n-1$.

Example 3.2.2. Let

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

and suppose that our desired pole locations are $\{-1,-1\}$. This yields $\pi_{\text{des}}(s) = (s+1)^2 = s^2+2s+1$. The actual closed-loop characteristic equation is $(s^2+2s)D_c(s) + N_c(s)$. So we immediately see that the controller $C(s) = 1$ places the poles at the desired locations.

It is of interest to see what happens if we insist on using a first order controller to place the poles at $\{-1,-1,-2\}$. The pole placement design equations become

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 2 & 1 & 0 & 0 \\ 0 & 2 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} f_1 \\ f_0 \\ g_1 \\ g_0 \end{bmatrix} = \begin{bmatrix} 1 \\ 4 \\ 5 \\ 2 \end{bmatrix}.$$

This yields the unique solution $f_1 = 1$, $f_0 = 2$, $g_1 = 1$, $g_0 = 2$ and the controller

$$C(s) = \frac{s+2}{s+2} = 1.$$

▲

Next we consider pole placement using strictly proper controllers. Strictly proper controllers are advantageous in some applications. For example, when the reference $r(t)$ is discontinuous (e.g., a sequence of steps) and we don't want that to produce discontinuous control signals that could potentially damage the actuators. In this case our controller has the form

$$C(s) = \frac{N_c(s)}{D_c(s)} = \frac{g_{m-1}s^{m-1} + g_{m-2}s^{m-2} + \cdots + g_1 s + g_0}{f_m s^m + f_{m-1}s^{m-1} + \cdots f_1 s + f_0}$$

and we have the following result.

Theorem 3.2.3. *Suppose that Assumption 3.1.1 holds. There exists an mth order strictly proper controller that solves the PPP for any symmetric set $\{\lambda_1, \ldots, \lambda_{n+m}\}$ if, and only if, $m \geq n$.*

This result says that if we want a strictly proper controller, we need to increase its order by 1 compared to the proper case. Design equations for strictly proper controllers are obtained as follows. Take a set of desired pole locations $\{\lambda_1, \ldots, \lambda_{2n}\}$. We have $2n$ desired locations now because our controller has to be at least order n. Generate a desired closed-loop characteristic polynomial

$$\pi_{\text{des}}(s) = \prod_{i=1}^{2n}(s - \lambda_i) =: s^{2n} + \alpha_{2n-1}s^{2n-1} + \alpha_{2n-2}s^{2n-2} + \cdots + \alpha_1 s + \alpha_0.$$

Compare coefficients between π_{des} and $\pi(s) = D_p(s)D_c(s) + N_p(s)N_c(s)$ to get the design equation

$$f_n = 1 \tag{3.7a}$$

$$\begin{bmatrix} 1 & 0 & \cdots & 0 & b_n & \cdots & \cdots & 0 \\ a_{n-1} & 1 & & \vdots & b_{n-1} & b_n & & \vdots \\ \vdots & \ddots & \ddots & \vdots & \vdots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & 1 & \vdots & & \ddots & b_n \\ a_0 & \cdots & \cdots & a_{n-1} & b_0 & \cdots & \cdots & b_{n-1} \\ 0 & \ddots & & \vdots & 0 & \ddots & & \vdots \\ \vdots & & \ddots & \vdots & \vdots & & \ddots & \vdots \\ 0 & \cdots & \cdots & a_0 & 0 & \cdots & \cdots & b_0 \end{bmatrix} \begin{bmatrix} f_{n-1} \\ \vdots \\ \vdots \\ f_0 \\ g_{n-1} \\ \vdots \\ \vdots \\ g_0 \end{bmatrix} = \begin{bmatrix} \alpha_{2n-1} - a_{n-1} \\ \alpha_{2n-2} - a_{n-2} \\ \vdots \\ \alpha_n - a_0 \\ \alpha_{n-1} \\ \vdots \\ \alpha_0 \end{bmatrix}. \tag{3.7b}$$

Example 3.2.3. We repeat Example 3.2.1 but this time we'll design a strictly proper controller for the plant

$$P(s) = \frac{1}{s(s+1)} = \frac{1}{s^2 + s + 0} =: \frac{b_2 s^2 + b_1 s + b_0}{s^2 + a_1 s + a_0}.$$

The plant has order $n = 2$ so the our controller has the form

$$C(s) = \frac{g_1 s + g_0}{f_2 s^2 + f_1 s + f_0}.$$

Select our desired pole locations to be $\{-3 - j, -3 + j, -10, -11\}$ which yields the desired closed-loop ch.p.

$$\pi_{\text{des}}(s) = (s + 3 + j)(s + 3 - j)(s + 10)(s + 11) = s^4 + 27s^3 + 246s^2 + 870s + 1100.$$

Equation (3.7) yields $f_2 = 1$ and

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} f_1 \\ f_0 \\ g_1 \\ g_0 \end{bmatrix} = \begin{bmatrix} 27 - 1 \\ 246 - 0 \\ 870 \\ 1100 \end{bmatrix}.$$

Solving gives $f_1 = 26$, $f_0 = 220$, $g_1 = 650$, $g_0 = 1100$ and the controller

$$C(s) = \frac{650s + 1100}{s^2 + 26s + 220}$$

The closed-loop TF from r to y is

$$\frac{Y(s)}{R(s)} = \frac{C(s)P(s)}{1 + C(s)P(s)} = \frac{630s + 990}{(s^2 + 6s + 10)(s + 10)(s + 11)}.$$

The settling time of the closed-loop step response is 1.88 seconds and meets the specifications. ▲

The MATLAB code below was used to design and simulate the controller from the above example.

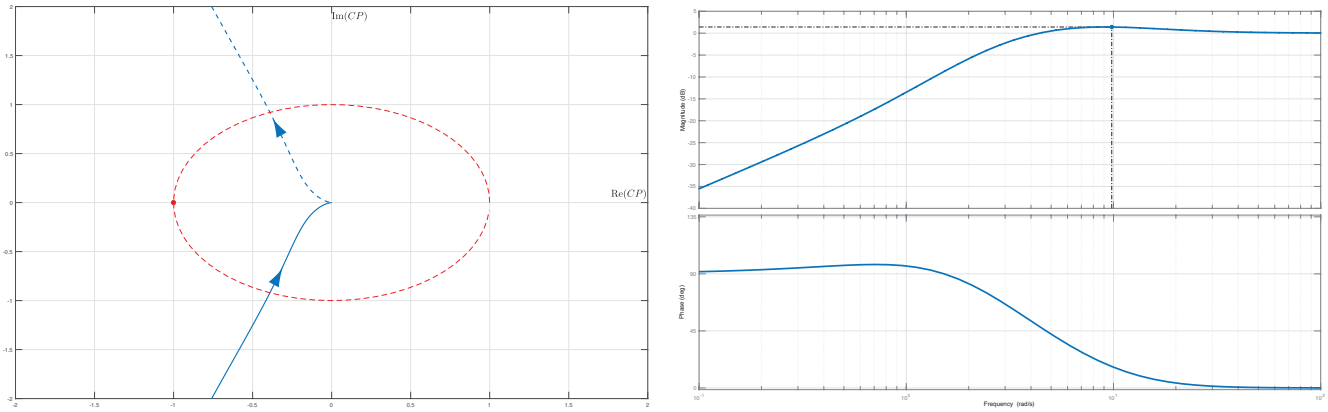
```
Np = [0 0 1];
Dp = [1 1 0];
P = tf(Np, Dp); % plant
n = length(Dp)-1; %order of plant

S = zeros(2*n); % Sylvester matrix to be constructed

for i=1:n
    S(i:i+n, i) = Dp';
    S(i:i+n, i+n) = Np';
end

% desired characteristic polynomial
Lambda = [-3-1i -3+1i -10 -11]; % desired pole locations

if (length(Lambda) ~= 2*n)
    disp('Incorrect number of desired pole locations selected');
    C = [];
else
    pides = poly(Lambda); % desired ch.p.

    % compute controller gains
    gains = S\(pides(2:end) - [Dp(2:n+1)'; zeros(n,1)]);
    Dc = [1 gains(1:n)'];
    Nc = gains(n+1:2*n)';
    C = tf(Nc, Dc);% controller
end

% simulate closed-loop response
G = feedback(C*P, 1); % unity feedback system
step(G); %step response
```

The next example emphasizes the fact that, when doing pole placement design, we have no say in where the closed-loop zeros end up. So we should use all the tools at our disposal and evaluate the frequency response of the closed-loop system.

Example 3.2.4. In Example 3.2.1 the plant and controller were

$$P(s) = \frac{1}{s(s + 1)}, \qquad C(s) = \frac{55s + 100}{s + 15}.$$

Figure 3.4 shows the Nyquist plot of $C(s)P(s)$ and Bode plot of the sensitivity function $S(s) = 1/(1+C(s)P(s))$. From the Nyquist plot in Figure 3.4a we deduce that the phase margin is around 67° degrees and the gain margin is infinite. From the Bode plot in Figure 3.4b we get a peak magnitude of 1.41 dB. The stability margin equals



(a) Nyquist plot. *(b) Bode plot.*

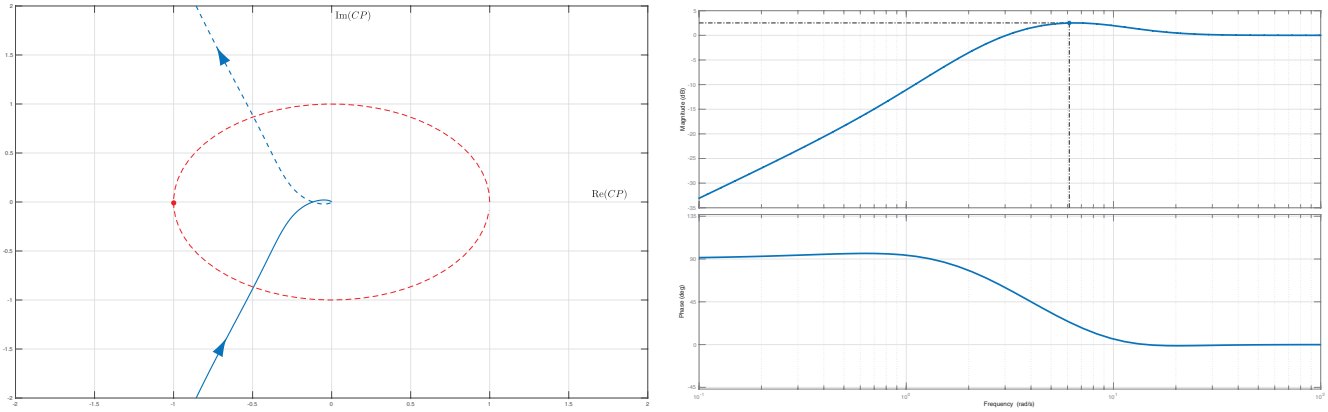
Figure 3.4: Nyquist plot of $C(s)P(s)$ and Bode plot of $1/(1+CP)$ for the plant and controller from Example 3.2.1.

the reciprocal of the peak response (not in dB) and therefore we have a stability margin[2] of 0.93. This controller is quite robust to plant model uncertainty.

In Example 3.2.3 we designed a strictly proper pole placement controller

$$P(s) = \frac{1}{s(s+1)}, \qquad C(s) = \frac{650s + 1100}{s^2 + 26s + 220}.$$

Figure 3.5 shows the Nyquist plot of $C(s)P(s)$ and Bode plot of the sensitivity function $S(s) = 1/(1+C(s)P(s))$ for this plant-controller pair. The phase margin is similar to the previous design but the gain margin is now



(a) Nyquist plot. *(b) Bode plot.*

Figure 3.5: Nyquist plot of $C(s)P(s)$ and Bode plot of $1/(1+CP)$ for the plant and controller from Example 3.2.3.

18.5 dB and no longer infinite. The stability margin is now 0.88 which is slightly worse (but still pretty good). This controller is not as robust to plant model uncertainty as our first design. In deciding which controller to use, we would have to decide which feature is more important. On the one hand, if we value robustness to model uncertainty, then we should pick the controller from Example 3.2.1. On the other hand, if we must avoid discontinuous control signals arising from a discontinuous reference signal, then the strictly proper controller from Example 3.2.3 is preferable. ▲

[2]Recall that the stability margin of a system equals the distance from the critical point -1 to the closest point on the Nyquist plot. For good robustness you typically want a stability margin of 0.5 or greater.

3.3 Pole placement and tracking error

In this section we solve the Pole Placement Problem (P.P.P.) while at the same time satisfying design specifications on steady-state tracking error, i.e., given some number $e_{\text{ss}}^{\max} \geq 0$, we want to satisfy a specification of the form

$$|e_{\text{ss}}| = |r(t) - y_{\text{ss}}| \leq e_{\text{ss}}^{\max}.$$

We can incorporate this specification in pole placement design by increasing the order of the controller.

3.3.1 Asymptotic step tracking

Asymptotic tracking or "perfect tracking" refers to the situation in which we want zero steady-state tracking error, i.e., $e_{\text{ss}}^{\max} = 0$. Suppose we have a stable unity feedback continuous-time control system. Then the steady-state tracking error for a step input is

$$\begin{aligned} e_{\text{ss}} &= \lim_{t\to\infty} e(t) = \lim_{s\to 0} sE(s) \qquad \text{(FVT applies since closed-loop system is stable)} \\ &= \lim_{s\to 0} s\frac{1}{1+PC}\frac{1}{s} \\ &= \lim_{s\to 0} \frac{1}{1+PC}. \end{aligned}$$

In Section 2.8 we showed that for asymptotic step tracking the product PC must have a pole at[3] $s = 0$. Using this fact, we can modify the pole placement design approach to achieve asymptotic step tracking as follows:

Case 1 If $P(s)$ has a pole at $s = 0$, then we can use regular pole placement from Section 3.2.

Case 2 If $P(s)$ has a zero at $s = 0$, then asymptotic tracking is not possible.

Case 3 If $P(s)$ has neither a pole at $s = 0$ nor a zero at $s = 0$, choose $C(s) = C_1(s)/s$ and design $C_1(s)$ using pole placement for the augmented plant $P(s)/s$.

The main drawback of this approach is that the complexity of $C(s)$ increases. If $P(s)$ has order n, then the augmented plant $P(s)/s$ has order $n+1$ which means that $C_1(s)$ must be of order n to guarantee that PPP is solvable. The final controller $C_1(s)/s$ has order $n+1$.

Example 3.3.1. Let's design a controller for the unstable and nonminimum phase plant

$$P(s) = \frac{1-s}{s^2+1}.$$

Our specifications are (i) a maximum settling time of 2 seconds, (ii) maximum overshoot of $\mathrm{e}^{-\pi}$ and (iii) asymptotic step tracking. Since the plant doesn't already have a pole at the origin, our controller must provide one in order to meet spec (iii). So we'll design for the augmented plant

$$\frac{P(s)}{s} = \frac{1-s}{s^3+s}.$$

The transient specifications yield the good region

$$\mathbb{C}_g = \{s \in \mathbb{C} : \operatorname{Re}(s) \leq -2\} \cap \{s \in \mathbb{C} : |\arg(s)| \geq 3\pi/4\}.$$

[3] For asymptotic *ramp tracking* the product PC must have two poles at $s = 0$. More generally, one should apply the internal model principle (Theorem 2.8.2).

The augmented plant has order $n = 3$ so to place the closed-loop poles in the good region we need a controller of order 2

$$C_1(s) = \frac{g_2 s^2 + g_1 s + g_0}{f_2 s^2 + f_1 s + f_0}.$$

Let's place the closed-loop poles at $\{-2 \pm j, -8, -9, -10\}$ so that $\pi_{\text{des}}(s) = s^5 + 31s^4 + 355s^3 + 1823s^2 + 4090s + 3600$. The pole placement design equation (3.6) becomes

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & -1 & 0 & 0 \\ 0 & 1 & 0 & 1 & -1 & 0 \\ 0 & 0 & 1 & 0 & 1 & -1 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} f_2 \\ f_1 \\ f_0 \\ g_2 \\ g_1 \\ g_0 \end{bmatrix} = \begin{bmatrix} 1 \\ 31 \\ 355 \\ 1823 \\ 4090 \\ 3600 \end{bmatrix}$$

Solving the equation gives

$$C_1(s) = \frac{4564s^2 + 2772s + 3600}{s^2 + 31s + 4918}$$

and the final controller is

$$C(s) = \frac{C_1(s)}{s} = \frac{4564s^2 + 2772s + 3600}{s(s^2 + 31s + 4918)}.$$

▲

Exercise 3.1. Repeat the above example but this time design $C_1(s)$ to be strictly proper.

3.3.2 Practical step tracking

Suppose that our specifications are less stringent and $e_{\text{ss}}^{\max} \neq 0$. Then, assuming the feedback system is stable and $r(t) = \mathbf{1}(t)$,

$$e_{\text{ss}} = \lim_{s \to 0} sE(s) = \lim_{s \to 0} s \frac{1}{1 + PC} \frac{1}{s} = \lim_{s \to 0} \frac{1}{1 + PC}.$$

If $P(0) \neq \infty$, i.e., no pole at $s = 0$ and $P(0) \neq 0$, i.e., no zero at $s = 0$, then

$$e_{\text{ss}} = \frac{1}{1 + P(0)C(0)}$$

and e_{ss} depends on $C(0)$. We know that, if $P(s)$ has order n, an $(n-1)$th order controller $C(s)$ can arbitrarily assign the closed-loop poles. We will add extra parameters to $C(s)$ so that we can adjust $C(0)$ and hence adjust e_{ss}. The most straightforward change is to increase the order of the controller by one

$$C(s) = \frac{g_n s^n + \cdots + g_1 s + g_0}{f_n s^n + \cdots + f_1 s + f_0}.$$

Let's see how this works in the second order case. Given the plant

$$P(s) = \frac{b_2 s^2 + b_1 s + b_0}{s^2 + a_1 s + a_0}$$

choose a second order controller

$$C(s) = \frac{g_2 s^2 + g_1 s + g_0}{f_2 s^2 + f_1 s + f_0}.$$

The closed-loop characteristic polynomial is

$$\begin{aligned}\pi(s) &= D_pD_c + N_pN_c \\ &= (f_2 + b_2g_2)s^4 + (a_1f_2 + b_1g_2 + f_1)\,s^3 + (f_0 + a_0f_2 + a_1f_1 + b_0g_2 + b_1g_1)\,s^2 \\ &\quad + (a_0f_1 + a_1f_0 + b_0g_1 + b_1g_0)\,s + a_0f_0 + b_0g_0.\end{aligned}$$

Select four poles $\{\lambda_1, \lambda_2, \lambda_3, \lambda_4\}$ in the good region to define the desired closed-loop characteristic polynomial

$$\pi_{\text{des}}(s) = (s-\lambda_1)(s-\lambda_2)(s-\lambda_3)(s-\lambda_4) =: s^4 + \alpha_3 s^3 + \alpha_2 s^2 + \alpha_1 s + \alpha_0.$$

Comparing $\pi(s)$ and $\pi_{\text{des}}(s)$, we have 5 equations and 6 unknowns. We add a 6th equation by considering the tracking error

$$|e_{\text{ss}}| = \left|\frac{1}{1+P(0)C(0)}\right| \le e_{\text{ss}}^{\max}.$$

This constraint is satisfied if

$$\frac{1}{1+\frac{b_0}{a_0}\frac{g_0}{f_0}} = e_{\text{ss}}^{\max} \qquad \Longleftrightarrow \qquad \frac{b_0}{a_0}\frac{g_0}{f_0} = \frac{1}{e_{\text{ss}}^{\max}} - 1.$$

Let δ be any real number that satisfies

$$\delta \ge \frac{1}{e_{\text{ss}}^{\max}} - 1.$$

So the tracking error specification is satisfied if

$$b_0g_0 - \delta a_0 f_0 = 0. \tag{3.8}$$

This is a linear equation of the unknown controller parameters g_0, f_0. We add equation (3.8) to our pole placement design equations to obtain

$$\left[\begin{array}{cccccc} 1 & 0 & 0 & b_2 & 0 & 0 \\ a_1 & 1 & 0 & b_1 & b_2 & 0 \\ a_0 & a_1 & 1 & b_0 & b_1 & b_2 \\ 0 & a_0 & a_1 & 0 & b_0 & b_1 \\ 0 & 0 & a_0 & 0 & 0 & b_0 \\ \hdashline 0 & 0 & -\delta a_0 & 0 & 0 & b_0 \end{array}\right] \left[\begin{array}{c} f_2 \\ f_1 \\ f_0 \\ g_2 \\ g_1 \\ g_0 \end{array}\right] = \left[\begin{array}{c} 1 \\ \alpha_3 \\ \alpha_2 \\ \alpha_1 \\ \alpha_0 \\ \hdashline 0 \end{array}\right]. \tag{3.9}$$

The partition lines indicate the extra equation arising from the steady-state specification. If N_p and D_p are coprime and $P(s)$ has no poles or zeros at $s = 0$, then the above design equation has a unique solution. This approach generalizes to nth order plants.

Example 3.3.2. Consider, as usual, the unity feedback continuous-time control system with

$$P(s) = \frac{1}{(s+0.5)^2} = \frac{1}{s^2 + s + 0.25}$$

and given specifications:

(i) closed-loop stability,

(ii) maximum step tracking error of 5%,

(iii) step response satisfies

- overshoot less than or equal to 20%,

– settling time less than or equal to 4 seconds.

First convert specifications (i) and (iii) into a "good region." Stability requires that $\mathbb{C}_g \subseteq \mathbb{C}^-$. Additionally

$$T_\mathrm{s} \leq 4 \Rightarrow \zeta\omega_n \geq 1$$

which means that

$$\mathbb{C}_g \subseteq \{s \in \mathbb{C} : \mathrm{Re}\,(s) \leq -1\}\,.$$

Furthermore

$$\begin{aligned}
&\%\mathrm{OS} \leq 0.2 \\
&\Rightarrow \zeta \geq -\frac{\ln \%\mathrm{OS}}{\sqrt{\pi^2 + (\ln \%\mathrm{OS})^2}} = 0.4459 \\
&\Rightarrow \theta \leq \arccos(0.4459) \\
&\theta \leq 62.9^\circ \ (\approx 1.1 \text{ rad}).
\end{aligned}$$

Intersecting these regions in the complex plane we obtain

$$\mathbb{C}_g = \{s \in \mathbb{C} : \mathrm{Re}\,(s) \leq -1\} \cap \{s \in \mathbb{C} : |\arg(s)| \geq 2.04\}\,.$$

In order to guarantee that we can place all the closed-loop poles in $\mathbb{C}_g$ and satisfy the tracking error specification, we select a control law with the same order as the plant

$$C(s) = \frac{g_2 s^2 + g_1 s + g_0}{f_2 s^2 + f_1 s + f_0}.$$

The closed-loop system will be 4th order so we must choose 4 desired pole locations in $\mathbb{C}_g$. We'll place two poles at the corners of the region and the other two further to the left $\{\lambda_1, \lambda_2, \lambda_3, \lambda_4\} = \{-1+j1.95, -1-j1.95, -5-5\}$. This results in the desired closed-loop ch.p.

$$\pi_\mathrm{des}(s) = (s-\lambda_1)(s-\lambda_2)(s-\lambda_3)(s-\lambda_4) = s^4 + 12s^3 + 49.8s^2 + 98.03s + 120.1.$$

Next we incorporate the tracking error specification

$$\delta = \frac{1}{e_\mathrm{ss}^\mathrm{max}} - 1 = 19.$$

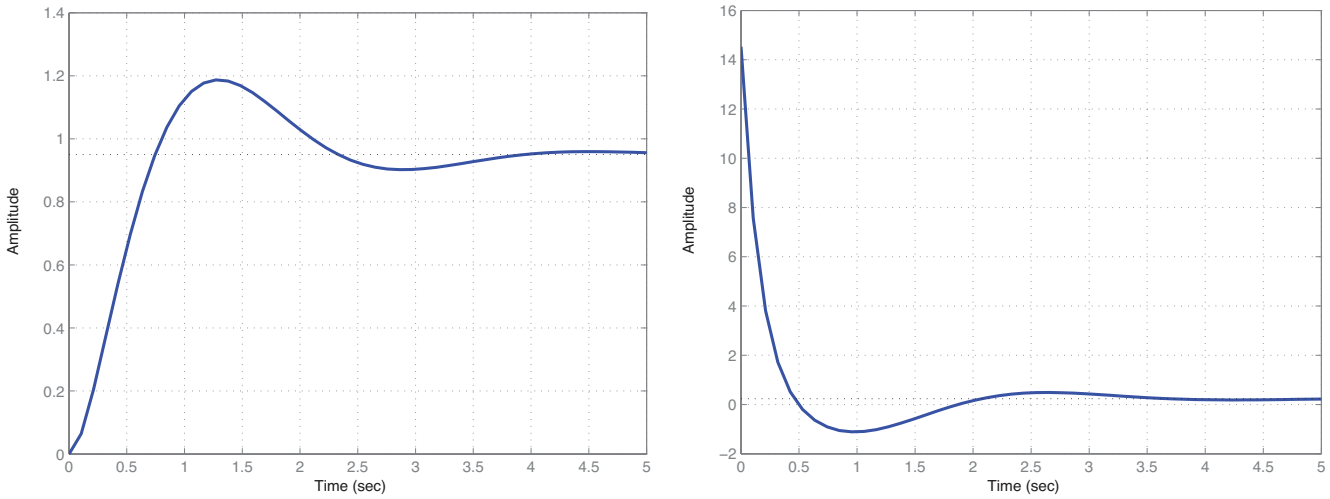
The resulting design equations are

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 \\ \frac{1}{4} & 1 & 1 & 1 & 0 & 0 \\ 0 & \frac{1}{4} & 1 & 0 & 1 & 0 \\ 0 & 0 & \frac{1}{4} & 0 & 0 & 1 \\ 0 & 0 & -\frac{19}{4} & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} f_2 \\ f_1 \\ f_0 \\ g_2 \\ g_1 \\ g_0 \end{bmatrix} = \begin{bmatrix} 1 \\ 12 \\ 49.8 \\ 98.03 \\ 120.1 \\ 0 \end{bmatrix}.$$

Solving this equation for the controller gains yields

$$C(s) = \frac{14.53s^2 + 71.26s + 114.1}{s^2 + 11s + 24.02}. \tag{3.10}$$

Figure 3.6a shows the closed-loop unit step response and Figure 3.6b shows the control signal. The overshoot on the step response is not acceptable, it is about 25%. The settling time is $T_\mathrm{s} = 3.8$ and $e_\mathrm{ss} = 0.05$. The reason the overshoot specification is not satisfied is because the non-dominant poles at $s = -5$ affected the response. We could iterate the design by changing the desired pole locations but instead we'll try a different approach in the next example. ▲



(a) Step response. *(b) Control signal.*

Figure 3.6: Step response and control signal for Example 3.3.2 with control law (3.10).

Example 3.3.3. (Cancelling Stable Plant Poles) To remedy the deficiencies with our design from the previous example we'll use stable pole-zero cancellation of the plant poles at $s = -0.5$. This cancellation will result in two of the roots of characteristic polynomial being at $s = -0.5$. However, their effect will not be observable at the output because they will not appear in the TF $Y(s)/R(s)$. Choose as our control law

$$C(s) = \frac{g_2 (s + 0.5)^2}{f_2 s^2 + f_1 s + f_0}$$

so that the closed-loop characteristic polynomial becomes

$$\pi(s) = g_2 (s + 0.5)^2 + (s + 0.5)^2 (f_2 s^2 + f_1 s + f_0) = (s + 0.5)^2 (f_2 s^2 + f_1 s + f_0 + g_2)$$

As expected, two of the closed-loop poles are fixed at $s = -0.5$. The effect of these poles will not be visible in the transfer function Y/R, but their effect will be visible in the other transfer functions E/R, U/R. We place the remaining two poles in the good region $\{\lambda_1, \lambda_2\} = \{-1 + j1.95, -1 - j1.95\}$. The design equation in this case reduces to

$$f_2 s^2 + f_1 s + f_0 + g_2 = (s + 1 - j1.95)(s + 1 + j1.95) = s^2 + 2s + 4.803.$$

and for the steady-state error, using formula (3.8), the equation is

$$\frac{g_2}{4} - \frac{19}{4} f_0 = 0.$$

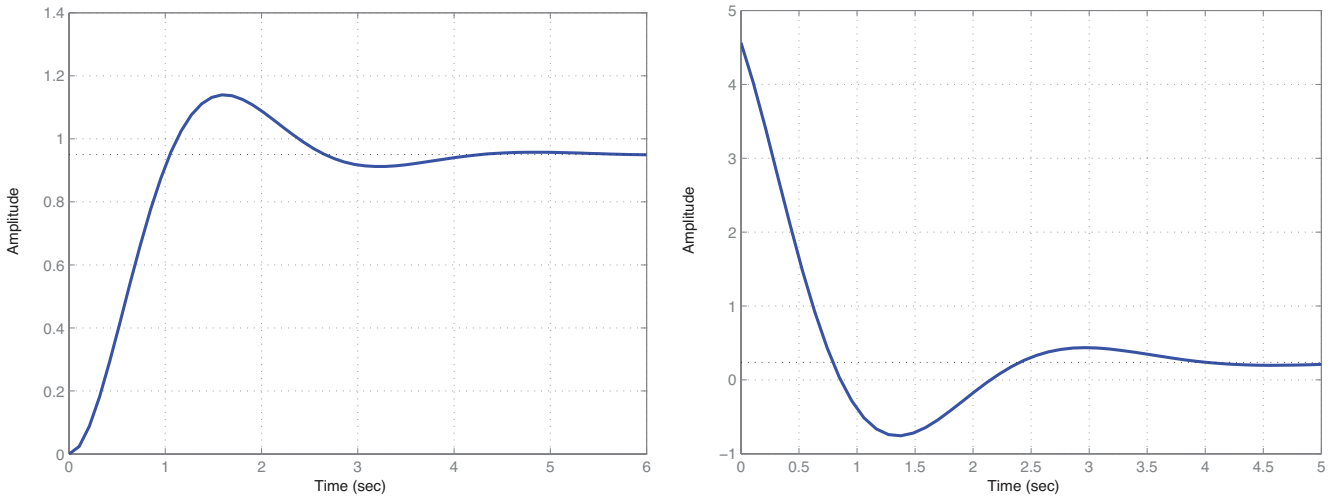
Combining these constraints we get

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & -\frac{19}{4} & \frac{1}{4} \end{bmatrix} \begin{bmatrix} f_2 \\ f_1 \\ f_0 \\ g_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \\ 4.803 \\ 0 \end{bmatrix}.$$

The solution to this equation is $(f_2, f_1, f_0, g_2) = (1, 2, 0.24, 4.56)$ and the control law is

$$C(s) = 4.56 \frac{(s + 0.5)^2}{s^2 + 2s + 0.24}. \tag{3.11}$$

The resulting step response and control signal are shown in Figures 3.7a and 3.7b respectively. The percentage overshoot of the response is $\%\text{OS} = 0.199$, the settling time is $T_s = 3.8$ and $e_{ss} = 0.05$. We could reduce the steady-state error by choosing a larger value for δ. ▲



(a) Step response. *(b) Control signal.*

Figure 3.7: Step response and control signal for Example 3.3.3 with control law (3.11).

3.A Appendix: Polynomials

Consider two polynomials $a, b \in \mathbb{R}[s]$

$$a(s) = a_n s^n + a_{n-1}s^{n-1} + \cdots a_1 s + a_0, \qquad a_n \neq 0$$
$$b(s) = b_m s^m + b_{m-1}s^{m-1} + \cdots b_1 s + b_0, \qquad b_m \neq 0.$$

The polynomials $a(s)$, $b(s)$ are coprime if they do not have any common roots. There are a bunch of ways to test if $a(s)$ and $b(s)$ are coprime from their coefficients. In this appendix we present a method that is relevant to pole placement. Define an $(n+m) \times m$ real matrix from $a(s)$ as

$$\mathbf{T}(a(s), m) \coloneqq \begin{bmatrix} a_n & 0 & \cdots & 0 \\ a_{n-1} & a_n & \ddots & 0 \\ \vdots & a_{n-1} & \ddots & 0 \\ a_0 & \vdots & \ddots & a_n \\ 0 & a_0 & \ddots & a_{n-1} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & a_0 \end{bmatrix}$$

and an $(n+m) \times n$ matrix from $b(s)$

$$\mathbf{T}(b(s), n) \coloneqq \begin{bmatrix} b_m & 0 & \cdots & 0 \\ b_{m-1} & b_m & \ddots & 0 \\ \vdots & b_{m-1} & \ddots & 0 \\ b_0 & \vdots & \ddots & b_m \\ 0 & b_0 & \ddots & b_{m-1} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & b_0 \end{bmatrix}.$$

Each of these matrices have the property that elements in their diagonals are equal. Such matrices are called Toeplitz matrices. Further define the square $(n+m)\times(n+m)$ matrix

$$\mathbf{S}(a(s),b(s)) \coloneqq \begin{bmatrix}\mathbf{T}(a(s),m) & \mathbf{T}(b(s),n)\end{bmatrix}.$$

This is called a Sylvester matrix. We have the following theorem

Theorem 3.A.1. *The following statements are equivalent*

(a) $a(s)$ *and* $b(s)$ *are coprime.*

(b) $\det\left(\mathbf{S}(a(s),b(s))\right)\neq 0$.

(c) There exist unique polynomials $x,y\in\mathbb{R}[s]$ *with* $\deg\left(x(s)\right)<m$ *and* $\deg\left(y(s)\right)<n$ *such that*

$$a(s)x(s)+b(s)y(s)=1.$$

The equation in condition (c) of Theorem 3.A.1 is called a Bezout equation.

Example 3.A.1. (Coprime) Consider $a(s)=s^2-2s$ and $b(s)=s-1$. These polynomials are coprime. Their associated Sylvester matrix is

$$\mathbf{S}(a(s),b(s))=\begin{bmatrix}1 & 1 & 0\\ -2 & -1 & 1\\ 0 & 0 & -1\end{bmatrix}.$$

We have $\det\left(\mathbf{S}(a(s),b(s))\right)=-1$ as expected. We now turn the Bezout equation into $n+m=2+1=3$ linear equations. Let $x(s)=x_0$ and $y(s)=y_1s+y_0$. Then expanding the polynomials out and equating coefficients we get

$$\mathbf{S}(a(s),b(s))\begin{bmatrix}x_0\\ y_1\\ y_0\end{bmatrix}=\begin{bmatrix}0\\ 0\\ 1\end{bmatrix}.$$

The unique solution is $x(s)=-1$ and $y(s)=s-1$. ▲

Turning the Bezout equation into $n+m$ linear equations as in the above example makes it easy to see why conditions (b) and (c) in Theorem 3.A.1 are equivalent.

Example 3.A.2. (Not Coprime) Consider $a(s)=s^2-s$ and $b(s)=s-1$. Both polynomials have a root at $s=1$ so they are not coprime. The associated Sylvester matrix is

$$\mathbf{S}(a(s),b(s))=\begin{bmatrix}1 & 1 & 0\\ -1 & -1 & 1\\ 0 & 0 & -1\end{bmatrix}.$$

The first two columns are equal and so clearly we have $\det\left(\mathbf{S}(a(s),b(s))\right)=0$ as expected. You can check that the Bezout equation has no solution. ▲

Chapter 4

Discretization of continuous-time controllers

In this chapter we look at one of the main approaches for digital controller design: A continuous-time controller is already in place that satisfies the performance specifications. We then use a discrete-time approximation to the continuous-time controller, expecting that this should lead to a good digital controller.

Contents

4.1 Introduction

Suppose we have designed a controller that works well for the system in Figure 4.1. From the analog controller

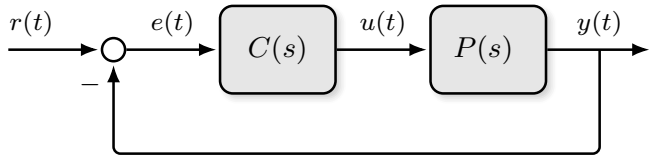


Figure 4.1: Continuous-time unity feedback control system.

$C(s)$ we want to select a sampling period T and a digital control $D[z]$ so that the control works well for the system in Figure 4.2. More concisely, we want to find $D[z]$ so that the upper system in Figure 4.3 is

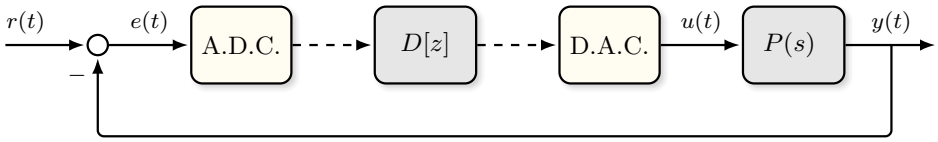


Figure 4.2: Sampled-data unity feedback control system.

approximated by the lower system.

4.2 Ideal sample and zero order hold

We begin with the interface components between continuous-time and discrete-time. First the ideal sampler shown in Figure 4.4. Here T is the sampling period and the output $y(kT)$ is viewed as a function of k. We

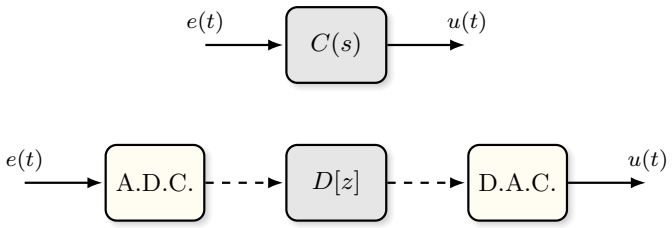


Figure 4.3: Discrete approximation of a continuous control law.

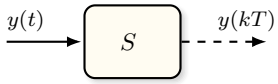
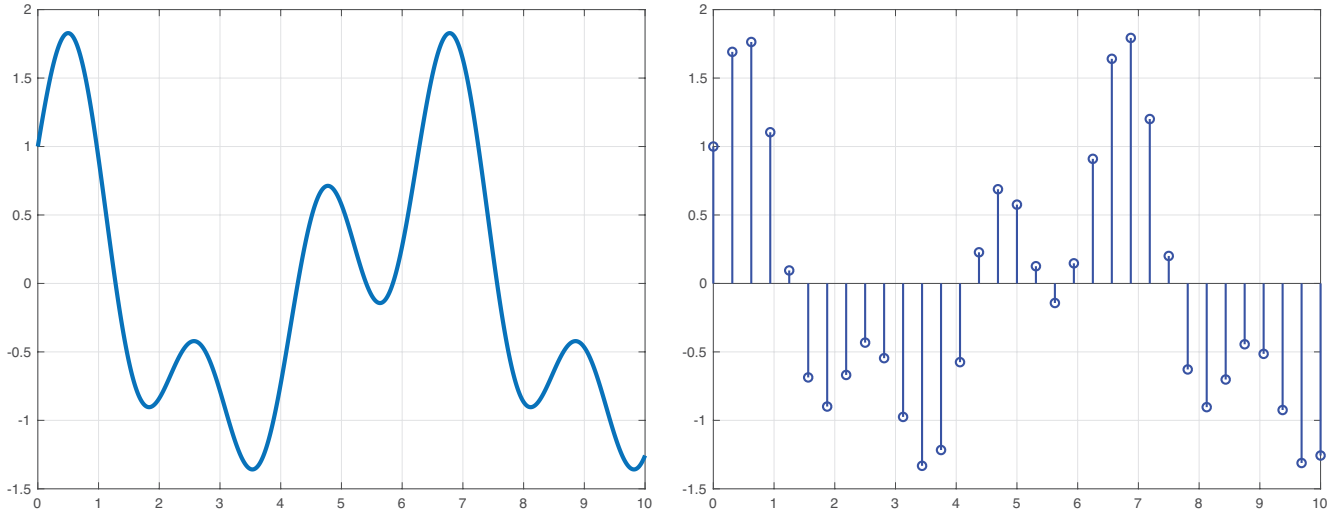


Figure 4.4: Ideal sample block.

usually write $y[k] = y(kT)$. In this course we will always model an A/D using an ideal sampler. The action of the ideal sampler is illustrated in Figure 4.5.



(a) Input $y(t)$. *(b) Output $y(kT)$.*

Figure 4.5: Action of the ideal sample block.

Next, the hold operator shown in Figure 4.6 models a D/A block. There are more general models but for

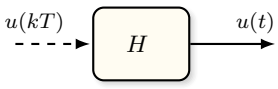


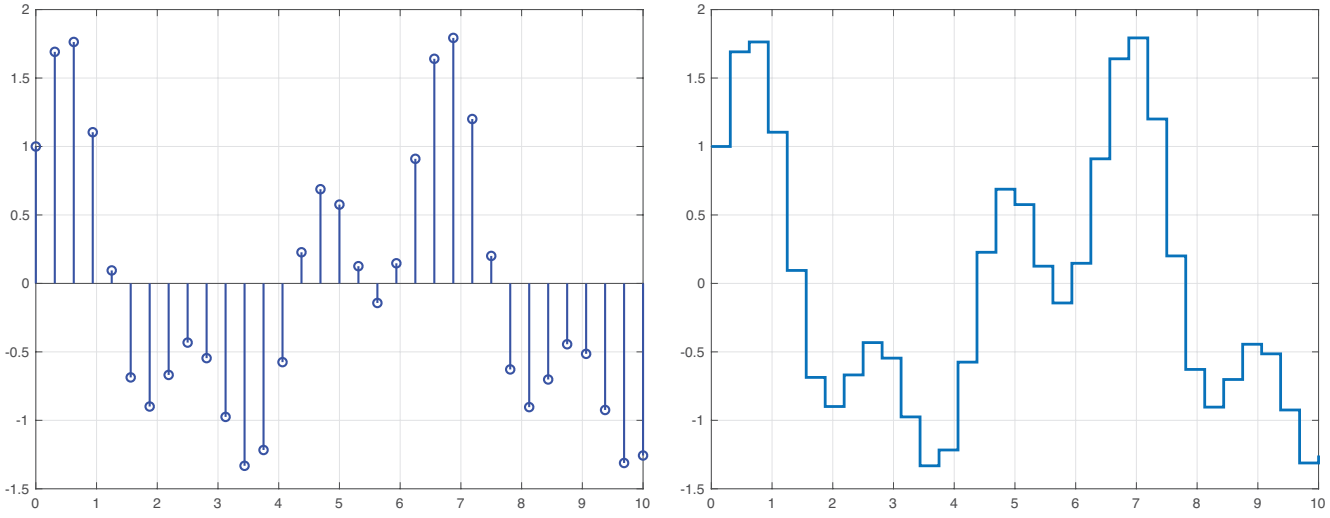
Figure 4.6: Zero order hold.

our purposes this one is sufficient. The hold operates at the same period T as the sampler and the two blocks are always assumed to be synchronized. The definition of the output of the hold is

$$u(t) = u(kT), \qquad kT \le t < (k+1)T.$$

The action of the zero order hold is illustrated in Figure 4.7.

Exercise 4.1. Prove that both the ideal sampler S and the zero order hold H are linear systems.



(a) Input $u(kT)$. *(b) Output $u(t)$.*

Figure 4.7: Action of the zero order hold block.

Thus $S \circ H$ and $H \circ S$, the composition of linear functions, are linear. The thing that makes real digital control nonlinear is the quantizer (cf. Example 5.3.1). We ignore quantization in this course. This amounts to assuming that the interface elements have infinite precision.

(a) Sample, then hold $H \circ S$. *(b) Hold, then sample $S \circ H$.*

Figure 4.8: Connecting the interface elements.

Let's examine what happens when we connect our idealized interface elements in series as shown in Figure 4.8. The system in Figure 4.8a is $H \circ S$ because $u = H(S(y))$. If we were to sketch $u(t)$ when $y(t) = t$, the ramp, we'll see that $u(t) = y(t)$ only at the sample instants $t = kT$. Next let $y(t) = \mathbf{1}(t)$ be the unit step at time $t = 0$. You can check that $u(t)$ is a unit step too. Now let $y(t)$ be the unit step delayed by $T/2$ seconds $y(t) = \mathbf{1}(t - T/2)$, then you'll see that the output is $u(t) = \mathbf{1}(t - T)$, the unit step delayed by T seconds. So shifting the input by $T/2$ seconds resulted in the output being shifted by T seconds. The conclusion is that $H \circ S$ is linear but it is not time-invariant. Therefore the system $H \circ S$ does not have a transfer function.

The system in Figure 4.8b is $S \circ H$ – hold and then sample. For this system the output is always equal to the input so $S \circ H$ is the identity system.

In terms of S and H, a digital controller has the form shown in Figure 4.9. The input is sampled; the

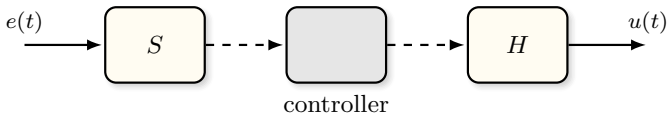


Figure 4.9: Digital controller in terms of ideal sample and hold.

sampled values are processed in discrete-time; then there's conversion back to continuous time. Even if the controller is doing time-invariant processing, the overall digital controller from $e(t)$ to $u(t)$ is not time-invariant, so it doesn't have a transfer function.

4.3 Discrete approximations

The implicit assumption is that the continuous-time controller performs satisfactorily. Therefore the discretization will be constructed to preserve certain properties of the continuous-time controller. There are two common types of discretization.

1. Approximate the continuous-time controller using numerical integration. The most commonly used approach is the trapezoidal approximation (bilinear transformation).
2. Choose $D[z]$ so that it matches the response of $C(s)$ at the sample instants for certain types of input signals. These include the impulse-invariance method and the step-invariance method.

4.3.1 Numerical integration

Consider a continuous-time controller which is an integrator $C(s) = 1/s$ with input $e(t)$ and output $u(t)$. Then

$$\begin{aligned} U(s) &= \frac{1}{s}E(s) \\ \Rightarrow\ \dot{u}(t) &= e(t) \\ \Rightarrow\ u(t) - u(t_0) &= \int_{t_0}^{t} e(\tau)\mathrm{d}\tau \\ \Rightarrow\ u(t) &= u(t_0) + \int_{t_0}^{t} e(\tau)\mathrm{d}\tau. \end{aligned}$$

If we sample $u(t)$ at the rate T, then the difference between successive samples is given by

$$u(kT) - u((k-1)T) = \int_{(k-1)T}^{kT} e(\tau)\mathrm{d}\tau$$

so that at the sampling instances we have the exact relationship

$$u[k] = u[k-1] + \int_{(k-1)T}^{kT} e(\tau)\mathrm{d}\tau = u[k-1] + (\text{area under } e(t) \text{ over } (k-1)T \le t < kT)$$

where $u[k] \coloneqq u(kT)$ and $u[k-1] \coloneqq u((k-1)T)$. Our goal is to approximate the integral

$$\int_{(k-1)T}^{kT} e(\tau)\mathrm{d}\tau. \tag{4.1}$$

Left-side rule

For the left-side rule we approximate (4.1) by assuming that $e(t)$ is constant over $(k-1)T \le t < kT$ with value $e((k-1)T)$, see Figure 4.10. This approximation gives

$$\int_{(k-1)T}^{kT} e(\tau)\mathrm{d}\tau \approx e[k-1]T$$

which results in the approximated control signal

$$u[k] = u[k-1] + Te[k-1].$$

If we take z-transforms of this expression with zero initial conditions we get the discrete-time TF

$$\frac{U[z]}{E[z]} = \frac{T}{z-1}.$$

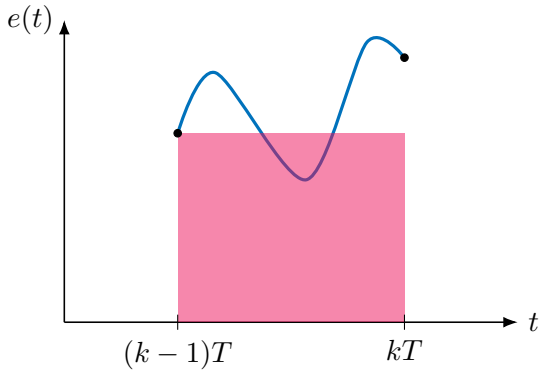


Figure 4.10: Left-side rule approximation.

This TF is just the continuous-time controller $C(s) = 1/s$ where we've made the substitution

$$s = \frac{z-1}{T} \qquad \textbf{(left-side rule)}. \tag{4.2}$$

In summary, to approximate *any* analog controller $C(s)$ using the left-side rule, apply the substitution (4.2) to obtain the digital controller

$$D_{\text{lsr}}[z] = C\left(\frac{z-1}{T}\right).$$

The TF $D_{\text{lsr}}[z]$ defines the difference equation to be implemented in software.

Example 4.3.1. The left-side rule approximation of the controller

$$C(s) = \frac{s^2+2s+1}{s^2+4s+4} = \left(\frac{s+1}{s+2}\right)^2$$

equals

$$D[z] = C\left(\frac{z-1}{T}\right) = \left(\frac{z-1+T}{z-1+2T}\right)^2.$$

▲

Remark 4.3.1. The left-side rule is very conveniently expressed in state-space form. Suppose we have a CT controller with state-space model

$$\begin{aligned}\dot{x}_c(t) &= A_c x(t) + B_c e(t)\\ u(t) &= C_c x_c(t) + D_c e(t).\end{aligned}$$

Taking Laplace transforms (see Section 5.2.1) we get

$$\begin{aligned}sX_c(s) &= A_c X(s) + B_c E(s)\\ U(s) &= C_c X_c(s) + D_c E(s).\end{aligned}$$

To get the left-side rule approximation make the substitution (4.2)

$$\begin{aligned}\frac{z-1}{T}X_c[z] &= A_c X_c[z] + B_c E[z]\\ U[z] &= C_c X_c[z] + D_c E[z].\end{aligned}$$

Going back to the time-domain gives

$$\begin{aligned}x_c[k+1] &= (I + TA_c)x_c[k] + TB_c e[k]\\ u[k] &= C_c x[k] + D_c e[k].\end{aligned}$$

So the state-space formula for the left-side rule is $(A_c, B_c, C_c, D_c) \mapsto (I+TA_c, TB_c, C_c, D_c)$. It is exactly Euler's approximation of an ODE. ♦

Right-side rule

For the right-side rule we approximate (4.1) by assuming that $e(t)$ is constant over $(k-1)T \le t < kT$ with value $e(kT)$, see Figure 4.11. This approximation gives

$$\int_{(k-1)T}^{kT} e(\tau)\mathrm{d}\tau \approx e[k]T$$

which results in the approximated control signal

$$u[k] = u[k-1] + Te[k].$$

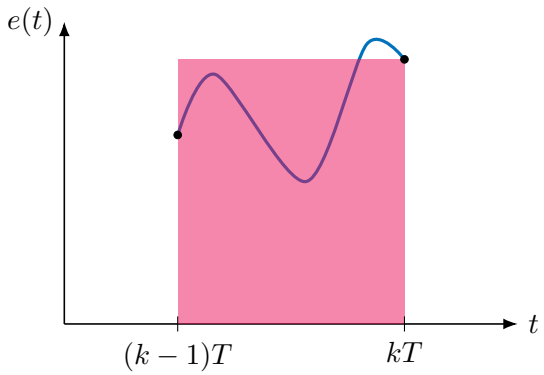


Figure 4.11: Right-side rule approximation.

If we take z-transforms of this expression with zero initial conditions we get the discrete-time TF

$$\frac{U[z]}{E[z]} = \frac{Tz}{z-1}.$$

This TF is just the continuous-time controller $C(s) = 1/s$ where we've made the substitution

$$s = \frac{z-1}{Tz} \qquad \textbf{(right-side rule)}. \tag{4.3}$$

In summary, to approximate *any* analog controller $C(s)$ using the right-side rule, apply the substitution (4.3) to obtain the digital controller

$$D_{\mathrm{rsr}}[z] = C\left(\frac{z-1}{Tz}\right).$$

The TF $D_{\mathrm{rsr}}[z]$ defines the difference equation to be implemented in software.

Example 4.3.2. The right-side rule approximation of the controller

$$C(s) = \frac{s^2+2s+1}{s^2+4s+4} = \left(\frac{s+1}{s+2}\right)^2$$

equals

$$D[z] = C\left(\frac{z-1}{Tz}\right) = \left(\frac{(T+1)z-1}{(2T+1)z-1}\right)^2.$$

▲

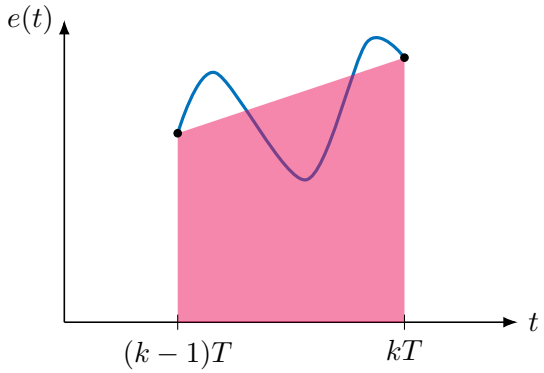


Figure 4.12: Trapezoidal approximation.

Trapezoidal rule

This method is motivated by considering the trapezoidal approximation of an integrator. We approximate the integral (4.1) as shown in Figure 4.12 using linear interpolation. That is, we assume that $e(t)$ is a line with slope $(e[k] - e[k-1])/T$ over the time interval $(k-1)T \leq t < kT$. This approximation gives

$$\int_{(k-1)T}^{kT} e(\tau)\mathrm{d}\tau \approx \int_{(k-1)T}^{kT} \left(\frac{e[k] - e[k-1]}{T}\right) \tau \; \mathrm{d}\tau - (k-1)e[k] + ke[k-1]$$
$$= \frac{T}{2}\left(e[k] + e[k-1]\right).$$

Error analysis for the trapezoidal rule shows that the error is $\mathcal{O}(T^3)$

$$\int_{(k-1)T}^{kT} e(\tau)\mathrm{d}\tau = \frac{T}{2}\left(e[k] + e[k-1]\right) + MT^3 e^{(3)}(\tau)$$

where M is a constant and τ is a time between $(k-1)T$ and kT. Using the trapezoidal rule we get the approximation

$$u[k] = u[k-1] + \frac{T}{2}\left(e[k] + e[k-1]\right).$$

If we take z-transforms of this expression with zero initial conditions we get the discrete-time TF

$$\frac{U[z]}{E[z]} = \frac{T}{2}\frac{z+1}{z-1}.$$

This TF is just the continuous-time controller $C(s) = 1/s$ where we've made the substitution

$$s = \frac{2}{T}\frac{z-1}{z+1} \qquad \textbf{(bilinear transformation)} \tag{4.4}$$

In summary, to approximate *any* analog controller $C(s)$ using the trapezoidal rule, apply the substitution (4.4) to obtain the digital controller

$$D_{\mathrm{bt}}[z] = C\left(\frac{2}{T}\frac{z-1}{z+1}\right).$$

The TF $D_{\mathrm{bt}}[z]$ defines the difference equation to be implemented in software.

Example 4.3.3. Consider the controller

$$\frac{U(s)}{E(s)} = C(s) = \frac{3s+1}{s^2+2s+1}.$$

The discretized controller obtained from the trapezoidal rule is given by

$$\begin{aligned} D[z] &= C\left(\frac{2}{T}\frac{z-1}{z+1}\right) \\ &= \frac{3\left(\frac{2}{T}\frac{z-1}{z+1}\right)+1}{\left(\frac{2}{T}\frac{z-1}{z+1}\right)^2 + 2\left(\frac{2}{T}\frac{z-1}{z+1}\right)+1} \\ &= \frac{6T(z-1)(z+1)+T^2(z+1)^2}{4(z-1)^2+4T(z-1)(z+1)+T^2(z+1)^2} \\ &= \frac{(T^2+6T)z^2+2T^2z+T^2-6T}{(T^2+4T+4)z^2+(2T^2-8)z+T^2-4T} \\ &= \frac{T^2+6T+2T^2z^{-1}+(T^2-6T)z^{-2}}{T^2+4T+4+(2T^2-8)z^{-1}+(T^2-4T)z^{-2}}. \end{aligned}$$

Define the constants

$$\begin{aligned} a_0 &\coloneqq T^2+4T+4, \qquad a_1 \coloneqq 2T^2-8, \qquad a_2 = T^2-4T \\ b_0 &\coloneqq T^2+6T, \qquad b_1 \coloneqq 2T^2, \qquad b_2 \coloneqq T^2-6T \end{aligned}$$

so that we can write $D[z]$ compactly as

$$D[z] = \frac{b_0+b_1z^{-1}+b_2z^{-2}}{a_0+a_1z^{-1}+a_2z^{-2}}.$$

Cross multiplying and returning to the time-domain we obtain the difference equation to be implemented in software

$$a_0u(kT)+a_1u((k-1)T)+a_2u((k-2)T) = b_0e(kT)+b_1e((k-1)T)+b_2e((k-2)T).$$

To implement this difference equation the computer will have to store two past values of the tracking error and two past values of the control signal in memory. ▲

Remark 4.3.2. The calculations in the previous example can be done in MATLAB using the following code.

```
s = tf('s');
C = (3*s + 1)/(s^2 + 2*s + 1);
T = 0.01; % sampling period - MATLAB won't let you leave T as a parameter
D = c2d(C, T, 'tustin');
```

♦

Remark 4.3.3. The trapezoidal rule is also known as Tustin's method after the British engineer whose work on nonlinear circuits stimulated a great deal of interest in this approach. The transformation (4.4) is called the bilinear transformation from consideration of its mathematical form [Franklin et al., 1998]. ♦

Remark 4.3.4. It can be derived that if (A_c, B_c, C_c, D_c) is a continuous-time state-space model for $C(s)$, then the discrete-time state-space model for $D[z]$, obtained via the bilinear transformation, is (A_d, B_d, C_d, D_d) where

$$A_d = \left(I-\frac{T}{2}A_c\right)^{-1}\left(I+\frac{T}{2}A_c\right), \qquad B_d = \frac{T}{2}\left(I-\frac{T}{2}A_c\right)^{-1}B, \qquad C_d = C_c(I+A_d), \qquad D_d = D_c+C_cB_d.$$

This state formula is valid provided the indicated inverse exists, that is, $2/T$ is not an eigenvalue of A_c. ♦

Comparison of the approximations based on numerical integration

A reasonable requirement for the three approximations we've seen is that

(a) As $T \to 0$, the approximations should become perfect, i.e., the behaviour of the discretized system approaches that of the continuous-time system.

(b) The continuous-time controller is $C(s)$ stable if, and only if its approximation $D[z]$ is stable (preservation of stability).

It can be shown that all three methods satisfy property (a). On the other hand, only the bilinear transformation satisfies property (b). To see this, suppose that $C(s)$ has a pole at $s = \lambda$ with $\mathrm{Re}\,(\lambda) < 0$. The bilinear transformation maps the continuous-time pole of $C(s)$ at $s = \lambda$ to a discrete-time pole of $D[z]$ at

$$z = \frac{1 + \frac{T}{2}\lambda}{1 - \frac{T}{2}\lambda}.$$

Looking at Figure 4.13, it is clear that the pole $\lambda \in \mathbb{C}^-$ gets mapped to one whose magnitude is less than one.

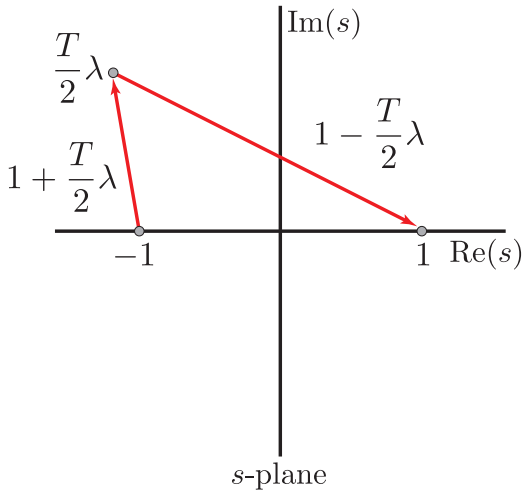


Figure 4.13: Bilinear transformation preserves stability.

As $T \to 0$, all three methods are very similar. In practice the bilinear transformation is the most popular because it preserves stability of the controller for any sampling period T. Note that preserving stability of the controller does not mean that stability of the closed-loop system is preserved; closed-loop stability depends on the sampling period.

4.3.2 Step-invariant transformation

A simple and common way to approximate $C(s)$ is shown in Figure 4.14. The discrete-time system $D[z] = SCH$ is the **step invariant transformation** of the continuous-time system $C(s)$. Why step-invariant? Because the two systems in Figure 4.15 have the same step response. The step-invariance method is what MATLAB uses by default when you type `c2d`. It is also the discretization method we'll use when we discretize the plant to do direct design in Chapter 7.

In Chapter 7 we'll show that when we compute $D[z]$ from $C(s)$ by the step-invariance method, the poles of $C(s)$ are mapped to the poles of $D[z]$ by the mapping $z = \mathrm{e}^{sT}$. Therefore, if $\mathrm{Re}\,(s) < 0$, then $|z| < 1$ and stability is preserved. Unfortunately, there is no simple relationship between the zeros of $C(s)$ and those of $D[z]$. Non-minimum phase zeros of $C(s)$ can be mapped to minimum phase zeros of $D[z]$. In fact, there may even be no zeros present in $C(s)$, yet $D[z]$ may have zeros in $|z| \geq 1$.

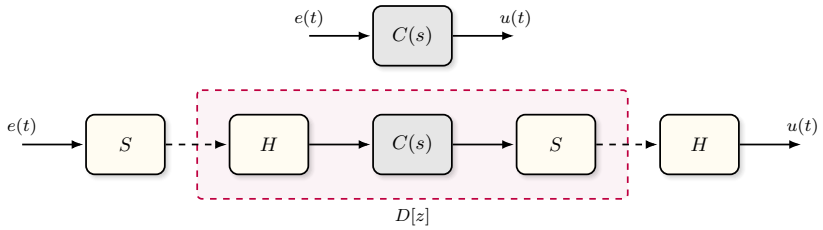


Figure 4.14: Step-invariance method.

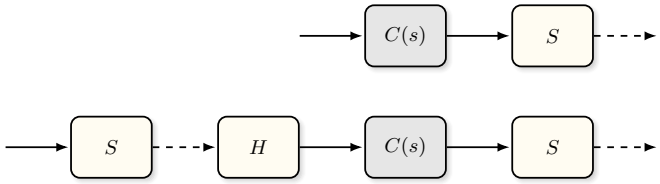


Figure 4.15: These two systems have the same step response.

Example 4.3.4. Suppose

$$C(s) = \frac{1}{(s+1)(s^2+s+1)}.$$

The controller is stable since all its poles have negative real part and it's minimum phase since it has no zeros. For $T = 0.1$ we find, using the MATLAB command `c2d`, that

$$D[z] = 1.6 \times 10^{-4} \frac{(z+3.549)(z+0.255)}{(z-0.905)(z-0.9477+j0.82)(z-0.9477-j0.82)}.$$

The discretized controller has all its poles inside the unit disk as expected but it also has two zeros at -3.549 and -0.255! ▲

The above observations indicate that there is no simple way to quantitatively determine the effects of discretization on the properties of the continuous-time transfer function. Therefore, if we have a continuous-time controller $C(s)$ which works satisfactorily, we should just try to use a discretization method which preserves reasonably well the qualitative properties of $C(s)$, and hope that the resulting digital controller will also perform satisfactorily. Unlike direct discrete-time design, its performance is not guaranteed, so we always need to verify the properties of the closed-loop system using simulation.

4.4 Design based on approximating continuous-time controllers

Now we turn to the following digital control design procedure which is often called emulation:

1. Design a continuous-time controller $C(s)$, if one does not already exist, to achieve the required design specifications.
2. Approximate $C(s)$ using one of the discretization procedures to give $D[z]$.
3. Simulate the closed-loop sampled-data system.
4. If simulations are satisfactory, then implement $D[z]$ as either a difference equation or a state-space model on an embedded system.

Example 4.4.1. (Lead controller design) Consider the plant model

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

in the sampled-data configuration of Figure 4.2 with a fixed sampling period of 0.2 seconds. In addition to feedback stability, our given specifications for the closed-loop system are:

1. asymptotic step tracking,
2. percentage overshoot $\%\text{OS} \leq 5\%$,
3. settling time $T_\text{s} \leq 3$ seconds.

> **Exercise 4.2.** Draw a root locus plot for this plant and the proportional controller $C(s) = K_p$ to check that this controller can't place complex conjugate closed-loop poles in the good region.

Since the plant has a pole at the origin, we will achieve asymptotic step tracking so long as we stabilize the loop. We'll design a lead controller using the procedure from Appendix 2.B to meet the remaining specs

$$C(s) = K\frac{\alpha T s + 1}{Ts + 1}, \qquad \alpha > 1,\ T > 0,\ K > 0.$$

The overshoot specification, using the equation (3.3), results a desired phase margin of $\Phi_\text{pm}^\text{des} = 65^\circ$ and the settling time specification gets converted to a desired bandwidth of $\omega_\text{BW}^\text{des} = 2$ rad/s.

To meet the bandwidth specification we pick $\hat{K} = K\sqrt{\alpha}$ so that the gain crossover frequency of $\hat{K}P(j\omega)$ equals 2 rad/s. This gives $\hat{K} = 5.6$. From the Bode plot of $\hat{K}P(j\omega)$ we see that in order to meet the phase margin specification we need to add $\phi_\text{max} = \Phi_\text{pm}^\text{des} - \Phi_\text{pm} = 20^\circ$ of phase at $\omega_m = 2$ rad/s. Using the lead controller design equation (2.21) we get

$$\alpha = \frac{1 + \sin(\phi_\text{max})}{1 - \sin(\phi_\text{max})} = 2.03.$$

This fixes the value of K to be $K = \hat{K}/\sqrt{\alpha} = 3.95$. To get the phase addition at the correct frequency we use (2.19)

$$T = \frac{1}{\omega_m\sqrt{\alpha}} = 0.3523.$$

Our continuous-time lead controller is therefore

$$C(s) = \frac{8.014s + 11.2}{s + 2.84}.$$

The Bode plot of $C(j\omega)P(j\omega)$ gives $\omega_\text{gc} = 2$ rad/s and $\Phi_\text{pm} = 65^\circ$ as desired. Simulating the closed-loop step response we see that the settling time is 1.69 seconds and the overshoot is 3.74% so we meet the specifications.

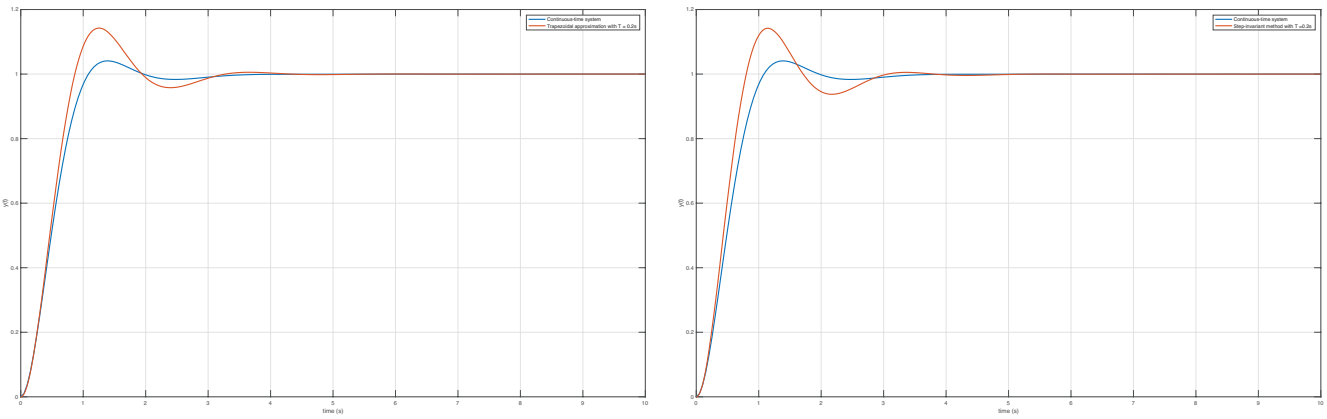
Next we approximate our continuous-time controller for implementation using the code below

```
Dbt = c2d(C, 0.2, 'tustin'); % Trapezoidal approximation
Dsi = c2d(C, 0.2); % step-invariant method
```

to obtain

$$D_\text{bt}[z] = \frac{7.114z - 5.369}{z - 0.5578}, \qquad D_\text{si}[z] = \frac{8.014z - 6.304}{z - 0.5668}.$$

Figure 4.16 compares the response of the purely continuous-time system to the sampled-data system using both of these approximations. In both cases the performance of the sampled-data system is worse than the continuous-time system and fails to meet the overshoot specification. This is because the sampling period of 0.2 is quite large and the approximations are poor. ▲



(a) Bilinear transformation (trapezoidal approximation). *(b) Step-invariant method.*

Figure 4.16: Sampled-data responses of the closed-loop system from Example 4.4.1.

4.4.1 Worsening the plant to improve performance of digital controller

In designing the controller $C(s)$ in Example 4.4.1 we completely ignored the presence of the A/D and D/A. It is often useful to add a refinement in the design of the continuous-time controller. We know that the digital controller will contain sample and hold operators which themselves are mathematical idealizations of real A/D, D/A converters. We can partially account for their effects by designing a continuous-time controller for an augmented plant that approximates the effect of the sample/hold operation.

The combination of sample and then hold from Figure 4.8a is time-varying so it has no TF but we can approximate $H \circ S$ with a system that has the same impulse response. We need the system with impulse response

$$r(t) = \frac{1}{T}\mathbf{1}(t) - \frac{1}{T}\mathbf{1}(t-T).$$

The factor $1/T$ is incorporated to get the output to have unit area. The Laplace transform of $r(t)$ is

$$R(s) = \frac{1 - \mathrm{e}^{-sT}}{sT}. \tag{4.5}$$

Although, strictly speaking, the sample/hold system does not have a transfer function, we can use (4.5), the Laplace transform of its impulse response, as an *approximation* of it. To obtain the frequency response of the sample/hold system, consider the following

$$\begin{aligned} R(j\omega) &= \frac{1 - \mathrm{e}^{-j\omega T}}{j\omega T} \\ &= \mathrm{e}^{-j\omega\frac{T}{2}} \frac{\mathrm{e}^{j\omega\frac{T}{2}} - \mathrm{e}^{-j\omega\frac{T}{2}}}{j\omega T} \\ &= \mathrm{e}^{-j\omega\frac{T}{2}} \frac{2\sin\left(\omega\frac{T}{2}\right)}{\omega T} \\ &= \mathrm{e}^{-j\omega\frac{T}{2}} \frac{\sin\left(\omega\frac{T}{2}\right)}{\omega\frac{T}{2}}. \end{aligned}$$

Observe that at low frequencies

$$R(s) \approx \mathrm{e}^{-s\frac{T}{2}}.$$

So at low frequencies, the sample/hold operation acts like a time delay of $T/2$ seconds. Given the above, a reasonable design approach is to design the controller $C(s)$ for the delayed plant $\mathrm{e}^{-s\frac{T}{2}}P(s)$. How do we design

controllers for $\mathrm{e}^{-s\frac{T}{2}}P(s)$? If we are designing using frequency domain methods, then we simply adjust the phase Bode plot of our plant and proceed as the next example illustrates.

Example 4.4.2. (Control design in the frequency domain) We return to the design problem from Example 4.4.1 but this time we design for the delayed (worsened) plant

$$P_{\mathrm{w}}(s) \coloneqq \mathrm{e}^{-s\frac{T}{2}}\frac{1}{s(s+2)}.$$

The specifications remain unchanged. We design a lead controller. To meet the bandwidth specification we again pick $\hat{K}$ so that the gain crossover frequency of $\hat{K}P_{\mathrm{w}}(j\omega)$ is 2 rad/s; the delay has no effect on this value $\hat{K} = 5.6$. From the Bode plot of $\hat{K}P_{\mathrm{w}}(j\omega)$ we see that in order to meet the phase margin specification we need to add $\phi_{\max} = 31.3°$ of phase at $\omega_m = 2$ rad/s. Using the lead controller design equations we get

$$\alpha = \frac{1+\sin(\phi_{\max})}{1-\sin(\phi_{\max})} = 3.1622$$

which yields $K = \hat{K}/\sqrt{\alpha} = 3.1623$. To get this phase addition at the desired frequency we use

$$T = \frac{1}{\omega_m\sqrt{\alpha}} = 0.2823.$$

Our overall continuous-time lead controller is

$$C(s) = 10\frac{s+11.2}{s+3.542}.$$

The bilinear and step-invariant approximations of this controller are, respectively,

$$D_{\mathrm{bt}}[z] = \frac{8.211z-6.557}{z-0.4769}, \qquad D_{\mathrm{si}}[z] = \frac{10z-8.395}{z-0.4924}.$$

Figure 4.17 shows the closed-loop response of the sampled-data system with the controllers D_{bt} and D_{si} applied

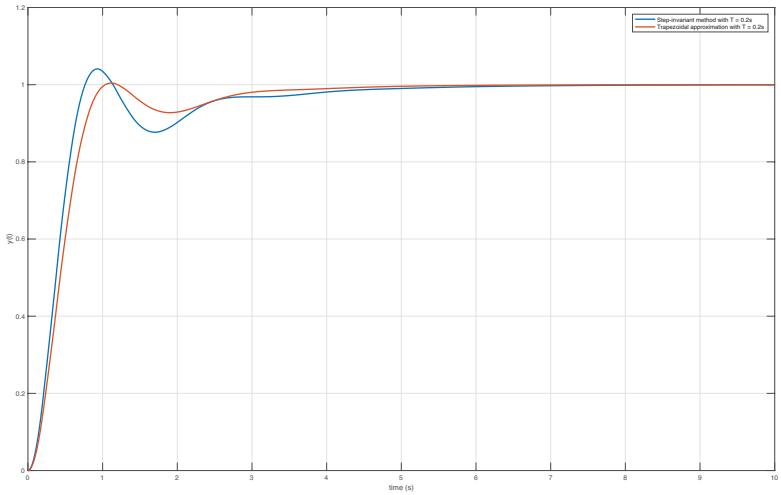


Figure 4.17: Closed-loop step response of sampled-data system from Example 4.4.2.

to the original plant $P(s)$. We see that the overshoot specification is met but the response is now a bit sluggish. This gives some justification to the idea of worsening the plant for control design. ▲

If we're designing $C(s)$ in the s-domain or the time-domain instead of the frequency domain, then we would like to approximate the delay e^{-sT} with a rational function. Bring in the first order **Padé approximation**[1]

$$\mathrm{e}^{-sT} \approx \frac{1 - \frac{T}{2}s}{1 + \frac{T}{2}s}.$$

Based on this approximation, the sample/hold operator transfer function would be approximated by

$$R(s) = \frac{1 - \mathrm{e}^{-sT}}{sT} \approx \frac{1}{1 + s\frac{T}{2}}.$$

To recap, when using s-domain or time-domain design methods, we will design the continuous-time controller $C(s)$ to satisfy the design specifications for the worsened plant

$$P_{\mathrm{w}}(s) \coloneqq \frac{1}{1 + s\frac{T}{2}} P(s) \approx R(s)P(s)$$

which models, in some sense, the effect of the sample and hold operations.

Example 4.4.3. (Control design in the s-domain) We again return to the design problem from Example 4.4.1 but this time we'll design the controller using pole placement. Since this is an s-domain design technique we'll design for the worsened plant

$$P_{\mathrm{w}}(s) = \frac{1}{1 + 0.1s} \frac{1}{s(s+2)} = \frac{10}{s(s+10)(s+2)}.$$

From the time-domain specifications we get that the good region is

$$\mathbb{C}_g = \{s \in \mathbb{C} : \mathrm{Re}\,(s) \leq -4/3\} \cap \{s \in \mathbb{C} : |\arg(s)| \geq 0.74\pi\}.$$

The worsened plant is 3rd order so we need a second order controller

$$C(s) = \frac{g_2 s^2 + g_1 s + g_0}{f_2 s^2 + f_1 s + f_0}.$$

Our desired pole locations are $\{-3 \pm j, -6, -7, -8\}$ which gives the desired ch.p. $\pi_{\mathrm{des}}(s) = s^5 + 27s^4 + 282s^3 + 1422s^2 + 3476s + 3360$. Solving the design equation (3.6) gives the controller

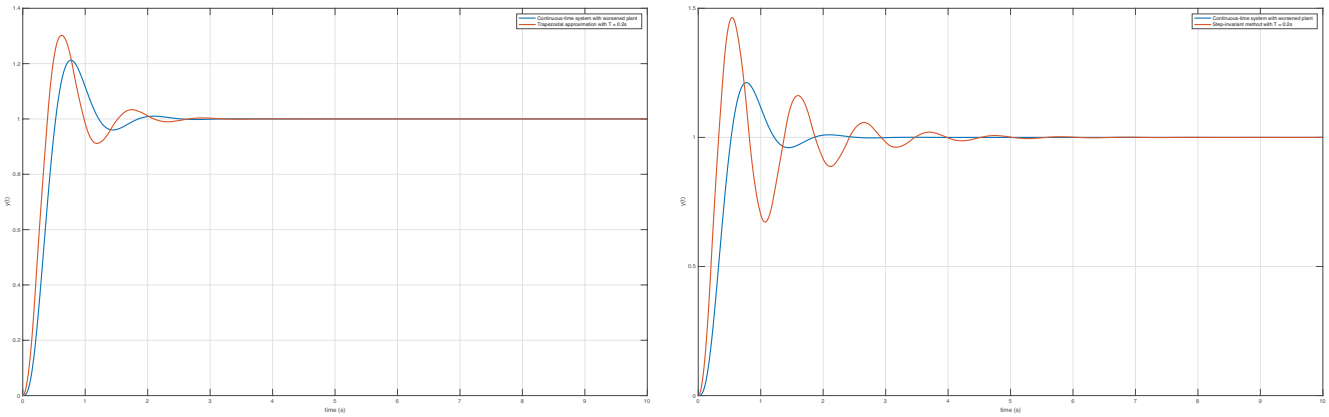
$$C(s) = \frac{27.6s^2 + 367.2s + 672}{s^2 + 15s + 82}.$$

The bilinear and step-invariant approximations of this controller are, respectively,

$$D_{\mathrm{bt}}[z] = \frac{21.4z^2 - 12.58z - 0.7229}{z^2 - 0.1084z + 0.09639}, \qquad D_{\mathrm{si}}[z] = \frac{27.6z^2 - 19.93z - 0.9949}{z^2 - 0.2355z + 0.04979}.$$

The step responses for the sampled-data systems using the two approximations are shown in Figure 4.18. We see that the performance of $D_{\mathrm{bt}}[z]$ applied to $P(s)$ is comparable to the performance of $C(s)$ applied to the worsened plant $P_{\mathrm{w}}(s)$. This provides some justification for the technique of worsening the plant model. The performance of $D_{\mathrm{si}}[z]$ is inferior to that of $D_{\mathrm{bt}}[z]$ and not very predictable. ▲

[1] More accurate approximations for e^{-sT} can be obtained using higher order Padé approximations. The MATLAB function `pade(T,d)` approximates e^{-sT} as a rational function with numerator and denominator of degree d.



(a) Bilinear transformation (trapezoidal approximation). *(b) Step-invariant method.*

Figure 4.18: Sampled-data responses of the closed-loop system from Example 4.4.3.

Chapter 5

Nonlinear systems

Virtually *all* physical systems are nonlinear in nature. In this chapter we introduce nonlinear systems and investigate basic methods for designing controllers for nonlinear models.

Contents

5.1 State-space models

State-space models are a way of expressing mathematical models in a standard form.

Example 5.1.1. (Cart with Air Resistance) Consider a cart on wheels, driven by a force u and subject to air resistance as in Figure 5.1. Typically air resistance creates a force depending on the velocity $\dot{y}$; let's say this

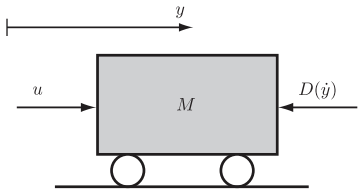


Figure 5.1: A simple cart subject to air resistance.

force is a possibly nonlinear function $D(\dot{y})$. Assuming M is constant, Newton's second law gives

$$M\ddot{y} = u - D(\dot{y}).$$

We are going to put this in a standard form by defining two so-called state variables x_1 and x_2, in this example position and velocity:

$$x_1 \coloneqq y, \qquad x_2 \coloneqq \dot{y}.$$

Then

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= \frac{1}{M}u - \frac{1}{M}D(x_2) \\ y &= x_1. \end{aligned}$$

These equations have the form

$$\begin{aligned} \dot{x} &= f(x,u) \qquad \text{(state equation)} \\ y &= h(x) \qquad \text{(output equation)} \end{aligned} \tag{5.1}$$

where

$$x := \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, \qquad u := \text{applied force}$$

$$f : \mathbb{R}^2 \times \mathbb{R} \to \mathbb{R}^2, \qquad f(x_1, x_2, u) = \begin{bmatrix} x_2 \\ \frac{1}{M}u - \frac{1}{M}D(x_2) \end{bmatrix}$$

$$h : \mathbb{R}^2 \to \mathbb{R}, \qquad h(x_1, x_2) = x_1.$$

The function f is nonlinear if D is; h is linear. Equation (5.1) constitutes a state-space model of the system, and x is called the **state** or **state vector**. Here the plant is a possibly nonlinear system, u (applied force) is the input, y (cart position) is the output, and

$$x = \begin{bmatrix} \text{cart position} \\ \text{cart velocity} \end{bmatrix}$$

is the state of the system. (We'll define state later.)

As a special case, suppose the air resistance is a linear function of velocity

$$D(x_2) = dx_2, \qquad d \text{ is a real constant.}$$

Then f is a linear function of x, u:

$$f(x,u) = Ax + Bu, \quad A := \begin{bmatrix} 0 & 1 \\ 0 & -d/M \end{bmatrix}, \; B := \begin{bmatrix} 0 \\ 1/M \end{bmatrix}.$$

Defining $C := \begin{bmatrix} 1 & 0 \end{bmatrix}$, we get the state-space model

$$\begin{aligned} \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} &= \begin{bmatrix} 0 & 1 \\ 0 & -\frac{d}{M} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{M} \end{bmatrix} u \\ y &= \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}. \end{aligned} \tag{5.2}$$

In other words, when $D(\dot{y})$ is linear, the state-space model has the structure

$$\dot{x} = Ax + Bu, \qquad y = Cx.$$

This model is of a linear time-invariant (LTI) system.

▲

Generalizing Example 5.1.1, we can say that an important class of models is

$$\begin{aligned} \dot{x} &= f(x,u), \; f : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n \\ y &= h(x,u), \; h : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^p. \end{aligned} \tag{5.3}$$

This model is nonlinear, time-invariant. The input u has dimension m, the output y dimension p, and the state x dimension n. An example where $m = 2, p = 2, n = 4$ is shown in Figure 5.2. There we have

$$u = (u_1, u_2), \qquad y = (y_1, y_2), \qquad x = (y_1, \dot{y}_1, y_2, \dot{y}_2).$$

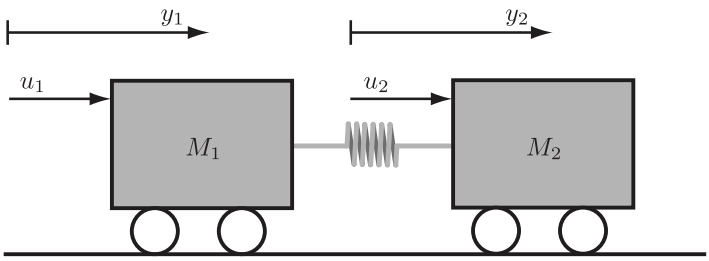


Figure 5.2: A multiple-input multiple-output example.

The LTI special case of (5.3) is

$$\begin{aligned} \dot{x} &= Ax + Bu,\ A \in \mathbb{R}^{n\times n},\ B \in \mathbb{R}^{n\times m} \\ y &= Cx + Du,\ C \in \mathbb{R}^{p\times n},\ D \in \mathbb{R}^{p\times m}. \end{aligned} \tag{5.4}$$

In this course we mostly study single-input single-output systems, i.e., $m = 1$ and $p = 1$.

Now we turn to the concept of the **state of a system**. Roughly speaking, the vector $x(t_0)$ encapsulates all the system dynamics up to time t_0, that is, no additional prior information is required. More precisely, the concept is this: For any t_0 and t_1, with $t_0 < t_1$, knowing $x(t_0)$ and knowing $\{u(t) : t_0 \le t \le t_1\}$, we can compute $x(t_1)$, and hence $y(t_1)$.

Example 5.1.2. (Choosing State Variables)

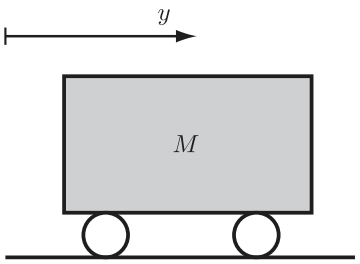


Figure 5.3: A simple cart with no air resistance, no applied force.

Consider the cart system in Figure 5.3 with no applied forces and no air resistance

$$M\ddot{y} = 0.$$

If we try the one-dimensional state $x = y$, then knowing $x(t_0)$ without $\dot{y}(t_0)$, we could not solve for the future cart position. Similarly $x = \dot{y}$ won't work. Since the equation of motion is second order, we need two initial conditions at $t = t_0$, implying we need a 2-dimensional state vector. In general for mechanical systems it is customary to take the state vector x to consist of positions and velocities of all masses. In this example if we take $x_1 = y$ (position), $x_2 = \dot{y}$ (velocity), then the state-space model is

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= 0 \\ y &= x_1. \end{aligned}$$

▲

The concept of a state extends to discrete-time systems.

Example 5.1.3. (Digital Filter) Consider a weighted averaging algorithm defined recursively by the difference equation

$$y[k+2] = 3y[k+1] + 2y[k] + u[k].$$

Here $y[k] \in \mathbb{R}$ is the output of the algorithm at the kth iteration and the above equation represents the update rule. We can recursively solve for the output sequence $\{y[k]\}_{k\geq 0}$ as long as we are given the values

$$y[-1], \qquad y[-2]$$

and the input sequence $\{u[k]\}_{k\geq 0}$. The concept of a state extends to this system: The state vector $x[k_0]$ encapsulates all the system dynamics up to time k_0. For any integers k_0 and k_1, with $k_0 < k_1$, knowing $x[k_0]$ and knowing $\{u[k] : k_0 \leq k \leq k_1\}$, we can compute $x[k_1]$, and hence $y[k_1]$. As in the previous example, neither $y[k-1]$ nor $y[k-2]$ on their own contain enough information to compute future values of y. These are not valid state vectors. You should check that a valid choice of state vector for this system is $x[k] = (x_1[k], x_2[k]) := (y[k], y[k+1])$ and that the corresponding state-space model is

$$\begin{aligned} x_1[k+1] &= x_2[k] \\ x_2[k+1] &= 2x_1[k] + 3x_2[k] + u[k] \\ y[k] &= x_1[k]. \end{aligned}$$

▲

Why are state models useful? First, there's a rich control theory for them. This is covered in ECE488. You may have heard of the Kalman filter; it is based on a state model. Second, for linear systems they give a convenient data type to store a model, namely, the matrices A, B, C, D which makes state models useful for computations (see the design procedures in Chapter 9). Third, they give a simple way to linearize, namely, compute Jacobians. More on this later (Section 5.2). Finally, state-space models provide a systematic method for obtaining discrete-time models from continuous-time models (see Chapter 7).

Example 5.1.4. (Mass-Spring-Damper) Consider the system in Figure 5.4. The dynamic equation is[1]

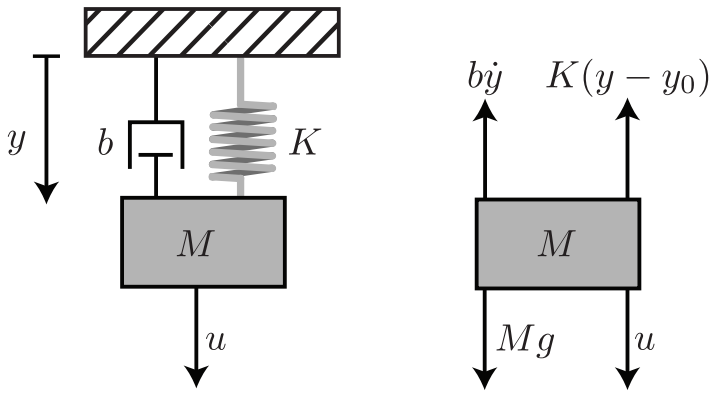


Figure 5.4: Another mass-spring-damper.

$$M\ddot{y} = u + Mg - K(y - y_0) - b\dot{y}.$$

It's appropriate to take

$$x = (x_1, x_2), \quad x_1 = y, \; x_2 = \dot{y}$$

and then we get the equations

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= \frac{1}{M}u + g - \frac{K}{M}x_1 + \frac{K}{M}y_0 - \frac{b}{M}x_2 \\ y &= x_1. \end{aligned}$$

[1] Here we are assuming that the spring is unstretched / uncompressed when $y = y_0$ (cf. Example 5.1.6).

This has the form

$$\begin{aligned} \dot{x} &= Ax + Bu + c \\ y &= Cx, \end{aligned}$$

where

$$A = \begin{bmatrix} 0 & 1 \\ -\frac{K}{M} & -\frac{b}{M} \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ \frac{1}{M} \end{bmatrix}, \; c = \begin{bmatrix} 0 \\ g + \frac{K}{M} y_0 \end{bmatrix}, \quad C = \begin{bmatrix} 1 & 0 \end{bmatrix}.$$

The constant vector c is known, and hence is taken as part of the system rather than as a signal. This system is nonlinear. Can you see why? ▲

Example 5.1.5. (RLC Circuit) Recall the RLC circuit from Example 2.2.4.

$$-\dot{u} + R\dddot{y} + L\ddot{y} + \frac{1}{C} y = 0.$$

It is natural to take the state variables to be the voltage drop across C and the current through L because together these variables characterize the energy stored in the circuit. Let

$$\begin{aligned} x_1 &\coloneqq \text{capacitor voltage} = \frac{1}{C} \int_0^t y(\tau) \mathrm{d}\tau \\ x_2 &\coloneqq \text{inductor current} = y. \end{aligned}$$

Then the KVL equation becomes

$$-u + Rx_2 + L\dot{x}_2 + x_1 = 0$$

and the capacitor equation becomes

$$\dot{x}_1 = \frac{1}{C} x_2.$$

Thus

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & \frac{1}{C} \\ -\frac{1}{L} & -\frac{R}{L} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{L} \end{bmatrix} u.$$

Since this is an LTI system, the state model of this system has the structure

$$\begin{aligned} \dot{x} = Ax + Bu, \quad A &= \begin{bmatrix} 0 & \frac{1}{C} \\ -\frac{1}{L} & -\frac{R}{L} \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ \frac{1}{L} \end{bmatrix} \\ y = Cx + Du, \quad C &= \begin{bmatrix} 0 & 1 \end{bmatrix}, \quad D = 0. \end{aligned}$$

▲

Example 5.1.6. (Car Suspension) Find the governing differential equations and a state model for the simplified quarter model of a car suspension system shown in Figure 5.5. Here M_1 represents 1/4 of the mass of the car chassis while M_2 represents half the axle mass. We assume that the origin of our coordinates, i.e., the zero position $(q_1, q_2) = (0, 0)$, is the position at which all the springs are balanced by the force gravity. In this way we can ignore gravity (cf. Example 5.1.4). The variable q_1 represents how far M_1 has moved from its rest position. The variable q_2 represents how far M_2 has moved from its rest position. The variable d represents the distance from the road to M_2. It is an **exogenous signal**, i.e., it comes from the "outside world."

Applying Newton's equations for translational motion to each mechanical component we obtain the differential equations governing this mechanical system

$$\begin{aligned} M_1\ddot{q}_1 + b_1\dot{q}_1 + k_1 q_1 &= b_1\dot{q}_2 + k_1 q_2 \\ M_2\ddot{q}_2 + b_1\dot{q}_2 + (k_1 + k_2)q_2 &= k_1 q_1 + b_1\dot{q}_1 + k_2 d. \end{aligned}$$

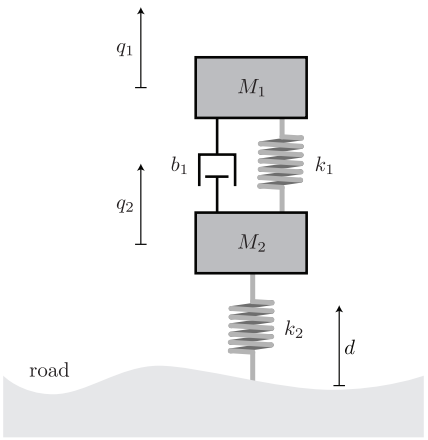


Figure 5.5: Simplified model of a car suspension system.

Exercise 5.1. Re-derive the above differential equations and verify that they are correct.

We now find a state model for this system. As usual, for mechanical systems, we take the position and velocity of each mass as state variables. In this case let

$$x \coloneqq (x_1, x_2, x_3, x_4), \quad x_1 \coloneqq q_1, \ x_2 \coloneqq \dot{q}_1, \ x_3 \coloneqq q_2, \ x_4 \coloneqq \dot{q}_2.$$

Let the input to the system be the road height $u \coloneqq d$. Let the output of the system be the position of the chassis $y \coloneqq q_1$. We get the equations

$$\begin{aligned}
\dot{x}_1 &= x_2 \\
\dot{x}_2 &= \frac{1}{M_1}\left(-k_1 x_1 - b_1 x_2 + k_1 x_3 + b_1 x_4\right) \\
\dot{x}_3 &= x_4 \\
\dot{x}_4 &= \frac{1}{M_2}\left(k_1 x_1 + b_1 x_2 - (k_1 + k_2) x_3 - b_1 x_4 + k_2 u\right) \\
y &= x_1.
\end{aligned}$$

This has the form

$$\begin{aligned}
\dot{x} &= Ax + Bu \\
y &= Cx,
\end{aligned}$$

where

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -\frac{k_1}{M_1} & -\frac{b_1}{M_1} & \frac{k_1}{M_1} & \frac{b_1}{M_1} \\ 0 & 0 & 0 & 1 \\ \frac{k_1}{M_2} & \frac{b_1}{M_2} & -\frac{k_1+k_2}{M_2} & \frac{b_1}{M_2} \end{bmatrix}, \qquad B = \begin{bmatrix} 0 \\ 0 \\ 0 \\ \frac{k_2}{M_2} \end{bmatrix}, \qquad C = \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix}.$$

▲

Example 5.1.7. (DC Motor) We return to the DC motor from Example 2.2.5. If we do not assume that $L \approx 0$, then the equations of motion are

$$\begin{aligned}
J\ddot{\theta} &= -b\dot{\theta} + K_\tau i \\
-v + Ri + L\frac{\mathrm{d}i}{\mathrm{d}t} &+ K_e\dot{\theta} = 0.
\end{aligned}$$

For state variables we take the angular position and velocity of the mechanical subsystem and the inductor current for the electrical subsystem

$$x \coloneqq (x_1, x_2, x_3), \quad x_1 \coloneqq \theta, \; x_2 \coloneqq \dot{\theta}, \; x_3 \coloneqq i.$$

Let the input to the system be the applied voltage $u \coloneqq v$. Let the output of the system be the angular position of the motor shaft $y \coloneqq \theta$. We get the equations

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= \frac{1}{J}\left(-bx_2 + K_\tau x_3\right) \\ \dot{x}_3 &= \frac{1}{L}\left(-K_e x_2 - Rx_3 + u\right) \\ y &= x_1. \end{aligned}$$

This is an LTI system so the state model has the form

$$\begin{aligned} \dot{x} &= Ax + Bu \\ y &= Cx, \end{aligned}$$

where

$$A = \begin{bmatrix} 0 & 1 & 0 \\ 0 & -\frac{b}{J} & \frac{K_\tau}{J} \\ 0 & -\frac{K_e}{L} & -\frac{R}{L} \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ 0 \\ \frac{1}{L} \end{bmatrix}, \quad C = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}.$$

▲

Example 5.1.8. (Magnetic-ball suspension system) Consider an electromagnet suspending an iron ball as in Figure 5.6. Let the input be the voltage u and the output the position y of the ball below the magnet; let

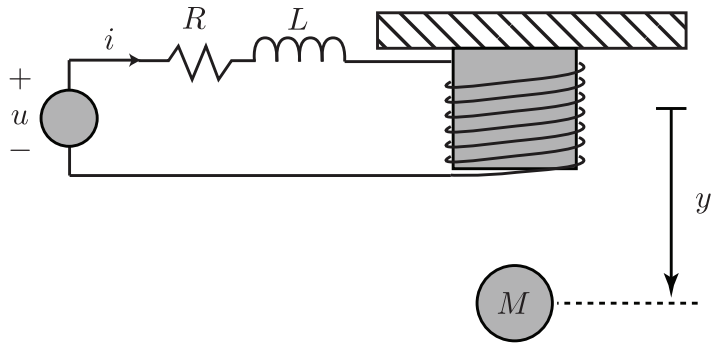


Figure 5.6: A magnetic-ball suspension system.

i denote the current in the circuit. Then

$$L\frac{\mathrm{d}i}{\mathrm{d}t} + Ri = u.$$

Also, it can be derived that the magnetic force on the ball has the form Ki^2/y^2 where K a constant. Thus

$$M\ddot{y} = Mg - K\frac{i^2}{y^2}$$

where g is the acceleration due to gravity. Realistic numerical values are $M = 0.1\text{Kg}$, $R = 15\Omega$, $L = 0.5\text{H}$, $K = 0.0001\text{Nm}^2/\text{A}^2$, $g = 9.8\text{m/s}^2$. Substituting these numbers gives the equations

$$0.5\frac{\mathrm{d}i}{\mathrm{d}t} + 15i = u.$$

$$0.1\ddot{y} = 0.98 - 0.0001\frac{i^2}{y^2}.$$

Define the state variables $x = (x_1, x_2, x_3) \coloneqq (i, y, \dot{y})$. Then the nonlinear state model is $\dot{x} = f(x, u)$, $y = h(x, u)$ where

$$f(x,u) = \begin{bmatrix} -30x_1 + 2u \\ x_3 \\ 9.8 - 0.001\frac{x_1^2}{x_2^2} \end{bmatrix}, \qquad h(x,u) = x_2.$$

▲

Example 5.1.9. (Ball and beam system) Consider a ball and beam system as illustrated in Figure 5.7. A precise mathematical model of the ball and beam system is complicated. Fortunately, approximate models are sufficient for feedback control. The input to this system is the applied voltage u to the DC motor and the

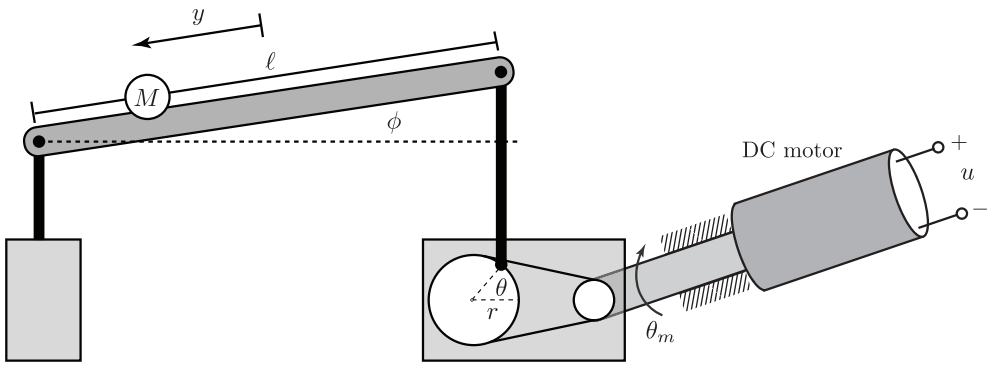


Figure 5.7: A ball and beam system.

output y is the deviation of the ball from the mid-point of the beam.

We start by modelling the ball rolling on the beam. We approximate this subsystem as a point mass rolling along a frictionless surface. Newton's second law gives

$$M\ddot{y} = Mg\sin(\phi)$$

where ϕ is the angle of the beam. Next, we model the relationship between the beam angle ϕ and the angle θ of the big disk. Assuming the connecting rod is always vertical we have that

$$\sin(\phi) = \frac{r}{\ell}\sin(\theta)$$

where r is the radius of the big disk and ℓ is the length of the beam.

Exercise 5.2. Derive the above expression.

The angular displacement of the big disk θ and that of the rotor of the motor θ_m are related by the gear ratio K_g, i.e.,

$$\theta = K_g\theta_m.$$

Finally, if we use the motor model from Example 5.1.7 we have

$$\begin{aligned} J\ddot{\theta}_m &= -b\dot{\theta}_m + K_\tau i \\ -v + Ri + L\frac{\mathrm{d}i}{\mathrm{d}t} &+ K_e\dot{\theta}_m = 0. \end{aligned}$$

The ball dynamics and the motor dynamics are actually coupled since the load inertia J depends on the ball position while the ball velocity $\dot{y}$ depends on the motor velocity $\dot{\theta}_m$. Nevertheless, the above approximation is good enough for control design.

If we take our state vector to be

$$x = (x_1, x_2, x_3, x_4, x_5) \coloneqq (y, \dot{y}, \theta_m, \dot{\theta}_m, i)$$

then we get the nonlinear state model $\dot{x} = f(x, u)$, $y = h(x)$ given by

$$\dot{x} = \begin{bmatrix} x_2 \\ \frac{gr}{\ell} \sin(K_g x_3) \\ x_4 \\ \frac{1}{J}(-bx_4 + K_\tau x_5 \\ \frac{1}{L}(-K_e x_4 - R x_5 + u) \end{bmatrix}$$
$$y = x_1.$$

▲

Exercise 5.3. Obtain a state-space model for the ball and beam system from Example 5.1.9 using the simplified DC motor model from Example 2.2.5 instead of the DC motor model from Example 5.1.7.

Exercise 5.4. In this section we have presented many examples of the state-space models. Which ones were nonlinear?

5.2 Linearization

We saw that nonlinear systems can be modelled by equations of the form (5.3) which we re-write here

$$\dot{x} = f(x, u)$$
$$y = h(x, u).$$

This state-space model includes the linear version (5.4) as a special case, i.e., $f(x, u) = Ax + Bu$, $h(x, u) = Cx + Du$, but also includes a much larger class of systems that includes robotic manipulators, mobile and humanoid robots, population dynamics, economic systems etc. There are techniques for controlling nonlinear systems, but that's an advanced subject.

Example 5.2.1. (Physical Pendulum) Consider the physical pendulum shown in Figure 5.8a. It consists of a rod made of a homogenous material of length ℓ and mass M hanging from a pivot point. Its center of mass is at the midpoint $\ell/2$. A motor connected to the rod at the pivot point provides a torque τ. Assume there is no friction. In this case the equation of motion is

(a) Schematic diagram. *(b) Picking coordinates.*

Figure 5.8: A physical (rod) pendulum.

$$\ddot{\theta} = \frac{3}{M\ell^2}\tau - 1.5\frac{g}{\ell}\sin(\theta).$$

For state variables we take the position and velocity of the pendulum

$$x := (x_1, x_2), \quad x_1 := \theta, \; x_2 := \dot{\theta}.$$

The input is the applied torque $u := \tau$ and the output is taken to be the pendulum's angle $y := \theta$. We get the equations

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= -1.5\frac{g}{\ell}\sin(x_1) + \frac{3}{M\ell^2}u \\ y &= x_1. \end{aligned}$$

This has the form (5.3) where

$$f(x,u) = \begin{bmatrix} x_2 \\ -1.5\frac{g}{\ell}\sin(x_1) + \frac{3}{M\ell^2}u \end{bmatrix}, \qquad h(x,u) = x_1.$$

This system is nonlinear because $f(x,u)$ contains the term $\sin(x_1)$; you can't find constant matrices A, B such that $f(x,u)$ equals $Ax + Bu$.

Observe that if $(x,u) = (0,0)$, then $\dot{x} = f(0,0) = 0$ which means that $x(t) = (x_1(t), x_2(t))$ is constant for all t. Physically, when the pendulum is in the downward position ($x_1 = 0$) with no velocity ($x_2 = 0$) and no applied torque ($u = 0$), then the pendulum stays in the downward position for all time. For this reason the configuration $(x_1, x_2) = (0,0)$, $u = 0$ is called an equilibrium configuration. ▲

Generalizing the intuition developed in the previous example we have the following definition.

Definition 5.2.1. Consider a system in state-variable form (5.3). A constant pair $(\overline{x}, \overline{u})$ is called an **equilibrium configuration of** (5.3) if $f(\overline{x}, \overline{u}) = (0, 0, \ldots, 0)$. The constant $\overline{x}$ is called an **equilibrium point**.

Many systems can be linearized about an equilibrium configuration. In other words, although almost every physical system contains nonlinearities, oftentimes its behaviour *within a certain operating range of an equilibrium configuration* can be reasonably approximated by that of a linear model. The linear approximation serves as the basis for control design. One reason for approximating the nonlinear system (5.3) by a linear model of the form (5.4) is that, by doing so, one can apply rather simple and systematic linear control design techniques. Keep in mind, however, that a linearized model will only be good a approximation to the nonlinear model when the system operates in a sufficiently small range around an equilibrium point. In this section we review how to make these approximations.

Example 5.2.2. (Equilibria of the Physical Pendulum) Back to the pendulum example. Suppose we turn off the motor and apply no torque, i.e., we set $u = \overline{u} = 0$. Let's use the definition above to find all corresponding equilibria. To do this we must solve the equation $f(\overline{x}, \overline{u}) = 0$ for $\overline{x}$. In other words, we must solve

$$\begin{aligned} \overline{x}_2 &= 0 \\ -1.5\frac{g}{\ell}\sin(\overline{x}_1) &= 0 \end{aligned}$$

for $\overline{x}_1$, $\overline{x}_2$. Thus the equilibria of the pendulum with $\overline{u} = 0$ are given by

$$\overline{x} = \begin{bmatrix} k\pi \\ 0 \end{bmatrix}, \quad k \text{ is an integer.}$$

Physically, this means that the pendulum is at equilibrium whenever the angle θ is either 0 (pointed downward) or π (pointed upward), and the angular velocity $\dot{\theta}$ is zero.

Now suppose that we turn on the motor in such a way that it produces the constant torque $u = \overline{u} \neq 0$. The corresponding equilibria satisfy $f(\overline{x}, \overline{u}) = 0$, i.e.,

$$\begin{aligned} \overline{x}_2 &= 0 \\ -1.5\frac{g}{\ell}\sin(\overline{x}_1) + \frac{3}{M\ell^2}\overline{u} &= 0. \end{aligned}$$

This yields the equilibria

$$\overline{x} = \begin{bmatrix} \arcsin\left(\frac{2\overline{u}}{gM\ell}\right) + 2\pi k \\ 0 \end{bmatrix}, \qquad k \text{ is an integer.}$$

If we pick a position $\overline{x}_1 \in (-\pi, \pi]$ and set $u = \overline{u} = Mg\ell\sin(\overline{x}_1)$, then the state

$$\overline{x} = \begin{bmatrix} \overline{x}_1 \\ 0 \end{bmatrix}$$

is an equilibrium point of the pendulum. Physically, this means that by imparting a suitable constant torque to the pendulum one can make the pendulum be at rest at any desired angle $\overline{x}_1$. For instance, by imparting the torque $u = \overline{u} = 0.5Mg\ell$, the configuration $\overline{x}_1 = \pi/2$, $\overline{x}_2 = 0$ is an equilibrium of the pendulum. ▲

Exercise 5.5. Consider the ball and beam system from Example 5.1.9. Find all the equilibria of the system when the applied voltage is zero, i.e., when $u = \overline{u} = 0$.

We now describe how to linearize a nonlinear state model. First, **assume** there is an equilibrium configuration, that is, a constant solution $x(t) = \overline{x}, u(t) = \overline{u}$. This is equivalent to saying that $0 = f(\overline{x}, \overline{u})$, i.e., $(\overline{x}, \overline{u})$ is an equilibrium configuration. Now consider the perturbations from the equilibrium configuration:

$$\delta x(t) \coloneqq x(t) - \overline{x}, \qquad \delta u(t) \coloneqq u(t) - \overline{u}, \qquad \delta x(t), \delta u(t) \text{ small.}$$

Using the Taylor series expansion around $(\overline{x}, \overline{u})$ we have

$$\begin{aligned} \dot{x}(t) &= f(x(t), u(t)) \\ &= f(\overline{x}, \overline{u}) + A\delta x(t) + B\delta u(t) + \text{higher order terms} \end{aligned}$$

where

$$A \coloneqq \left.\frac{\partial f}{\partial x}\right|_{(\overline{x},\overline{u})}, \qquad B \coloneqq \left.\frac{\partial f}{\partial u}\right|_{(\overline{x},\overline{u})}.$$

Since $f(\overline{x}, \overline{u}) = 0$, we have the linearized equation to be

$$\dot{\delta x} = A\delta x + B\delta u.$$

Similarly, the output equation $y = h(x, u)$ linearizes to

$$\delta y = C\delta x + D\delta u,$$

where

$$C \coloneqq \left.\frac{\partial h}{\partial x}\right|_{(\overline{x},\overline{u})}, \quad D \coloneqq \left.\frac{\partial h}{\partial u}\right|_{(\overline{x},\overline{u})}.$$

Summary

Linearizing $\dot{x} = f(x,u),\ y = h(x,u)$: Select, if one exists, an equilibrium configuration $(\overline{x},\overline{u})$. Compute the four Jacobians, A, B, C, D, of f and h at the equilibrium point. Then the linearized system is

$$\begin{aligned}\dot{\delta x} &= A\delta x + B\delta u,\\ \delta y &= C\delta x + D\delta u.\end{aligned}$$

Under mild conditions (sufficient smoothness of f and h), this linearized system is a valid approximation of the nonlinear one in a sufficiently small neighbourhood of the equilibrium configuration.

Example 5.2.3. (Physical Pendulum) Suppose that we want to control the physical pendulum

$$\begin{aligned}\dot{x}_1 &= x_2\\ \dot{x}_2 &= -1.5\frac{g}{\ell}\sin(x_1) + \frac{3}{M\ell^2}u\\ y &= x_1\end{aligned}$$

near its upright position, i.e., $\bar{y} = \pi$. In this example we compute a linear model that approximates the pendulum near this position. First we find the equilibrium configuration $(\overline{x},\overline{u})$ corresponding to the desired output $\bar{y}$. Using the results of Example 5.2.2 we get the equilibrium configuration (verify!)

$$(\overline{x},\overline{u}) = \left(\begin{bmatrix}\pi\\0\end{bmatrix}, 0\right).$$

To get the linearization at $(\overline{x},\overline{u})$ we have to compute the four matrices A, B, C, D:

$$\begin{aligned}A &= \left.\frac{\partial f}{\partial x}\right|_{(\overline{x},\overline{u})} = \begin{bmatrix}0 & 1\\ 1.5\frac{g}{\ell} & 0\end{bmatrix}, \quad B = \left.\frac{\partial f}{\partial u}\right|_{(\overline{x},\overline{u})} = \begin{bmatrix}0\\ \frac{3}{M\ell^2}\end{bmatrix}\\ C &= \left.\frac{\partial h}{\partial x}\right|_{(\overline{x},\overline{u})} = \begin{bmatrix}1 & 0\end{bmatrix}, \qquad D = \left.\frac{\partial h}{\partial u}\right|_{(\overline{x},\overline{u})} = 0.\end{aligned}$$

The linearized model is

$$\begin{aligned}\dot{\delta x} &= \begin{bmatrix}0 & 1\\ 1.5\frac{g}{\ell} & 0\end{bmatrix}\delta x + \begin{bmatrix}0\\ \frac{3}{M\ell^2}\end{bmatrix}\delta u\\ \delta y &= \begin{bmatrix}1 & 0\end{bmatrix}\delta x\end{aligned}$$

where $\delta x = x - \overline{x}$, $\delta u = u - \overline{u}$ and $\delta y = y - \overline{y}$ measure how far the pendulum is from equilibrium. ▲

Example 5.2.4. (Magnetic-Ball Suspension System) Suppose that we want to control the magnetic-ball suspension system

$$\begin{aligned}\dot{x}_1 &= -30x_1 + 2u\\ \dot{x}_2 &= x_3\\ \dot{x}_3 &= 9.8 - 0.001\frac{x_1^2}{x_2^2}\\ y &= x_2.\end{aligned}$$

with the ball near the position $\overline{y} = 0.01$m. Let's compute a linear model that approximates this system near this position. First we find the equilibrium configuration $(\overline{x},\overline{u})$ corresponding to the desired output $\overline{y}$. To find $(\overline{x},\overline{u})$ we solve $f(\overline{x},\overline{u}) = 0$, $h(\overline{x},\overline{y}) = \overline{y}$, i.e.,

$$\begin{aligned}0 &= -30\overline{x}_1 + 2\overline{u}\\ 0 &= \overline{x}_3\\ 0 &= 9.8 - 0.001\frac{\overline{x}_1^2}{\overline{x}_2 2}\\ 0.01 &= \overline{x}_2.\end{aligned}$$

Thus

$$\overline{x} = \begin{bmatrix} 0.99 \\ 0.01 \\ 0 \end{bmatrix}, \qquad \overline{u} = 14.85.$$

To get the linearization at $(\overline{x}, \overline{u})$ we have to compute the four matrices A, B, C, D:

$$A = \left.\frac{\partial f}{\partial x}\right|_{(\overline{x},\overline{u})} = \begin{bmatrix} -30 & 0 & 0 \\ 0 & 0 & 1 \\ -0.002\frac{x_1}{x_2^2} & 0.002\frac{x_1^2}{x_2^3} & 0 \end{bmatrix}_{(\overline{x},\overline{u})} = \begin{bmatrix} -30 & 0 & 0 \\ 0 & 0 & 1 \\ -19.8 & 1940 & 0 \end{bmatrix}, \qquad B = \left.\frac{\partial f}{\partial u}\right|_{(\overline{x},\overline{u})} = \begin{bmatrix} 2 \\ 0 \\ 0 \end{bmatrix},$$

$$C = \left.\frac{\partial h}{\partial x}\right|_{(\overline{x},\overline{u})} = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix}, \qquad D = \left.\frac{\partial h}{\partial u}\right|_{(\overline{x},\overline{u})} = 0.$$

The linearized model is

$$\begin{aligned} \delta\dot{x} &= A\delta x + B\delta u \\ \delta y &= C\delta x + D\delta u, \end{aligned}$$

where $\delta x = x - \overline{x}$, $\delta u = u - \overline{u}$ and $\delta y = y - \overline{y}$ measure how far the system is from equilibrium. ▲

5.2.1 Obtaining a transfer function from a linear state-space model

Let's find the transfer function for the LTI state-space model

$$\begin{aligned} \dot{x} &= Ax + Bu \\ y &= Cx + Du. \end{aligned}$$

Take Laplace transforms with zero initial conditions[2]

$$\begin{aligned} sX(s) &= AX(s) + BU(s) \\ Y(s) &= CX(s) + DU(s). \end{aligned}$$

Eliminate $X(s)$

$$\begin{aligned} &(sI - A)X(s) = BU(s) \\ &\Rightarrow X(s) = (sI - A)^{-1}BU(s) \\ &\Rightarrow Y(s) = \underbrace{\left(C(sI - A)^{-1}B + D\right)}_{\text{transfer function}} U(s) =: G(s)U(s). \end{aligned}$$

A convenient notation for the transfer function of (A, B, C, D) is

$$\left[\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right](s) \coloneqq C\left(sI - A\right)^{-1}B + D. \tag{5.5}$$

Remark 5.2.2. The transfer matrix of a 4-tuple (A, B, C, D) of real matrices is always real rational and proper, i.e., each entry $g_{ij}(s)$ of $G(s)$ has the form

$$\frac{b_n s^n + b_{n-1}s^{n-1} + \cdots + b_1 s + b_0}{s^n + a_{n-1}s^{n-1} + \cdots + a_1 s + a_0}, \qquad a_i, b_i \in \mathbb{R}.$$

To understand why, express the inverse $(sI - A)^{-1}$ using Cramer's rule as

$$(sI - A)^{-1} = \frac{1}{\det(sI - A)} \operatorname{adj}(sI - A)$$

[2] Here $X(s)$ denotes the component wise LT of the vector $x(t)$. That is, if $x(t) = (x_1(t), \ldots, x_n(t))$ then $X(s) = (X_1(s), \ldots, X_n(s))$.

where $\det(\cdot)$ is the determinant of a square matrix and $\operatorname{adj}(\cdot)$ is the adjugate (classical adjoint) of a square matrix.

The determinant of the $n \times n$ matrix $sI - A$ is an nth degree polynomial in s with real coefficients. The adjoint of $sI - A$ is an $n \times n$ matrix. Its (i,j)th entry equals $(-1)^{i+j}$ multiplied by the determinant of the $(n-1) \times (n-1)$ submatrix obtained from $sI - A$ by eliminating the jth row and the ith column. This means that each entry in $\operatorname{adj}(sI - A)$ is a polynomial of degree *strictly less than* n with real coefficients.

Combining these observations we deduce that each entry in $(sI - A)^{-1}$ is a *strictly proper*, real rational function of s. Thus we conclude that each entry in $C(sI - A)^{-1}B$ is a linear combination of strictly proper functions and is therefore strictly proper. Finally, if D is non-zero, then $C(sI - A)^{-1}B + D$ is proper. ⧫

This construction naturally leads to the converse question known as the **realization problem**: Given $G(s)$, find A, B, C, D such that

$$G(s) = C(sI - A)^{-1}B + D.$$

A solution exists if, and only if, $G(s)$ is rational and proper. The solution is never unique.

Summary

Let us recap our procedure for getting the transfer function of a system:

1. Apply first principles to get differential equations governing the behaviour of the system. Put these equations in state form. In general these are nonlinear.
2. Find an equilibrium, if there is one. If there is more than one equilibrium, you have to select one. If there isn't even one, this method doesn't apply.
3. Linearize about the equilibrium point to get an LTI state model.
4. Take the Laplace transform of the linearized state model with zero initial state.
5. Solve for the output $Y(s)$ in terms of the input $U(s)$.

The transfer function from input to output satisfies

$$Y(s) = G(s)U(s).$$

In general $G(s)$ is a matrix. In the SISO case, $G(s)$ is a scalar-valued transfer function.

Example 5.2.5. (Physical Pendulum) In Example 5.2.1 we found the nonlinear state model of a physical pendulum (step 1). In Example 5.2.3 we found the pendulum's equilibrium configuration corresponding to being upright (step 2) and we also linearized about the equilibrium configuration (step 3) to obtain an LTI state model

$$\begin{aligned}\dot{\delta x} &= \begin{bmatrix} 0 & 1 \\ 3\frac{g}{\ell} & 0 \end{bmatrix} \delta x + \begin{bmatrix} 0 \\ \frac{3}{M\ell^2} \end{bmatrix} \delta u \\ \delta y &= \begin{bmatrix} 1 & 0 \end{bmatrix} \delta x.\end{aligned}$$

We now find the system's transfer function (step 4, step 5) as

$$\begin{aligned}G(s) &= C(sI - A)^{-1}B + D \\ &= \frac{1}{\det(sI - A)} C \operatorname{adj}(sI - A) B + D \\ &= \frac{1}{s^2 - 3\frac{g}{\ell}} \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} s & 1 \\ 3\frac{g}{\ell} & s \end{bmatrix} \begin{bmatrix} 0 \\ \frac{3}{M\ell^2} \end{bmatrix} \\ &= \frac{3}{M\ell^2} \frac{1}{s^2 - 3\frac{g}{\ell}}.\end{aligned}$$

This system is unstable as you'd intuitively expect for a pendulum at its upright position. ▲

5.2.2 Control design using linearization

The linearized model is useful for doing control design. The idea is to use standard LTI design tools to design an LTI controller for the linearized system and then implement the controller on the nonlinear system. Let's see how this works in an example.

Example 5.2.6. (Mass-Spring-Damper) Suppose we want to control the mass-spring-damper system (cf. Example 2.2.3) shown in Figure 5.9 around the position $\overline{q} = 0.5$m.

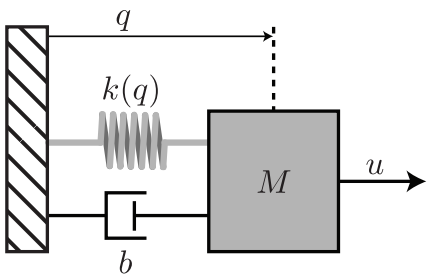


Figure 5.9: Mass-spring-damper system.

The spring here is nonlinear; it's a *softening spring*. For a relatively small displacement, the restoring force is linear. For a large displacement the restoring force is no longer linear. In this example we have

$$k(q) = K(1 - a^2q^2)q, \qquad |aq| < 1, \qquad K > 0.$$

It's called a softening spring because beyond a certain displacement, a large positional increment produces a small force increment.

Exercise 5.6. Use software to plot the restoring force of the softening spring versus q for $K = 2$, $a = 0.1$, $q \in (-10, 10)$.

The equation of motion for this system is

$$M\ddot{q} = u - b\dot{q} - k(q).$$

Since we are interested in controlling the mass around the position $\overline{q} = 0.5$m we take the position q as the output of the system. We choose $x = (x_1, x_2) \coloneqq (q, \dot{q})$ as the state vector and get the state-space model

$$\begin{aligned}
\dot{x}_1 &= x_2 \\
\dot{x}_2 &= \frac{1}{M}\Big(- K\left(1 - a^2x_1^2\right)x_1 - bx_2 + u\Big) \\
y &= x_1.
\end{aligned}$$

Next we find the equilibrium configuration $(\overline{x}, \overline{u})$ corresponding to the desired output $\overline{y} = 0.5$. As we've seen, this is found by solving

$$\begin{aligned}
0 &= \overline{x}_2 \\
0 &= \frac{1}{M}\left(-K\left(1 - a^2\overline{x}_1^2\right)\overline{x}_1 - b\overline{x}_2 + \overline{u}\right) \\
0.5 &= \overline{x}_1.
\end{aligned}$$

The solution is (verify!)

$$\begin{aligned}
\overline{x}_1 &= 0.5 \\
\overline{x}_2 &= 0 \\
\overline{u} &= \frac{K}{2}\left(1 - 0.25a^2\right).
\end{aligned}$$

Now we compute Jacobians and evaluate them at $(\overline{x}, \overline{u})$. The matrices A, B, C, D are (verify!)

$$A = \left.\frac{\partial f}{\partial x}\right|_{(\overline{x},\overline{u})} = \begin{bmatrix} 0 & 1 \\ -\frac{K}{M}\left(1 - 3a^2x_1^2\right) & -\frac{b}{M} \end{bmatrix}_{(\overline{x},\overline{u})} = \begin{bmatrix} 0 & 1 \\ -\frac{K}{M}\left(1 - 0.75a^2\right) & -\frac{b}{M} \end{bmatrix}, \qquad B = \left.\frac{\partial f}{\partial u}\right|_{(\overline{x},\overline{u})} = \begin{bmatrix} 0 \\ \frac{1}{M} \end{bmatrix},$$
$$C = \left.\frac{\partial h}{\partial x}\right|_{(\overline{x},\overline{u})} = \begin{bmatrix} 1 & 0 \end{bmatrix}, \qquad D = \left.\frac{\partial h}{\partial u}\right|_{(\overline{x},\overline{u})} = 0.$$

The linearized system is $\delta\dot{x} = A\delta x + B\delta u$, $\delta y = C\delta x + D\delta u$ where $\delta x = x - \overline{x}$, $\delta u = u - \overline{u}$ and $\delta y = y - \overline{y}$ represent the deviation from the equilibrium configuration. The transfer function of this system from the input $\Delta U \coloneqq \mathscr{L}\{\delta u\}$ to the output $\Delta Y \coloneqq \mathscr{L}\{\delta y\}$ is

$$\begin{aligned} \frac{\Delta Y}{\Delta U} = P(s) &= C(sI - A)^{-1}B \\ &= \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} s & -1 \\ \frac{K}{M}\left(1 - 0.75a^2\right) & s + \frac{b}{M} \end{bmatrix}^{-1} \begin{bmatrix} 0 \\ \frac{1}{M} \end{bmatrix} \\ &= \frac{1}{s^2 + \frac{b}{M}s + \frac{K}{M}\left(1 - 0.75a^2\right)} \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} s + \frac{b}{M} & 1 \\ -\frac{K}{M}\left(1 - 0.75a^2\right) & s \end{bmatrix} \begin{bmatrix} 0 \\ \frac{1}{M} \end{bmatrix} \\ &= \frac{1/M}{s^2 + \frac{b}{M}s + \frac{K}{M}\left(1 - 0.75a^2\right)}. \end{aligned}$$

We now have a TF to work with for control design. Suppose that we've identified that $M = K = b = 1$ and $a^2 = 1/3$. Then

$$\frac{\Delta Y}{\Delta U} = \frac{1}{s^2 + s + 0.75}.$$

We'll design a Proportional-Integral (P.I.) controller to stabilize the plant and make the mass approach the desired position of 0.5. You can verify that the PI controller

$$C(s) = K_p + \frac{K_i}{s}, \qquad K_p = 1, \qquad K_i = 0.5$$

gives perfect steady state tracking for steps and yields a stable closed loop system. It may not be the best controller because the complex poles of the closed-loop system are lightly damped (i.e. close to the imaginary axis) and, moreover, the pole at the origin may make the settling time a little long. However, for the sake of this example, let's assume that the step response is satisfactory and proceed.

We have designed the control law based on the linearized system but at the end of the day we have to implement the control law on the nonlinear plant. In order to account for this we must introduce the following bias terms illustrated in Figure 5.10.

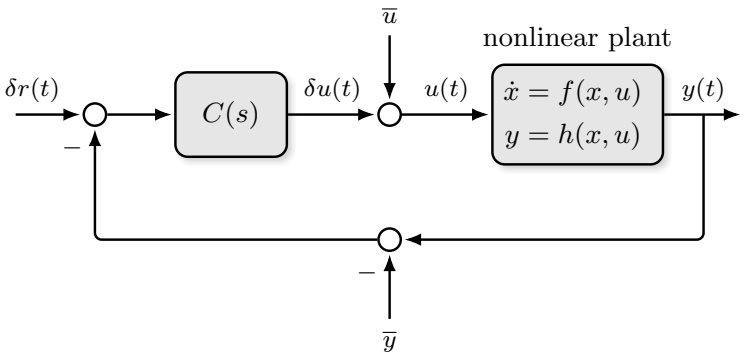


Figure 5.10: Overall control structure when designing using linearization.

If the controller we designed works well for the linearized plant, it will work well for the nonlinear system so long as the nonlinear system's state remains close to the equilibrium configuration. ▲

Exercise 5.7. Simulate the system in Figure 5.10 using the controller and plant from Example 5.2.6.

5.3 Inverting static nonlinearities

A nonlinearity is called **static** or **memoryless** if it has no dynamics. This means that the output at any time t only depends on the input at time t.

Example 5.3.1. (Static versus Non-Static Nonlinearities) Consider a single-input single-output nonlinear system with input u and output v shown in Figure 5.11. If Φ models quantization, then it has the graph shown

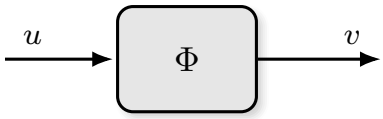


Figure 5.11: A nonlinear system.

in Figure 5.12a. This is a static nonlinearity because the output v at time t only depends on the input u at time t. Instead, if the block Φ models hysteresis, then it has the graph shown in Figure 5.12b. This isn't a

(a) Quantization. *(b) Relay with hysteresis.*

Figure 5.12: Static versus non-static nonlinearities.

static nonlinearity because the output v at time t depends $u(t)$ as well as previous values of u. ▲

It is sometimes possible to perfectly cancel the effect of a static nonlinearity by properly choosing the control input. Consider the plants shown in Figure 5.13 where Φ is a static nonlinearity. Figure 5.13a is a general

(a) Static nonlinearity at the input. *(b) Static nonlinearity at the output.*

Figure 5.13: Static nonlinearities in series with a state model.

block diagram of a control system in state-space form with a static nonlinearity at its input. In Figure 5.13b the control system has a static nonlinearity at its output.

First consider the system in Figure 5.13a. Suppose that there exists a function Ψ such that

$$\text{for any } w \in \mathbb{R} \qquad \Phi(\Psi(w)) = w.$$

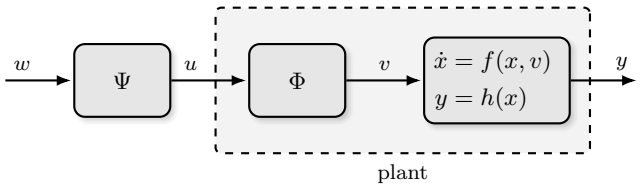


Figure 5.14: Inverting an input static nonlinearity.

If we choose the preliminary control signal $u = \Psi(w)$, then the block diagram from Figure 5.13a is modified to become the diagram in Figure 5.14. Since $\Phi(\Psi(w)) = w$ we get that $w(t) = v(t)$ and we have effectively "cancelled" the effect of the nonlinearity Φ. The relationship between the "new" input signal w and the output y is

$$\dot{x} = f(x, w)$$
$$y = h(x).$$

If f and h happen to be linear functions, i.e., $f(x, u) = Ax + Bu$, $h(x) = Cx$, then the relationship between w and y is linear. Taking Laplace transforms (see Section 5.2.1) we get

$$Y(s) = G(s)W(s), \qquad G(s) = C(sI - A)^{-1}B.$$

In this case we are free to use all our LTI design tools to control the plant. The next example illustrates the usefulness of inverting static nonlinearities at the input to an otherwise linear plant.

Example 5.3.2. (Control Design Using Nonlinear Inversion) Consider the plant in Figure 5.15. Here

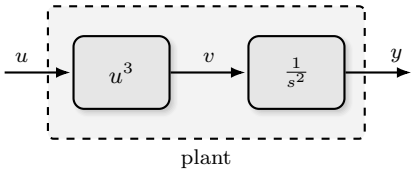


Figure 5.15: LTI system with a static nonlinearity at its input.

$\Phi(u) = u^3$. If we apply the preliminary control $u = \Psi(w) = \sqrt[3]{w}$ then input-output map from $W(s) \coloneqq \mathscr{L}\{w\}$ to $Y(s) \coloneqq \mathscr{L}\{y\}$ is

$$\frac{Y(s)}{W(s)} = \frac{1}{s^2}.$$

The closed-loop system with this preliminary control applied is shown in Figure 5.16. Suppose that our speci-

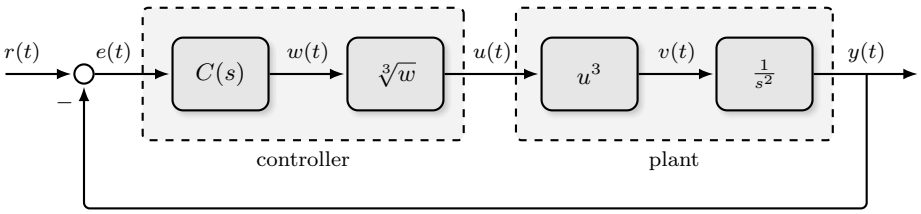


Figure 5.16: The plant from Figure 5.15 in unity feedback configuration with cancellation of static input nonlinearity.

fications for the closed-loop system are

1. Closed-loop bandwidth of 10 rad/s.
2. Phase margin of at least 20°.

Since we've inverted the static nonlinearity we are free to use our LTI design tools to come up with $C(s)$. To meet the specifications we'll design a lead controller (see Appendix 2.B)

$$C(s) = K\frac{\alpha Ts+1}{Ts+1}, \qquad \alpha > 1, T > 0, K > 0.$$

To meet the bandwidth specification, define $\hat{K} = K\sqrt{\alpha}$ and pick $\hat{K}$ so that the gain crossover frequency of $\hat{K}P(j\omega)$ is 10 rad/s. This gives $\hat{K} = 100$.

To meet the phase margin specification we need to add at least $\phi_{\max} = 20°$ of phase at $\omega_m = 10$rad/s. Using the lead controller design equation (2.21) we get

$$\alpha = \frac{1+\sin(\phi_{\max})}{1-\sin(\phi_{\max})} = 2.0396$$

and so $K = \hat{K}/\sqrt{\alpha} = 70$. To get this phase addition at the desired frequency we use (2.19)

$$\omega_m = \frac{1}{T\sqrt{\alpha}} \Rightarrow T = \frac{1}{\omega_m\sqrt{\alpha}} = 0.07.$$

Our overall controller is

$$C(s) = 70\frac{0.1428s+1}{0.07s+1} = 142.8\frac{(s+7)}{s+14.28}.$$

The Bode plot of $C(j\omega)P(j\omega)$ is shown in Figure 5.17. From the figure we see that the phase margin is 20° and the gain crossover frequency is 10 rad/s as desired. ▲

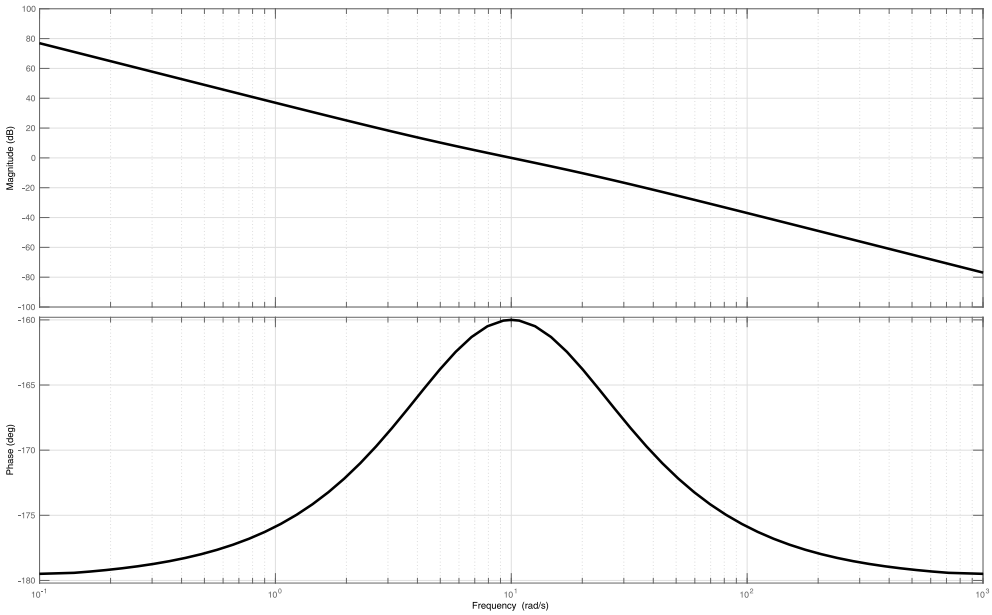


Figure 5.17: Bode plot of $C(j\omega)P(j\omega)$ from Example 5.3.2.

Exercise 5.8. Simulate the closed-loop system from Example 5.3.2.

Remark 5.3.1. Since our controllers are implemented on a computer a more detailed model of the interface between the digital controller and the plant in Example 5.3.2 is shown in Figure 5.18. If the D/A simply holds the value of $u[k]$ over one period T, it should be clear that $v(t) = w[k]$ for $kT \le t < (k+1)T$. This is precisely because the nonlinearity we are cancelling is static. Therefore the presence of the D/A does not cause any implementation issues in this inversion. ♦

Figure 5.18: Detailed model of the interface between controller and plant from Example 5.3.2.

Next we turn to output nonlinearities as in Figure 5.13b. Suppose that there exists a function Ψ such that

$$\text{for any } v \in \mathbb{R} \qquad \Psi(\Phi(v)) = v.$$

Further suppose that instead of using the signal y for feedback, we use the signal $\Psi(y)$. Then the block diagram of 5.13b is modified to become the diagram in Figure 5.19.

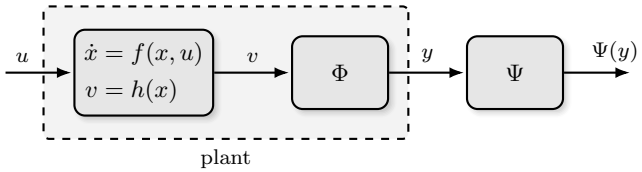


Figure 5.19: Inverting an output static nonlinearity.

Since $\Psi(\Phi(v)) = v$ we have effectively "cancelled" the effect of the nonlinearity Φ. The relationship between the input u and the signal v is simply $\dot{x} = f(x, u)$, $v = h(x)$. Once again this is particularly useful when f and h are linear functions, i.e., $f(x, u) = Ax + Bu$, $h(x) = Cx$. When f and h are linear we get a linear relationship between u and v. Taking Laplace transforms (see Section 5.2.1), we get the TF

$$V(s) = G(s)U(s), \qquad G(s) = C(sI - A)^{-1}B$$

and we are once again free to use all our LTI design tools to control the plant. We illustrate these ideas in an example.

Example 5.3.3. (Nonlinear Sensor) Consider a DC motor (cf. Example 2.2.5) where the angular position $\theta(t)$ of the motor's shaft is measured by a nonlinear sensor. The setup is illustrated in Figure 5.20. The output

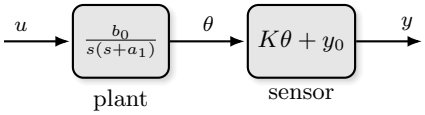


Figure 5.20: DC motor with a nonlinear position sensor.

y of the sensor is a voltage that depends on the angular displacement of the DC motor's shaft. To be specific, suppose that

$$\Phi(\theta) = K\theta + y_0, \qquad K, y_0 \in \mathbb{R},$$

i.e., the measured voltage y is a linear affine function of the shaft angle θ. If we define

$$\Psi(y) \coloneqq \frac{1}{K}(y - y_0)$$

then $\Psi(\Phi(\theta)) = \theta$. If we use the signal $\Psi(y)$ for feedback instead of y, then we have a linear system from the input signal u to the signal $\Psi(y)$. The implementation details are illustrated in Figure 5.21. We can now use our linear control design techniques in order to design the block $C(s)$. ▲

Remark 5.3.2. Since our controllers are implemented on a computer a more detailed model of the interface between the sensor and the controller from Example 5.3.3 is shown in Figure 5.22. It should be clear that $\theta[k]$ only equals $\theta(t)$ at the sampling instances and not between samples where changes in $\theta(t)$ are not reflected in $\theta[k]$. This does not cause any implementation issues because our control action only depends on the sampled signal $\theta[k]$. ♦

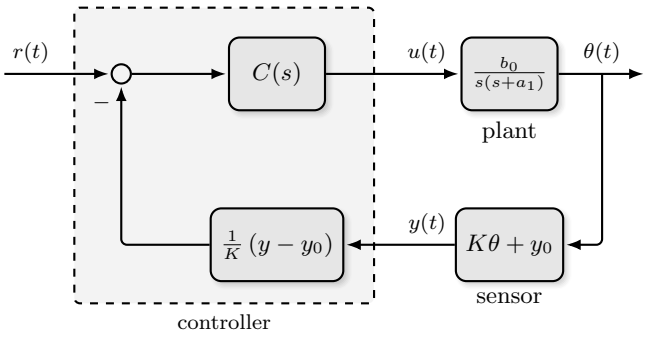


Figure 5.21: Controlling a DC motor with a nonlinear position sensor.

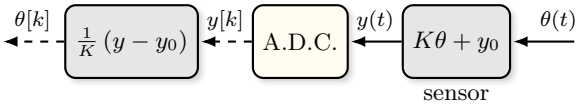


Figure 5.22: Detailed model of the interface between sensor and controller from Example 5.3.3.

Generalizing these examples we would like to understand when such cancellations are possible. The two key properties we exploited were:

1. The nonlinearity $\Phi : \mathbb{R} \to \mathbb{R}$ is static and appears at either the input to the system or at the output of the system.

2. If the nonlinearity Φ is at the input, then it has a right inverse (see definitions below). If the nonlinearity Φ is at the output, then it has a left inverse (see definitions below).

The first condition depends on the location of the nonlinearity. In terms of state-space models, the system must have one of the following two forms (cf. Figure 5.13)

$$\begin{aligned} \dot{x} &= f(x, v) & & & \dot{x} &= f(x, u) \\ y &= h(x) & & & v &= h(x) \\ v &= \Psi(u) & \text{(static nonlinearity } \Psi \text{ at input)}, & & y &= \Psi(v) & \text{(static nonlinearity } \Psi \text{ at output)}. \end{aligned}$$

The second condition depends on whether or not the function Ψ is onto (in the case of input nonlinearities) or one-to-one (in the case of output nonlinearities).

Definition 5.3.3. Let $\Phi : \mathbb{R} \to \mathbb{R}$ be a real-valued function of a real variable.

(i) The function Φ is **one-to-one** if $y_1 \neq y_2$ implies $\Phi(y_1) \neq \Phi(y_2)$.

(ii) The function Φ is **onto** if for any $v \in \mathbb{R}$ there exists a $u \in \mathbb{R}$ such that $v = \Phi(u)$.

Example 5.3.4. The nonlinear function $\Phi(u) = u^3$ is both onto and one-to-one (draw its graph to convince yourself). On the other hand the static nonlinearity $\Phi(u) = |u|$ is neither onto nor one-to-one. The nonlinear function $\arctan(u)$ is one-to-one but it is not onto. Finally, the function $u \sin(u)$ is onto but not one-to-one. ▲

The properties of being onto and one-to-one tell us whether or not a function is invertible.

Definition 5.3.4. Let $\Phi : \mathbb{R} \to \mathbb{R}$ be a real-valued function of a real variable.

(i) A function Ψ is a **right inverse** of Φ if, for all $w \in \mathbb{R}$, $\Phi(\Psi(w)) = w$.

(ii) A function Ψ is a **left inverse** of Φ if, for all $y \in \mathbb{R}$, $\Psi(\Phi(y)) = y$.

Proposition 5.3.5 ([MacLane and Birkhoff, 1999, Theorem I.1])**.** *A function $\Phi : \mathbb{R} \to \mathbb{R}$ has a right inverse if, and only if, Φ is onto. A function $\Phi : \mathbb{R} \to \mathbb{R}$ has a left inverse if, and only if, Φ is one-to-one.*

Example 5.3.5. The function $\Phi(u) = u^3$ is one-to-one and onto. Therefore, by Proposition 5.3.5 it has both a left and a right inverse. They are equal to $\Psi(w) = \sqrt[3]{w}$.

On the other hand consider the static nonlinearity $\Phi(u) = \arctan(u)$. This function is one-to-one but not onto. By Proposition 5.3.5 it has a left inverse but no right inverse. A left inverse is $\Psi(v) = \tan(v)$. ▲

Remark 5.3.6. In practice it may not be possible to cancel static nonlinearities for a variety of reasons. First, the nonlinearity may not appear in the right place, i.e., neither at the input nor the output. Second the nonlinearity may not have the appropriate inverse. Third, if we do not have good knowledge of the function Φ, then we will end up making the wrong choice of Ψ. ⧫

Example 5.3.6. (Saturation Nonlinearity) Consider the saturation nonlinearity

$$\begin{aligned} &\text{sat} : \mathbb{R} \to \mathbb{R} \\ &u \mapsto \begin{cases} u & \text{if } u_{\min} \leq u \leq u_{\max}, \\ u_{\min} & \text{if } u < u_{\min}. \\ u_{\max} & \text{if } u > u_{\max} \end{cases} \end{aligned}$$

whose graph is shown in Figure 5.23.

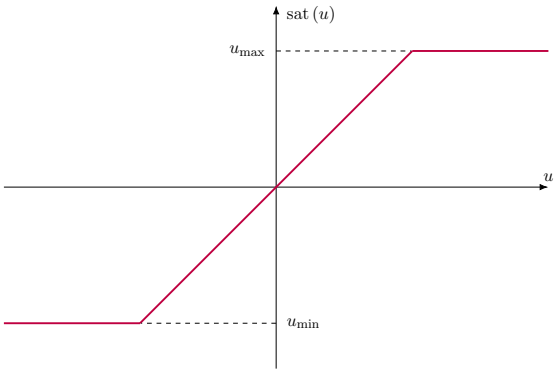


Figure 5.23: Saturator nonlinearity.

This is a common static nonlinearity in control systems. It's clearly not onto nor one-to-one and therefore it is neither right nor left invertible.

▲

Remark 5.3.7. For the purpose of the lab, the best approach to dealing with a saturator is to simply ignore it during the design stage. Then, ensure that for all practical operating conditions saturation is avoided. This approach usually requires lots of simulation and can be conservative. ⧫

The next example illustrates that even if a nonlinearity is not right invertible as per Definition 5.3.4, it may still have a *local right inverse*.

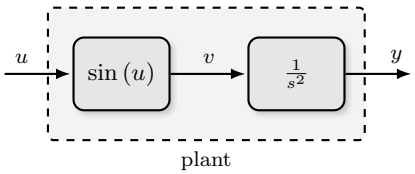


Figure 5.24: LTI system with a locally right invertible static nonlinearity at its input.

Example 5.3.7. (Local Cancellation of Static Nonlinearities) Consider the system in Figure 5.24. Here $\Phi(u) = \sin(u)$. This static nonlinearity is not onto and is therefore not right invertible. However, if we can guarantee that $u(t)$ only takes values between $-\pi/2$ and $\pi/2$, then Φ does have a right inverse. More precisely, if $\Psi(w) = \arcsin(w)$, then

$$\text{for all } -1 \leq w \leq 1 \qquad \Phi(\Psi(w)) = w.$$

Therefore, so long as we only allow our control signal $w(t)$ to take values between -1 and 1, we can effectively cancel the nonlinearity Φ and obtain a linear system from the input w to the output y. ▲

It is tempting to extend the inversion idea from this section to non-static nonlinearities. In doing so we have to be careful because our controllers are implemented on embedded computers as the next example shows.

Example 5.3.8. (Inverting Dynamic Nonlinearities) Consider the mass-spring-damper system from Example 5.2.6 with a softening spring. If we take the controller u to be

$$u = -Ka^2y^3 + w \tag{5.6}$$

then the dynamics from the new input w to the output y are modeled by

$$\begin{aligned}\dot{x}_1 &= x_2\\ \dot{x}_2 &= \frac{1}{M}\left(-Kx_1 - bx_2 + w\right)\\ y &= x_1.\end{aligned}$$

This is now a second order linear system because we've used the control u to cancel out the nonlinear term in the softening spring[3]. We use the technique from Section 5.2.1 to get the TF from $W(s) \coloneqq \mathscr{L}\{w\}$ to $Y(s) \coloneqq \mathscr{L}\{y\}$ as

$$\frac{Y(s)}{W(s)} = \frac{1/M}{s^2 + \frac{b}{M}s + \frac{K}{M}}.$$

Say that $M = b = a = 1$ and $K = 0.75$. Then the TF is the same as in Example 5.2.6 and we can use the same PI controller

$$C(s) = K_p + \frac{K_i}{s}, \qquad K_p = 1, \qquad K_i = 0.5.$$

Figure 5.25 shows the implementation of this controller. In this figure $D[z]$ represents the discretization of the continuous-time controller $C(s)$. The term $-Kay^3$ implements the nonlinear cancellation (5.6). The problem here is that the nonlinear cancellation is now approximate because the value of y is not constant between sample periods but our control signal is. Therefore, if the sampling period of the A/D and D/A blocks is large, the system will not behave like a linear system. Figure 5.26a shows the step response of the continuous-time system as well as the step response of the sampled-data system when the sampling rate is 0.1 seconds. In this case the two responses are similar.

Figure 5.26b shows the step response of the continuous-time system as well as the step response of the sampled-data system when the sampling rate is 0.5 seconds. In this case the two responses are very different and the sampled-data system is actually unstable. ▲

[3]This is an example of feedback linearization. We have obtained a linear system through the use of feedback. Feedback linearization is an important technique for nonlinear control design.

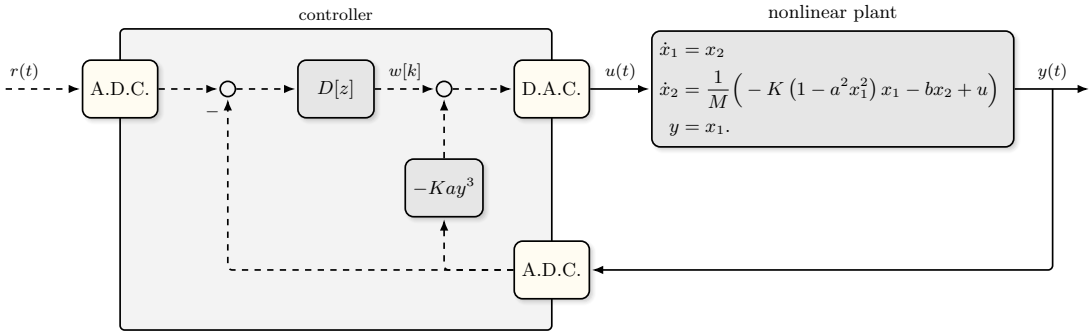
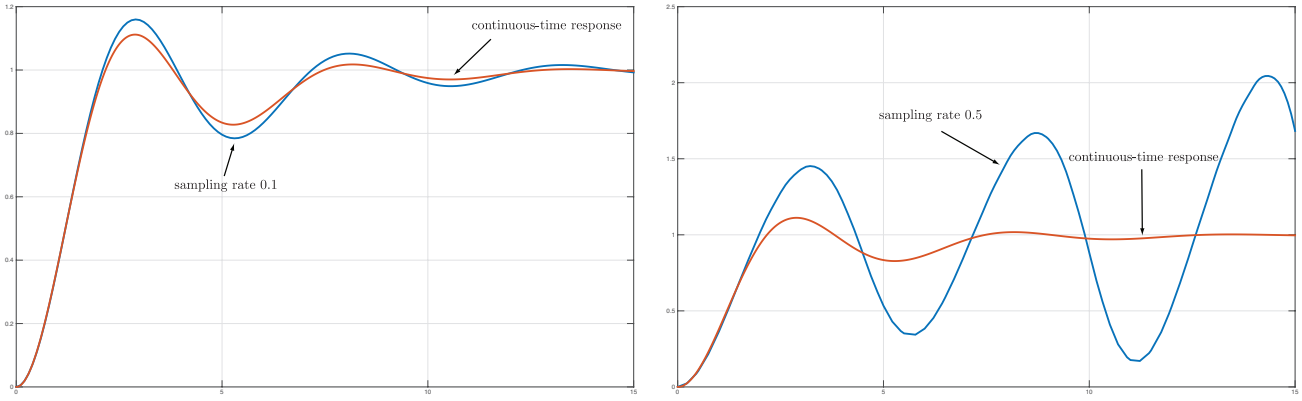


Figure 5.25: Implementing the controller from Example 5.3.8.



(a) Sampling period $T = 0.1$ seconds. *(b) Sampling period $T = 0.5$ seconds.*

Figure 5.26: Step responses from Example 5.3.8.

5.4 Static friction

Static friction can make control design challenging. In this section we give an elementary description of friction models and some simple strategies for dealing with static friction.

Example 5.4.1. (Friction Models) Consider the mass-damper system in Figure 5.27. There is a nonlinear

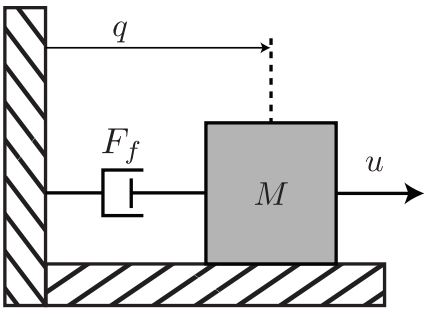


Figure 5.27: Mass-spring-damper system.

damper which models the force F_f due to friction and an applied force u. The system model is (cf. Example 2.2.3)

$$M\ddot{q} + F_f = u.$$

The friction force F_f can have components due to **static**, **Coulomb** and **viscous** friction.

When the mass M is at rest there is a static friction force F_s that acts parallel to the surface on which M sits. This force is limited to take values in the interval $[-\mu_s Mg, \mu_s Mg]$ where $0 < \mu_s < 1$ is the static friction

coefficient. This force takes whatever value, between its limits, to keep the mass at rest. For motion to begin, there must be a force acting on M to overcome the static friction. In the above model this means that, when the system is at rest, the force due to static friction can be viewed as a function of u and it has the graph shown in Figure 5.28.

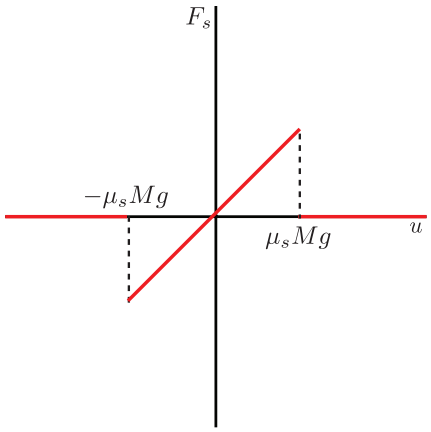


Figure 5.28: Model of force due to static friction.

Once the static friction has been overcome and motion has started, the friction force F_f is usually modelled as a function of the velocity $\dot{q}$. Coulomb friction F_c has a constant magnitude $\mu_k Mg$ where μ_k is the kinetic friction coefficient. The graph of F_c is shown in Figure 5.29.

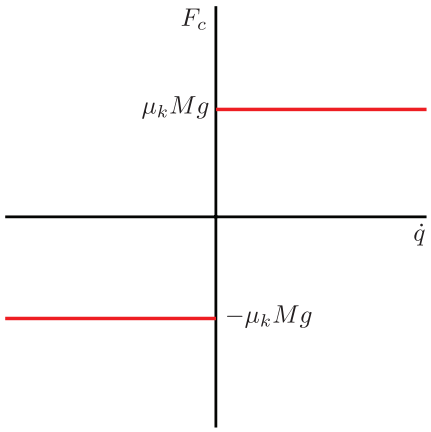


Figure 5.29: Model of force due to Coulomb friction.

As the mass moves in a viscous medium like air or lubricant, there is a frictional force F_v due to viscosity. For small velocities is it common to model this as a linear function of $\dot{q}$, i.e., $F_v(\dot{q}) = c\dot{q}$ for some $c > 0$. Figure 5.30 shows a friction model when both Coulomb and linear viscous friction are present. When static, Coulomb and viscous friction are present the dynamics of the system become nonlinear. These nonlinearities, in particular the static friction nonlinearity, make control difficult. ▲

Static friction is a particular concern when motors are involved. Motor static friction can (i) increase steady-state error (ii) induce oscillations when integral control is used and (iii) lead to instability for open-loop unstable plants.

Example 5.4.2. (DC Motor with Static Friction) Consider the control system in Figure 5.31 where $\Phi(u)$ models static friction. The reference input is a pulse train with amplitude 1, a period of 60 seconds and duty cycle of 50 percent.

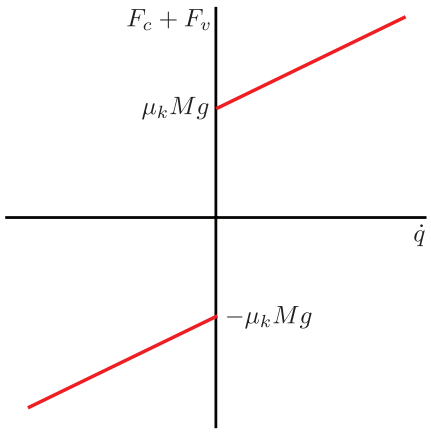


Figure 5.30: Model of force due to viscous and Coulomb friction.

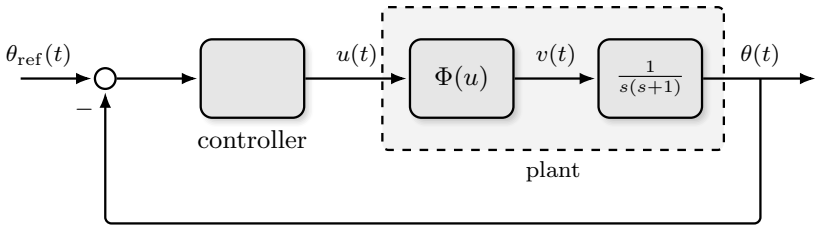
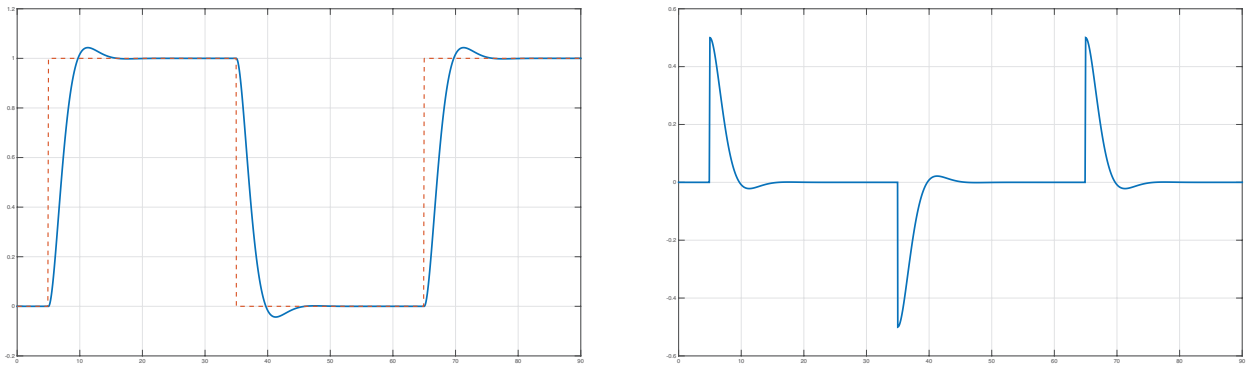


Figure 5.31: DC motor with static friction nonlinearity at its input.

First suppose that there is no static friction, i.e., $\Phi(u) = u$, and we choose the proportional controller $C(s) = 0.5$. Then the step response and control signal are shown in Figure 5.32. As expected, since the plant has an integrator, we get asymptotic tracking with zero steady-state error.



(a) $\theta(t)$ and $\theta_{\text{ref}}(t)$. *(b) Control signal $u(t)$.*

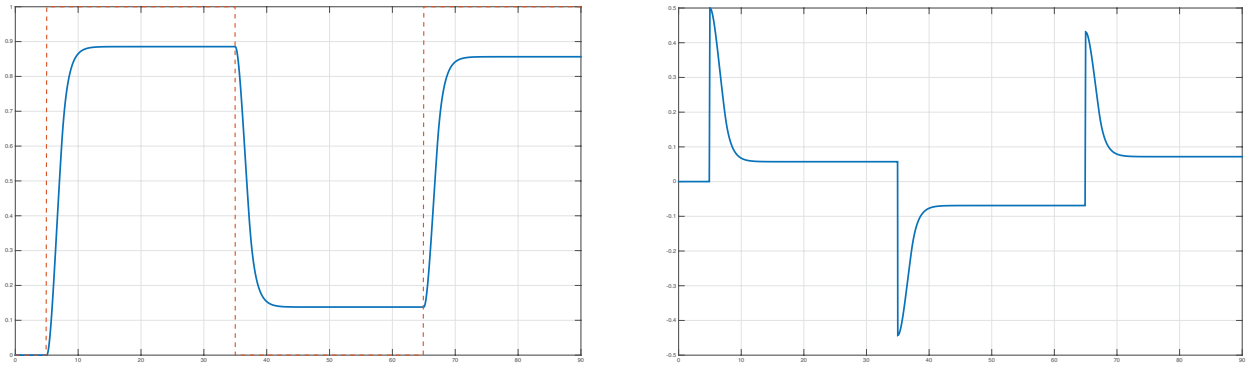
Figure 5.32: Pulse train response from Example 5.4.2 with no static friction and proportional control.

Now suppose that the static friction is crudely modeled as (cf. Example 5.4.1)

$$\Phi(u) = \begin{cases} u & |u| \geq 0.2 \\ 0 & |u| < 0.2. \end{cases} \tag{5.7}$$

Using the same proportional controller $C(s) = 0.5$ as before we get the response and control signals shown in Figure 5.33. The static friction has introduced steady-state tracking error and, as expected, worsened the closed-loop system's performance. ▲

We now present two approaches to dealing with static friction in control systems.



(a) $\theta(t)$ and $\theta_{\text{ref}}(t)$. *(b) Control signal $u(t)$.*

Figure 5.33: Pulse train response from Example 5.4.2 with static friction and proportional control.

5.4.1 Dither signals

A **dither** signal is a periodic signal with small amplitude that oscillates about zero. Dither signals can be used to overcome static friction by forcing all the parts of the system to "jitter," essentially keeping the plant constantly excited. The jitter signal is added to the control signal.

The main advantage of using dither signals is that it is very simple and often works. The main disadvantages are that always applying dither signals (i) wears out actuators (ii) excites unmodeled plant dynamics and (iii) often requires tuning because static friction can change from day-to-day.

Example 5.4.3. (Using Dither Signals to Overcome Static Friction) Returning to Example 5.4.2 we implement the dither signal $d(t)$ as shown in Figure 5.34.

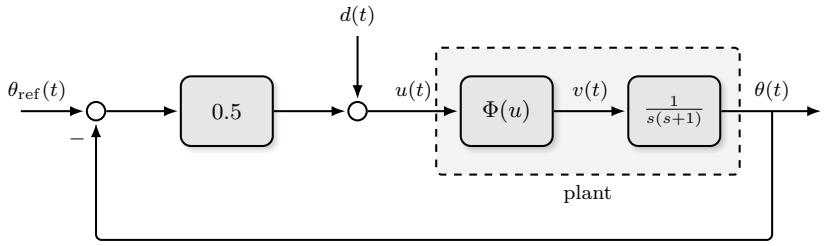


Figure 5.34: DC motor with static friction nonlinearity at its input and dither signal $d(t)$.

Since our crude model of static friction is (5.7) we pick a dither signal with amplitude 0.2. The frequency of the dither signal is picked somewhere above the bandwidth of the closed-loop system so that it doesn't affect the output too much. In this case we pick $d(t) = 0.2\sin(100t)$. The simulation result is shown in Figure 5.35.

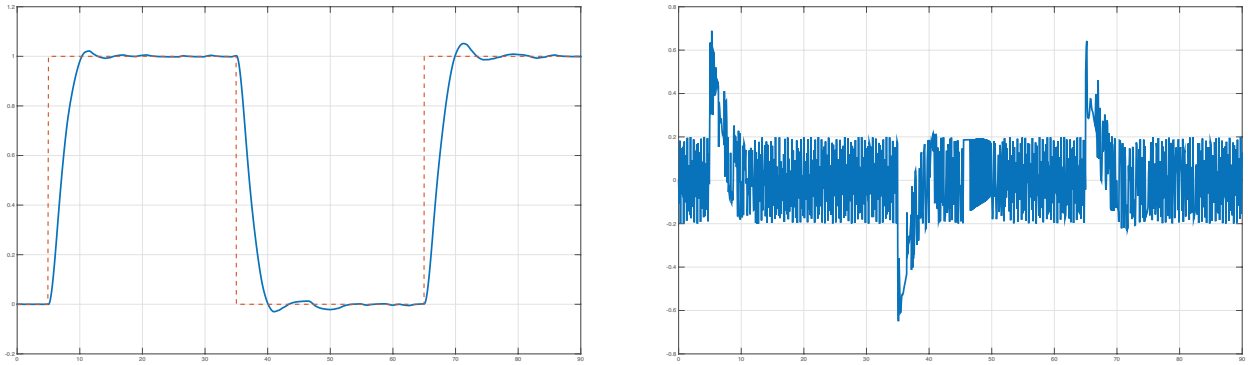
The dither signal has improved the steady-state tracking despite the presence of static friction. On the other hand the control signal is very "jittery" which could lead to accelerated wear and tear of actuators. ▲

5.4.2 Model static friction as a dead zone nonlinearity

We gave a crude model of static friction in Example 5.4.1. In reality the coefficient μ_s varies according to the temperature and humidity; it's time-varying. Even if we assume that μ_s is constant, our crude model isn't static nor is it right invertible. To overcome these problems we use an even cruder model of static friction: we model static friction as a deadzone nonlinearity shown in Figure 5.36.

The deadzone nonlinearity is onto and is therefore right invertible. It can be used to *very* roughly model static friction. This means that we can use the inversion technique to cancel its effect as shown in Figure 5.37

The inversion approach is easy to implement because the parameters u_+ and u_- from Figure 5.36 are easy to measure. It also avoids causing unnecessary wear and tear like the dither method. On the other hand, the dead



(a) $\theta(t)$ and $\theta_{\text{ref}}(t)$. *(b) Control signal $u(t)$.*

Figure 5.35: Pulse train response from Example 5.4.3.

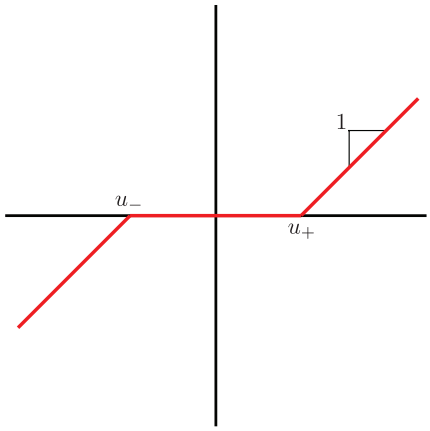


Figure 5.36: Deadzone nonlinearity.

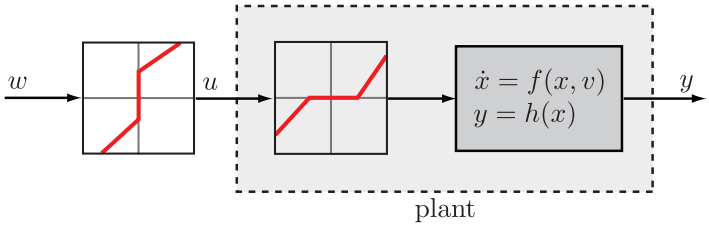


Figure 5.37: Inverting a deadzone nonlinearity.

zone nonlinearity model for the static friction is very crude and may not always work very well. Furthermore, the values of u_+ and u_- may need to be re-determined occasionally because the static friction coefficient μ_s is time-varying.

Example 5.4.4. (Using Deadzone Inversion to Overcome Static Friction) Returning to Example 5.4.2 we model the static friction nonlinearity (5.7) as a deadzone nonlinearity with $u_+ = -u_- = 0.2$. The implementation is shown in Figure 5.38. The simulation result is shown in Figure 5.39.

The deadzone inversion has improved the steady-state tracking despite the presence of static friction. On the other hand, the nature of our inverting function is such that the control signal will never settle at zero. ▲

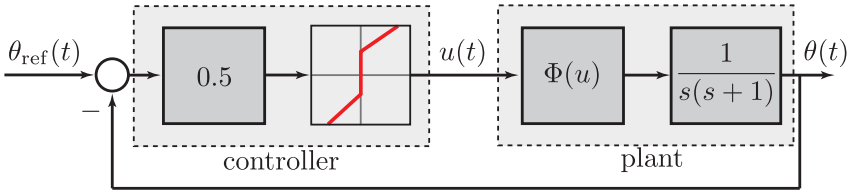
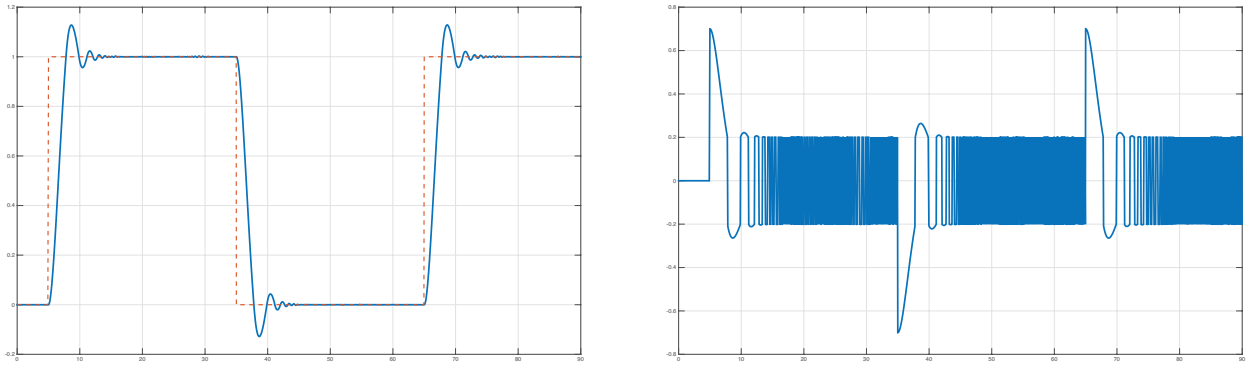


Figure 5.38: DC motor with deadzone inversion.



(a) $\theta(t)$ and $\theta_{\text{ref}}(t)$. *(b) Control signal $u(t)$.*

Figure 5.39: Pulse train response from Example 5.4.4.

5.5 Describing functions

In this section, the concept of describing functions are introduced, and it is demonstrated that they can be used to predict the existence of periodic solutions in feedback systems. We begin with an example to motivate our analysis.

Example 5.5.1. (Thermostat) Most thermostats are on-off controllers with a built-in hysteresis. A typical control loop looks like that in Figure 5.40. The control signal equals 1 to turn on the furnace and equals 0 to

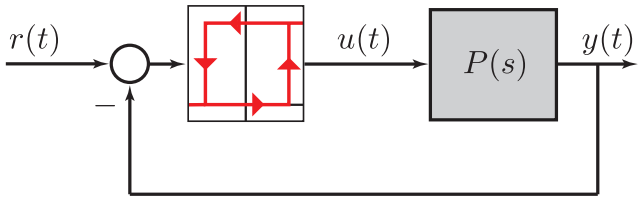


Figure 5.40: Thermostat control.

turn off the furnace. The hysteresis prevents "chattering" of the control signal. Let $r(t) = \mathbf{1}(t)$, $P(s) = 2/(s+1)$ and let the furnace turn on when $r(t) - y(t) > 0.5$; let the furnace turn off when $r(t) - y(t) < -0.2$. Simulation results are shown in Figure 5.41. The oscillations in the output response are due to the hysteresis. ▲

While it is fairly easy to predict the existence of periodic solutions in the above example, we are interested predicting the existence of periodic solutions in less obvious scenarios. In general, proving the existence of periodic solutions is a very difficult and deep problem[4]. Our approach is not completely rigorous but is actually very effective in practice. It is a method based on something called a describing function. The advantage of this approach is that we can use linear systems tools; the disadvantage is that the approach may not always work because we are making approximations along the way.

[4]Look up, for example, Hilbert's 16th problem.

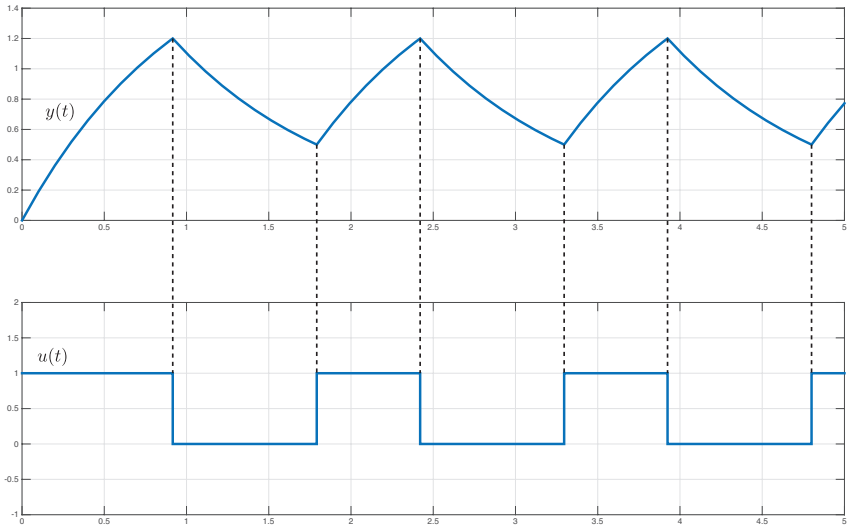


Figure 5.41: Output signal (top) and control signal (bottom) for thermostat system.

5.5.1 Optimal Quasi-Linearization

Consider the setup in Figure 5.42. Given a nonlinear system and a reference input $r(t)$, we are interested in

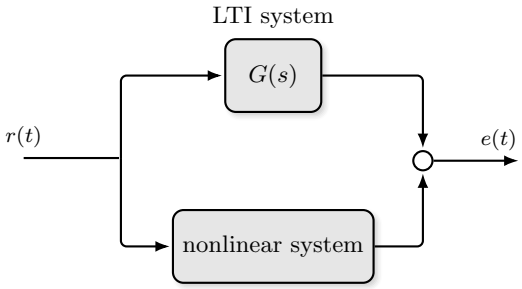


Figure 5.42: Comparing the response of an LTI system to a nonlinear system.

finding the "best" LTI system $G(s)$ in the sense that G minimizes the error

$$E(G) = \lim_{T\to\infty} \frac{1}{T} \int_0^T e^2(t)\ \mathrm{d}t.$$

The LTI system G is called the **optimal quasi-linearization** of the nonlinear system. Under certain technical assumptions which we don't cover, the solution to this problem is known [Vidyasagar, 2002, Theorem 4.1.12].

Remark 5.5.1. The approximation G is best for a given reference input. If the reference input $r(t)$ changes, so too does the optimal choice of G. Moreover, for a given reference input $r(t)$, the system G is not unique. ♦

By far, the most common reference signal is a sinusoid

$$r(t) = a\sin(\omega t), \qquad a, \omega > 0.$$

The next result, given without proof, justifies the describing function method.

Theorem 5.5.2 ([Vidyasagar, 2002, Theorem 4.1.36]). *Let $r(t) = a\sin(\omega t)$ and let y be the output of a nonlinear system under the input $r(t)$. Assume that*

(i) The output of the nonlinear system has the form $y(t) = y_{\text{tr}}(t) + y_{\text{ss}}(t)$ where $y_{\text{tr}}(t)$ is the transient response and $y_{\text{ss}}(t)$ is the steady-state response.

(ii) The steady-state response y_{ss} of the nonlinear system due to the input $r(t)$ is periodic with period $2\pi/\omega$ and

$$\int_0^\infty y_{\text{tr}}^2(t)\ \mathrm{d}t < \infty.$$

(iii) The transfer function $G(s)$ is proper and has all its poles in the open left half complex plane.

Under these conditions, G is an optimal quasi-linearization of the nonlinear system for the input $r(t)$ if, and only if

$$G(j\omega) = \frac{a_1(a,\omega) + jb_1(a,\omega)}{a}$$

where

$$a_1(a,\omega)\sin(\omega t) + b_1(a,\omega)\cos(\omega t)$$

is the first harmonic of $y_{\text{ss}}(t)$.

Remark 5.5.3. The condition on $G(j\omega)$ from Theorem 5.5.2 is called the **principle of harmonic balance**. The theorem says that the optimal choice of G is one whose steady-state response due to the sinusoidal input $r(t) = a\sin(\omega t)$ precisely matches the first harmonic of the steady-state response of the nonlinear system. Since this condition only needs to hold at the frequency ω, it's clear that there are infinitely many optimal choices of G. ♦

Definition 5.5.4. Let $r(t)$ and $y(t)$ be as in Theorem 5.5.2. The **describing function** of the nonlinear system is the complex-valued function

$$\eta(a,\omega) = \frac{a_1(a,\omega) + jb_1(a,\omega)}{a}. \tag{5.8}$$

While there are infinitely many optimal choices of the linear system G, once a and ω are fixed, the describing function is unique. Describing functions for static nonlinearities do not depend on the frequency ω.

Proposition 5.5.5. *The describing function of a static nonlinearity of the form $y(t) = \Phi(r(t))$ where $\Phi : \mathbb{R} \to \mathbb{R}$, is independent of ω.*

Recall that a function $\Phi : \mathbb{R} \to \mathbb{R}$ is **odd** if $\Phi(x) = -\Phi(-x)$. The describing function of an odd, static nonlinearity is a real-valued function.

Proposition 5.5.6. *The describing function of an odd, static nonlinearity of the form $y(t) = \Phi(r(t))$ where $\Phi : \mathbb{R} \to \mathbb{R}$, is real-valued.*

5.5.2 Constructing describing functions

In this section we explain how to compute the describing function of a nonlinear system and provide some examples. Let $r(t) = a\sin(\omega t)$ where $a \in \mathbb{R}$ and $\omega > 0$ are fixed constants. For a given nonlinear system which satisfies all the assumptions of Theorem 5.5.2, let $y_{\text{ss}}(t)$ denote its steady-state response due to the input $r(t)$.

By assumption this means that $y_{\text{ss}}(t)$ is $2\pi/\omega$-periodic, i.e., for any time t

$$y_{\text{ss}}(t) = y_{\text{ss}}(t + 2\pi/\omega).$$

As you know from your calculus and signals and systems courses [Greenberg, 1998], this function can be expressed using a trigonometric Fourier series. Specifically, if $y_{\text{ss}}(t)$ is periodic with period $T := 2\pi/\omega$, then the **Fourier series** of $y_{\text{ss}}(t)$ is

$$a_0 + \sum_{n=0}^{\infty} a_n \sin(n\omega t) + \sum_{n=1}^{\infty} b_n \cos(n\omega t)$$

where

$$a_0 := \frac{1}{T}\int_{-\frac{T}{2}}^{\frac{T}{2}} y_{\text{ss}}(t)\ \mathrm{d}t, \qquad a_n := \frac{2}{T}\int_{-\frac{T}{2}}^{\frac{T}{2}} y_{\text{ss}}(t)\sin(n\omega t)\ \mathrm{d}t, \qquad b_n := \frac{2}{T}\int_{-\frac{T}{2}}^{\frac{T}{2}} y_{\text{ss}}(t)\cos(n\omega t)\ \mathrm{d}t,$$

$n \in \{1, 2, 3, \ldots\}$, are called the **Fourier coefficients** of $y_{\text{ss}}(t)$. The coefficients a_1, b_1 appear in the describing function (5.8).

Example 5.5.2. (Sign Nonlinearity) Consider the sign nonlinearity in Figure 5.43. This nonlinearity can model, for example, an ideal relay. Our objective is to find the describing function of this nonlinearity for the

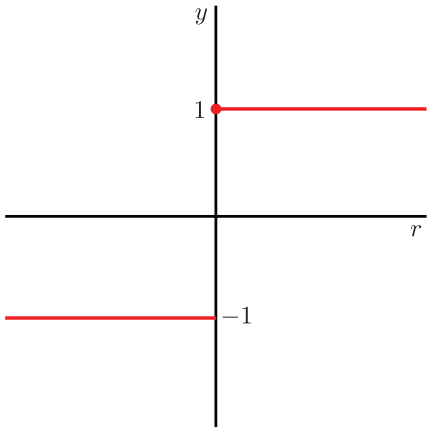


Figure 5.43: Sign nonlinearity.

reference input $r(t) = a\sin(\omega t)$. By Propositions 5.5.5 and 5.5.6 we expect that the describing function will be real-valued and only depend on a, not ω. Figure 5.44 shows the output of the nonlinearity when the input is $r(t) = a\sin(\omega t)$.

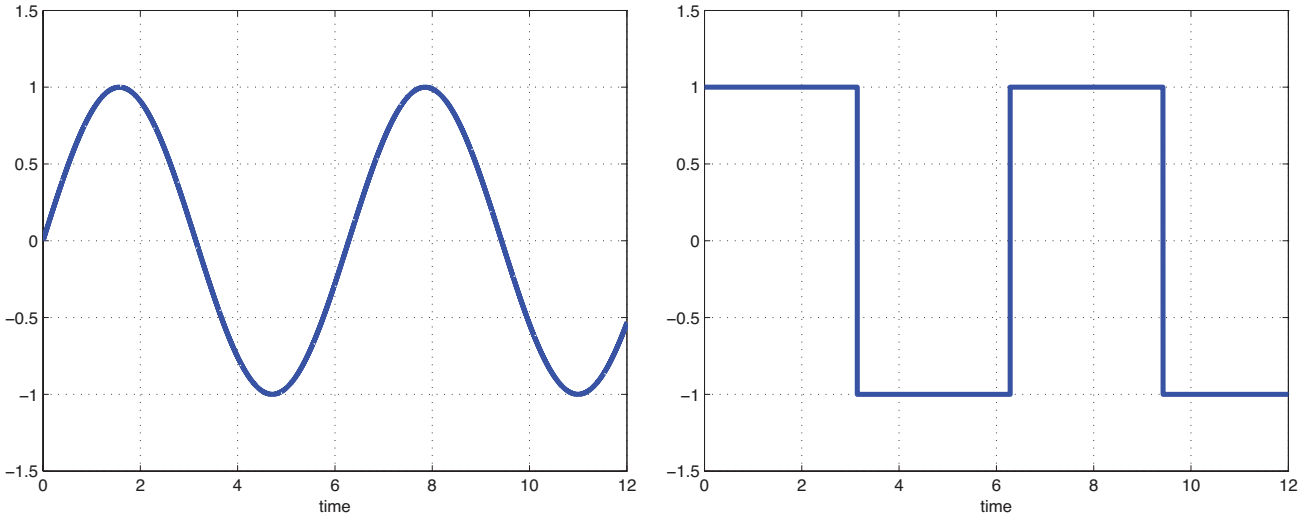
Since the nonlinearity is memoryless it's clear that $y(t) = y_{\text{ss}}(t)$, i.e., there is no transient component to the response and that $y(t)$ has period $T = 2\pi/\omega$ with

$$y_{\text{ss}}(t) = \begin{cases} -1 & -T/2 < t < 0 \\ 1 & 0 \le t \le T/2. \end{cases}$$

Computing the first components of the Fourier series expansion of $y_{\text{ss}}(t)$ we get

$$a_1 = \frac{2}{T}\int_{-\frac{T}{2}}^{\frac{T}{2}} y_{\text{ss}}(t)\sin(\omega t)\ \mathrm{d}t = \frac{4}{T}\int_{0}^{\frac{T}{2}} \sin(\omega t)\ \mathrm{d}t = \frac{4}{\pi}$$

$$b_1 = \frac{2}{T}\int_{-\frac{T}{2}}^{\frac{T}{2}} y_{\text{ss}}(t)\cos(\omega t)\ \mathrm{d}t = \frac{2}{T}\int_{-\frac{T}{2}}^{0} -\cos(\omega t)\ \mathrm{d}t + \frac{2}{T}\int_{0}^{\frac{T}{2}} \cos(\omega t)\ \mathrm{d}t = 0 + 0 = 0.$$



(a) $r(t) = a\sin(\omega t)$, $a = 1$, $\omega = 1$. *(b) Output* $y(t) = \Phi(r(t))$.

Figure 5.44: Input and output signals to the sign nonlinearity.

Therefore the describing function is

$$\begin{aligned}\eta(a,\omega) &= \frac{a_1(a,\omega) + jb_1(a,\omega)}{a} \\ &= \frac{a_1(a,\omega)}{a} \\ &= \frac{4}{a\pi}.\end{aligned}$$

This makes intuitive sense. If $a \ll 1$, i.e., the input signal has a small amplitude, then the output of the sign nonlinearity is comparatively large. If $a \gg 1$, then the sign nonlinearity attenuates the signal since it can only output ± 1. ▲

Example 5.5.3. (Piecewise Linear Memoryless System) Consider the memoryless nonlinearity Φ with graph shown in Figure 5.45.

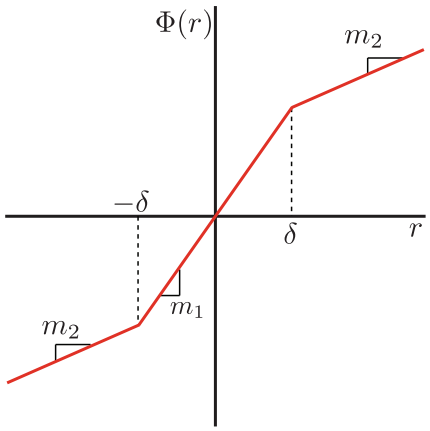


Figure 5.45: Piecewise linear system of Example 5.5.3.

If the input is $r(t) = a\sin(\omega t)$ and $|a| < \delta$, then the output is simply $\Phi(r(t)) = m_1 a\sin(\omega t)$. The describing function for this reference input is a simple gain

$$\eta(a,\omega) = m_1, \qquad |a| \le \delta.$$

On the other hand, if $|a| > \delta$ then the gain reduces to m_2 when the input amplitude exceeds δ. In this case a routine but tedious calculation of the Fourier coefficients gives

$$\eta(a,\omega) = (m_1 - m_2) f(\delta/a) + m_2, \qquad |a| > \delta$$

where

$$f(x) = \frac{2}{\pi}\left(\arcsin(x) + x\sqrt{1-x^2}\right). \tag{5.9}$$

Figure 5.45 depicts the case $m_1 > m_2 > 0$ but the describing function we've obtained is valid for any choice of m_1, m_2. ▲

Example 5.5.4. (Deadzone) The deadzone nonlinearity from Figure 5.36 corresponds to the special case of the nonlinearity in Example 5.5.3 in which $m_1 = 0$ and $m_2 = 1$. In this case we get, for the reference input $r(t) = a\sin(\omega t)$, the describing function

$$\eta(a,\omega) = \begin{cases} 0 & |a| \le \delta \\ -f(\delta/a) + 1 & |a| > \delta \end{cases}$$

where f is given by (5.9). ▲

Example 5.5.5. (Saturator) The saturation nonlinearity from Example 5.3.6 and Figure 5.23 corresponds to the special case of the nonlinearity in Example 5.5.3 in which $m_1 = 1$ and $m_2 = 0$. In this case we get, for the reference input $r(t) = a\sin(\omega t)$, the describing function

$$\eta(a,\omega) = \begin{cases} 1 & |a| \le \delta \\ f(\delta/a) & |a| > \delta \end{cases}$$

where f is given by (5.9). ▲

Remark 5.5.7. All the examples we've done involve static nonlinearities. Things are more complicated when there are nonlinearities with memory because it is more difficult to identify the steady-state output for a given reference signal. ♦

Remark 5.5.8. There are tables of describing functions for common nonlinearities so that in practice you don't need to re-derive them. For example, consult the books [Šiljak, 1968] and [Gelb and Velde, 1968]. Alternatively, there are online resources with re-prints of these tables. ♦

5.5.3 Periodic solutions

In this section we consider the system in Figure 5.46, where Φ is a nonlinear element satisfying all the assumptions of Theorem 5.5.2 and $G(s)$ is a BIBO stable LTI plant, and show how describing functions can be used to predict the existence of periodic solutions.

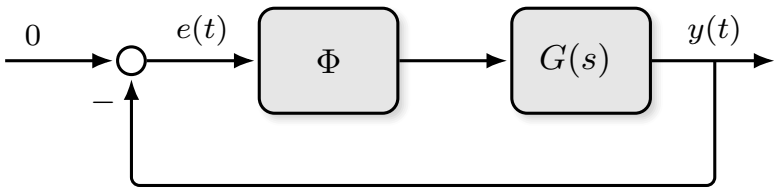


Figure 5.46: Setup considered in Section 5.5.3.

The informal argument proceeds as follows. In steady-state, since $r(t) = 0$ in Figure 5.46, we must have that[5]

$$e(t) = -G(s)\Phi(e). \tag{5.10}$$

Now **assume** that the above equation has a periodic solution of the form

$$e(t) = a\sin(\omega t).$$

Then, since Φ satisfies the assumptions of Theorem 5.5.2, its steady-state response is periodic with period $2\pi/\omega$ and the first harmonic of its output is

$$a_1(a,\omega)\sin(\omega t) + b_1(a,\omega)\cos(\omega t).$$

The describing function has the form

$$\eta(a,\omega) = \frac{a_1(a,\omega) + jb_1(a,\omega)}{a}.$$

The output of the nonlinear element now passes through the BIBO stable LTI system. Therefore, in steady-state, the first harmonic of the signal coming out of the linear block $G(s)$ is (cf. Theorem 2.6.1)

$$|G(j\omega)|a_1(a,\omega)\sin(\omega t + \measuredangle G(j\omega)) + |G(j\omega)|b_1(a,\omega)\cos(\omega t + \measuredangle G(j\omega)).$$

The essence of the approach is to now **equate** the negation of this expression to $e(t)$. Strictly speaking this is not correct since this is not equivalent to (5.10). The rational for doing this is that if $G(s)$ is a low pass system, then the output of $G(s)$ is almost equal to the first harmonic, i.e., the higher order harmonics coming out of the nonlinear element are filtered away. Equating $e(t) = a\sin(\omega t)$ to the negation of the above expression and passing to the frequency domain we get the condition

$$1 + G(j\omega)\eta(a,\omega) = 0. \tag{5.11}$$

If there is a pair (a,ω) which satisfies (5.11), then one deduces that there is a solution $e(t)$ "close" to $a\sin(\omega t)$. Solving (5.11) analytically for a and ω is usually quite difficult. In the special case when the describing function η doesn't depend on ω, i.e., Φ is a static nonlinearity (see Proposition 5.5.5), we can write (5.11) as

$$G(j\omega) = -\frac{1}{\eta(a)}. \tag{5.12}$$

This equation can be solved graphically by plotting $G(j\omega)$ on the complex plane as a function of ω, which is just the Nyquist plot of $G(s)$, and $-1/\eta(a)$ on the same copy of the complex plane. Things are even simpler when the nonlinearity is odd (see Proposition 5.5.6) since in that case its describing function is real-valued. We illustrate this with an example.

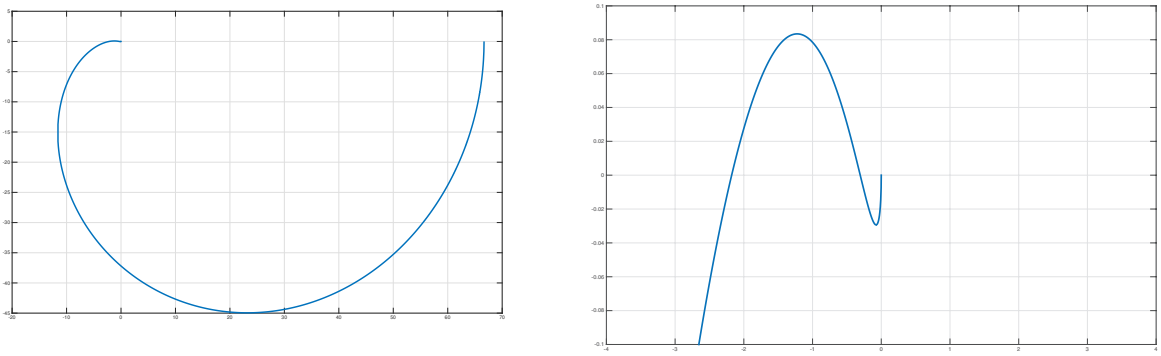
Example 5.5.6. Consider the feedback system in Figure 5.46 where Φ is a saturator nonlinearity (see Examples 5.5.3 and 5.5.5) with $m_1 = 4$, $m_2 = 0$ and $\delta = 1$. Then

$$\eta(a,\omega) = \begin{cases} 4 & |a| \leq 1 \\ \frac{8}{\pi}\left(\arcsin(1/a) + \frac{1}{a}\sqrt{1-\frac{1}{a^2}}\right) & |a| > 1 \end{cases}$$

Furthermore let

$$G(s) = \frac{(s+20)^2}{(s+1)(s+2)(s+3)}.$$

The Nyquist plot of G is shown in Figure 5.47.



(a) Nyquist plot. *(b) Zoomed view of Nyquist plot near origin*

Figure 5.47: Nyquist plot of G in Example 5.5.6

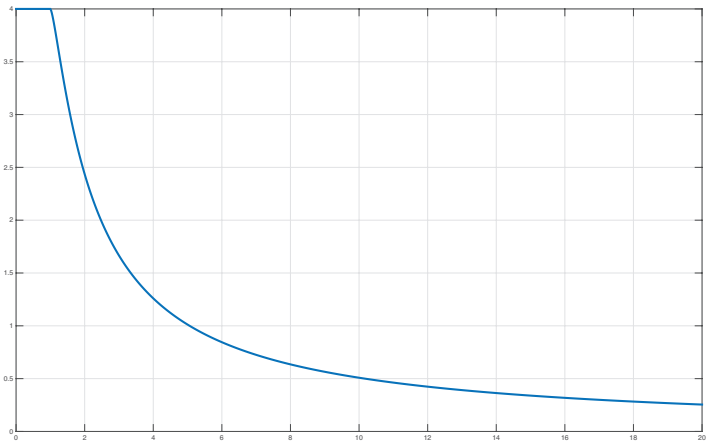


Figure 5.48: Plot of $\eta(a)$ versus a for the saturator nonlinearity of Example 5.5.6.

From the Nyquist plot we obtain that $G(j\omega)$ crosses the real axis around the values 66.7, -2.2 and -0.3 corresponding to the frequencies 0 rad/s, 5.4 rad/s and 11.9 rad/s. Of course we don't care about the crossing that corresponds to 0 rad/s since that solution is constant.

Next we plot $\eta(a)$ versus a. This is shown in Figure 5.48. We seek those values of a where (5.12) is satisfied. In this case we want

$$G(j5.4) \approx -2.2 = -\frac{1}{\eta(a_1)}, \qquad G(j11.9) \approx -0.3 = -\frac{1}{\eta(a_2)}.$$

Graphically these points are illustrated in Figure 5.49. Rearranging we get

$$\eta(a_1) = 0.4545, \qquad \eta(a_2) = 3.3333.$$

From Figure 5.48 we get the values

$$a_1 \approx 11.2, \qquad a_2 \approx 1.4.$$

We conclude that we expect there to be two periodic solutions for this system. The first should be close to $11.2\sin(5.4t)$ and the second is close to $1.4\sin(11.9t)$. Simulating the system from Figure 5.46 we get the output response shown in Figure 5.50.

We indeed have a periodic steady-state solution with an amplitude of around 10.16 and a frequency of around 5.56 rad/s. This is very close to what we expected based on our describing function analysis. Pretty cool! ▲

[5] There is some abuse of notation here since we are mixing the time-domain and the s-domain.

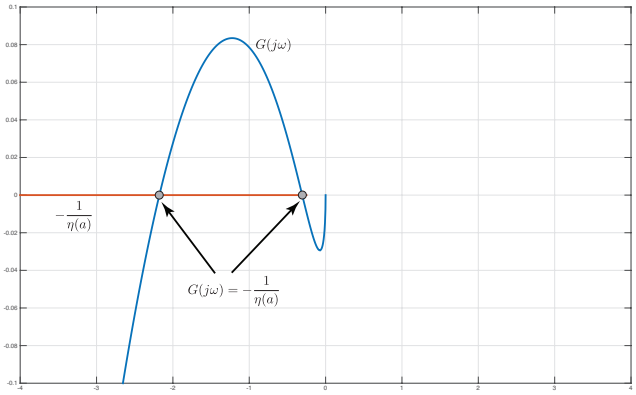


Figure 5.49: Looking for points at which (5.12) holds for Example 5.5.6.

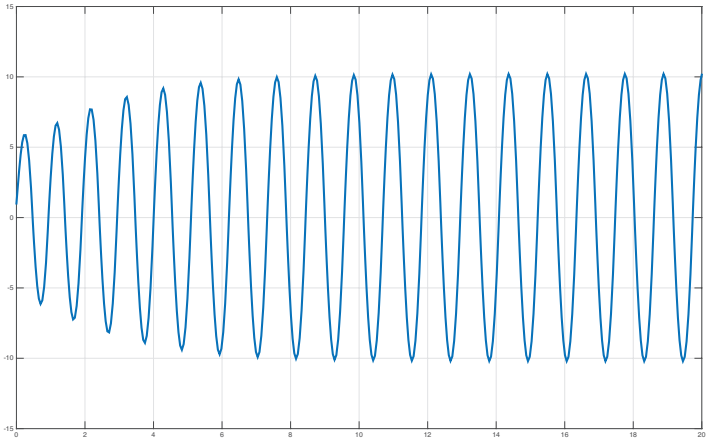


Figure 5.50: Output of system from Example 5.5.6.

Remark 5.5.9. Be careful using describing function analysis to predict the existence of periodic solutions. Describing function analysis may predict the existence of periodic solutions when there are none. A periodic solution may exist even if describing function analysis doesn't predict one. Finally, the predicted amplitude and frequency are only approximations and can be far from the true values. ♦

5.5.4 Stability of Periodic Solutions

In Example 5.5.6 the describing function approach predicted two different periodic solutions but we only observed one in simulation. This is because one of the solutions is "stable" while the other is "unstable." Informally, stability in this context means that if the system is perturbed slightly away from its periodic solution, it eventually returns to the same periodic solution once the perturbation vanishes. If the periodic solution is stable, then there is a good chance that it will occur in the real system.

First recall the Nyquist stability criterion for the system in Figure 5.51. If $G(s)$ is strictly proper and BIBO

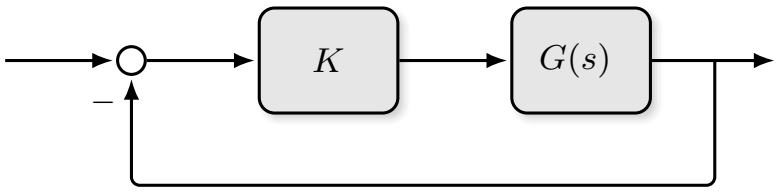


Figure 5.51: Feedback system.

stable, then the Nyquist stability criterion tells us that the closed-loop system is stable if, and only if, the Nyquist plot of $G(s)$ has zero encirclement of the point $-1/K$.

Let's apply this to our describing function setup. For simplicity consider the case of an odd, static nonlinearity. In this case its describing function is real-valued and only depends on the amplitude a of the periodic solution. Our block diagram is shown in Figure 5.52. Again we assume that $G(s)$ is proper and BIBO stable.

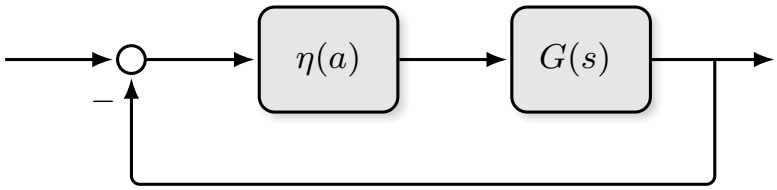


Figure 5.52: Feedback system with real-valued describing function.

Fix the value of a. Then the point $-1/\eta(a)$ lies somewhere on the real axis. If this point is encircled by the Nyquist plot of $G(s)$, then the system is unstable and therefore the amplitude a of oscillations will increase. If the point $-1/\eta(a)$ is not encircled by the Nyquist plot of $G(s)$, then the system is stable and the amplitude of oscillations will decrease thereby reducing a. See Figure 5.53 for an illustration.

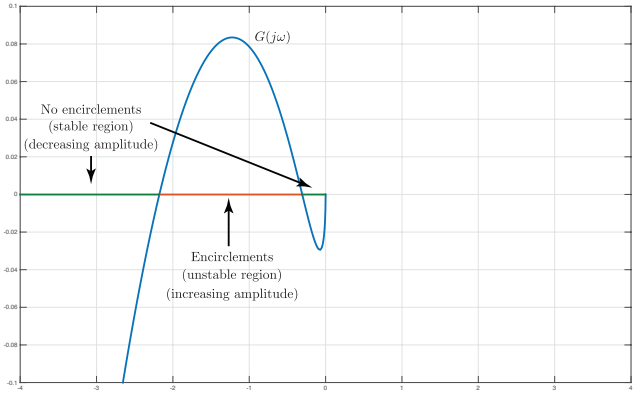


Figure 5.53: Stable and unstable parts of the Nyquist plot tell us where the amplitude of oscillation increases and decreases.

Using this observation, we consider the values of a at which the function $-1/\eta(a)$ intersects the Nyquist plot of $G(s)$. If an increase in a pushes the point $-1/\eta(a)$ into a stable region, then the system response will tend to want to decrease a and we have a stable solution.

Similarly, if a decrease in a pushes the point $-1/\eta(a)$ into an unstable region, then the system response will cause a to increase and we again have a stable solution.

Example 5.5.7. We return to Example 5.5.6. In Figure 5.54 we have plotted the Nyquist plot of G and used an arrow to indicate the direction that the point $-1/\eta(a)$ moves in as the value of a is increased.

Since the direction of $-1/\eta(a)$ as a increases (indicated by arrows in Figure 5.54) opposes the stability expansion/contraction around the left most intersection point (cf. Figure 5.53), we expect the corresponding periodic solution to be stable. On the other hand, the direction of increase for $-1/\eta(a)$ does not oppose the stability expansion/contraction just to left of the right intersection point. Hence we expect this periodic solution to be unstable. This is in agreement with what we observed in the simulation from Example 5.5.6.

If the arrow in Figure 5.54 had been pointing in the other direction, then we would expect that the periodic solution corresponding to the right-most intersection would be stable and the periodic solution corresponding to the left-most intersection would be unstable. ▲

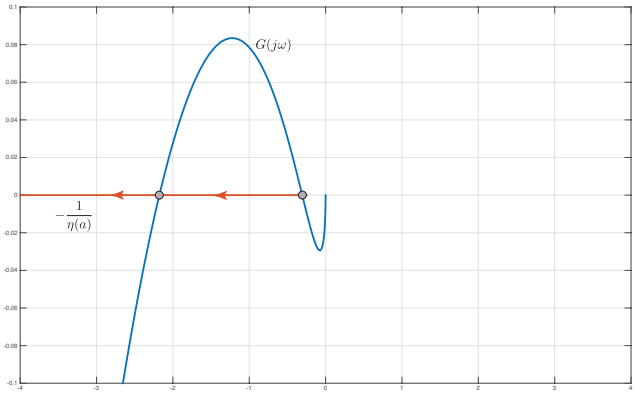


Figure 5.54: Checking for stability of limit cycles.

Chapter 6

Discrete-time linear systems

Just as for continuous-time linear time-invariant systems, there are three common ways to describe discrete-time linear systems: 1) difference equation models 2) transfer function models 3) state-space models. We shall study how to use each of these models for analysis, and show how to go back and forth from one description to another.

Contents

6.1 Difference equations

One of the common ways to model a linear DT system is a **difference equation**. The general form of a causal system is

$$y[k] + a_1 y[k-1] + \cdots + a_n y[k-n] = b_0 u[k] + b_1 u[k-1] + \cdots + b_m u[k-m] \tag{6.1}$$

where k is the discrete-time variable, an integer; $u[k]$ is the input sequence; $y[k]$ is the output sequence; and a_i, b_i are real, constant coefficients. Some simple examples are

$$\begin{aligned} y[k] &= u[k-1], && \text{delay system} \\ y[k] &= u[k+1], && \text{not causal.} \end{aligned}$$

Example 6.1.1. (Moving average) Consider the DT system defined by

$$y[k] = \frac{1}{5}\left(u[k] + u[k-1] + u[k-2] + u[k-3] + u[k-4]\right).$$

The current output of the system is the average of its last 5 inputs; thus we call it the 5-point moving average. This is an equation of the form (6.1) in which and $b_i = 0.2$, $i \in \{0, 1, 2, 3, 4\}$ and all other coefficients are zero. Consider next a twenty-point moving average

$$y[k] = 0.05\left(u[k] + u[k-1] + \cdots + u[k-18] + u[k-19]\right).$$

If we subtract 1 from every time index we get

$$y[k-1] = 0.05\left(u[k-1] + u[k-2] + \cdots + u[k-19] + u[k-20]\right).$$

Subtract this equation from the original difference equation for the twenty-point moving average to get

$$y[k] - y[k-1] = 0.05\left(u[k] - u[k-20]\right)$$

which is of the form (6.1) with $a_1 = -1$, $b_0 = 0.05$, $b_{20} = -0.05$. ▲

Exercise 6.1. The discrete-time impulse is the signal with $\delta[k] = 1$ if $k = 0$ and $\delta[k] = 0$ otherwise. Find the output sequence of the 5-point moving average system when its input is a discrete-time impulse. This is called the **impulse response** of the system.

Example 6.1.2. (Savings account) Consider a savings account in a bank. Let $u[k]$ be the amount of money deposited or withdrawn on the kth day and $y[k]$ be the total amount of money in the account at the end of the kth day. The savings account can be considered a DT system with input $u[k]$ and output $y[k]$. It is a causal system because the current output $y[k]$ does not depend on future inputs. Assume an interest rate of 0.015% per day, compounded daily. Then the system can be modelled by

$$y[k] = y[k-1] + 0.00015y[k-1] + u[k]$$

or, rearranging,

$$y[k] - 1.00015y[k-1] = u[k]$$

which is again a system of the form (6.1). ▲

Given an input sequence and a set of initial conditions, how do we solve for the corresponding output sequence?

Example 6.1.3. Consider

$$y[k] - 2y[k-1] = 2u[k] + u[k-1].$$

Obviously, we can solve this equation for $\{y[k]\}_{k\geq 0}$ if we know $y[-1]$, $\{u[k]\}_{k\geq -1}$ by recursion

$$y[k] = 2y[k-1] + 2u[k] + u[k-1], \qquad k \geq 0.$$

For example, say that $u[k] = k\mathbf{1}[k]$ and that $y[-1] = 1$, then

$$\begin{aligned} y[0] &= 2y[-1] + 2u[0] + u[-1] = 2, \\ y[1] &= 2y[0] + 2u[1] + u[0] = 6, \\ y[2] &= 2y[1] + 2u[2] + u[1] = 17, \\ y[3] &= 2y[2] + 2u[3] + u[2] = 42, \end{aligned}$$

etc. However, we may like to determine a closed form solution for y given u. ▲

As with linear differential equations with constant coefficients, the general solution of (6.1) can be written as $y = y_n + y_f$, where y_n is the general solution to the homogeneous equation

$$y[k] + a_1 y[k-1] + \cdots + a_n y[k-n] = 0.$$

The signal y_n is also called the **natural** or **zero-input** response because it is determined entirely by the characteristics or the *nature* of the system, not the input. The signal y_f is the **forced** response (also called the **zero-state** response) and is obtained by solving (6.1) under the assumption that $y[k]$ and $u[k]$ equal zero for all $k < 0$.

6.1.1 Natural response

For ordinary differential equations with constant coefficients, the trial solution is a complex exponential, $e^{\lambda t}$. This is because if you differentiate such a function, you get another function of the same form.

Example 6.1.4. Consider

$$\ddot{y} + 3\dot{y} - 2y = 0.$$

Try $y(t) = e^{\lambda t}$:

$$\lambda^2 e^{\lambda t} + 3\lambda e^{\lambda t} - 2e^{\lambda t} = 0.$$

This equation is true for all k if, and only if

$$\lambda^2 + 3\lambda - 2 = 0.$$

This second order polynomial has two roots $\{-1, -2\}$. So the general form of the natural response is

$$y_n(t) = c_1 e^{-t} + c_2 e^{-2t}.$$

▲

For linear constant coefficient difference equations we take $y_n[k] = \lambda^k$ as a trial solution. This is because if you replace k with $k-1$ you get another function of the same form.

Example 6.1.5. Consider

$$y[k] - y[k-1] - 4y[k-2] + 4y[k-3] = 0.$$

Substitute $y[k] = \lambda^k$:

$$\lambda^k - \lambda^{k-1} - 4\lambda^{k-2} + 4\lambda^{k-3} = 0.$$

This equation is true if, and only if

$$\lambda^3 - \lambda^2 - 4\lambda + 4 = 0.$$

The above polynomial has roots $\{1, 2, -2\}$ which yields three possible solutions $y_1[k] = 1^k$, $y_2[k] = 2^k$, $y_3[k] = (-2)^k$. They're linearly independent functions (review this concept from Linear Algebra). Thus the general form of the natural response is

$$y_n[k] = c_1(1)^k + c_2 2^k + c_3(-2)^k.$$

▲

Example 6.1.6. Returning to the difference equation from Example 6.1.3 we have the homogenous equation $y[k] - 2y[k-1] = 0$. Try $y[k] = \lambda^k$ to get the polynomial $\lambda - 2 = 0$. Its lone root is 2 and therefore the natural response $y_n[k] = c2^k$. ▲

The best way to get the forced response is via the z-transform which we treat next.

6.2 z-Transforms

The z-transform is to discrete-time systems as the Laplace transform is to continuous-time systems. Let $x[k]$ be a real-valued discrete-time signal. Its **z-transform** is defined to be[1]

$$X[z] \coloneqq \sum_{k=0}^{\infty} x[k] z^{-k}. \tag{6.2}$$

[1] More precisely this is the *one-sided* z-transform – there's a two-sided z-transform that you may have encountered in a DSP course.

Here z is a complex variable. We use the symbol $\mathscr{Z}(x)$ to denote the z-transform of a signal $x[k]$. Not every DT signal has a z-transform. The sufficient condition for $x[k]$ to have a z-transform is that it satisfy a growth bound

$$|x[k]| \leq M\rho^k, \qquad k \geq 0$$

where M and ρ are positive constants. Then, writing z in polar form $z = re^{j\theta}$ with $r > \rho$, we have

$$\begin{aligned}
|X[z]| &= \left| \sum_{k=0}^{\infty} x[k] \left(re^{j\theta} \right)^{-k} \right| \\
&\leq \sum_{k=0}^{\infty} \left| x[k] \left(re^{j\theta} \right)^{-k} \right| \\
&= \sum_{k=0}^{\infty} |x[k]|\, r^{-k} \\
&\leq M \sum_{k=0}^{\infty} \left(\frac{\rho}{r} \right)^k \\
&= M \frac{1}{1 - \frac{\rho}{r}} \\
&< \infty \qquad \text{since } r > \rho.
\end{aligned}$$

Thus the region of convergence for the series includes the exterior of the disk $|z| > \rho$. The smallest ρ defines its region of convergence.

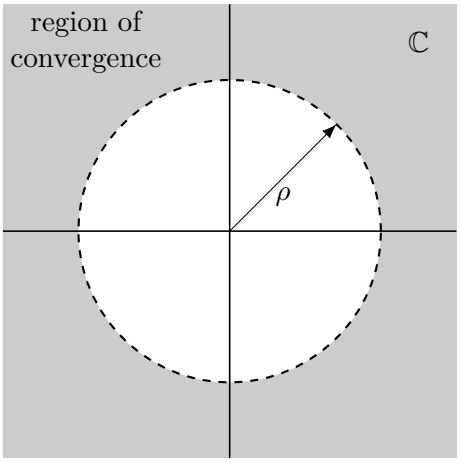


Figure 6.1: Region of convergence for z-transform.

Example 6.2.1.

$$x[k] = 1.3^k, \qquad k \geq 0.$$

$$X[z] = \mathscr{Z}\left(1.3^k\right) = \sum_{k=0}^{\infty} 1.3^k z^{-k}.$$

This is an infinite power series and is not very useful. If we re-write the sum as a geometric series we get

$$X[z] = \sum_{k=0}^{\infty} \left(1.3z^{-1}\right)^k = \frac{1}{1 - 1.3z^{-1}} = \frac{z}{z - 1.3}$$

which is a simple rational function of z. Observe that the above equality only holds when $|1.3z^{-1}| < 1$ or $|z| > 1.3$. For example, if $z = 1$, the infinite sum diverges and does not equal $1/(1 - 1.3) = -10/3$. Thus we have that

$$\mathscr{Z}\left(1.3^k\right) = \frac{z}{z - 1.3}, \qquad \text{ROC} : \{z \in \mathbb{C} : |z| > 1.3\}.$$

▲

Example 6.2.2. More generally consider

$$x[k] = a^k, \qquad k \geq 0.$$

$$X[z] = \mathscr{Z}\left(a^k\right) = \sum_{k=0}^{\infty} \left(az^{-1}\right)^k = \frac{1}{1 - az^{-1}} = \frac{z}{z - a}, \qquad |z| > |a|.$$

Special case: if $a = 1$, then we obtain the z-transform of the unit step $\mathbf{1}[k]$. ▲

The signal in the next example grows too fast and so it doesn't have a z-transform.

Example 6.2.3.

$$x[k] = k^k, \qquad k \geq 0.$$

$$X[z] = \sum_{k=0}^{\infty} \left(\frac{k}{z}\right)^k = \lim_{j \to \infty} \sum_{k=0}^{j} \left(\frac{k}{z}\right)^k.$$

The above summation fails to converge for any $z \in \mathbb{C}$. This is because for any z with finite magnitude, the term $\left|\frac{k}{z}\right|$ is not less than one for all k. ▲

6.2.1 Inverse z-transform

Suppose we have $X[z]$ and we want to find $x[k]$. The inversion formula is

$$x[k] = \frac{1}{2\pi j} \oint_C X[z] z^{k-1} \, \mathrm{d}z. \tag{6.3}$$

This integral is a contour integral in the complex plane around a circle C inside the region of convergence as shown in Figure 6.2. In particular, selecting $z = re^{j\theta}$ in (6.3), yields

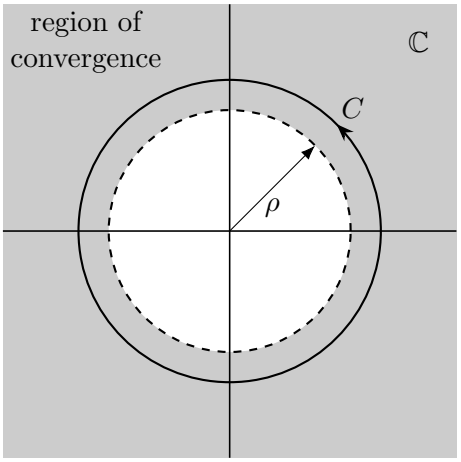


Figure 6.2: Counter integral in the inversion formula for z-transforms.

$$x[k] = \frac{r^k}{2\pi} \int_0^{2\pi} X[re^{j\theta}] e^{jk\theta} \, d\theta.$$

We will use the symbol $\mathscr{Z}^{-1}\{X[z]\}$ to denote the inverse z-transform of $X[z]$. The region of convergence is important if we want to use the integral formula to compute the inverse z-transform.

Example 6.2.4. Consider the z-transform of 1.3^k from Example 6.2.1

$$X[z] = \frac{z}{z - 1.3}, \qquad \text{ROC} : \{z \in \mathbb{C} : |z| > 1.3\}.$$

To get $x[k]$ consider the integral formula (6.3). If $z = re^{j\theta}$ with $r > 1.3$, then the circle traced out as θ goes from 0 to 2π is inside the ROC. We get

$$x[k] = \frac{1}{2\pi j} \int_0^{2\pi} X[re^{j\theta}] \left(re^{j\theta}\right)^{k-1} rje^{j\theta} d\omega = \frac{r^k}{2\pi} \int_0^{2\pi} X[re^{j\theta}] e^{jk\theta} d\theta.$$

If we evaluate this integral with $r = 2$ we obtain the original sequence

$$x[k] = \begin{cases} 1.3^k & k \geq 0 \\ 0 & k < 0. \end{cases}$$

However, if we select $r = 1$ in the above integral so that the resulting contour *isn't* in the ROC, then we get

$$x[k] = \begin{cases} 0 & k \geq 0 \\ -1.3^k & k < 0 \end{cases}.$$

In conclusion, without specifying the ROC, the inversion formula (6.3) may not yield the original $x[k]$. ▲

Since a discrete-time signal in computer control is normally defined for $k \geq 0$, it invariably gives rise to a z-transform with a region of convergence being the exterior of a disk with a sufficiently large radius. For this reason, the region of convergence for a transform $X[z]$ is often omitted with the understanding that it is the exterior of the smallest disk that includes all the poles of $X[z]$. In this case the relationship between $x[k]$ and $X[z]$ is one-to-one. Once we obtain a z-transform $X[z]$ we automatically assume its inverse z-transform to be a signal only defined for $k \geq 0$. Under this assumption we can generate tables of z-transforms so that in practice, if you have $X[z]$, you can get $x[k]$ using a table like Table 6.1.

We list, without proof, the basic properties of z-transforms. Consult any introductory book on signals and systems, e.g., [Lathi, 1992], for their proof.

Properties of z-transforms

Let $x[k]$ and $y[k]$ be real-valued sequences that are zero for $k < 0$, let a be a real constant and let $m \geq 0$ be an integer.

(i) $\mathscr{Z}\{x[k] + y[k]\} = \mathscr{Z}\{x[k]\} + \mathscr{Z}\{y[k]\}$ (superposition)

(ii) $\mathscr{Z}\{ax[k]\} = a\mathscr{Z}\{x[k]\}$ (homogeneity)

(iii) $\mathscr{Z}\{x[k-m]\} = z^{-m}\mathscr{Z}\{x[k]\} + x[-m] + z^{-1}x[-m+1] + \cdots + z^{-m+1}x[-1]$ (backward shift)

(iv) $\mathscr{Z}\{x[k+m]\} = z^m\mathscr{Z}\{x[k]\} - \left(z^m x[0] + z^{m-1}x[1] + \cdots + zx[m-1]\right)$ (forward shift)

(v) $\mathscr{Z}\{x * y\} = \mathscr{Z}\{x[k]\}\,\mathscr{Z}\{y[k]\}$ (convolution)

(vi) $\mathscr{Z}\{a^k x[k]\} = X[z/a]$. (multiplication by a^k)

Table 6.1: Important (one-sided) z-transforms.

Description	Time-domain $x[k]$	z-Domain $X[z]$
Unit step	$\mathbf{1}[k]$	$\frac{z}{z-1}$
Impulse	$\delta[k]$	1
Ramp	k	$\frac{z}{(z-1)^2}$
Exponential	a^k	$\frac{z}{z-a}$
Exponential I	ka^k	$\frac{az}{(z-a)^2}$
Exponential II	k^2a^k	$\frac{az(z+a)}{(z-a)^3}$
Sine	$\sin(\theta k)$	$\frac{z\sin(\theta)}{z^2-2\cos(\theta)z+1}$
Cosine	$\cos(\theta k)$	$\frac{z(z-\cos(\theta))}{z^2-2\cos(\theta)z+1}$
Growing/decaying sine	$a^k\sin(\theta k)$	$\frac{za\sin(\theta)}{z^2-2a\cos(\theta)z+a^2}$
Growing/decaying cosine	$a^k\cos(\theta k)$	$\frac{z(z-a\cos(\theta))}{z^2-2a\cos(\theta)z+a^2}$

Example 6.2.5. (Partial fraction expansion for taking inverse transforms)

$$X[z] = \frac{z+1}{(z-10)(z+4)}$$

A useful trick is to write out the partial fraction expansion of $X[z]/z$ so that we get terms we can easily invert. We only have simple poles for $X[z]$ so the partial fraction expansion we seek is of the form

$$\frac{X[z]}{z} = \frac{z+1}{z(z-10)(z+4)} = \frac{c_1}{z} + \frac{c_2}{z-10} + \frac{c_3}{z+4}.$$

Hence we have

$$\begin{aligned} c_1 &= z\frac{z+1}{z(z-10)(z+4)}\bigg|_{z=0} = -\frac{1}{40} \\ c_2 &= (z-10)\frac{z+1}{z(z-10)(z+4)}\bigg|_{z=10} = \frac{11}{140} \\ c_3 &= (z+4)\frac{z+1}{z(z-10)(z+4)}\bigg|_{z=-4} = -\frac{3}{56}. \end{aligned}$$

Therefore

$$X[z] = -\frac{1}{40} + \frac{11}{140}\frac{z}{z-10} - \frac{3}{56}\frac{z}{z+4}$$

and using linearity of the z-transform we can invert term-by-term to get

$$x[k] = -\frac{1}{40}\delta[k] + \frac{11}{140}10^k - \frac{3}{56}(-4)^k, \qquad k \geq 0.$$

▲

Just as in CT, there is a final-value theorem for DT signals which relates the time-domain to the z-domain.

Theorem 6.2.1 (Final-value theorem). *Let $x[k]$ be a signal defined for $k \geq 0$ and let its z-transform $X[z]$ be rational and proper.*

(a) If $X[z]$ has all its poles in $|z| < 1$, then $x[k]$ converges to zero as $k \to \infty$.

(a) If $X[z]$ has all its poles in $|z| < 1$ except for a simple pole at $z = 1$, then

$$\lim_{k\to\infty} x[k] = \lim_{z\to 1}(z-1)X[z]. \tag{6.4}$$

(c) In all other cases, $x[k]$ does not approach a constant as $k \to \infty$.

6.3 Solving difference equations by z-transforms

We start with an example.

Example 6.3.1. Consider the system

$$y[k] - 2y[k-1] = u[k], \qquad y[-1] = 1, \quad u[k] = k\mathbf{1}[k].$$

In Example 6.1.6 we found the natural response to be $y_n[k] = c2^k$. In this example we'll use z-transforms to get the forced response. Set all initial conditions to zero and take z-transforms of everything in sight

$$\begin{aligned} Y[z] - 2z^{-1}Y[z] &= U[z] \\ \Rightarrow Y[z] &= \frac{z}{z-2}U[z]. \end{aligned}$$

Since $u[k] = k$, from Table 6.1 we have

$$U[z] = \frac{z}{(z-1)^2}.$$

Therefore

$$Y[z] = \frac{z}{z-2}\frac{z}{(z-1)^2}.$$

Write out the partial fraction expansion for $Y[z]/z$ as

$$\frac{Y[z]}{z} = \frac{c_1}{z-2} + \frac{c_2}{z-1} + \frac{c_3}{(z-1)^2}$$

and solve for $c_1 = 2$, $c_2 = -2$, $c_3 = -1$. We get that

$$Y[z] = 2\frac{z}{z-2} - 2\frac{z}{z-1} - \frac{z}{(z-1)^2}$$

and, going back to the time-domain, the forced response is

$$y_f[k] = 2(2)^k - 2(1)^k - k, \qquad k \geq 0.$$

The total response is $y = y_n + y_f$ which is

$$y[k] = (c+2)(2)^k - 2(1)^k - k, \qquad k \geq 0.$$

We can't solve for c by subbing in $k = -1$ and using the fact that $y[-1] = 1$. This is because the solution to our forced response assumes that $k \geq 0$. Therefore, to obtain c we first find the value of $y[0]$ consistent with the input and the constraint $y[-1] = 1$. We did this in Example 6.1.3 to get $y[0] = 2$. Subbing in $k = 0$ and solving yields $c = 2$ and so the total response is

$$y[k] = 4(2)^k - 2(1)^k - k, \qquad k \geq 0.$$

▲

We can also obtain the total response using solely z-transforms.

Example 6.3.2. Consider again the system from Example 6.3.1

$$y[k] - 2y[k-1] = u[k], \qquad y[-1] = 1, u[k] = k.$$

Take z-transforms of everything in sight but this time don't set the initial conditions to zero

$$\begin{aligned} Y[z] - 2\left(z^{-1}Y[z] + y[-1]\right) &= U[z] \\ \Rightarrow\ Y[z] &= \underbrace{\frac{2z}{z-2}y[-1]}_{\text{natural response}} + \underbrace{\frac{z}{z-2}U[z]}_{\text{forced response}} \\ &= 2\frac{z}{z-2} + \frac{z}{z-2}U[z]. \end{aligned}$$

Use the expression for $U[z]$ to get

$$Y[z] = 2\frac{z}{z-2} + \frac{z}{z-2}\frac{z}{(z-1)^2}.$$

Write out the partial fraction expansion for $Y[z]/z$ as

$$\frac{Y[z]}{z} = \frac{2}{z-2} + \frac{c_1}{z-2} + \frac{c_2}{z-1} + \frac{c_3}{(z-1)^2}$$

and solve for $c_1 = 2$, $c_2 = -2$, $c_3 = -1$. We get that

$$Y[z] = \frac{4z}{z-2} - \frac{2z}{z-1} - \frac{z}{(z-1)^2}$$

and hence

$$y[k] = 4(2)^k - 2(1)^k - k, \qquad k \geq 0.$$

▲

6.4 Transfer functions

Linear time-invariant systems, and only linear time-variant systems, have transfer function. Consider the equation

$$y[k] + a_1 y[k-1] + \cdots + a_n y[k-n] = b_0 u[k] + b_1 u[k-1] + \cdots + b_m u[k-m]$$

Take z-transforms assuming $u[k]$, $y[k]$ equal zero for $k < 0$:

$$Y[z] + a_1 z^{-1}Y[z] + \cdots + a_n z^{-n}Y[z] = b_0 U[z] + b_1 z^{-1}U[z] + \cdots + b_m z^{-m}U[z].$$

Thus

$$\frac{Y[z]}{U[z]} = G[z], \qquad G[z] := \frac{b_0 + b_1 z^{-1} + \cdots + b_m z^{-m}}{1 + a_1 z^{-1} + \cdots + a_n z^{-n}}.$$

The function $G[z]$ is the transfer function; its inverse transform, $g[k] = \mathscr{Z}^{-1}\{G[z]\}$, is the impulse response function; and in the time domain, $y[k]$ equals the convolution of $g[k]$ and $u[k]$

$$y[k] = \sum_{m=0}^{k} g[k-m]u[m].$$

The terms in Definition 2.2.1 apply without any modification to DT transfer functions.

Example 6.4.1. (Improper TF) We only study proper DT transfer functions because improper TFs correspond to non-causal systems. Consider the system

$$y[k] = u[k+1]$$

which is non-causal because the current output depends on future inputs. Taking z-transforms under the assumption of zero initial conditions yields

$$Y[z] = zU[z].$$

The corresponding TF is $G[z] = z$ which is improper. ▲

Example 6.4.2. Here's an application of the final-value theorem: Consider a system with transfer function

$$\frac{Y[z]}{U[z]} = G[z] = \frac{z+0.35}{(z-0.5)(z+0.5)(z-0.1)}.$$

Since all the poles of $G[z]$ are in $|z| < 1$, if $u[k] = \mathbf{1}[k]$, then $y[k]$ converges as $k \to \infty$ to

$$\lim_{z\to 1}(z-1)G[z]U[z] = \lim_{z\to 1} zG[z] = G[1] = 2.$$

▲

To conclude this section we mention that a rational TF can be represented in two forms such as

$$G[z] = \frac{8z^3 - 24z - 16}{2z^5 + 20z^4 + 98z^3 + 268z^2 + 276z} = \frac{8z^{-2} - 24z^{-3} - 16z^{-5}}{2 + 20z^{-1} + 98z^{-2} + 268z^{-3} + 276z^{-4}}.$$

The first is said to be in **positive power form** while the latter in **negative power form**. For CT systems we always use the positive power form. If we use the positive power form in the DT setting, then many of our results from the CT setting can be directly applied and *vide versa*. Thus, with the exception of the next subsection, we exclusively use the positive power form.

6.4.1 Rational transfer functions and difference equations

Using the backward shift property of z-transforms and assuming that $x[k] = 0$ when $k < 0$, we have $\mathscr{Z}\{x[k-m]\} = z^{-m}X[z]$. Using this fact we can readily obtain a difference equation from a rational transfer function and *vice versa*.

Example 6.4.3. Consider the TF model

$$\frac{Y[z]}{U[z]} = \frac{z^2 + 2z}{2z^3 + z^2 - 0.4z + 0.8} = \frac{z^{-1} + 2z^{-2}}{2 + z^{-1} - 0.4z^{-2} + 0.8z^{-3}}.$$

Re-write this as

$$\left(2 + z^{-1} - 0.4z^{-2} + 0.8z^{-3}\right) Y[z] = \left(z^{-1} + 2z^{-2}\right) U[z]$$

and then take its inverse z-transform

$$2y[k] + y[k-1] - 0.4y[k-2] + 0.8y[k-3] = u[k-1] + 2u[k-2].$$

This is a third-order linear LTI difference equation. Conversely, given an LTI difference equation, we can readily obtain its transfer function using the backward shift property. ▲

The reason we used the negative power form in the above example is because $\mathscr{Z}\{x[k+1]\} = zX[z] - x[0]$ and we can't assume that $x[0]$ is zero. Thus the use of the positive-power form to go from a TF to a difference equation will be more complicated. However the final result will be the same.

6.5 State-space models

Linear control theory is based on transfer function models and state-space models. We now focus on the latter. Discrete-time systems that are linear, time-invariant, causal, finite-dimensional, and having proper transfer functions have state models,

$$x[k+1] = Ax[k] + Bu[k] \tag{6.5a}$$
$$y[k] = Cx[k] + Du[k]. \tag{6.5b}$$

Here x, u and y are vectors that depend on time k and A, B, C, D are real constant matrices.

6.5.1 Deriving state-space models

How to get a state-space model depends on what we have to start with.

Example 6.5.1. (Deriving a state-space model from a difference equation) Suppose that we have the system

$$y[k] - \frac{1}{2}y[k-1] + \frac{3}{2}y[k-2] = \frac{1}{2}u[k-2].$$

First re-write the difference equation in the form

$$y[k+2] - \frac{1}{2}y[k+1] + \frac{3}{2}y[k] = \frac{1}{2}u[k].$$

The natural state vector is

$$x[k] = \begin{bmatrix} x_1[k] \\ x_2[k] \end{bmatrix} := \begin{bmatrix} y[k] \\ y[k+1] \end{bmatrix}.$$

Then

$$\begin{aligned} x_1[k+1] &= x_2[k] \\ x_2[k+1] &= -\frac{3}{2}x_1[k] + \frac{1}{2}x_2[k] + \frac{1}{2}u[k] \\ y[k] &= x_1[k]. \end{aligned}$$

This is a model of the form (6.5) with

$$A = \begin{bmatrix} 0 & 1 \\ -\frac{3}{2} & \frac{1}{2} \end{bmatrix}, \qquad B = \begin{bmatrix} 0 \\ \frac{1}{2} \end{bmatrix}, \qquad C = \begin{bmatrix} 1 & 0 \end{bmatrix}, \qquad D = 0.$$

▲

The technique of Example 6.5.1 extends to any difference equation of the form

$$y[k] + a_1 y[k-1] + \cdots + a_n y[k-n] = b_n u[k-n],$$

or, equivalently, any TF of the form

$$\frac{Y[z]}{U[z]} = \frac{b_n z^{-n}}{1 + a_1 z^{-1} + \cdots + a_{n-1} z^{-(n-1)} + a_n z^{-n}} = \frac{b_n}{z^n + a_1 z^{n-1} + \cdots + a_{n-1} z + a_n}.$$

Exercise 6.2. Find state-space models for (i) $y[k]+y[k-1]+4y[k-2] = u[k-2]$ and (ii) $y[k]+y[k-1]+4y[k-3] = u[k-3]$.

Let's see how to convert a difference equation into a state-space model when the relative degree of the associated transfer function is less than n.

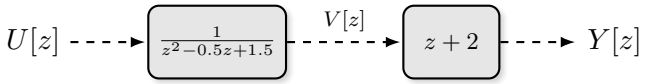
Example 6.5.2. (Deriving a state-space model from a difference equation) Suppose that we have the system

$$y[k] - \frac{1}{2}y[k-1] + \frac{3}{2}y[k-2] = u[k-1] + 2u[k-2].$$

Take z-transforms with no initial conditions to get a TF

$$Y[z] = \frac{z^{-1} + 2z^{-2}}{1 - 0.5z^{-1} + 1.5z^{-2}}U[z] = \frac{z+2}{z^2 - 0.5z + 1.5}U[z].$$

Introduce an intermediate signal $v[k]$:



$$Y[z] = (z+2)V[z], \qquad V[z] \coloneqq \frac{1}{z^2 - 0.5z + 1.5}U[z].$$

Then going back to the time domain we have

$$\begin{aligned} v[k+2] - 0.5v[k+1] + 1.5v[k] &= u[k] \\ y[k] &= v[k+1] + 2v[k]. \end{aligned}$$

Taking $x[k] = (x_1[k], x_2[k]) \coloneqq (v[k], v[k+1])$ as our state vector we get

$$\begin{aligned} x_1[k+1] &= x_2[k] \\ x_2[k+1] &= -1.5x_1[k] + 0.5x_2[k] + u[k] \\ y[k] &= 2x_1[k] + x_2[k]. \end{aligned}$$

This is an LTI state-space model (6.5) in which

$$A = \begin{bmatrix} 0 & 1 \\ -1.5 & 0.5 \end{bmatrix}, \qquad B = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \qquad C = \begin{bmatrix} 2 & 1 \end{bmatrix}, \qquad D = 0.$$

▲

The technique of Example 6.5.2 extends to the difference equation

$$y[k] + a_1 y[n-1] + \cdots + a_n y[k-n] = b_1 u[k-1] + \cdots + b_m u[k-m]. \tag{6.6}$$

where a_i, b_i are real constants. Without loss of generality, by possibly adding terms whose coefficients are zero, let's assume that $n = m$. Then the transfer function generated by the above difference equation is

$$G[z] = \frac{b_1 z^{n-1} + b_2 z^{n-2} + \cdots + z b_{n-1} + b_n}{z^n + a_1 z^{n-1} + \cdots + a_{n-1} z + a_n}. \tag{6.7}$$

In summary, using the technique from Example 6.5.2, if we are given either the difference equation (6.6) (with $n = m$) or the transfer function (6.7), then we can immediately write down a state-space model (6.5) where

$$A = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 0 & 1 \\ -a_n & -a_{n-1} & -a_{n-2} & \cdots & -a_2 & -a_1 \end{bmatrix}, \qquad B = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix}, \qquad C = \begin{bmatrix} b_n & \cdots & \cdots & b_1 \end{bmatrix}, \qquad D = 0. \tag{6.8}$$

Example 6.5.3. (TF to state-space model) Consider a system with TF

$$\frac{Y[z]}{U[z]} = \frac{z^2 + 2z}{z^3 + z^2 - 0.4z + 0.8}.$$

Using the technique derived above, a state-space model for this system is

$$\begin{aligned} x[k+1] &= \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -0.8 & 0.4 & 1 \end{bmatrix} x[k] + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} u[k] \\ y[k] &= \begin{bmatrix} 0 & 2 & 1 \end{bmatrix} x[k]. \end{aligned}$$

▲

The transfer function (6.7) is strictly proper (equivalently (6.6) doesn't depend on the current value of the input u) which is why the D matrix equals zero in our state-space model. If the TF is not strictly proper, then $D \neq 0$ and system is said to have a **feedforward** component.

Example 6.5.4. (System with feedforward component) Consider a system with model

$$y[k] + 2y[k-1] - 7y[k-2] = 3u[k] - u[k-1] + 2u[k-2] - 6u[k-3].$$

The associated TF is

$$\frac{Y[z]}{U[z]} = \frac{3z^3 - z^2 + 2z - 6}{z^3 + 2z^2 - 7z}.$$

Divide the numerator polynomial by the denominator to re-write the TF in the form

$$\frac{Y[z]}{U[z]} = \frac{3z^3 - z^2 + 2z - 6}{z^3 + 2z^2 - 7z} = 3 + \frac{-7z^2 + 23z - 6}{z^3 + 2z^2 - 7z}.$$

Using (6.8) we can read of the coefficients in the state-space model to obtain

$$\begin{aligned} x[k+1] &= \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 7 & -2 \end{bmatrix} x[k] + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} u[k] \\ y[k] &= \begin{bmatrix} -6 & 23 & -7 \end{bmatrix} x[k] + 3u[k]. \end{aligned}$$

▲

Remark 6.5.1. The method for obtaining a state-space model derived in this section also applies to continuous-time systems. Given the differential equation

$$y^{(n)} + a_{n-1}y^{(n-1)} + \cdots + a_1\dot{y} + a_0 y = b_0 u + b_1 \dot{u} + \cdots b_{n-1}u^{(n-1)},$$

or, equivalently, a TF of the form

$$\frac{Y(s)}{U(s)} = \frac{b_{n-1}s^{n-1} + \cdots + b_1 s + b_0}{s^n + a_{n-1}s^{n-1} + \cdots + a_1 s + a_0},$$

then a state-space realization of this system is given by

$$\begin{aligned} \dot{x} &= Ax + Bu \\ y &= Cx + Du \end{aligned}$$

where

$$A = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ldots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 0 & 1 \\ -a_0 & -a_1 & -a_2 & \cdots & -a_{n-2} & -a_{n-1} \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix}, \quad C = \begin{bmatrix} b_0 & b_1 & \cdots & b_{n-2} & b_{n-1} \end{bmatrix}, \quad D = 0.$$

♦

6.5.2 From state-space to transfer function

Let's find the transfer function for the LTI state model[2]

$$\begin{aligned} x^+ &= Ax + Bu \\ y &= Cx + Du. \end{aligned}$$

Take z-transforms with zero initial conditions[3]

$$\begin{aligned} zX[z] &= AX[z] + BU[z] \\ Y[z] &= CX[z] + DU[z]. \end{aligned}$$

Eliminate $X[z]$

$$\begin{aligned} &(zI - A)X[z] = BU[z] \\ &\Rightarrow X[z] = (zI - A)^{-1}BU[z] \\ &\Rightarrow Y[z] = \underbrace{\big(C(zI - A)^{-1}B + D\big)}_{\text{transfer function } G[z]} U[z]. \end{aligned}$$

The TF of a discrete-time state-space model is $G[z] = C(zI - A)^{-1}B + D$. We again we use the convenient notation

$$\left[\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right][z] := C\,(zI - A)^{-1}\,B + D. \tag{6.9}$$

Recall, from Section 5.2.1, the TF of a continuous-time state-space model is $G(s) = C(sI - A)^{-1}B + D$. Since these two functions are identical (only the arguments have different symbols), Remark 5.2.2 also applies to DT systems.

Example 6.5.5. Let's find the TF model of the system

$$\begin{aligned} x[k+1] &= \begin{bmatrix} 1 & 1 \\ -1 & 1 \end{bmatrix} x[k] + \begin{bmatrix} 1 \\ 1 \end{bmatrix} u[k] \\ y[k] &= \begin{bmatrix} 1 & 2 \end{bmatrix} x[k] + u[k]. \end{aligned}$$

Performing the calculations derived above we obtain

$$\begin{aligned} \frac{Y[z]}{U[z]} &= C(zI - A)^{-1}B + D \\ &= \begin{bmatrix} 1 & 2 \end{bmatrix} \begin{bmatrix} z-1 & -1 \\ 1 & z-1 \end{bmatrix}^{-1} \begin{bmatrix} 1 \\ 1 \end{bmatrix} + 1 \\ &= \begin{bmatrix} 1 & 2 \end{bmatrix} \frac{1}{(z-1)^2 + 1} \begin{bmatrix} z-1 & 1 \\ -1 & z-1 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \end{bmatrix} + 1 \\ &= \begin{bmatrix} 1 & 2 \end{bmatrix} \frac{1}{z^2 - 2z + 2} \begin{bmatrix} z \\ z-2 \end{bmatrix} + 1 \\ &= \frac{3z - 4}{z^2 - 2z + 2} + 1 \\ &= \frac{z^2 + z - 2}{z^2 - 2z + 2}. \end{aligned}$$

▲

Remark 6.5.2. The MATLAB commands `tf2ss` and `ss2tf` can be used, respectively, to convert TF models to state-space models and *vice versa*. ♦

[2] Just as we use a dot to denote derivative in continuous time, it's convenient to use a plus to denote time advance in discrete time. Thus $x^+[k] = x[k+1]$. Then we can drop the k in our state-space equations.

[3] Here $X[z]$ denotes the component wise ZT of the vector $x[k]$. That is, if $x[k] = (x_1[k], \ldots, x_n[k])$ then $X[z] = (X_1[z], \ldots, X_n[z])$.

6.6 Stability

We study two different notions of stability. One for state models and one for transfer function models.

6.6.1 Stability of state-space models

The notion of stability we study in this section concerns state-space models. The question is: For the system

$$x[k+1] = Ax[k], \qquad x[0] \in \mathbb{R}^n, \tag{6.10}$$

will $x[k]$ converge to 0 as k goes to infinity, for every $x[0]$?

Definition 6.6.1. System (6.10) is **asymptotically stable** if $x[k] \to 0$ as $k \to \infty$ for every initial condition.

To develop a test for asymptotic stability, we must first understand how $x[k]$ evolves over time. If the initial state is $x[0]$, then under the dynamics (6.10) we get $x[1] = Ax[0]$. At the next time-step: $x[2] = Ax[1] = A^2x[0]$. Similarly $x[3] = Ax[2] = A^3x[0]$ etc.

Theorem 6.6.2. *The unique solution to $x[k+1] = Ax[k]$, $x[0] = x_0$, is $x[k] = A^k x_0$.*

The matrix A^k is called the **transition matrix**: It maps the state at time 0 to the state at time k. This means that asymptotic stability is equivalent to the condition that $A^k \to 0$ as $k \to \infty$.

Take the z-transform of the difference equation $x[k+1] = Ax[k]$ without setting initial conditions to zero

$$\begin{aligned} zX[z] - zx[0] &= AX[z] \\ \Rightarrow\ X[z] &= z(zI - A)^{-1}x[0] = (I - z^{-1}A)^{-1}x[0]. \end{aligned}$$

Comparing this expression to the time-domain solution from Theorem 6.6.2 we have that A^k and $(I - z^{-1}A)^{-1}$ are z-transform pairs

$$\mathscr{Z}\left\{A^k\right\} = (I - z^{-1}A)^{-1}.$$

This fact can be used to compute A^k by hand for small problems ($n \leq 3$).

Example 6.6.1. (Computing the transition matrix using z-transforms) Calculate A^k for the matrix

$$A = \begin{bmatrix} 1 & 1 \\ -1 & 1 \end{bmatrix}.$$

From above we know that $A^k = \mathscr{Z}^{-1}\{z\,(zI - A)^{-1}\}$. We proceed to perform the necessary calculations:

$$zI - A = \begin{bmatrix} z-1 & -1 \\ 1 & z-1 \end{bmatrix}$$

inverting a 2×2 matrix is easy

$$(zI - A)^{-1} = \frac{1}{(z-1)^2 + 1}\begin{bmatrix} z-1 & 1 \\ -1 & z-1 \end{bmatrix} = \frac{1}{z^2 - 2z + 2}\begin{bmatrix} z-1 & 1 \\ -1 & z-1 \end{bmatrix}.$$

Find the inverse z-transform of each entry in $z(zI - A)^{-1}$ to obtain

$$A^k = \sqrt{2}^k \begin{bmatrix} \cos\left(\frac{\pi}{4}k\right) & \sin\left(\frac{\pi}{4}k\right) \\ -\sin\left(\frac{\pi}{4}k\right) & \cos\left(\frac{\pi}{4}k\right) \end{bmatrix}.$$

Since A^k does not approach zero as k gets large, the system $x[k+1] = Ax[k]$ is not asymptotically stable. ▲

If A is a diagonal matrix, then it is easy to compute A^k.

Example 6.6.2. (Diagonal Matrix)

$$A = \begin{bmatrix} 3 & 0 \\ 0 & 1 \end{bmatrix}, \qquad A^k = \begin{bmatrix} 3^k & 0 \\ 0 & 1 \end{bmatrix}.$$

When A is a diagonal matrix then A^k is obtained by raising each diagonal element of A to the power k. ▲

Another way to compute A^k is by using eigenvalues and eigenvectors. We won't cover the general theory and instead present an example.

Example 6.6.3. (Computing the transition matrix using diagonalization)

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

The MATLAB command `[V, D] = eigs(A)` produces

$$V = \begin{bmatrix} 1 & -1 \\ -1 & 2 \end{bmatrix}, \qquad D = \begin{bmatrix} -1 & 0 \\ 0 & -2 \end{bmatrix}.$$

The eigenvalues of A are the diagonal elements of D. The columns of V are the corresponding eigenvectors. So, for example,

$$A \begin{bmatrix} 1 \\ -1 \end{bmatrix} = -1 \begin{bmatrix} 1 \\ -1 \end{bmatrix}.$$

It follows that $AV = VD$ (verify this) and therefore that $A^k = VD^kV^{-1}$ (prove this). The nice thing about D^k is that it is easy to compute since D is a diagonal matrix (cf. Example 6.6.2). In this case

$$D^k = \begin{bmatrix} (-1)^k & 0 \\ 0 & (-2)^k \end{bmatrix}.$$

Then

$$A^k = VD^kV^{-1} = \begin{bmatrix} 1 & -1 \\ -1 & 2 \end{bmatrix} \begin{bmatrix} (-1)^k & 0 \\ 0 & (-2)^k \end{bmatrix} \begin{bmatrix} 2 & 1 \\ 1 & 1 \end{bmatrix} = \begin{bmatrix} 2(-1)^k - (-2)^k & (-1)^k - (-2)^k \\ -2(-1)^k + 2(-2)^k & -(-1)^k + 2(-2)^k \end{bmatrix}.$$

▲

Remark 6.6.3. The approach in Example 6.6.3 works when A has n linearly independent eigenvectors so that A is diagonalizable. Otherwise the theory is more complicated and requires the **Jordan form** of the matrix A. ♦

Fortunately, the next result means that we do not need to compute A^k to determine whether or not (6.10) is asymptotically stable.

Proposition 6.6.4. *$A^k \to 0$ as $k \to \infty$ if, and only if, every eigenvalue of A has magnitude less than 1.*

Proof. If A has n linearly independent eigenvectors, then it can be diagonalized by a similarity transform so that

$$A = VDV^{-1}$$

where D is a diagonal matrix whose diagonal entries are the eigenvalues of A. Then

$$\begin{aligned} A^k \to 0 &\iff VD^kV^{-1} \to 0 \\ &\iff D^k \to 0 \\ &\iff \lambda_i^k \to 0, \ \forall i \\ &\iff |\lambda_i| < 0, \ \forall i. \end{aligned}$$

The proof in the general case is more complicated but not more enlightening. ■

Let's say that the matrix A is stable if its eigenvalues all have magnitude less than one. Then the system $x[k+1] = Ax[k]$ is asymptotically if, and only if, A is stable.

Example 6.6.4. Re-visiting the matrix from Example 6.6.1

$$A = \begin{bmatrix} 1 & 1 \\ -1 & 1 \end{bmatrix}.$$

The eigenvalues of A are found by solving $\det(zI - A) = 0$ to yield $z = 1 \pm j$. Both eigenvalues have magnitude $\sqrt{2}$ and therefore by Proposition 6.6.4 the matrix is not asymptotically stable. If instead

$$A = \begin{bmatrix} -0.5 & 10 \\ 0 & 0.7 \end{bmatrix},$$

then the eigenvalues are -0.5 and 0.7. By Proposition 6.6.4 the matrix is asymptotically stable. ▲

6.6.2 Bounded-input bounded-output stability

Consider an LTI system with input $u[k]$ and output $y[k]$ and TF $G[z]$. The natural concept of stability in an input-output setting is that of bounded-input, bounded-output stability. The discussion in this section mirrors Section 2.3 very closely. In particular Definitions 2.3.1 and 2.3.2 are essentially unchanged. The next theorem is the DT counterpart to Theorem 2.3.3.

Theorem 6.6.5. *Assume $G[z]$ is proper and rational. Then the following three statements are equivalent:*

1. *The system is BIBO stable.*
2. *The impulse-response function $g[k]$ is absolutely summable, i.e., $\sum_{k=0}^{\infty} |g[k]| < \infty$.*
3. *Every pole of the transfer function $G[z]$ has magnitude less than one.*

Example 6.6.5. (Moving average) Consider the 5-point moving average from Example 6.1.1. In Exercise 6.1 you found it's impulse response

$$g[k] = \begin{cases} 0.2 & 0 \leq k \leq 4 \\ 0 & \text{otherwise.} \end{cases}$$

The impulse response only has a finite number of non-zero terms so it's clearly absolutely summable. Therefore, by Theorem 6.6.5 the system is BIBO stable which makes good intuitive sense. Equivalently, the corresponding TF from u to y is

$$\frac{Y[z]}{U[z]} = \frac{1}{5} \frac{z^4 + z^3 + z^2 + z^1 + 1}{z^4}.$$

This TF has all its poles at $z = 0$ so that, again by Theorem 6.6.5, the system is BIBO stable. ▲

Example 6.6.6. (FIR systems) Generalizing the previous example, a DT system is defined to be a Finite Impulse Response (F.I.R.) system if its impulse response has a finite number of non-zero terms. By Theorem 6.6.5, *every* FIR system is BIBO stable. ▲

Exercise 6.3. Determine whether or not the savings account model from Example 6.1.2 is BIBO stable.

Example 6.6.7. (Summer) Consider the DT summer

$$y[k] = y[k-1] + u[k].$$

The output at time k is the sum of the previous output and the current input. You can check that the impulse response of this system is $g[k] = \mathbf{1}[k]$. This signal is not absolutely summable so the summer is not BIBO stable. Equivalently, the corresponding TF

$$\frac{Y[z]}{U[z]} = \frac{z}{z-1}$$

has a pole at $z = 1$ and is therefore unstable. ▲

6.6.3 Stability of state-space models and BIBO stability

Consider a single-input single-output system modelled by

$$\begin{aligned} x[k+1] &= Ax[k] + Bu[k] \\ y[k] &= Cx[k] + Du[k]. \end{aligned}$$

or

$$\begin{aligned} Y[z]/U[z] &= G[z] \\ &= C\left(zI - A\right)^{-1} B + D \\ &= \frac{1}{\det\left(zI - A\right)} C \operatorname{adj}\left(zI - A\right) B + D. \end{aligned}$$

From the last expression we see that the poles of $G[z]$ are a *subset* of the eigenvalues of A. Thus

$$\text{asymptotic stability of state-space model} \implies \text{BIBO stability of TF model.}$$

Usually the poles of $G[z]$ are identical to the eigenvalues of A, that is, the two polynomials

$$\det\left(zI - A\right), \qquad C \operatorname{adj}\left(zI - A\right) B + D \det\left(zI - A\right)$$

have no common roots, i.e., they're coprime. In these cases, the two stability concepts are equivalent.

Example 6.6.8. Consider the single-input single-output system modelled by

$$\begin{aligned} x[k+1] &= \begin{bmatrix} 0 & 1 \\ 0.25 & 0 \end{bmatrix} x[k] + \begin{bmatrix} 2 \\ 0 \end{bmatrix} u[k] \\ y[k] &= \begin{bmatrix} 1 & 0 \end{bmatrix} x[k] + u[k]. \end{aligned}$$

The eigenvalues of the A matrix are found by solving $\det\left(zI - A\right) = z^2 - 0.25 = 0$ to get $z = \pm 0.5$. By Proposition 6.6.4 this system is asymptotically stable and therefore we can immediately conclude that it is also BIBO stable. To verity this, the system's transfer function is

$$G[z] = C(zI - A)^{-1}B + D = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} \frac{z}{z^2-0.25} & \frac{1}{z^2-0.25} \\ \frac{0.25}{z^2-0.25} & \frac{z}{z^2-0.25} \end{bmatrix} \begin{bmatrix} 2 \\ 0 \end{bmatrix} + 1 = \frac{z^2 + 2z - 0.25}{z^2 - 0.25}.$$

The poles of G are $z = \pm 0.5$ and by Theorem 6.6.5 the system is BIBO stable. In this example the poles of G equal the eigenvalues of A because the polynomials

$$\det(zI - A) = z^2 - 0.25$$

and

$$C \operatorname{adj}\left(zI - A\right) B + D \det\left(zI - A\right) = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} z & 1 \\ 0.25 & z \end{bmatrix} \begin{bmatrix} 2 \\ 0 \end{bmatrix} + z^2 - 0.25 = z^2 + 2z - 0.25$$

are coprime. ▲

Exercise 6.4. Show that the two notions of stability presented are *not* equivalent for a system with

$$A = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \qquad B = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \qquad C = \begin{bmatrix} 1 & -1 \end{bmatrix}, \qquad D = 0.$$

6.7 Stability of feedback systems

We turn to feedback systems. This section closely mirrors Section 2.4. The only difference is that for discrete-time systems the stable region of the complex plane is the interior of the unit disk. For this reason the discussion is brief. We make the following standing assumption throughout this section to ensure that the feedback system

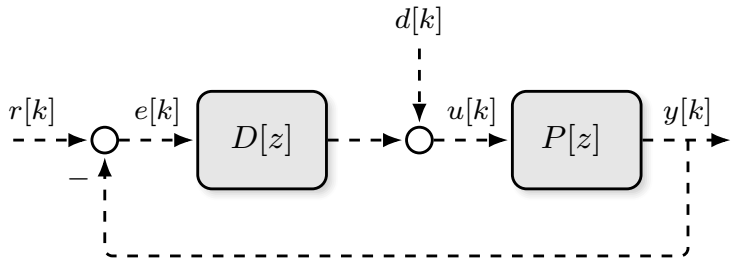


Figure 6.3: Unity feedback system.

in Figure 6.3 is well-posed.

Assumption 6.7.1. The plant and controller TFs are rational, $D[z]$ is proper, and $P[z]$ is strictly proper. ◀

6.7.1 Internal stability

For this concept we set the reference input $r[k]$ and the disturbance input $d[k]$ to zero, i.e., $r[k] = d[k] = 0$. We bring in state-space models for the plant and controller as in Figure 6.4. The closed-loop equations are

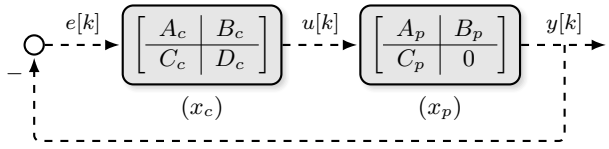


Figure 6.4: Internal stability.

$$\begin{aligned} x_p^+ &= A_p x_p + B_p u \\ x_c^+ &= A_c x_c + B_c e \\ u &= C_c x_c + D_c e \\ e &= -C_p x_p. \end{aligned}$$

By Assumption 6.7.1 the plant is strictly proper ($D_p = 0$). Plugging the expressions for u and e into the state equations gives (verify!)

$$\begin{aligned} x_p^+ &= (A_p - B_p D_c C_p) x_p + B_p C_c x_c \\ x_c^+ &= -B_c C_p x_p + A_c x_c. \end{aligned}$$

Let the closed-loop state be $x_{cl} \coloneqq (x_p, x_c)$ so that

$$x_{cl}^+ = A_{cl}x_{cl} \coloneqq \begin{bmatrix} A_p - B_pD_cC_p & B_pC_c \\ -B_cC_p & A_c \end{bmatrix} \begin{bmatrix} x_p \\ x_c \end{bmatrix}. \tag{6.11}$$

Definition 6.7.2. The feedback system in Figure 6.3 is **internally stable** if the state-space model (6.11) is asymptotically stable.

The concept of internal stability means that with no exogenous inputs applied ($r = d = 0$), the internal states $x_p[k]$, $x_c[k]$ decay to zero for every initial state of the plant $x_p[0]$ and the controller $x_c[0]$. By Proposition 6.6.4 the feedback system is internally stable if, and only if, all the eigenvalues of the matrix A_{cl} have magnitude less than one.

Example 6.7.1. (Proportional-integral control) Consider the unstable plant

$$P[z] = \frac{1}{z+2}$$

with a PI controller

$$D[z] = K_p + K_i\frac{1}{z-1}.$$

Find gains K_i, K_p such that the closed-loop system is internally stable. Bring in state-space models (Section 6.5.1). First the plant

$$\begin{aligned} x_p^+ &= -2x_p + u \\ y &= x_p, \end{aligned}$$

then the controller

$$\begin{aligned} x_c^+ &= x_c + e \\ u &= K_ix_c + K_pe. \end{aligned}$$

The closed-loop matrix (6.11) in this example equals

$$A_{cl} = \begin{bmatrix} -2 - K_p & K_i \\ -1 & 1 \end{bmatrix}.$$

The eigenvalues of A_{cl} are the roots of

$$\det(zI - A_{cl}) = z^2 + (1 + K_p)z + K_i - K_p - 2.$$

If we pick $K_p = -1$ and $K_i = 1$, then the eigenvalues of A_{cl} are $\{0, 0\}$ and the closed-loop system is internally stable. The controller is

$$D[z] = -1 + \frac{1}{z-1} = \frac{2-z}{z-1}.$$

▲

6.7.2 Input-output stability

This concept is the same as in Section 2.4.

Definition 6.7.3. The feedback system in Figure 6.3 is **input-output stable** provided e, u, and y are bounded signals whenever r and d are bounded signals; briefly, the system from (r, d) to (e, u, y) is BIBO stable.

Just as in continuous-time, there are six closed-loop transfer functions from (r, d) to (e, u, u):

$$R \text{ to } E : \frac{1}{1+PD}, \qquad R \text{ to } U : \frac{D}{1+PD}, \qquad R \text{ to } Y : \frac{PD}{1+PD},$$
$$D \text{ to } E : \frac{-P}{1+PD}, \qquad D \text{ to } U : \frac{1}{1+PD}, \qquad D \text{ to } Y : \frac{P}{1+PD}.$$

Write

$$P = \frac{N_p}{D_p}, \qquad D = \frac{N_c}{D_c}.$$

The least common multiple of the denominators of these six transfer functions is

$$\pi[z] = D_p[z]D_c[z] + N_p[z]N_c[z]$$

which we again call the characteristic polynomial of the feedback system in Figure 6.3.

Theorem 6.7.4. *The feedback system in Figure 6.3 is input-output stable if, and only if, its characteristic polynomial has no roots with magnitude greater than or equal to one.*

Example 6.7.2. Recall the plant and PI controller from Example 6.7.1

$$P[z] = \frac{1}{z+2}, \qquad D[z] = \frac{2-z}{z-1}.$$

The ch.p. of this system is

$$\pi[z] = D_p[z]D_c[z] + N_p[z]N_c[z] = (z+2)(z-1) + 2 - z = z^2.$$

Since π has all its roots inside the unit disk, the feedback system is I/O stable. ▲

6.7.3 Internal stability and input-output stability

We have given two definitions for what it means for the system in Figure 6.3 to be stable. Internal stability (Definition 6.7.2) dealt with the closed-loop state-space model whereas input-output stability (Definition 6.7.3) dealt with the various transfer functions in the loop. It can be shown that the roots of the characteristic polynomial $\pi[z]$ are a subset of the eigenvalues of A_{cl}, i.e., the roots of $\pi[z]$ are a subset of the roots of $\det(zI - A_{cl})$. Hence

$$\text{internal stability} \implies \text{input-output stability.}$$

Usually the roots of $\pi[z]$ are identical to the eigenvalues of A_{cl} (cf. Example 6.7.2). In these cases, the two stability concepts are equivalent. In this course internal stability and input-output stability of a feedback system are assumed to be equivalent and hence when we talk about feedback stability there is no ambiguity. This is reasonable for most practical applications.

6.8 Identifying polynomials with roots in the open unit disk

In practice, one often checks stability using numerical software to compute the roots of a polynomial. However, if some of the coefficients of the polynomial are not fixed, e.g., the sampling period, then we can't solve for the roots numerically. In this section we discuss two methods for checking if the roots of a polynomial all have magnitude less than one without actually finding them.

Consider an nth order polynomial $\pi \in \mathbb{R}[z]$ with real coefficients

$$\pi[z] = z^n + a_1 z^{n-1} + \cdots + a_{n-1}s + a_n, \ a_i \in \mathbb{R}, \ n \neq 0. \tag{6.12}$$

Definition 6.8.1. A polynomial $\pi \in \mathbb{R}[z]$ is **Schur** if all its roots have magnitude less than one.

6.8.1 Necessary conditions for a polynomial to be Schur

Before presenting some algebraic tests for a polynomial to be Schur, we list some necessary, but not sufficient, conditions for (6.12) to be Schur.

(i) $\pi[1] > 0$

Proof. Let $z \to \infty$ along the real axis. Then $\pi[z] \to z^n \to \infty$. If $\pi[1] < 0$ then there is some $z > 1$ along the real axis at which $\pi[z] = 0$. ■

(ii) $(-1)^n \pi[-1] > 0$. The proof is the same as above but now take the limit as $z \to -\infty$ along the negative real axis.

(iii) $|a_n| < 1$.

Proof. Write

$$\pi[z] = z^n + a_1 z^{n-1} + \cdots + a_{n-1} z + a_n = \prod_{i=1}^{n} (z - \xi_i)$$

where $\xi_i \in \mathbb{C}$ are the roots of $\pi[z]$. Then

$$\pi[0] = a_n = \prod_{i=1}^{n} (-\xi_i).$$

If $|a_n| \geq 1$ then at least one of the ξ_i must have magnitude greater than or equal to one. ■

Example 6.8.1.

$$\begin{aligned} &z^4 + 0.3z^3 - z^2 - 0.9 && \text{(a bad root)} \\ &z^3 + 0.1z + 1.1 && \text{(a bad root)} \\ &z^3 + 0.1z^2 - 0.6z + 0.1 && \text{(don't know).} \end{aligned}$$

▲

6.8.2 The Routh-Hurwitz criterion

The Routh-Hurwitz criterion is used for continuous-time systems to determine if a polynomial has all its roots in $\mathbb{C}^-$. Consider a polynomial

$$\pi(s) = s^n + a_{n-1} s^{n-1} + \cdots + a_1 s + a_0, \ a_i \in \mathbb{R}, \ n \neq 0.$$

We construct Table 6.2. In Table 6.2 the first two rows are populated using the coefficients of π. The third row is computed from the first two

$$r_{2,0} = \frac{a_{n-1}a_{n-2} - a_{n-3}}{a_{n-1}}, \quad r_{2,1} = \frac{a_{n-1}a_{n-4} - a_{n-5}}{a_{n-1}}, \quad r_{2,2} = \frac{a_{n-1}a_{n-6} - a_{n-7}}{a_{n-1}}, \ldots$$

The remaining rows are computed recursively using the same pattern

$$r_{i,j} = \frac{r_{i-1,0} r_{i-2,j+1} - r_{i-2,0} r_{i-1,j+1}}{r_{i-1,0}}, \quad i \in \{2, \ldots, n\}, \quad j \in \left\{0, \ldots, \left\lfloor \frac{n-i}{2} \right\rfloor \right\}. \tag{6.13}$$

Table 6.2: Routh array.

$\mathbf{s^n}$	$r_{0,0} = 1$	$r_{0,1} = a_{n-2}$	$r_{0,2} = a_{n-4}$	$r_{0,3} = a_{n-6}$	$\cdots$
$\mathbf{s^{n-1}}$	$r_{1,0} = a_{n-1}$	$r_{1,1} = a_{n-3}$	$r_{1,2} = a_{n-5}$	$r_{1,3} = a_{n-7}$	$\cdots$
$\mathbf{s^{n-2}}$	$r_{2,0}$	$r_{2,1}$	$r_{2,2}$	$r_{2,3}$	$\cdots$
$\mathbf{s^{n-3}}$	$r_{3,0}$	$r_{3,1}$	$r_{3,2}$	$r_{3,3}$	$\cdots$
$\vdots$	$\vdots$	$\vdots$	$\vdots$		
$\mathbf{s^2}$	$r_{n-2,0}$	$r_{n-2,1}$	0	$\cdots$	
$\mathbf{s^1}$	$r_{n-1,0}$	0	0	$\cdots$	
$\mathbf{s^0}$	$r_{n,0}$	0	0	$\cdots$	

The construction terminates if we end up with a zero in the first column. There are at most $n+1$ rows in the final table.

Theorem 6.8.2 (Routh-Hurwitz Stability Criterion)**.** *Consider a polynomial $\pi \in \mathbb{R}[s]$ given by (6.12) and its associated Routh array in Table 6.2. The number of roots of π with real part greater than or equal to zero equals the number of sign changes in the first column of the array.*

Example 6.8.2.

$$\pi(s) = a_2 s^2 + a_1 s + a_0, \qquad a_2 \neq 0.$$

$\mathbf{s^2}$	a_2	a_0	0
$\mathbf{s^1}$	a_1	0	0
$\mathbf{s^0}$	$\frac{a_1 a_0}{a_1} = a_0$	0	0

We conclude that all the roots of π are in $\mathbb{C}^-$ if, and only if, a_0, a_1, a_2 all have the same sign. ▲

If we apply the Routh-Hurwitz criteria to the ch.p. of a discrete-time system then we do not get any useful information. Instead we use the bilinear transformation from Chapter 4

$$s = \frac{2}{T}\frac{z-1}{z+1}, \qquad z = \frac{1+\frac{T}{2}s}{1-\frac{T}{2}s}.$$

The bilinear transformation preserves stability for any sampling period so choose $T = 2$ to simplify calculations

$$z = \frac{1+v}{1-v}.$$

Then $v \in \mathbb{C}^-$ if, and only if, $|z| < 1$. Here we use the symbol $v \in \mathbb{C}$ instead of $s \in \mathbb{C}$ to make it clear that we are not going back to continuous-time, rather we are using the mathematical properties of the bilinear transformation to be able to apply the Routh-Hurwitz stability test. The procedure is the following

1. Given a polynomial (6.12), form the polynomial

$$\pi[z]\big|_{z=\frac{1+v}{1-v}} = \pi\left[\frac{1+v}{1-v}\right].$$

2. Multiply the polynomial obtained in step (1) by $(1-v)^n$ to obtain the nth degree polynomial

$$\hat{\pi}(v) := (1-v)^n \pi\left[\frac{1+v}{1-v}\right].$$

3. The polynomial $\pi[z]$ is Schur if, and only if, the roots of $\hat{\pi}(v)$ all have negative real part and the degree of $\hat{\pi}$ equals the degree of π.

Example 6.8.3. Given the second order polynomial $\pi[z] = z^2 + a_0 z + a_1$ we form

$$\begin{aligned}\hat{\pi}(v) &= (1-v)^n \pi\left[\frac{1+v}{1-v}\right] \\ &= (1-v)^2 + a_0(1-v)(1+v) + a_1(1+v)^2 \\ &= (1-a_0+a_1)v^2 + (2-2a_1)v + (1+a_0+a_1)\end{aligned}$$

The Routh array for this polynomial is

$$\begin{array}{c|cc} \mathbf{v^2} & 1-a_0+a_1 & 1+a_0+a_1 \\ \mathbf{v^1} & 2-2a_1 & 0 \\ \mathbf{v^0} & 1+a_0+a_1 & 0. \end{array}$$

By the Routh-Hurwitz criteria, the roots of $\hat{\pi}(v)$ are in $\mathbb{C}^-$ if, and only if,

$$\operatorname{sgn}(1-a_0+a_1) = \operatorname{sgn}(2-2a_1) = \operatorname{sgn}(1+a_0+a_1).$$

These conditions are equivalent to

$$\begin{aligned}|a_1| &< 1 \\ |a_0| &< |1+a_1|.\end{aligned}$$

The roots of $\pi[z]$ are inside the unit disk, if, and only if a_0, a_1 satisfy the above conditions. ▲

Remark 6.8.3. The real power of using the Routh-Hurwitz criterion is that it can be used to find the range of a parameter, like a controller gain, for which a system is stable. ♦

6.8.3 The Jury test

E.I. Jury [Jury, 1964], building on earlier results, developed discrete-time counterparts of the Routh array and Routh-Hurwitz criterion. His calculation was organized in tabular form and came to be known as **Jury's table** and the **Jury test**.

Consider the polynomial (6.12) which we re-write for convenience

$$\pi[z] = z^n + a_1 z^{n-1} + \cdots + a_{n-1}s + a_n,\ a_i \in \mathbb{R},\ n \neq 0.$$

Using $\pi[z]$, we construct Table 6.3. In Table 6.3 the first two rows are populated using the coefficients of π. The third row is computed from the first two

$$j_{1,0} = \frac{j_{0,0}j_{0,0} - j_{0,n}j_{0,n}}{j_{0,0}}, \quad j_{1,1} = \frac{j_{0,1}j_{0,0} - j_{0,n-1}j_{0,n}}{j_{0,0}}, \quad j_{1,2} = \frac{j_{0,2}j_{0,0} - j_{0,n-2}j_{0,n}}{j_{0,0}}, \ldots$$

The fourth row is the reversal of the third row and the the remaining rows are computed recursively using the same pattern

$$j_{i,k} = \frac{j_{i-1,k}j_{i-1,0} - j_{i-1,n-i+1-k}j_{i-1,n-i+1}}{j_{i-1,0}}, \quad i \geq 1, \quad k \geq 1. \tag{6.14}$$

We continue this pattern along each row until we end up with zeros. We call $j_{i,0}$ the *subsequent leading coefficients*.

Theorem 6.8.4 (Jury test for discrete-time stability)**.** *Consider a polynomial $\pi \in \mathbb{R}[z]$ given by (6.12) and its associated Jury table 6.3. The polynomial is Schur if, and only if, every subsequent leading coefficient is positive. If any subsequent leading coefficient is zero or negative, then the polynomial is not Schur.*

Table 6.3: Jury table.

$\mathbf{z^n}$	$j_{0,0}=1$	$j_{0,1}=a_1$	$\cdots$	$j_{0,n-2}=a_{n-2}$	$j_{0,n-1}=a_{n-1}$	$j_{0,n}=a_n$
	$j_{0,n}=a_n$	$j_{0,n-1}=a_{n-1}$	$\cdots$	$j_{0,2}=a_2$	$j_{0,1}=a_1$	$j_{0,0}=1$
$\mathbf{z^{n-1}}$	$j_{1,0}$	$j_{1,1}$	$\cdots$	$j_{1,n-2}$	$j_{1,n-1}$	0
	$j_{1,n-1}$	$j_{1,n-2}$	$\cdots$	$j_{1,1}$	$j_{1,0}$	
$\mathbf{z^{n-2}}$	$j_{2,0}$	$j_{2,1}$	$\cdots$	$j_{2,n-2}$	0	
	$j_{2,n-2}$	$j_{2,n-3}$	$\cdots$	$j_{2,0}$		
$\vdots$	$\vdots$	$\vdots$	$\vdots$			
$\mathbf{z^1}$	$j_{n-1,0}$	$j_{n-1,1}$	0			
	$j_{n-1,1}$	$j_{n-1,0}$				
$\mathbf{z^0}$	$j_{n,0}$	0				

Example 6.8.4. We'll use the Jury test to find conditions for $\pi[z]=z^2+a_0z+a_1$ to be Schur. The associated Jury table is

$\mathbf{z^2}$	1	a_0	a_1
	a_1	a_0	1
$\mathbf{z^1}$	$1-a_1^2$	$a_0-a_0a_1$	0
	$a_0-a_0a_1$	$1-a_1^2$	
$\mathbf{z^0}$	$\frac{(1-a_1^2)^2-(a_0-a_0a_1)^2}{1-a_1^2}$		

By Theorem 6.8.4, π is Schur if, and only if

$$1-a_1^2>0 \qquad \text{and} \qquad (1-a_1^2)^2-(a_0-a_0a_1)^2>0.$$

The condition $1-a_1^2>0$ holds if and only if $|a_1|<1$. Then

$$\begin{aligned}
&(1-a_1^2)^2-(a_0-a_0a_1)^2>0\\
&\Leftrightarrow ((1+a_1)(1-a_1))^2-a_0^2(1-a_1)^2>0\\
&\Leftrightarrow (1-a_1)^2\left((1+a_1)^2-a_0^2\right)>0\\
&\Leftrightarrow (1+a_1)^2-a_0^2>0\\
&\Leftrightarrow |a_0|<|1+a_1|.
\end{aligned}$$

These are the same conditions from Example 6.8.3 using Routh-Hurwitz and the bilinear transformation. ▲

Remark 6.8.5. The Jury test is easy to implement in software. For example, for the polynomial $\pi[z]=z^3-z^2+2z-0.7$, the MATLAB code below computes the rows of the associated Jury table.

```
j0 = [1 -1 2 -0.7];
j0 = [j0;flip(j0)];
j1 = [det(j0(:, [1 4])) det(j0(:, [2 4])) det(j0(:, [3 4]))];
j1 = [j1;flip(j1)];
j2 = [det(j1(:, [1 3])) det(j1(:, [2 3]))];
j2 = [j2; flip(j2)];
j3 = det(j2);
```

♦

6.9 Frequency response

In Section 2.6 we reviewed the fact that if you apply a sinusoidal input to a BIBO stable continuous-time LTI system, then the steady-state output is also a sinusoid of the same frequency with a possible change in magnitude and phase. We now turn to discrete-time systems and show that the same property holds.

Suppose we have a BIBO stable system

$$y[k] + a_1 y[k-1] + \cdots + a_n y[k-n] = b_0 u[k] + b_1 u[k-1] + \cdots + b_m u[k-m].$$

or equivalently

$$\frac{Y[z]}{U[z]} = G[z] = \frac{b_0 + b_1 z^{-1} + \cdots + b_m z^{-m}}{1 + a_1 z^{-1} + \cdots + a_n z^{-n}}.$$

Let us apply to it the input $u[k] = a\mathrm{e}^{j\theta k}$. Note that $u[k]$ isn't a real-valued sequence, however, its employment will simplify the derivation. The resulting output is

$$Y[z] = G[z]U[z] = G[z]\frac{az}{z - \mathrm{e}^{j\theta}}.$$

Perform a partial fraction expansion of $Y[z]/z$

$$\frac{Y[z]}{z} = G[z]\frac{a}{z - \mathrm{e}^{j\theta}} = \frac{c_1}{z - \mathrm{e}^{j\theta}} + \text{terms due to poles of } G[z]$$

with $c_1 = aG[\mathrm{e}^{j\theta}]$. Thus we have

$$Y[z] = aG[\mathrm{e}^{j\theta}]\frac{z}{z - \mathrm{e}^{j\theta}} + z \times (\text{terms due to poles of } G[z])\,.$$

Since G was assumed to be stable, all the responses due to its poles approach 0 as $k \to \infty$. Thus the steady-state output is

$$\lim_{k\to\infty} y[k] = aG[\mathrm{e}^{j\theta}]\mathrm{e}^{j\theta k}.$$

If we write the complex number $G[\mathrm{e}^{j\theta}]$ in polar form $G[\mathrm{e}^{j\theta}] = \left|G[\mathrm{e}^{j\theta}]\right| \mathrm{e}^{j\measuredangle G[\mathrm{e}^{j\theta}]}$ then

$$\lim_{k\to\infty} y[k] = a\left|G[\mathrm{e}^{j\theta}]\right| \mathrm{e}^{j\left(\theta k + \measuredangle G[\mathrm{e}^{j\theta}]\right)} = a\left|G[\mathrm{e}^{j\theta}]\right| \left(\cos\left(\theta k + \measuredangle G[\mathrm{e}^{j\theta}]\right) + j\sin\left(\theta k + \measuredangle G[\mathrm{e}^{j\theta}]\right)\right).$$

We list some special cases of the above derivation as a theorem.

Theorem 6.9.1. *Assume $G[z]$ is rational, proper, and all its poles have magnitude less than one. Then the steady-state response to the input $u[k] = a\mathrm{e}^{j\theta k}$ is*

$$y[k] = aG[\mathrm{e}^{j\theta}]\mathrm{e}^{j\theta k}. \tag{6.15}$$

In particular

- *The steady-state response to the input $u[k] = a\cos(\theta k)$ is*

$$y[k] = a|G[\mathrm{e}^{j\theta}]|\cos\left(\theta k + \measuredangle G[\mathrm{e}^{j\theta}]\right). \tag{6.16}$$

- *The steady-state response to the input $u[k] = a\sin(\theta k)$ is*

$$y[k] = a|G[\mathrm{e}^{j\theta}]|\sin\left(\theta k + \measuredangle G[\mathrm{e}^{j\theta}]\right). \tag{6.17}$$

- *The steady-state response to the input $u[k] = a\mathbf{1}[k]$ is*

$$y[k] = aG[1]. \tag{6.18}$$

Therefore the steady-state response of a BIBO stable LTI system to a discrete-time sinusoid input of frequency θ is also a discrete-time sinusoid of the same frequency. The amplitude of the output sinusoid is $\left|G[\mathrm{e}^{j\theta}]\right|$ times the input amplitude, and the phase of the output is shifted by $\measuredangle G[\mathrm{e}^{j\theta}]$ with respect to the input phase.

Definition 6.9.2. Assume $G[z]$ is rational, proper, and all its poles have magnitude less than one.

(a) The function $(-\pi, \pi] \to \mathbb{C}$, $\theta \mapsto G[\mathrm{e}^{j\theta}]$ is the **frequency response** of G.

(b) The function $(-\pi, \pi] \to \mathbb{R}$, $\theta \mapsto \left|G[e^{j\theta}]\right|$ is the **amplitude** or **magnitude response** of G.

(c) The function $(-\pi, \pi] \to (-\pi, \pi]$, $\theta \mapsto \measuredangle G[\mathrm{e}^{j\theta}]$ is the **phase response** of G.

Hence, the frequency response of a BIBO stable discrete-time LTI system $G[z]$ is determined by setting $z = \mathrm{e}^{j\theta}$ and varying θ. Here's the important point: Under the stated assumptions about the system ($G[z]$ is rational, proper, and all poles inside the unit disk), the region of convergence of the z-transform includes the unit circle. Therefore it is legitimate to set $z = \mathrm{e}^{j\theta}$ in $G[z]$ to get $G[\mathrm{e}^{j\theta}]$, which, by the way, equals the discrete-time Fourier transform of the impulse response $g[k]$.

Remark 6.9.3. If the impulse response $g[k] = \mathscr{Z}^{-1}\{G[z]\}$ is a real-valued function, then the complex conjugate of $G[\mathrm{e}^{j\theta}]$ is $G[\mathrm{e}^{-j\theta}]$. This implies that

$$|G[\mathrm{e}^{j\theta}]| = |G[e^{-j\theta}]| \qquad \text{(even function)}$$

and

$$\measuredangle G[\mathrm{e}^{j\theta}] = -\measuredangle G[e^{-j\theta}] \qquad \text{(odd function)}.$$

Thus when we plot the frequency response of a DT system with real rational transfer function, we only need to plot the response for θ in the range $[0, \pi]$ to get the complete picture (cf. Remark 2.6.3). ♦

Example 6.9.1. Consider the system

$$G[z] = \frac{z+1}{10z - 0.8}$$

Then its frequency response is

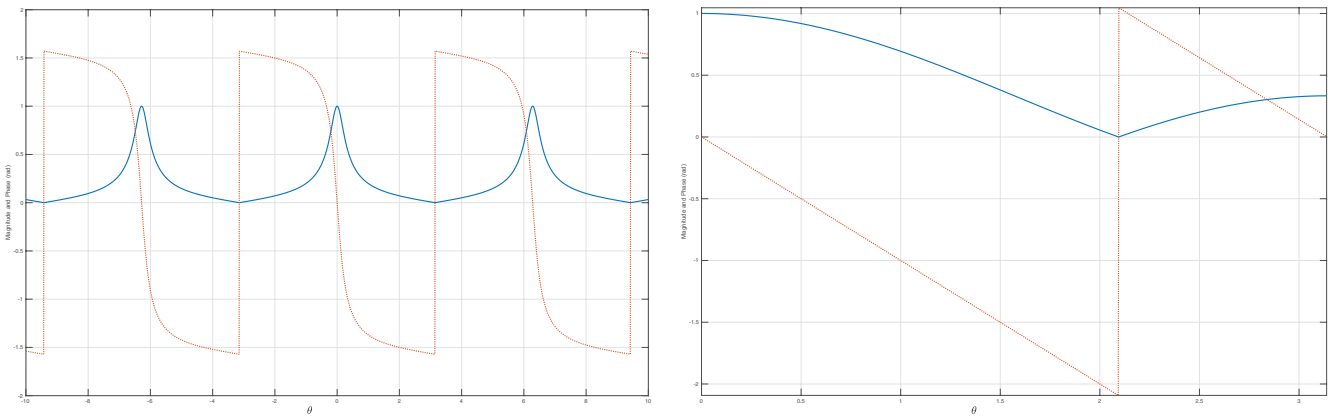
$$G[\mathrm{e}^{j\theta}] = \frac{\mathrm{e}^{j\theta} + 1}{10\mathrm{e}^{j\theta} - 0.8}.$$

Unlike CT systems, calculating useful expressions for the magnitude and phase response is complicated so we don't do it by hand. MATLAB can easily plot the frequency response using the code below.

```
n=[1 1];d=[10 -8];
w=-10:0.01:10;
G=freqz(n,d,w);
plot(w,abs(G),w,angle(G),':', 'LineWidth', 2);
```

Figure 6.5a shows the resulting plot. There we see, as expected, the 2π-periodicity of the frequency response and the even and odd nature, respectively, of the magnitude and phase responses. ▲

Remark 6.9.4. MATLAB always assumes that a discrete-time system arises from sampling a continuous-time system with sampling rate T. Therefore, by default, the `bode` command will plot the magnitude and phase Bode plots over the range $[0, \pi/T]$ where T is the sampling rate of the DT plant. For purely DT systems like the ones we've studied in this chapter, you can set $T = 1$. ♦



(a) Example 6.9.1. *(b) Example 6.9.2.*

Figure 6.5: Magnitude and phase response for two different DT systems.

Example 6.9.2. Consider the setup in Figure 6.6 where the A/D block operates at a sampling period of $T = 0.1$ seconds. Suppose that the DT system in Figure 6.6 computes the 3-point moving average

$$\frac{Y[z]}{U[z]} = G[z] = \frac{1}{3}\frac{z^2 + z + 1}{z^2}.$$

Find the steady-state output. Figure 6.5b shows the frequency response of $G[z]$. The input to the DT plant is $\cos(100kT) = \cos(10k)$ which is of the form $\cos(\theta k)$ with $\theta = 10$. Express θ as $\theta = \theta_0 + n2\pi$ with $\theta_0 \in (-\pi, \pi]$

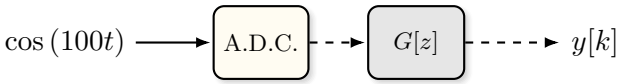


Figure 6.6: Applying a sampled CT sinusoid to a DT plant.

in the "baseband." This gives $\theta = -2.56 + 4\pi$. From the figure we have that $|G[e^{-j2.56}]| = |G[e^{j2.56}]| \approx 0.22$ and $\measuredangle G[e^{-j2.56}] = -\measuredangle G[e^{j2.56}] \approx -0.6$. Therefore, by Theorem 6.9.1, the steady-state output is

$$y[k] = 0.22\cos(10k - 0.6) = 0.22\cos(-2.56k - 0.6).$$

▲

Chapter 7

Sampled-data systems

In this chapter we learn how to discretize a plant and study its properties. The discretized plant model will serve as the basis for doing control design in discrete-time.

Contents

7.1 Introduction

We work with the unity-feedback sampled-data control system in Figure 7.1. We take the viewpoint of the

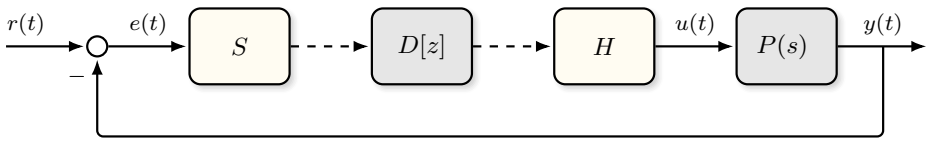


Figure 7.1: Sampled-data unity feedback control system.

controller, which sees a discrete-time system. Using linearity of the sample operator, we can rearrange the block diagram in Figure 7.1 to obtain the equivalent system in Figure 7.2. The discrete-time system $P_d[z] \coloneqq SPH$

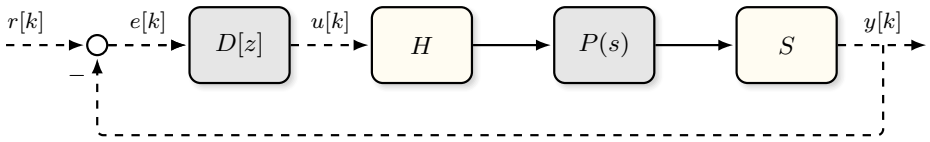


Figure 7.2: Equivalent configuration for a sampled-data unity feedback control system.

in Figure 7.2 is the step-invariant transformation of the continuous-time system $P(s)$ (cf. Section 4.3.2).

7.2 State-space analysis

We now derive the discrete-time system equations that arise as a result of discretizing continuous-time state-space models. Consider the system in Figure 7.3. Suppose that the plant $P(s)$ has state-space model (cf.

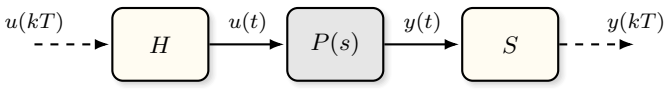


Figure 7.3: Step-invariance method applied to an LTI system.

Equation (5.4))

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) + Du(t). \end{aligned} \tag{7.1}$$

The solution to this differential equation for a given input signal $u(t)$ and a given initial condition $x(0) = x_0 \in \mathbb{R}^n$ is [Nielsen, 2017, Chapter 3]

$$\begin{aligned} x(t) &= \mathrm{e}^{tA}x_0 + \int_0^t \mathrm{e}^{(t-\tau)A}Bu(\tau)\mathrm{d}\tau \\ y(t) &= C\mathrm{e}^{tA}x_0 + \int_0^t C\mathrm{e}^{(t-\tau)A}Bu(\tau)\mathrm{d}\tau + Du(t). \end{aligned} \tag{7.2}$$

These are integral (convolution) equations giving $x(t)$ and $y(t)$ explicitly in terms of x_0 and $u(\tau)$, $0 \leq \tau \leq t$. Here e^{At} is the matrix exponential.

Definition 7.2.1. The **matrix exponential** of $A \in \mathbb{R}^{n\times n}$ is

$$\mathrm{e}^A := \sum_{k=0}^{\infty} \frac{1}{k!}A^k = I + A + \frac{1}{2!}A^2 + \frac{1}{3!}A^3 + \cdots .$$

It can be proved that the matrix exponential converges for every matrix A. If A is $n \times n$ then e^A is too; if A is not square, then e^A is undefined. Using the matrix exponential we can define the matrix valued function $\mathbb{R} \to \mathbb{R}^{n\times n}$ given by $t \mapsto \mathrm{e}^{tA}$. It has the properties:

1. $\mathrm{e}^{tA}\big|_{t=0} = I$ (identity matrix).
2. $\mathrm{e}^{t_1A}\mathrm{e}^{t_2A} = \mathrm{e}^{(t_1+t_2)A}$. Note that $\mathrm{e}^{tA_1}\mathrm{e}^{tA_2} \neq \mathrm{e}^{t(A_1+A_2)}$. Equality holds if and only if A_1 and A_2 commute, i.e., $A_1A_2 = A_2A_1$.
3. $\left(\mathrm{e}^A\right)^{-1} = \mathrm{e}^{-A}$ which implies that $\left(\mathrm{e}^{tA}\right)^{-1} = \mathrm{e}^{-tA}$.
4. A and e^{At} commute.
5. $\frac{\mathrm{d}}{\mathrm{d}t}\mathrm{e}^{tA} = A\mathrm{e}^{tA} = \mathrm{e}^{tA}A$.
6. $\mathscr{L}\left\{\mathrm{e}^{tA}\right\} = (sI - A)^{-1}$.

Example 7.2.1. Calculate e^{tA} for the matrix

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

From above we know that $\mathrm{e}^{tA} = \mathscr{L}^{-1}\left\{(sI - A)^{-1}\right\}$. Therefore we can proceed to perform the necessary calculations:

$$sI - A = \begin{bmatrix} s & -1 \\ 2 & s+3 \end{bmatrix}$$

and hence

$$\begin{aligned}(sI - A)^{-1} &= \frac{1}{s^2 + 3s + 2}\begin{bmatrix} s+3 & 1 \\ -2 & s \end{bmatrix} \\ &= \frac{1}{(s+1)(s+2)}\begin{bmatrix} s+3 & 1 \\ -2 & s \end{bmatrix}.\end{aligned}$$

Now use partial fractions on each entry of the above matrix

$$(sI - A)^{-1} = \begin{bmatrix} \frac{2}{s+1} - \frac{1}{s+2} & \frac{1}{s+1} - \frac{1}{s+2} \\ \frac{-2}{s+1} + \frac{2}{s+2} & \frac{-1}{s+1} + \frac{2}{s+2} \end{bmatrix}.$$

It is now easy to find the inverse Laplace transform of each entry in $(sI - A)^{-1}$ to obtain

$$\mathrm{e}^{tA} = \mathscr{L}^{-1}\left\{(sI - A)^{-1}\right\} = \begin{bmatrix} 2\mathrm{e}^{-t} - \mathrm{e}^{-2t} & \mathrm{e}^{-t} - \mathrm{e}^{-2t} \\ -2\mathrm{e}^{-t} + 2\mathrm{e}^{-2t} & -\mathrm{e}^{-t} + 2\mathrm{e}^{-2t} \end{bmatrix}.$$

▲

We state the key result of this section as a theorem whose proof is in Appendix 7.A.

Theorem 7.2.2. *The step-invariant transformation of (7.1) with sampling period T is the discrete-time system*

$$\begin{aligned} x[k+1] &= A_d x[k] + B_d u[k] \\ y[k] &= Cx[k] + Du[k] \end{aligned} \tag{7.3}$$

where $x[k] = x(kT)$, $u[k] = u(kT)$, $y[k] = y(kT)$ and

$$A_d \coloneqq \mathrm{e}^{TA} \tag{7.4a}$$

$$B_d \coloneqq \int_0^T \mathrm{e}^{\tau A}\mathrm{d}\tau B. \tag{7.4b}$$

Summary

Step-invariant transformation of a state-space model: a continuous-time state model (A, B, C, D) is mapped to the discrete-time state model (A_d, B_d, C, D), where A_d and B_d are given by (7.4). We will call this mapping c2d because that's what it is called in MATLAB. In terms of transfer functions,

$$P(s) = C(sI - A)^{-1}B + D$$

is mapped to

$$P_d[z] = C(zI - A_d)^{-1}B_d + D.$$

Remark 7.2.3. If A happens to be nonsingular, then we can explicitly integrate

$$\int_0^T \mathrm{e}^{\tau A}\mathrm{d}\tau = A^{-1}\left(\mathrm{e}^{AT} - I\right).$$

This fact can be used to simplify the computation of B_d.

Proof.

$$\int_0^T Ae^{\tau A}\mathrm{d}\tau = \int_0^T \frac{\mathrm{d}}{\mathrm{d}t}e^{\tau A}\mathrm{d}\tau = \mathrm{e}^{\tau A}\big|_0^T = \mathrm{e}^{TA} - I.$$

■

♦

Remark 7.2.4. The discretized state model obtained in Theorem 7.2.2 is *exact*, valid for all T — there are no approximations. ♦

Remark 7.2.5. The determination of A_d and B_d involves the determination of e^{tA}. As illustrated in Example 7.2.1, the computation of e^{tA} can be carried out using Laplace transforms.

Example 7.2.2. (First order system) The continuous-time state-space model is given by

$$\begin{aligned} \dot{x} &= ax + bu, \qquad a \neq 0, \\ y &= cx + du \end{aligned}$$

with x, y and u scalars. To compute the step-invariant transformation of this system for any sampling period $T > 0$ we have

$$a_d = \mathrm{e}^{aT}.$$

Since a is invertible (cf. Remark 7.2.3)

$$b_d = a^{-1}\left(\mathrm{e}^{aT} - 1\right)b = \frac{b}{a}\left(\mathrm{e}^{aT} - 1\right).$$

Putting this together we get

$$\begin{aligned} x[k+1] &= \mathrm{e}^{aT}x[k] + \frac{b}{a}\left(\mathrm{e}^{aT} - 1\right)u[k] \\ y[k] &= cx[k] + du[k]. \end{aligned}$$

In terms of transfer functions, the discretized system is (cf. Section 6.5.2)

$$P_d[z] = c(z - a_d)^{-1}b_d + d = \frac{cb}{a}\frac{\mathrm{e}^{aT} - 1}{z - \mathrm{e}^{aT}} + d.$$

▲

Example 7.2.3. (First order system - numerical) Find the step-invariant transformation of the system

$$\begin{aligned} \dot{x} &= -2x + u \\ y &= 3x \end{aligned}$$

for the sampling period is $T = 1$. This is just a special case of Example 7.2.2 so we immediately get

$$\begin{aligned} x[k+1] &= \mathrm{e}^{-2}x[k] + \frac{1}{2}\left(1 - \mathrm{e}^{-2}\right)u[k] \\ y[k] &= 3x[k]. \end{aligned}$$

or, equivalently,

$$P_d[z] = \frac{3}{2}\frac{1 - \mathrm{e}^{-2}}{z - \mathrm{e}^{-2}}.$$

▲

Exercise 7.1. Repeat Example 7.2.2 for the case $a = 0$.

Example 7.2.4. (Double integrator) Consider

$$\dot{x} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} x + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u$$
$$y = \begin{bmatrix} 1 & 0 \end{bmatrix} x.$$

This is called the double integrator because

$$\ddot{y} = \ddot{x}_1 = \dot{x}_2 = u, \qquad \text{i.e.,} \qquad \frac{Y(s)}{U(s)} = \frac{1}{s^2}.$$

Observe that $A^2 = 0$ (A is **nilpotent**[1]) so using Definition 7.2 of the matrix exponential we get

$$\mathrm{e}^{tA} = I + At = \begin{bmatrix} 1 & t \\ 0 & 1 \end{bmatrix}.$$

Alternatively we could use LTs to calculate e^{tA}. In that case

$$(sI - A)^{-1} = \begin{bmatrix} s & -1 \\ 0 & s \end{bmatrix}^{-1} = \frac{1}{s^2} \begin{bmatrix} s & 1 \\ 0 & s \end{bmatrix} = \begin{bmatrix} \frac{1}{s} & \frac{1}{s^2} \\ 0 & \frac{1}{s} \end{bmatrix}$$

and hence

$$\mathrm{e}^{tA} = \mathscr{L}^{-1}\left\{(sI - A)^{-1}\right\} = \begin{bmatrix} 1 & t \\ 0 & 1 \end{bmatrix}.$$

By either method we get

$$A_d = \mathrm{e}^{TA} = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix}$$

and

$$B_d = \int_0^T \mathrm{e}^{\tau A} \mathrm{d}\tau B = \int_0^T \begin{bmatrix} 1 & \tau \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} \mathrm{d}\tau = \int_0^T \begin{bmatrix} \tau \\ 1 \end{bmatrix} \mathrm{d}\tau = \begin{bmatrix} \frac{T^2}{2} \\ T \end{bmatrix}.$$

The discretized system is therefore

$$x[k+1] = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix} x[k] + \begin{bmatrix} \frac{T^2}{2} \\ T \end{bmatrix} u[k]$$
$$y[k] = \begin{bmatrix} 1 & 0 \end{bmatrix} x[k].$$

The discretized transfer function is given by

$$P_d[z] = C(zI - A_d)^{-1} B_d = \frac{T^2}{2} \frac{z+1}{(z-1)^2}.$$

▲

[1] A square matrix A is nilpotent if there exists a $k \geq 0$ such that $A^k = 0$. It is nilpotent if, and only if, all of its eigenvalues are zero.

In the previous two example we had the following phenomenon: If the continuous-time system has a pole at $s = \lambda$, then its step-invariant transformation has a pole at $z = \mathrm{e}^{\lambda T}$. For instance, in Example 7.2.3, the continuous-time system has a pole at $s = -2$ while the discretized system has a pole at $z = \mathrm{e}^{-2T}$. In Example 7.2.4 the continuous-time system has two poles at $s = 0$ while the discretized system has two poles at $z = \mathrm{e}^0 = 1$.

The proof that this relationship between the poles of a continuous-time system and its step-invariant transform holds in general can be done using state-space representations of LTI systems and the **spectral mapping theorem**[2]. Let λ be an eigenvalue of A. Then there is a non-zero, possibly complex, vector v such that $Av = \lambda v$ (eigenvector). Then

$$\begin{aligned} A_d v = \mathrm{e}^{TA} v &= \left(I + TA + \frac{T^2}{2} A^2 + \frac{T^3}{3!} A^3 + \cdots \right) v \\ &= v + TAv + \frac{T^2}{2} A^2 + \frac{T^3}{3!} A^3 v + \cdots . \end{aligned}$$

But $A^2 v = A(\lambda v) = \lambda A v = \lambda^2 v$; $A^3 v = A(A^2 v) = \lambda^2 A v = \lambda^3 v$ etc. Therefore

$$\begin{aligned} A_d v &= \left(1 + T\lambda + \frac{T^2}{2} \lambda^2 + \frac{T^3}{3!} \lambda^3 + \cdots \right) v \\ &= \mathrm{e}^{\lambda T} v. \end{aligned}$$

This shows that $\mathrm{e}^{\lambda T}$ is an eigenvalue of A_d with corresponding eigenvector v.

Thus, as noted in Section 4.3.2, the step-invariant transformation preserves stability. If $\lambda = \alpha + j\beta$ is an eigenvalue of A with $\mathrm{Re}\,(\lambda) = \alpha < 0$, then $\mathrm{e}^{\lambda T} = \mathrm{e}^{\alpha T} \mathrm{e}^{j\beta T}$ is an eigenvalue of A_d. Furthermore, since $T > 0$ we have

$$|\mathrm{e}^{\lambda T}| = |\mathrm{e}^{\alpha T}||\mathrm{e}^{j\beta T}| = |\mathrm{e}^{\alpha T}| < 1.$$

On the other hand, again as noted in Section 4.3.2, there is no simple relationship between the zeros of the continuous-time system and its step-invariant transformation.

Example 7.2.5. In the previous examples we started from a continuous-time state-space model. However, often in continuous-time we begin with a TF model. In this example we will find the step-invariant transformation of

$$P(s) = \frac{1}{s^2(s+1)}.$$

The first step is to find a continuous-time state-space model. We can use the exact same technique in continuous-time as we did in Section 6.5.1, see Remark 6.5.1. In this case we get

$$A = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & -1 \end{bmatrix}, \qquad B = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \qquad C = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}, \qquad D = 0.$$

The second step is to compute e^{tA} so that we can get our discretized model. In this example I would use the fact $\mathrm{e}^{tA} = \mathscr{L}^{-1}\left\{ (sI - A)^{-1} \right\}$.

Exercise 7.2. Show that

$$\mathrm{e}^{tA} = \begin{bmatrix} 1 & t & -1 + t + \mathrm{e}^{-t} \\ 0 & 1 & 1 - \mathrm{e}^{-t} \\ 0 & 0 & \mathrm{e}^{-t} \end{bmatrix}.$$

[2]Suppose A is square; suppose $f(z)$ is a function analytic at the eigenvalues of A; then the eigenvalues of $f(A)$ equal the numbers $f(\lambda)$, as λ ranges over the eigenvalues of A.

The third step is to compute A_d and B_d:

$$A_d = \mathrm{e}^{TA} = \begin{bmatrix} 1 & t & -1+T+\mathrm{e}^{-T} \\ 0 & 1 & 1-\mathrm{e}^{-T} \\ 0 & 0 & \mathrm{e}^{-T} \end{bmatrix}$$

$$B_d = \int_0^T \mathrm{e}^{\tau A}\mathrm{d}\tau B = \int_0^T \begin{bmatrix} -1+\tau+\mathrm{e}^{-\tau} \\ 1-\mathrm{e}^{-\tau} \\ \mathrm{e}^{-\tau} \end{bmatrix} \mathrm{d}\tau = \begin{bmatrix} -T+\frac{T^2}{2}+1-\mathrm{e}^{-T} \\ T-1+\mathrm{e}^{-T} \\ 1-\mathrm{e}^{-T} \end{bmatrix}.$$

We now have a discrete-time state-space model. The fourth step is to get the discrete-time TF model $P_d[z] = C(zI - A_d) + B_d$, if needed.

Exercise 7.3. Compute $P_d[z]$ when $T = 1$. Check your answer with MATLAB.

▲

Remark 7.2.6. The form of A_d lends itself to an easy computation of the associated transition matrix A_d^k, required in the solution of the sampled-data state equation:

$$A_d^k = \left(\mathrm{e}^{AT}\right)^k = \mathrm{e}^{AkT}.$$

Since we would have determined e^{At} as part of the calculation for A_d, we simply replace t by kT to get A_d^k. ♦

7.3 Transform analysis

So far, we have described how to obtain $P_d[z]$ starting from either (A, B, C, D) or $P(s)$. Is there a way to go from $P(s)$ directly to $P_d[z]$? Yes, here's a procedure. Consider again the system in Figure 7.3.

1. Let $u[k] = u(kT) = \mathbf{1}[k]$. Then $U[z] = z/(z-1)$.
2. Then $u(t)$ is a continuous-time unit step $\mathbf{1}(t)$ so $U(s) = 1/s$.
3. Then $Y(s) = P(s)U(s)$. Get $y(t)$, the step response of P.
4. Sample $y(t)$ to get $y(kT) = y[k]$ and then $Y[z]$.
5. Finally, the step-invariant transformation of P is

$$\frac{Y[z]}{U[z]} = \frac{z-1}{z}\mathscr{Z}\left(\mathscr{L}^{-1}\left\{\frac{P(s)}{s}\right\}\Big|_{t=kT}\right). \tag{7.5}$$

Example 7.3.1. Let's re-do Example 7.2.3 using transform methods. The continuous-time system is

$$P(s) = \frac{3}{s+2}.$$

We perform each step in (7.5) carefully. We have that

$$\frac{P(s)}{s} = \frac{3}{s(s+2)} = \frac{1.5}{s} - \frac{1.5}{s+2}$$

so that

$$y(t) = \mathscr{L}^{-1}\left(\frac{P(s)}{s}\right) = \frac{3}{2}\left(1-\mathrm{e}^{-2t}\right)\mathbf{1}(t).$$

Sample this signal every T seconds

$$y[k] = \frac{3}{2}\left(1 - \mathrm{e}^{-2kT}\right), \qquad k \geq 0$$

and then take its z-transform

$$\begin{aligned} Y[z] = \mathscr{Z}\{y[k]\} &= \frac{3}{2}\frac{z}{z-1} - \frac{3}{2}\frac{z}{z-\mathrm{e}^{-2T}} \\ &= \frac{3}{2}\frac{z(1-\mathrm{e}^{-2T})}{(z-1)(z-\mathrm{e}^{-2T})}. \end{aligned}$$

Finally, we get

$$P_d[z] = \frac{z-1}{z}Y[z] = \frac{3}{2}\frac{1-\mathrm{e}^{-2T}}{z-\mathrm{e}^{-2T}}.$$

▲

MATLAB uses the state-space approach to compute the step-invariant transformation.

Exercise 7.4. Find $\mathsf{c2d}(P)$ for $P(s) = 1/s$.

Exercise 7.5. Consider the systems in Figure 7.4. The top system is $\mathsf{c2d}(P_1)\mathsf{c2d}(P_2)$. The bottom system is $\mathsf{c2d}(P_1P_2)$. Show that the two systems are not equal. Conclude that in general $\mathsf{c2d}(P_1)\mathsf{c2d}(P_2) \neq \mathsf{c2d}(P_1P_2)$.

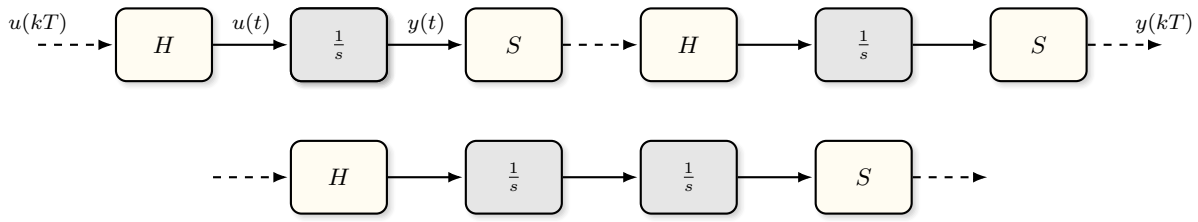


Figure 7.4: Exercise 7.5.

Exercise 7.6. Show that if $P(s) = 1$, then $P_d[z] = \mathsf{c2d}(P(s)) = 1$.

Exercise 7.7. Prove that the function $\mathsf{c2d}$ is linear, i.e., $\mathsf{c2d}(P_1 + P_2) = \mathsf{c2d}(P_1) + \mathsf{c2d}(P_2)$.

We summarize the results of Sections 7.2 and 7.3 below.

Summary: Step-invariant transformation

$$\begin{array}{ccc} (A,B,C,D) & \xrightarrow{(7.4)} & (A_d,B_d,C,D) \\ \Big\downarrow{\scriptstyle (5.5)} & & \Big\downarrow{\scriptstyle (6.9)} \\ P(s) & \xrightarrow{(7.5)} & P_d[z] \end{array}$$

7.4 The effect of sampling

We've been studying the setup in Figure 7.3. What effect do the sample and hold have on the continuous-time plant $P(s)$ in producing $P_d[z]$? We compare the properties of $P(s)$ and $P_d[z] = \mathsf{c2d}(P(s))$.

Example 7.4.1. (Pathological sampling) As usual consider the setup in Figure 7.3 where the plant is an oscillator which oscillates at the sampling frequency, i.e.,

$$P(s) = \frac{\omega_{\mathrm{s}} s}{s^2 + \omega_{\mathrm{s}}^2}, \qquad \omega_{\mathrm{s}} = \frac{2\pi}{T}.$$

Apply the unit step $\mathbf{1}[k]$ as an input to this system. Then $u(t) = \mathbf{1}(t)$ and therefore

$$y(t) = \mathscr{L}^{-1}\left\{\frac{\omega_{\mathrm{s}}}{s^2 + \omega_{\mathrm{s}}^2}\right\} = \sin(\omega_{\mathrm{s}} t).$$

After the sample block we get

$$y[k] = y(t)|_{t=kT} = \sin\left(\frac{2\pi}{T} kT\right) = \sin(2\pi k) = 0.$$

It follows that $P_d[z] = 0$. So we have a non-zero system $P(s)$ whose discretization is zero. But $P(s) = 0$ also has a discretization equal to zero. So two *different* continuous-time systems get mapped to the same discrete-time system via $\mathsf{c2d}$. This shows that the mapping $P(s) \mapsto P_d[z]$ is not one-to-one. In other words, the linear map $\mathsf{c2d}$ has a non-trivial null space. ▲

7.4.1 Frequency response

Consider the setup in Figure 7.5, we'd like to compare $y(t)$ and $y[k]$ in steady-state when $u(t)$ is a sinusoidal input. We start with the sampling of signals. Consider the sinusoid $u(t) = \mathrm{e}^{j\omega t}$ of frequency ω radians per

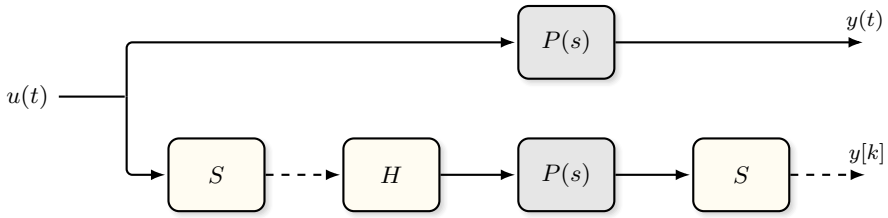


Figure 7.5: Comparing the frequency responses of $P(s)$ and $P_d[z]$.

second. Suppose it's sampled with sampling period T; this yields $u(kT) = \mathrm{e}^{j\omega Tk}$. We treat this sampled signal as the discrete-time signal

$$u[k] = \mathrm{e}^{j\omega Tk} = \mathrm{e}^{j\theta k},$$

where $\theta = \omega T$ has units of radians. Now the ω-axis can be divided into non-overlapping intervals of width $2\pi/T$:

$$\ldots, \left(-\frac{\pi}{T}, \frac{\pi}{T}\right], \left(\frac{\pi}{T}, \frac{3\pi}{T}\right], \ldots$$

Let us suppose ω is in the interval

$$\left(\frac{\pi}{T}, \frac{3\pi}{T}\right].$$

Then $\omega = \omega_0 + \frac{2\pi}{T}$, where ω_0 is in the "baseband" $\left(-\frac{\pi}{T}, \frac{\pi}{T}\right]$. Also,

$$u[k] = \mathrm{e}^{j(\omega_0 T + 2\pi)k} = \mathrm{e}^{j\omega_0 Tk} = \mathrm{e}^{j\theta_0 k},$$

where $\theta_0 := \omega_0 T$ is in the interval $(-\pi, \pi]$.

Now back to setup in Figure 7.5. We only consider the relationship when $u(t)$ is a low frequency signal. For the general comparison see Appendix 7.B. Let $\omega_s = 2\pi/T$ be the sampling frequency and let $\omega_N = \omega_s/2 = \pi/T$ be the Nyquist frequency. Assume $u(t) = e^{j\omega t}$ with $\omega \ll \omega_N$. Of course, in steady-state, the output of the top path is $y(t) = P(j\omega)e^{j\omega t}$. For the bottom path, the sampled signal is $e^{j\theta k}$ with $\theta = \omega T \ll \pi$ and so the steady-state output is $y[k] = P_d[e^{j\theta}]e^{j\theta k}$.

Let's look more closely at the bottom path. Since $\omega \ll \omega_N$, the signal coming out of the hold block is very close to $e^{j\omega t}$. So the output of $P(s)$, in steady-state is $P(j\omega)e^{j\omega t}$ and hence $y[k]$ is very close to $P(j\omega)e^{j\omega kT}$. We conclude that

$$P_d[e^{j\theta}]e^{j\theta k} \approx P(j\omega)e^{j\omega Tk}.$$

Thus at **low frequencies**

$$P_d[e^{j\omega T}] \approx P(j\omega)$$

or, in terms of $\theta = \omega T$,

$$P_d[e^{j\theta}] \approx P\left(j\frac{\theta}{T}\right).$$

At high frequencies the signal coming out of the zero order hold is no longer approximately equal $e^{j\omega t}$. It will contain higher frequency components due to aliasing and the zero order hold. A rule of thumb is that $P_d[e^{j\omega T}] \approx P(j\omega)$ when $\omega \le \omega_N/5$. Appendix 7.B discusses this further.

Example 7.4.2. Consider

$$P(s) = \frac{1}{s^2 + 0.1s + 1}, \qquad P_d[z] = \mathsf{c2d}(P) = \frac{0.19(z + 0.98)}{z^2 - 1.57z + 0.94}$$

where we've used the sampling frequency $\omega_s = 10$ rad/s. The Bode plots for these systems are shown in Figure 7.6. As expected they are very similar at low frequencies but start to deviate significantly near the

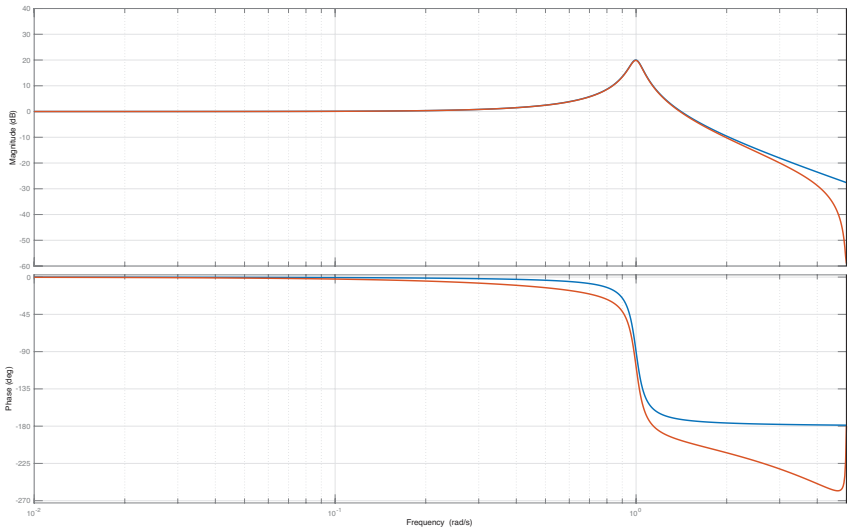
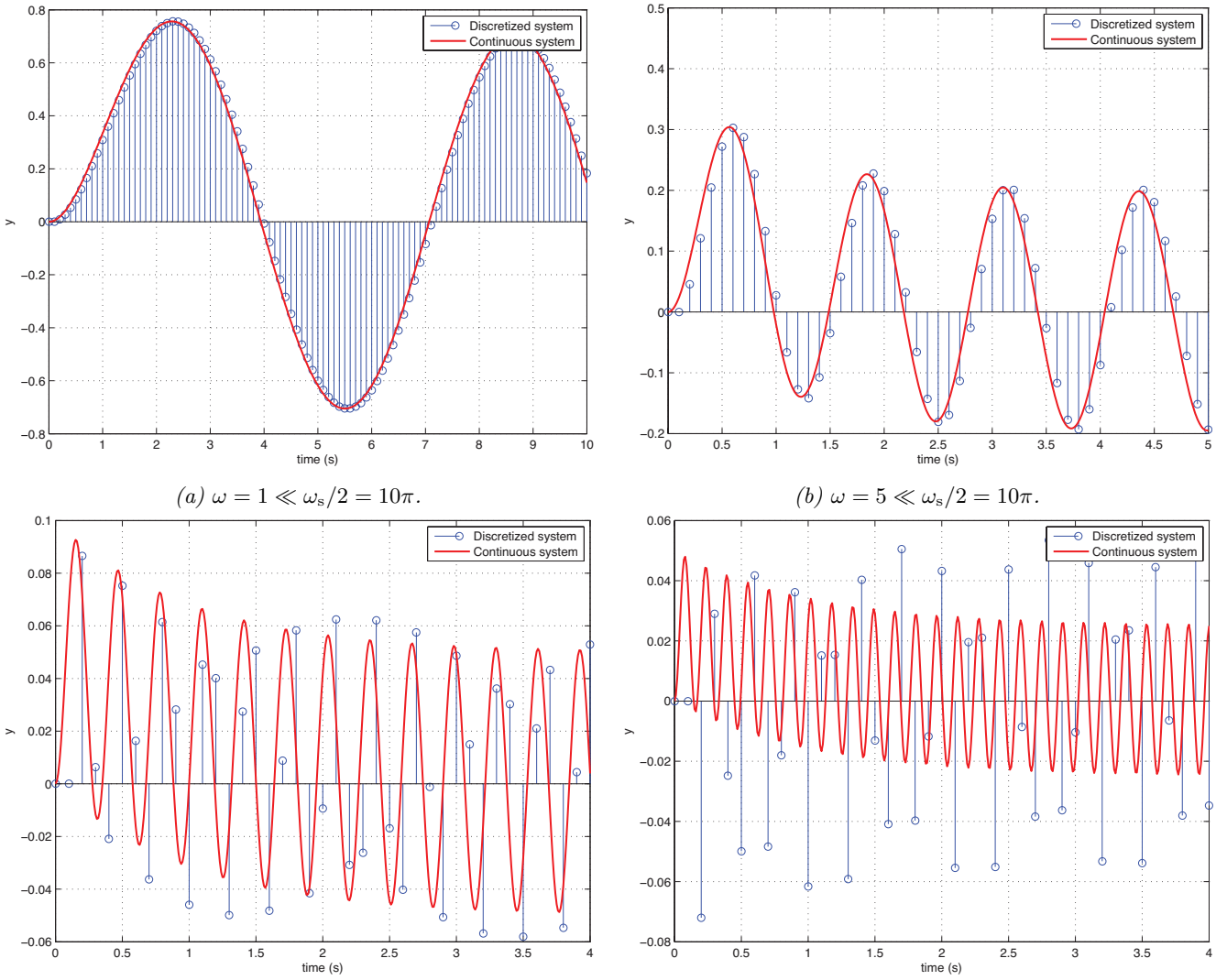


Figure 7.6: The Bode plots for $P(s)$ (blue) and $P_d[z]$ (red) from Example 7.4.2.

Nyquist frequency 5 rad/s. ▲

Example 7.4.3. Consider the setup in Figure 7.5 with $T = 0.1$ and

$$P(s) = \frac{1}{s + 1}, \qquad P_d[z] = \mathsf{c2d}(P(s)) = \frac{0.09516}{z - 0.9048}.$$



(a) $\omega = 1 \ll \omega_s/2 = 10\pi$.

(b) $\omega = 5 \ll \omega_s/2 = 10\pi$.

(c) $\omega = 20 < \omega_s/2 = 10\pi$.

(d) $\omega = 40 > \omega_s/2 = 10\pi$.

Figure 7.7: Example 7.4.3.

If we apply the input $u(t) = \cos(\omega t)$ then, based on the above discussion, we expect that in steady-state the signals $y(t)$ and $y[k]$ will have similar amplitudes and phases for $\omega \ll \omega_N = \omega_s/2 = 10\pi$. Figure 7.7 illustrates these results. The two responses begin to differ significantly as ω gets larger and larger. ▲

The next example reinforces the idea that the frequency response of the discrete-time system associated to a sampled-data system will only give meaningful results at low frequencies.

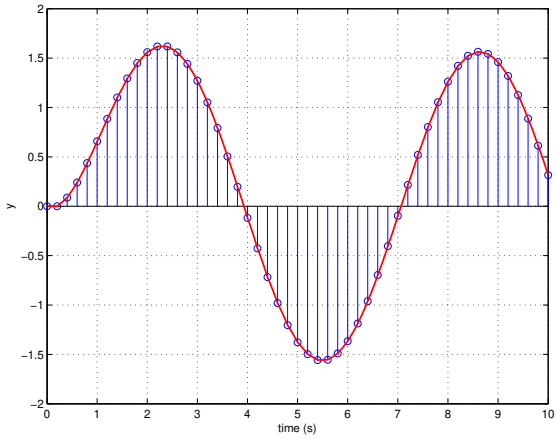
Example 7.4.4. Consider the unity feedback sampled-data system in Figure 7.2 with $T = 0.2$ and

$$P(s) = \frac{1}{s-1}, \qquad P_d[z] = \mathsf{c2d}(P(s)) = \frac{\mathrm{e}^T - 1}{z - \mathrm{e}^T}.$$

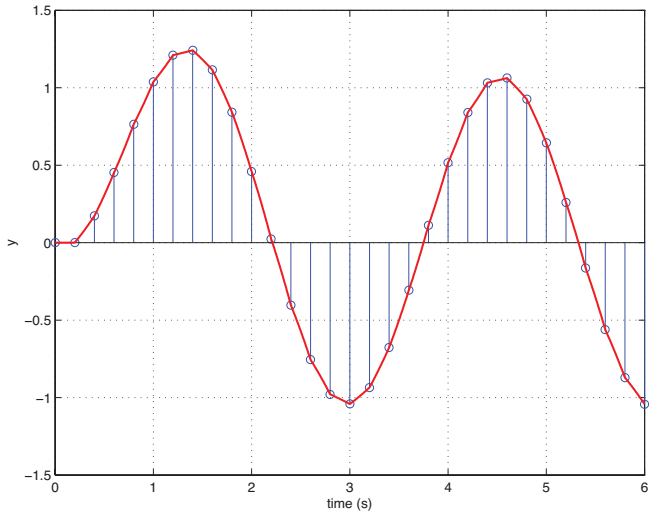
We use a proportional controller $D[z] = 2$ so that the ch.p of the closed-loop system

$$\pi[z] = z - \mathrm{e}^T + 2\left(\mathrm{e}^T - 1\right) = z - 2 + \mathrm{e}^T$$

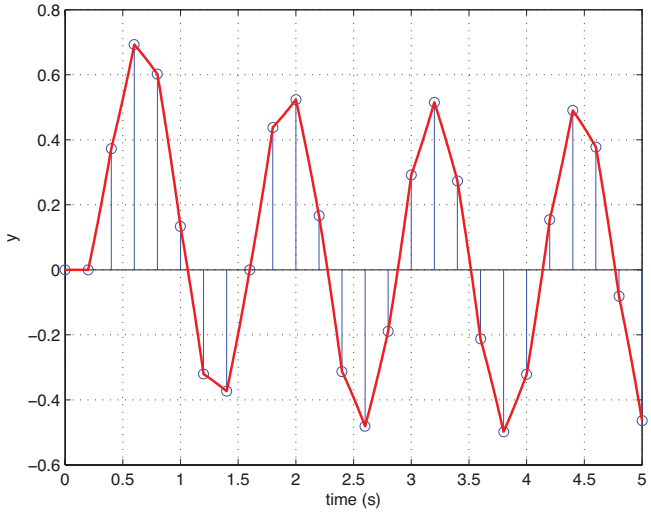
is Schur and the feedback system is I/O stable. Apply the reference signal $r(t) = \cos(\omega t)$ so that $r[k] = \cos(\omega T k)$. We now simulate the closed-loop sampled-data system for various values of ω. In Figure 7.8 we are plotting $y(t)$ in red and $y[k] = y(kT)$ in blue.



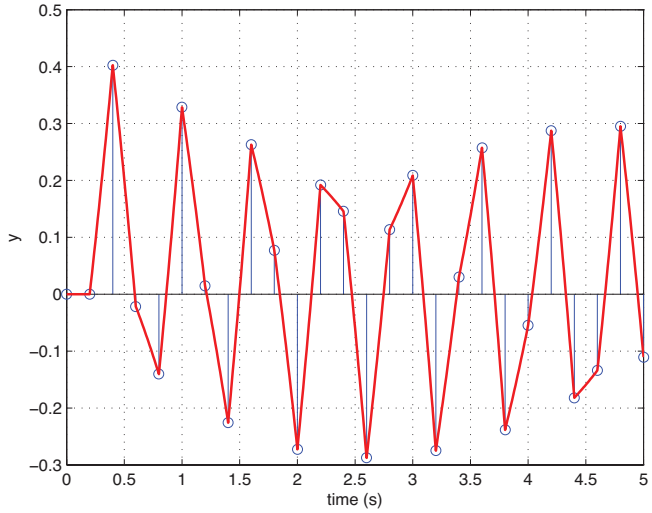
(a) $\omega = 1 \ll \omega_s/2 = 5\pi$.



(b) $\omega = 2 < \omega_s/2 = 5\pi$.



(c) $\omega = 5 < \omega_s/2 = 5\pi$.



(d) $\omega = 10 < \omega_s/2 = 5\pi$.

Figure 7.8: Example 7.4.4.

When $\omega = 1$ the steady state output is very similar to what you'd expect from a continuous-time system. When $\omega = 2$ we start to see some high frequency content in the output of the sampled-data system. By the time we get to $\omega = 10$ we are getting terrible inter-sample behaviour.

The conclusion is that, when treating a sampled-data system as a discrete-time system, the notion of a 'frequency response' makes sense as long as $\omega \ll \omega_s$. The rule of thumb is $\omega \leq \omega_s/10 = \omega_N/5$. ▲

7.5 Pathological sampling

The scenario in Example 7.4.1 is called **pathological sampling**. Recall that in the step-invariant transformation, the poles of $P(s)$ (eigenvalues of A) are mapped to poles of $P_d[z]$ (eigenvalues of A_d) via the mapping $z = e^{sT}$. The sampling frequency ω_s is called **pathological** (relative to A) if A has two different eigenvalues, $\lambda_1 \neq \lambda_2$, that become equal eigenvalues of A_d, i.e., get mapped to the same eigenvalue of A_d, i.e.,

$$\lambda_1 \neq \lambda_2 \text{ are eigenvalues of } A \qquad \text{and} \qquad e^{\lambda_1 T} = e^{\lambda_2 T}.$$

This happens if, and only if,

$$\lambda_1 = \lambda_2 + j\omega_s k, \qquad k \in \mathbb{Z}. \tag{7.6}$$

Exercise 7.8. Show that $e^{\lambda_1 T} = e^{\lambda_2 T}$ if, and only if (7.6) holds.

Example 7.5.1. Suppose A has eigenvalues at $\{0, 0, \pm j, 1 \pm j2\}$ as illustrated in Figure 7.9. Find the pathological sampling frequencies.

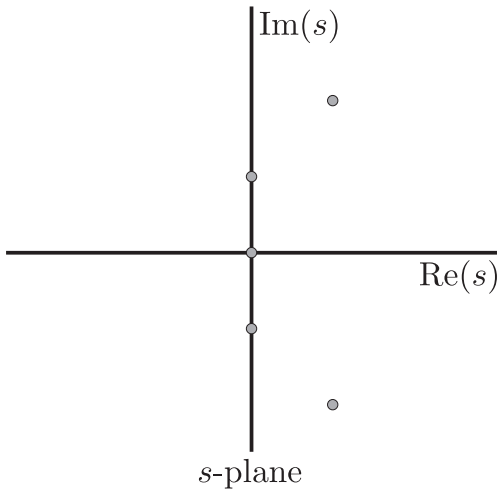


Figure 7.9: Eigenvalues of A in Example 7.5.1.

The pathological sampling frequencies can be calculated as follows. The eigenvalues go through two vertical lines. For the line $\mathrm{Re}\,(s) = 0$: The lowermost eigenvalue is $s = -j$. The distances from it to the other eigenvalues on this line are 1 and 2. Thus the sampling frequency is pathological if $k\omega_s = 1$ or $k\omega_s = 2$ for some positive integer k. Thus the following frequencies are pathological:

$$\left\{\frac{1}{k} : k \geq 1\right\} \cup \left\{\frac{2}{k} : k \geq 1\right\}.$$

For the line $\mathrm{Re}\,(s) = 1$: The lowermost eigenvalue is $s = 1 - 2j$. The distance from it to the other eigenvalue on this line is 4. Thus the sampling frequency is pathological if $k\omega_s = 4$ for some positive integer k. Thus the following frequencies are pathological:

$$\left\{\frac{4}{k} : k \geq 1\right\}.$$

Since we have looked at all possible vertical lines, the set of all pathological sampling frequencies is

$$\left\{\frac{1}{k} : k \geq 1\right\} \cup \left\{\frac{2}{k} : k \geq 1\right\} \cup \left\{\frac{4}{k} : k \geq 1\right\}.$$

Since 4 is divisible by 1 and 2, the set of all pathological sampling frequencies is

$$\left\{\frac{4}{k} : k \geq 1\right\}.$$

Notice that this set has a maximal element, namely 4 rad/s; therefore, ω_s will be non-pathological if we sample fast enough. In this example, we avoid pathological sampling if our sampling period satisfies $T < \pi/2$. ▲

Pathological sampling should be avoided when treating a sampled-data system as a discrete-time system because it can result in a form of 'system aliasing' where the dynamic behaviour of the plant is not fully captured at the sampling instances. Pathological sampling can also destroy the controllability and observability of a system[3] .

Example 7.5.2. Let ω_s denote the sampling frequency in radians per second, i.e., $\omega_s := 2\pi/T$. Consider once again the oscillator from Example 7.4.1. A state-space model of this system is

$$\dot{x} = \begin{bmatrix} 0 & 1 \\ -\omega_s^2 & 0 \end{bmatrix} x + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u.$$

The discretization has

$$B_d = \int_0^T \mathrm{e}^{A\tau} \mathrm{d}\tau \; B = A^{-1}(\mathrm{e}^{TA} - I)B = 0.$$

So the pair (A, B) is controllable while the pair (A_d, B_d) is not[4]. This example shows that sampling can destroy controllability. ▲

It turns out that controllability and observability are preserved if the sampling frequency is non-pathological.

Theorem 7.5.1. *If the sampling frequency is non-pathological, then*

(i) (A, B) *controllable implies* (A_d, B_d) *controllable,*

(ii) (C, A) *observable implies* (C, A_d) *observable.*

Example 7.5.2 shows that controllability can be lost by sampling. What about the converse? Can we gain controllability by sampling? No.

Theorem 7.5.2. *If* (A, B) *isn't controllable, then neither is* (A_d, B_d).

7.6 Comments on selecting the sampling period

There are several constraints and competing considerations one must make when selecting the sampling period for a sampled-data system.

(a) We should select T small enough to avoid pathological sampling so that we can effectively do direct design.

(b) In light of our study on the frequency response of sampled data systems, we should select T so that the closed-loop bandwidth of the system satisfies $\omega_s \geq 10\omega_{BW}$. This is particularly important when doing design by emulation.

(c) T should be small enough so that the inter-sample behaviour is good. This is best addressed through simulation.

(d) T should otherwise be as large as possible to keep the processor speed as low as possible (for cost) and so that all computations can be done in real-time.

(e) A continuous-time plant with slow dynamics together with fast sampling may yield samples with very small differences. This can lead to numerical issues with finite precision arithmetic.

[3] This discussion uses some notions covered in Chapter 9.

[4] In the sense that the controllability matrix (9.2) has rank 2 for (A, B) and it has rank 0 for (A_d, B_d).

(f) The sampling period is often dictated by other considerations unrelated to control, e.g., sensors and actuators may be connected to a bus with fixed rates.

In our study of the frequency response we have seen that sampling high frequency inputs leads to bad intersample behaviour. This is due to aliasing. Therefore, when possible, it is a good idea to apply a low pass analog filter to all signals prior to sampling. A rule of thumb is to use a low pass filter with a cutoff frequency less than or equal to $\omega_s/5$. See Appendix 1.A.

7.A Proof of Theorem 7.2.2

For any two times $\tau_1 \geq \tau_0$, we have

$$x(\tau_1) = \mathrm{e}^{(\tau_1-\tau_0)A}x(\tau_0) + \int_{\tau_0}^{\tau_1} \mathrm{e}^{(\tau_1-\tau)A}Bu(\tau)\mathrm{d}\tau.$$

Denote the sampling instances by $t_k \coloneqq kT$, $k = 0, 1, 2, \ldots$. Then, letting t be any time in the interval $[t_k, t_{k+1})$, we have

$$x(t) = \mathrm{e}^{(t-t_k)A}x(t_k) + \int_{t_k}^{t} \mathrm{e}^{(t-\tau)A}Bu(\tau)\mathrm{d}\tau, \qquad t_k \leq t < t_{k+1}.$$

Now $u(\tau)$, being the output of the hold, is constant ($u(t_k)$) over the interval $[t_k, t_{k+1})$ so we can pull it out of the integral. Thus

$$x(t) = \mathrm{e}^{(t-t_k)A}x(t_k) + \int_{0}^{t-t_k} \mathrm{e}^{\tau A}\mathrm{d}\tau Bu(t_k).$$

Define the matrix valued functions

$$\Phi(t) \coloneqq \mathrm{e}^{tA}, \qquad \Gamma(t) \coloneqq \int_{0}^{t} \mathrm{e}^{\tau A}\mathrm{d}\tau B.$$

Then, for $t \in [t_k, t_{k+1})$, we can write

$$x(t) = \Phi(t - t_k)x(t_k) + \Gamma(t - t_k)u(t_k).$$

Letting $t \to t_{k+1}$ we get

$$x(t_{k+1}) = \Phi(t_{k+1} - t_k)x(t_k) + \Gamma(t_{k+1} - t_k)u(t_k).$$

Since $t_{k+1} - t_k = T$ for all k, we get

$$x(t_{k+1}) = \Phi(T)x(t_k) + \Gamma(T)u(t_k).$$

Define

$$A_d \coloneqq \Phi(T) = \mathrm{e}^{TA}$$

$$B_d \coloneqq \Gamma(T) = \int_{0}^{T} \mathrm{e}^{\tau A}\mathrm{d}\tau B.$$

Thus the state equation at the sampling instances is exactly

$$x((k+1)T) = A_d x(kT) + B_d u(kT).$$

The output at the sampling instances is simply

$$y(kT) = Cx(kT) + Du(kT).$$

7.B Relationship between $P(j\omega)$ and $P_d[\mathrm{e}^{j\theta}]$

Consider the setup in Figure 7.10. The first step is to establish a relationship between the Fourier transform of

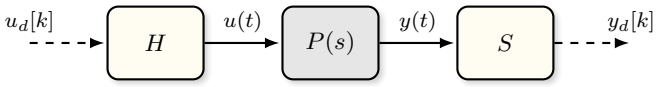


Figure 7.10: Comparing the frequency response of $P(s)$ and $P_d[z] = \mathsf{c2d}(P(s))$.

$y(t)$ and the discrete-time Fourier transform of $y_d[k] = y(kT)$. You can think of this as the frequency domain action of the ideal sampler.

The **periodic extension** of a function $Y(j\omega)$ is

$$Y_{\mathrm{e}}(j\omega) := \sum_{k=-\infty}^{\infty} Y(j\omega + jk\omega_{\mathrm{s}}).$$

Here ω_{s} is the sampling frequency. Notice that Y_{e} is a periodic function of frequency with period ω_{s}:

$$Y_{\mathrm{e}}(j(\omega + \omega_{\mathrm{s}})) = \sum_{k=-\infty}^{\infty} Y(j\omega + j\omega_{\mathrm{s}} + jk\omega_{\mathrm{s}}) = Y_{\mathrm{e}}(j\omega).$$

Lemma 7.B.1. *The Fourier transforms of $y(t)$ and $y_d[k]$ are related by*

$$Y_d(\mathrm{e}^{j\omega T}) = \frac{1}{T} Y_{\mathrm{e}}(j\omega). \tag{7.7}$$

The second step is to derive the relationship between the discrete-time Fourier transform of $u_d[k]$ and the continuous-time Fourier transform $u(t) = H(u_d)$. You can think of this as the frequency domain action of the zero order hold.

Lemma 7.B.2. *The Fourier transforms of $u(t)$ and $u_d[k]$ are related by*

$$U(j\omega) = \frac{1 - \mathrm{e}^{-j\omega T}}{j\omega} U_d(\mathrm{e}^{j\omega T}). \tag{7.8}$$

Note that the term multiplying U_d is $TR(j\omega)$ where $R(j\omega)$ is the TF in (4.5) from Section 4.4.1 with $s = j\omega$. Now we put these two results together.

Theorem 7.B.3. *The frequency responses $P(j\omega)$ and $P_d[\mathrm{e}^{\omega T}]$ are related by*

$$P_d[\mathrm{e}^{j\omega T}] = \sum_{k=-\infty}^{\infty} P(j\omega + jk\omega_{\mathrm{s}}) R(j\omega + jk\omega_{\mathrm{s}}). \tag{7.9}$$

Proof. Let $u_d[k]$ be the unit impulse. Then $U_d[\mathrm{e}^{j\omega T}] = 1$ and

$$U(j\omega) = TR(j\omega) \qquad \text{(by Lemma 7.B.2)}.$$

Of course, $Y(j\omega) = P(j\omega)U(j\omega)$ so

$$Y(j\omega) = TP(j\omega)R(j\omega).$$

Since the input u_d was an impulse, the output $y_d[k]$ is the impulse response and so

$$Y_d[\mathrm{e}^{j\omega T}] = P_d[\mathrm{e}^{j\omega T}] = \frac{1}{T} Y_\mathrm{e}(j\omega) \qquad \text{(by Lemma 7.B.1).}$$

Now plug in the expression we derived for $Y(j\omega)$ into Y_e to get the result. ■

Thus, at each frequency ω, $P_d[\mathrm{e}^{j\omega T}]$ depends not only on $P(j\omega)$, but also on all the values

$$P(j\omega + jk\omega_\mathrm{s}), \qquad k \in \mathbb{Z}.$$

In the ideal case where $P(j\omega)$ is bandlimited on the interval $(-\omega_\mathrm{N}, \omega_\mathrm{N})$, that is $P(j\omega) = 0$ for $|\omega| \geq \omega_\mathrm{N}$, then

$$P_d[\mathrm{e}^{j\omega T}] = P(j\omega)R(j\omega), \qquad -\omega_\mathrm{N} < \omega \leq \omega_\mathrm{N}.$$

In particular, at low frequencies, $P_d[\mathrm{e}^{j\omega T}] \approx P(j\omega)$.

Chapter 8

Discrete-time control design I: Transfer functions

The method here is to discretize the plant and design the controller in the discrete-time domain. This approach is sometimes called *direct design*.

Contents

8.1 Introduction

We work with the unity-feedback sampled-data control system in Figure 8.1. From the controller's point of

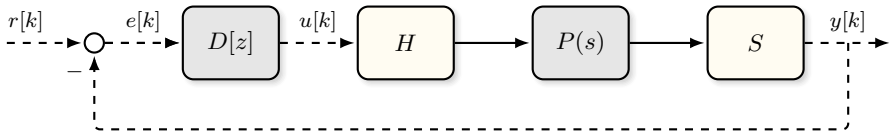


Figure 8.1: Sampled-data unity feedback control system for direct design.

view, a sampled-data system is a purely discrete-time system as shown[1] in Figure 8.2.

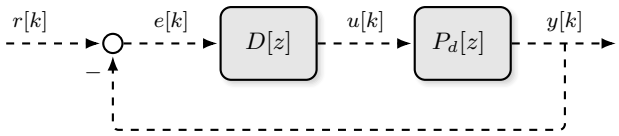


Figure 8.2: A sampled data system as seen by the controller where $P_d[z] = \mathsf{c2d}(P(s))$.

8.2 Pole placement using polynomials

Pole placement for discrete-time systems is almost exactly the same as for continuous-time systems. Write

$$P_d[z] = \frac{N_p[z]}{D_p[z]} = \frac{b_0 z^n + b_1 z^{n-1} + \cdots + b_{n-1} z + b_n}{z^n + a_1 z^{n-1} + \cdots + a_{n-1} z + a_n}$$

[1] This figure doesn't show any disturbances but we will at times consider disturbance signals during controller design.

where N_p and D_p are coprime. The controller is real rational and proper

$$D[z] = \frac{N_c[z]}{D_c[z]} = \frac{g_1 z^{n-1} + g_2 z^{n-2} + \cdots + g_{n-1} z + g_n}{f_1 z^{n-1} + f_2 z^{n-2} + \cdots + f_{n-1} z + f_n}.$$

The closed-loop ch.p. is

$$\pi[z] = D_p[z] D_c[z] + N_p[z] N_c[z]$$

which has order $2n-1$. Choose $2n-1$ desired pole locations $\{\xi_1, \ldots, \xi_{2n-1}\} \subset \mathbb{D}_g$ where $\mathbb{D}_g$ is the "good region" of the z-plane which is always contained in the open unit disk. The desired pole locations generate a desired closed-loop characteristic polynomial

$$\begin{aligned} \pi_{\text{des}}[z] &= \prod_{i=1}^{2n-1} (z - \xi_i) \\ &=: z^{2n-1} + \alpha_{2n-2} z^{2n-2} + \cdots + \alpha_1 z + \alpha_0. \end{aligned}$$

So the problem is, given N_p, D_p, $\pi_{\text{des}} \in \mathbb{R}[z]$, find polynomials N_c, D_c such that

$$D_p[z] D_c[z] + N_p[z] N_c[z] = \pi_{\text{des}}[z].$$

This is the exact same problem we solved in Chapter 3 except we've just replaced the symbol s with the symbol z. Therefore we get the same design equation (3.6) which we re-write here for convenience

$$\begin{bmatrix} 1 & 0 & \cdots & 0 & b_0 & \cdots & \cdots & 0 \\ a_1 & 1 & & \vdots & b_1 & b_0 & & \vdots \\ \vdots & \ddots & \ddots & \vdots & \vdots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & 1 & \vdots & & \ddots & b_0 \\ a_n & \cdots & \cdots & a_1 & b_n & \cdots & \cdots & b_1 \\ 0 & \ddots & & \vdots & 0 & \ddots & & \vdots \\ \vdots & & \ddots & \vdots & \vdots & & \ddots & \vdots \\ 0 & \cdots & \cdots & a_n & 0 & \cdots & \cdots & b_n \end{bmatrix} \begin{bmatrix} f_1 \\ \vdots \\ \vdots \\ f_n \\ g_1 \\ \vdots \\ \vdots \\ g_n \end{bmatrix} = \begin{bmatrix} 1 \\ \alpha_{2n-2} \\ \vdots \\ \alpha_n \\ \alpha_{n-1} \\ \vdots \\ \alpha_0 \end{bmatrix}. \tag{8.1}$$

Pole placement does not guarantee that the zeros of the closed-loop system are placed in desirable locations which can limit its practical use. One issue that isn't clear, which we address next, is how to pick the desired pole locations in the z-plane.

8.2.1 Picking desired z-plane pole locations

We have seen that the step-invariant transformation maps a pole at $s = \lambda$ of $P(s)$ (eigenvalue of A) to a pole at $z = \xi$ of $P_d[z]$ (eigenvalue of A_d) via $\lambda \mapsto \mathrm{e}^{T\lambda}$. If we define $\mathbb{C}_g$ in the s-plane using the techniques from Chapter 3, then c2d maps it to the "good region" in the z-plane

$$\mathbb{D}_g \coloneqq \left\{ z \in \mathbb{C} : z = \mathrm{e}^{sT}, s \in \mathbb{C}_g \right\}.$$

Since $\mathbb{C}_g$ is very easy to obtain, we'll use the following procedure to pick desired z-plane pole locations.

1. Use specifications like $T_{\text{s}}^{\max}$, $\Phi_{\text{pm}}^{\min}$, $\%\text{OS}^{\max}$ to define a good region $\mathbb{C}_g$ of the s-plane.
2. Choose $2n-1$ desired pole locations $\{\lambda_1, \ldots, \lambda_{2n-1}\} \in \mathbb{C}_g$. Ensure that there are two dominant complex conjugate poles so that the closed-loop response behaves like an underdamped second order system.
3. The desired pole locations in the z-plane are given by $\xi_1 \coloneqq \mathrm{e}^{T\lambda_1}$, …, $\xi_{2n-1} \coloneqq \mathrm{e}^{\lambda_{T2n-1}}$.

Example 8.2.1. (Picking desired z-plane pole locations) Consider the sampled-data control system in Figure 8.2 with a 3rd order plant and sampling period $T = 0.1$. The given closed-loop specifications are (i) stability (ii) settling time for step inputs less than or equal to 3 seconds (iii) phase margin of at least $50°$.

The settling time specification yields the constraint

$$\{s \in \mathbb{C} : \mathrm{Re}\,(s) < -4/3\}\,.$$

Converting the phase margin specification into a damping ratio specification using (3.4) we get that $\zeta \geq 0.5$. Converting this into an overshoot specification using (2.8) and finally into an angle constraint $\theta^{\max} = \pi/3$ using (2.9), we get the constraint

$$\{s \in \mathbb{C} : |\arg(s)| \geq 2\pi/3\}\,.$$

So

$$\mathbb{C}_g = \{s \in \mathbb{C} : \mathrm{Re}\,(s) < -4/3\} \cap \{s \in \mathbb{C} : |\arg(s)| \geq 2\pi/3\}\,.$$

Now pick $2n - 1 = 5$ desired pole locations in $\mathbb{C}_g$. We pick two complex conjugate dominant poles so that the total response is approximately second order and the rest are chosen further to the left. Let's say

$$\{\lambda_1, \lambda_2, \lambda_3, \lambda_4, \lambda_5\} = \{-5 + j, -5 - j, -20, -21, -22\}\,.$$

We map these to desired z-plane locations via

$$\{\xi_1, \xi_2, \xi_3, \xi_4, \xi_5\} = \left\{\mathrm{e}^{-0.5}\mathrm{e}^{j0.1}, \mathrm{e}^{-0.5}\mathrm{e}^{-j0.1}, \mathrm{e}^{-2}, \mathrm{e}^{-2.1}, \mathrm{e}^{-2.2}\right\}.$$

▲

8.2.2 Step tracking

Suppose that we also have a specification on the steady-state tracking error of the closed-loop system

$$|e_{\mathrm{ss}}| = \lim_{t\to\infty} |e(t)| = \lim_{t\to\infty} |r(t) - y(t)| < e_{\mathrm{ss}}^{\max}.$$

In direct design the sampled-data system is treated as a discrete-time system so we'll aim for

$$\lim_{k\to\infty} |r(kT) - y(kT)| = \lim_{k\to\infty} |r[k] - y[k]| < e_{\mathrm{ss}}^{\max}.$$

We can incorporate tracking error specifications into pole placement at the expense of increased controller order.

Asymptotic step tracking

Suppose we have a stable unity feedback sampled-data control system. Then the steady-state tracking error for a step input is[2]

$$\begin{aligned} e_{\mathrm{ss}} &= \lim_{k\to\infty} e[k] = \lim_{z\to 1}(z-1)E[z] \qquad \text{(FVT applies since closed-loop system is stable)} \\ &= \lim_{z\to 1}(z-1)\frac{1}{1+P_dD}\frac{z}{z-1} \\ &= \lim_{z\to 1}\frac{1}{1+P_dD}. \end{aligned}$$

Therefore $e_{\mathrm{ss}} = 0$ if, and only if, P_dD has a pole at $z = 1$. We can modify the pole placement design approach to achieve asymptotic step tracking as follows:

Case 1 If $P_d[z]$ has a pole at $z = 1$, then we can use regular pole placement.

[2] Here we are using the discrete-time final-value theorem 6.2.1.

Case 2 If $P_d[z]$ has a zero at $z = 1$, then asymptotic tracking is not possible.

Case 3 If $P_d[z]$ has neither a pole at $z = 1$ nor a zero at $z = 1$, choose $D[z] = D_1[z]/(z-1)$ and design $D_1[z]$ using pole placement for the augmented plant $P_d[z]/(z-1)$.

A drawback of this approach is that the complexity of $D[z]$ increases. The final controller $D[z] = D_1[z]/(z-1)$ will have order $n+1$.

Remark 8.2.1. If the continuous-time plant $P(s)$ has a pole at $s = 0$ then its discretization $P_d[z] = \mathsf{c2d}(P(s))$ will have a pole at $z = \mathrm{e}^{0T} = 1$. ♦

Practical step tracking

Suppose that our specifications are less stringent and we only require that $|e_{\text{ss}}| \leq e_{\text{ss}}^{\max}$, $e_{\text{ss}}^{\max} \neq 0$. If the closed-loop system is stable and $P_d[z]$ has no poles or zeros at $z = 1$, then

$$|e_{\text{ss}}| = \left| \frac{1}{1 + P_d[1]D[1]} \right|$$

and e_{ss} depends on the steady-state gain of the controller $D[1]$. If we increase the order of $D[z]$, then we can adjust $D[1]$ while still being able to place the closed-loop poles. The most straightforward change is to increase the order of the controller by one

$$D[z] = \frac{g_0 z^n + g_1 z^{n-2} + \cdots + g_{n-1} z + g_n}{f_0 z^n + f_1 z^{n-1} + \cdots + f_{n-1} z + f_n}.$$

The tracking error constraint is satisfied if

$$\frac{1}{1 + P_d[1]D[1]} = e_{\text{ss}}^{\max} \qquad \Longleftrightarrow \qquad P_d[1]D[1] = \frac{1}{e_{\text{ss}}^{\max}} - 1.$$

Let δ be any real number that satisfies

$$\delta \geq \frac{1}{e_{\text{ss}}^{\max}} - 1.$$

Then the tracking error specification is satisfied if

$$P_d[1]D[1] = \delta. \tag{8.2}$$

This is a linear equation of the unknown controller parameters g_i, f_i. We add the linear equation (8.2) to our pole placement design equations to obtain

$$\left[\begin{array}{cccccccc} 1 & 0 & \cdots & 0 & b_0 & \cdots & \cdots & 0 \\ a_1 & 1 & & \vdots & b_1 & b_0 & & \vdots \\ \vdots & \ddots & \ddots & \vdots & \vdots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & 1 & \vdots & & \ddots & b_0 \\ a_n & \cdots & \cdots & a_1 & b_n & \cdots & \cdots & b_1 \\ 0 & \ddots & & \vdots & 0 & \ddots & & \vdots \\ \vdots & & \ddots & \vdots & \vdots & & \ddots & \vdots \\ 0 & \cdots & \cdots & a_n & 0 & \cdots & \cdots & b_n \\ \hline -\delta & \cdots & \cdots & -\delta & P_d[1] & \cdots & \cdots & P_d[1] \end{array}\right] \left[\begin{array}{c} f_0 \\ \vdots \\ \vdots \\ f_n \\ g_0 \\ \vdots \\ \vdots \\ g_n \end{array}\right] = \left[\begin{array}{c} 1 \\ \alpha_{2n-1} \\ \alpha_{2n-2} \\ \vdots \\ \vdots \\ \vdots \\ \alpha_1 \\ \hline \alpha_0 \\ 0 \end{array}\right]. \tag{8.3}$$

Example 8.2.2. We re-solve the design problem from Example 3.3.2 but this time in discrete-time with sampling period $T = 0.05$. The design specifications are:

(i) closed-loop stability,

(ii) maximum step tracking error of 5%,

(iii) step response satisfies

- overshoot less than or equal to 20%,
- settling time less than or equal to 4 seconds.

The plant is given by

$$P(s) = \frac{1}{s^2 + s + 0.25}, \qquad P_d[z] = \mathsf{c2d}(P(s)) = \frac{0.001229z + 0.001209}{z^2 - 1.951z + 0.9512}.$$

We already picked desired s-plane pole locations in Example 3.3.2. We use the same ones in this example so that $\{\lambda_1, \lambda_2, \lambda_3, \lambda_4\} = \{-1 + j1.95, -1 - j1.95, -5 - 5\}$. The corresponding desired z-plane pole locations are

$$\{\xi_1, \xi_2, \xi_3, \xi_4, \xi_5\} = \left\{\mathrm{e}^{-0.05+j0.0975}, \mathrm{e}^{-0.05-j0.0975}, \mathrm{e}^{-0.25}, \mathrm{e}^{-0.25}\right\}$$

and the desired closed-loop ch.p.

$$\pi_{\text{des}}[z] = (z - \xi_1)(z - \xi_2)(z - \xi_3)(z - \xi_4) = z^4 - 3.451z^3 + 4.4606z^2 - 2.5578z + 0.5488.$$

Next we incorporate the tracking error specification

$$\delta = \frac{1}{e_{\text{ss}}^{\max}} - 1 = 19.$$

The resulting design equations are

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ -1.9506 & 1 & 0 & 0.0012 & 0 & 0 \\ 0.9512 & -1.9506 & 1 & 0.0012 & 0.0012 & 0 \\ 0 & 0.9512 & -1.9506 & 0 & 0.0012 & 0.0012 \\ 0 & 0 & 0.9512 & 0 & 0 & 0.0012 \\ -19 & -19 & -19 & 4 & 4 & 4 \end{bmatrix} \begin{bmatrix} f_0 \\ f_1 \\ f_2 \\ g_0 \\ g_1 \\ g_2 \end{bmatrix} = \begin{bmatrix} 1 \\ -3.451 \\ 4.4606 \\ -2.5578 \\ 0.5488 \\ 0 \end{bmatrix}.$$

Solving this equation for the controller gains yields

$$D[z] = \frac{13.74z^2 - 24.42z + 10.9}{z^2 - 1.52z + 0.56}. \tag{8.4}$$

Figure 8.3 shows the closed-loop step response. This figure shows both the sampled-data and discrete-time responses. As expected the responses match at the sampling instances. ▲

8.3 Control design in the frequency domain

It is hard to do direct design in the frequency domain because discrete-time Bode plots are messy. In particular we cannot use the piecewise linear approximations that make continuous-time Bode plots easy to sketch. One ramification of this is that the design equations from your analog control course don't easily map over. We get around this problem by using the bilinear transformation, as we did in Section 6.8.2, so that the discrete-time

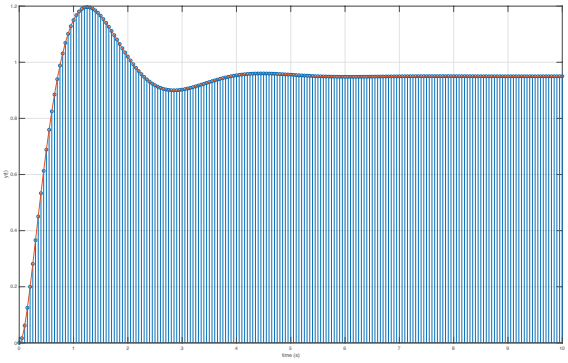


Figure 8.3: Closed-loop step response for Example 8.2.2 with control law (8.4)

design problem looks like a continuous-time design problem and so that we can use our well-known design equations.

Bring in the bilinear transformation

$$v = \frac{z-1}{z+1}, \qquad z = \frac{1+v}{1-v}.$$

The design methodology is to take a discrete-time plant $P_d[z]$ and then transform to the v-plane via

$$P_d(v) \coloneqq P_d\left[\tfrac{1+v}{1-v}\right]. \tag{8.5}$$

We apply our continuous-time design techniques to $P_d(v)$ and design $C(v)$. Finally we convert $C(v)$ back to discrete-time via

$$D[z] = C\left(\frac{z-1}{z+1}\right). \tag{8.6}$$

Why is this a good strategy? First, the bilinear mapping preserves stability of the closed-loop systems since we are applying to both the plant and the controller. Second, the bilinear transformation preserves steady-state calculations: Suppose we have designed $C(v)$ to meet, say, a step tracking specification for the plant (8.5). The steady-state tracking error in the v-domain is

$$\lim_{v\to 0} vE(v) = \lim_{v\to 0} v\frac{1}{1+P_d(v)C(v)}\frac{1}{v} = \lim_{v\to 0}\frac{1}{1+P_d(v)C(v)}.$$

Now take $C(v)$ and map it back to the z-domain via (8.6). In the z-domain the steady-state tracking error is

$$e_{\text{ss}} = \lim_{z\to 1}(z-1)E[z] = \lim_{z\to 1}(z-1)\frac{1}{1+D[z]P_d[z]}\frac{z}{z-1} = \lim_{z\to 1}\frac{1}{1+D[z]P_d[z]}.$$

Since the bilinear transformation maps $v=0$ to $z=1$, these quantities are equal.

Third, the bilinear transformation gives a simple relationship between the gain and phase crossover frequencies of $C(v)P_d(v)$ and those of $D[z]P_d[z]$. Let $\theta \in [0,\pi)$ and set $z = \mathrm{e}^{j\theta}$. Then

$$\begin{aligned} v &= \frac{\mathrm{e}^{j\theta}-1}{\mathrm{e}^{j\theta}+1} \\ &= \frac{\mathrm{e}^{j\theta/2}-\mathrm{e}^{-j\theta/2}}{\mathrm{e}^{j\theta/2}+\mathrm{e}^{-j\theta/2}} \\ &= \frac{2j\sin(\theta/2)}{2\cos(\theta/2)} \\ &= j\tan(\theta/2). \end{aligned}$$

So frequencies in the z-domain get mapped to frequencies in the v-domain via $\omega = \tan(\theta/2)$ and therefore

$$C(j\tan(\theta/2))P(j\tan(\theta/2)) = D[\mathrm{e}^{j\theta}]P_d[\mathrm{e}^{j\theta}].$$

This means that, for example, if $D[z]P_d[z]$ has gain crossover frequency θ_{gc} then $C(v)P_d(v)$ has gain crossover frequency

$$\omega_{\mathrm{gc}} = \tan\left(\frac{\theta_{\mathrm{gc}}}{2}\right).$$

Similarly with the phase crossover frequency θ_{pc}. This suggests the following frequency domain design procedure

Design procedure

The given continuous-time specifications are:

(a) Increase phase margin to be at least equal to $\Phi_{\mathrm{pm}}^{\mathrm{des}}$ (determined by damping ratio or robustness requirements).

(b) *One* of the following:

(i) steady-state specification (determined by tracking or disturbance rejection requirement), or

(ii) a desired closed-loop bandwidth (determined by time-domain specifications or noise rejection requirements).

Design a digital controller $D[z]$ as follows.

1. Pick T as small as is feasible.
2. Compute $P_d[z] = \mathsf{c2d}(P(s))$.
3. Convert to the v-domain via
$$P_d(v) \coloneqq P_d\left[\frac{1+v}{1-v}\right].$$
4. Design a lead, lag or lead-lag controller $C(v)$ for $P_d(v)$ to meet the continuous-time specifications.
5. Convert $C(v)$ to the z-domain
$$D[z] = C\left(\frac{z-1}{z+1}\right).$$
6. Simulate the sampled-data system using the discrete-time controller from the previous step.

Consult Appendices 2.A 2.B and 2.C for, respectively, continuous-time lag, lead and lead-lag design procedures.

Example 8.3.1. Consider the system in Figure 8.1 with plant

$$P(s) = \frac{1}{s^2 + 2s + 0.75}$$

and fixed sampling period $T = 0.05$. Design $D[z]$ so that the closed-loop system is stable and

(i) steady-state step tracking error of no more than 5%,

(ii) phase margin of at least 60 degrees.

We apply the step-invariant transformation to get

$$P_d[z] = \mathsf{c2d}(P(s)) = 0.012\frac{(z+0.9672)}{z^2-1.903z+0.9048}.$$

Convert this model to the v-plane using the bilinear transformation

$$P_d(v) = P_d\left[\frac{1+v}{1-v}\right] = -1.04\times 10^{-5}\frac{(v-1)(v+60)}{v^2+0.05v+0.0005}.$$

The above steps are easily done in MATLAB.

```
P = tf([0 0 1], [1 2 0.75]); % ct plant
T = 0.05; % sampling period
Pd = c2d(P, T); % step-invariant discretization
[n, d] = tfdata(Pd);
Pdtemp = tf(n, d, 2);
Pv = d2c(Pdtemp, 'tustin'); % P_d(v)
```

We'll design a lead controller

$$C(v) = K\frac{\alpha Tv+1}{Tv+1}, \qquad \alpha>1,\ T>0,\ K>0$$

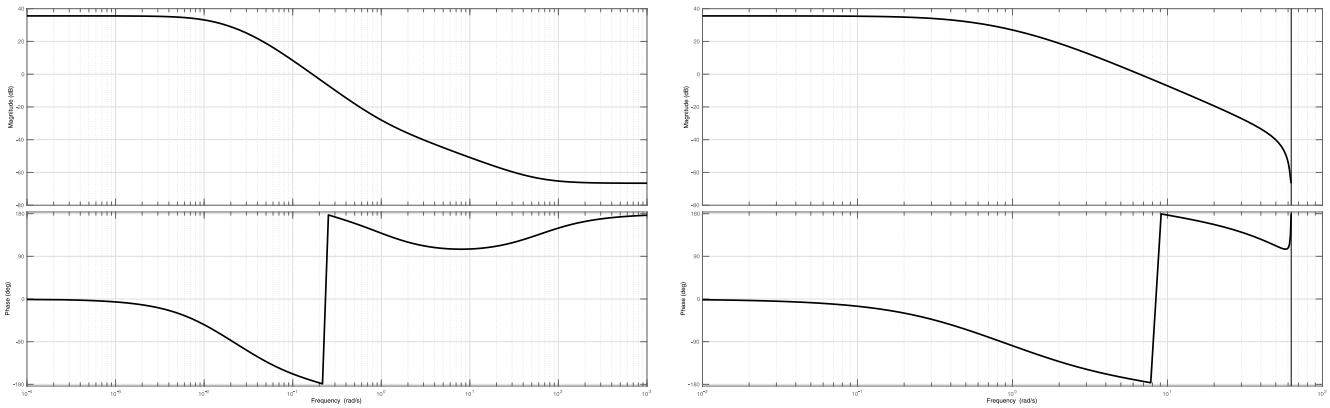
using the procedure in Appendix 2.B. Define $\hat{K} = K\sqrt{\alpha}$. From the steady-state specification we get

$$\lim_{v\to 0}\frac{1}{1+\hat{K}P_d(v)} = \frac{1}{1+\hat{K}P_d(0)} \le 0.05$$

which yields the constraint

$$\hat{K} \ge 14.25.$$

Boost $\hat{K}$ by 10dB (a guess) to account for the magnitude distortion from α: $\hat{K} = 14.25\sqrt{10} = 45.06$. The Bode plots of $\hat{K}P_d(v)$ as well as $\hat{K}P_d[z]$ are shown in Figure 8.4. From the Bode plot of $\hat{K}P_d(v)$ in Figure 8.4a we



(a) $\hat{K}P_d(v)$. *(b)* $\hat{K}P_d[z]$.

Figure 8.4: The Bode plots $KP(v)$ and $KP_d[z]$ with $K = 15$ in Example 8.3.1.

find that phase margin and the the gain crossover frequency are, respectively

$$\Phi_{\text{pm}} = 7.7^\circ, \qquad \omega_{\text{gc}} = 0.17 \text{ rad/s}.$$

As expected, this corresponds to $\theta_{\text{gc}} = 2\arctan(\omega_{\text{gc}}) = 0.3368$ radians in Figure 8.4b. Set $\omega_m = \omega_{\text{gc}} = 0.17$ rad/s. To meet the spec we need to add

$$\phi_{\max} = \Phi_{\text{pm}}^{\text{des}} - \Phi_{\text{pm}} = 60^\circ - 7.7^\circ = 52.3^\circ.$$

Using (2.21) we get

$$\alpha = \frac{1+\sin(\phi_{\max})}{1-\sin(\phi_{\max})} = 8.6.$$

This yields

$$K = \frac{\hat{K}}{\sqrt{\alpha}} = 15.37.$$

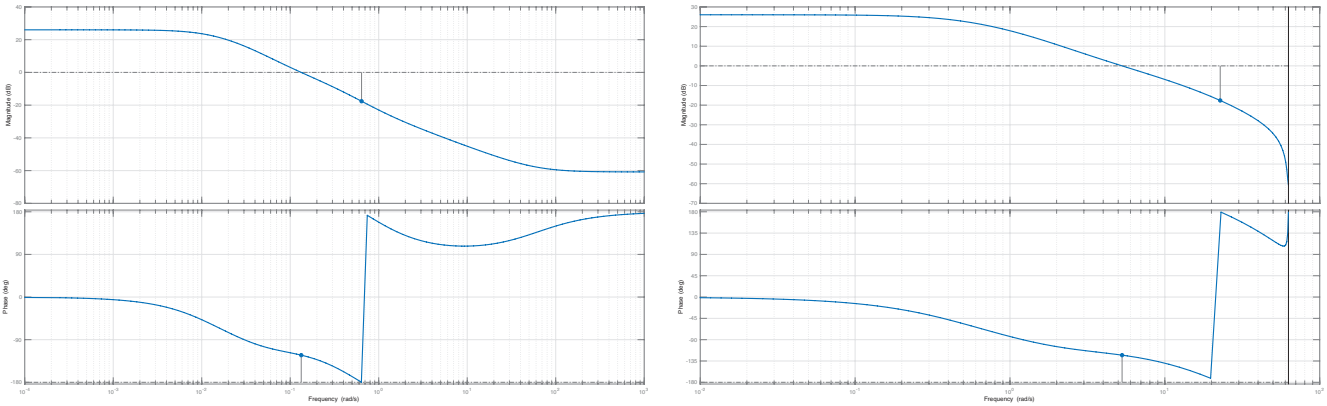
Finally, use (2.19) to get

$$T = \frac{1}{\omega_m \sqrt{\alpha}} = 2.05.$$

The controller is

$$C(v) = K\frac{\alpha T v + 1}{Tv+1} = \frac{270.4v + 15.37}{2.047v+1} = 132.1\frac{(v+0.057)}{v+0.49}.$$

The Bode plot of the loop TF $C(v)P_d(v)$ is shown in Figure 8.5a from which we get $\Phi_{\text{pm}} = 60°$ and $\omega_{\text{gc}} = 0.17$ rad/s. We now convert the controller back to the z-plane for implementation



(a) $C(v)P(_dv)$. *(b)* $D[z]P_d[z]$.

Figure 8.5: The Bode plots $C(v)P(v)$ and $D[z]P_d[z]$ in Example 8.3.1.

$$D[z] = C\left(\frac{z-1}{z+1}\right) = \frac{93.8z - 83.71}{z - 0.3436}. \tag{8.7}$$

The Bode plot of the discrete-time loop TF $D[z]P_d[z]$ is shown in Figure 8.5b from which we get $\Phi_{\text{pm}} = 60°$ at $\theta_{\text{gc}} = 2\arctan(0.17) = 2.84$ radians as expected. The following code converts back to discrete-time.

```
C = K*tf([alpha*T 1], [T 1]); % C(v)
D = c2d(C, 2, 'tustin'); % bilinear tsfm with T = 2
[n, d] = tfdata(D);
Ts = 0.05; % actual sampling period
D = tf(n, d, Ts); % convert to correct sampling period
```

The closed-loop step response of the sample-data as well as the discrete-time system are shown in Figure 8.6. ▲

Remark 8.3.1. It might appear that our frequency domain design procedure for discrete-time systems is equivalent to design by emulation. That is not the case. Here we are first discretizing the plant so that the sampling period is taken into account in the design. We are only using the mathematical properties of the bilinear transformation to allow us to use convenient frequency domain design formulae. We are not "un-discretizing" the plant.

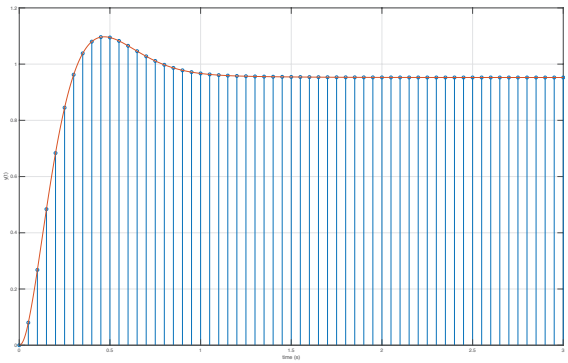


Figure 8.6: Closed-loop step response for Example 8.3.1 with control law (8.7)

Chapter 9

Discrete-time control design II: State-space

We continue our study of control design in discrete-time. We design controllers in the time-domain using state-space models. The main topics are: the stabilization problem, observers and output-feedback stabilization, and the problem of tracking a reference and/or rejecting a disturbance.

Contents

9.1 The stabilization problem

Consider the system

$$x^+ = Ax + Bu.$$

We regard 0 as the desired equilibrium point for the state x. Assume that the state $x[k]$ is available from a sensor for all k. We search for a **state-feedback controller** $u[k] = Fx[k]$, F a real matrix. Then the controlled system is

$$x^+ = Ax + Bu = Ax + BFx = (A + BF)x.$$

The block diagram is shown in Figure 9.1.

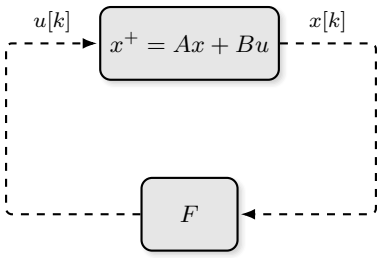


Figure 9.1: Stabilization problem.

Stabilization Problem. *Given (A, B), find F so that $A + BF$ is stable.*

To solve this problem, we introduce the notion of controllability.

9.1.1 Controllability

Consider

$$x^+ = Ax + Bu, \qquad x[0] = 0. \tag{9.1}$$

For historical reasons, controllability is defined initially as a reachability property. Reachability isn't an important control objective *per se*, but it does capture in a simple way the concept of control authority.

Definition 9.1.1. System (9.1) is **controllable** (or the pair (A, B) is **controllable**) if for every target vector x_f, there is a time $k_f \geq 0$ and a control signal $u[k]$, $k \in \{0, 1, \ldots, k_f - 1\}$, such that $x[k_f] = x_f$.

Controllability says that every state is reachable starting from the origin. Let's characterize which states can be reached starting from the origin. First

$$x[1] = Bu[0].$$

Thus, at time $k = 1$, the state can reach any vector in the column span of the matrix B and no other vector. Then

$$x[2] = Ax[1] + Bu[1] = ABu[0] + Bu[1].$$

Thus, at time $k = 2$, the state can reach any vector in the column span of $\begin{bmatrix} B & AB \end{bmatrix}$, and no other vector. Continuing in this way, define the **controllability matrix**

$$W_c \coloneqq \begin{bmatrix} B & AB & \cdots & A^{n-1}B \end{bmatrix}. \tag{9.2}$$

Thus W_c has n rows and nm columns where n is the dimension of x and m is the dimension of u. So the columns of W_c are vectors in the state space $\mathbb{R}^n$. The span of the columns of W_c are exactly the set of vectors that are reachable. Thus, (A, B) is controllable if and only if the column span of W_c equals the entire state space $\mathbb{R}^n$, and this is true if, and only if, the rank of W_c equals n. In the single-input case the matrix W_c is square and the system is controllable if, and only if, W_c is nonsingular.

It is convenient to say that an eigenvalue λ of A is *controllable* if

$$\operatorname{rank} \begin{bmatrix} A - \lambda I & B \end{bmatrix} = n.$$

This rank test is called the[1] **PBH test**. Then, another test for controllability is: (A, B) is controllable if, and only if, each eigenvalue of A is controllable.

Summary: controllability

every state is reachable $\Leftrightarrow$ (A, B) is controllable
$\Leftrightarrow$ $\operatorname{rank} W_c = n$
$\Leftrightarrow$ $\operatorname{rank} \begin{bmatrix} B & AB & A^2B & \cdots & A^{n-1}B \end{bmatrix} = n$
$\Leftrightarrow$ $\operatorname{rank} \begin{bmatrix} A - \lambda I & B \end{bmatrix} = n$ for every eigenvalue λ of A.

Example 9.1.1.

$$A = \begin{bmatrix} 2 & 0 & 2 \\ 3 & 1 & 0 \\ 1 & 4 & 1 \end{bmatrix}, \qquad B = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \qquad W_c = \begin{bmatrix} B & AB & A^2B \end{bmatrix} = \begin{bmatrix} 0 & 2 & 6 \\ 0 & 0 & 6 \\ 1 & 1 & 3 \end{bmatrix}.$$

You can check that $\det(W_c) = 12$ which shows that $\operatorname{rank}(W_c) = 3$ and the pair (A, B) is controllable. ▲

[1] Named after Popov, Belevitch and Hautus.

Remark 9.1.2. The controllability matrix (9.2) can be computed via the MATLAB command `ctrb`. The code below performs all the calculations of Example 9.1.1.

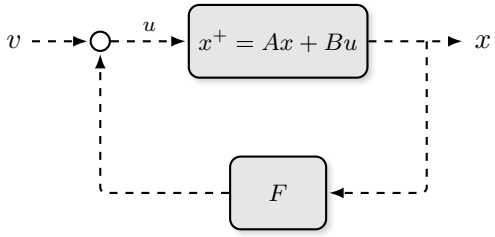
```
A = [2 0 2; 3 1 0; 1 4 1];
B = [0;0;1];
Wc = ctrb(A,B);
det(Wc)
```

♦

The most important fact about controllable systems is that their eigenvalues can be re-assigned by state feedback. That is, consider applying the control signal

$$u = Fx + v, \qquad v \text{ is an external input}$$

to the system (9.1). Here $F \in \mathbb{R}^{m\times n}$ and $v(t) \in \mathbb{R}^m$ is a new independent input.



The new state-space model is

$$x^+ = (A + BF)x + Bv$$

and (A, B) is transformed to $(A + BF, B)$. We have the famous Pole-Assignment Theorem.

Theorem 9.1.3 (Pole-Assignment Theorem)**.** *The pair (A, B) is controllable if, and only if, for every set of desired eigenvalues, there exists a matrix F such that $A + BF$ has exactly that set of eigenvalues.*

Remark 9.1.4. Theorem 9.1.3 is called the "pole placement" theorem in reference to the fact that the eigenvalues of $A + BF$ are the poles of the closed-loop system transfer function

$$\left[\begin{array}{c|c} A+BF & B \\ \hline I & 0 \end{array}\right](z) = (zI - A - BF)^{-1}B.$$

Here the system output is taken to be the entire state vector. ♦

9.1.2 A procedure for computing state feedback matrices

Next we discuss a procedure for designing the state feedback matrix F which assigns the eigenvalues of $A+BF$. In Section 6.5.1 we saw that we can go from any proper discrete-time TF to a state-space model given by the matrices (6.8). For convenience we re-write (A, B):

$$A = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ldots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 0 & 1 \\ -a_n & -a_{n-1} & -a_{n-2} & \cdots & -a_2 & -a_1 \end{bmatrix}, \qquad B = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix}. \tag{9.3}$$

A matrix of the form of A above is called a **companion matrix**. Its characteristic polynomial is

$$\det(zI - A) = z^n + a_1 z^{n-1} + a_2 z^{n-2} + \cdots + a_{n-1} z + a_n.$$

When A is a companion matrix and B is as above, then the pair (A, B) is said to be in **controllable canonical form**. A pair (A, B) in controllable canonical form is controllable.

Example 9.1.2.

$$A = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -a_3 & -a_2 & -a_1 \end{bmatrix}, \qquad B = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \qquad W_c = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & -a_3 \\ 1 & -a_3 & -a_2 + a_3^2 \end{bmatrix}.$$

The columns of W_c span $\mathbb{R}^3$ so the pair is controllable. ▲

It is easy to design F when (A, B) are in controllable canonical form, i.e., have the form (9.3), as the next example illustrates.

Example 9.1.3. (Pole placement for a pair in controllable canonical form) Design a state feedback matrix $F = \begin{bmatrix} f_1 & f_2 & f_3 \end{bmatrix}$ for the pair

$$A = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & -1 & -1 \end{bmatrix}, \qquad B = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

that places the eigenvalues of $A + BF$ at $\{-0.1, -0.2, -0.3\}$. We have

$$A + BF = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & -1 & -1 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \begin{bmatrix} f_1 & f_2 & f_3 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 + f_1 & -1 + f_2 & -1 + f_3 \end{bmatrix}.$$

Since $A + BF$ is a companion matrix its characteristic polynomial can be obtained by inspecting the last row

$$z^3 + (1 - f_3)z^2 + (1 - f_2)z + (-1 - f_1).$$

The desired ch.p. is

$$(z + 0.1)(z + 0.2)(z + 0.3) = z^3 + 0.6z^2 + 0.11z + 0.006.$$

Equating coefficients in these two polynomials gives the linear equation

$$\begin{bmatrix} 0 & 0 & -1 \\ 0 & -1 & 0 \\ -1 & 0 & 0 \end{bmatrix} \begin{bmatrix} f_1 \\ f_2 \\ f_3 \end{bmatrix} = \begin{bmatrix} -0.4 \\ -0.89 \\ 1.006 \end{bmatrix}$$

which has the unique solution

$$F = \begin{bmatrix} -1.006 & 0.89 & 0.4 \end{bmatrix}.$$

▲

It turns out that if, and only if, (A, B) is controllable and B is $n \times 1$, there exists a coordinate change which transforms (A, B) into controllable canonical form.

Theorem 9.1.5. *Suppose (A, B) is controllable and B is $n \times 1$. Let the ch.p. of A be given by*

$$z^n + a_1 z^{n-1} + a_2 z^{n-2} + \cdots + a_{n-1} z + a_n.$$

Define

$$\tilde{A} \coloneqq \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \cdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 0 & 1 \\ -a_n & -a_{n-1} & -a_{n-2} & \cdots & -a_2 & -a_1 \end{bmatrix}, \qquad \tilde{B} \coloneqq \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix}.$$

Then there exists an invertible matrix $W \in \mathbb{R}^{n \times n}$ such that

$$W^{-1} A W = \tilde{A}, \qquad W^{-1} B = \tilde{B}.$$

The theorem states that if a single-input system is controllable, then A is similar to a companion matrix and B in the transformed coordinates is the nth standard basis vector in $\mathbb{R}^n$. We don't prove this theorem but, given a controllable pair (A, B), Theorem 9.1.5 suggests a procedure for designing the state-feedback matrix that assigns the closed-loop eigenvalues.

1. Compute the controllability matrix W_c of (A, B).
2. Find the ch.p. of A
$$z^n + a_1 z^{n-1} + a_2 z^{n-2} + \cdots + a_{n-1} z + a_n$$
and use it to define $\tilde{A}$ as in Theorem 9.1.5.
3. Let $\tilde{W}_c$ be the controllability matrix of $(\tilde{A}, \tilde{B})$ where $\tilde{B}$ is as in Theorem 9.1.5.
4. Define $W \coloneqq W_c \tilde{W}_c^{-1}$. Then $\tilde{A} = W^{-1} A W$, $\tilde{B} = W^{-1} B$.
5. Compute the unique $\tilde{F} \in \mathbb{R}^{1 \times n}$ so that $\tilde{A} + \tilde{B}\tilde{F}$ has eigenvalues in the desired locations
$$\tilde{F} = \begin{bmatrix} a_n - \alpha_0 & a_{n-1} - \alpha_1 & \cdots & a_1 - \alpha_{n-1} \end{bmatrix}.$$
where the α_i come from the desired ch.p. $z^n + \alpha_{n-1} z^{n-1} + \cdots + \alpha_1 z + \alpha_0$.
6. Set $F = \tilde{F} W^{-1}$.

Example 9.1.4. (Double integrator) In Example 7.2.4 we discretized the double integrator plant to get

$$x^+ = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix} x + T \begin{bmatrix} \frac{T}{2} \\ 1 \end{bmatrix} u.$$

In this example we'll design a state feedback $u = Fx$ so that the closed-loop system has ch.p. $z^2 + \alpha_1 z + \alpha_0$. First check controllability:

$$W_c = \begin{bmatrix} B & AB \end{bmatrix} = T \begin{bmatrix} \frac{T}{2} & \frac{3}{2}T \\ 1 & 1 \end{bmatrix}.$$

Since $\det(W_c) = -T^3 \neq 0$ the system is controllable and we can assign the closed-loop eigenvalues. We now follow the above procedure. The ch.p. of A is

$$(z-1)(z-1) = z^2 - 2z + 1 \qquad \Rightarrow \qquad \tilde{A} = \begin{bmatrix} 0 & 1 \\ -1 & 2 \end{bmatrix}.$$

This yields

$$\tilde{W}_c = \begin{bmatrix} \tilde{B} & \tilde{A}\tilde{B} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 1 & 2 \end{bmatrix} \qquad \Rightarrow \qquad W = W_c \tilde{W}_c^{-1} = T \begin{bmatrix} \frac{T}{2} & \frac{T}{2} \\ -1 & 1 \end{bmatrix}.$$

We now pick $\tilde{F}$ to get the desired eigenvalues for

$$\tilde{A} + \tilde{B}\tilde{F} = \begin{bmatrix} 0 & 1 \\ -1 + \tilde{f}_1 & 2 + \tilde{f}_2 \end{bmatrix}.$$

This is a companion matrix and has ch.p. $z^2 - (2 + f_2)z + 1 - f_1$. Comparing to our desired ch.p we get

$$\tilde{F} = \begin{bmatrix} 1 - \alpha_0 & -\alpha_1 - 2 \end{bmatrix}$$

and therefore

$$F = \tilde{F} W^{-1} = -\frac{1}{2T^2} \begin{bmatrix} 2(\alpha_0 + \alpha_1 + 1) & T(\alpha_1 - \alpha_0 + 3) \end{bmatrix}.$$

▲

Example 9.1.5. (Double integrator - numerical) To be specific, suppose that $T = 0.1$ and the desired closed-loop poles are $\{0.2, 0.5\}$. Then the desired ch.p. is $(z - 0.2)(z - 0.5) = z^2 - 0.7z + 0.1$ so that $\alpha_1 = -0.7$ and $\alpha_0 = 0.1$. Substituting numerical values into the expression for F from Example 9.1.4 we get

$$F = -\begin{bmatrix} 40 & 11 \end{bmatrix}.$$

You can check that the eigenvalues of

$$A + BF = \begin{bmatrix} 0.8 & 0.045 \\ -4 & -0.1 \end{bmatrix}$$

are exactly $\{0.2, 0.5\}$. Figure 9.2 shows how this controller is implemented. ▲

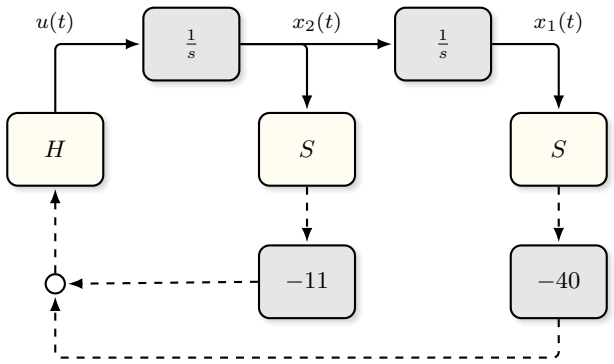


Figure 9.2: Implementing the controller from Example 9.1.5.

Remark 9.1.6. The procedure we've described is tedious by hand for large problems but is well-suited to numerical implementation. The MATLAB commands `place` or `acker` can be used to compute state feedback matrices. The above is easily done in MATLAB using the following program.

```
A = [0 1;0 0]; B =[0;1]; C = [1 0]; D=0;
P = ss(A,B,C,D); %ct plant
T = 0.1; % sampling period
Pd = c2d(P, T); %step-invariant transformation
Ad = Pd.a; Bd=Pd.b; Cd=Pd.c; Dd=Pd.d; % dt state matrices
Xi = [0.2 0.5]; %desired pole locations
F = -place(Ad, Bd, Xi); % feedback matrix
```

A related notion is that of stabilizability.

Definition 9.1.7. System (9.1) is **stabilizable** (or the pair of matrices (A, B) is **stabilizable**) if there exists a matrix F so that the eigenvalues of $A + BF$ are all inside the open unit disk.

Not surprisingly, (A, B) is stabilizable if, and only if, all the (unstable) eigenvalues of A with $|\lambda| \geq 1$ are controllable. By Theorem 9.1.3, if (A, B) is controllable, then this implies that (A, B) is stabilizable. The converse implication does not hold.

Example 9.1.6. Consider a discrete-time system in state-space form with

$$A = \frac{1}{4}\begin{bmatrix} 1 & 3 \\ 3 & 1 \end{bmatrix}, \qquad B = \begin{bmatrix} 1 \\ 1 \end{bmatrix}.$$

If possible, design a state feedback $u = Fx$, so that the closed-loop system is stable. We start by checking controllability. The controllability matrix for this pair is

$$W_c = \begin{bmatrix} B & AB \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}.$$

Since $\det(W_c) = 0$, the pair (A, B) isn't controllable. So we can't arbitrarily assign the eigenvalues of $A+BF$ but we may still be able to stabilize. Let's check stabilizability. The eigenvalues of A are (verify!) $\sigma(A) = \{-0.5, 1\}$. There is one stable eigenvalue -0.5 and one unstable eigenvalue 1. The pair (A, B) is stabilizable if, and only if, all of A's unstable eigenvalues are controllable. We apply the PBH test to the unstable eigenvalue

$$\operatorname{rank}\begin{bmatrix} A - (1)I & B \end{bmatrix} = \operatorname{rank}\begin{bmatrix} -3/4 & 3/4 & 1 \\ 3/4 & -3/4 & 1 \end{bmatrix} = 2.$$

The unstable eigenvalue is controllable and therefore (A, B) is stabilizable. To design the stabilizing feedback, our strategy is to not bother moving the stable eigenvalue at -0.5 — it isn't controllable and can't be moved anyway. Instead, we'll shift the unstable (but controllable) eigenvalue at 1 to, say, 0.1. Our desired ch.p. is

$$\pi_{\text{des}}[z] = \underbrace{(z + 0.5)}_{\text{fixed}}(z - 0.1) = z^2 + 0.4z - 0.05.$$

Taking $F = \begin{bmatrix} f_1 & f_2 \end{bmatrix}$ we have

$$A + BF = \begin{bmatrix} 0.25 + f_1 & 0.75 + f_2 \\ 0.75 + f_1 & 0.25 + f_2 \end{bmatrix}$$

and

$$\det(zI - A - BF) = \det\begin{bmatrix} z - 0.25 - f_1 & -0.75 - f_2 \\ -0.75 - f_1 & z - 0.25 - f_2 \end{bmatrix} = z^2 + (-0.5 - f_1 - f_2)z \ - 0.5f_1 - 0.5f_2 - 0.5.$$

Equating coefficients between $\det(zI - A - BF)$ and $\pi_{\text{des}}[z]$ yields the equation

$$\begin{bmatrix} -1 & -1 \\ -0.5 & -0.5 \end{bmatrix}\begin{bmatrix} f_1 \\ f_2 \end{bmatrix} = \begin{bmatrix} 0.9 \\ 0.45 \end{bmatrix}.$$

This equation doesn't have a unique solution. I'll pick $F = -\begin{bmatrix} 0.45 & 0.45 \end{bmatrix}$. ▲

9.1.3 Deadbeat control

Deadbeat control refers to a type of system response that is not possible for continuous-time systems. It is based on placing all of the closed-loop poles of the system at the origin in the z-plane. Start with a homogeneous linear discrete-time system in state space form

$$x^+ = Ax, \qquad x[0] = x_0.$$

Then

$$\begin{aligned} & x[k] \to 0 \text{ in finite time for any } x_0 \\ \iff & A^k \to 0 \text{ in finite time} \\ \iff & A^n = 0 \\ \iff & \text{all eigenvalues of } A \text{ are zero} \\ \iff & A \text{ is nilpotent.} \end{aligned}$$

Returning to state feedback, choosing $\pi_{\text{des}}[z] = z^n$ as the desired closed-loop characteristic polynomial leads to

$$x^+ = (A + BF)x$$

where $A + BF$ is nilpotent. Then $x[k] \to 0$ in finite time $k \leq n$. It is not possible to get this kind of behaviour from the continuous-time system $\dot{x} = (A + BF)x$ because the response $x(t) = \mathrm{e}^{t(A+BF)}x(0)$ will always be a decaying exponential and cannot go to zero in finite time.

9.1.4 Intersample ripple

Consider a continuous-time system

$$\dot{x} = Ax + Bu.$$

Discretize via c2d

$$x^+ = A_d x + B_d u_d.$$

Suppose that a state feedback $u = F_d x$ is designed so that $A_d + B_d F_d$ has all its eigenvalues inside the unit

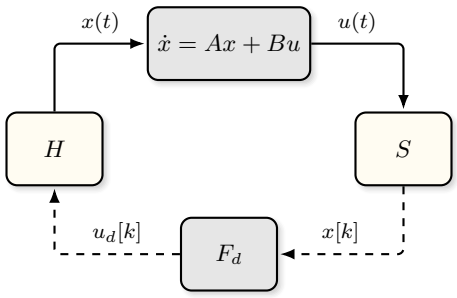


Figure 9.3: State feedback in a sampled-data configuration.

disk. The setup is shown in Figure 9.3. Then $x[k] = x(kT)$ converges to zero as k goes to infinity no matter the initial condition $x(0)$. Does $x(t)$ also converge to zero? Yes!

Proof. Consider a general sample period $[kT, (k+1)T)$. Let t be anytime in this sample period and let $\delta = t - kT$. From the basic theory $x(t)$ depends on the state at start of the sample period and the input over the time interval $[kT, t)$:

$$x(t) = \mathrm{e}^{A(t-kT)}x(kT) + \int_{kT}^{t} \mathrm{e}^{A(t-\tau)}Bu(\tau)\mathrm{d}\tau.$$

The control is constant over the sampling period so

$$x(t) = \mathrm{e}^{A(t-kT)}x(kT) + \int_{kT}^{t} \mathrm{e}^{A(t-\tau)}B\mathrm{d}\tau \; u(kT).$$

Perform a change of variables $\sigma := t - \tau$ in the intergral to get

$$x(t) = \mathrm{e}^{A(t-kT)}x(kT) - \int_{\delta}^{0} \mathrm{e}^{A\sigma}B\mathrm{d}\sigma\ u(kT)$$
$$= \mathrm{e}^{A(t-kT)}x(kT) + \int_{0}^{\delta} \mathrm{e}^{A\sigma}B\mathrm{d}\sigma\ u(kT).$$

Substitute the control input $u(kT) = F_d x(kT)$:

$$x(t) = \left(\mathrm{e}^{A(t-kT)} + \int_{0}^{\delta} \mathrm{e}^{A\sigma}B\mathrm{d}\sigma F_d\right) x(kT).$$

The term multiplying $x(kT)$ is just some matrix, say M:

$$x(t) = Mx(kT).$$

Replace t with $kT + \delta$:

$$x(kT + \delta) = Mx(kT).$$

Since $x(kT)$ converges to zero as $k \to \infty$, it follows that $x(kT + \delta) \to 0$ as $k \to \infty$. Since δ is arbitrary, we see that the continuous-time state converges to zero at all intersample times. ■

Here's a followup point: If $A_d + B_d F_d$ is nilpotent, then $x(kT)$ converges to zero in finite time. Does $x(t)$ converge to zero in finite time too? Yes, the proof is the same as above. Does this contradict the comment at the end of Section 9.1.3? No.

9.2 Output-feedback stabilization

In Section 9.1 we solved the stabilization problem under the assumption that the entire state x is available for feedback. Realistically, you usually can't measure all the states because (i) it may not be physically possible to take the measurements or (ii) the required sensors are too expensive. Normally, as suggested by our standard block diagrams in Figures 8.1, 8.2, only $y[k]$ is available for feedback. This is where observers come in – one uses an observer to estimate the value of the state vector using the information available from sensors that measure the system output.

Consider the setup in Figure 9.4. The plant has input u, output y and state-space model (A, B, C, D). The controller reads y and decides on the control action u and has state-space model (A_c, B_c, C_c, D_c).

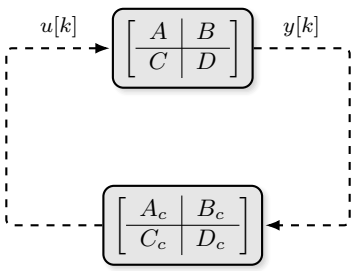


Figure 9.4: Output feedback stabilization.

In this section we design observer-based controllers to solve the following problem.

Output-Feedback Stabilization Problem. *Given (A, B, C, D), find a controller (A_c, B_c, C_c, D_c) so that the system in Figure 9.4 is internally stable.*

Recall from Section 6.7.1 that this system is internally stable if the closed-loop state vector converges to zero for any initial condition of the plant and the controller. To solve this problem, we introduce the notion of an observer.

9.2.1 Observers

An observer is a dynamic system that reconstructs the state of a system using only input-output measurements. In Figure 9.5 the observer block reads the input $u[k]$ to the plant, measures the output $y[k]$ from the plant, and uses this data to generate an estimate $\hat{x}[k]$ of the plant's internal state $x[k]$. We want to design an observer so that the **state estimation error** $e[k] \coloneqq x[k] - \hat{x}[k]$ goes to zero as $k \to \infty$ no matter how the plant and observer are initialized. We will then use the state estimate $\hat{x}$ to implement the controllers from Section 9.1.

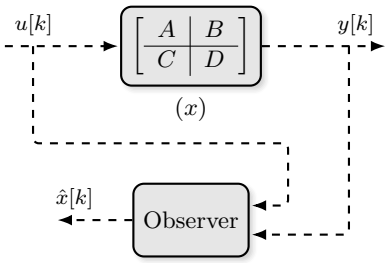


Figure 9.5: Observer block diagram.

The (known) model of the plant is

$$\begin{aligned} x^+ &= Ax + Bu \\ y &= Cx + Du. \end{aligned}$$

If we have a state estimate $\hat{x}[k]$ at time step k, then a good choice for the estimated output at time step k is

$$\hat{y}[k] = C\hat{x}[k] + Du[k].$$

A reasonable model of our observer is then

$$\hat{x}^+ = \underbrace{A\hat{x} + Bu}_{\text{plant emulator}} + \underbrace{L\,(\hat{y} - y)}_{\text{correction term}}. \tag{9.4}$$

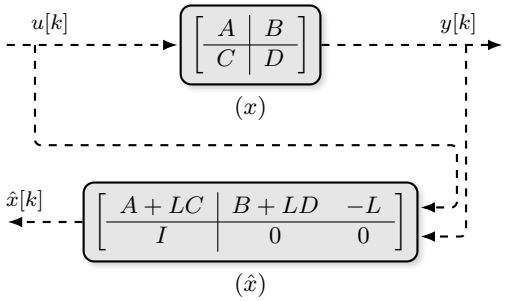
When does such an observer work? Consider the estimation error $e[k] = x[k] - \hat{x}[k]$. Then

$$\begin{aligned} e^+ &= x^+ - \hat{x}^+ \\ &= Ax + Bu - (A\hat{x} + Bu + L\,(\hat{y} - y)) \\ &= Ax - A\hat{x} - L\,(\hat{y} - y) \\ &= Ax - A\hat{x} - L\,(C\hat{x} + Du - Cx - Du) \\ &= (A + LC)(x - \hat{x}) \\ &= (A + LC)e. \end{aligned}$$

Therefore, the observation error $e[k]$ goes to zero for *any* initial condition $e[0]$ if, and only if, $A + LC$ is stable! Notice that the eigenvalues of $A + LC$ equal the eigenvalues of $(A + LC)^\top = A^\top + C^\top L^\top$. Therefore, we can arbitrarily place the eigenvalues of $A + LC$ if, and only if, $(A^\top, C^\top)$ is controllable. In summary, the proposed observer is given by

$$\hat{x}^+ = (A + LC)\hat{x} + (B + LD)u - Ly \tag{9.5}$$

where L is designed so that $A + LC$ has all its eigenvalues inside the open unit disk.



9.2.2 Observability

In the previous section we showed, using the observer (9.5), that the question of whether or not the estimation error goes to zero does not depend on the control input. Hence, the relevant system is

$$\begin{aligned} x^+ &= Ax \\ y &= Cx. \end{aligned} \tag{9.6}$$

Definition 9.2.1. System (9.6) is **observable** (or the pair (C, A) is **observable**) if for every initial state $x[0]$ there exists a time $k_f \geq 0$, such that $x[0]$ can be always be computed from the data $\{y[0], y[1], \ldots, y[k_f]\}$.

Observability says that the plant's initial condition $x[0]$ can be computed using output measurements. Let's characterize when this is possible. First

$$y[0] = Cx[0].$$

At the next time step

$$y[1] = Cx[1] = CAx[0].$$

Continuing this way, define the **observability matrix**

$$W_o := \begin{bmatrix} C \\ CA \\ CA^2 \\ \vdots \\ CA^{n-1} \end{bmatrix}. \tag{9.7}$$

Thus W_o has np rows and n columns where n is the dimension of x and p is the dimension of y. The equation

$$W_o x[0] = \begin{bmatrix} y[0] \\ y[1] \\ \vdots \\ y[n-1] \end{bmatrix}$$

has a unique solution $x[0]$ if, and only if the columns of W_o are linearly independent. Thus (C, A) is observable if, and only if the rank of W_o equals n. In the single-output case the matrix W_o is square and the system is observable if, and only if W_o is non-singular.

Equivalently, the pair (C, A) is observable if, and only if,

$$\operatorname{rank} \begin{bmatrix} A - \lambda I \\ C \end{bmatrix} = n, \qquad \text{for each eigenvalue } \lambda \text{ of } A.$$

By analogy with controllable eigenvalues and in view of the condition above, we say that an eigenvalue λ of A is *observable* if

$$\operatorname{rank}\begin{bmatrix} A-\lambda I \\ C \end{bmatrix} = n.$$

Summary: observability

$x[0]$ is always computable using output data $\Leftrightarrow$ (C, A) is observable
$\Leftrightarrow$ $\operatorname{rank} W_o = n$
$\Leftrightarrow$ $\operatorname{rank}\begin{bmatrix} A-\lambda I \\ C \end{bmatrix} = n$ for every eigenvalue λ of A
$\Leftrightarrow$ the eigenvalues of $A + LC$ can be freely assigned.

A related notion is that of detectability.

Definition 9.2.2. System (9.1) is **detectable** (or the pair of matrices (C, A) is **detectable**) if there exists a matrix L so that the eigenvalues of $A + LC$ are all inside the open unit disk.

So there exists an observer of the form (9.5) if, and only if (C, A) is detectable. The following three conditions are equivalent (i) (C, A) is detectable (ii) $(A^\top, C^\top)$ is stabilizable (iii) every eigenvalue of A with $|\lambda| \geq 1$ is observable.

Remark 9.2.3. The observability matrix (9.7) can be computed via the MATLAB command `obsv`. ♦

Example 9.2.1. (Double integrator) In Example 7.2.4 we discretized the double integrator plant to get

$$x^+ = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix} x + T \begin{bmatrix} \frac{T}{2} \\ 1 \end{bmatrix} u$$
$$y = \begin{bmatrix} 1 & 0 \end{bmatrix} x.$$

The observability matrix for this system is

$$W_o = \begin{bmatrix} C \\ CA \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 1 & T \end{bmatrix}.$$

Since $\det(W_o) = T$, the discretized double integrator is observable for all sampling periods $T > 0$. Let's design an observer for this system. We can do this using the same procedure we used for pole placement with the matrices $(A^\top, C^\top)$. I'll place the poles of $A + LC$ at the origin in order to get deadbeat behaviour and have our estimates converge as fast as possible. The procedure from Section 9.1.2 with

$$A^\top = \begin{bmatrix} 1 & 0 \\ T & 1 \end{bmatrix}, \qquad C^\top = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \qquad \pi_{\text{des}}[z] = z^2$$

yields

$$L^\top = -\begin{bmatrix} 2 & \frac{1}{T} \end{bmatrix}.$$

You can verify that the eigenvalues of

$$A + LC = \begin{bmatrix} -1 & T \\ -\frac{1}{T} & 1 \end{bmatrix}$$

are all at the origin. The observer is implemented in software as $\hat{x}^+ = (A + LC)\hat{x} + Bu - Ly$, i.e.,

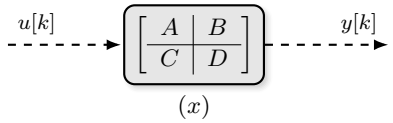
$$\hat{x}^+ = \begin{bmatrix} -1 & T \\ -\frac{1}{T} & 1 \end{bmatrix} \hat{x} + T \begin{bmatrix} \frac{T}{2} \\ 1 \end{bmatrix} u - \begin{bmatrix} -2 \\ -\frac{1}{T} \end{bmatrix} y.$$

Notice that only u and y are needed to implement this system both of which are available. ▲

9.2.3 Observer-based control

We are now in a position to think about a two stage controller: The first stage is an observer to generate an estimate of the plant's state; the second stage is to feed back this estimate as though it were the state. Let us see this when the goal is to stabilize an unstable plant. Take the plant to be

$$\begin{aligned} x[k+1] &= Ax[k] + Bu[k] \\ y[k] &= Cx[k] + Du[k]. \end{aligned}$$



Assumption 9.2.4. (A, B) is stabilizable and (C, A) is detectable. ◀

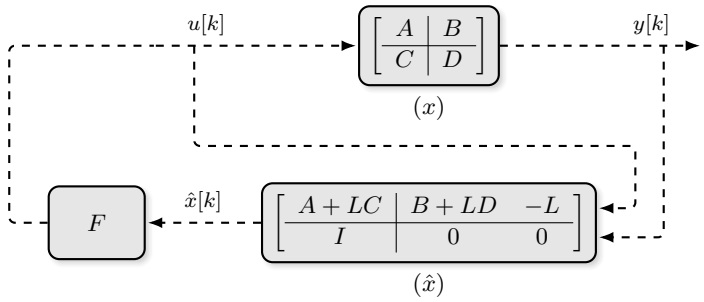
Choose F so that $A + BF$ is stable; choose L so that $A + LC$ is stable. Hook up the observer

$$\hat{x}[k+1] = (A + LC)\hat{z}[k] + (B + LD)u[k] - Ly[k]$$

to the plant as in the preceding section. Finally, close the loop via the control

$$u[k] = F\hat{x}[k].$$

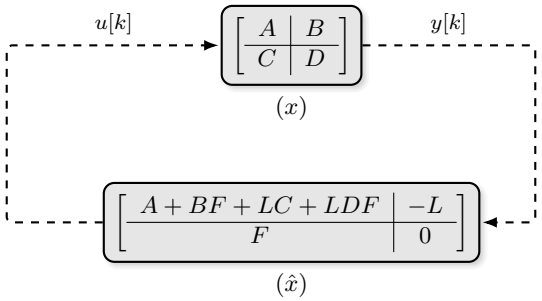
In this way we get the block diagram.



The block diagram can be simplified by substituting $u = F\hat{x}$ into the observer's state equation

$$\begin{aligned} \hat{x}[k+1] &= (A + BF + LC + LDF)\hat{x}[k] - Ly[k] \\ u[k] &= F\hat{x}[k] \end{aligned}$$

which yields the block diagram



Now we want to show the closed-loop system is internally stable, i.e., A_{cl} has all its eigenvalues inside the open unit disk. The observation error $e[k] = x[k] - \hat{x}[k]$ evolves according to

$$e^+ = (A + LC)e.$$

Also

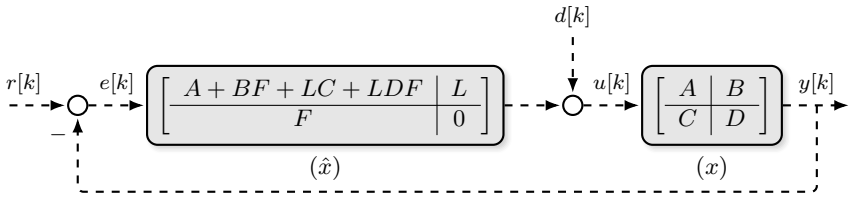
$$\begin{aligned} x^+ &= Ax + Bu \\ &= Ax + BF\hat{x} \\ &= Ax + BF(x - e) \\ &= (A + BF)x - BFe. \end{aligned}$$

Therefore, the closed-loop dynamics are governed by

$$\begin{bmatrix} x^+ \\ e^+ \end{bmatrix} = \begin{bmatrix} A + BF & -BF \\ 0 & A + LC \end{bmatrix} \begin{bmatrix} x \\ e \end{bmatrix}.$$

Since the closed-loop matrix is block triangular, its eigenvalues are those of $A + BF$ and $A + LC$. Therefore $x[k]$ and $e[k]$ converge to zero as $k \to \infty$ for any initial condition. Thus $x[k]$ and $\hat{x}[k]$ converge to zero as $k \to \infty$ for any initial condition. Thus A_{cl} is stable and the closed-loop system is internally stable.

Let's put the observer based controller into the more conventional form of a unity feedback system with a reference input r, an input disturbance d and in which the controller only has access to the tracking error $e = r - y$



Note the sign change in L in the controller due to the presence of negative feedback; when $r = 0$, $d = 0$ we recover the original block diagram. In the unity feedback setup the controller only has access to the tracking error e, not the plant output y.

To get a TF from E to U for the observer based controller, define

$$A_c \coloneqq A + BF + LC + LDF, \qquad B_c \coloneqq L, \qquad C_c \coloneqq F$$

so that

$$D[z] = C_c(zI - A_c)^{-1}B_c.$$

Example 9.2.2. (Double integrator) Let's return to the double integrator example. Design an observer-based controller so that the system's step response satisfies (i) settling time is less than 4 seconds for step inputs and (ii) the percentage overshoot is less than $\mathrm{e}^{-\pi}$ with a sampling period of $T = 0.05$ seconds.

The plant model is

$$\begin{aligned} x^+ &= \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix} x + T \begin{bmatrix} \frac{T}{2} \\ 1 \end{bmatrix} u \\ y &= \begin{bmatrix} 1 & 0 \end{bmatrix} x. \end{aligned}$$

We showed in Example 9.1.4 that this system is controllable. We showed in Example 9.2.1 that it's observable.

We start by picking the desired pole locations based on the transient specifications. The good region is

$$\mathbb{C}_g = \{s \in \mathbb{C} : \operatorname{Re}(s) < -1\} \cap \{s \in \mathbb{C} : |\arg(s)| \geq 3\pi/4\}.$$

Let's pick the desired s-plane pole locations as $\{-5 + j, -5 - j\}$ which are in $\mathbb{C}_g$. We map these to desired z-plane locations via

$$\{\xi_1, \xi_2\} = \left\{\mathrm{e}^{(-5+j)T}, \mathrm{e}^{(-5-j)T}\right\} = \left\{\mathrm{e}^{-0.25}\mathrm{e}^{j0.05}, \mathrm{e}^{-0.25}\mathrm{e}^{-j0.05}\right\}.$$

Design a state feedback matrix F so that the eigenvalues of $A+BF$ equal ξ_1, ξ_2. Using either the formula we derived in Example 9.1.4 or the command `place` in MATLAB we get

$$F = -\begin{bmatrix} 20.3503 & 8.3781 \end{bmatrix}.$$

Next we design an observer. We'll use the observer from Example 9.2.1 because it already has deadbeat behaviour to ensure fast convergence of our state estimates. The observer gain matrix is

$$L = -\begin{bmatrix} 2 \\ \frac{1}{T} \end{bmatrix} = -\begin{bmatrix} 2 \\ 20 \end{bmatrix}.$$

Our controller is implemented in software as

$$\begin{aligned} \hat{x}^+ &= A_c\hat{x} + B_c e \\ u &= C_c\hat{x} \end{aligned}$$

where

$$A_c = A + BF + LC = \begin{bmatrix} -1.0254 & 0.0395 \\ -21.0175 & 0.5811 \end{bmatrix}, \qquad B_c = L, \qquad C_c = F.$$

The transfer function of our controller is

$$D[z] = C_c(zI - A_c)^{-1}B_c = 208.3\frac{z - 0.9023}{z^2 + 0.4443z + 0.2349}. \tag{9.8}$$

From the controller TF (9.8) and the plant TF (see Example 7.2.4) you can check that the ch.p of the closed-loop system is

$$\pi[z] = D_pD_c + N_pN_c = z^2\left(z^2 - 1.5557z + 0.6065\right).$$

The roots of this polynomial are $\left\{e^{-0.25}e^{j0.05}, e^{-0.25}e^{-j0.05}, 0, 0\right\}$ as expected. Simulation results are shown in Figure 9.6. We did not meet the specifications. We should iterate by changing the pole locations. ▲

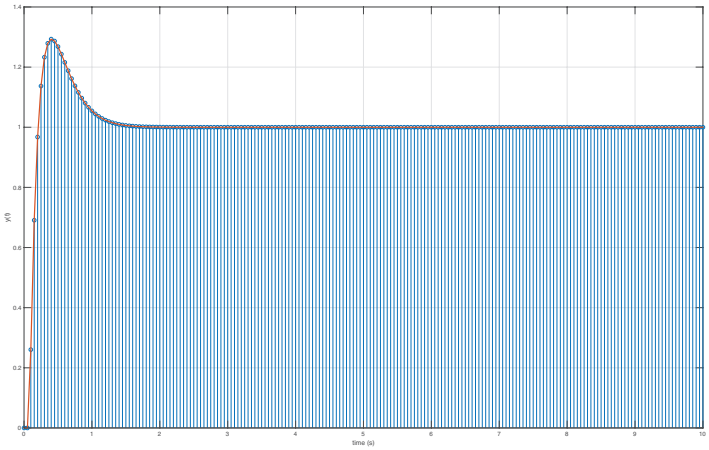


Figure 9.6: Simulation result for Example 9.2.2.

Remark 9.2.5. The following MATLAB script was used to do the calculations for this example.

```
A = [0 1;0 0]; B =[0;1]; C = [1 0]; D=0;
P = ss(A,B,C,D); %ct plant
T = 0.05; % sampling period
Pd = c2d(P, T); %step-invariant transformation
Ad = Pd.a; Bd=Pd.b; Cd=Pd.c; Dd=Pd.d; % dt state matrices
```

```
Xi = exp([-5+1i -5-1i]*T); %desired pole locations
F = -place(Ad, Bd, Xi); % feedback matrix
L = (-acker(Ad', Cd', [0 0]))'; % observer gain
Ac = Ad + Bd*F + L*Cd; Bc = L; Cc = F;
[n, d] = ss2tf(Ac, Bc, Cc, 0);
D = tf(n, d, T); % controller TF
```

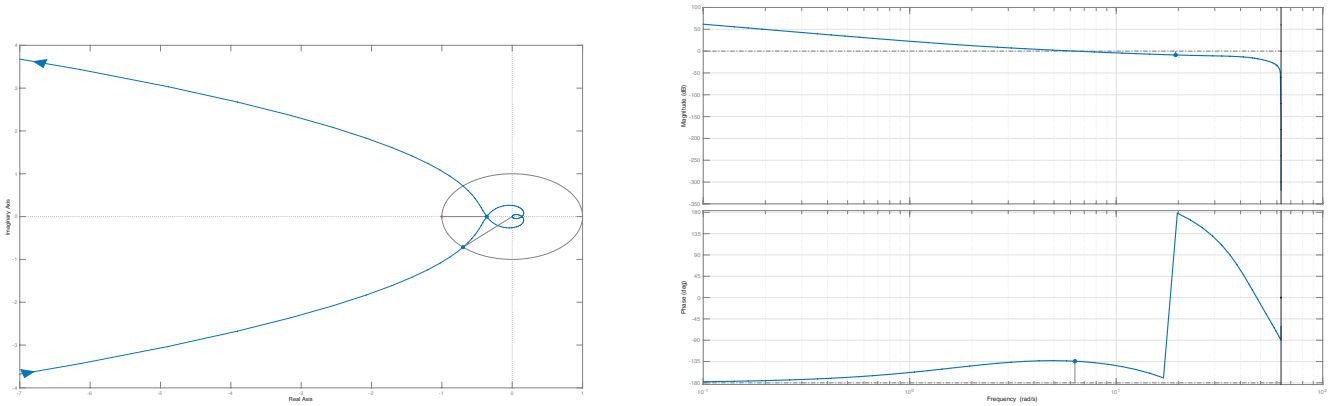
♦

The next example emphasizes the fact that, when doing state-space design, we should use all the tools at our disposal and evaluate the frequency response of the closed-loop system.

Example 9.2.3. In Example 9.2.2 the plant and controller were

$$P_d[z] = \frac{(0.05)^2}{2}\frac{z+1}{(z-1)^2}, \qquad D[z] = 208.3\frac{z-0.9023}{z^2+0.4443z+0.2349}.$$

Figure 9.7 shows the Nyquist and Bode plots of $D[z]P_d[z]$. From either of the plots in Figure 9.7 we deduce



(a) Nyquist plot. *(b) Bode plot.*

Figure 9.7: Nyquist and Bode plots of $D[z]P_d[z]$ for the plant and controller from Example 9.2.2.

that the phase margin is around 45° degrees and the gain margin is around 9db. We conclude that while this controller solves the output stabilization problem, the stability margins could be larger. ▲

9.3 Tracking and regulation

This section develops the state-space theory of tracking and regulation–the plant output should track a reference signal, such as a step, ramp, or sinusoid, and/or a plant disturbance should be rejected.

9.3.1 Introduction

Consider a cart like the one in Figure 9.8. The disturbance d is a external force while the force u is the control input. We want the cart to track a reference r in spite of the disturbance. Doing simple "$f = ma$" modelling we get the plant model

$$M\ddot{y} = u - Ky - d.$$

To keep things simple let's assume that $M = 1$, $K = 1$ so that

$$\ddot{y} = u - y - d.$$

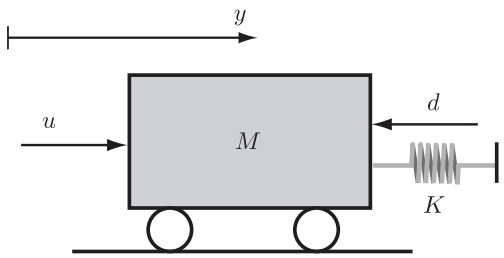


Figure 9.8: A simple cart subject to disturbing force.

Suppose we know that r is a constant (or a step) but we don't know its value; and we know that d is a sinusoid of frequency 10 rad/s but of unknown amplitude and phase. Then we know equations that can generate these signals, namely,

$$\dot{r} = 0, \qquad \ddot{d} + 100d = 0.$$

We call these two equations together the exomodel, "exo" meaning "from outside the plant." The only unknowns are the initial conditions of these equations. Since we're not saying anything about the magnitudes of these signals, the most we could try to achieve is asymptotic regulation: $r(t) - y(t)$ tends to zero. We want to design a controller to achieve this. We restrict the controller to have input $r - y$ and output u.

We're going to develop a state-space theory for this problem, so it's convenient to make a state-space model of the plant and exomodel. The setup is a plant with state x_1 and an exomodel with state x_2 like this:

$$\begin{aligned} \dot{x}_1 &= A_1 x_1 + A_3 x_2 + B_1 u \\ \dot{x}_2 &= A_2 x_2 \\ e &= D_1 x_1 + D_2 x_2. \end{aligned}$$

The output e is the signal that we want to go to zero, typically a tracking error. The exogenous signal x_2 also enters the plant via $A_3 x_2$, a disturbance. It is natural to assume that all the eigenvalues of A_2 are unstable (but no assumptions are made yet about A_1). For conciseness, the two states can be combined:

$$\dot{x} = Ax + Bu, \quad e = Dx, \quad x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, \quad A = \begin{bmatrix} A_1 & A_3 \\ 0 & A_2 \end{bmatrix}, \quad B = \begin{bmatrix} B_1 \\ 0 \end{bmatrix}, \quad D = \begin{bmatrix} D_1 & D_2 \end{bmatrix}.$$

Let's setup the cart example like this. Take as state variables:

$$x_1 \coloneqq (y, \dot{y}), \qquad x_2 = (r, d, \dot{d}), \qquad x = (x_1, x_2)$$

and the output $e = r - y$. We have

$$A = \left[\begin{array}{cc:ccc} 0 & 1 & 0 & 0 & 0 \\ -1 & 0 & 0 & -1 & 0 \\ \hdashline 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & -100 & 0 \end{array}\right], \qquad B = \left[\begin{array}{c} 0 \\ 1 \\ \hdashline 0 \\ 0 \\ 0 \end{array}\right], \qquad D = \left[\begin{array}{cc:ccc} -1 & 0 & 1 & 0 & 0 \end{array}\right].$$

The partition lines indicate the blocks

$$A = \begin{bmatrix} A_1 & A_3 \\ 0 & A_2 \end{bmatrix}, \quad B = \begin{bmatrix} B_1 \\ 0 \end{bmatrix}, \quad D = \begin{bmatrix} D_1 & D_2 \end{bmatrix}.$$

The eigenvalues of A_2 are $0, \pm 10j$, the frequencies of r and d. So A_2 is completely unstable–no stable eigenvalues.

Following the methodology of this chapter, select a sampling period T, say $T = 0.1$ and apply `c2d` to this model to get

$$x^+ = A_d x + B_d u, \qquad e = Dx.$$

The matrix A_d retains its block triangular structure

$$A_d = \left[\begin{array}{cc:ccc} 0.995 & 0.0998 & 0 & -0.0046 & -0.0002 \\ -0.0998 & 0.995 & 0 & -0.084 & -0.0046 \\ \hdashline 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0.5403 & 0.0841 \\ 0 & 0 & 0 & -8.4147 & 0.5403 \end{array}\right], \qquad B_d = \left[\begin{array}{c} 0.0050 \\ 0.0998 \\ \hdashline 0 \\ 0 \\ 0 \end{array}\right].$$

The exomodel has eigenvalues 1, $0.540 \pm 0.841j$. These are the continuous-time eigenvalues 0, $\pm 10j$ mapped via $\lambda \mapsto \mathrm{e}^{\lambda T}$. The discrete-time reference signal is the sampled $r(t)$, and the discrete-time disturbance is the sampled $d(t)$. The block diagram is shown Figure 9.9.

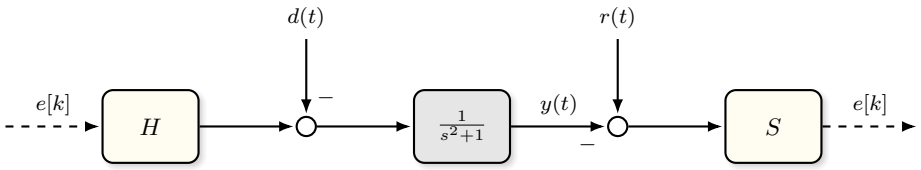


Figure 9.9: Sampled-data framework for tracking and regulation for the cart example.

Regulator Problem. *Design a controller with input e and output u, such that the feedback loop is stable, meaning that the plant state $x_1[k]$ and the controller state go to zero when $x_2[0] = 0$, and the output is regulated, meaning that $e[k]$ goes to zero for all initial conditions.*

The block diagram for the problem is shown in Figure 9.10.

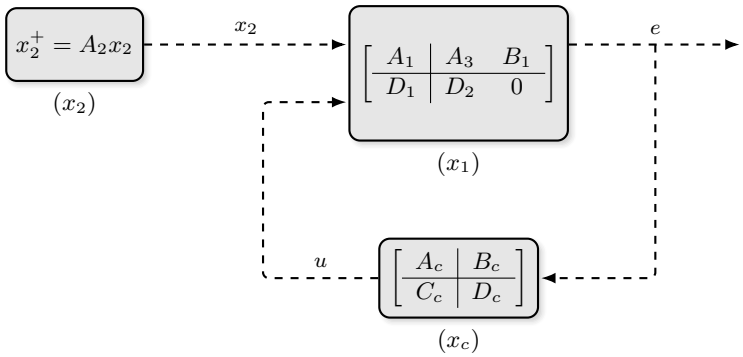


Figure 9.10: Closed-loop system for tracking and regulation.

9.3.2 Techical preliminaries

In this section we develop the tools to solve the regulator problem. The notation is local to this section; for example, there will be an A_1 but it won't be the same as in other sections; however, A_2 will be the same.

Consider the system

$$x^+ = Ax, \qquad e = Dx, \qquad A = \begin{bmatrix} A_1 & A_3 \\ 0 & A_2 \end{bmatrix}, \qquad D = \begin{bmatrix} D_1 & D_2 \end{bmatrix}$$

where A_1 is stable and A_2 has all its eigenvalues outside the open unit disk, i.e., $|z| \geq 1$. We're interested in when $e[k]$ goes to zero for every $x[0]$. If $A_3 = 0$, that is A is block diagonal, the question is easy: $e[k]$ goes to zero for every $x[0]$ if, and only if $D_2 = 0$. This follows from the time-domain solution

$$e[k] = D_1 A_1^k x_1[0] + D_2 A_2^k x_2[0].$$

To answer this question in general, it would be beneficial to transform A so that it becomes block diagonal.

Lemma 9.3.1. *If A_1 is stable and A_2 has all its eigenvalues with $|z| \geq 1$, then for every A_3 there exists a unique matrix X such that*

$$A_1X - XA_2 + A_3 = 0. \tag{9.9}$$

We don't prove this lemma but simply mention that this is an instance of a *Sylvester matrix equation*. Sylvester showed that the solution X exists and is unique if, and only if A_1 and A_2 do not have any common eigenvalues. The hypothesis in Lemma 9.3.1 ensures that A_1 and A_2 do not have common eigenvalues.

With the matrix X from Lemma 9.3.1 in hand, we can block diagonalize A using a similarity transformation:

$$T = \begin{bmatrix} I & X \\ 0 & I \end{bmatrix}$$

You can check that

$$T^{-1} = \begin{bmatrix} I & -X \\ 0 & I \end{bmatrix}, \qquad T^{-1}AT = \begin{bmatrix} A_1 & 0 \\ 0 & A_2 \end{bmatrix}.$$

Under the same transformation D becomes

$$DT = \begin{bmatrix} D_1 & D_1X + D_2 \end{bmatrix}$$

Thus, $e[k]$ goes to zero for any $x[0]$ if, and only if, $D_1X + D_2 = 0$. Let's summarize this section as a lemma.

Lemma 9.3.2. *Suppose*

$$x^+ = Ax, \qquad e = Dx, \qquad A = \begin{bmatrix} A_1 & A_3 \\ 0 & A_2 \end{bmatrix}, \qquad D = \begin{bmatrix} D_1 & D_2 \end{bmatrix}$$

where A_1 is stable and A_2 has no eigenvalues in the open unit disk. Then $e[k]$ goes to zero for every $x[0]$ if, and only if, $D_1X + D_2 = 0$ where X is the unique solution of

$$A_1X - XA_2 + A_3 = 0.$$

An important special case is when the reference signal and the disturbances are all constants. Then $A_2 = I$ and the unique solution to (9.9) is $X = -(A_1 - I)^{-1}A_3$. We'll state this special case as a corollary.

Corollary 9.3.3. *Suppose*

$$x^+ = Ax, \qquad e = Dx, \qquad A = \begin{bmatrix} A_1 & A_3 \\ 0 & I \end{bmatrix}, \qquad D = \begin{bmatrix} D_1 & D_2 \end{bmatrix}$$

where A_1 is stable. Then $e[k]$ goes to zero for every $x[0]$ if, and only if, $-D_1(A_1 - I)^{-1}A_3 + D_2 = 0$.

Exercise 9.1. In order to develop some intuition about the previous result, consider the system below where x_1, x_2 and e are scalars

$$\begin{aligned} x_1^+ &= a_1x_1 + a_3x_2 \\ x_2^+ &= x_2 \\ e &= d_1x_1 + d_2x_2. \end{aligned}$$

Find the TF $G[z]$ from x_2 to e. Find the steady-state gain $G[1]$ of this TF. Corollary 9.3.3 requires this steady-state gain to equal zero.

9.3.3 Solution to the regulator problem

To review, the setup is a plant with state x_1 and an exosystem with state x_2 like this

$$\begin{aligned} x_1^+ &= A_1x_1 + A_3x_2 + B_1u \\ x_2^+ &= A_2x_2 \\ e &= D_1x_1 + D_2x_2. \end{aligned}$$

The two states can be combined

$$x^+ = Ax + Bu, \qquad e = Dx, \qquad A = \begin{bmatrix} A_1 & A_3 \\ 0 & A_2 \end{bmatrix}, \qquad B = \begin{bmatrix} B_1 \\ 0 \end{bmatrix}, \qquad D = \begin{bmatrix} D_1 & D_2 \end{bmatrix}.$$

We assume that all the eigenvalues of A_2 are unstable (but no assumption is made yet about A_1). We will also assume that (D, A) is detectable because we're going to use an observer. The solution is going to mirror our solution to the output-feedback stabilization problem. We'll first look for a state feedback controller, $u = Fx$. Then we implement it via $u = F\hat{x}$ where $\hat{x}$ is from an observer with input e.

So let $u = Fx$. Then the controlled system is

$$x^+ = (A + BF)x, \qquad e = Dx,$$

where

$$A + BF = \begin{bmatrix} A_1 & A_3 \\ 0 & A_2 \end{bmatrix} + \begin{bmatrix} B_1 \\ 0 \end{bmatrix} \begin{bmatrix} F_1 & F_2 \end{bmatrix} = \begin{bmatrix} A_1 + B_1F_1 & A_3 + B_1F_2 \\ 0 & A_2 \end{bmatrix}.$$

Clearly, feedback stability is equivalent to $A_1 + B_1F_1$ being stable. Then asymptotic regulation is equivalent to the condition that $e[k]$ go to zero for every $x[0]$.

Theorem 9.3.4. *Assume A_2 has only unstable eigenvalues. Then the regulator problem is solvable by some $u = Fx$ if, and only if (A_1, B_1) is stabilizable and there exist matrices X, U such that*

$$A_1X - XA_2 + A_3 + B_1U = 0 \qquad \textit{(Regulator equations)} \tag{9.10a}$$

$$D_1X + D_2 = 0. \tag{9.10b}$$

Proof. Necessity. Assume that $u = Fx$ solves the regulator problem. Then $A_1 + B_1F_1$ must be stable and therefore (A_1, B_1) is stabilizable. By Lemma 9.3.2, asymptotic regulation means that there exists a matrix X such that

$$(A_1 + B_1F_1)X - XA_2 + A_3 + B_1F_2 = 0, \qquad D_1X + D_2 = 0. \tag{9.11}$$

We can re-write this expression in the form (9.10) by defining $U := F_1X + F_2$.

Sufficiency. Choose F_1 so that $A_1 + B_1F_1$ is stable. Solve the regulator equations (9.10) for X and U and set $F_2 = U - F_1X$. Then (9.11) holds, so asymptotic regulation follows from Lemma 9.3.2. ∎

Remark 9.3.5. The controller constructed in the proof of sufficiency in Theorem 9.3.4 is

$$u = \begin{bmatrix} F_1 & U - F_1X \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = F_1(x_1 - Xx_2) + Ux_2$$

which can be given an intuitive interpretation. The signal $Xx_2[k]$ and the control signal $Ux_2[k]$ can be viewed, respectively, as a *desired state trajectory* for the plant and an *open-loop control signal* that together yield zero regulation error $e = 0$. To see this, note that if the plant state is initialized as $x_1[0] = Xx_2[0]$ and the control signal $u[k] = Ux_2[k]$ is applied, then

$$\begin{aligned} x_1[1] &= A_1x_1[0] + A_3x_2[0] + B_1Ux_2[0] \\ &= (A_1X + A_3 + B_1U)\, x_2[0] \\ &= XA_2x_2[0] \\ &= Xx_2[1]. \end{aligned}$$

So $x_1[1] = Xx_2[1]$. Continuing in this way we see that $x_1[k] = Xx_2[k]$ for all $k \geq 0$. Furthermore

$$e[k] = D_1x_1[k] + D_2x_2[k] = (D_1X + D_2)x_2[k] = 0.$$

Therefore we can view the controller as consisting of a stabilizing term $F_1(x_1 - Xx_2)$ which drives the plant state x_1 to the reference state Xx_2 together with a feedforward term Ux_2 which drives the plant state along the desired trajectory. ⧫

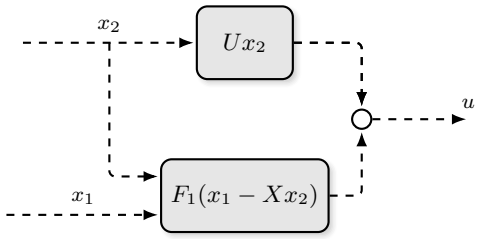


Figure 9.11: Controller structure in the full state feedback case.

Now we turn to designing a controller with input e, not x. Assuming that (D, A) is detectable, we can select L so that $A + LD$ is stable. The observer is

$$\hat{x}^+ = A\hat{x} + Bu + L(D\hat{x} - e).$$

Setting $u = F\hat{x}$ we get the observer based controller

$$\begin{aligned} \hat{x}^+ &= (A + BF + LD)\hat{x} - Le \\ u &= F\hat{x}. \end{aligned}$$

Theorem 9.3.6. *Assume (D, A) is detectable and A_2 has only unstable eigenvalues. Then the regulator problem is solved by the observer based controller if $u = Fx$ is a state-feedback solution.*

Instead of a proof, we'll do the cart example from start to finish.

Example 9.3.1. (Cart/spring with disturbance) We start with the discretized model (dropping the subscript "d") $x^+ = Ax + Bu$, $e = Dx$ where

$$A = \left[\begin{array}{cc:ccc} 0.995 & 0.0998 & 0 & -0.0046 & -0.0002 \\ -0.0998 & 0.995 & 0 & -0.084 & -0.0046 \\ \hdashline 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0.5403 & 0.0841 \\ 0 & 0 & 0 & -8.4147 & 0.5403 \end{array}\right], \qquad B = \left[\begin{array}{c} 0.005 \\ 0.0998 \\ \hdashline 0 \\ 0 \\ 0 \end{array}\right], \qquad D = \left[\begin{array}{cc:ccc} -1 & 0 & 1 & 0 & 0 \end{array}\right].$$

We check that A_2 has no stable eigenvalues, that (D, A) is detectable (in fact observable) and that (A_1, B_1) is stabilizable (in fact controllable).

Next we select F_1 to stabilize $A_1 + B_1F_1$. I'll place the poles at $\mathrm{e}^{(-5\pm j)T}$ to yield

$$F_1 = -\begin{bmatrix} 1.4722 & 0.783 \end{bmatrix}.$$

Next we have to check the solvability of

$$A_1X - XA_2 + A_3 + B_1U = 0, \qquad D_1X + D_2 = 0$$

for X and U. The easiest way to do this is to try and solve them. So write

$$X = \begin{bmatrix} x_{11} & x_{12} & x_{13} \\ x_{21} & x_{22} & x_{23} \end{bmatrix}, \qquad U = \begin{bmatrix} u_{11} & u_{12} & u_{13} \end{bmatrix}$$

and plug them into the above equation. You'll get 9 equations and 9 unknowns. Solving yields

$$X = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & -0.0009 \end{bmatrix}, \qquad U = \begin{bmatrix} 1 & 0.9194 & 0.502 \end{bmatrix}.$$

The state feedback controller is therefore

$$F = \begin{bmatrix} F_1 & F_2 \end{bmatrix} = \begin{bmatrix} F_1 & U - F_1 X \end{bmatrix} = \begin{bmatrix} -1.4722 & -0.783 & 16.1013 & 0.9194 & 0.0437 \end{bmatrix}.$$

Finally, assigning the eigenvalues of $A + LD$ at $\left\{0, \mathrm{e}^{-20T}, \mathrm{e}^{-30T}, \mathrm{e}^{-40T}, \mathrm{e}^{-50T}\right\}$ yields

$$L = \begin{bmatrix} -83.3491 & 38.5829 & -87.2095 & -103.5444 & 627.6507 \end{bmatrix}^\top.$$

The resulting controller state equations are

$$\begin{aligned} \hat{x}^+ &= (A + BF + LD)\hat{x} - Le \\ u &= F\hat{x} \end{aligned}$$

which yields the TF from e to u of

$$D[z] = \frac{486.8z^4 - 1443z^3 + 1890z^2 - 1277z + 356.6}{z^5 + 0.5728z^4 - 1.659z^3 + 0.8143z^2 + 1.053z - 1.781}.$$

The controller has poles at 1, $0.540 \pm 0.841j$, as it must to track the step and reject the disturbance. The structure of our controller has the familiar block diagram in Figure 9.12. ▲

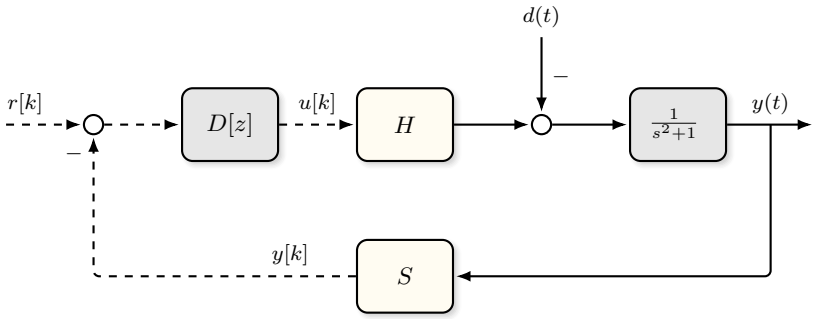


Figure 9.12: Closed-loop block diagram for Example 9.3.1.

Design procedure

Let's summarize our results with a design procedure. With (A_1, B_1) stabilizable and (D, A) detectable.

1. Choose F_1 so that $A_1 + B_1 F_1$ is stable.
2. Solve
$$A_1 X - X A_2 + A_3 + B_1 U = 0, \qquad D_1 X + D_2 = 0$$
for X and U and set
$$F = \begin{bmatrix} F_1 & U - F_1 X \end{bmatrix}.$$
3. Choose L so that $A + LD$ is stable. The state observer is
$$\hat{x}^+ = A\hat{x} + Bu + L(D\hat{x} - e).$$

4. The observer based controller is
$$\hat{x}^+ = (A + BF + LD)\hat{x} - Le$$
$$u = F\hat{x}.$$

Example 9.3.2. (Cart/spring with no disturbance) Consider the cart/spring with no disturbance and only a constant reference. Discretizing the model with $T = 0.1$ gives

$$A = \left[\begin{array}{cc:c} 0.995 & 0.0998 & 0 \\ -0.0998 & 0.995 & 0 \\ \hdashline 0 & 0 & 1 \end{array}\right], \qquad B = \left[\begin{array}{c} 0.0050 \\ 0.0998 \\ \hdashline 0 \end{array}\right], \qquad D = \left[\begin{array}{cc:c} -1 & 0 & 0 \end{array}\right].$$

Select F_1 so that $A_1 + B_1F_1$ is stable. Arbitrarily selecting the eigenvalues to be zero yields

$$F_1 = \begin{bmatrix} -99.1 & -15 \end{bmatrix}.$$

Next we solve (9.12) which in this example simplifies to

$$\begin{bmatrix} A_1 - I & B_1 \\ D_1 & 0 \end{bmatrix} \begin{bmatrix} X \\ U \end{bmatrix} = - \begin{bmatrix} A_3 \\ D_2 \end{bmatrix}$$

which gives

$$X = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \qquad U = 1.$$

Therefore

$$F = \begin{bmatrix} F_1 & U - F_1X \end{bmatrix} = \begin{bmatrix} -99.1 & -15 & 100 \end{bmatrix}.$$

Finally, assigning the eigenvalues of $A + LD$ at zero we get

$$L = \begin{bmatrix} -97.1 & 24.8 & -100 \end{bmatrix}^\top.$$

The resulting controller is

$$\hat{x}^+ = (A + BF + LD)\hat{x} - Le$$
$$u = F\hat{x}.$$

which has TF

$$\frac{U[z]}{E[z]} = D[z] = \frac{768z^2 - 1091z + 423.4}{z^3 + 2z^2 - 0.875z - 2.12}.$$

The controller has a pole at $z = 1$ as it must. ▲

We now take a minute to discuss how to solve the regulator equations. The regulator equations are linear in the unknowns X and U. Therefore they can be converted into a conventional linear algebraic equation. One way to do this is to take the columns of a matrix and stack them up to form a vector. For example, denoting the columns of a matrix X by x_1, x_2, …, x_N, let

$$\operatorname{vec}(X) \coloneqq \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix}.$$

If we vectorize X in this way, then the matrix product A_1X becomes

$$(I \otimes A_1) \operatorname{vec}(X)$$

where I is the identity matrix with the same number of columns of X and $\otimes$ is the Kronecker product

$$I \otimes A_1 = \text{block diag}\,(A_1, A_1, \ldots, A_1).$$

The vectorizing of the product of the form XA_2 is a little trickier. It's $\left(A_2^\top \otimes I\right) \text{vec}\,(X)$ where I is the identity matrix with the same number of rows as X. In this way we can write the regulator equations (9.10) as

$$\begin{bmatrix} I_{n_2} \otimes A_1 - A_2^\top \otimes I_{n_1} & I_{n_2} \otimes B_1 \\ I_{n_2} \otimes D_1 & 0_{qn_2 \times mn_2} \end{bmatrix} \begin{bmatrix} \text{vec}\,(X) \\ \text{vec}\,(U) \end{bmatrix} = - \begin{bmatrix} \text{vec}\,(A_3) \\ \text{vec}\,(D_2) \end{bmatrix} \tag{9.12}$$

where $n_1 \coloneqq \dim(x_1)$, $n_2 \coloneqq \dim(x_2)$, $m = \dim(u)$ and $q = \dim(e)$. Thus solvability is equivalent to a rank test.

Remark 9.3.7. The MATLAB code below was used to design the controller from Example 9.3.2.

```
n = 2; %plant state dim
q = 1; % exo dim
m = 1; % number of inputs
A = [0  1 0; -1 0 0; 0 0 0]; B = [0;1;0]; C = [1 0 0];
D1 = [-1 0]; D2 = [1]; D = [D1 D2];
T = 0.1;
P = ss(A, B, C, 0); % ct plant
Pd = c2d(P, T); %step-invariant transformation
Ad = Pd.a; Bd=Pd.b; Cd=Pd.c; Dd=Pd.d; % dt state matrices
A1 = Ad(1:n, 1:n); A2 = Ad(n+1:n+q, n+1:n+q); B1 = Bd(1:n,:); A3 = Ad(1:n, n+1:n+q);
F1 = -acker(A1, B1, [0 0]);
L = -(acker(Ad', D', [0 0 0]))';
vecA3 = reshape(A3, [], 1); vecD2 = reshape(D2, [], 1);
M = [kron(eye(q), A1)-kron(A2', eye(n)) kron(eye(q), B1); kron(eye(q), D1) zeros(q, q*m)];
SOLN = M\[-vecA3;-vecD2]; % solve regulator equations
X = reshape(SOLN(1:n*q), n, q);
U = reshape(SOLN(n*q+1:n*q+q), m, q);
F = [F1 U-F1*X];
Ac = Ad+Bd*F + L*D; Bc = -L; Cc = F; % controller state model
[Nd, Dd] = ss2tf(Ac, Bc, Cc, 0);
D = tf(Nd, Dd, T); % controller TF
```

♦

9.3.4 More examples

Example 9.3.3. (Cart with neither spring nor disturbance) The continuous-time model $\ddot{y} = u$ is a double integrator which we've studied a lot. The reference $r(t)$ is a step of unknown magnitude. The discretized model is therefore

$$x^+ = \left[\begin{array}{cc|c} 1 & T & \frac{T}{2} \\ 0 & 1 & T \\ \hline 0 & 0 & 1 \end{array}\right] x + T \left[\begin{array}{c} \frac{T}{2} \\ 1 \\ \hline 0 \end{array}\right] u, \qquad e = \left[\begin{array}{cc|c} -1 & 0 & 1 \end{array}\right] x$$

As usual, the partition lines indicate the blocks

$$A = \begin{bmatrix} A_1 & A_3 \\ 0 & A_2 \end{bmatrix}, \quad B = \begin{bmatrix} B_1 \\ 0 \end{bmatrix}, \quad D = \begin{bmatrix} D_1 & D_2 \end{bmatrix}.$$

We confront a problem here. The observability matrix

$$\begin{bmatrix} D \\ DA \\ DA^2 \end{bmatrix} = \begin{bmatrix} -1 & 0 & 1 \\ -1 & -T & 1 \\ -1 & -2T & 1 \end{bmatrix}$$

is singular so the system isn't observable. Applying the PBH test to the unstable eigenvalue 1 to check detectability

$$\operatorname{rank}\begin{bmatrix} D \\ A-I \end{bmatrix} = \operatorname{rank}\begin{bmatrix} -1 & 0 & 1 \\ 0 & T & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} = 2.$$

So the pair (D, A) isn't detectable and we won't be able to construct an observer. The reason that (D, A) is unobservable is because our setup is redundant: Since the plant is a double integrator, step tracking will automatically follow from stability. So modelling the reference r is unnecessary. In fact, we should not have modelled r.

Let's begin anew

$$x^+ = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix} x + T \begin{bmatrix} \frac{T}{2} \\ 1 \end{bmatrix} u, \qquad e = \begin{bmatrix} -1 & 0 \end{bmatrix} x$$

Now the pair (D, A) is observable and the regulation problem has been reduced to an output feedback stabilization problem. Thus we can use, say, the controller from Example 9.2.2. ▲

The next example is one where asymptotic step tracking is not feasible. We just need the plant to have a zero at $z = 1$.

Example 9.3.4. (Regulator problem isn't solvable) Consider the continuous-time plant

$$\ddot{y} = \dot{u} - y.$$

The TF of this system is

$$P(s) = \frac{s}{s^2 + 1}.$$

Discretizing at $T = 0.05$ yields

$$P_d[z] = \mathsf{c2d}(P(s)) = 0.05\frac{z-1}{z^2 - 1.998z + 1}$$

which has a zero at $z = 1$. Bringing in a state-space model for P_d and an exomodel yields

$$x^+ = \left[\begin{array}{cc:c} 0 & 1 & 0 \\ -1 & 1.998 & 0 \\ \hdashline 0 & 0 & 1 \end{array}\right] x + \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} u, \qquad e = \left[\begin{array}{cc:c} 0.05 & -0.05 & 1 \end{array}\right] x.$$

> **Exercise 9.2.** By hand or using MATLAB, check that (A_1, B_1) is controllable and (D, A) is observable.

Select F_1 so that $A_1 + B_1F_1$ is stable. Arbitrarily selecting the eigenvalues to be zero yields

$$F_1 = \begin{bmatrix} 1 & -1.998 \end{bmatrix}.$$

Next we attempt to solve the regulator equations (9.10), or equivalently (9.12). The numbers in (9.12) are

$$\begin{bmatrix} -1 & 1 & 0 \\ -1 & 0.998 & 1 \\ 0.05 & -0.05 & 0 \end{bmatrix} \begin{bmatrix} X \\ U \end{bmatrix} = - \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}.$$

This isn't solvable. ▲

The situation may arise where the reference value r is actually known. Then the problem is really one of stabilization about a non-zero equilibrium point as the next example illustrates.

Example 9.3.5. (Cart/spring with no disturbance) Consider again $\ddot{y} = u - y$ where the desired output is $y = r = 1$. Of course, we could use regulator theory but since we know the specific reference signal r in this case, we don't need to. The discretized plant model, with sampling period $T = 0.1$, is

$$x^+ = Ax + Bu, \qquad y = Cx, \qquad A = \begin{bmatrix} 0.995 & 0.0998 \\ -0.0998 & 0.995 \end{bmatrix}, \qquad B = \begin{bmatrix} 0.0050 \\ 0.0998 \end{bmatrix}, \qquad C = \begin{bmatrix} 1 & 0 \end{bmatrix}.$$

We want to stabilize the system at $y = x_1 = 1$. First find an equilibrium configuration $(\bar{x}, \bar{u})$ for this value of y:

$$\begin{aligned} \bar{x} &= A\bar{x} + B\bar{u} \\ 1 &= C\bar{x}. \end{aligned}$$

Write this equation in matrix notation

$$\begin{bmatrix} A - I & B \\ C & 0 \end{bmatrix} \begin{bmatrix} \bar{x} \\ \bar{u} \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

and solve to get

$$(\bar{x}, \bar{u}) = \left(\begin{bmatrix} 1 \\ 0 \end{bmatrix}, 0 \right).$$

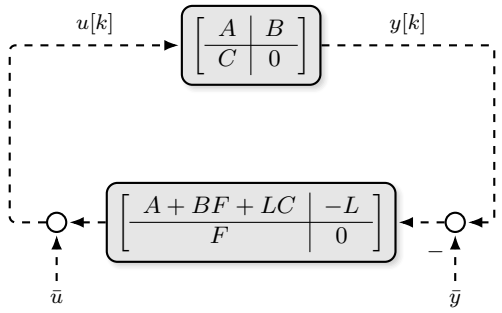
Define the deviations from the equilibrium $\delta x \coloneqq x - \bar{x}$, $\delta u \coloneqq u - \bar{u}$, $\delta y \coloneqq y - \bar{y}$ so that

$$\begin{aligned} \delta x^+ &= A\delta x + B\delta u \\ \delta y &= C\delta x. \end{aligned}$$

So the problem of driving y to 1 is just an output-feedback stabilization problem and so long as (A, B) is stabilizable and (C, A) is detectable, we can solve it using an observer based controller

$$\begin{aligned} \delta\hat{x}^+ &= (A + BF + LC)\delta\hat{x} - L\delta y \\ \delta u &= F\delta\hat{x} \end{aligned}$$

with $u = \delta u + \bar{u}$.



Exercise 9.3. Determine F and L to solve this problem.

▲

Bibliography

[Åström and Murray, 2019] Åström, K. and Murray, R. (2019). *Feedback systems: An introduction for scientists and engineers*. Princeton University Press, 2 edition. 3

[Åström and Wittenmark, 1997] Åström, K. and Wittenmark, B. (1997). *Computer controlled systems : Theory and design*. Prentice-Hall, New Jersey, 3rd edition. 7

[Barker, 1952] Barker, R. (1952). The pulse transfer function and its application to sampling servo systems. *Proceedings of the Institute of Electrical Engineering*, IV:302–317. 7

[Egerstedt, 2011] Egerstedt, M. (2011). Complex networks: Degrees of control. *Nature*, 473(7346):158–159. 3

[Franklin et al., 1998] Franklin, G., Powell, J., and Workman, M. (1998). *Digital Control of Dynamic Systems*. Ellis-Kagle Press, 3 edition. 69

[Garin and Schenato, 2010] Garin, F. and Schenato, L. (2010). A survey on distributed estimation and control applications using linear consensus algorithms. In *Networked Control Systems*, pages 75–107. Springer. 3

[Gelb and Velde, 1968] Gelb, A. and Velde, W. V. (1968). *Multiple-input describing functions and nonlinear system design*. McGraw-Hill. 110

[Greenberg, 1998] Greenberg, M. (1998). *Advanced Engineering Mathematics*. Prentice-Halll, Upper Saddle River, N.J., U.S.A. 108

[Hellerstein et al., 2004] Hellerstein, J., Diao, Y., Parekh, S., and Tilbury, D. M. (2004). *Feedback Control of Computing Systems*. Wiley-IEEE Press, New York, U.S.A. 3

[Hurewicz, 1947] Hurewicz, W. (1947). Filters and servo systems with pulsed data. In James, H., Nichols, N., and Philips, R., editors, *Theory of servomechanisms*. McGraw-Hill, New York. 7

[Jury, 1960] Jury, E. (1960). Recent advances in the field of sampled-data and digital control systems. In *Proceedings of IFAC Conference*, pages 240–246, Moscow. 7

[Jury, 1964] Jury, E. (1964). *Theory and application of the z-transform method*. Wiley, New York. 139

[Karlsson, 2020] Karlsson, N. (2020). Feedback control in programmatic advertising: The frontier of optimization in real-time bidding. *IEEE Control Systems Magazine*, 40(5):40–77. 3

[Keynes, 1936] Keynes, J. M. (1936). *The General Theory of Employment, Interest and Money*. MacMillan, London, U.K. 3

[Kuo, 1963] Kuo, B. (1963). *Analysis and synthesis of sampled-data control systems*. Prentice-Hall, New Jersey, U.S.A. 7

[Lalwani et al., 2006] Lalwani, C., Disney, S. M., and Towill, D. R. (2006). Controllable, observable and stable state space representations of a generalized order-up-to policy. *International Journal of Production Economics*, 101:172–184. 3

[Lathi, 1992] Lathi, B. (1992). *Linear Systems and Signals*. Berkeley-Cambridge, Carmichael, California, U.S.A. 121

[MacLane and Birkhoff, 1999] MacLane, S. and Birkhoff, G. (1999). *Algebra*. A.M.S. Chelsea, 3rd edition. 98

[Nielsen, 2017] Nielsen, C. (2017). Introduction to feedback control. SE380 Course Notes. 145

[Ragazzini and Franklin, 1958] Ragazzini, J. and Franklin, G. (1958). *Sampled-data control systems*. McGraw-Hill, New York, U.S.A. 7

[Ragazzini and Zadeh, 1952] Ragazzini, J. and Zadeh, L. (1952). The analysis of sampled-data systems. *Transactions AIEE*, 71(II):225–234. 7

[Ren and Beard, 2008] Ren, W. and Beard, R. (2008). *Distributed Consensus in Multi-vehicle Cooperative Control*. Springer. 3

[Shannon, 1949] Shannon, C. (1949). Communication in the presence of noise. *Proceedings of the IRE*, 37(1):10–21. 7

[Šiljak, 1968] Šiljak, D. (1968). *Nonlinear systems: The parameter analysis and design*. Wiley. 110

[Tsypkin, 1950] Tsypkin, Y. (1950). Theory of discontinuous control. *Avtomatika i Telemekhanika*, 3. 7

[Vidyasagar, 2002] Vidyasagar, M. (2002). *Nonlinear Systems Analysis*. SIAM, Philadelphia, P.A., U.S.A., 2nd edition. 106, 107

[Williams, 1978] Williams, T. (1978). Two decades of change: A review of the 20 year history of digital control. In *Proceedings of the 3rd Jerusalem Conference on Information Technology*, pages 739–744, Jerusalem, Israel. 7

Index