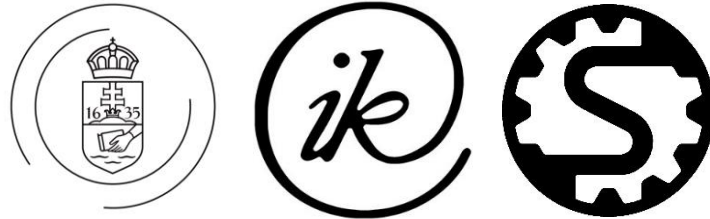


EÖTVÖS LORÁND UNIVERSITY
FACULTY OF INFORMATICS
SAVARIA INSTITUTE OF TECHNOLOGY



Manual on Control Techniques for Engineers

By Luis Rubio Rodríguez

Table of Contents

Preface.....	
1. Chapter 1: Introduction	1
1.1 Signals.....	1
1.2 Systems	2
1.3 Control Systems	5
2. Chapter 2: Mathematical modelling	9
2.1 Models of signals	9
2.1.1 Classification of signals according to signal processors.....	10
2.1.2 Types of signals in control.....	11
2.2 Fourier or frequency analysis of signals	14
2.3 Models of Systems.....	16
2.3.1 Classification of systems.....	16
2.3.2 Properties of systems	17
2.4 Description of systems by ordinary non-homogeneous differential equations with constant coefficients.....	18
2.4.1 First order systems	18
2.4.2 Second order systems.....	21
2.4.3 High order systems	23
2.5 State space representation	24
2.6 Control systems.....	25
2.6.1 Main properties	26
2.6.2 Properties according to the requirements of the designer of the control system	26
3 Chapter 3: Laplace and inverse Laplace transforms	29
3.1 Laplace and inverse Laplace transforms.....	29
3.1.1 Properties of Laplace transforms	30
3.1.2 Table of Laplace transforms	32
3.1.3 Laplace inverse transform	32
3.2 Transfer function.....	34
3.3 Block diagrams	36

4	Chapter 4: Transient response of first- and second-order systems	41
4.1	Transient response of first order transfer function.....	41
4.1.1	Step response of first order system	43
4.1.2	Impulse response of first order system	46
4.2	Transient response of second order transfer function	47
4.2.1	Step response of second order system	47
4.2.2	Impulse response of second order system.....	55
5	Chapter 5: Steady state errors analysis	58
6	Chapter 6: Stability of continuous-time systems	65
6.1	Stability	65
6.2	Routh-Hurwitz Criterion.....	66
6.3	Root Locus.....	69
6.4	Frequency domain analysis.....	76
6.5	Bode plot.....	78
6.6	Mention to Nyquist polar plot.....	81
7	Chapter 7: Z-transform and inverse Z-transform.....	87
7.1	Z-transform	87
7.2	The region of convergence of the Z-transform	89
7.3	Discrete-time signals.....	92
7.4	Z-transform table of basic sequences.....	95
7.5	Inverse Z-transform methods	95
7.6	The concept of a discrete-time system. From difference equations to transfer function. 98	
7.6.1	Pulse transfer function	99
7.6.2	Transfer function: poles and zeros	100
7.7	Stability of discrete-time domain systems	101
7.8	State variables	102
7.9	From state space to pulse transfer function.....	103
7.10	Sampled data control systems.....	104
7.11	Analysis of closed-loop discrete-time systems	107
7.11.1	Introduction.....	107
7.11.2	Zero-hold equivalences	108
8	Chapter 8: PID control.....	119

9	Chapter 9: Control structures	147
9.1	Feed-forward control	147
9.2	Two degrees of freedom control	148
9.3	Supervisory control	148
9.4	Hierarchical control	150
9.5	Soft sensors	152
9.6	Cascade structure	153
10	Chapter 10: Advanced control methods.....	156
10.1	Model predictive	157
10.2	Adaptive control	157
10.3	Neural network control	159
10.4	Sliding mode control.....	159
10.5	Robust control.....	162
10.6	Optimal control	162
10.7	Intelligent control.....	163
10.7.1	Learning control.....	163
10.7.2	Expert control.....	163
10.7.3	Fuzzy control	167
	Acknowledgments.....	

Preface

This manuscript is intended to introduce the topic of control techniques to engineers. The manual introduces signals and systems, which are the basic components of a process. It follows how to manipulate signals to obtain the required behavior of the system. This part is known as control systems.

Chapter 1 reviews the foundations of every part -signals, systems, and control systems- mathematically and intuitively to gain a basic understanding of these areas. It provides examples to be found in your daily life and in industry and academia that hold significance. *Chapter 2* introduces the representation of the systems to be controlled from a standard and universal perspective, as well as the mathematical modeling of signals and systems. Physical systems, independent of their nature, have in common that they may be modeled by very similar means, differential equations. *Chapter 3* deals with Laplace and inverse Laplace transforms as a human attempt to simplify the mathematics behind differential equations. The Laplace domain allows the conversion of challenging differential equations into easier and more familiar polynomial equations, providing a useful standpoint to deliver analysis and results of the physical systems under consideration. *Chapter 4* presents the transient response of first- and second-order systems against step and impulse inputs. Chapter 5 introduces steady-state errors. *Chapter 6* considers the stability of the systems from a control point of view. It shows how to manipulate the system with variations in some parameters of the control loop. *Chapter 7* informs the discrete-time domain, or Z-transform, where the use of electronics helps to implement the controllers. *Chapter 8* covers the most widely used control algorithm in industry, the PID, or proportional integral and derivative control. Furthermore, *Chapter 9* dives into control system structures, which are widely applied in industrial systems. *Chapter 10 ends the booklet* with an introduction to advanced control techniques that may be applied to uncertain or non-linear systems.

The book intends to introduce these concepts from a mathematical point of view to provide a good baseline for developing the main ideas behind control techniques. Moreover, a software-based helpline is accomplished to facilitate the calculus of mathematics. For this reason, some of the chapters are accompanied by programming code, which can be useful when dealing with control systems. Finally, examples and exercises are included to support the learning process.

Chapter 1: Introduction

This chapter introduces the foundations of every part, which consists of control system loops, namely, signals, systems, and control. Hereby, a conventional introduction is presented, which allows the reader to gain an understanding of them.

- ✓ **Introduction to Signals.**
- ✓ **Introduction to Systems.**
- ✓ **Introduction to Control Systems.**

The universal language, called mathematics in engineering, and its implementation through algorithms are used to state the basics of this field. Combining these mathematical tools with insight and understanding of the control system to be controlled, the reader may learn to make reasoning from a practical point of view for the use of mathematical resources to solve challenges in the control systems field.

This chapter reviews the foundations of every part -signals, systems, and control systems- mathematically and intuitively to a gain basic understanding of these areas. It provides examples you may find in your daily life and in industry and academia that hold significance.

1.1 Signals

Signals are presented in nature in different ways, such as light luminescence or intensity, humidity, temperature, wind speed, etc. (see figure 1.1). Our ability to read and monitor the magnitudes and orientations of these signals becomes a springboard to handle them for practical purposes.



Figure 1.1: Signals in nature.

A signal is a magnitude that varies in time and/or position. If the magnitude is fixed, it is known as a parameter or constant variable. Signals have many backgrounds associated to them, for instance, physical, chemical, biological, economical, or social, to cite some [1, 2].

A signal is intended to convey information and energy. A signal is a function of one or more independent variables that contain some information. Ex.: voice, photo, video, sound, etc. Noise is also a signal, but the information conveyed by noise is unwanted, hence it is considered undesirable [1, 2].

Signals are represented through a scale, table, discrete-time graph, function, or graph. Electronic devices that can provide signals are known as **signal generators**. These signals are later managed by **signal processors**. These devices commonly input to the control system. Signal generators provide signals, and signal processors modify the input signal according to the system requirements. Previously, the signal must have been measured by a sensor or transducer, which are responsible for handling (representing, saving, processing, etc.) them [1, 2].

1.2 Systems

A system is a collection of elements or components that are organized for a common purpose. Systems, primarily, may be classified as static and dynamic systems. This text deals with dynamic systems, but it is intended to recognise both [3,4,5].

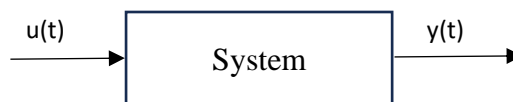


Figure 1.2: Representation of a system where an input signal into it generates an output signal.

Figure 1.2 represents a system where an input signal, $u(t)$, is transformed into an output signal, $y(t)$ by the inherent behavior of the system.

A **static system** is a system in which the output at any instant of time depends on the input sample at the same time. In other words, the system in which the output depends only on the present input at any instant of time is known as the static system. A static system is a memoryless system. But this does not mean a specific lack of movement.

For instance, a gear (Figure 1.3) or a static balance may be moved, but the system is static as the output is an algebraic form of the input (i.e., input and output are related by a constant number). Gears are well-established units of mechanical systems. Gearboxes consist of two or several gears that change the movement or vary its velocity, among other functionalities, while transmitting movement. The gear of the figure may be represented with the following equation: $\frac{\omega_1(t)}{R_1} = \frac{\omega_2(t)}{R_2}$, where ω_1 and R_1 are angular velocity and radii of the big gear and ω_2 and R_2 are angular velocity and radii of the small gear. Both angular velocities are in opposite directions. In this case, according to figure 1.2, the input to the system is $\omega_1(t)$, and the output is $\omega_2(t)$.



Figure 1.3: Gear box.

The system, represented by the relation output to input, $\frac{\omega_2(t)}{\omega_1(t)} = \frac{R_2}{R_1}$, is static because the ratio of transmission $\frac{R_2}{R_1}$ is constant.

At this level of understanding, **system dynamics** means history but not necessarily movement, which means that the current behavior depends on past actions. Information and energy flow among systems and within the environment [3,4,5]. Examples of dynamic systems are cars, electronic devices, the human body, etc. Some other examples of dynamic systems are:

- a) A bathtub is a simple example of a dynamic system. Water flows into the tub through a faucet, being the input to the system, and leaves the tub through a drain, being the output to the system (Figure 1.4).



Figure 1.4: Bathtub.

- b) Another example of a dynamic system is a pot of water set on a burner (Figure 1.5). In this case, energy, rather than matter, flows through the system.



Figure 1.5: Pot of water.

- c) A car is another example of an everyday means of a dynamic system, the current position and speed depend on the past positions and speeds.

Furthermore, **system dynamics** is a methodology and mathematical modeling technique to frame, understand, and discuss complex issues and problems. Originally developed in the 1950s to help corporate managers improve their understanding of industrial processes, system dynamics is an aspect of systems theory as a method to understand the dynamic behavior of complex systems. The basis of the method is the recognition that the structure of any system - the many circulars, interlocking, sometimes time-delayed relationships among its components - is often just as important in determining its behavior as the individual components themselves. It is also claimed that because there are often properties of the-whole which cannot be found among the properties of the-elements, in some cases, the behavior of the whole cannot be explained in terms of the behavior of the parts. This property is called an emergency property [6].

Typical **examples** of systems are electrical circuits and mechanical systems, which are well defined by first-, second- or upper-order differential equations [7, 8]. Electrical circuits can behave as static and dynamic systems. The circuit is composed just of a resistance which acts as a static system, and the circuit is composed of a resistance and capacitor in series functions as a dynamic system or circuit. Some examples are shown in figure 1.6, which includes the dynamic system composed of a resistance, a capacitor, and an inductance, apart from the ones mentioned beforehand.

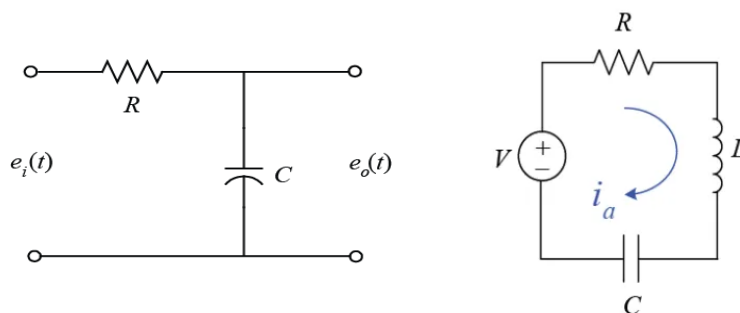


Figure 1.6: Different electrical circuits. Just resistance is a static system. By adding one capacitor to the resistance, the system works dynamically because the capacitor is charged at a certain voltage. This is a first-order system. If an inductance is also considered in the circuit, the circuit becomes a second-order system.

Another well-known example of a system for mechanical engineers is the spring-mass-damper system. In figure 1.7, a mass is attached to a spring with a damper, and a force, $F(t)$, is applied to the system, leading to some movement, sometimes oscillatory. The system is governed by the equation: $m\ddot{x} + b\dot{x} + kx = F$, where $x(t)$ represents the movement of the system (output) against the force applied (input) with $\dot{x}(t)$ being the velocity and $\ddot{x}(t)$ the acceleration.

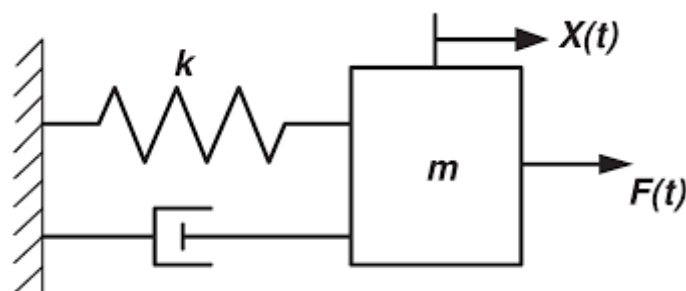


Figure 1.7: Mass-spring-damper mechanical system.

1.3 Control systems

A **control system** manages, commands, directs, or regulates the behavior of other devices or systems using control loops. It can range from a single home heating controller using a thermostat controlling a domestic boiler to large industrial control systems that are used for controlling processes or machines. Control appears in most industrial systems, but also in nature, economics, and life itself [7, 8].

The control system selects the action to get the desired behavior of the overall system to be controlled, designs the controller to generate these actions, and tunes the controller properties to adapt the system to changes.

Logic control systems for industrial and commercial machinery were historically first implemented as control systems interconnected with electrical or mechanical relays and cam timers using ladder logic [9]. They work based on sequential and combinational logic.

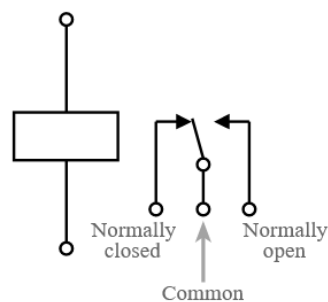


Figure 1.8: Relay with the positions normally closed or open.

Today, most such systems are constructed with microcontrollers or more specialized programmable logic controllers (PLCs). The notation of ladder logic is still in use as a programming method for PLCs [9]. Logic controllers may respond to switches and sensors and can cause the machinery to start and stop various operations using actuators. Logic controllers are used to sequence mechanical operations in many applications using open-closed (on-off) basic logic, as shown in figure 1.8. Examples include automatic elevators, car washing machines, garage doors, and other systems with interrelated operations. An automatic sequential control system may trigger a series of mechanical actuators in the correct sequence to perform a task. For example, various electric and pneumatic transducers may fold and glue a cardboard box, fill it with product, and then seal it in an automatic packaging machine. PLC software can be written in many ways: ladder diagrams, SFC (sequential function charts), or statement lists.



Figure 1.9: Three first steps of the car wash machine example.

One example is the automated car wash machine in Figure 1.9, which sequentially follows the following steps: 1) The car entry sensor triggers the process and turns the conveyor belt on to move the car. 2) The soapy water sprinkler turns on when the stage 1 sensor detects the car and does the process for a certain time. 3) The brusher turns on when stage 2 sensors detect the car and do the process for a certain time. 4) The clean water sprinkler turns on when stage 3 sensors detect the car and do the process for a certain time. 5) The dryer turns on when stage 4 sensors detect the car and do the process for a certain time. 6) The car exit sensor triggers the conveyor motor to turn off.

Furthermore, on-off control uses a feedback controller that switches abruptly between two states. A simple bi-metallic domestic thermostat can be described as an on-off controller. When the temperature in the room, process variable (PV), goes below the user setting, set point (SP), the heater is switched on. Another example is a pressure switch on an air compressor. When the pressure (PV) drops below the setpoint (SP), the compressor is powered. Refrigerators and vacuum pumps contain similar mechanisms. Simple on-off control systems like these can be cheap and effective [7, 8].

For continuously modulated control, a feedback controller is used to automatically control a process or operation. The control system compares the value or status of the process variable (PV) being controlled with the desired value or setpoint (SP) and applies the difference as a control signal to bring the process variable output of the plant to the same value as the set point. **Feedback control plays a key role in control systems when the model of the system is not accurately known** [7, 8].

There are two common classes of control action: **open loop** and **closed loop** [7, 8].

- a) In an open-loop control system (Figure 1.9), the control action from the controller is independent of the process variable. An example of this is a central heating boiler controlled only by a timer. The control action is the switching on or from the boiler. The process variable is the building temperature. This controller operates the heating system for a constant time, regardless of the temperature of the building.

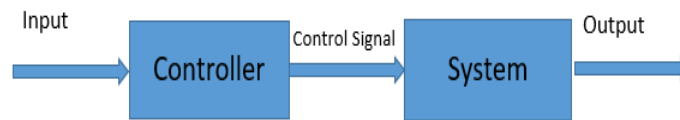


Figure 1.10: Open-loop block diagram.

- b) In a closed-loop control system (Figure 1.11), the control action from the controller is dependent on the desired and actual process variables. In the case of the boiler analogy, this would utilize a thermostat to monitor the building temperature and feedback a signal to ensure the controller output maintains the building temperature close to that set on the thermostat. A closed loop controller has a feedback loop, which ensures the controller exerts a control action to control a process variable at the same value as the set point. For this reason, closed-loop controllers are also called feedback controllers.

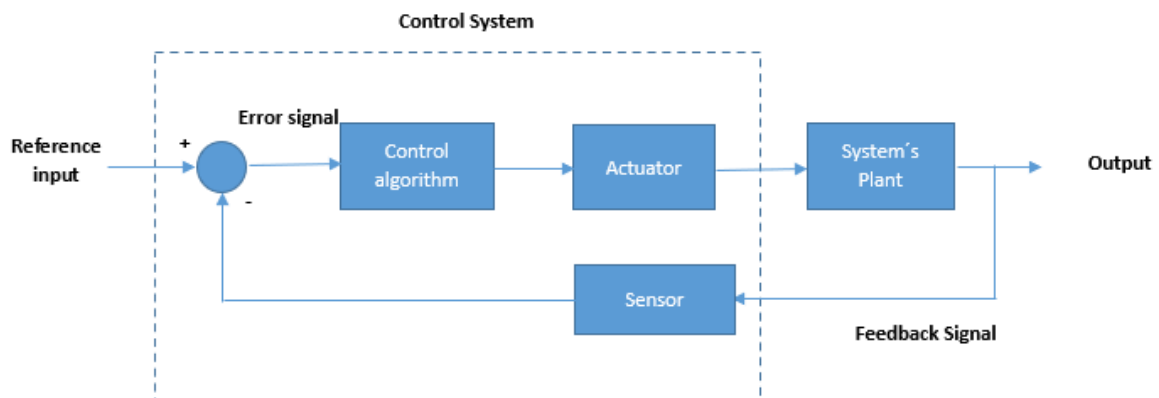


Figure 1.11: Control System Feedback Loop.

In the case of feedback systems, a control loop including sensors, control algorithms, and actuators is arranged to regulate a variable at a setpoint (SP) or reference input. An everyday example is the cruise control on a road vehicle; where external influences such as hills would cause speed changes, and the driver could alter the desired set speed. The proportional-integral-derivative (PID) algorithm in the controller restores the actual speed to the desired speed in the optimum way, with minimal delay or overshoot, by controlling the power output of the vehicle's engine [10].

Control systems that include some sensing of the results they are trying to achieve make use of feedback and can adapt to varying circumstances to some extent. Open-loop control systems do not make use of feedback and run only in pre-arranged ways. These concepts are further reinforced in this manual in the coming chapters.

References:

- [1] Ziemer, R.E., Tranter, W.H., and Fannin, D.R. (1983), Signals and Systems: Continuous and Discrete.
- [2] Ziemer, R.E. and Tranter, W.H., and Fannin, D.R. (1989), Signals and Systems: Continuous and Discrete. ISBN: 9780024316301. Macmillan.
- [3] Soliman, Samir S. and Srinath, Mandyam D. (1990), Continuous and Discrete Signals and Systems, ISBN: 0131712578, Prentice-Hall, Inc.
- [4] Mandal, M., and Asif, A. (2007), Continuous and discrete-time signals and systems. ISBN: 9781108477864. Cambridge University Press.
- [5] Oppenheim, Alan V., and Willsky, Alan S. and Nawab, S. Hamid (1996). Signals and Systems (2nd Ed.), ISBN: 0138147574, Prentice-Hall, Inc.
- [6] Morgan Kaufmann, Principles of Computer System Design, Chapter 1: Systems, 2009, Pages 1-42, ISBN: 9780123749574, <https://doi.org/10.1016/B978-0-12-374957-4.00010-4>.
- [7] Kuo, Benjamin C. (1975). Automatic control systems. Englewood Cliffs, N.J. : Prentice-Hall.
- [8] Ogata, K. (2010) Modern Control Engineering. 5th Edition, Pearson, Upper Saddle River.
- [9] Mulindi, J. (2020). The Introduction to Programmable Logic Controllers for Beginners: A Transition from Relay Control Systems to PLC Systems. ISBN: 979-8622471582.
- [10] <https://ctms.engin.umich.edu/CTMS/index.php?example=CruiseControl§ion=ControlPID>

Chapter 2: Mathematical modeling

This chapter introduces the mathematical foundations of every part, which consists of control system loops, namely, signals, systems, and control. Herewith, an insightful introduction is presented, which allows the reader to gain a perceptive understanding of them.

- ✓ Models of Signals.
- ✓ Models of Systems.
- ✓ Models of Control Systems.

The terminology for signals and systems is as follows: a system is any process that generates an output signal in response to an input signal. An independent variable is a variable (often denoted by $x(t)$) whose variation does not depend on that of another. Continuous signals are usually represented with parentheses, while discrete signals use brackets. All signals use lowercase letters, reserving the uppercase for the frequency domain (which will be presented in later sections). Unless there is a better name available, the input signal is called $x(t)$ or $x[n]$, while the output is called $y(t)$ or $y[n]$. In this text, it is intended to deal with continuous-time signals, systems, and control algorithms, as well as an introductory chapter to discrete-time. Figure 2.1 represents the continuous and discrete time systems and the signals associated with them.

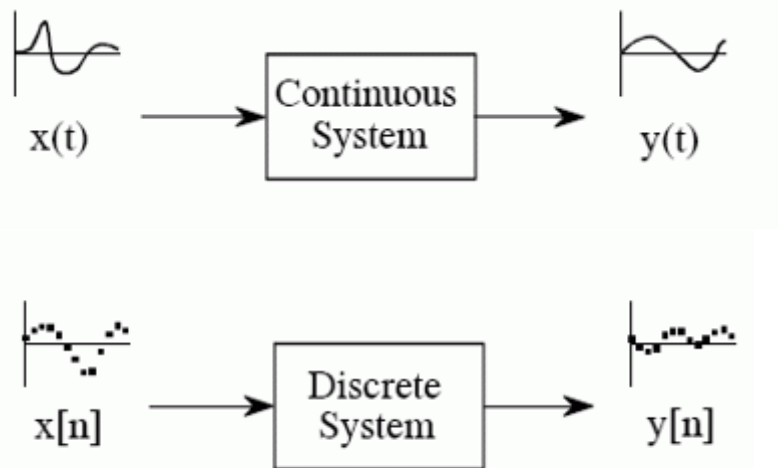


Figure 2.1: A system is any process that generates an output signal in response to an input signal.

2.1 Models of Signals

In this section, some mathematical functions are provided that represent the signals that are input to the system. These signals are usually used to test or define the behavior of the system. Normally, any system can accept any kind of signal as input, and they are not restricted to the ones mentioned in this section. They are classified from the classical point of view and according to the signal processor.

2.1.1 Classification of signals according to signal processors

In a more general way and according to the dealt system, signals may be classified as:

a) Binary: The simplest possible signal of any kind that can be employed to transmit messages, the binary signal consists of **only two possible values**. These values are represented by the binary digits, or bits, 1 and 0, as shown in Figure 2.2.

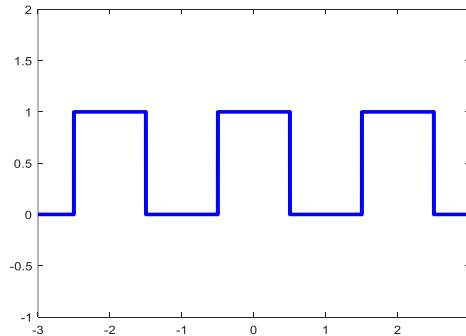


Figure 2.2: Binary signal

b) Continuous: A continuous signal or a continuous-time signal is a **varying quantity (a signal) whose domain, which is often time, is a continuum** (e.g., a connected interval of the reals). That is, the function's domain is an uncountable set of elements. To contrast, a discrete-time signal has a countable domain, like the natural numbers. As an example, Figure 2.3 contains a continuous-time sinusoidal signal.

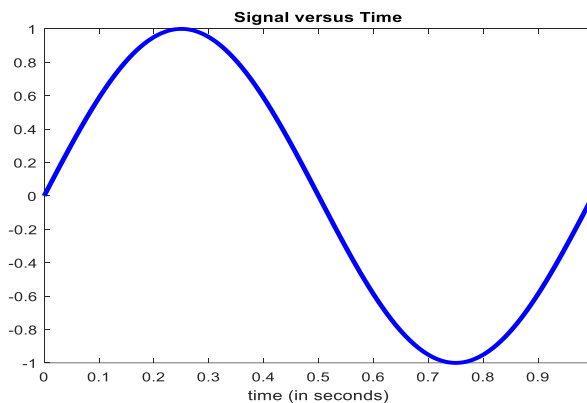


Figure 2.3: Continuous-time sinusoidal signal

c) Digital: A digital signal is a **signal that represents data as a sequence of discrete values**; at any given time, it can only take on, at most, one of a finite number of values. Simple digital signals represent information in discrete bands of analog levels (Figure 2.4). All levels within a band of values represent the same information state.

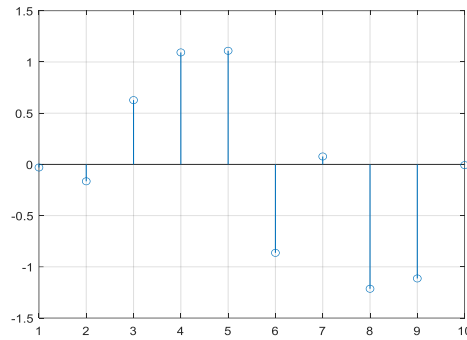


Figure 2.4: Digital signal

d) Stochastic signals: A stochastic signal (Figure 2.5) is used to describe a non-deterministic signal, i.e., **a signal with some kind of uncertainty**. A random signal is, by definition, a stochastic signal with whole uncertainty, i.e., with an auto-correlation function with an impulse at the origin and a power spectrum completely flat.

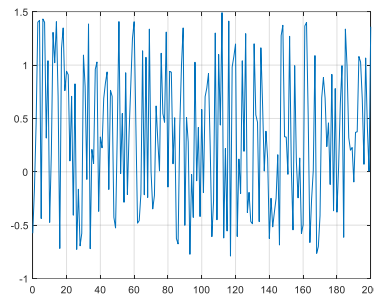


Figure 2.5: Stochastic signal.

2.2.2 Types of signals in control

In classic control theory, signals may be classified as, **step, ramp, parabolic, sinusoidal, and impulse signals**. These signals are willing to check the stability and robustness of the system.

a) Unit Step Function or Heaviside Functions: The unit step function or Heaviside step function is denoted by $u(t)$ or, $\theta(t)$. It is defined as:

$$u(t) = \begin{cases} 1 & t \geq 0 \\ 0 & t < 0 \end{cases} \quad (2.1)$$

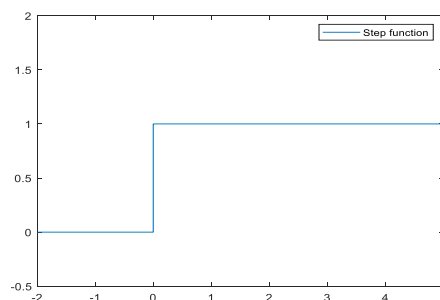


Figure 2.6: Continuous-time step function.

Figure 2.6 shows a continuous-time step function. The step function is used as the best test signal. The area under the unit step function is unity. Step functions can be used to define staircase functions, which can have any form of the following aspect:

$$u(t) = \begin{cases} 7 & t \geq 8 \\ 12 & 6 \leq t < 8 \\ -4 & t < 6 \end{cases} \quad (2.2)$$

b) Ramp Signal: When a signal gives the constant velocity of the actual input signal, it is known as a velocity signal or function. The continuous-time unit ramp signal is that function that starts at $t = 0$ and increases linearly with time. It is denoted by $r(t)$. Mathematically, the continuous-time unit ramp signal is defined as follows from the unit step function:

$$r(t) = \begin{cases} t & t \geq 0 \\ 0 & t < 0 \end{cases} \quad (2.3)$$

From the above equation, the ramp signal is a signal whose magnitude varies linearly. The graphical representation of the continuous-time unit ramp signal is shown in figure 2.7.

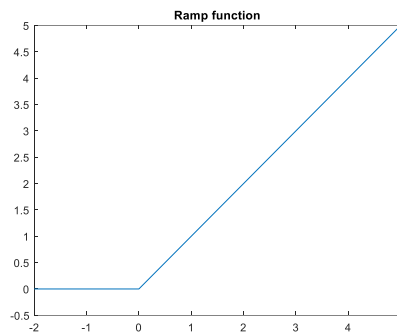


Figure 2.7: Continuous ramp signal.

Note that the relationship between step and ramp functions is, $r(t) = \int u(t)dt = \int dt = t, t > 0$ or $u(t) = \frac{dr(t)}{dt}$.

c) Parabolic signal: When a signal gives the constant acceleration distinction of an actual input signal, such a signal is known as a **parabolic signal** or **parabolic function**. It is also known as unit **acceleration signal**. The unit parabolic signal starts at $t = 0$.

The continuous-time unit parabolic signal is a unit parabolic signal that is defined for every instant of positive time. It is denoted by $p(t)$. Mathematically, $p(t)$ is given as:

$$p(t) = \begin{cases} \frac{t^2}{2} & t \geq 0 \\ 0 & t < 0 \end{cases} \quad (2.4)$$

The graphical representation of the continuous-time parabolic signal $p(t)$ is shown in figure 2.8.

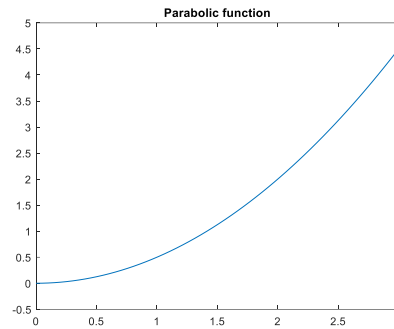


Figure 2.8: Continuous-time parabolic function

d) Sinusoidal signals: Sinusoidal signals can be defined as a periodic signal with a waveform as similar to that of a sine wave. Sinusoidal are the smoothest signals with no abrupt variation in their amplitude; the amplitude witnesses gradual change with time. If it is considered a sinusoidal signal $y(t)$ having an amplitude A , frequency f , and phase of quantity θ , then the signal can be represented as:

$$y(t) = A \sin(2\pi ft + \theta) \quad (2.5)$$

It has the form previously shown in figure 2.3. Sinusoidal signals are inputs to numerous systems in engineering, such as rotary machines and motors, alternating current or voltage circuits, and many other applications.

e) Another important signal is the *impulse signal, or Dirac delta function*. The notion of a delta function is extremely useful in the analysis of signals and systems, although it may feel unnatural on first exposure. Although the concept of the delta function can be made completely rigorous, rather than get sidetracked with too much mathematical detail and sophistication, the aim here is to provide some intuition and ability to work with the delta function. On the other hand, it is important to have enough rigor so that this important tool is used properly.

When Heaviside functions are introduced, it is noted that they are switches that change the function at specific times. However, Heaviside functions are not suited to forcing functions that exert a “large” force over a “small” time frame.

Examples of this kind of forcing function would be a hammer striking an object or a short in an electrical system. In both cases, a large force (or voltage) would be exerted on the system over a very short time frame. The Dirac Delta function is used to deal with these kinds of forcing functions.

There are many ways to define the Dirac Delta function, but there are three main properties of the Dirac Delta function that we need to be aware of. These are,

1. $\delta(t - a) = 0$ if $t \neq a$. (2.6)

2. $\int_{a-\varepsilon}^{a+\varepsilon} \delta(t - a) dt = 1$ with $\varepsilon > 0$. (2.7)

3. $\int_{a-\varepsilon}^{a+\varepsilon} f(t)\delta(t - a) dt = f(a)$ with $\varepsilon > 0$. (2.8)

At $t=a$, the Dirac Delta function is sometimes thought of as having an “infinite” value. So, the Dirac Delta function is a function that is zero everywhere except one point, and at that point it can be thought of as either undefined or as having an “infinite” value.

Note that the integrals in the second and third properties are true for any interval containing $t=a$ ($a=0$, in figure 2.9), provided that a is not one of the endpoints. The limits given are needed to prove the properties, and so they are also given in the properties. It is supposed that they are true, provided that the integration of the function contains $t=a$ over an interval.

This is a very strange function. It is zero everywhere except one point, and yet the integral of any interval containing that one point has a value of 1. The Dirac Delta function is not a real function as we think of them. It is instead an example of something called a **generalized function** or **distribution**.

Despite the strangeness of this “function”, it does a very nice job of modeling sudden shocks or large forces in a system.

Note that often the second and third properties are given with limits of infinity and negative infinity, but they are valid for any interval in which $t=a$ is in the interior of the interval.

In figure 2.9, the Dirac delta function is shown, such that t_1 tends to infinity and $a=0$.

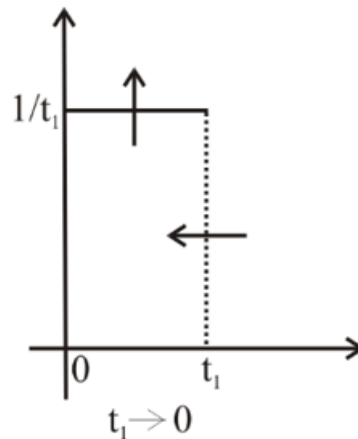


Figure 2.9: Dirac's delta function.

These are some examples, but the types of signals are not limited to them.

2.2 Fourier or frequency analysis of signals

An important aspect of signals is their representation in the frequency domain which gives relevant information. First, it is shown the information given by a periodic function and later it is extended to any function.

A function $f(x)$ is called **periodic function** with period p , if for all x there holds

$$f(x + p) = f(x) \quad (2.9)$$

If $f(x)$ has a period p , and sub-periods at $2p$, $3p$, and so on. In general,

$$f(x + np) = f(x) \quad (2.10)$$

Where $n=1,2,3,\dots$

In what follows it will be considered the representation of functions $f(x)$ of period 2π . Suppose that $f(x)$ is a function of period 2π that can be represented by a convergent series of the form:

$$f(x) = a_0 + a_1 \cos x + b_1 \sin x + a_2 \cos 2x + b_2 \sin 2x + \dots \quad (2.11)$$

Then the *Fourier series* of $f(x)$ is given by

$$f(x) = a_0 + \sum_{n=1}^{\infty} (a_n \cos nx + b_n \sin nx) \quad (2.12)$$

Where a_0 , a_n , and b_n are called Fourier coefficients.

Assuming $f(x)$ is known, the Fourier coefficients can be computed using the Euler formulas:

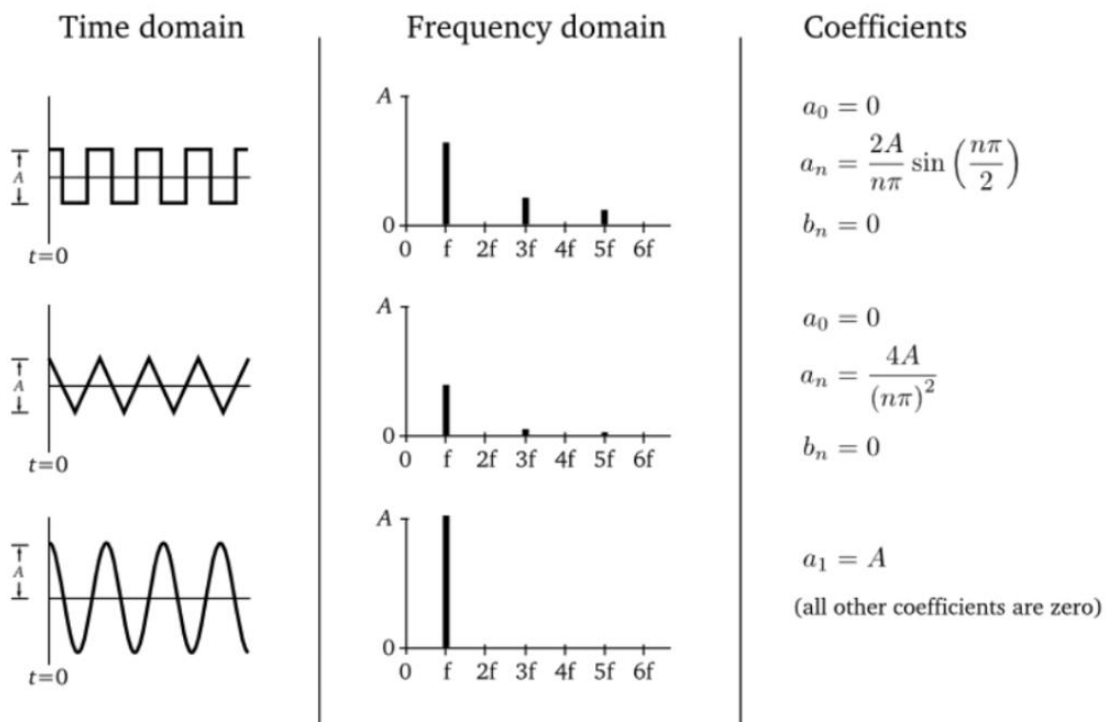
$$a_0 = \frac{1}{2\pi} \int_{-\pi}^{\pi} f(x) dx \quad (2.13)$$

$$a_n = \frac{1}{\pi} \int_{-\pi}^{\pi} f(x) \cos(nx) dx \quad (2.14)$$

$$b_n = \frac{1}{\pi} \int_{-\pi}^{\pi} f(x) \sin(nx) dx \quad (2.15)$$

Where $n=1,2,3\dots$. These coefficients comprise the **spectrogram** of the studied signal.

Some examples for spectrograms of common signals are the following:



The importance of the Fourier analysis relies on that every periodic function or signal has a corresponding Fourier series representation, if the function fulfils the Dirichlet conditions for the convergence of Fourier series which state that: 1) **The function must be single-valued, periodic, and finite**, 2) it should have a finite number of maxima and minima in any given period, and 3) it should have a finite number of discontinuities, but the discontinuities should not be infinite.

Moreover, constant signals can be seen as periodic function with relatively long periods of time. Therefore, every periodic signal has a Fourier series representation, that is, every periodic signal can be seen as a combination of fundamental harmonic functions.

The general Fourier transform is defined as:

$$F(k) = \int_{-\infty}^{\infty} f(t) e^{-2\pi jkt} dt \quad (2.16)$$

which reveals the frequency decomposition of a signal that does not need to be periodic. As a result, any kind of input to the system can be dealt with Fourier analysis independently of the previous classification. Note that the integration is from $-\infty$ to ∞ , i.e., it is assumed to have knowledge of the function $f(t)$ for $t \in (-\infty, \infty)$. In practice, it rarely has this knowledge for signals, and it is operated on a limited time interval or window of time $t \in [t_a, t_b]$.

2.3 Models of Systems

A system is any process that generates an output signal in response to an input signal. It is normally composed of a set of interconnected elements. The whole system's behavior is more important than the sum of its parts. Systems are inter-interactive among them and represent dynamics.

2.3.1 Classification of Systems

a) Physical systems: the system is presented by its physical structure; how the system is composed of its parts is shown in real life. The power of linear dynamic systems analysis is that many types of different nature systems can be modeled with the same type of differential equation, so the analysis of different physical systems can use the same approach. This analogy among physical systems facilitates the study and control of dynamic systems from the perspective of control.

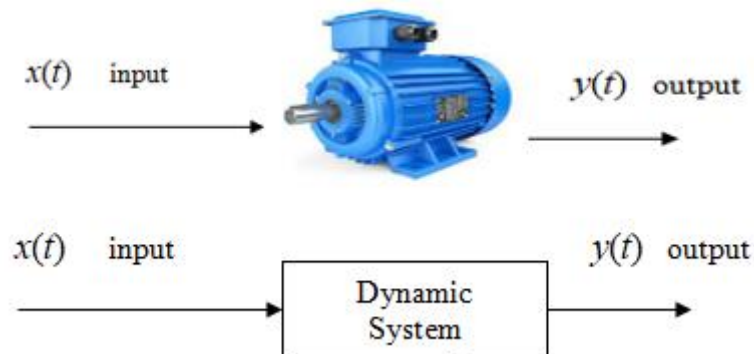


Figure 2.10: Dynamic system representation

b) Schematic representation: A schematic is defined as a picture that shows something in a simple way, using symbols. A **schematic diagram** is a picture that represents the components of a process, device, or other object using abstract, often standardized symbols and lines. Schematic diagrams only depict the significant components of a system; though some details in the diagram may also be exaggerated or introduced to facilitate the understanding of the system, the system is represented by its different parts.

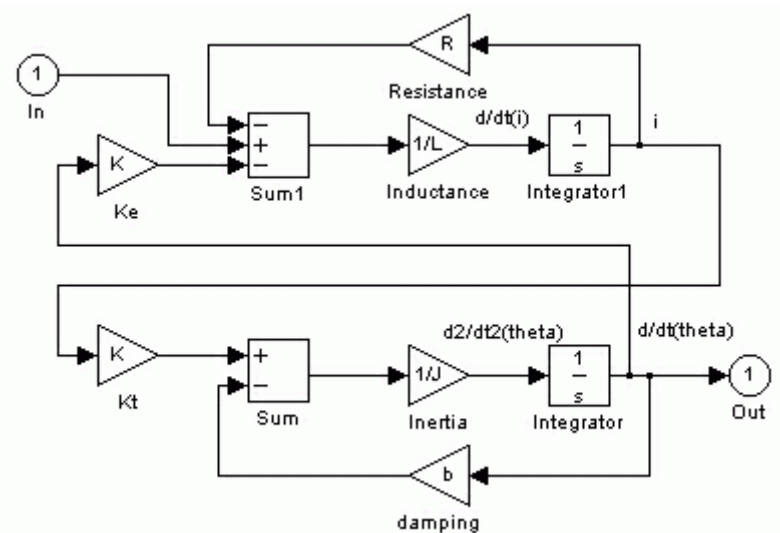


Figure 2.11: Simulink model motor composed by integration of electrical and mechanical equivalent circuits [1].

2.3.2 Properties of systems

a) Linearity: A general deterministic system can be described by an operator, H , that maps an input, $x(t)$, as a function of t to an output, $y(t)$, a type of black box description (see figure 1.1).

A system is linear if and only if it satisfies the superposition principle, or equivalently both the additivity and homogeneity properties, without restrictions (that is, for all inputs, all scaling constants, and all time).

The superposition principle means that a linear combination of inputs to the system produces a linear combination of the individual zero-state outputs (that is, outputs setting the initial conditions to zero) corresponding to the individual inputs.

Mathematically, for a continuous-time system, given two arbitrary inputs $x_1(t), x_2(t)$ as well as their respective zero-state outputs, $y_1(t) = H\{x_1(t)\}$ and $y_2(t) = H\{x_2(t)\}$ then a linear system must satisfy:

$$\alpha y_1(t) + \beta y_2(t) = H\{\alpha x_1(t) + \beta x_2(t)\}$$

for any scalar values α and β , for any input signals $x_1(t)$ and $x_2(t)$, and for all time t .

b) Non-linearity: A system is defined to be nonlinear if the laws governing the time evolution of its state variables depend on the values of these variables in a manner that deviates from proportionality, i.e., a system that does not fulfill the additivity and homogeneity properties.

c) Multi-variable system: A system which have more than one input and one or more outputs, since such systems have more than one variable, they are called multivariable systems.

d) Time-invariant system: A time-invariant system is one whose behavior (its response to inputs) does not change with time. The concept is blurry, but it is mathematically acceptable.

2.4 Description of systems by ordinary non-homogeneous differential equations with constant coefficients

The dynamic performance of physical systems is obtained by utilizing the physical laws of mechanical, electrical, fluid, and thermodynamic systems. Physical systems are modelled with linear differential equations with constant coefficients when possible. Other models can be derived from more general differential equations.

2.4.1 First-order systems

The mathematical model of heat transformation in fluids, tank systems, RC circuits, etc. may be represented by **first-order differential equations**. The analogies of these systems make them to behave in similar way depending on the parameters which define their dynamics.

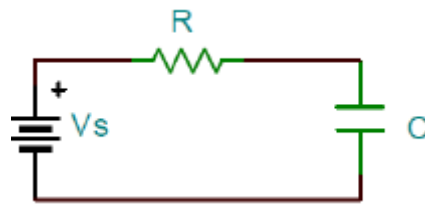


Figure 2.12: A series RC circuit driven by a voltage source.

A **RC circuit** is represented in figure 2.12. A series RC network is connected across a constant voltage source, V_s (Figure 2.12). Kirchhoff's voltage law is used to model the circuit behavior as voltage balance:

$$V_R + V_C = V_s \quad (2.17)$$

V_R is the voltage over the resistance, V_C is the voltage within the capacitor, and V_s is the voltage provided to the circuit. The voltage within the capacitor can be calculated as:

$$V_C(t) = \frac{Q(t)}{C} \quad (2.18)$$

Where the accumulated charge is, $Q(t) = \int_{-\infty}^t I(\tau) d\tau$, the current over the circuit is expressed as $I(t) = C \frac{dV_C}{dt}$. So, Kirchhoff's law results in:

$$V_s(t) = V_C(t) + RC \frac{dV_C}{dt} \quad (2.19)$$

and the system can be represented as the following first-order differential equation:

$$\frac{dV_C}{dt} = \frac{1}{RC} (V_s(t) - V_C(t)) \quad (2.20)$$

or,

$$\frac{dV_C}{dt} + \frac{1}{RC} V_C(t) = \frac{1}{RC} V_s(t) \quad (2.21)$$

Another system that is represented by a first-order differential equation is the water tank shown in Fluid Mechanics. Please note that every physical system that is represented as

first-order system may be subject to the same mathematical approach due to their analogies.

A **water tank** is represented in figure 2.13, where the average velocity of the jet is approximated as $V = \sqrt{2g/D_{tank}h}$, where h is the height of the tank measured from the center of the hole, g is the gravitational acceleration, and D_{tank} is the diameter of the tank. The conservation of mass (mass balance) relation for a control volume (CV, the volume of the tank in this case) undergoing any process is given in rate form as:

$$\frac{dm_{CV}}{dt} = \dot{m}_i - \dot{m}_e \quad (2.22)$$

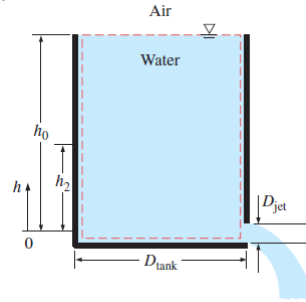


Figure 2.13: Water tank scheme.

During this process, no mass enters the control volume ($\dot{m}_i=0$), and the mass flow rate of discharged water is,

$$\dot{m}_e = (\rho VA)_{out} = \rho \sqrt{2g/D_{tank}h} A_{jet} \quad (2.23)$$

Where $A_{jet} = \pi D_{jet}^2/4$ is the cross-sectional area of the jet, which is constant. Noting that the density of the water is constant, the mass of water in the tank at any time is:

$$m_{CV} = \rho V = \rho A_{tank} h \quad (2.24)$$

Where $A_{tank} = \pi D_{tank}^2/4$ is the base area of the cylindrical tank. Substituting eqs. (2) and (3) into (1), mass balance eq. (1) gives:

$$-\rho \sqrt{2gh} A_{jet} = \frac{d(\rho A_{tank} h)}{dt} \rightarrow -\rho \sqrt{2gh} \left(\pi D_{jet}^2/4 \right) = \frac{d(\rho (\pi D_{tank}^2/4) h)}{dt} \quad (2.25)$$

$$-\sqrt{2g/D_{tank}h} \left(\frac{D_{jet}^2}{D_{tank}^2} \right) = \frac{dh}{dt} \quad (2.26)$$

Or,

$$\frac{dh}{dt} + \sqrt{2g/D_{tank}h} \left(\frac{D_{jet}^2}{D_{tank}^2} \right) = 0 \quad (2.27)$$

Please note the equation depends on the relation of the average velocity of the jet and the height of the water in the tank, $V = f(h)$. The simplest model for resistance is a so-called linear leak: that $f(h)$ is proportional to h , which is represented in this example.

In case $\dot{m}_i \neq 0$, the input to the system is considered and the previous equation takes the form:

$$\frac{dh}{dt} + \sqrt{2g/D_{tank}}h \left(\frac{D_{jet}^2}{D_{tank}^2} \right) = \dot{m}_i \quad (2.28)$$

Another interesting system that is represented by a first-order differential equation is the cooling and heat transfer from a solid body to a fluid.

Figure 2.14, [2], represents **a solid which absorbs or transfer heat from a fluid**. These systems may be represented mathematically by Newton’s cooling law, which models heat flow in fluids by convection:

$$q\alpha(T_a - T_b) = h(T_a - T_b) \quad (2.29)$$

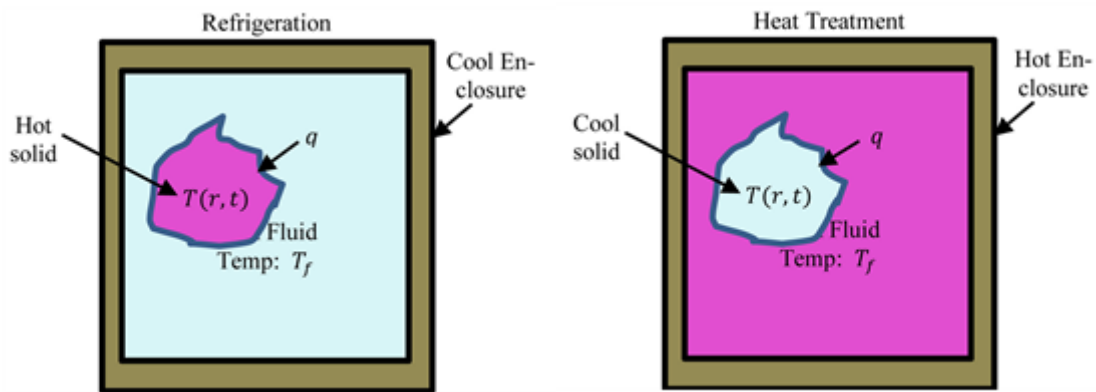


Figure 2.14: Heat transferring solids submerged in a fluid [2].

Equation (2.29) show heat flows from point A (solid) to point B (fluid) as $T_a > T_b$, which express the heat flux between points A and B, being h heat transfer coefficient. Thus

$$q = h(T_a(t) - T_f) = h(T(t) - t_f) \quad (2.30)$$

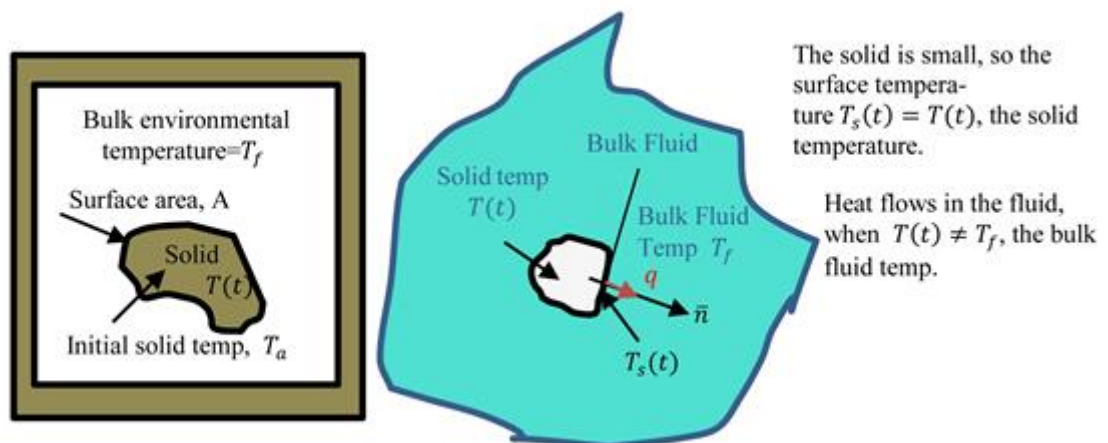


Figure 2.15: Mathematical model of heat transfer [2].

Equation (2.31) satisfies the heat transformation between systems in A and B (solid and bulk fluid). The first law of thermodynamics dictates that, to produce temperature change in a solid $\Delta T(t)$ during the time-period Δt

$$Q = -\rho c V \Delta T(t) = q A_s \Delta t = h A_s (T(t) - T_f) \Delta t \quad (2.31) \text{ or,}$$

$$\frac{dT(t)}{dt} = -\frac{h}{\rho c V} A_s (T(t) - T_f) = -\alpha A_s (T(t) - T_f) \quad (2.32)$$

As h, ρ, c, V are constants. Thus, the change of the temperature of the submerged solid $T(t)$ is continuous respect to time t , i.e., $\Delta t \rightarrow 0$ and if we replace the constant surface area as to a generic symbol A , Eq. (2.31) is expressed in the form of first order differential equation Eq. (2.32), and adding initial conditions leads to:

$$\frac{dT(t)}{dt} = -\alpha A_s (T(t) - T_f) \quad (2.33) \text{ with initial conditions. } T(t)|_{t=0} = T(0) = T_0.$$

The initial temperature of an object is 80°C . It is place in a refrigerator which is keeping at $T_f = 5^\circ\text{C}$. If $\alpha = 0.002$ and $A_s = 0.2\text{m}^2$, find the time in which the object cools down.

$$\frac{dT(t)}{dt} = -\alpha A_s (T(t) - T_f) \quad (2.34) \text{ with initial conditions } T(t)|_{t=0} = T(0) = T_0 = 80^\circ\text{C};$$

$$\frac{dT(t)}{(T(t)-T_f)} = -\alpha A_s dt; \quad (2.35)$$

integrating $T(t) = T_f + K e^{-\alpha A_s t}$ (2.36).

By using $T_0 = 80^\circ\text{C}$, the integration constant $K=75$, the solution of the equation is:

$$T(t) = 5 + 75e^{-0.0004t} \quad (2.37)$$

In the frequency domain, the entrance to the system is T_f and the output is $T(t)$. The equation:

$$\frac{dT(t)}{dt} + \alpha A_s T(t) = \alpha A_s T_f \quad (2.38).$$

2.4.2 Second order systems

Linear ordinary differential equations which describe physical systems in a broad variety of disciplines such as mechanical or electrical engineering, or biological systems are the foundation of control systems. **Second order differential equation** is a specific type of differential equation that consists of a derivative of a function of order 2 and no other higher-order derivative of the function appears in the equation. Typical mechanical systems (see figure 3.3) are described by first or second order differential equations.

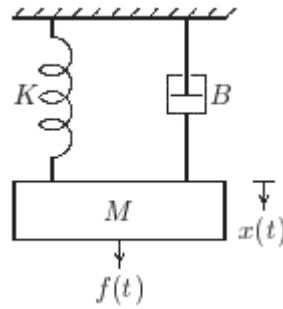


Figure 2.16: Typical second order mechanical system defined by spring, k , mass, m , and damper k , applying a force $f(t)$.

If $x(t)$ is the displacement from the resting position and $u(t)$ is the applied force, using 2nd Newton law, the motion of the system is described by the equation:

$$m\ddot{x}(t) + b\dot{x}(t) + kx(t) = f(t) \quad (2.39)$$

where $\ddot{x}(t)$ is the second derivative, and $\dot{x}(t)$ the first derivative respect to time. It is supposed that initial conditions are known; $x(t=0) = x_0$ and $\dot{x}(0) = \dot{x}_1$ being x_0 and \dot{x}_1 known numbers. This equation is further studied in the next chapter in the time-domain and frequency domain, using the Laplace transform, to depict the system behavior.

Another useful representation of the linear differential equations is the state variable description which reduces higher order differential systems to an equivalent set of first equation systems. As illustrative example, consider the differential equation given in (1). Let $x_1(t) = x(t)$, $x_2(t) = \dot{x}(t)$ be the new variables called **state variables**. Then, the system may be described as:

$$\dot{x}_1(t) = x_2(t) \quad \text{and} \quad \dot{x}_2(t) = -\frac{b}{m}x_2(t) - \frac{k}{m}x_1(t) + \frac{1}{m}f(t) \quad (2.40)$$

with initial conditions $x_1(0) = y_0$ and $x_2(0) = y_1$, equation (2) leads to:

$$\begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -k/m & -b/m \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} + \begin{bmatrix} 0 \\ 1/m \end{bmatrix} u(t); \quad y(t) = [1 \ 0] \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} \quad (2.41)$$

which general form is

$$\dot{x}(t) + Ax(t) = Bu(t); \quad y(t) = Cx(t) + Du(t) \quad (2.42)$$

Here $x(t)$ is a 2x1 vector (a column vector) with elements the two state variables $x_1(t)$ and $x_2(t)$. It is called the *state vector*. The variable $u(t)$ is the *input* and $x(t)$ the *state* of the system. The first equation is a vector differential equation called the *state equation*. The second equation is an algebraic equation called the *output equation*. In the above example $D=0$; D is called the *direct link*, as it directly connects the input to the output, as opposed to connecting through $x(t)$ and the dynamics of the system. The above description is the *state variable or state space description* of the system. The advantage is that system descriptions can be written in a standard form (the state space form) for which many mathematical results exist.

2.4.3 Higher order systems

The systems we deal with may be any order, not just first and second. So, they may be presented as linear non-homogeneous differential equations of order n with constant coefficients. These equations may be written as:

$$y^{(n)}(x) + a_1 y^{(n-1)}(x) + \dots + a_{n-1} y'(x) + a_n y(x) = b_0 \quad (2.43)$$

where a_1, a_2, \dots, a_n are constants which may be real or complex.

Every linear homogeneous differential equation with constant coefficients may be treated as follow. Using the linear differential operator $L(D)$, this equation can be represented as $L(D)y(x) = b_0$, where $L(D) = D^n + a_1 D^{n-1} + \dots + a_{n-1} D + a_n$ (3.21). For each differential operator with constant coefficients, we can introduce the characteristic polynomial:

$$L(\lambda) = \lambda^n + a_1 \lambda^{n-1} + \dots + a_{n-1} \lambda + a_n \quad (2.44)$$

which is called the *characteristic equation* of the differential equation.

According to the fundamental theorem of algebra, a polynomial of degree n has exactly n roots, counting multiplicity. In this case the roots can be both real and complex. Normally, **higher order systems are represented as combination of first and second order systems** having the properties of the appropriate mixture.

The model of a DC motor is an example of three degree of freedom. It directly provides rotary motion and, coupled with wheels or drums and cables, can offer translational motion. The electric equivalent circuit of the armature and the free-body diagram of the rotor are shown in figure 3.6.

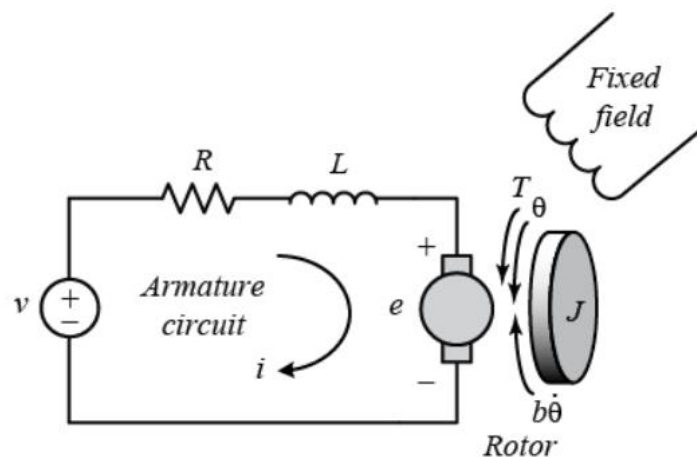


Figure 2.17: DC armature motor schematic representation [3].

Typical parameters of the motor are given by [3]:

$$\begin{aligned} J &= 3.2284e-6; \% \text{ Rotacional inertia [Nms}^2\text{/rad]} \\ b &= 3.5077e-6; \% \text{ Viscous friction [Nms/rad]} \\ K_T &= 0.0274; \% \text{ Torque constant [Nm/A]} \\ K_e &= 0.0274; \% \text{ back emf constant [Vs/rad]} \\ R &= 4; \% \text{ Armature resistance [\Omega]} \end{aligned}$$

$$L = 2.75e-6; \% \text{ Armature inductance [H]}$$

The input to the system is the Voltage (V) applied to the motor armature, while the output is the position of the shaft (θ). The rotor and the shaft are assumed to be rigid. It is further assumed a viscous friction model, that means, the friction torque is proportional to the shaft angular velocity.

In general, the torque generated by a current motor is proportional to the armature current and the strength of the magnetic field. It is assumed the magnetic field to be constant and therefore the motor torque, T , is proportional to the armature torque by a proportional factor, K_T , as shown in the equation. This is referred to as an armature-controlled motor:

$$T = K_T i \quad (2.45).$$

The back emf, e , is proportional to the angular velocity of the shaft by a constant factor K_b ,

$$e = K_b \dot{\theta} \quad (2.46).$$

In SI units, K_T and K_b are equal, and it is assumed $K = K_T = K_b$

From 2nd Newton's law and Kirchhoff's voltage law, the governing equations are derived:

$$J\ddot{\theta} + b\dot{\theta} = Ki \quad (2.47)$$

$$L \frac{di}{dt} + Ri = V - K\dot{\theta} \quad (2.48).$$

2.5 State space representation

The state space representation has the following characteristics for a generic linear, time invariant, n-dimensional system.

The state space representation of a system is given by two equations :

$$\dot{x}(t) = Ax(t) + Bu(t); \quad (2.49)$$

$$y(t) = Cx(t) + Du(t) \quad (2.50)$$

The first equation is called the state equation, the second equation is called the output equation. For an n^{th} order system (i.e., it can be represented by an n^{th} order differential equation) with r inputs and m outputs the size of each of the matrices is as follows:

- x is $nx1$ (n rows by 1 column vector); x is called the state vector, it is a function of time
- A is nxn ; A is the state matrix, a constant
- B is $n \times r$; B is the input matrix, a constant
- u is $rx1$; u is the input, a function of time
- C is $m \times n$; C is the output matrix, a constant
- D is $m \times r$; D is the direct transition (or feedthrough) matrix, a constant

- y is $m \times 1$; y is the output, a function of time

The state equation has a single first order derivative of the state vector on the left, and the state vector, $x(t)$, and the input $u(t)$ on the right. There are no derivatives on the right-hand side. The output equation has the output on the left, and the state vector, $x(t)$, and the input $u(t)$ on the right. There are no derivatives on the right-hand side.

For systems with a single input and single output (i.e., most of the systems we will consider) these variables become (with $r=1$ and $m=1$):

$$\dot{x}(t) = Ax(t) + Bu(t); \quad (2.51)$$

$$y(t) = Cx(t) + Du(t); \quad (2.52)$$

The notation of the state space is very compact. Even large systems can be represented by two simple equations. Because all systems are represented by the same notation, it is very easy to develop general techniques to solve these systems. Also, computers easily simulate first order equations.

For the case of the DC armature controller motor, the transfer function or the equations of the system are represented in the state space as:

$$\frac{d}{dt} \begin{bmatrix} \theta \\ \dot{\theta} \\ i \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & -b/J & K/J \\ 0 & -K/L & -b/L \end{bmatrix} \begin{bmatrix} \theta \\ \dot{\theta} \\ i \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 1/L \end{bmatrix} \quad (2.53)$$

$$y = [1 \quad 0 \quad 0] \begin{bmatrix} \theta \\ \dot{\theta} \\ i \end{bmatrix} \quad (2.54)$$

2.6 Control Systems

A control system is defined as a system of devices that manages, commands, directs, or regulates the behavior of other devices or systems to achieve a desired result. A control system achieves this through control loops, which are a process designed to maintain a process variable at a desired specific form. In other words, the **definition of a control system** can be simplified as a system, which controls other systems.

As human civilization is being modernized day by day the demand for automation has increased alongside it. Automation requires control over systems of interacting devices. In recent years, control systems have played a central role in the development and advancement of modern technology and civilization. Practically every aspect of our day-to-day life is affected by some type of control system. Examples of control systems in your day-to-day life include an air conditioner, a refrigerator, an air conditioner, a bathroom toilet tank, an automatic iron, and many processes within a car – such as cruise control. In industrial settings, we find control systems in the quality control of products, weapons systems, transportation systems, power systems, space technology, robotics, manufacturing, etc.

2.6.1 Main properties

a) Stability: The stability of a control system is defined as the ability of any system to provide a bounded output when a bounded input is applied to it. More specifically, we can say, that stability allows the system to reach the steady-state and remain in that state for that input even after variation in the parameters of the system.

b) Controllability: Controllability is defined as the ability of a control system to reach a definite state from a fixed (initial) state in a finite time. It is considered as an important property of the control system as it defines the behavior of the control system. The theory of controllability was proposed in **1960** by **R. Kalman**. To be able to do what it is needed with the given dynamic system under control input, the system must be controllable.

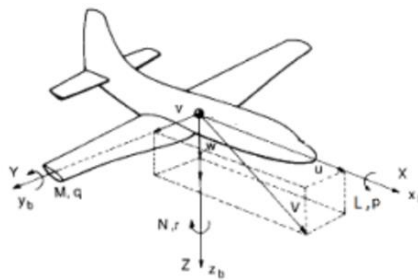


Figure 2.18: Controllability, the ability to control the system.

c) Observability: Observability is a measure of how well internal states of a system can be inferred from knowledge of its external outputs. In control theory, the observability and controllability of a linear system are mathematical properties which behaves dual. To see what is going on inside the system under observation, the system must be observable.

2.6.2 Properties according to the requirements of the designer of the control system

a) Regulation: control regulation has the function of maintaining a designated variable or characteristic of the system equal to a predefined set value. It performs the activity of managing or maintaining a range of values in a machine. The measurable property of a device is managed closely by specified conditions or an advance set value; or it can be a variable according to a predetermined arrangement scheme. It can be used generally to connote any set of various controls or devices for regulating or controlling items or objects.

b) Tracking: Trajectory tracking control is used to effect desired trajectories of a device. To track specified trajectories more precisely, or be able to follow more general trajectories, many tracking control algorithms have been proposed.

c) Disturbance rejection: To address unexpected signals that cause the system to move away from the target value, the controller uses a purpose known as disturbance rejection, which processes the disturbance and provides commands that correct for these unknown conditions.

d) Noise filtering: Noise filtering is a set of processes that is performed to remove the noise contained with the data acquired on construction and infrastructure sites. The data obtained need to be processed, and the contained noise should be eliminated or reduced.

e) Robustness or sensibility: The robustness refers to the ability of a control system to withstand parameter variations in the plant transfer function, and still maintain the stability and performance goals.

f) Optimality: Optimality is a mathematical optimization that deals with finding a control for a dynamical system over a time such that an objective function is optimized. It has numerous applications in science, engineering, and operations research.

References [4- 10] are starting points to delve into the systems and signals addressed in this chapter.

Exercises:

1. Represent by a differential equation the motion of an inertial mass, m , acted by a force, $f(t)$, in the presence of kinetic friction, represented by b , is governed by Newton's second law of motion given in figure 3.6.

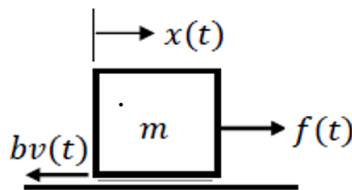


Figure 2.19: Motion of an inertial mass with applied force under surface friction.

2. A parallel RL network is connected across a constant current source, I_s , as shown in figure 3.7. Represent the circuit by a first-order ODE, where the variable of interest is the inductor current, i_L , and Kirchoff's current law is applied at a node.



Figure 2.20: A parallel RL circuit driven by a constant current source.

3. Demonstrate that RLC circuit of the figure results in a second order differential equation of the form:

$$L \frac{d^2 Q}{dt^2} + R \frac{dQ}{dt} + \frac{1}{C} Q = V(t) \text{ with } Q(t) = Q_0 + \int_0^t I(\tau) d\tau$$

Being $V(t)$ the applied voltage to the circuit and $I(t)$ the electrical current which circulates over the circuit.

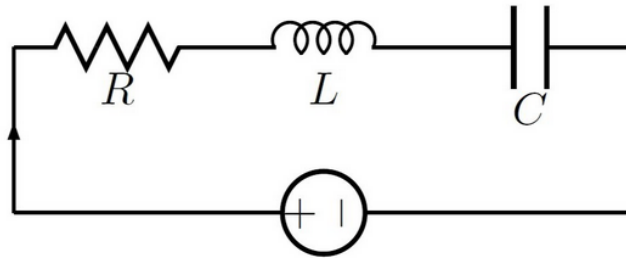


Figure 2.21: A series RLC circuit driven by a voltage source.

References:

- [1] <https://ctms.engin.umich.edu/CTMS/index.php?example=MotorSpeed§ion=SimulinkModeling>
- [2] Rehan, Z. (2020) Application of First-Order Differential Equation to Heat Convection in Fluid. *Journal of Applied Mathematics and Physics*, **8**, 1456-1462. doi: [10.4236/jamp.2020.88111](https://doi.org/10.4236/jamp.2020.88111).
- [3] <https://ctms.engin.umich.edu/CTMS/index.php?example=MotorSpeed§ion=SystemModeling>
- [4] Ziemer, R.E., Tranter, W.H., and Fannin, D.R. (1983). Signals and Systems: Continuous and Discrete.
- [5] Ziemer, R.E. and Tranter, W.H. and Fannin, D.R. (1989), Signals and Systems: Continuous and Discrete. 9780024316301. Macmillan.
- [6] Soliman, Samir S. and Srinath, Mandyam D. (1990). Continuous and Discrete Signals and Systems, 1990, ISBN: 0131712578, Prentice-Hall, Inc.
- [7] Mandal, M. and Asif, A. (2007). Continuous and discrete-time signals and Systems. 9781108477864. Cambridge University Press.
- [8] Oppenheim, Alan V. and Willsky, Alan S. and Nawab, S. Hamid, (1996). Signals and Systems (2nd Ed.), 0138147574, Prentice-Hall, Inc.
- [9] Kuo, Benjamin C. (1975). Automatic control systems. Englewood Cliffs, N.J : Prentice-Hall.
- [10] Ogata, K. (2010) Modern Control Engineering. 5th Edition, Pearson, Upper Saddle River .

Chapter 3: Laplace and inverse Laplace transforms

This chapter addresses the Laplace transform. Laplace transform converts time-domain functions and operations into Laplace domain or transform. It facilitates the algebra behind the equations, providing a graphical solution to time-domain differential equations in a parameterized form that represents physical systems. The inverse Laplace transform performs the transformation between the frequency and time domains of the system, providing a bidirectional relationship. It is important to recall that the Fourier transform is used to study signals and the Laplace transform to study systems.

3.1 Laplace and inverse Laplace transforms

Laplace transforms convert time-domain functions and operations into Laplace domain, $f(t) \rightarrow F(s)$, ($t \in R, s \in C$), where R represents the set of real numbers and C the set of complex numbers. With this conversion, linear differential equations (LDE) are transformed into algebraic expressions in the complex plane, providing a simplified solution for key LDE characteristics.

The transform has many applications in science and engineering because it is a tool for solving differential equations. It transforms linear differential equations into algebraic equations and convolution into multiplication [1, 2]. The Laplace transform presents a linear solution to solve high-order equations in the frequency domain, which Fourier transforms do not.

For suitable functions f , the Laplace transform is the integral:

$$L\{f\}(s) = \int_0^{\infty} f(t)e^{-st} dt \quad (3.1)$$

The mathematics of classical control theory depends on linear ordinary differential equations, which commonly arise in all scientific disciplines. Control theory emphasizes a powerful Laplace transform expression of linear differential equations.

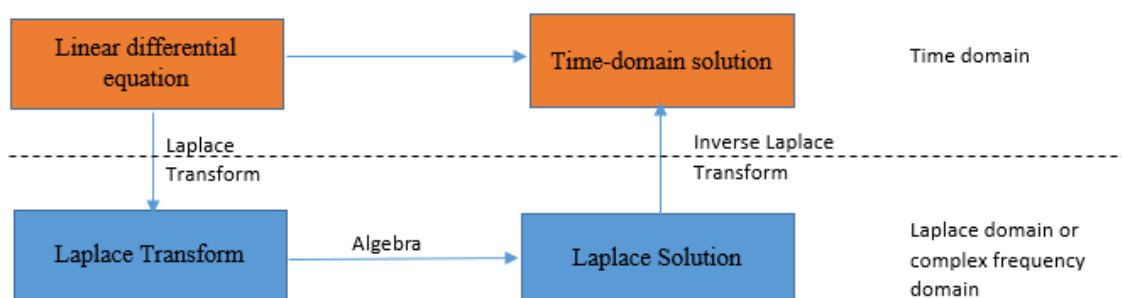


Figure 3.1: Representation of the relationships between time domain solutions and Laplace solutions.

One example of how a function is transformed into the Laplace domain from the definition is the following: for a function $f(t) = e^{-2t}$, it is plugged into the definition of the Laplace transform:

$$F(s) = L\{f\}(s) = \int_0^{\infty} f(t)e^{-st} dt \quad (3.2)$$

$$F(s) = \int_0^{\infty} e^{-st} e^{-2t} dt = \int_0^{\infty} e^{-st-2t} dt = \int_0^{\infty} e^{-(s+2)t} dt$$

Since s is a constant, it is integrated:

$$F(s) = \left. \frac{-1}{(s+2)} e^{-(s+2)t} \right|_0^{\infty}$$

$$F(s) = \lim_{b \rightarrow \infty} \left. \frac{-1}{(s+2)} e^{-(s+2)t} \right|_0^b$$

Evaluating the interval,

$$F(s) = \lim_{b \rightarrow \infty} \frac{-1}{(s+2)} e^{-(s+2)b} + \frac{1}{(s+2)} e^{-(s+2)0} = \frac{-1}{(s+2)} e^{-\infty} + \frac{1}{(s+2)} e^0 = \frac{1}{(s+2)}$$

It is possible to check the solution in Table 3.1 given below.

3.1.1 Properties of Laplace Transforms

The properties of the Laplace transforms provide the tool to get polynomial algebraic equations from differential equations and the inverse transformation. The properties are the following [3]:

- a. **Unicity:** $L[x(t)] \leftrightarrow X(s)$ is unique.
- b. **Linearity:** If $L[x(t)] \leftrightarrow X(s)$ & $L[y(t)] \leftrightarrow Y(s)$, then $L[ax(t) \mp by(t)] \leftrightarrow aX(s) \mp bY(s)$.
- c. **Time shifting:** If $L[x(t)] \leftrightarrow X(s)$, then $L[x(t - t_0)] \leftrightarrow e^{-st_0} X(s)$.
- d. **Frequency shifting:** If $L[x(t)] \leftrightarrow X(s)$, then $L[e^{s_0 t} x(t)] \leftrightarrow X(s - s_0)$.
- e. **Time reversal:** If $L[x(t)] \leftrightarrow X(s)$, then $L[x(-t)] \leftrightarrow X(-s)$.
- f. **Time scaling:** If $L[x(t)] \leftrightarrow X(s)$, then $L[x(at)] \leftrightarrow \frac{1}{|a|} X\left(\frac{s}{a}\right)$.
- g. **Differentiation:** If $L[x(t)] \leftrightarrow X(s)$, then $L\left[\frac{dx(t)}{dt}\right] \leftrightarrow sX(s) - sX(0)$, and then $L\left[\frac{d^n x(t)}{dt^n}\right] \leftrightarrow s^n X(s) - s^{n-1} X(0) - s^{n-2} \left[\frac{dx}{dt}\right]_{t=0} - \dots - s \left[\frac{d^{n-2} x}{dt^{n-2}}\right]_{t=0} - \left[\frac{d^{n-1} x}{dt^{n-1}}\right]_{t=0}$.
- h. **Integration:** If $L[x(t)] \leftrightarrow X(s)$, then $L\left[\int x(t) dt\right] \leftrightarrow \frac{1}{s} X(s)$, and then $L\left[\int \int \int \dots n \dots \int x(t) dt\right] \leftrightarrow \frac{1}{s^n} X(s) - \frac{1}{s^{n-1}} X(0) - \frac{1}{s^{n-2}} \left[\int x(t) dt\right]_{t=0} - \dots - \frac{1}{s} \left[\int \int \int \dots n-2 \dots \int x(t) dt\right]_{t=0} - \left[\int \int \int \dots n-1 \dots \int x(t) dt\right]_{t=0}$

At this point, the definition of convolution is introduced, as it plays a key role in the properties of the Laplace transform. A convolution is an integral that expresses the amount of overlap of one function $g(t)$ as it is shifted over another function $f(t)$. It therefore "blends" one function with another. The convolution is sometimes also known by its German name, *faltung* ("folding").

Abstractly, a convolution is defined as a product of functions $g(t)$ and $f(t)$ that are objects in R^n . The convolution of two functions $g(t)$ and $f(t)$ over a finite range $[0, t]$ is given by

$$[f * g](t) \equiv \int_0^t f(\tau)g(t - \tau)d\tau \quad (3.3)$$

where the symbol $[f * g](t)$ denotes the convolution of $g(t)$ and $f(t)$.

The convolution is more often taken over an infinite range,

$$f * g \equiv \int_{-\infty}^{\infty} f(\tau)g(t - \tau)d\tau = \int_{-\infty}^{\infty} g(\tau)f(t - \tau)d\tau \quad (3.4)$$

with the variable, in this case, t implied.

Now, the rest of the properties are introduced:

- i. **Multiplication:** If $L[x(t)] \leftrightarrow X(s)$ and $L[y(t)] \leftrightarrow Y(s)$, then, if $L[x(t)y(t)] \leftrightarrow \frac{1}{2\pi j} X(s) * Y(s)$, $j = \sqrt{-1} \in \mathbb{C}$.
- j. **Convolution:** Let $f(t)$ and $g(t)$ be causal functions with Laplace transforms $F(s)$ and $G(s)$ respectively, i.e., $L\{f(t)\} = F(s)$ and $L\{g(t)\} = G(s)$. Then it can be shown that

$$L^{-1}\{F(s)G(s)\} = (f * g)(t)$$
 or equivalently $L(f * g)(t) = F(s)G(s)$.
- k. **Initial value theorem:** The initial value theorem of Laplace transform enables us to calculate the initial value of a function $x(t)$ (i.e., $x(0)$) directly from its Laplace transform $X(s)$ without the need for finding the inverse Laplace transform of $X(s)$.

The **initial value theorem** of Laplace transforms states that, if:

$$x(t) \leftrightarrow L^{-1}[X(s)] \quad (3.5)$$

Then,

$$\lim_{t \rightarrow 0} x(t) = x(0) = \lim_{s \rightarrow \infty} sX(s) \quad (3.6)$$

Proof of this theorem may be found in [3] and [4].

- l. **Final value theorem:** Consider a continuous physical function $f(t)$, a continuous derivative $\frac{df}{dt}$, and a Laplace transform $L[f(t)] \leftrightarrow F(s)$. The final-value theorem expresses the final, steady-state value of $f(t)$ in terms of $F(s)$ as:

$$\lim_{t \rightarrow \infty} f(t) = \lim_{s \rightarrow 0} (sF(s)) \quad (3.7).$$

This theorem is useful for finding the final value because it is almost always easier to derive the Laplace transform and evaluate the limit on the right-hand side, than to derive the equation for $f(t)$ and evaluate the limit on the left-hand side. Final-value theorem Equation (3.7) is valid provided that $\lim_{t \rightarrow \infty} f(t)$ exists (i.e., is a finite, constant value). But

Equation (3.7) must be applied with care because the theorem itself fails to distinguish between functions for which the limit exists and functions that have no limit. Indeed, the theorem can predict falsely that an unstable system has a limit when, in fact, there is none, i.e., that $\lim_{t \rightarrow \infty} f(t) \rightarrow \pm\infty$. So, the condition for applying the theorem is that if $f(t)$ is a function on $(0, \infty)$ and $F(s)$ is its Laplace transform, then $\lim_{t \rightarrow \infty} f(t)$ exists and is equal

to $\lim_{s \rightarrow 0} sF(s)$ on the condition that all of the poles of $sF(s)$ *lie in the left half-plane* (i.e., the system is stable).

The derivation of the final-value theorem is based on a definition of the Laplace transform equation and the derivative of the Laplace transform equation:

$$L \left[\frac{df(t)}{dt} \right] = \int_0^{\infty} e^{-st} \left[\frac{df(t)}{dt} \right] dt = sF(s) - f(0) \quad (3.8)$$

Taking the limit of all terms as $s \rightarrow 0$ gives

$$\lim_{s \rightarrow 0} \left(\frac{df(t)}{dt} \right) = \int_0^{\infty} \left(\frac{df(t)}{dt} \right) dt = \lim_{s \rightarrow 0} [sF(s)] - f(0) \quad (3.9)$$

Now the integral is evaluated easily:

$$\int_0^{\infty} \left(\frac{df(t)}{dt} \right) dt = f(\infty) - f(0) = \lim_{s \rightarrow 0} [sF(s)] - f(0) \Rightarrow f(\infty) = \lim_{s \rightarrow 0} [sF(s)] \quad (3.10)$$

This completes the derivation of the final-value theorem.

3.1.2 Table of Laplace Transforms

$f(t)$	$L[f(t)] \leftrightarrow F(s)$
Unitary impulse; $\delta(t)$	1
Unitary step; 1	$\frac{1}{s}, Re(s) > 0$
Unitary ramp; t	$\frac{1}{s^2}, Re(s) > 0$
t^n	$\frac{n!}{s^{(n+1)}}, Re(s) > 0$
e^{at}	$F(s) = \frac{1}{(s-a)}, Re(s) > 0$
$\sin(at)$	$F(s) = \frac{a}{(s^2 + a^2)}, Re(s) > 0$
$\cos(at)$	$F(s) = \frac{s}{(s^2 + a^2)}, Re(s) > 0$
$e^{at} \sin(bt)$	$F(s) = \frac{b}{(s^2 - a^2) + b^2}, Re(s) > a $
$e^{at} \cos(bt)$	$F(s) = \frac{(s-a)}{(s^2 - a^2) + b^2}, Re(s) > a $
$t^n e^{at}$	$F(s) = \frac{n!}{(s-a)^{(n+1)}}, Re(s) > a$
Delay: $y(t-T)$	$e^{-sT} Y(s)$

Table 3.1: Laplace transform table [5].

3.1.3 Laplace inverse transform

Given a function $F(s)$, the *inverse Laplace transform* of F , denoted by $L^{-1}[F]$, is that function f whose Laplace transform is F or, mathematically,

$$f(t) = L^{-1}\{F(s)\} \leftrightarrow L\{f(t)\} = F(s) \quad (3.11)$$

Finding the Laplace transform of a function is not very complicated if a table of transforms is provided. This section deals with going the other way. From a given

transform, $F(s)$, output the original time domain function (or functions) the system has. Please note that this procedure is generally more difficult and lengthier than simply taking transforms. In these cases, it is said that the *Inverse Laplace Transform* of $F(s)$ is calculated, and it is used with the following notation: $f(t) = L^{-1}\{F(s)\}$.

As with Laplace transforms, the following property helps to take the inverse transform. Given the two Laplace transforms $F(s)$ and $G(s)$ then:

$$L^{-1}\{aF(s) + bG(s)\} = aL^{-1}\{F(s)\} + bL^{-1}\{G(s)\} \quad (3.12)$$

for any constants a , and b .

So, the inverse transform of the individual transforms property may be used to put any constants back in and then add or subtract the results back up.

So, one final time, partial fractions are a fact when using Laplace transforms to solve differential equations. When the factors of the denominator are of the first degree, but some are repeated, assume unknown numerators for each factor:

- If a term is present twice, make the fractions the corresponding term and its second power.
- If a term is present three times, make the fractions the term and its second and third powers.

Example 3.1: Find the inverse transform of $G(s) = \frac{6}{s(s^2+9)}$ using partial fractions.

Solution:

$$\begin{aligned} G(s) &= \frac{6}{s(s^2+9)} = \frac{2}{3} \frac{1}{s} - \frac{2}{3} \frac{s}{s^2+9} \text{ and } L^{-1}\left(\frac{6}{s(s^2+9)}\right) = L^{-1}\left(\frac{2}{3} \frac{1}{s}\right) - L^{-1}\left(\frac{2}{3} \frac{s}{s^2+9}\right) = \\ &= \frac{2}{3} L^{-1}\left(\frac{1}{s}\right) - \frac{2}{3} L^{-1}\left(\frac{s}{s^2+9}\right) = \frac{2}{3} u(t) - \frac{2}{3} (\cos 3t - 1)u(t) \end{aligned}$$

Example 3.2: Determine $x(t)$ and then verify the initial value theorem of the function given by, $X(s) = \frac{1}{s+3}$.

Solution:

The given function is,

$$X(s) = \frac{1}{s+3}$$

Taking the inverse Laplace transform of $X(s)$,

$$x(t) = L^{-1}[X(s)] = L^{-1}\left[\frac{1}{s+3}\right] \rightarrow x(t) = e^{-3t}$$

just using the table of Laplace transforms.

Therefore, the initial value of the function is,

$$x(0) = [x(t)]_{t=0} \rightarrow x(0) = [e^{-3t}]_{t=0} = e^0 = 1$$

Again, by the initial value theorem, we obtain,

$$\begin{aligned} \lim_{t \rightarrow 0} x(t) &= x(0) = \lim_{s \rightarrow \infty} s \frac{1}{(s+3)} \\ x(0) &= \lim_{s \rightarrow \infty} s \frac{1}{(s+3)} = 1 \end{aligned}$$

Hence, the initial value theorem is verified for the given function and for the stable transfer function $X(s)$, as its pole lies in the negative real part of the s -plane ($s = -3$), which is the necessary condition to apply the theorem.

3.2 Transfer function

A **transfer function** represents the relationship between the output signal of a control system and the input signal, for all possible input values. A block diagram is a visualization of the control system which uses blocks to represent the transfer function, and arrows which represent the various input and output signals.

Every control system has a reference input, often called excitation or cause, that works through a transfer function to create a controlled output or response.

Therefore, the cause-and-effect relationship between the output and input is related to each other through a **transfer function**.



Figure 3.2: The relationship between the input and the output of a system is defined as transfer function.

In Laplace Transform domain, if the input (reference) is represented by $R(s)$ and the output is represented by $Y(s)$, then the transfer function will be:

$$G(s) = \frac{Y(s)}{R(s)} \Rightarrow Y(s) = G(s) \cdot R(s) \quad (3.13)$$

That is, the transfer function of the system multiplied by the input function gives the output function of the system.

Transfer function is defined as the ratio of the Laplace transform of the output to the Laplace transform of the input, assuming zero initial conditions.

$$G(s) = \frac{Y(s)}{R(s)} \quad (3.14)$$

The procedure for determining the transfer function of a control system is as follows:

1. The equations for the system are given or obtained.
2. Now the Laplace transform of the system equations is taken, assuming the initial conditions are zero.
3. Specify the system output and input.
4. Lastly, the ratio of the Laplace transform of the output is taken to the Laplace transform of the input, which is the required transfer function.

Inputs and outputs in a control system may differ. For instance, electric motors take electrical signals as inputs and produce mechanical outputs to rotate, while generators take mechanical inputs to generate electrical outputs. But for mathematical analysis of a system, all kinds of signals should be represented in a similar form. This is done by transforming all kinds of signals into their Laplace form. Also, the transfer function of a system is represented by the Laplace form by dividing the output the Laplace transfer function by the input Laplace transfer function. Hence, a basic block diagram of a control system can be represented as:



Figure 3.3: Representation of the relationship between the input and output of a system in the Laplace domain.

where, $R(s) = L\{r(t)\}$, $Y(s) = L\{y(t)\}$, and $G(s) = \frac{L\{y(t)\}}{L\{r(t)\}}$ and, $r(t)$ and $y(t)$ are the time domain functions of the input and output signals, respectively.

The transfer function provides a basis for determining important system response characteristics without solving the complete differential equation. In a general form the transfer function is composed of m zeros and n poles being n , m arbitrary number and $m \leq n$. So, the transfer function can be written as:

$$H(s) = \frac{Y(s)}{R(s)} = \frac{b_0s^m + b_1s^{m-1} + \dots + b_m}{a_0s^n + a_1s^{n-1} + \dots + a_n} \quad (3.15)$$

It is often convenient to factor the polynomials in the numerator and denominator, and to write the transfer function in those factors:

$$H(s) = \frac{Y(s)}{R(s)} = \frac{K(s-q_1)(s-q_2)\dots(s-q_n)}{(s-p_1)(s-p_2)\dots(s-p_n)} \quad (3.16)$$

where the numerator and denominator polynomials, $Y(s)$ and $R(s)$ have real coefficients defined by the system's differential equation and $K = b_0/a_0$. As written in Eq. (3.16) the q_i 's are the roots of the equation $Y(s) = 0$ and are defined to be the system zeros, and p_i 's are the roots of the equation $R(s) = 0$ and are defined to be the system poles. All the coefficients of polynomials $Y(s) = 0$ and $R(s) = 0$ are real, therefore the poles and zeros must be either purely real or appear in complex conjugate pairs. *If a pole and zeros are sited in the same place in the s -plane they can be simplified in the transfer function.*

Moreover, *the influence of the zeros and poles is larger if they are close to the origin of the s -plane. If they are placed far away to the origin, they can be neglected in the transfer function as their influence is minimal.* In general, the real part of a pole indicates how quickly the transient portion of the corresponding mode decays to zero (assuming negative real part). Therefore, a transfer function which has one (or more) poles much farther to the left in the complex plane (more negative) than the other poles, their effect on the dynamic response will be hidden by the slower, more dominant poles. This is called as **Model Reduction**.

The transfer function of a system can be converted to the steady state equations and vice versa. In MATLAB, the functions `ss2tf(A,B,C,D)` and `tf2ss(num, den)` allows these transformations:

```
>> % First define state space system
>> A=[0 1 0; 0 0 1; -3 -4 -2];
>> B=[0; 0; 1];
>> C=[5 1 0];
>> [n,d]=ss2tf(A,B,C,D)
```



```

              n =
              0      0      1.0000      5.0000
              d =
1.0000      2.0000      4.0000      3.0000

>> mySys_tf=tf(n,d)
Transfer function:
          s + 5
-----

```

The reader can write an script to check the functionality ss2tf.

There are two major ways of obtaining a transfer function for the control system. The ways are:

- **Block Diagram Method:** It is not convenient to derive a complete transfer function for a complex control system. Therefore, the transfer function of each element of a control system is represented by a block diagram. Block diagram reduction techniques are applied to obtain the desired transfer function.
- **Signal Flow Graphs:** The modified form of a block diagram is a signal flow graph. Block diagrams visually outline a control system, while signal flow graphs provide a more compact representation.

3.3 Block diagram

A block diagram is a diagram of a system in which the principal parts or functions are represented by blocks connected by lines that show the relationships between the blocks. They are heavily used in engineering in hardware design, electronic design, software design, and process flow diagrams.

Block diagrams are typically used for higher-level, less detailed descriptions that are intended to clarify overall concepts without concern for the details of implementation. Contrast this with the schematic diagrams and layout diagrams used in electrical engineering, which show the implementation details of electrical components and physical construction.



Figure 3.4: Block diagram representation

Block diagrams represent the frequency domain, or Laplace transform, of the dynamic system.

Systems may be further classified according to the input signals. If they are continuous, discrete, fuzzy, etc., the system is then called a continuous system, a discrete system, or a fuzzy system. Moreover, systems may also be classified in relation to the connected signals to them. How the signals a system acquires provide another type of system, namely, continuous or discrete, deterministic or stochastic, mono-variable or multi-variable. Finally, the intrinsic behavior corresponding to a mathematical operator

characterizes the systems as static or dynamic, linear or non-linear, time-independent or time-dependent, and concentrated or distributed systems.

There are three main types of block diagrams: series decomposition, parallel decomposition, loop arrangement, and any combination of them. They are defined as:

a) Series decomposition: the series decomposition diagram is the block diagram resulting from two or more block diagrams addressed in series, as shown in Figure 3.5.

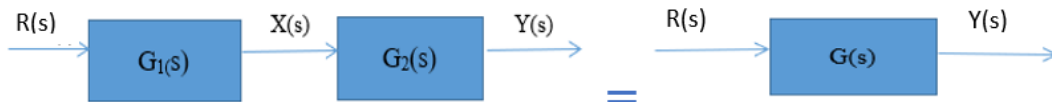


Figure 3.5: Series decomposition diagram.

The resulting block diagram may be calculated as:

$$\begin{cases} G_2(s) = \frac{Y(s)}{X(s)} \\ G_1(s) = \frac{X(s)}{R(s)} \end{cases} \quad G(s) = G_2(s)G_1(s) = \frac{Y(s)}{R(s)} \quad (3.15)$$

being $G_1(s)$, $G_2(s)$, $G(s)$ transfer functions.

b) Parallel decomposition: a parallel decomposition diagram is the block diagram resulting from two or more block diagrams addressed in parallel, as shown in Figure 3.6.

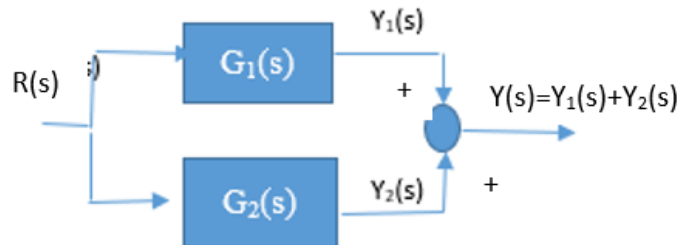


Figure 3.6: Parallel decomposition diagram.

The resulting block diagram may be calculated as:

$$\begin{cases} G_2(s) = \frac{Y_2(s)}{R(s)} \\ G_1(s) = \frac{Y_1(s)}{R(s)} \end{cases} \quad G(s) = G_2(s) + G_1(s) = \frac{Y(s)}{R(s)} \quad (3.16)$$

being $G_1(s)$, $G_2(s)$, $G(s)$ transfer functions.

c) Loop arrangement: a loop arrangement or feedback loop diagram is the block diagram resulting from two or more block diagrams addressed in a closed loop or feedback loop, as shown in Figure 3.7.

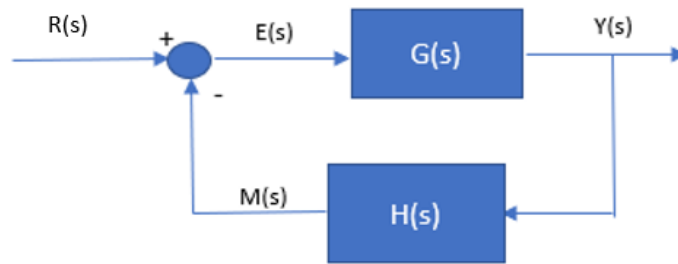


Figure 3.7: Loop arrangement diagram.

The resulting block diagram may be calculated as:

$$\left. \begin{array}{l} Y(s) = G(s)E(s) \\ M(s) = H(s)Y(s) \\ E(s) = R(s) - M(s) \end{array} \right\} \frac{Y(s)}{R(s)} = \frac{G(s)}{1+G(s)H(s)} \quad (3.17)$$

Being $H(s)$, $G(s)$ transfer functions of the sensor and the system.

Any combination of the previous ones may be considered typical block diagrams in control theory.

References [1–7] are starting points to delve into the systems and signals addressed in this chapter. For signal flow graphs, reference [7] presents a comprehensive introduction to them.

Exercises:

1. Find the inverse Laplace transform using the table 3.1 of the each of the following Laplace functions.

- a. $Y_1(s) = L[y_1(t)] = \frac{2}{3+5s}$
- b. $Y_2(s) = L[y_2(t)] = \frac{5s}{s^2+9}$
- c. $Y_3(s) = L[y_3(t)] = \frac{3s+2}{s^2+25}$
- d. $Y_4(s) = L[y_4(t)] = \frac{5}{(s+2)^3}$
- e. $Y_5(s) = L[y_5(t)] = \frac{s+3}{s^2+3s+2}$
- f. $Y_6(s) = L[y_6(t)] = \frac{s^2+s+1}{s^3+s}$

2. Given the block diagram in Figure 3.8:

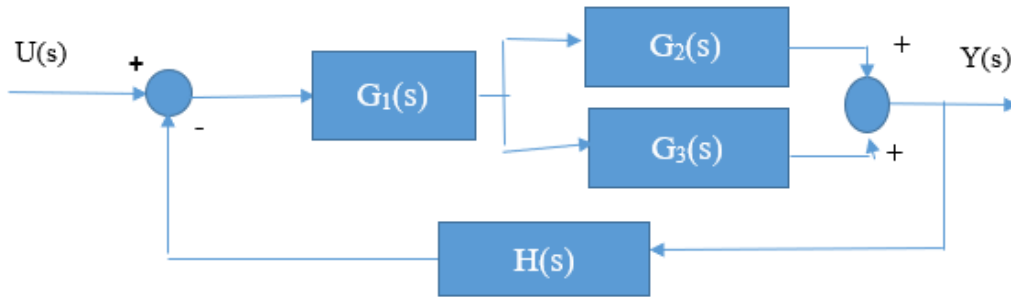


Figure 3.8: Loop arrangement diagram.

What is the equivalent relation between the output and input?

3. Calculate the equivalent relation between the input and the output of the following block diagrams (Figures 3.9 and 3.10). In the diagrams, $d=d(s)$ is the disturbance to the system. To deal with the problem, first, suppose $d(s)=0$ and calculate $Y(s)$ as a function of $R(s)$. Then, suppose $R(s)=0$ and calculate $Y(s)$ as a function of $d(s)$. Sum both and that is the result.

a.

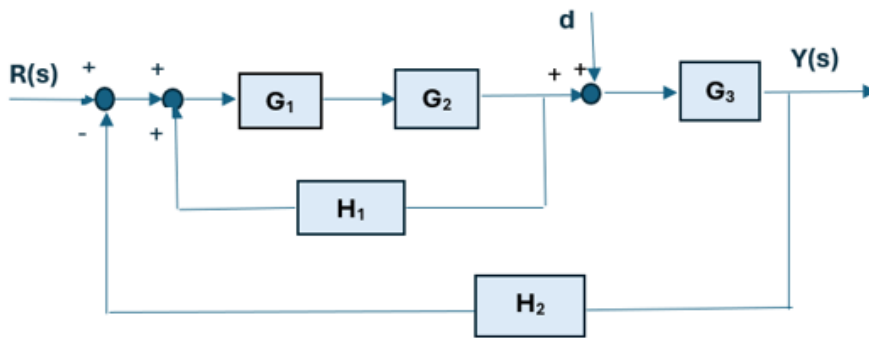


Figure 3.9: Loop arrangement diagram.

b.

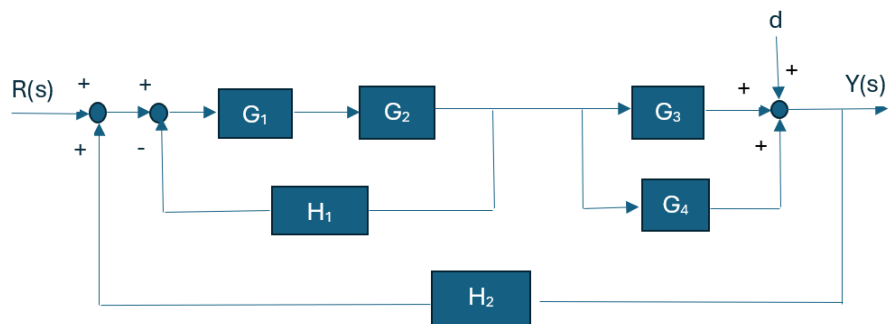


Figure 3.10: Loop arrangement diagram.

4. Demonstrate that the block diagram in Figure 3.11 has an output of the form:

$$y = \frac{FGK}{1 + GKH} r + \frac{G_d G}{1 + GKH} d_e + \frac{1}{1 + GKH} n$$

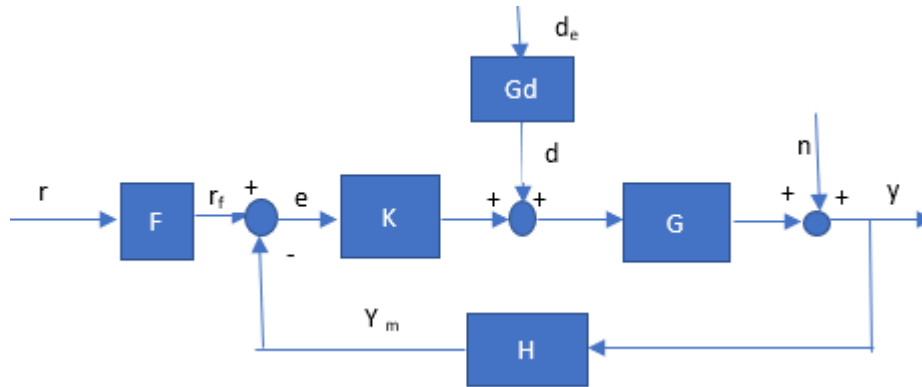


Figure 3.11: System and feedback control under disturbances (d_e), noise (n) signals, and filtered reference signal.

Analyze and explain the block diagram.

Hint: suppose first $d_e = n = 0$ and calculate the relationship between the reference and the output, second: $r = n = 0$, and calculate the relation between the disturbance and the output, and third: $r = d_e = 0$ and calculate the relation between the noise and the output. Finally, sum all three.

References:

- [1] Devi, Rekha. "Applications of Laplace Transformation." *Research Journal of Science and Technology* 9, no. 1 (2017): 167. <http://dx.doi.org/10.5958/2349-2988.2017.00027.4>
- [2] Pérez-Esteve, Salvador. "Convolution operators for the one-sided Laplace transformation." *Časopis pro pěstování matematiky* 110, no. 1 (1985): 69–76. <http://dx.doi.org/10.21136/cpm.1985.118223>.
- [3] Sayood, K. (2021). *Properties of Laplace Transforms*. In: *Signals and Systems. Synthesis Lectures on Signal Processing*. Springer, Cham. https://doi.org/10.1007/978-3-031-02545-7_23
- [4] P.P.G. Dyke (2014). *An introduction to Laplace transforms and Fourier series*. Springer undergraduate mathematics series, 2nd Edition.
- [5] Fleisch, D. (2022). *A Student's Guide to Laplace transforms (Student's Guides)*. Cambridge: Cambridge University Press. doi:10.1017/9781009089531.
- [6] Beerends, R., Ter Morsche, H., Van den Berg, J., & Van de Vrie, E. (2003). *Fourier and Laplace Transforms*. Cambridge: Cambridge University Press. doi:10.1017/CBO9780511806834.
- [7] Farid Golnaraghi and Benjamin C. Kuo. (2009). *Automatic Control Systems*, 9th Edition. Wiley. doi:978-0470048962.

Chapter 4: Transient Response of First- and Second-order Systems

This chapter addresses the transient response of first- and second-order systems. They are analyzed in the Laplace domain, but the properties and characteristics are visualized in the time domain. The most significant characteristics of these systems are presented with mathematical contextualization.

4.1 Transient response of first-order transfer function

In this section, the responses of first order systems against unity step input and impulse are introduced. The output of the system is analyzed to determine the typical response parameters of these systems. The idea behind this is to set standard parameters that dictate the behavior of first-order systems. This idea will help to design and tune the parameters of the controller associated with a first-order system to enhance the system's response and performance.

Consider a first-order transfer function (strictly proper or stable, i.e., the pole of the transfer function lies on the left-hand side of the s-plane):

$$G(s) = \frac{Y(s)}{R(s)} = \frac{b_0}{s+a_0} \quad (4.1)$$

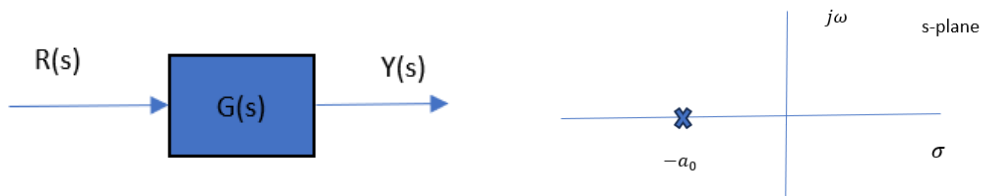


Figure 4.1: Block diagram representation and pole placement in the s-plane of a first-order system.

It is common to write $G(s)$ as:

$$G(s) = \frac{K}{\tau s + 1} = \frac{b_0}{s + a_0} \quad (4.2)$$

$$\text{i.e., } a_0 = \frac{1}{\tau}; \quad b_0 = \frac{K}{\tau}.$$

Example 4.1: $G(s) = \frac{3}{s+2} = \frac{1.5}{0.5s+1}$; $a_0 = 2$; $b_0 = 3$; $K = 1.5$; $\tau = 0.5$.

The pole of the system is at $s = -a_0$ or $s = -\frac{1}{\tau}$. τ is called the *time constant*. K is called the DC-gain, or steady state gain.

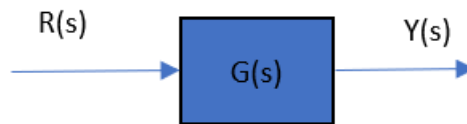


Figure 4.2: Block diagram representation of a general system.

What is the differential equation corresponding to the input-output system. The following equation:

$$Y(s) = \frac{K}{\tau s + 1} R(s) \quad (4.3)$$

becomes

$$(s + 1/\tau)Y(s) = K/\tau R(s) \quad (4.4)$$

which is equivalent to

$$\dot{y}(t) + 1/\tau y(t) = K/\tau r(t) \quad (4.5)$$

If it is considered the effect of both the input $r(t)$ and the initial condition $y(0)$. Taking the Laplace transform of the differential equation, this time including the initial condition, yields:

$$sY(s) - y(0) + 1/\tau Y(s) = K/\tau R(s) \quad (4.6)$$

or

$$Y(s) = \frac{y(0)}{s + 1/\tau} + \frac{K/\tau}{s + 1/\tau} R(s) \quad (4.7)$$

Note that the initial condition could be represented in the differential equation by an input $y(0)\delta(t)$ where δ is the unit impulse function, as:

$$\dot{y}(t) + \frac{1}{\tau}y(t) = \frac{K}{\tau}r(t) + y(0)\delta(t) \quad (4.8)$$

In the block diagram representation, it is shown as

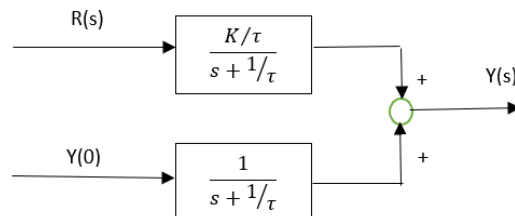


Figure 4.3: Block diagram representation of a first-order system considering initial conditions.

By superposition, the response of the system is the sum of the response due to the initial condition alone (the free response) and the response due to the input $R(s)$ (the forced response).

4.1.1 Step response of a first-order system

If $R(s) = 1/s$, i.e., $r(t)$ is the unit step function, then the forced response (step response) is given (supposing initial conditions equal to zero) as

$$Y(s) = \frac{K/\tau}{s+1/\tau} \frac{1}{s} = \frac{K}{s} - \frac{K}{s+1/\tau} \quad (4.9)$$

In the time domain,

$$y(t) = K(1 - e^{-t/\tau})r(t) \quad (4.10)$$

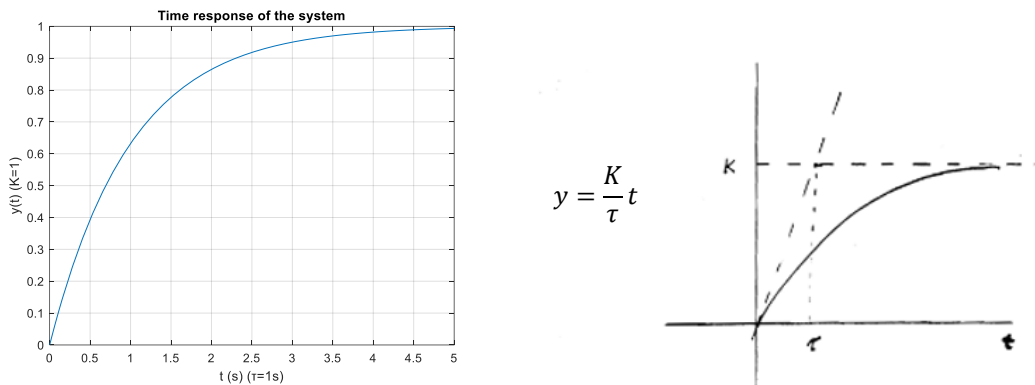


Figure 4.4: Output response of a first order system and slope of the response against step input.

If the response continues to increase at its initial rate, it will reach its steady state value K after τ – seconds.

It is said that the forced response is composed of two terms: 1) $-Ke^{-t/\tau}$ called the transient response and 2) K called the steady state response.

Then, the slope of $y(t)$ at $t = 0$ is

$$\left. \frac{d}{dt} y(t) \right|_{t=0} = \left. \frac{K}{\tau} e^{-t/\tau} \right|_{t=0} = \frac{K}{\tau} \quad (4.11)$$

That means that the transient response will decay to zero after τ – seconds. In practice it is said that the system reaches about 63% ($1 - e^{-1} = 0.37$) after one time constant and has reached the steady state after four times constants.

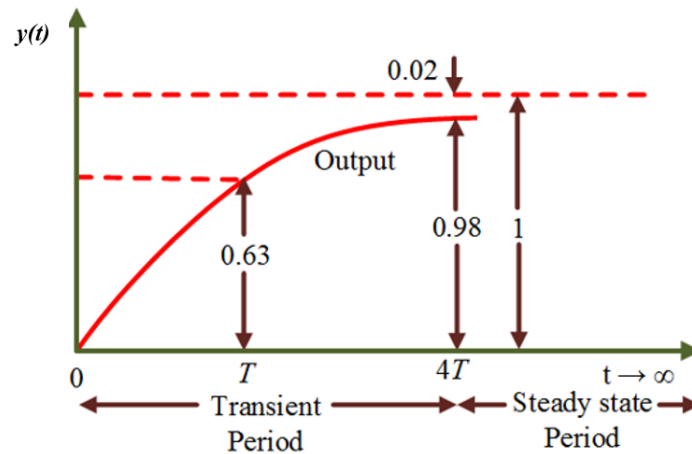


Figure 4.5: Unit step input time response of a first order control system against step input.

Example 4.2: $G(s) = \frac{5}{s+2} = \frac{2.5}{0.5s+1}$; the time constant $\tau = 0.5$ and the steady state value for a unit step input $K = 2.5$.

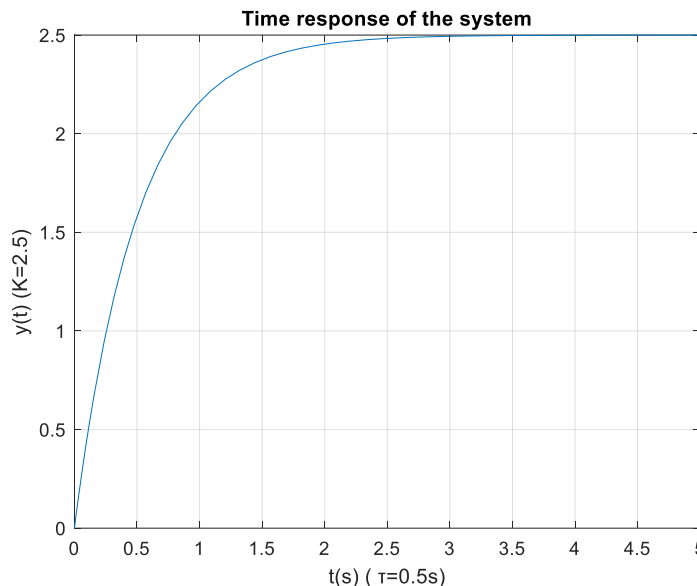


Figure 4.6: Unit time step response of the system given in Example 4.2.

The classification of system responses into forced and free responses, and transient and steady-state responses is not limited to first-order systems but can be applied to transfer functions $G(s)$ of any order.

The DC-gain of any transfer function is defined as $G(0)$ and the steady state value of the system to a unit step input, provided that the system has a steady state value. This follows from the final value theorem:

$$\lim_{t \rightarrow \infty} y(t) = \lim_{s \rightarrow 0} sY(s) = \lim_{s \rightarrow 0} sG(s)R(s) = G(0) \text{ if } R(s) = \frac{1}{s} \quad (4.12)$$

provided that $sG(s)$ has no poles on the right-hand side.

As a result, the following characteristics of first-order systems are defined:

a) Time Constant of a First-Order Control System

The time constant can be defined as the time it takes for the step response to rise to 63%, or 0.63 of its final value. It is referred to as $t = 1/a$. Please note that the reciprocal of the time constant unit is 1/seconds, or frequency.

The parameter “a” is called the exponential frequency. Because the derivative of e^{-at} is $-a$ at $t = 0$. So, the time constant is considered a transient response specification for a **first-order control system**.

It is possible to control the speed of response by setting the poles. Because the further the pole from the imaginary axis, the faster the transient response is. So, it is feasible to set poles distant from the imaginary axis to speed up the whole process.

b) Rise Time of a First Order Control System

The rise time is defined as the time for the waveform to go from 0.1 to 0.9, or 10% to 90% of its final value. For the equation of rising time, we put 0.1 and 0.9 in the general first-order system equation, respectively.

$$\text{For } t = 0.1, y(t) = 1 - e^{-a(0.1)}, T_r = \frac{0.11}{a}.$$

$$\text{For } t = 0.9, y(t) = 1 - e^{-a(0.9)}, T_r = \frac{2.31}{a}.$$

Taking the difference: $T_r = \frac{2.31}{a} - \frac{0.11}{a} = \frac{2.2}{a}$ (4.13). Here is the equation for rising time. If the value of the parameter a is known, it is feasible to compute the step response of a given system. This shows that the qualitative approach works well with first order systems.

c) Settling Time of a First Order Control System

The settling time is defined as the time for the response to reach and stay within 2% of its final value. Normally, the percentage is limited to 5% of its final value. Both percentages are generally considered. The equation of settling time is given by $T_s = 4/a$ (4.14).

By using these three transient response specifications, it is easily computed the step response of a given system. That's why this qualitative technique is useful for order systems equations.

As previously seen, a standard form of first order transfer functions is defined as:

$$G(s) = \frac{Y(s)}{R(s)} = \frac{K}{\tau s + 1}$$

The important characteristics of the standard form are as follows:

- a. The denominator must be of the form $\tau s + 1$.

- b. The coefficient of the s term in the denominator is the system time constant τ .
- c. The numerator is the steady-state gain K .

4.1.2 Impulse response of a first-order system

Recall that an impulse is a large force applied over a very short period. In practice, an example of an impulse would be a hammer striking a surface. Mathematically, a unit impulse is referred to as a Dirac delta function, denoted by $\delta(t)$. It is called a unit impulse because its area is 1. As shown in figure 4.7, the force is applied over the time from 0 to t_1 . Therefore, as t_1 approaches zero, for the area to remain equal to 1 the height must approach infinity.

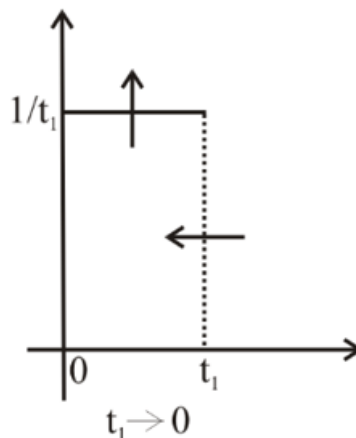


Figure 4.7: Dirac delta function.

Consider the **unit impulse signal** as an input to the first order system, $r(t) = \delta(t)$.

Apply the Laplace transform on both sides, $R(s) = 1$.

Consider the equation of a first-order system, $Y(s) = (\tau s + 1)R(s)$, please note the DC-gain $K=1$ in this case, and substitute, $R(s) = 1$ in the equation:

$$Y(s) = \frac{1}{(\tau s + 1)}(1) = \frac{1}{(\tau s + 1)} \quad (4.15)$$

Rearrange the above equation in one of the standard forms of Laplace transforms,

$$Y(s) = \frac{1}{\tau(s + 1/\tau)} = \frac{1}{\tau} \frac{1}{(s + 1/\tau)} \quad (4.16)$$

Apply the inverse Laplace transform on both sides,

$$y(t) = \frac{1}{\tau} e^{-\frac{t}{\tau}} u(t) \quad (4.17)$$

The unit impulse response is shown in the following figure.

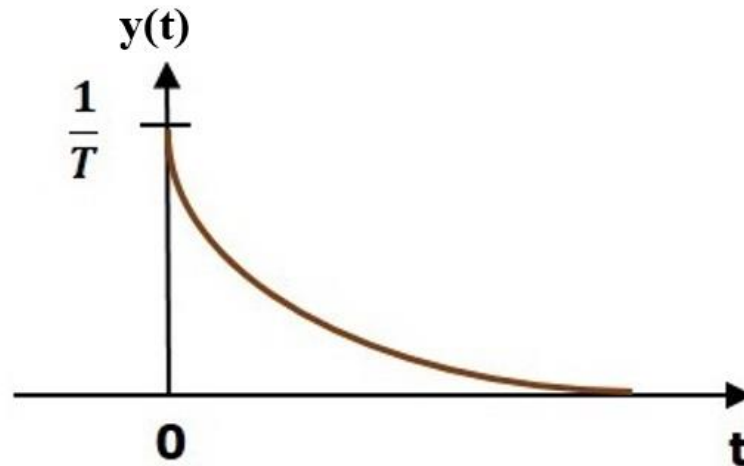


Figure 4.8: Unit impulse response of a first order system.

The **unit impulse response**, $y(t)$ is an exponential decaying signal for positive values of 't', and it is zero for negative values of 't'.

4.2 Transient response of second-order transfer function

The following differential equation is used here to investigate the transient response of second order differential equations using Laplace transform:

$$m\ddot{y}(t) + b\dot{y}(t) + ky(t) = r(t) \quad (4.18)$$

which it's Laplace transform for $\dot{y}(0) = y(0) = 0$ $Y(s) = \frac{R(s)}{ms^2 + bs + k}$

The general expression of the transfer function of the second order control system can be expressed in canonical form as; $\frac{Y(s)}{R(s)} = \frac{K\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2}$ (4.19), here ζ, ω_n^2 are the damping ratio and the natural frequency of the system, respectively. Rearranging the formula; $Y(s) = \frac{K\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} R(s)$. Using this as a base, the time response of a second order control system is analyzed by examining the unit step response of a second-order control system in the frequency domain, before converting it into the time domain.

4.2.1 Step response of a second-order system

Consider the general form of a second order transfer function

$$G(s) = \frac{Y(s)}{R(s)} = \frac{b_0}{s^2 + a_1 s + a_0} \quad (4.20)$$

The standard form of this transfer function is

$$G(s) = \frac{Y(s)}{R(s)} = \frac{K\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} \quad (4.21)$$

ω_n is called the natural frequency (or undamped natural frequency), ζ is called the damping ratio. The roots of the characteristic equation $s^2 + 2\zeta\omega_n s + \omega_n^2 = 0$ give the poles of the system.

$$s^2 + 2\zeta\omega_n s + \omega_n^2 = 0$$

$$s^2 + 2\zeta\omega_n s + \zeta^2\omega_n^2 - \zeta^2\omega_n^2 + \omega_n^2 = 0$$

$$(s + \zeta\omega_n)^2 - \zeta^2\omega_n^2 + \omega_n^2 = 0$$

$$(s + \zeta\omega_n)^2 + (\omega_n\sqrt{1 - \zeta^2})^2 = 0$$

$$s_1 = -\zeta\omega_n + j\omega_n\sqrt{1 - \zeta^2} \text{ and } s_2 = -\zeta\omega_n - j\omega_n\sqrt{1 - \zeta^2}$$

These two roots of the equation, or these two values of s , represent the poles of the transfer function of that system. The real part of the roots represents the **damping**, and the imaginary part represents the **damped frequency** of the response. A general analysis of the different cases is given as follows (see Figure 4.9):

Case i: The two roots are imaginary when $\zeta = 0$. *Undamped system.*

Case ii: The two roots are real and equal when $\zeta = 1$. *Critically damped system ($\zeta = 1$).*

Case iii: The two roots are real but not equal when $\zeta > 1$. *Overdamped system ($\zeta > 1$).*

Case iv: The two roots are complex conjugate when $0 < \zeta < 1$. *Under damped system.*

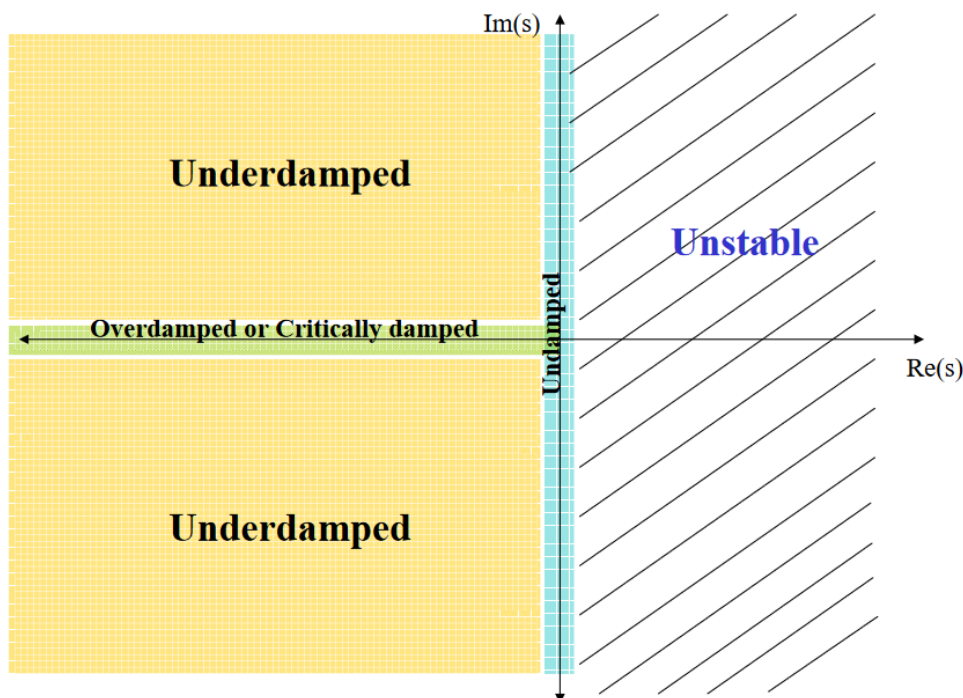


Figure 4.9: Pole placement for undamped, overdamped, or critically damped and underdamped systems.

The location of the roots of the characteristics equation for various values of ζ and ω_n is shown in Figure 4.10 and the corresponding time response for a second order control system is shown in the Figure 4.11 below.

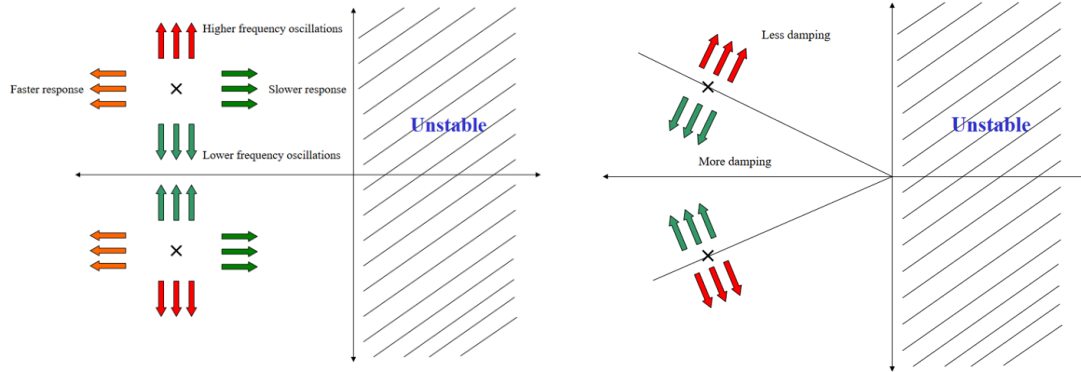


Figure 4.10: Characteristics of the second order underdamped and undamped systems in function of the natural frequency ω_n (left) and damping ratio ζ (right).

Follow these steps to get the response (output) of the second order system in the time domain.

1. Take the Laplace transform of the input signal, $R(s)$.
2. Consider the equation $Y(s) = \frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} R(s)$.
3. Substitute the $R(s)$ value in the above equation.
4. Do partial fractions of $Y(s)$ if required.
5. Apply the inverse Laplace transform to $Y(s)$.

If a unit step function is considered as the input of the system ($r(t) = 1$ or $R(s) = \frac{1}{s}$), the system is studied for the four cases that appeared. These are:

a) Case i: $\zeta = 0$, $Y(s) = \frac{\omega_n^2}{s^2 + \omega_n^2} R(s)$, $R(s) = \frac{1}{s}$, so, $Y(s) = \frac{1}{s} \frac{\omega_n^2}{s^2 + \omega_n^2}$, applying inverse Laplace transform on both sides: $y(t) = 1 - \cos(\omega_n t)$. So, the unit step response of the second-order system when $\zeta = 0$, will be a continuous time signal with constant amplitude and frequency. As in this expression, $y(t) = 1 - \cos(\omega_n t)$ (4.22) there is no exponential term, and the time response of the control system is *un-damped* for the unit step input function with a zero-damping ratio.

b) Case ii: $\zeta = 1$, $Y(s) = \frac{\omega_n^2}{s^2 + 2\omega_n s + \omega_n^2} R(s)$, $R(s) = \frac{1}{s}$, $Y(s) = \frac{1}{s} \frac{\omega_n^2}{s^2 + 2\omega_n s + \omega_n^2} = \frac{1}{s} \frac{\omega_n^2}{(s + \omega_n)^2}$, making partial fractions, $Y(s) = \frac{1}{s} \frac{\omega_n^2}{(s + \omega_n)^2} = \frac{A}{s} + \frac{B}{s + \omega_n} + \frac{C}{(s + \omega_n)^2}$, and solving $Y(s) = \frac{1}{s} - \frac{1}{s + \omega_n} - \frac{\omega_n}{(s + \omega_n)^2}$, using the inverse Laplace transform from the table, $y(t) = 1 - e^{-\omega_n t}(1 + \omega_n t)$ (4.23). So, the unit step response of the second order system will try to reach the step input in a steady state. $y(t) = 1 - e^{-\omega_n t}(1 + \omega_n t)$. In this expression of the output signal, there is no oscillating part in the subjective unit step function. And hence, at this time, the response of the second-order control system is referred to as *critically damped*.

c) Case iii: $0 < \zeta < 1$, the characteristic equation is modified as follows:

$$s^2 + 2\zeta\omega_n s + \omega_n^2 = 0; s^2 + 2\zeta\omega_n s + \zeta^2\omega_n^2 - \zeta^2\omega_n^2 + \omega_n^2 = 0; (s + \zeta\omega_n)^2 - \zeta^2\omega_n^2 + \omega_n^2 = 0;$$

$(s + \zeta\omega_n)^2 + (\omega_n\sqrt{1-\zeta^2})^2 = 0$ or $(s + \zeta\omega_n)^2 + \omega_n^2(1-\zeta^2) = 0$. So, the transfer function may be represented as: $Y(s) = \frac{\omega_n^2}{(s+\zeta\omega_n)^2 + \omega_n^2(1-\zeta^2)} R(s)$, with $R(s) = \frac{1}{s}$. Making partial fractions:

$$Y(s) = \frac{1}{s} \frac{\omega_n^2}{(s+\zeta\omega_n)^2 + \omega_n^2(1-\zeta^2)} = \frac{A}{s} + \frac{Bs+C}{(s+\zeta\omega_n)^2 + \omega_n^2(1-\zeta^2)}$$

$$\text{So, } Y(s) = \frac{1}{s} - \frac{s + 2\zeta\omega_n}{(s + \zeta\omega_n)^2 + \omega_n^2(1-\zeta^2)}$$

$$= \frac{1}{s} - \frac{s + \zeta\omega_n}{(s + \zeta\omega_n)^2 + \omega_n^2(1-\zeta^2)} - \frac{\zeta\omega_n}{(s + \zeta\omega_n)^2 + \omega_n^2(1-\zeta^2)}$$

$$= \frac{1}{s} - \frac{s + \zeta\omega_n}{(s + \zeta\omega_n)^2 + (\omega_n\sqrt{1-\zeta^2})^2} - \frac{\zeta}{\sqrt{1-\zeta^2}} \frac{\omega_n\sqrt{1-\zeta^2}}{(s + \zeta\omega_n)^2 + (\omega_n\sqrt{1-\zeta^2})^2}$$

And $\omega_d = \omega_n\sqrt{1-\zeta^2}$. Applying the Laplace inverse transform,

$$y(t) = 1 - e^{-\zeta\omega_n t} \cos(\omega_d t) - \frac{\zeta}{\sqrt{1-\zeta^2}} e^{-\zeta\omega_n t} \sin(\omega_d t) =$$

$$= \left(1 - \frac{e^{-\zeta\omega_n t}}{\sqrt{1-\zeta^2}} (\sqrt{1-\zeta^2} \cos(\omega_d t) + \zeta \sin(\omega_d t))\right) \quad (4.24).$$

If $\sqrt{1-\zeta^2} = \sin\theta$, then $\zeta = \cos\theta$, substitute these values in the previous equation:

$$y(t) = \left(1 - \frac{e^{-\zeta\omega_n t}}{\sqrt{1-\zeta^2}} (\sin\theta \cos(\omega_d t) + \cos\theta \sin(\omega_d t))\right) = 1 - \frac{e^{-\zeta\omega_n t}}{\sqrt{1-\zeta^2}} \sin(\omega_d t + \theta) \text{ with } \omega_d = \omega_n\sqrt{1-\zeta^2} \quad (4.25).$$

So, the unit-step response of the second-order system has damped oscillations (decreasing amplitude) when ζ lies between zero and one.

The error signal is given by $e(t) = y(t) - r(t)$, and hence,

$$e(t) = \frac{e^{-\zeta\omega_n t}}{\sqrt{1-\zeta^2}} \sin(\omega_d t + \phi) \quad (4.26)$$

From the above expression, the error of the signal is of the oscillation type with exponentially decaying magnitude when $\zeta < 1$.

The frequency of the oscillation is ω_d , and the time constant of exponential decay is $1/\zeta\omega_n$. Where ω_d is referred to as the damped frequency of the oscillation and ω_n is the natural frequency of the oscillation. The term ζ affects that damping significantly, and hence this term is called damping ratio. The systems that behave in this manner are known as *underdamped* systems.

d) Case iv: $\zeta > 1$, we modify the characteristic equation as follows:

$$s^2 + 2\zeta\omega_n s + \omega_n^2 = 0; s^2 + 2\zeta\omega_n s + \zeta^2\omega_n^2 - \zeta^2\omega_n^2 + \omega_n^2 = 0; (s + \zeta\omega_n)^2 - \zeta^2\omega_n^2 + \omega_n^2 = 0; \text{ as } \zeta > 1,$$

$$(s + \zeta\omega_n)^2 - (\omega_n\sqrt{1-\zeta^2})^2 = 0 \text{ or } (s + \zeta\omega_n)^2 - \omega_n^2(1-\zeta^2) = 0.$$

So, the transfer function may be represented as: $Y(s) = \frac{\omega_n^2}{(s+\zeta\omega_n)^2 - \omega_n^2(1-\zeta^2)} R(s)$, with $R(s) = \frac{1}{s}$. Making partial fractions: $Y(s) = \frac{1}{s} \frac{\omega_n^2}{(s+\zeta\omega_n)^2 - \omega_n^2(1-\zeta^2)} = \frac{1}{s} \frac{\omega_n^2}{(s+\zeta\omega_n + \omega_n\sqrt{1-\zeta^2})(s+\zeta\omega_n - \omega_n\sqrt{1-\zeta^2})} =$

$$\frac{A}{s} + \frac{B}{(s+\zeta\omega_n+\omega_n\sqrt{1-\zeta^2})} + \frac{C}{(s+\zeta\omega_n-\omega_n\sqrt{1-\zeta^2})}. \text{ Solving the equation, we may get; } A = 1 ; B = \frac{1}{2\sqrt{\zeta^2-1}(\zeta-\sqrt{\zeta^2-1})}; C = \frac{-1}{2\sqrt{\zeta^2-1}(\zeta+\sqrt{\zeta^2-1})}$$

The output of the system $y(t)$ against step input is:

$$y(t) = 1 - \frac{e^{-(\zeta-\sqrt{\zeta^2-1})\omega_n t}}{2\sqrt{\zeta^2-1}(\zeta-\sqrt{\zeta^2-1})} + \frac{e^{-(\zeta+\sqrt{\zeta^2-1})\omega_n t}}{2\sqrt{\zeta^2-1}(\zeta+\sqrt{\zeta^2-1})} \quad (4.27)$$

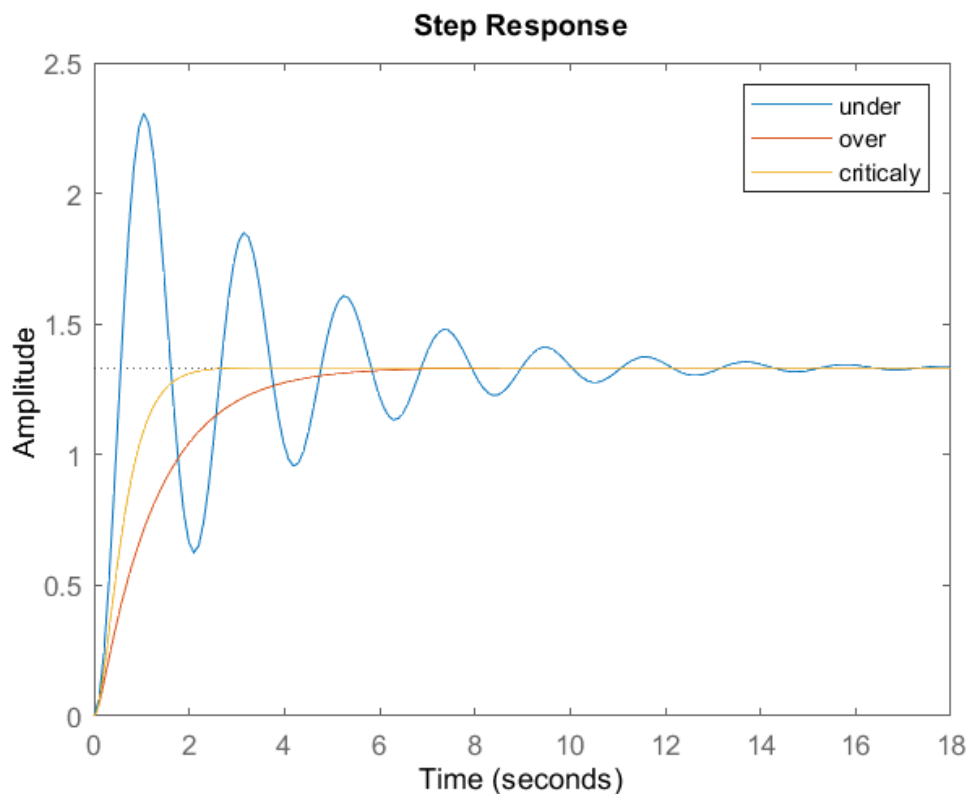


Figure 4.11: Critical, under, and overdamped time domain response of second-order system.

Since it is overdamped, the unit step response of the second-order system when $\zeta > 1$ will never reach step input in the steady state.

In the above expression, there are two-time constants.

$$T_1 = \frac{1}{(\zeta - \sqrt{\zeta^2 - 1})\omega_n}$$

and

$$T_2 = \frac{1}{(\zeta + \sqrt{\zeta^2 - 1})\omega_n}$$

For the value of ζ comparatively much greater than one, the effect of a faster time constant on the time response can be neglected, and the time response expression finally comes as:

$$y(t) = 1 - \frac{e^{-(\zeta - \sqrt{\zeta^2 - 1})\omega_n t}}{2\sqrt{\zeta^2 - 1}(\zeta - \sqrt{\zeta^2 - 1})} \quad (4.28)$$

The time constant of the response is:

$$T = \frac{1}{(\zeta - \sqrt{\zeta^2 - 1})\omega_n}$$

The time response expression of a second order control system subject to a unit step input function is given below.

$$y(t) = 1 - \frac{e^{-\zeta\omega_n t}}{\sqrt{1-\zeta^2}} \sin\left\{(\omega_n\sqrt{1-\zeta^2})t + \tan^{-1}\left(\frac{\sqrt{1-\zeta^2}}{\zeta}\right)\right\} \quad (4.29)$$

The term $\zeta\omega_n$ is responsible for damping of the output response. It is already examined that when the value of ζ (also known as damping ratio) is less than unity, the oscillation of the response decays exponentially with a time constant $1/\zeta\omega_n$. This is called under-damped response. On the other hand, when ζ is greater than unity, the response of the unit step input given to the system, does not exhibit an oscillating part in it. This fact is called an overdamping response. The case when the damping ratio is unity, that is, $\zeta = 1$, was also considered. In this situation, the damping of the response is governed by the natural frequency ω_n only. The actual damping in that condition is known as critical damping of the response.

As it is already seen in the associated expressions of the time response of the control system subject to the input step function, the oscillation part is present in the response when the damping ratio (ζ) is less than one and it is not present in the response when damping ratio is equal to one. That means the oscillation part of the response just disappears when the damping ratio becomes one. That is why the damping of the response at $\zeta = 1$ is known as critical damping. More precisely, when the damping ratio is unity, the response is critically dampened, and then the damping is known as critical damping. The ratio of the time constant of critical damping to that of actual damping is known as the damping ratio. The time constant of the time response of the control system is $1/\zeta\omega_n$ when $\zeta \neq 1$, and the time constant is $1/\omega_n$ when $\zeta = 1$.

$$\frac{\text{Time Constant of critical damping}}{\text{Time constant of actual damping}} = \frac{\frac{1}{\omega_n}}{\frac{1}{\zeta\omega_n}} = \zeta$$

The performance of the control system can be expressed in terms of the transient response to a unit-step input function because it is easy to generate. If it is considered a second-order control system in which a unit-step input signal is given, it is also reflected that the system is initially at rest. That is, all the initial conditions of the system are zero. The time

response characteristics of the system under a damping condition is shown in Figure 4.12. There are a few common terms in transient response characteristics, among which are:

- a) **Delay time** (θ_p) is the time required to reach 50% of its final value by a time response signal during its first cycle of oscillation.
- b) **Rise time** (t_r) is the time required to reach the final value of an under damping time response signal during its first cycle of oscillation. If the signal is overdamped, then rise time is counted as the time required by the response to rise from 10% to 90% of its final value.
- c) **Peak time** (t_p) is simply the time required by the response to reach its first peak, i.e., the peak of the first cycle of oscillation, or first overshoot.
- d) **Maximum overshoot** (M_p) is the straight-way difference between the magnitude of the highest peak of the time response and the magnitude of its steady state. The maximum overshoot is expressed in terms of the percentage of the steady-state value of the response. As the first peak of a response is normally the maximum in magnitude, the maximum overshoot is simply the normalized difference between the first peak and the steady-state value of a response, $M_p(\%) = \frac{c(t_p) - c(\infty)}{c(\infty)} * 100(\%)$ (4.30).
- e) **Settling time** (t_s) is the time required for a response to become steady. It is defined as the time required by the response to reach and remain steady within a specified range of 2 % to 5 % of its final value.
- f) **Steady-state error** (e_{ss}) is the difference between actual output and desired output over an infinite range of time, $e_{ss} = \lim_{t \rightarrow \infty} [r(t) - y(t)]$. In terms of the response of the systems, y_{ss} , the Laplace domain, supposing a second order system:

$$y_{ss} = \lim_{s \rightarrow 0} sY(s) = \lim_{s \rightarrow 0} s \left(\frac{1}{s} \right) G(S) = \lim_{s \rightarrow 0} \frac{K \omega_n^2}{s^2 + 2\zeta \omega_n s + \omega_n^2} = \frac{K \omega_n^2}{\omega_n^2} = K$$

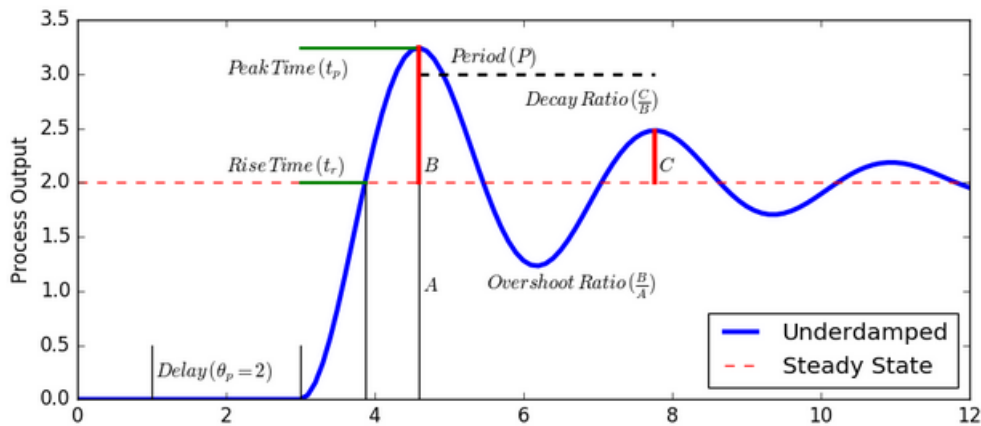


Figure 4.12: Characteristics of the second order underdamped system.

Some formulas are derived from the definitions and mathematical approaches (Figure 4.12). These are defined as follow:

a) Rise Time Formula

The expression of an underdamped second-order control system with a unit step input function,

$$\begin{aligned}
 y(t) &= 1 - \frac{e^{-\zeta\omega_n t}}{\sqrt{1-\zeta^2}} \sin \left\{ \left(\omega_n \sqrt{1-\zeta^2} \right) t + \tan^{-1} \left(\frac{\sqrt{1-\zeta^2}}{\zeta} \right) \right\} \\
 &= 1 - \frac{e^{-\zeta\omega_n t}}{\sqrt{1-\zeta^2}} \sin \left\{ \left(\omega_n \sqrt{1-\zeta^2} \right) t + \phi \right\}
 \end{aligned}$$

Again, as per definition, the magnitude of output signal at rise times, is 1. That is $y(t) = 1$, hence,

$$\begin{aligned}
 y(t) = 1 &= 1 - \frac{e^{-\zeta\omega_n t}}{\sqrt{1-\zeta^2}} \sin \left\{ \left(\omega_n \sqrt{1-\zeta^2} \right) t + \phi \right\} \Rightarrow \frac{e^{-\zeta\omega_n t}}{\sqrt{1-\zeta^2}} \sin \left\{ \left(\omega_n \sqrt{1-\zeta^2} \right) t + \phi \right\} = 0 \Rightarrow \\
 \sin \left\{ \left(\omega_n \sqrt{1-\zeta^2} \right) t + \phi \right\} &= 0 \Rightarrow \left\{ \left(\omega_n \sqrt{1-\zeta^2} \right) t + \phi \right\} = \pi \Rightarrow \mathbf{t_r} = \frac{\pi - \phi}{\omega_n \sqrt{1-\zeta^2}} \quad (4.31)
 \end{aligned}$$

b) Peak Time Formula

As per definition, at the peak time, the response curve reaches to its maximum value. Hence, at that point, $\frac{dy(t)}{dt} = 0$

$$\begin{aligned}
 \frac{dy(t)}{dt} &= -\frac{e^{-\zeta\omega_n t}}{\sqrt{1-\zeta^2}} \omega_n \sqrt{1-\zeta^2} \cos \left\{ \left(\omega_n \sqrt{1-\zeta^2} \right) t + \phi \right\} - \frac{(-\zeta\omega_n) e^{-\zeta\omega_n t}}{\sqrt{1-\zeta^2}} \sin \left\{ \left(\omega_n \sqrt{1-\zeta^2} \right) t + \phi \right\} \\
 &= 0 \Rightarrow \\
 \frac{e^{-\zeta\omega_n t}}{\sqrt{1-\zeta^2}} [-\omega_n \sqrt{1-\zeta^2} \cos \left\{ \left(\omega_n \sqrt{1-\zeta^2} \right) t + \phi \right\} + \frac{\zeta\omega_n e^{-\zeta\omega_n t}}{\sqrt{1-\zeta^2}} \sin \left\{ \left(\omega_n \sqrt{1-\zeta^2} \right) t + \phi \right\}] &= 0 \Rightarrow \\
 \omega_n \sqrt{1-\zeta^2} \cos \left\{ \left(\omega_n \sqrt{1-\zeta^2} \right) t + \phi \right\} &= \frac{\zeta\omega_n e^{-\zeta\omega_n t}}{\sqrt{1-\zeta^2}} \sin \left\{ \left(\omega_n \sqrt{1-\zeta^2} \right) t + \phi \right\} \Rightarrow \\
 \tan \left\{ \left(\omega_n \sqrt{1-\zeta^2} \right) t + \phi \right\} &= \frac{\omega_n \sqrt{1-\zeta^2}}{\zeta} = \tan \phi \Rightarrow \\
 \left(\omega_n \sqrt{1-\zeta^2} \right) t &= n\pi, \quad \text{where } n = 1, 2, 3 \dots \Rightarrow \\
 \mathbf{t_p} &= \frac{\pi}{\omega_n \sqrt{1-\zeta^2}} \quad (4.32)
 \end{aligned}$$

Because the maximum overshoot occurs at the first overshoot or frequency, the peak time arises at that frequency.

c) Maximum Overshoot Formula

If the expression of the peak time is placed in the expression of the output response $y(t)$, and it is obtained,

$$\begin{aligned}
 y(t)_{max} &= 1 - \frac{e^{-\zeta\omega_n t_p}}{\sqrt{1-\zeta^2}} \sin \left\{ \left(\omega_n \sqrt{1-\zeta^2} \right) t_p + \phi \right\} \Rightarrow y(t)_{max} = 1 - \\
 \frac{e^{-\zeta\omega_n \frac{\pi}{\omega_n \sqrt{1-\zeta^2}}}}{\sqrt{1-\zeta^2}} \sin \left\{ \left(\omega_n \sqrt{1-\zeta^2} \right) \frac{\pi}{\omega_n \sqrt{1-\zeta^2}} + \phi \right\} &\Rightarrow y(t)_{max} = 1 - \frac{e^{-\frac{\zeta\pi}{\sqrt{1-\zeta^2}}}}{\sqrt{1-\zeta^2}} \sin \{ \pi + \phi \} = 1 - \\
 \frac{e^{-\frac{\zeta\pi}{\sqrt{1-\zeta^2}}}}{\sqrt{1-\zeta^2}} (-\sin \phi) &= 1 + \frac{e^{-\frac{\zeta\pi}{\sqrt{1-\zeta^2}}}}{\sqrt{1-\zeta^2}} \sin \phi = 1 + \frac{e^{-\frac{\zeta\pi}{\sqrt{1-\zeta^2}}}}{\sqrt{1-\zeta^2}} \sqrt{1-\zeta^2} = 1 + e^{-\frac{\zeta\pi}{\sqrt{1-\zeta^2}}} \Rightarrow M_p = \\
 y(t)_{max} - 1 &= \left(1 + e^{-\frac{\zeta\pi}{\sqrt{1-\zeta^2}}} \right) - 1 \Rightarrow \mathbf{M_p} = 100 * e^{-\frac{\zeta\pi}{\sqrt{1-\zeta^2}}} \quad (4.33)
 \end{aligned}$$

So, the maximum overshoot is the value that takes the output signal at the peak time.

d) Settling Time Formula

It is already defined that the settling time of a response is that time after which the response reaches its steady-state condition with a value above nearly 95% or 98% of its final value.

It is also observed that this duration is approximately four times the time constant of a signal. At the time constant of a second-order control system, $1/\zeta\omega_n$, the expiration of settling time can be given as, $t_s = \frac{4}{\zeta\omega_n}$ (4.34).

This information and further analysis may be found in any book about Laplace transform [5, 6], or automatic control, for example, [7].

The MATLAB command that outputs this information is; `S = stepinfo(sys)` [8].

4.2.2 Impulse response of a second-order system

The equation of motion describing the behavior of a second-order mass-spring-dashpot system with a unit impulse input is:

$$\frac{Y(s)}{R(s)} = \frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} \quad \text{with } R(s) = 1 \text{ which corresponds to: } r(t) = \delta(t).$$

For an *under-damped* system, ($\zeta < 1$), the response against impulse input, assuming zero initial conditions, is:

$$y(t) = \frac{1}{\omega_d} e^{-\zeta\omega_n t} \sin \omega_d t \quad (4.35)$$

With $\omega_d = \omega_n \sqrt{1 - \zeta^2}$.

For a *critically damped* system, ($\zeta = 1$), the response against impulse input, assuming zero initial conditions, is:

$$y(t) = t e^{-\omega_n t} \quad (4.36)$$

For an *over-damped* system, ($\zeta > 1$), with zero initial conditions, the response is

$$y(t) = \frac{1}{2\omega_n\sqrt{\zeta^2-1}} \left(e^{-\omega_n(\zeta-\sqrt{\zeta^2-1})t} - e^{-\omega_n(\zeta+\sqrt{\zeta^2-1})t} \right) \quad (4.37)$$

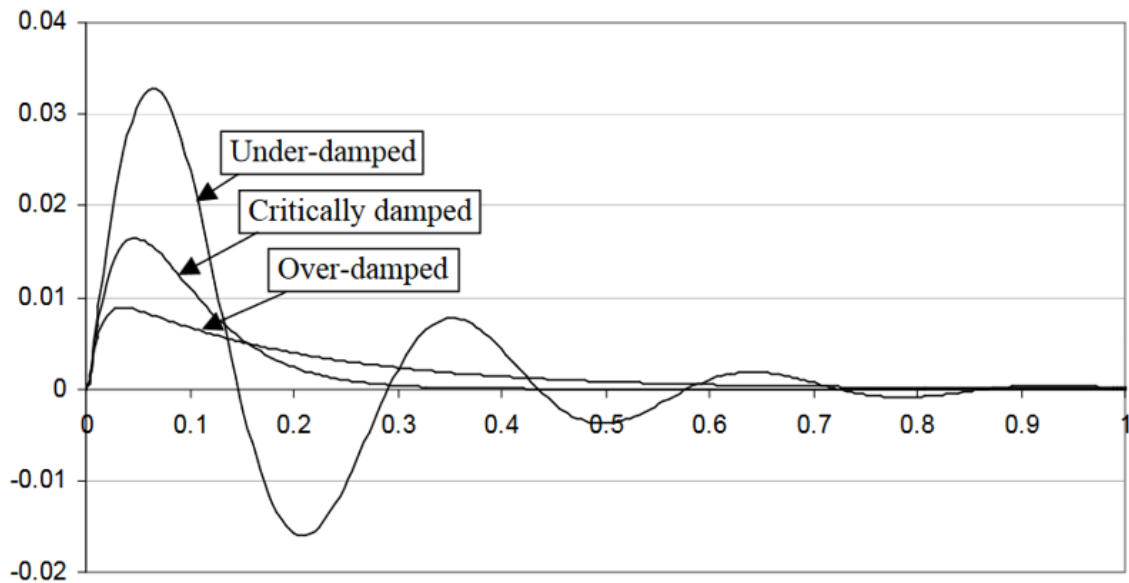


Figure 4.13: Characteristics of the second order system against impulse input.

Exercises:

1. Identify the time constant and the steady state gain of the first-order system, which has the following transfer functions against the unity step function:

a. $G_1(s) = \frac{Y(s)}{R(s)} = \frac{2}{s+3}$

b. $G_2(s) = \frac{Y(s)}{R(s)} = \frac{2}{3+5s}$

2. Consider the following systems:

4.3 $G_1(s) = \frac{1}{s^2+2s+1}$

4.4 $G_2(s) = \frac{1}{s^2+0.4s+1}$

4.5 $G_3(s) = \frac{1}{s^2+5s+1}$

4.6 $G_4(s) = \frac{1}{s^2+s+1}$

4.7 $G_5(s) = \frac{4}{s^2+2s+4}$

Find t_r (rise time), t_s (settling time), and M (overshoot) for the step response. Corroborate the results using MATLAB to plot the step responses. MATLAB command ($S = \text{stepinfo}(\text{sys})$).

Calculate the poles of the system and explain how the location of the poles is related to the properties of the step response.

References:

- [1] Devi, Rekha. "Applications of Laplace Transformation." *Research Journal of Science and Technology* 9, no. 1 (2017): 167. <http://dx.doi.org/10.5958/2349-2988.2017.00027.4>
- [2] Pérez-Esteva, Salvador. "Convolution operators for the one-sided Laplace transformation." *Časopis pro pěstování matematiky* 110, no. 1 (1985): 69–76. <http://dx.doi.org/10.21136/cpm.1985.118223>.
- [3] Sayood, K. (2021). *Properties of Laplace Transforms*. In: *Signals and Systems. Synthesis Lectures on Signal Processing*. Springer, Cham. https://doi.org/10.1007/978-3-031-02545-7_23
- [4] P.P.G. Dyke (2014). *An introduction to Laplace transforms and Fourier series*. Springer undergraduate mathematics series, 2nd Edition.
- [5] Fleisch, D. (2022). *A Student's Guide to Laplace Transforms (Student's Guides)*. Cambridge: Cambridge University Press. doi:10.1017/9781009089531.
- [6] Beerends, R., Ter Morsche, H., Van den Berg, J., & Van de Vrie, E. (2003). *Fourier and Laplace Transforms*. Cambridge: Cambridge University Press. doi:10.1017/CBO9780511806834.
- [7] Farid Golnaraghi and Benjamin C. Kuo. (2009). *Automatic Control Systems*, 9th Edition. Wiley. doi:978-0470048962.
- [8] <https://www.mathworks.com/help/ident/ref/dynamicsystem.stepinfo.html;jsessionid=ec7fe7897d5c27ca4824f737a4e0#d124e206325>

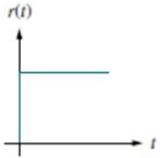
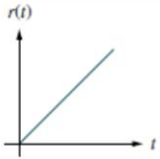
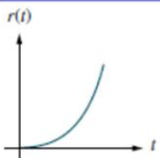
Chapter 5: Steady State Errors Analysis

The steady-state error is a measure of system accuracy. These errors arise from the nature of the inputs, the type of the system, and the nonlinearities of system components such as static friction, backlash, etc. These are generally aggravated by amplifier drifts, aging, or deterioration. The steady-state performance of a stable control system is generally judged by its steady state error to step, ramp, and parabolic inputs.

The steady-state error is a measure of system accuracy. These errors arise from the nature of the inputs, system type and from nonlinearities of system components such as static friction, backlash, etc. These are generally aggravated by amplifiers drifts, aging or deterioration. The steady state performance of a stable control system is generally judged by its steady state error to step, ramp, and parabolic inputs.

Steady-state error is defined as the difference between the input and output of a system in the limit as time goes to infinity (i.e., when the response has reached the steady state). The steady-state error will depend on the type of input (step, ramp, etc.) as well as the system type (0, I, or II) (zero, one or two poles at the origin, $s = 0$).

Table 5.1: Representation of inputs to the system: step, ramp, and parabola.

Waveform	Name	Physical interpretation	Time function	Laplace transform
	Step	Constant position	1	$\frac{1}{s}$
	Ramp	Constant velocity	t	$\frac{1}{s^2}$
	Parabola	Constant acceleration	$\frac{1}{2}t^2$	$\frac{1}{s^3}$

Steady-state error analysis is only useful for stable systems. It is necessary to check the system for stability before performing a steady-state error analysis. Many of the

techniques that are presented give an answer even if the system is unstable; obviously, this answer is meaningless for an unstable system.

Before talking about the relationships between steady-state error and system type, it is shown how to calculate error regardless of system type or input. Then, it starts deriving formulas that will be applied when a steady state-error analysis is performed. Steady-state error can be calculated from the open- or closed-loop transfer function for Unity feedback systems. For example, the following system is considered:



Figure 5.1: Block diagram representation of the Unity feedback system.

which is equivalent to the following system:

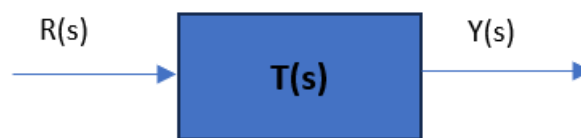


Figure 5.2: Equivalent block diagram of a unity feedback system from Figure 5.1.

The steady state error for this system is calculated from either the open or closed-loop transfer function using the final value theorem (remember that this theorem can only be applied if the denominator has no poles in the right-half plane, i.e., the system is stable):

$$e(\infty) = \lim_{s \rightarrow 0} \frac{sR(s)}{1+G(s)} \quad (5.1)$$

$$e(\infty) = \lim_{s \rightarrow 0} sR(s)[1 - T(s)] \quad (5.2)$$

Now, different inputs are plugged into the Laplace transforms, and the equations to calculate steady-state errors from open-loop transfer functions given different inputs are found:

1. Step Input ($R(s) = 1/s$):

$$e(\infty) = \frac{1}{1 + \lim_{s \rightarrow 0} G(s)} = \frac{1}{1 + K_p} \Rightarrow K_p = \lim_{s \rightarrow 0} G(s) \quad (5.3)$$

- Ramp Input ($R(s) = 1/s^2$):

$$e(\infty) = \frac{1}{\lim_{s \rightarrow 0} sG(s)} = \frac{1}{K_v} \Rightarrow K_v = \lim_{s \rightarrow 0} sG(s) \quad (5.4)$$

- Parabolic Input ($R(s) = 1/s^3$):

$$e(\infty) = \frac{1}{\lim_{s \rightarrow 0} s^2G(s)} = \frac{1}{K_a} \Rightarrow K_a = \lim_{s \rightarrow 0} s^2G(s) \quad (5.5)$$

When a controller is designed, it is usually wanted to compensate for the disturbances to a system. Let's say that we have the following system with a disturbance:

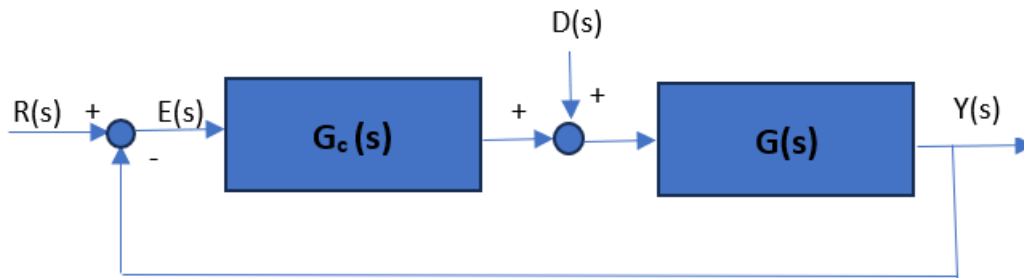


Figure 5.3: Block diagram representation of a unity feedback system with the controller and the disturbances.

It can be found the steady-state error for a step disturbance input with the following equation:

$$e(\infty) = \frac{1}{\lim_{s \rightarrow 0} \frac{1}{G(s)} + \lim_{s \rightarrow 0} G_c(s)} \quad (5.6)$$

Lastly, it is calculated steady-state error for non-unity feedback systems:

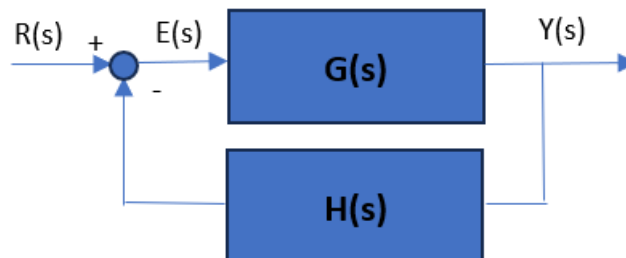


Figure 5.4: Block diagram representation of a non-unity feedback system.

By manipulating the blocks in the following manner:

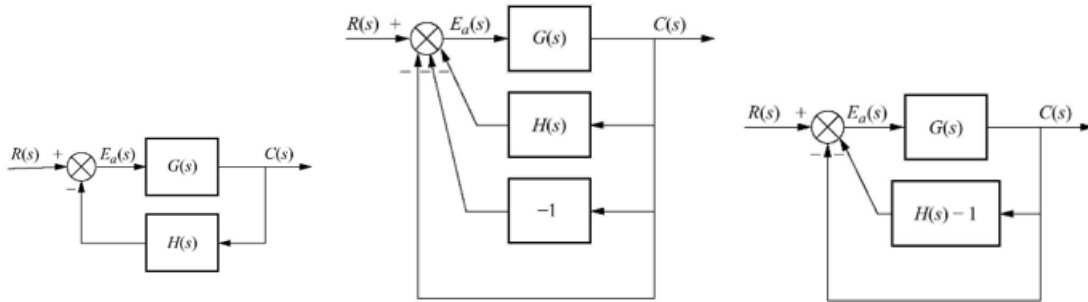


Figure 5.5: Steps for the conversion of the block diagram of a non-unity feedback system into a unity feedback system.

The system is modeled as a unity feedback system as follows:

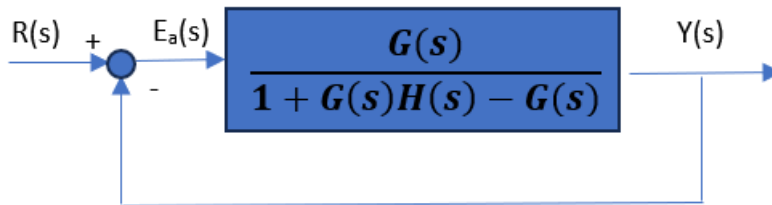


Figure 5.6: Conversion of a non-unity feedback system into a unity feedback system.

Now, simply the equations mentioned above are applied.

If the equations are referred to for calculating steady-state errors for unity feedback systems, it is found that certain constants are defined, known as the static error constants. These constants are the position constant (K_p), the velocity constant (K_v), and the acceleration constant (K_a). Knowing the value of these constants as well as the system type, we can predict if our system is going to have a finite steady-state error.

First, it addresses the system type. The system type is defined as the number of pure integrators in a system. That is, the system type is equal to the value of n when the system is represented as in the following figure:

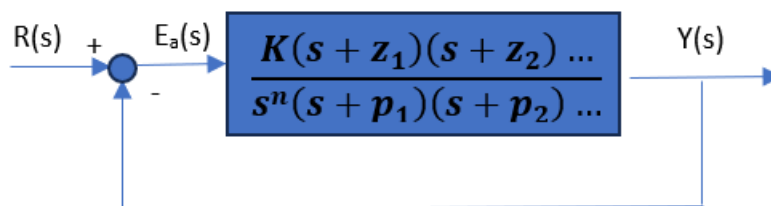


Figure 5.7: Block diagram representation of a unity feedback system with a general transfer function system.

Therefore, a system can be type 0 ($n=0$), type 1 ($n=1$), etc. The steady-state error relates to system types in the following way:

Table 5.2: Type 0 systems errors against step, ramp, and parabolic inputs.

Type 0 systems	Step input	Ramp input	Parabolic input
Steady state error formula	$\frac{1}{(1 + Kp)}$	$\frac{1}{Kv}$	$\frac{1}{Ka}$
Static error constant	$Kp = \text{constant}$	$Kv = 0$	$Ka = 0$
Error	$\frac{1}{(1 + Kp)}$	infinity	infinity

Table 5.3: Type 1 systems errors against step, ramp, and parabolic inputs.

Type 1 systems	Step input	Ramp input	Parabolic input
Steady state error formula	$\frac{1}{(1 + Kp)}$	$\frac{1}{Kv}$	$\frac{1}{Ka}$
Static error constant	$Kp = \text{infinity}$	$Kv = \text{constant}$	$Ka = 0$
Error	0	$\frac{1}{Kv}$	infinity

Table 5.4: Type 2 systems errors against step, ramp, and parabolic inputs.

Type 2 systems	Step input	Ramp input	Parabolic input
Steady state error formula	$\frac{1}{(1 + Kp)}$	$\frac{1}{Kv}$	$\frac{1}{Ka}$
Static error constant	$Kp = \text{infinity}$	$Kv = \text{infinity}$	$Ka = \text{constant}$
Error	0	0	$\frac{1}{Ka}$

Example 5.1: Consider the system in the following figure:

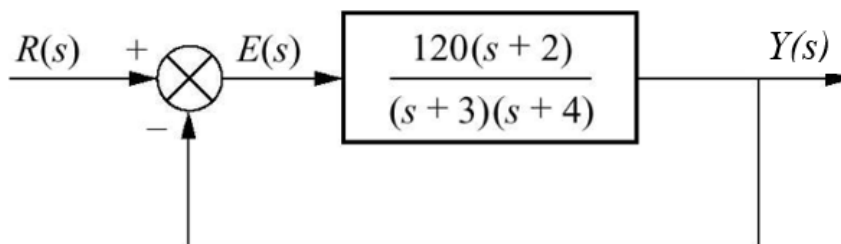


Figure 5.8: Block diagram representation of a unity feedback type 0 system.

Calculate the steady-state errors for five times the step and ramp inputs.

The steady state-error for an input which is five times the step, $r(t) = 5u(t)$ ($R(s) = \frac{5}{s}$):

$$e(\infty) = \lim_{s \rightarrow 0} \frac{s \frac{5}{s}}{1 + G(s)} = \frac{5}{1 + \lim_{s \rightarrow 0} G(s)} = \frac{5}{1 + 20} = \frac{5}{21}$$

The steady state error for an input which is five times the ramp $r(t) = 5tu(t)$ ($R(s) = \frac{5}{s^2}$):

$$e(\infty) = \lim_{s \rightarrow 0} \frac{s \frac{5}{s^2}}{1 + G(s)} = \frac{5}{\lim_{s \rightarrow 0} sG(s)} = \infty$$

Example 5.2: Consider the system in the following figure:

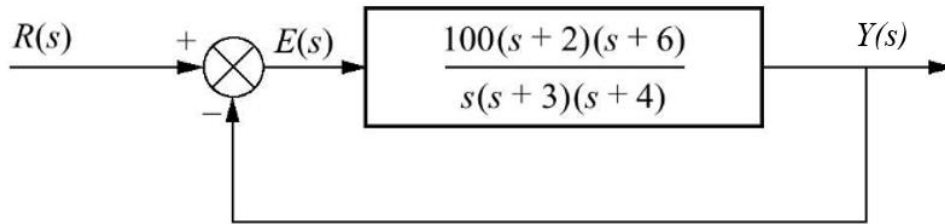


Figure 5.9: Block diagram representation of a unity feedback type I system.

Calculate the steady state errors for five times the step, ramp, and parabolic inputs.

The steady state error for an input which is five times the step, $r(t) = 5u(t)$ ($R(s) = \frac{5}{s}$):

$$e(\infty) = \lim_{s \rightarrow 0} \frac{s \frac{5}{s}}{1 + G(s)} = \frac{5}{1 + \lim_{s \rightarrow 0} G(s)} = \frac{5}{\infty} = 0$$

The steady state error for $r(t) = 5tu(t)$ ($R(s) = \frac{5}{s^2}$):

$$e(\infty) = \lim_{s \rightarrow 0} \frac{s \frac{5}{s^2}}{1 + G(s)} = \frac{5}{\lim_{s \rightarrow 0} sG(s)} = \frac{5}{100} = \frac{1}{20}$$

The steady state error for $r(t) = 5t^2u(t)$ ($R(s) = \frac{10}{s^3}$):

$$e(\infty) = \lim_{s \rightarrow 0} \frac{s \frac{10}{s^3}}{1 + G(s)} = \frac{10}{\lim_{s \rightarrow 0} s^2 G(s)} = \frac{10}{0} = \infty$$

Example 5.3: Consider the control system in the following figure:

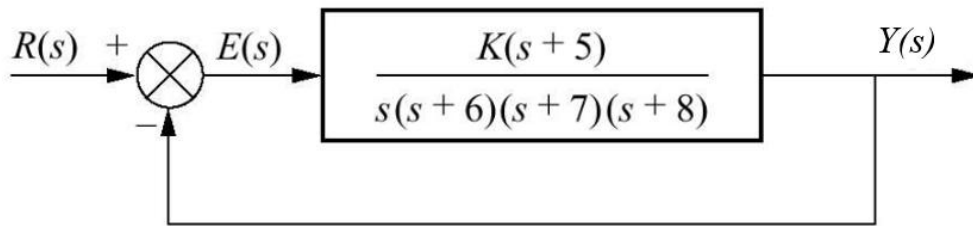


Figure 5.10: Block diagram representation of a unity feedback type I system.

Calculate K such as that the system has a finite steady state error with a constant velocity equal to 10.

Since the system is type I the finite steady-state error should be for a ramp input.

$$e(\infty) = \frac{1}{K_v}$$

So,

$$K_v = 10 = \lim_{s \rightarrow 0} sG(s) = \frac{5K}{6 \cdot 7 \cdot 8}$$

And $K = 672$.

Exercises:

1. A unity feedback system has the following transfer functions.

a. $G_1(s) = \frac{K}{(s+2)(s+3)}$

b. $G_2(s) = \frac{K}{s(s+2)(s+3)}$

c. $G_3(s) = \frac{K}{s^2(s+2)(s+3)}$

Calculate steady state errors for a unity step, ramp and parabolic inputs, supposing $K=1$. If $K \neq 1$ and unknown, define in which cases the system has a finite steady state-error with constant position, velocity, and acceleration. Calculate K such as that the system has a finite steady state-error equal to 10, for the elucidated cases. (Hint: classify the kind of system is dealt to calculate the corresponding finite steady-state error according to Tables 5.2- 5.4).

References:

[1] Farid Golnaraghi and Benjamin C. Kuo, 2017, Automatic Control Systems, 10th ed. McGraw-Hill Education.

Chapter 6: Stability of Continuous-time Systems

In this section, the stability of continuous time systems is provided. Stability is a key property of a dynamic system. First, a definition of stability based on the Laplace transform is given. There are several criteria to determine the stability of a dynamic system. The Routh-Hurwitz criterion and the root locus are first dealt with. Frequency analysis, i.e., Bode plot and Nyquist plot, are also introduced, giving examples and tools to examine them further.

6.1 Stability

Bounded input, bounded output (BIBO) stability is a form of stability often used for signal processing applications. The requirement for a linear, shift-invariant, continuous, or discrete time system to be BIBO stable is for the output to be bounded for every input to the system that is bounded.

There is a characterization of the stability of a continuous transfer function in the s-plane, which gives a definition of a stable, unstable, and marginally stable system.

1. **Stable System:** If all the roots of the characteristic equation of the transfer function of the system (i.e., the equation that makes the denominator of the transfer function equal to zero) lie on the **left** half of the s-plane (i.e., the roots are negatives), then the system is said to be a stable system.
2. **Marginally Stable System:** If one or more of the roots of the system lie on the imaginary axis of the s-plane and the rest lie on the left half of the s-plane, then the system is said to be marginally stable.
3. **Unstable System:** If any of the roots of the system lie on the **right** half of the s-plane, then the system is said to be an unstable system.

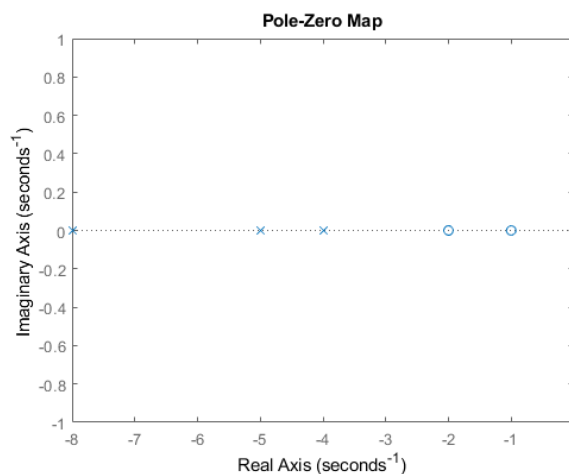


Figure 6.1: Poles and zeros colocation on the left-hand complex s-plane

$$\text{of the transfer function } H(S) = 10 \frac{(s+1)(s+2)}{(s+4)(s+5)(s+8)} .$$

Figure represents the position of zeros (o), the numerator of the transfer function $H(s) = 10 \frac{(s+1)(s+2)}{(s+4)(s+5)(s+8)}$ is equal to 0, this happens in -1 and -2, and poles (x), the denominator or characteristic equation of F(s) is equal to 0, this happens in -4, -5 and -8, in the s-plane. MATLAB uses the following function to represent figure:

```
% s-plane representation of zeros and poles of a transfer function
numerator = 10*[1 3 2]; % numerator of the transfer function in polynomial form
denominator = [1,17,92,160]; % denominator of the transfer function in polynomial form
sys = tf(numerator,denominator); % tf defines the transfer function
figure; iopzplot(sys); % iopzplot function plots zeros and poles in the s-plane
```

As it can be observed in figure all poles are positioned in the negative part of the s-plane, the system is stable.

6.2 Routh-Hurwitz Criterion

Despite the calculation exigencies of the Routh-Hurwitz criterion, it is very useful in certain cases. For instance, to find the poles of a Laplace transform that is a rational function in s , you need to factor the denominator polynomial, which is easy with computers. The problem is, for denominator polynomials of degree 5 and higher, there's in general no formula for factorizing the polynomial, which means that numerical methods are used. This is fine if you have numerical values for the coefficients of the denominator polynomial. But if the denominator polynomial has one or more coefficients written symbolically, then it is not possible to use numerical methods to factorize the denominator, and thus it is not possible to find the poles.

The Ziegler-Nichols method, which is not emphasized in this document, is an application of the Routh-Hurwitz criterion to tune a PID controller. During the tuning process with such a method, the value of a constant K , which makes the output signal of the system sinusoidal in the steady state must be determined. It is known that it happens when the Laplace transform of the output signal has no poles to the right part of the imaginary axis, has no repeated poles on the imaginary axis, and has just one (not more) simple pair of complex conjugate poles on the imaginary axis. Thus, to find the value of K , it is needed to see which value would create those poles. But to find those poles, it is required to factor the denominator, yet it is not possible because it has a symbolic variable K . Fortunately, it is still possible to find which value of K would produce sinusoidal oscillations of constant amplitude by using the Routh-Hurwitz table and criterion.

Necessary but not sufficient conditions for stability

At this point, this must follow some conditions to make any system stable, or it can be said that there are some necessary conditions to make the system stable.

Consider a system with a **characteristic equation** (the equation obtained by equating the characteristic polynomial, the denominator of the transfer function, to zero).

$$a_0 s^m + a_1 s^{m-1} + \dots + a_m = 0$$

1. All the coefficients of the equation should have the same sign and be positive.

2. There should be no missing term.

If all the coefficients have the same sign and there are no missing terms, there is no guarantee that the system will be stable. If the above-given conditions are not satisfied, then the system is said to be unstable. To check the stability of the system, the **Routh Hurwitz Criterion** is used. This criterion is given by A. Hurwitz and E.J. Routh.

Advantages of Routh-Hurwitz criterion:

1. It shows the stability of the system without solving the equation.
2. It is easy to determine the relative stability of the system.
3. By this method, the range of K for stability can be determined.
4. By this method, the point of intersection of the root locus with an imaginary axis can also be determined.

Limitations of Routh-Hurwitz criterion:

1. This criterion is applicable only for a linear system.
2. It does not provide the exact location of poles on the right and left half of the s -plane.
3. In case of the characteristic equation, it is valid only for real coefficients.

Statement of Routh-Hurwitz Criterion: Routh-Hurwitz stability criterion identifies the conditions when the poles of a polynomial cross into the right-hand half plane of s , and hence it would be considered as an unstable system in control engineering. Considering the following characteristic polynomial:

$$a_0s^m + a_1s^{m-1} + \dots + a_m = 0$$

when the coefficients a_0, a_1, \dots, a_m are all the same sign, and none is zero.

Step 1: Arrange all the coefficients of the above equation in two rows:

Row 1: a_0 a_2 a_4

Row 2: a_1 a_3 a_5

Step 2: From these two rows, a third row will be formed:

Row 1: a_0 a_2 a_4

Row 2: a_1 a_3 a_5

Row 3: b_1 b_3 b_5

Where, $b_1 = -\frac{1}{a_1} \begin{vmatrix} a_0 & a_2 \\ a_1 & a_3 \end{vmatrix} = -\frac{a_0 a_3 - a_1 a_2}{a_1}$ and $b_3 = -\frac{1}{a_1} \begin{vmatrix} a_0 & a_4 \\ a_1 & a_5 \end{vmatrix} = -\frac{a_0 a_5 - a_1 a_2}{a_1}$

Step 3: Now, the fourth row is built by using the second and third row:

Row 1: a_0 a_2 a_4

Row 2: a_1 a_3 a_5

Row 3: b_1 b_3 b_5

Row 4: c_1 c_3 c_5

Where, $c_1 = -\frac{1}{b_1} \begin{vmatrix} a_1 & a_3 \\ b_1 & b_3 \end{vmatrix} = -\frac{a_1 b_3 - b_1 a_3}{b_1}$ and $c_3 = -\frac{1}{b_1} \begin{vmatrix} a_1 & a_5 \\ b_1 & b_5 \end{vmatrix} = -\frac{a_1 b_5 - b_1 a_5}{b_1}$

Step 4: This procedure of forming a new row is continued as many times is necessary until it is reduced the matrix to a second order.

The **Routh Hurwitz criterion states** that a system is stable if and only if the roots of the first column have the same sign; if it does not have the same sign or there is a sign change, then the number of sign changes in the first column is equal to the number of roots of the characteristic equation in the right half of the s -plane, i.e., equal to the number of roots with positive real parts.

Example 6.1

Check the stability of the system whose characteristic equation is given by:

$$s^4 + 2s^3 + 6s^2 + 4s + 1 = 0$$

Solution:

Obtain the array of coefficients as follows:

$$\begin{array}{l} s^4 \quad 1 \quad 6 \quad 1 \\ s^3 \quad 2 \quad 4 \quad 0 \\ s^2 \quad 4 \quad 1 \quad 0 \\ s^1 \quad 3.5 \quad 0 \quad 0 \\ s^0 \quad 1 \quad 0 \quad 0 \end{array}$$

$$b_1 = -\frac{1}{a_1} \begin{vmatrix} a_0 & a_2 \\ a_1 & a_3 \end{vmatrix} = -\frac{1}{2} \begin{vmatrix} 1 & 6 \\ 2 & 4 \end{vmatrix} = -\frac{a_0 a_3 - a_1 a_2}{a_1} = -\frac{(1 * 4 - 2 * 6)}{2} = 4$$

$$c_1 = -\frac{1}{b_1} \begin{vmatrix} a_1 & a_3 \\ b_1 & b_3 \end{vmatrix} = -\frac{1}{4} \begin{vmatrix} 2 & 4 \\ 4 & 1 \end{vmatrix} = 3.5 \quad ; \quad b_3 = -\frac{1}{a_1} \begin{vmatrix} a_0 & a_4 \\ a_1 & a_5 \end{vmatrix} = -\frac{1}{2} \begin{vmatrix} 1 & 1 \\ 2 & 0 \end{vmatrix} = 1 \quad ; \quad \text{and}$$

$$d_1 = -\frac{1}{c_1} \begin{vmatrix} b_1 & b_3 \\ c_1 & c_3 \end{vmatrix} = -\frac{1}{3.5} \begin{vmatrix} 4 & 1 \\ 3.5 & 0 \end{vmatrix} = 1.$$

Since all the coefficients in the first column are of the same sign, i.e., positive, the given equation has no roots with positive real parts; therefore, the system is said to be stable.

Example 6.2

Check the stability of the system whose characteristic equation is given by:

$$s^3 + 10s^2 + 31s + 1030 = 0$$

Solution:

Obtain the array of coefficients as follows:

$$s^3 \quad 1 \quad 31 \quad 0$$

s^2	10	1030	0	(it is divided by 10 without loss of generality)
s^1	-72	0	0	
s^0	103	0	0	

As in this case, there is a sign change, it must be checked that the number of sign changes in the first column. This number of changes tells the number of roots of the characteristic equation that are in the right half of the s -plane, i.e., equals the number of roots with positive real parts or are unstable roots.

There are two sign changes: $1 \rightarrow -72$ and $-72 \rightarrow 103$, so two roots in the right half of the s -plane or unstable.

Note that any row can be multiplied or divided by any positive constant without changing the result.

The program “rhStabilityCriterion.m” presented in [2] calculates automatically if the system is stable or not according to Routh-Hurwitz criterion, and outputs the poles colocation in s -plane. The presented Routh-Hurwitz criterion further explains that from a differential equation system, difficult to solve, the characteristic equation of the system is transformed into a polynomial equation, through an algebraic transformation, and the system is studied in the s -plane, facilitating the more intuitive comprehension of the stability, robustness, and further properties and requirements of the system to be controlled.

6.3 Root Locus

In control and stability theory, **root locus analysis** is a graphical method for examining how the roots of a system change with variation of a certain system parameter, commonly a gain within a feedback system. This is a technique used as a stability criterion in the field of classical control theory, which can determine the stability of the system. The root locus plots the poles of the closed-loop transfer function in the complex s -plane as a function of a gain parameter. Root locus plots are a very useful way to predict the behavior of a closed-loop system as some parameter of the system, typically a gain, K , is changed.

In addition to determining the stability of the system, the root locus can be used to design the damping ratio (ζ) and natural frequency (ω_n) of a feedback system by selecting an appropriate value for the gain K . Lines of constant damping ratio can be drawn radially from the origin, and lines of constant natural frequency can be drawn as arcs whose center points coincide with the origin. By selecting a point along the root locus that coincides with a desired damping ratio and natural frequency, a gain K can be calculated and implemented in the controller. More elaborate techniques of controller design using the root locus are available in most control textbooks, and some examples are shown in this document too. For instance, lag, lead, PI, PD, and PID controllers can be designed approximately with this technique.

The definition of the damping ratio and natural frequency presumes that the overall feedback system is well approximated by a second-order system, i.e., the system has a dominant pair of poles. This is often not the case, so it is good practice to simulate the final design to check if the project goals are satisfied.

Closed-loop poles:

The root locus of an (open-loop) transfer function $H(s)$ is a plot of the locations (locus) of all possible closed-loop poles with some parameter, often a proportional gain K , varied between 0 and ∞ . The figure below shows a unity-feedback architecture, but the procedure is identical for any open-loop transfer function $H(s)$, even if some elements of the open-loop transfer function are in the feedback path.

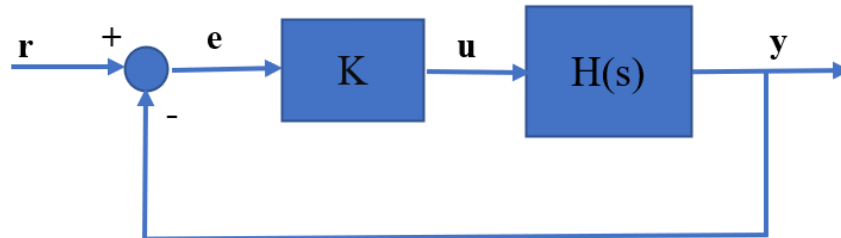


Figure 6.2: Closed-loop block diagram for root-locus analysis.

The closed-loop transfer function in this case is:

$$\frac{Y(s)}{R(s)} = \frac{KH(s)}{1 + KH(s)}$$

and thus, the poles of the closed-loop system are values of s such that $1 + KH(s) = 0$, called the characteristic equation.

If $H(s) = \frac{B(s)}{A(s)}$, then the characteristic equation, $1 + KH(s) = 0$, can be rewritten as, $A(s) + KB(s) = 0$ or $\frac{A(s)}{K} + B(s) = 0$.

Let n be the order of $A(s)$ and m be the order of $B(s)$ (the order of the polynomial corresponds to the highest power of s).

It is considered to be all positive values of K . In the limit as $K \rightarrow 0$, the poles of the closed-loop system are solutions of $A(s) = 0$ (poles of $H(s)$). In the limit as $K \rightarrow \infty$, the poles of the closed-loop system are solutions of $B(s) = 0$ (zeros of $H(s)$).

No matter the choice of K , **the closed-loop system has n poles**, where n is the number of poles of the open-loop transfer function $H(s)$. **The root locus then has n branches**, each branch starts at a pole of $H(s)$ and approaches a zero of $H(s)$. If $H(s)$ has more poles than zeros (as is often the case), $m < n$ and it is said that $H(s)$ has **zeros at infinity**. In this case, the limit of $H(s)$ as $s \rightarrow \infty$ is zero. The number of zeros at infinity is $n - m$, the number of open-loop poles minus the number of open-loop zeros, and it is the number of branches of the root locus that lead to the asymptotes.

Since the root locus consists of the locations of all possible closed-loop poles, the root locus helps to choose the value of the gain K to achieve the type of performance desired. If any of the selected poles are on the right-half complex plane, the closed-loop system will be unstable. The poles that are closest to the imaginary axis have the greatest influence on the closed-loop response, so even if a system has three or four poles, it may still behave like a second- or a first-order system, depending on the location(s) of the dominant pole(s).

Example 6.3: Consider an **open-loop** system that has a transfer function of

$$H(s) = \frac{Y(s)}{U(s)} = \frac{1}{s(s+1)(s+4)}$$

Obtain the DC-gain K of a proportional controller such that the damping ratio of the closed loop poles will be equal to 0.6. Obtain the root locus, step response, and time-domain specifications for the compensated system. For this purpose, use the MATLAB Control System Toolbox functions **rlocus** and **sgrid**(ζ , ω_n) to obtain the root locus and the gain K for $\zeta = 0.6$. Also use **ltiview** function to obtain the system step response and the time domain specifications.

Solution:

The following commands plot the transfer function and the constant line corresponding to $\zeta = 0.6$ and $\omega_n = 1$. Please, note the difference in defining the open loop transfer function with respect to the previous example.

```
num=1; % numerator transfer function
den=[1 5 4 0]; % denominator transfer function
rlocus(num, den);
hold on
sgrid(0.6, 1) % plots constant line  $\zeta = 0.6$  & constant line  $\omega_n = 1$ 
```

result in

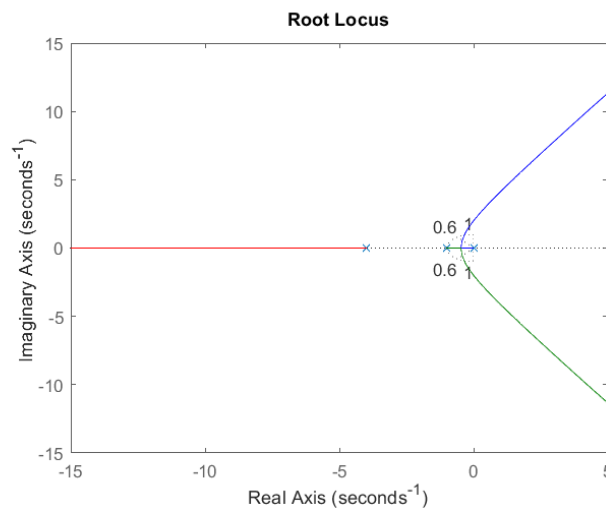


Figure 6.3: Root locus of the example and **sgrid**($\zeta = 0.6$, $\omega_n = 1$) picture.

Zoom in at the area of intersection, click at the intersection, hold and move the mouse at intersection, and adjust for damping: 0.6. Right-click in the intersection, and the information will pop up in the picture. The gain is found to be 2.06.

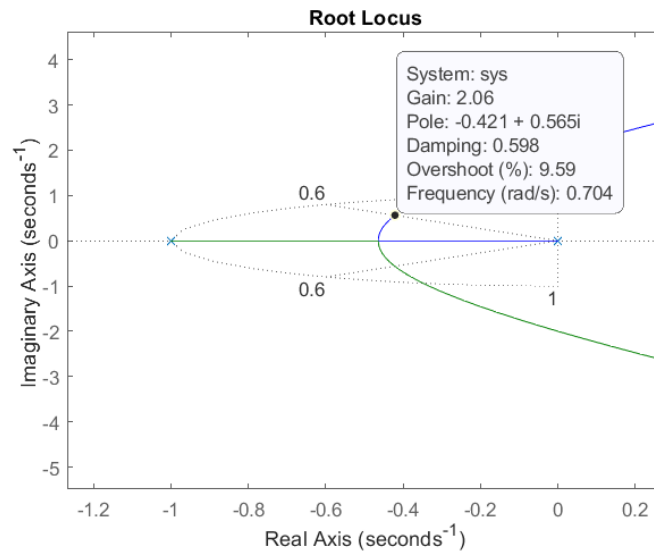


Figure 6.4: Zoom in on Figure 6.3.

In addition, the percentage overshoot and natural frequency are obtained, i.e., $M_p = 9.59\%$, and $\omega_n = 0.704$. To obtain the step response and time-domain specifications, it is used the following commands:

```
numc=2.06
denc=[1 5 4 2.06];
T=tf(numc, denc)
ltiview('step', T)
```

The result is shown in Figure 6.5. Right-click on the LTI Viewer, use Characteristics to mark peak response, peak time, settling time, and rise time. From the File Menu, use Print to Figure to obtain a figure plot.

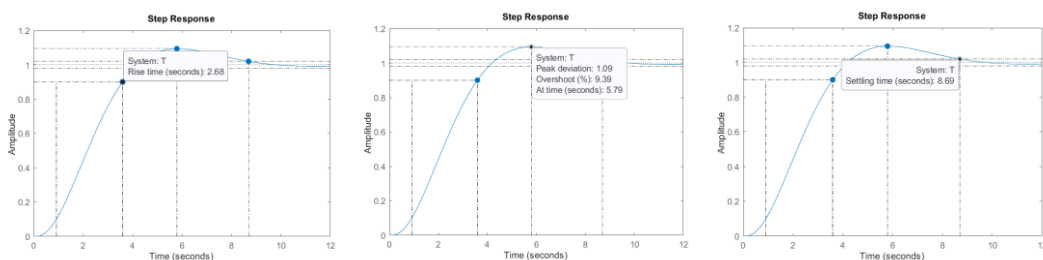


Figure 6.5: Step response with, rise time (a), peak time and overshoot (b), and settling time (c) values.

Example 6.4: Consider an **open-loop** system that has a transfer function:

$$H(s) = \frac{Y(s)}{U(s)} = \frac{s + 7}{s(s + 5)(s + 15)(s + 20)}$$

How the feedback controller for the system is designed using the root-locus method? *It is assuming the design criteria are 5% overshoot and 1 second rise time.*

Solution:

Create a m-file or MATLAB file titled, for instance, rootlocus.m. Within a m-file created in MATLAB, the transfer function model is defined, and the **rootlocus** command is used in MATLAB, for example, as follows:

```
s = tf('s'); % define s-variable as transfer function
sys = (s+7)/(s*(s+5)*(s+15)*(s+20)); % define transfer function
rlocus(sys); % call root locus
axis([-22 3 -15 15]) % define axis values.
```

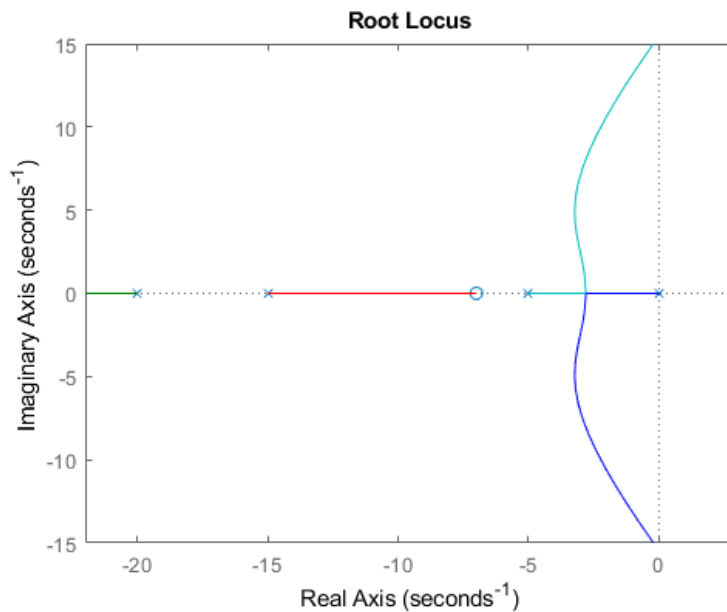


Figure 6.6: Poles and zeros colocation in the complex s-plane under variation of K -gain value.

Choosing a value of K from the root locus

The plot above shows all possible closed-loop pole locations for a pure proportional controller (varying K from 0 to ∞ show the zero and pole placement in the s-plane, note that the dynamics of the closed-loop are varied as the value of the proportional control K varies). In this case, not all these closed-loop pole locations indicate satisfaction with the design criteria. To determine what part of the locus is acceptable, the command **sgrid** (ζ , ω_n) is used to plot lines of constant damping ratio and natural frequency. These two arguments are the damping ratio (ζ) and the natural frequency (ω_n). In this case, it is needed to have an overshoot of less than 5% (which means a damping ratio ζ of greater than 0.7, to check this, use the overshoot formula and calculate ζ) and a rise time of 1 second (which means a natural frequency ω_n greater than 1.8, corroborate using the rise time formula). Introducing the **sgrid** command in the MATLAB file, figure 5.4 is plotted:

```
 $\zeta$  = 0.7; % define damping ratio
 $\omega_n$  = 1.8; % define natural frequency
sgrid( $\zeta$ ,  $\omega_n$ ) % use command sgrid
```

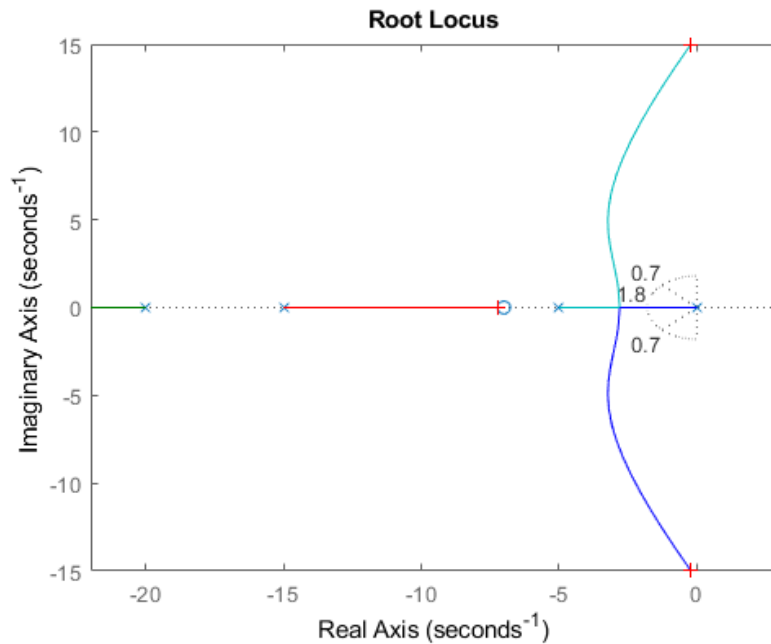


Figure 6.7: Poles and zeros colocation in the complex s -plane for selection of K -gain value.

On the plot above, the two dotted lines at about a 45-degree angle indicate pole locations with $\zeta = 0.7$; in between these lines, the poles will have $\zeta > 0.7$ and outside of these lines $\zeta < 0.7$. The semicircle indicates pole locations with a natural frequency $\omega_n = 1.8$; inside the circle, $\omega_n < 1.8$ and outside the circle $\omega_n > 1.8$.

To satisfy the control requirements, i.e., to make the overshoot less than 5%, the poles must be in between the two angled dotted lines, and to make the rise time shorter than 1 second, the poles must be outside of the dotted semicircle. So now it is known what part of the root locus, which possible closed-loop pole locations, satisfy the given requirements. All the poles in this location are in the left-half plane, so the closed-loop system will be stable.

From the plot above, it is seen that part of the root locus is inside the desired region. Therefore, in this case, only a proportional controller is needed to move the poles to the desired region (normally, two degrees of freedom are needed in the control to satisfy two requirements). The *rlocfind* command in MATLAB is used to choose the desired poles on the locus:

```
[k,poles] = rlocfind(sys) %use rlocfind command to choose poles
```

If the point where it is desired to choose the closed-loop poles is clicked on in the plot, the design criteria is satisfied at these points, and they are selected as the design of the controller.

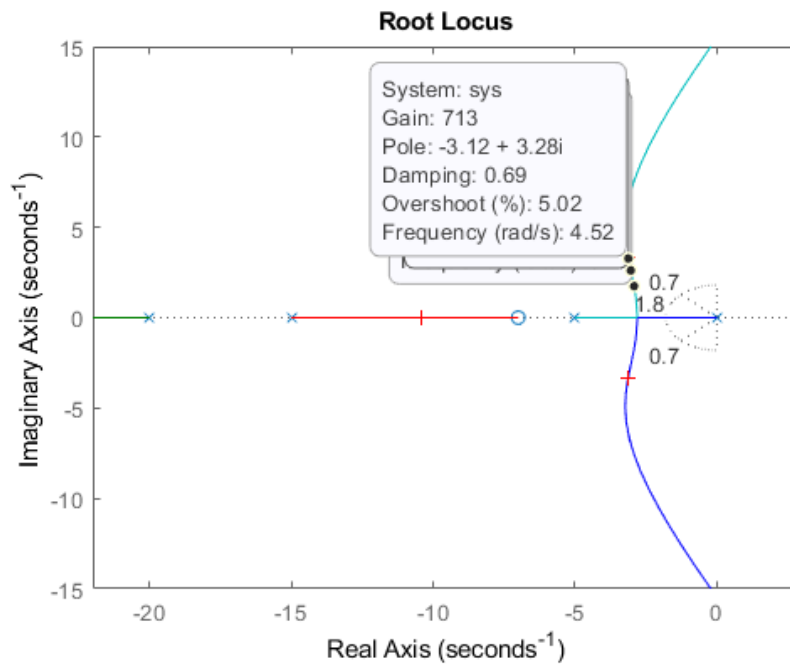


Figure 6.8: Poles and zeros colocation in the complex s-plane and selection criterion of K-gain value.

Note that since the root locus may have more than one branch, when a pole is selected, it is also identified where other closed-loop poles are for the same corresponding value of K . These poles will affect the response too. From the plot above, it is seen that of the four poles selected (indicated by "x" signs), the two closest to the imaginary axis are in the desired region. Since these poles tend to dominate the response, the requirements will be met for a proportional controller with this value of K .

Closed-Loop Response:

To verify the step response, it is needed to know the closed-loop transfer function. It is possible to computer reducing the block diagram or with the following MATLAB function (there is no need to enter a value for K if the `rlocfind` command was used):

```
K = 722.4108 %define k minuscule
```

```
%output of MATLAB; value of poles  
poles =
```

```
-23.3179 + 0.0000i  
-10.4289 + 0.0000i  
-3.1266 + 3.3195i  
-3.1266 - 3.3195i
```

```
K = 350; % input K value if rlocfind command was not used  
Sys_cl = feedback(K*sys, 1) % define the closed loop system
```

```
%MATLAB will output the transfer function of the closed-loop system
```


$$\text{Sys_cl} = \frac{722.4 s + 5057}{s^4 + 40 s^3 + 475 s^2 + 2222 s + 5057}$$

Continuous-time transfer function

The two arguments to the function `feedback` are the transfer function in the forward path and the transfer function in the feedback path of the open-loop system. In this case, the system is unity feedback. If there is feedback different from that non-unity, the MATLAB function `feedback`, shows how to find the closed-loop transfer function with a gain in the feedback path.

Checking the step response of the closed-loop system with the chosen value of K :

```
Step(sys_cl) % output the step function of the closed loop system
```

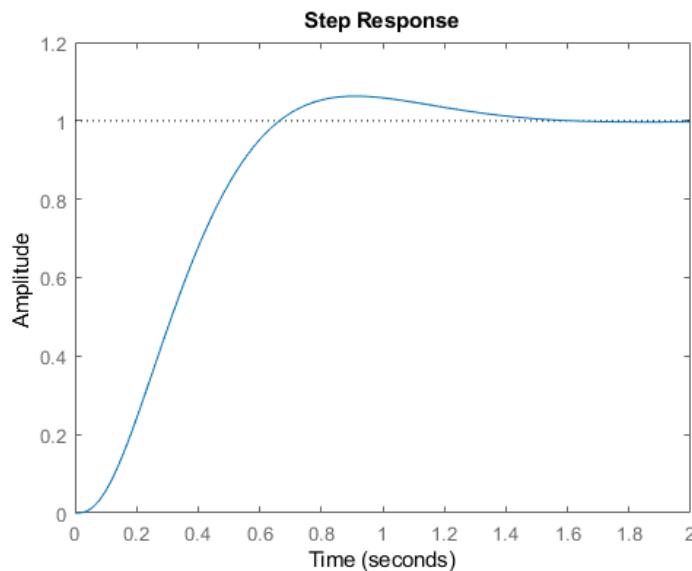


Figure 6.9: Response of the closed loop system after selection of K -value.

As it is expected, this response has an overshoot of less than 5% and a rise time of less than 1 second.

6.4 Frequency domain analysis

Until now, the response of the system was calculated against step and impulse input. In real systems, the input of the system can take any form. Frequency domain analysis provides a tool to deal with any kind of function in the input by representing this function to the system as a series decomposition of sines and cosines.

Recall that for a function $f(x)$, the Fourier series is given by

$$f(x) = a_0 + \sum_{n=1}^{\infty} (a_n \cos nx + b_n \sin nx)$$

which means that any arbitrary signal $u(t)$ can be represented by a Fourier series, i.e., as an infinite sum of weighted harmonic functions. Now, consider an input signal $u(t) = U \sin(\omega t)$ passed through an asymptotically stable transfer function $G(s)$.

Steady-state output will be $y(t) = U|G(j\omega)| \sin(\omega t + \angle G(j\omega))$, where $|G(j\omega)|$ represents the magnitude of $G(s)$ and $\angle G(j\omega)$ the phase.

To construct the *magnitude* and *phase* responses of a system represented by a transfer function $G(s)$ with a transport delay L , substitute $s = j\omega$. Then,

$$G(j\omega) = \frac{b_m(j\omega)^m + b_{m-1}(j\omega)^{m-1} + \dots + b_1(j\omega) + b_0}{a_n(j\omega)^n + a_{n-1}(j\omega)^{n-1} + \dots + a_1(j\omega) + a_0} e^{-L(j\omega)}$$

At a particular frequency ω_k

$$A_k = |G(j\omega_k)|, \phi_k = \arg(G(j\omega_k)) = \angle G(j\omega_k)$$

Where $|\cdot|$ denotes the absolute value, and $\arg(\cdot)$, the argument (or angle, in radians) of the complex value $G(j\omega_k)$.

Just a recall of the polar form of complex numbers. Figure 6.10 shows the representation of a complex number in a polar form, where the magnitude is $|z| = r = \sqrt{a^2 + b^2}$ and the argument or phase (angle) $\arg(z) = \angle z = \theta = \tan^{-1}\left(\frac{b}{a}\right)$.

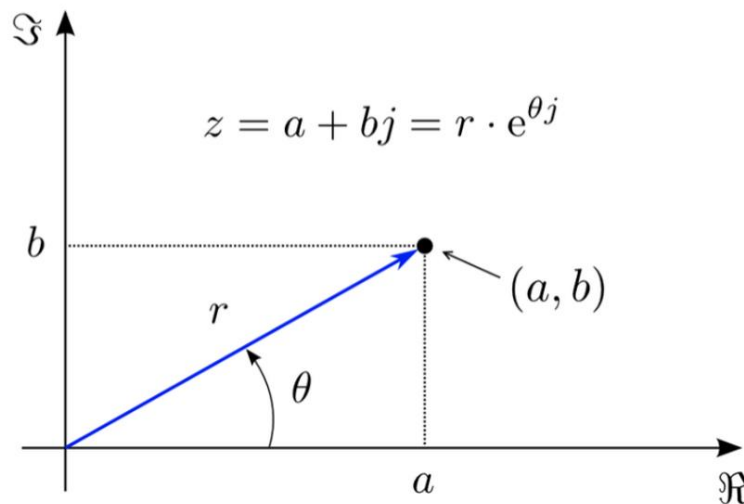


Figure 6.10: Polar coordinates plot of a complex number (axis: real and imaginary parts).

Frequency domain characteristics completely describe the behavior of a *linear, time-invariant system*. Frequency response is graphically represented in the following ways:

- Bode plot: two separate graphs for magnitude and phase against frequency, usually on logarithmic scales;
- Nyquist plot: a single graph depicting real vs. imaginary parts of the response covering the full range frequency;

Since it is possible to assess qualitative properties of the linear system under study (e.g., relative stability margins, i.e., how close the system is to instability), frequency domain analysis is essential in control design.

6.5 Bode plot

The Bode plot is a graph of the absolute value $|G(j\omega)|$ and phase shift $\angle G(j\omega)$ of a transfer function $G(s)$ evaluated in $s = j\omega$. The bode plot shows the system frequency response as a function of frequency ω , for all $\omega > 0$. The frequency axis ω is in logarithmic scale. The absolute value is expressed as decibels (dB),

$$|G(j\omega)|_{dB} = 20 \log_{10} |G(j\omega)|.$$

At the unity gain, when $|G(j\omega)| = 1$, then, $|G(j\omega)|_{dB} = 20 \log_{10} 1 = 0 \text{ dB}$.

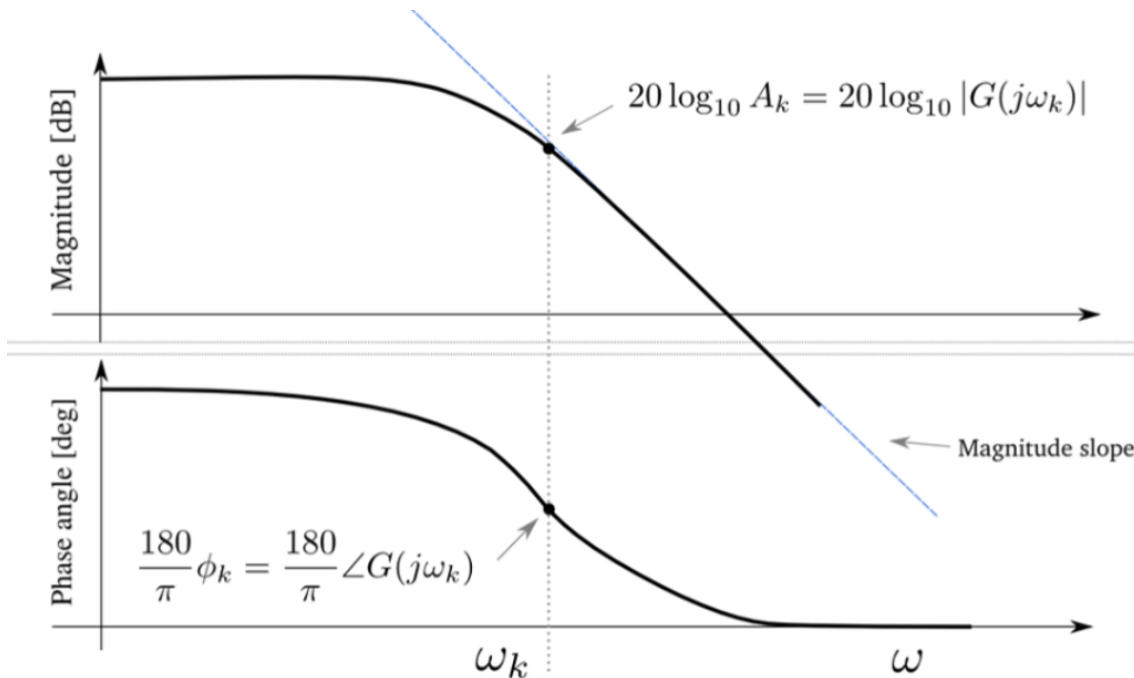


Figure 6.11: Bode magnitude (upper) and phase (bottom) plots.

To analyze the frequency response of a system, it is useful to rewrite the transfer function $G(s)$ in Bode form:

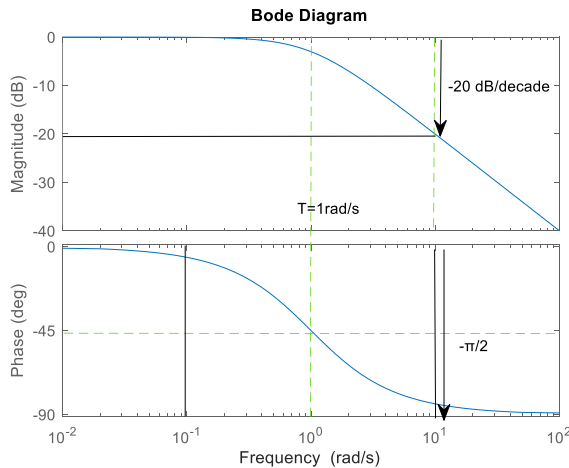
$$G(s) = \frac{K \prod_{i=1}^N (1 + s\tau_i)}{s^h \prod_{j=1}^M (1 + sT_j)} \frac{\prod_{i=1}^P (1 + \frac{2\zeta_i'}{\omega_{ni}} s + \frac{1}{\omega_{ni}^2} s^2)}{\prod_{j=1}^Q (1 + \frac{2\zeta_j''}{\omega_{nj}''} s + \frac{1}{\omega_{nj}''^2} s^2)}$$

where K is the Bode gain; h is the type of system, i.e., the number of poles in $s=0$; T_j (for real numbers $T_j > 0$) is a time constant; ζ_j'' is a damping ratio, with $-1 < \zeta_j'' < 1$; and ω_{nj}'' is a natural frequency of the system. So, the transfer function $G(s)$ of the formula above contains a $N+2P$ zeros (N single real zeros and P double real or complex conjugates zeros) and $h+M+2Q$ poles (h poles at origin, M single real poles, and Q double real or complex conjugates poles).

The influence of the zeros and poles in the bode diagram or plot is shown through some examples in MATLAB.

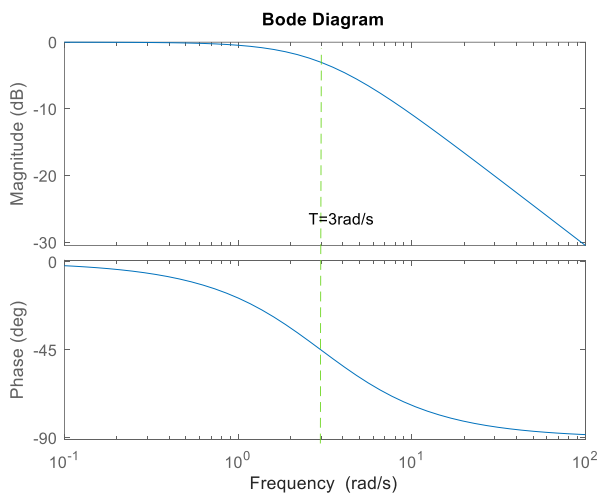
The plot displays the magnitude (in dB) and phase (in degrees) of the system response as a function of frequency. bode automatically determines frequencies to plot based on system dynamics.

Example 6.5: Bode plot of a first order stable system



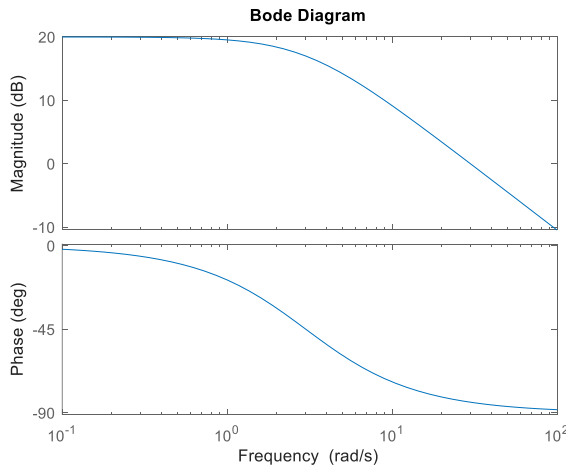
```
clc;
clear all;
close all;
num1 = [ 1 ];
den1 = [ 1 1 ];
disp('Transfer function :- ');
TF1 = tf(num1, den1)
bode(TF1)
```

Figure 6.12: Bode magnitude (upper) and phase (bottom) plots for a first order stable system(left). MATLAB code (right).



```
clc;
clear all;
close all;
num1 = [ 3 ];
den1 = [ 1 3 ];
disp('Transfer function :- ');
TF1 = tf(num1, den1)
bode(TF1)
```

Figure 6.13: Bode magnitude (upper) and phase (bottom) plots for a first order stable system(left). MATLAB code (right).



```

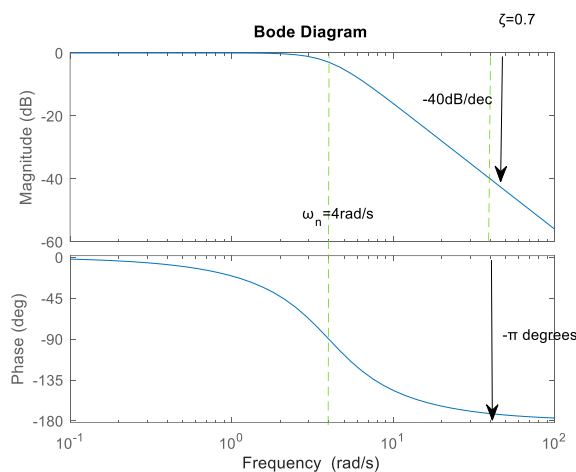
clc;
clear all;
close all;
num1 = [ 30 ];
den1 = [ 1 3 ];
disp ('Transfer function :- ');
TF1 = tf (num1 , den1)
bode (TF1)

```

Figure 6.14: Bode magnitude (upper) and phase (bottom) plots for a first order stable system(left). MATLAB code (right).

The influence of a stable pole in the bode diagram is depicted by the decrease in magnitude of 20 dB (-20dB) decade(from frequency 10^1 rad/s to frequency 10^2 rad/s) and $-\pi/2$ in phase per decade. In this case as, the pole at the frequency 30 rad/s starts the change. Please note the change in magnitude due to the DC-gain, K , according to the logarithm of $20 \log(K)$ when comparing figures 6.13 and 6.14.

Example 6.6: Bode plot of a second order with stable pair of complex poles (underdamped system)



```

clc;
clear all;
close all;
disp ('Transfer function :- ');
H = tf(16,[1 5.6 16])
bode ( H )

```

$$\omega_n = 4$$

Figure 6.15: Bode magnitude (upper) and phase (bottom) plots for a second order stable system with a pair of complex poles (left). MATLAB code (right).

The influence of a stable pole in the bode diagram is depicted by the decrease in magnitude of 40 dB (-40dB) decade (from frequency 10^1 rad/s to frequency 10^2 rad/s) and $-\pi$ degrees in phase per decade at ω_n .

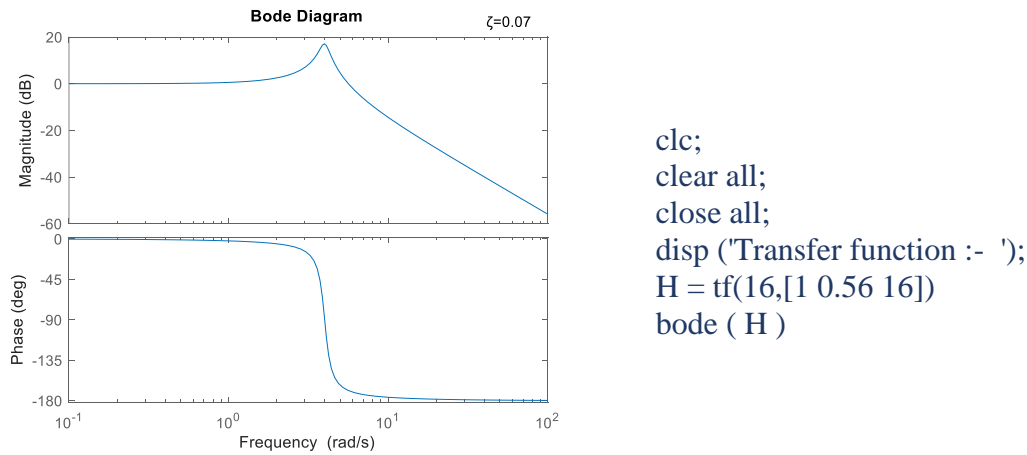


Figure 6.16: Bode magnitude (upper) and phase (bottom) plots for a second order stable system with a pair of complex poles (left). MATLAB code (right).

Table 6.1: Influence of the zeros and poles in the bode diagram.

Type	Condition	Magnitude	Phase
Stable real pole	$T > 0$	-20 dB/dec	$-\pi/2$
Unstable real pole	$T < 0$	-20 dB/dec	$+\pi/2$
Stable real zero	$\tau > 0$	$+20$ dB/dec	$+\pi/2$
Unstable real zero	$\tau < 0$	$+20$ dB/dec	$-\pi/2$
Pair of stable complex poles	$\xi > 0$	-40 dB/dec	$-\pi$
Pair of unstable complex poles	$\xi < 0$	-40 dB/dec	$+\pi$
Pair of stable complex zeros	$\xi' > 0$	$+40$ dB/dec	$+\pi$
Pair of unstable complex zeros	$\xi' < 0$	$+40$ dB/dec	$-\pi$

Sketching the Bode plot is just to get a rough idea of the characteristics of a system, to interpret results, and to detect potential errors obtained from calculations.

6.6 Mention about Nyquist polar plot

The Nyquist plot, or polar plot is the graph in polar coordinates of $G(j\omega)$ for $\omega \in [0, +\infty)$ in the complex plane. The Nyquist plot combines the Bode magnitude and phase plots and therefore provides a more compact representation.

The stability assessment of the Nyquist plot says; for an open-loop asymptotically stable system $G(s)$, the closed-loop system $W(s)$ is asymptotically stable if and only if the Nyquist plot $G(j\omega)$ does not encircle the critical point $-1 + j0$.

Example 6.7: Generate the Bode and Nyquist plots of a first order system, $G(s) = \frac{1}{s+1}$ and sine wave and step responses in MATLAB. Analyse results.

To generate the Bode and Nyquist plots and sine wave and step responses of the system described by the following transfer function, $G(s) = \frac{1}{s+1}$ in the time domain, MATLAB commands `bode(sys)`, `nyquist(sys)`, `y=lsim(sys,u,t)`, and `step(sys)` are used.

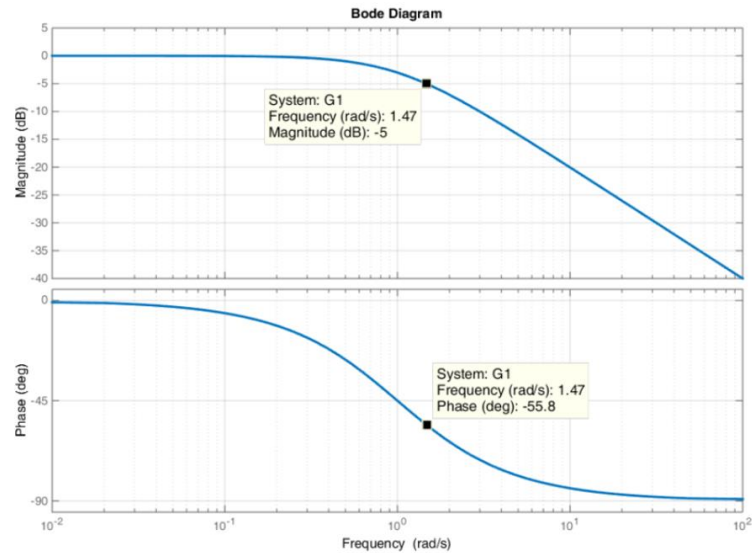


Figure 6.17: Bode magnitude (upper) and phase (bottom) plots of the first order system, $G(s) = \frac{1}{s+1}$.

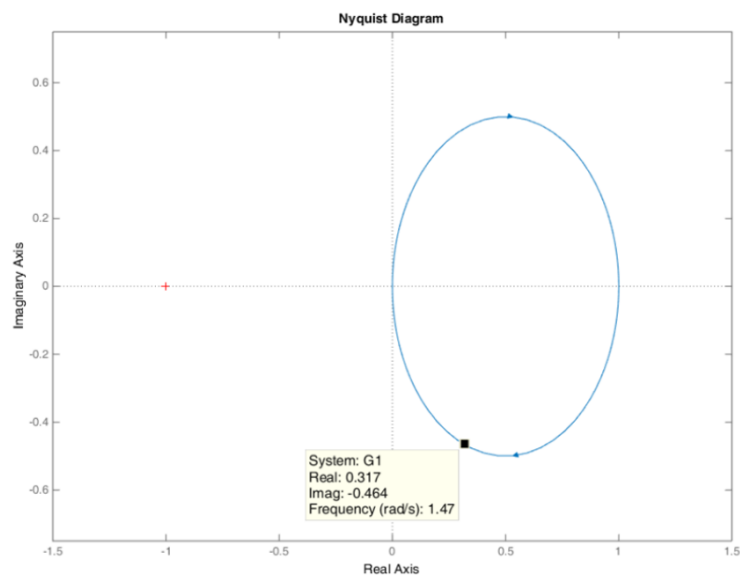


Figure 6.18: Nyquist plot of the first order system, $G(s) = \frac{1}{s+1}$.

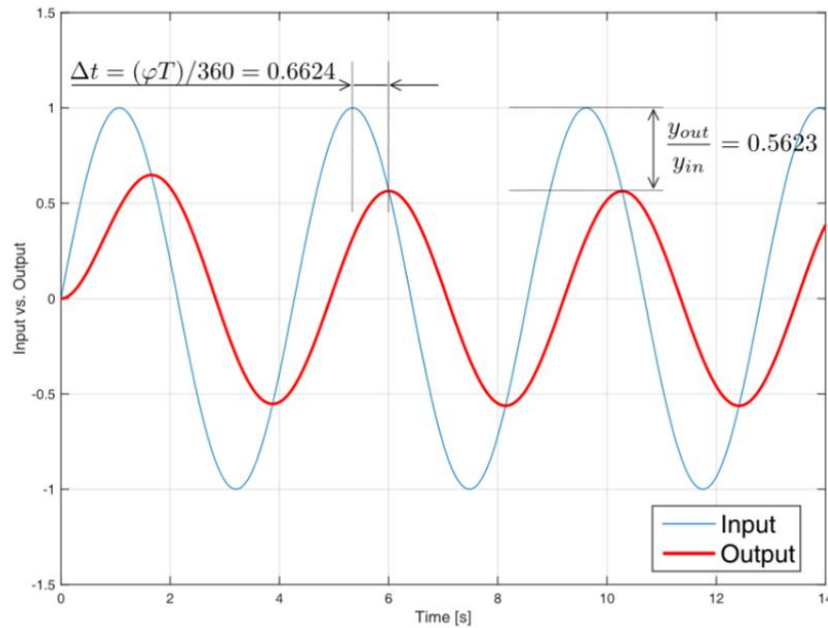


Figure 6.19: Time domain response of the first order system, $G(s) = \frac{1}{s+1}$, against sinusoidal input with $\omega = 1.47 \frac{\text{rad}}{\text{s}}$.

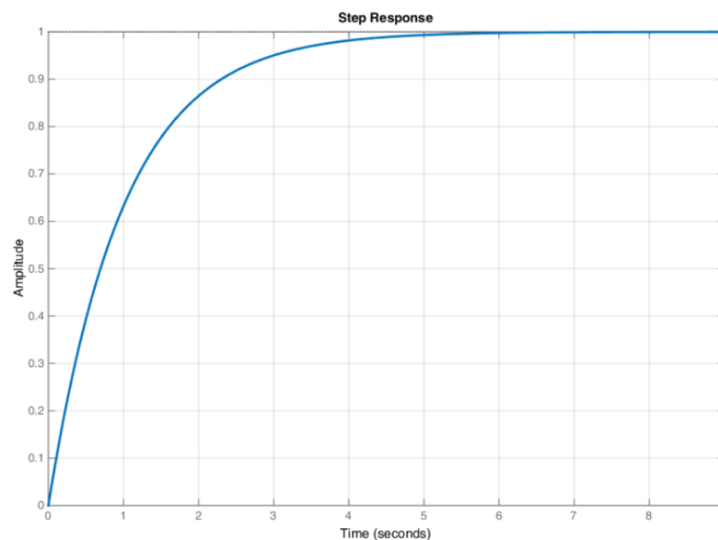


Figure 6.20: Time domain response of the first order system, $G(s) = \frac{1}{s+1}$, against unity step input.

It is left for the reader to analyze the graphs and try to understand what these analyses are intended for.

Example 6.8: Generate the Bode and Nyquist plots and sine wave and step responses of the system described by the second-order system transfer function, $G(s) = \frac{1}{5s^2+s+1}$. Analyse results.

To generate the Bode and Nyquist plots and sine wave and step responses of the system described by the following transfer function, $G(s) = \frac{1}{5s^2+s+1}$ in the time domain, the MATLAB commands `bode(sys)`, `nyquist(sys)`, `y=lsim(sys,u,t)`, and `step(sys)` are used.

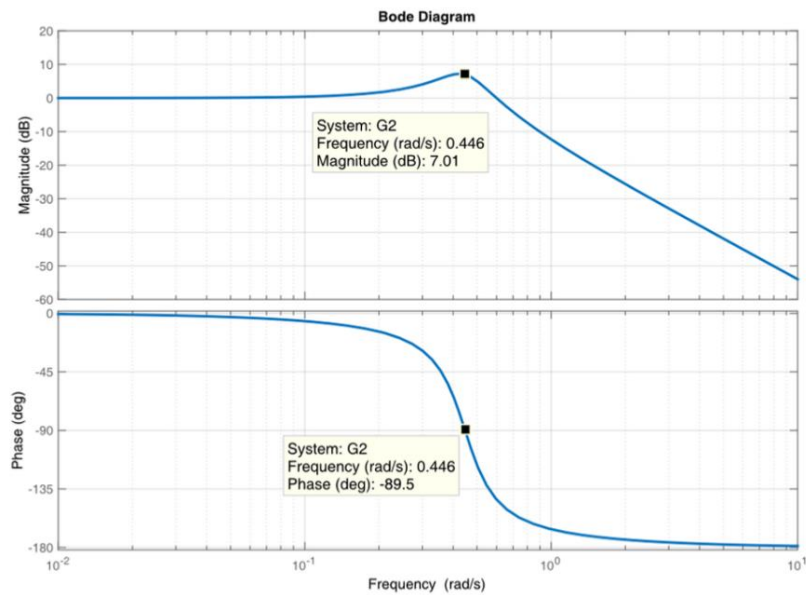


Figure 6.21: Bode magnitude (upper) and phase (bottom) plots of the second order system,

$$G(s) = \frac{1}{5s^2+s+1}.$$

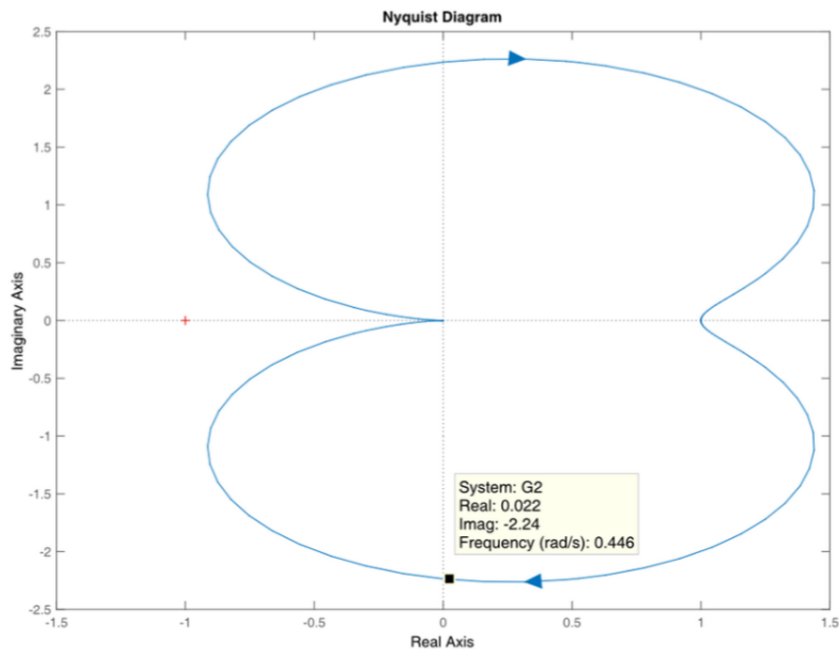


Figure 6.22: Nyquist plot of the second order system, $G(s) = \frac{1}{5s^2+s+1}$.

Sine wave response ($\omega = 0.446 \frac{rad}{s}$)

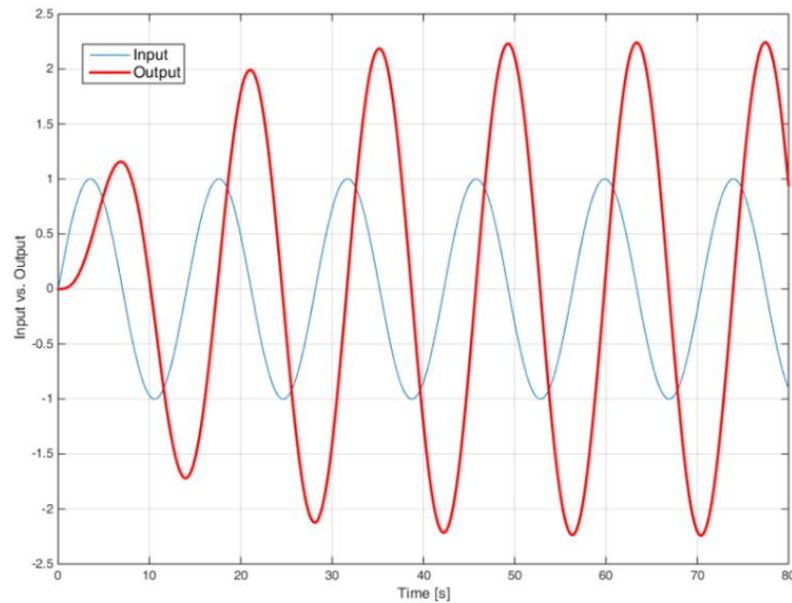


Figure 6.23: Time domain response of the second order system, $G(s) = \frac{1}{5s^2+s+1}$, against sinusoidal input with $\omega = 0.446 \frac{rad}{s}$.

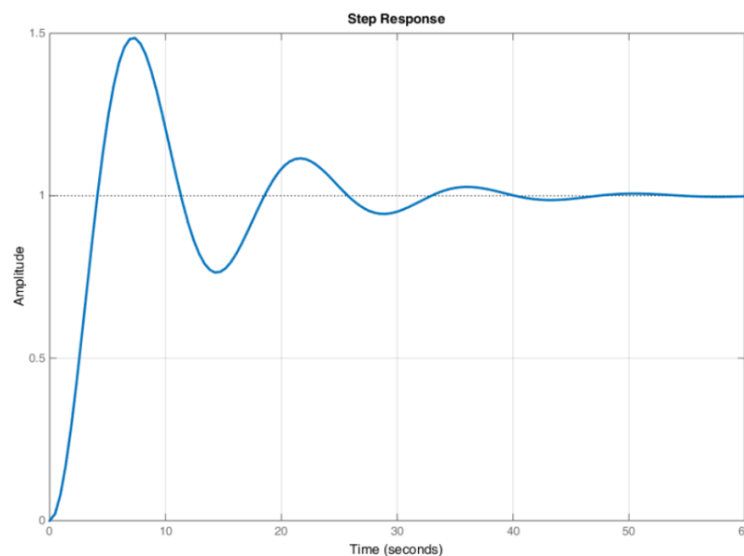


Figure 6.24: Time domain response of the underdamped second order system, $G(s) = \frac{1}{5s^2+s+1}$, against unity step input.

It is left for the reader to analyze the graphs and try to understand what these analyses are intended for.

Exercises:

1. Calculate if the following characteristic equations correspond to a stable system according to the Routh Hurwitz criterion.

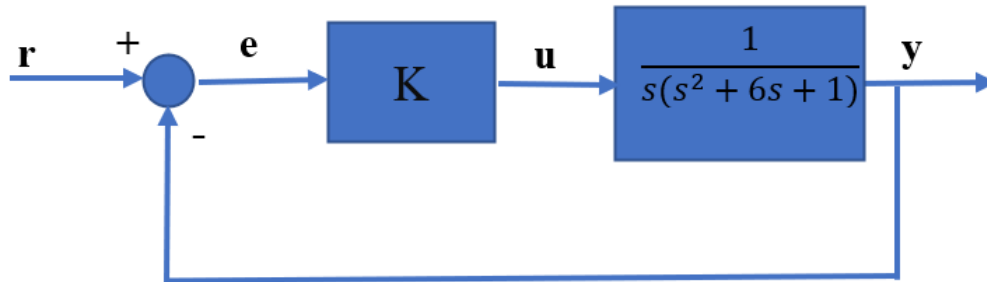
a. $Qa(s) = s^5 + 2s^4 + 2s^3 + 4s^2 + 11s + 10$

b. $Qb(s) = s^4 + s^3 + 3s^2 + 2s + 2$

c. $Qc(s) = s^3 + s^2 + s + 1$

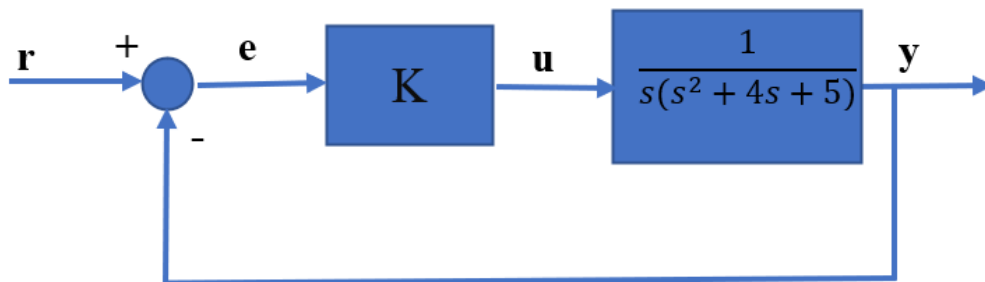
2. Root locus exercises:

a. Calculate the root locus of the following block diagram:



Can the system be stable? Obtain the time domain response for some values of K and comment on the results.

b. Calculate the root locus of the following block diagram:



Study the root locus plot and give adequate values of the gain K justifying the given solution.

References:

[1] A.N. Michel, L. Hou and D. Liu (2007). Stability of Dynamical Systems: Continuous, Discontinuous, and Discrete Systems. Springer Science & Business Media, ISBN: 0817646493, 9780817646493.

[2] Farzad Sagharchi (2022). Routh-Hurwitz stability criterion (<https://www.mathworks.com/matlabcentral/fileexchange/17483-routh-hurwitz-stability-criterion>) . MATLAB Central File Exchange.

[3] Nise, Norman S. Control Systems Engineering. 4th ed. Hoboken, NJ: John Wiley, 2004

Chapter 7: Z-transform and inverse Z-transform

This chapter addresses Z-transform. The Z-transform converts time-domain differential equations and operations into algebraic domains. It facilitates the algebra behind the equations, providing a graphical solution to time-domain differential equations in a parameterized form that represents physical systems. The inverse Z-transform performs the transformation between the algebraic (sequence of numbers) and time domains of the system, providing a bidirectional relationship.

The Laplace transform is a mathematical tool widely used in the analysis and design of linear and time-invariant continuous control systems, which are described by linear and time-invariant differential equations. Analysis of discrete-time LTI systems can be done using Z-transforms. It is a powerful mathematical tool to convert differential equations into algebraic equations. Applied computer-based controllers have the characteristic of using discrete-time signals and controllers based on the Z-transform. Figure 7.1 presents a digital controller for a continuous plant.

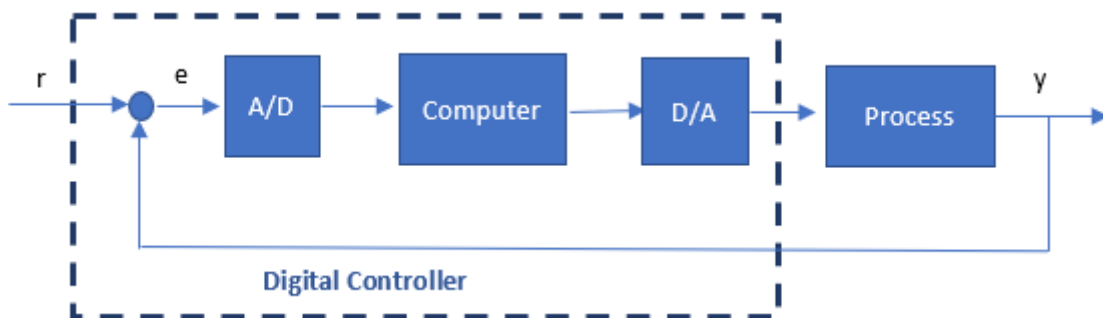


Figure 7.1: Digital controller

7.1 Z-transform

The Z-transform is a series representation that maps the time domain into the Z-domain. Z-domain is a complex domain, also known as complex frequency domain, consisting of a real axis (x-axis) and an imaginary axis (y-axis). A signal is usually defined as a *sequence* of real or complex numbers that is then converted to the Z-domain by the process of z-transformation.

The *two-sided* or *bilateral* Z-transform (ZT) of sequence $x[n]$ is defined as:

$$X(z) = Z\{x[n]\} = \sum_{n=-\infty}^{\infty} x[n] z^{-n} \quad (7.1)$$

From the previous definition, the unilateral (one-sided) Z-transform of a discrete time signal $x(n)$ is given as

$$X(z) = Z(x[KT]) = \sum_{k=0}^{\infty} X[KT]z^{-k} = X(0) + X(T)z^{-1} + X(2T)z^{-2} + X(3T)z^{-3} + \dots \quad (7.2)$$

$n=KT$ represents the discrete values of the function at certain instants given by multiples of the sampling period, T , where the signal is measured in the discrete-time domain. There are regions of the Z -plane where it converges and others where it does not. The power of the variable z indicates the position in time, which is a multiple of T , in which the signal has amplitude $x[KT]$.

The ZT operator transforms the sequence $x[n]$ to $X(z)$, a function of the continuous complex variable z . The relationship between a sequence and its transform is denoted as:

$$x[n] \xleftrightarrow{Z} X(z) \quad (7.3)$$

At this point, it is possible to establish the connection between the discrete-time Fourier transform (DTFT) and the ZT. Despite being aware of the importance of this relationship in engineering, the interested reader is referred to the reference [1] and references therein.

Some of the **properties** and **theorems of the Z-transform**, which are useful in the study of discrete-time control systems, are presented. They are the following:

1. *Unicity*: The Z-transform of a sequence is unique.
2. *Addition*: The Z-transform of the sum of the sequences of numbers is equal to the sum of the sum of the Z-transform of the sequences, $Z[x_1(k) + x_2(k)] = X_1(z) + X_2(z)$ where $Z[x_1(k)] = X_1(z)$ and $Z[x_2(k)] = X_2(z)$.
3. *Multiplication by a constant*: Z-transform of a sequence of numbers by a constant is equal to the Z-transform of the sequence multiplied by a constant: $Z[a * x(k)] = a * Z[x(k)]$.
4. *Linearity*: From properties 1 and 2, it is obtained: $Z[a * x_1(k) + b * x_2(k)] = a * X_1(z) + b * X_2(z)$ where $Z[x_1(k)] = X_1(z)$ and $Z[x_2(k)] = X_2(z)$.
5. *Time shifting or delay*: If $x(k)=0$ for $k<0$, $X(z)=Z[x(k)]$, and being n , a positive integer, then $Z[x(k-n)] = z^{-n}X(z)$ and $Z[x(k+n)] = z^n[X(z) - \sum_{k=0}^{n-1} x(k)z^{-k}]$ where the sum term is considered in the initial conditions.

From the equations:

$$Z[x(k+1)] = z X(z) - z x(0);$$

$$Z[x(k+2)] = z^2 X(z) - z^2 x(0) - z x(1);$$

$$Z[x(k+3)] = z^3 X(z) - z^3 x(0) - z^2 x(1) - z x(2);$$

multiplication of $X(z)$ by z^{-1} has the effect of delaying the signal $x(k)$ one sampling time and the multiplication of $X(z)$ by z that has the effect of moving forward the signal $x(k)$ by one sampling time.

6. *Time scaling by a complex exponential sequence*: being $X(z)=Z[x(t)]$, then the Z-transform of $y(t)=a^k x(t)$ is $Y(z)=Z[y(t)] = Z[a^k x(t)]=X(a^{-1}z)$.

Example 7.1: Calculate the Z-transform of $y(t)=a^k t$.

$$\text{Sol.: } Z(t) = \frac{Tz}{(z-1)^2}$$

$$\text{Then, } Z(a^k t) = \frac{T a^{-1} z}{(a^{-1} z - 1)^2} = \frac{T a z}{(z - a)^2} .$$

7. Differentiation in Z-Domain: The differentiation in the z-domain property of the Z-transform states that the multiplication by n in the time domain corresponds to the differentiation in the z-domain. This property is also called the multiplication by n property of the Z-transform. Therefore, if

$$x[n] \xleftrightarrow{z} X(z); ROC = R$$

then, according to the differentiation in the z-domain property,

$$nx[n] \xleftrightarrow{z} -z \frac{dX(z)}{dz}; ROC = R$$

7.2 The Region of Convergence (ROC) of the Z-Transform

The range of variation of z for which the Z-transform converges is called the region of convergence of the Z-transform: $X(z) = \sum_{n=-\infty}^{\infty} x(n)z^{-n}$.

The Z-transform has two parts, which are the expression and the Region of Convergence (ROC), respectively. Whether the Z-transform $X(z)$ of a signal $x(n)$ exists or not depends on the complex variable “z” as well as the signal itself. All complex values of “ $z = \text{Re}(j\omega)$ ” for which the summation in the definition converges form a region of convergence (ROC) in the Z-plane. A circle with $r = 1$ is called a unit circle, and the complex variable in the Z-plane is represented as shown below in figure 7.2. Please note that this is a simplified representation of the complex plane.

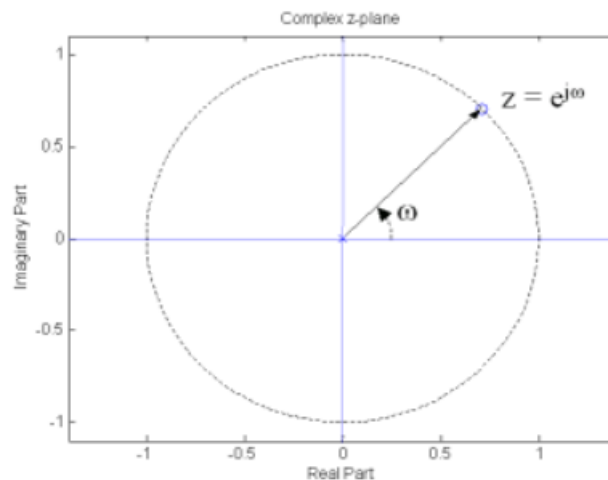


Figure 7.2: Complex variable in the Z-plane.

The concept of ROC can be understood easily by finding the Z-transform of two functions given below:

- a) $x(n) = a^n u(n)$ where $u(n)$ is the step function for $n \geq 0$

$$X(z) = \sum_{n=-\infty}^{\infty} a^n u(n) z^{-n} = \sum_{n=0}^{\infty} a^n z^{-n} = \sum_{n=0}^{\infty} (az^{-1})^n$$

For convergence of $X(z)$, it is required that $\sum_{n=0}^{\infty} |az^{-1}|^n < \infty$. Consequently, the region of convergence is that range of values of z for which $|az^{-1}| < 1$, or equivalently, $z > a$, and it is shown in Figure 7.3.

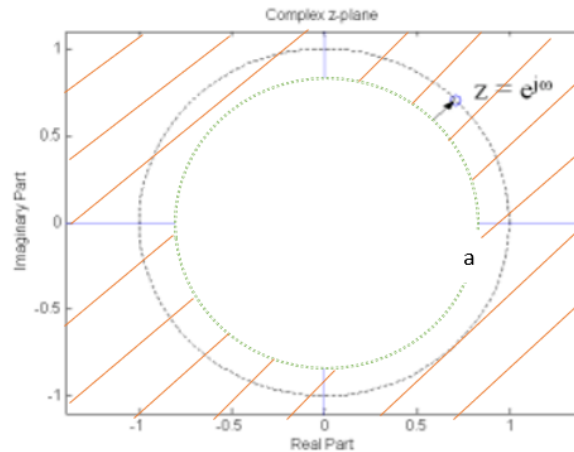


Figure 7.3: Z-plane for case a).

$$\text{Then, } X(z) = \sum_{n=0}^{\infty} (az^{-1})^n = \frac{1}{1-az^{-1}} = \frac{z}{z-a}$$

$$\text{b) } x(n) = -a^n u(-n-1)$$

$$\begin{aligned} X(z) &= \sum_{n=-\infty}^{\infty} -a^n u(-n-1) z^n = \sum_{n=-\infty}^{-1} a^n z^{-n} = - \sum_{n=1}^{\infty} (a^{-1}z)^n \\ &= 1 - \sum_{n=0}^{\infty} (a^{-1}z)^n = 1 - \frac{1}{1-a^{-1}z} = \frac{1}{1-az^{-1}} = \frac{z}{z-a} \end{aligned}$$

This result converges only when $a^{-1}z < 1$, or equivalently, $|z| < |a|$. The ROC is shown below:

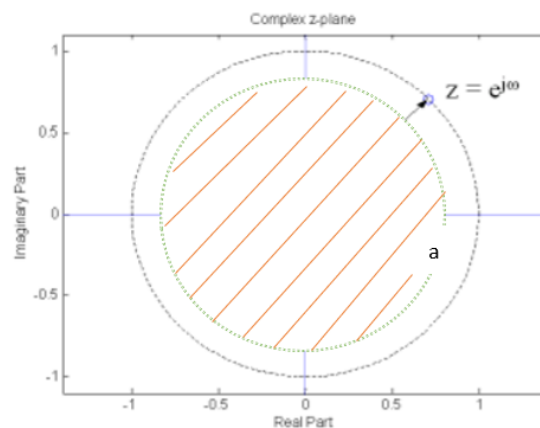


Figure 7.4: Z-plane for case b).

If it is considered the signals $a^n u(n)$ and $-a^n u(-n-1)$, it can be noticed that although the signals are differing, their Z-transforms are identical, which is $z/z-a$. Thus, it is possible to determine that to distinguish the Z-transforms uniquely, their ROC's must be specified.

Properties of ROC of Z-Transforms

- ROC of Z-Transform is indicated with a circle in the z-plane.
- The ROC does not contain any poles.
- If $x(n)$ is a finite distance causal sequence or right sided sequence, then the ROC is the entire z-plane except at $z = 0$.
- If $x(n)$ is a finite distance anti-causal sequence or left sided sequence, then the ROC is the entire z-plane except at $z = \infty$.
- If $x(n)$ is an infinite distance causal sequence, ROC is the exterior of the circle with radius a . i.e., $|z| > a$.
- If $x(n)$ is an infinite distance anti-causal sequence, ROC is the interior of the circle with radius a . i.e., $|z| < a$.
- If $x(n)$ is a finite distance two-sided sequence, then the ROC is the entire z-plane except at $z = 0$ & $z = \infty$.

The concept of the ROC can be explained by the following example:

Example 7.2: Find z-transform and ROC of the sequence $a^n u[n] + a^{-n} u[-n-1]$

$$Z[a^n u[n] + a^{-n} u[-n-1]] = Z[a^n u[n]] + Z[a^{-n} u[-n-1]] = \frac{1}{1-az^{-1}} + \frac{1}{1-az^{-1}} \quad (\text{check Table 7.1 below})$$

$$\text{ROC: } |z| > a \text{ and, ROC: } |z| < 1/a$$

The plot of the ROC has two conditions as $a > 1$ and $a < 1$ (or $1/a > 1$), as you do not know a .

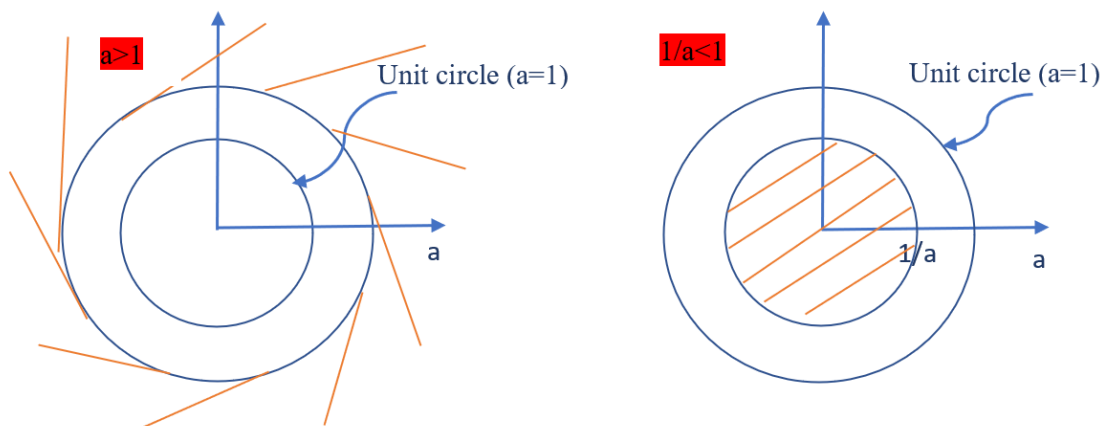


Figure 7.5: z-plane for case a), where the value of $a > 1$.

In this case, there is no ROC combination.

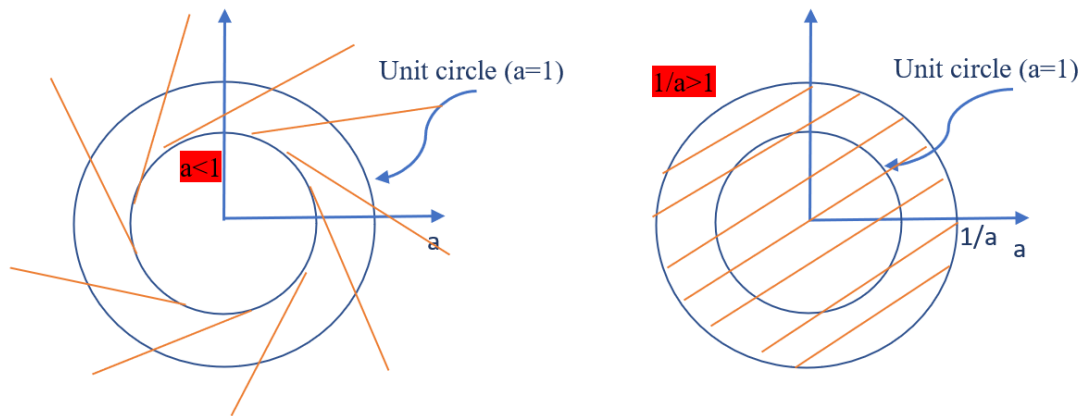


Figure 7.6: z-plane for case b), where the value of $a < 1$.

Here, the ROC combination is of the form: $a < |z| < 1/a$ and $a < |z| < 1/a$

Hence, for this problem, the Z-transform is possible when $a < 1$.

7.3 Discrete-time signals

Discrete-time signals are represented mathematically as sequences of numbers. A sequence of numbers x , in which the n th number in the sequence is denoted $x[n]$, is formally written as $x = \{x[n]\}$, $-\infty < n < \infty$, (7.4), where n is an integer. In a practical setting, such sequences can often arise from periodic sampling of an analog (i.e., continuous time) signal $x_a(t)$. In that case, the numeric value of the n th number in the sequence is equal to the value of the analog signal, $x_a(t)$, at time nT :

$$\text{i.e., } x[n] = x_a(nT), \quad -\infty < n < \infty. \quad (7.5)$$

The quantity T is the sampling period, and its reciprocal is the sampling frequency. Although sequences do not always arise from sampling analog waveforms, it is convenient to note that it is used $[]$ to enclose the independent variable of discrete-variable functions, and it is used $()$ to enclose the independent variable of continuous-variable functions to refer to $x[n]$ as the “ n th sample” of the sequence. Also, although, strictly speaking, $x[n]$ denotes the n th number in the sequence, the notation of Eq. (7.4) is often unnecessarily cumbersome, and it is convenient and unambiguous to refer to “the sequence $x[n]$ ” when that means the entire sequence, just as it is referred to “the analog signal $x(t)$.”

Discrete-time signals (i.e., sequences) are depicted graphically, as shown in Figure 7.7. Although the abscissa is drawn as a continuous line, it is important to recognize that $x[n]$ is defined only for integer values of n . It is not correct to think of $x[n]$ as being zero when n is not an integer; $x[n]$ is simply undefined for non-integer values of n .

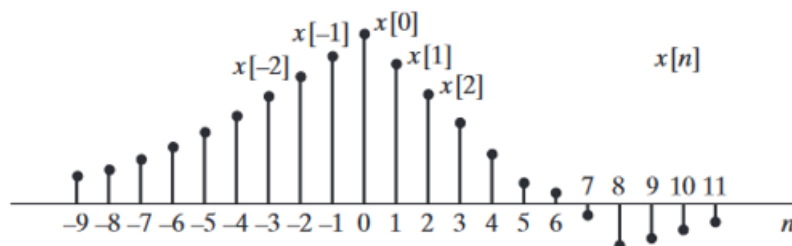


Figure 7.7: Graphic representation of a discrete-time signal

In discussing the theory of discrete-time signals and systems, several basic sequences are of particular importance. These sequences are shown in Figure 7.7 and will be discussed next.

The unit sample sequence (Figure 7.8a) is defined as the sequence:

$$\delta[n] = \begin{cases} 0, & n \neq 0 \\ 1, & n = 0 \end{cases} \quad (7.6)$$

The unit sample sequence plays the same role for discrete-time signals and systems that the unit impulse function (Dirac delta function) does for continuous-time signals and systems. For convenience, the unit sample sequence is referred to as a discrete time impulse or simply as an impulse. It is important to note that a discrete-time impulse does not suffer from the mathematic complications of a continuous-time impulse; its definition given in Eq. (7.6) is simple and precise.

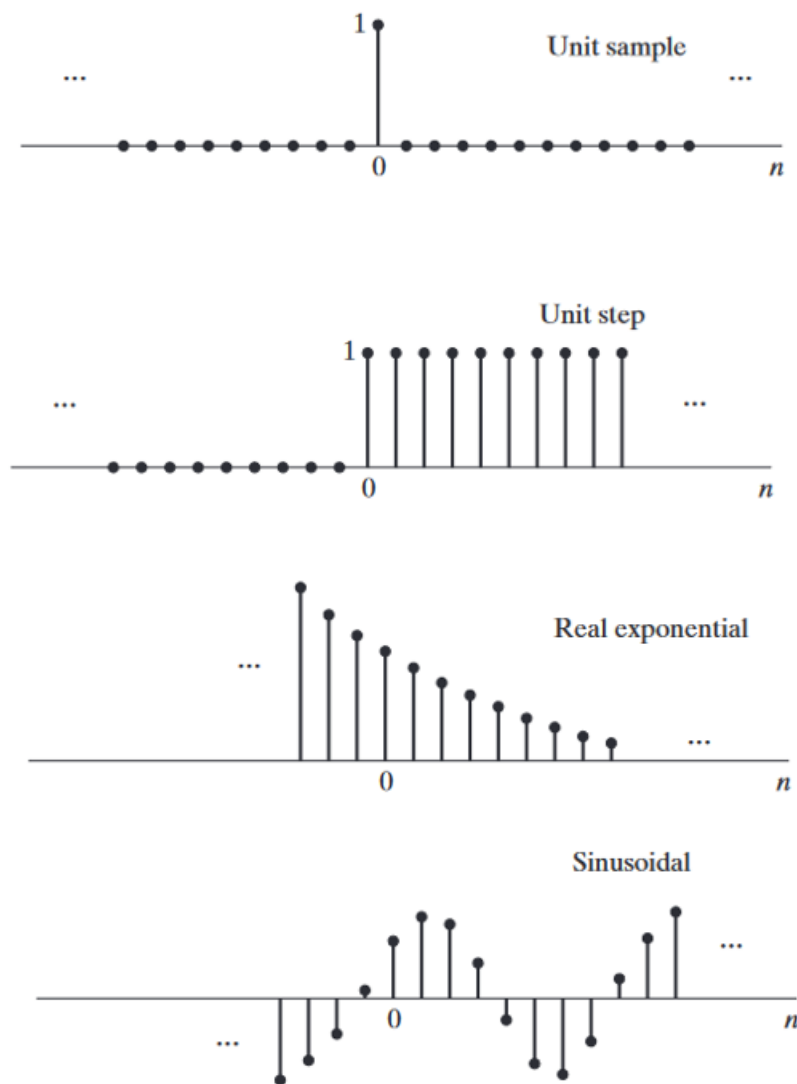


Figure 7.8: Some basic sequences: a) unit sample, b) unit step, c) real exponential, and d) sinusoidal. The sequences shown play important roles in the analysis and representation of discrete-time signals and systems.

One of the important aspects of the impulse sequence is that an arbitrary sequence can be represented as a sum of scaled, delayed impulses. For example, the sequence $p[n]$ in Figure 7.9 can be expressed as:

$$p[n] = a_{-3}\delta[n + 3] + a_1\delta[n - 1] + a_2\delta[n - 2] + a_7\delta[n - 7] \quad (7.7).$$

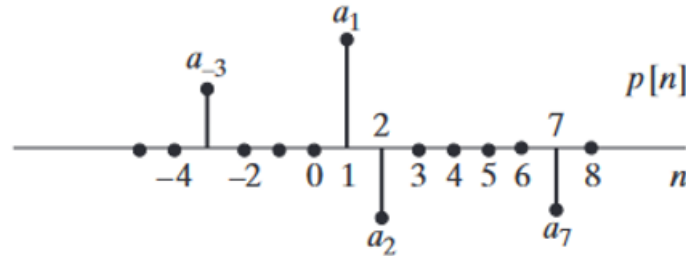


Figure 7.9: Example of a sequence to be represented as a sum of scaled, delayed impulses.

More generally, any sequence can be expressed as:

$$x[n] = \sum_{k=-\infty}^{\infty} x[k]\delta[n - k] \quad (7.8).$$

We will make specific use of Eq. (7.8) in discussing the representation of discrete time linear systems.

The unit step sequence (Figure 7.8b) is defined as:

$$u[n] = \begin{cases} 1, & n \geq 0 \\ 0, & n < 0 \end{cases} \quad (7.9).$$

The unit step is related to the unit impulse by:

$$u[n] = \sum_{k=-\infty}^n \delta[k] \quad (7.10)$$

that is, the value of the unit step sequence at (time) index n is equal to the accumulated sum of the value at index n and all previous values of the impulse sequence. An alternative representation of the unit step in terms of the impulse is obtained by interpreting the unit step in Figure 7.8b in terms of a sum of delayed impulses, as in Eq. (7.8). In this case, the nonzero values are all unity, so

$$u[n] = \delta[n] + \delta[n - 1] + \delta[n - 2] + \cdots \quad (7.11)$$

or,

$$u[n] = \sum_{k=0}^{\infty} \delta[n - k] \quad (7.12)$$

Yet another alternative, the impulse sequence can be expressed as the first backward difference of the unit step sequence, i.e.,

$$\delta[n] = u[n] - u[n - 1] \quad (7.13).$$

7.4 Z-Transform Table of Basic Signals

Table 7.1: Z-Transform Table of Basic Sequences.

$x(t)$	$X[z]$	ROC
$\delta [n]$	1	All z
$u(n)$	$\frac{1}{1 - z^{-1}}$	$ z > 1$
$-u(-n-1)$	$\frac{1}{1 - z^{-1}}$	$ z < 1$
$\delta(n-m)$	z^{-m}	All z except 0 or ∞
$a^n u[n]$	$\frac{1}{1 - az^{-1}}$	$ z > a$
$-a^n u[-n-1]$	$\frac{1}{1 - az^{-1}}$	$ z < a$
$na^n u[n]$	$\frac{az^{-1}}{(1 - az^{-1})^2}$	$ z > a$
$-na^n u[-n-1]$	$\frac{az^{-1}}{(1 - az^{-1})^2}$	$ z < a$
$a^n \cos(\omega_0 n) u[n]$	$\frac{1 - az^{-1} \cos \omega_0}{1 - 2az^{-1} \cos \omega_0 + a^2 z^{-2}}$	$ z > a$
$a^n \sin(\omega_0 n) u[n]$	$\frac{az^{-1} \sin \omega_0}{1 - 2az^{-1} \cos \omega_0 + a^2 z^{-2}}$	$ z > a$

7.5 Inverse Z-transform methods

If it is required to analyze a system, which is already represented in frequency domain, as discrete time signal, then, analogous to the Laplace transform, it is introduced the inverse Z-transformation.

Mathematically, it can be represented as:

$$x(n) = Z^{-1} [X(z)] \quad (7.14)$$

where $x(n)$ is the signal in the time domain and $X(z)$ is the signal in the frequency domain. Note that the Z-inverse transform of a signal $X(z)$ is obtained in the sampling times $n=kT$, $k=0, 1, 2, \dots$

Methods to Find Inverse Z-Transform

When the analysis is needed in discrete format, we convert the frequency domain signal back into discrete format through inverse Z-transformation. The following four ways to determine the inverse Z-transformation are considered:

- Long Division Method,
- Partial Fraction Expansion Method,
- Residue or Contour Integral Method,

a) Long Division Method

In this method, the Z-transform of the signal $x(z)$ can be represented as the ratio of polynomials:

$$x(z)=N(z)/D(z) \quad (7.15).$$

Dividing the numerator by the denominator, a series of polynomials is obtained:

$$X(z)=x(0)+x(1)z^{-1}+x(2)z^{-2}+\dots \quad (7.16).$$

The above sequence represents the series of the inverse Z-transforms of the given signal for $n \geq 0$ and the above system is causal.

However, for $n < 0$ the series can be written as:

$$x(z)=x(-1)z^1+x(-2)z^2+x(-3)z^3+\dots \quad (7.17).$$

Example 7.3: Calculate the inverse Z-transform of $X(z) = \frac{z}{z^2-3z+2}$. Sol: $X(z) = \frac{z}{z^2-3z+2} = z^{-1} + 3z^{-2} + 7z^{-3} + 15z^{-4} + \dots$. So, $x(0) = 0$, $x(1) = 1$, $x(2) = 3$, $x(3) = 7$, $x(4) = 15$, ..., $x(kT) = 2^k - 1$. **Not always is it possible to find a closed form of the series.**

In MATLAB, `deconv([1,0,0,0,0,0,0,0], [1,-3,2])`, the number of zeros depends on the number of the element it is required to calculate. In general, this method is adequate if it is expected to calculate the first values of the series numbers, as they don't have a closed form.

b) Partial Fraction Expansion Method

In this case, the signal is also expressed first in $N(z)/D(z)$ form. If it is a rational fraction, it will be represented as follows:

$$x(z)=(b_0+b_1z^{-1}+b_2z^{-2}+\dots+b_mz^{-m})/(a_0+a_1z^{-1}+a_2z^{-2}+\dots+a_nz^{-n}) \quad (7.18).$$

The above one is improper when $m < n$ and $a_n \neq 0$.

If the ratio is not proper, i.e., improper, then it must be converted to the proper form to solve it.

Example 7.4: Calculate the inverse Z-transform of $X(z) = \frac{z}{z^2-3z+2}$. $X(z) = \frac{z}{z^2-3z+2} = \frac{-1}{z-1} + \frac{2}{z-2}$, using the table of the Z-transform, $x(kT) = -1 + 2^k = 2^k - 1$

c) The Residue or Contour Integral Method

In this method, the inverse Z-transform $x[n]$ is obtained by summing residues of $[x(z)Z^{n-1}]$ at all poles. Mathematically, this may be expressed as:

$$x[n] = \sum_{\text{all poles}} \text{Res} (X(z)x^{n-1}) \quad (7.19).$$

corresponding to the poles $X(z)x^{n-1}$ that lie inside C that encloses $z = |R|$.

The residue of $X(z)x^{n-1}$ at a given pole, $z = z_i$ with multiplicity, m, can be calculated as:

$$\text{Res}_{z=z_i} = \frac{d^{m-1}}{dz^{m-1}} \left[\frac{(z-z_i)^m}{(m-1)!} X(z)x^{n-1} \right] \Big|_{z=z_i} \quad (7.20).$$

Example 7.5: Calculate the inverse Z-transform of the function $X(z) = \frac{z}{(z-1)(z-2)}$

Solution: First, it is calculated the poles of the function $X(z)z^{k-1}$:

$$X(z)z^{k-1} = \frac{z * z^{k-1}}{(z-1)(z-2)} = \frac{z^k}{(z-1)(z-2)}$$

It can be shown that for every value of $k=0,1,2,3,\dots$, the function $X(z)z^{k-1}$ has two single poles in $z=1$ and $z=2$, so the calculus of the residues may be done for all the values of k at the same time. Then, $x(k)=k_1+k_2$, where:

$$k_1 = \text{residue of } X(z)z^{k-1} \text{ in } z=1 = (z-1) \frac{z^k}{(z-1)(z-2)} \Big|_{z=1} = \frac{1^k}{1-2} = -1$$

$$k_2 = \text{residue of } X(z)z^{k-1} \text{ in } z=2 = (z-2) \frac{z^k}{(z-1)(z-2)} \Big|_{z=2} = \frac{2^k}{2-1} = 2^k$$

Finally,

$$x(k) = -1 + 2^k \quad k=0,1,2,\dots$$

Example 7.6 Calculate the inverse Z-transform of the function $X(z) = \frac{1}{z(z-e^{-2T})}$

Solution: First, it is calculated the poles of the function $X(z)z^{k-1}$

$$X(z)z^{k-1} = \frac{z^{k-1}}{z(z-e^{-2T})} = \frac{z^{k-2}}{(z-e^{-2T})}$$

As it may be observed, for $k=0$ the function $X(z)z^{k-1}$ has a simple pole in $z = e^{-2T}$ and a double pole in $z=0$; for $k=1$ the function $X(z)z^{k-1}$ has a simple pole in $z = e^{-2T}$ and a simple pole in $z=0$; and for $k \geq 2$ the function $X(z)z^{k-1}$ has a simple pole in $z = e^{-2T}$. As a result, it must be calculated residues in $k=0$, $k=1$ and $k \geq 2$ independently.

1. For $k=0$, $X(z)z^{k-1} = \frac{1}{z^2(z-e^{-2T})}$, so, $x(k) = k_{01} + k_{02}$

$$k_{01} = \text{residue of } X(z)z^{k-1} \text{ in } z = e^{-2T} = (z - e^{-2T}) \frac{1}{z^2(z - e^{-2T})} \Big|_{z=e^{-2T}} = \frac{1}{e^{-4T}} = e^{4T}$$

$$k_{02} = \text{residue of } X(z)z^{k-1} \text{ in the double pole } z = 0 = \frac{1}{(2-1)!} \frac{d^{2-1}}{dz^{2-1}} \left[z^2 \frac{1}{z^2(z - e^{-2T})} \right]_{z=0} = \frac{d}{dz} \left[\frac{1}{(z - e^{-2T})} \right]_{z=0} = \frac{-1}{(z - e^{-2T})^2} \Big|_{z=0} = \frac{-1}{(-e^{-2T})^2} = -e^{4T}$$

$$\text{Then, } x(0) = k_{01} + k_{02} = e^{4T} - e^{4T} = 0$$

2. For $k=1$, $X(z)z^{k-1} = \frac{1}{z(z - e^{-2T})}$, so, $x(k) = k_{11} + k_{12}$

$$k_{11} = \text{residue of } X(z)z^{k-1} \text{ in } z = e^{-2T} = (z - e^{-2T}) \frac{1}{z(z - e^{-2T})} \Big|_{z=e^{-2T}} = \frac{1}{e^{-2T}} = e^{2T}$$

$$k_{12} = \text{residue of } X(z)z^{k-1} \text{ in } z = 0 = z \frac{1}{z(z - e^{-2T})} \Big|_{z=0} = \frac{1}{e^{-2T}} = e^{2T}$$

$$\text{Then, } x(0) = k_{01} + k_{02} = e^{2T} - e^{2T} = 0$$

3. For $k \geq 2$, $X(z)z^{k-1} = \frac{z^{k-1}}{(z - e^{-2T})}$, so, $x(k) = k_1$

$$k_{01} = \text{residue of } X(z)z^{k-1} \text{ in } z = e^{-2T} = (z - e^{-2T}) \frac{z^{k-2}}{(z - e^{-2T})} \Big|_{z=e^{-2T}} = z^{k-2} \Big|_{z=e^{-2T}} = e^{-2T(k-2)}$$

$$\text{Then, } x(k) = k_1 = e^{-2T(k-2)} \quad k \geq 2$$

The total solution will be,

$$x(k) = \begin{cases} 0 & k = 0, 1 \\ e^{-2T(k-2)} & k \geq 2 \end{cases}$$

7.4 The concept of a discrete-time system. From difference equations to transfer functions.

A LTI system is completely characterized by its impulse response $h[n]$, or equivalently, the Z-transform of the impulse response $H(z)$ which is called the **transfer function**. Remember:

$$y[n] = x[n] * h[n] \xrightarrow{Z} Y(z) = X(z)H(z) \quad (7.20).$$

Note that it is obtained:

$$H(z) = \frac{Y(z)}{X(z)} \quad (7.21).$$

In case the impulse response is given to define the LTI system, it can be simply calculated the Z-transform to obtain $H(z)$, often called the **transfer function** of the system.

If the system is defined by a difference equation, it could be first calculated the impulse response and then calculate the Z-transform. But it is far easier to calculate the Z-transform of both sides of the difference equation.

Example 7.7: As a first example, it is considered the difference equation:

$$y[n] = x[n] + \alpha y[n - 1]$$

applying the Z-transform to both sides of the equation:

$$Z\{y[n]\} = Z\{x[n] + \alpha y[n - 1]\}$$

$$Y(z) = Z\{x[n]\} + \alpha Z\{y[n - 1]\}$$

$$Y(z) = X(z) + \alpha z^{-1}Y(z)$$

$$(1 - \alpha z^{-1})Y(z) = X(z)$$

$$H(z) = \frac{Y(z)}{X(z)} = \frac{1}{(1 - \alpha z^{-1})}$$

showing that $H(z)$ can be directly calculated from the difference equation without explicit construction of the (infinite length) impulse response function.

Example 7.8: Consider the following difference equation and calculate the transfer function.

$$y[n] = 1.5y[n-1] - 0.5y[n-2] + 0.5x[n]$$

Solution:

$$Y(z) = 1.5z^{-1}Y(z) - 0.5z^{-2}Y(z) + 0.5X(z)$$

$$H(z) = \frac{Y(z)}{X(z)} = \frac{0.5}{1 - 1.5z^{-1} + 0.5z^{-2}} = \frac{z^2}{2z^2 - 3z + 1}$$

7.6.1 Pulse transfer function

The transfer function of a continuous system relates the Laplace transform of the output with the Laplace transform of the input. Similarly, the pulse transfer function relates a sequence of samples at the output of a system to the sequence producing it. The input and output sequences are represented here as a discrete transfer function that relates the Z-transform of the output with the Z-transform of the input at sampling times.

The pulse transfer function of a discrete-time system is defined as the Z-transform of the output between the Z-transform of the input, taking initial conditions as nulls.

$$G(z) = \left. \frac{Y(z)}{R(z)} \right|_{\text{null initial conditions}} \quad (7.22).$$

Example 7.9: Calculate the pulse transfer function for the system described as the following difference equation:

$$y(k+3) = -0.3y(k+2) + y(k+1) - 0.5y(k) + u(k+3) - u(k+1) - 0.6u(k)$$

Solution: Calculating the Z-transform of the difference equation for null initial conditions:

$$\begin{aligned} (z^3 + 0.3z^2 - z + 0.5)Y(z) &= (4z^3 - z - 0.6)R(z) \Rightarrow G(z) = \frac{Y(z)}{R(z)} \\ &= \frac{4z^3 - z - 0.6}{z^3 + 0.3z^2 - z + 0.5} \end{aligned}$$

Example 7.10: Calculate the output of the discrete time system, whose pulse transfer function is $G(z) = \frac{1}{z-0.5}$

Solution: According to Table 7.1 the Z-transform of the unit step function is $R(z) = \frac{z}{z-1}$.

So, the Z-transform of the output will be $Y(z) = G(z)R(z) = \frac{1}{z-0.5} \frac{z}{z-1} = \frac{z}{(z-0.5)(z-1)} = \frac{-2z}{z-0.5} + \frac{2z}{z-1}$.

Calculating the anti-transform Z: $Y(z) \frac{-2z}{z-0.5} + \frac{2z}{z-1} \xRightarrow{Z^{-1}} y(kT) = -2 * (0.5)^k + 2 * 1^k = -2 * (0.5)^k + 2$

7.6.2 Transfer function: poles and zeros

The generic difference equation for an LTI system is:

$$a_0 y[k] + a_1 y[k-1] + \dots + a_n y[k-n] = b_0 x[k] + b_1 x[k-1] + \dots + b_n x[k-n] \quad (7.22).$$

It is taken that the number of terms is both equal to n. In practice, some of the a_i and b_i terms can be zero.

Taking the Z-transform on both sides of the equation and rearranging terms leads to the transfer function H(z):

$$H(z) = \frac{Y(z)}{X(z)} = \frac{b_0 + b_1 z^{-1} + \dots + b_n z^{-n}}{a_0 + a_1 z^{-1} + \dots + a_n z^{-n}} \quad (7.23).$$

Multiplying by z^n/z^n results in

$$H(z) = \frac{Y(z)}{X(z)} = \frac{b_0 z^n + b_1 z^{n-1} + \dots + b_n}{a_0 z^n + a_1 z^{n-1} + \dots + a_n} \quad (7.24).$$

The fundamental theorem of algebra, which states that every n^{th} order complex polynomial has exactly n roots. The roots of the nominator are called the **zeros** of the transfer function, and they are indicated with q_i , and the roots of the denominator are called the **poles** of H, and they are indicated as p_i . With these zeros and poles, it can be rewritten the transfer function as:

$$H(z) = \frac{Y(z)}{X(z)} = \frac{K(z-q_1)(z-q_2)\dots(z-q_n)}{(z-p_1)(z-p_2)\dots(z-p_n)} \quad (7.25).$$

The factor K expresses the fact that b_0 and a_0 need not be equal to one.

The zeros and poles of such a rational function in z for a system with real valued coefficients come in conjugate pairs. That is, if p is a pole, then p^* is a pole (its conjugate) as well. In the same way, when z is a zero of $H(z)$ then z^* is a zero as well.

For a system characterized by a difference equation with real coefficients,

- the **poles should be inside the unit circle** for the system to be stable,
- Any such system (arbitrary n) can be built as the cascade (sequence) of second order systems (each with $n=2$) and possibly a first order system $n=1$,

Because a LTI system is completely characterized by its transfer function $H(z)$, the system is also completely characterized by its set of zeros and poles (together with the gain factor K). Plotting the zeros and poles in the complex plane characterizes the LTI system. As an example, it is considered the LTI system with the transfer function $H(z)$:

$$H(z) = \frac{z^2 - 1.9z + 1}{z^2 - 1.8z + 0.9} = \frac{(z - 0.95 - 0.31j)(z - 0.95 + 0.31j)}{(z - 0.9 - 0.3j)(z - 0.9 + 0.3j)}$$

The poles and zeros are plotted in the complex plane, poles are indicated with a small cross \times and zeros are indicated with a small circle \circ . In the complex plane, the unit circle is also shown.

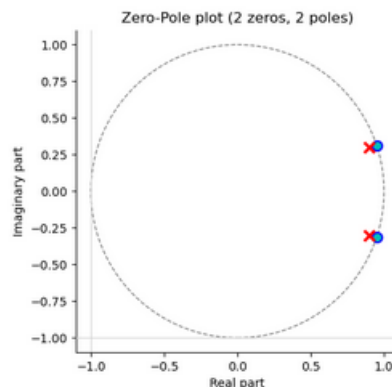


Figure 7.10: Poles and zeros placement in the z -plane for the previous transfer function.

7.7 Stability of discrete-time domain systems

It is distinguishing between internal system stability (stability of the system with zero-input response) and bounded input-bounded output (BIBO) stability (stability of the system with zero state response). As in the continuous time domain, discrete-time internal system stability depends on the location of the system eigenvalues, and system BIBO stability is determined by the nature of the system impulse response.

A discrete-time input-free system is stable if its zero-input response is bounded in time.

The linear discrete-time system is asymptotically stable if, in addition to being stable, its zero-input response tends to zero as time increases.

Internal system stability is related to the system eigenvalues. The discrete-time linear system eigenvalues are the solutions of the corresponding system characteristic equation.

If every eigenvalue of the system resides inside the unity circle of the Z-plane, the system is said to be asymptotically stable. If there are one or more eigenvalues that reside in the circle $z=1$ of the z-plane and the rest inside, the system is said to be marginally stable. And if there are one or more eigenvalues outside the circle of radius unity, the system is said to be unstable. Figure 7.11 represents graphically this idea.

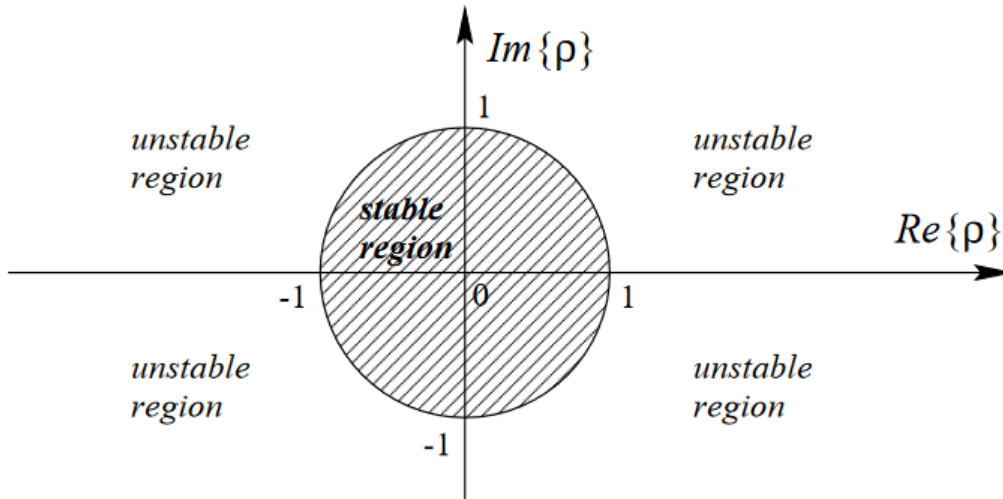


Figure 7.11: Z-plane stability regions.

For a discrete transfer function, there are two conditions, the causality and stability.

The causal condition for discrete-time LTI systems is as follows:

A discrete-time LTI system is causal when:

- ROC is outside the outermost pole.
- In the transfer function $H[z]$, the order of the numerator cannot be greater than the order of the denominator.

Stability Condition for Discrete Time LTI Systems

A discrete time LTI system is stable when:

- its system function $H[z]$ includes the unit circle $|z|=1$.
- all poles of the transfer function lie inside the unit circle $|z|=1$.

7.8 State variables

In the same vein as in continuous systems, discrete-time systems may be modeled as state variable equations. The model in state variables is obtained by decomposing the n^{th} -order difference equation into n -difference equations of first order. In linear systems, the structure of these equations of first order is such that they may be written as matrix equations.

Example 7.11: Calculate the state equation of a system described by,

$$y(k+2) = u(k) + 1.7y(k+1) - 0.72y(k) \quad (7.26)$$

Solution:

As equation (6.4) is a second-order difference equation, there will be two state variables. The selection of the state variables is not unique. Given a system, it is possible to choose different state variables to model its dynamic behavior. It is convenient to choose the ones that have physical meaning.

In this example, the following two state variables are chosen:

$$\begin{aligned}x_1(k) &= y(k) \\x_2(k) &= x_1(k + 1) = y(k + 1)\end{aligned}$$

where the state variables are the input and output of the system delayed by one sampling time. Substituting in the difference equation (7.26), the following state is obtained equation:

$$\begin{aligned}\begin{bmatrix} x_1(k + 1) \\ x_2(k + 1) \end{bmatrix} &= \begin{bmatrix} 0 & 1 \\ -0.72 & 1.7 \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u(k) \\ y(k) &= [1 \ 0] \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix}\end{aligned}$$

The general expression that describes the state equations of a linear and time-invariant system of n-order with r inputs and m outputs is:

$$\begin{aligned}x(k + 1) &= Ax(k) + Bu(k) \\ y(k) &= Cx(k) + Du(k)\end{aligned}$$

Where $x(k)$ is the state vector of $(nx1)$ order, $u(k)$ is the input vector of $(rx1)$ order, $y(k)$ is the output vector of $(mx1)$ order. A is the state matrix of $(n \times n)$ order, B is the input matrix of $(n \times r)$ order, C is the output matrix of $(m \times n)$ order. and D is the direct transition matrix of $(m \times r)$ order.

7.9 From state space to pulse transfer function

It has been stated that discrete-time systems of one input and one output may be modeled as pulse transfer functions.

Consider the system:

$$\begin{aligned}x(k + 1) &= Ax(k) + Bu(k) \\ y(k) &= Cx(k) + Du(k)\end{aligned}$$

applying the Z-transform,

$$\begin{aligned}zX(z) - zX(0) &= AX(z) + BU(z) \\ Y(z) &= cX(z) + DU(z)\end{aligned}$$

where $X(0)=0$, condition to calculate the pulse transfer function.

So, clearing $X(z)$ in the first equation, it is obtained:

$$X(z) = (zI - A)^{-1}B(z)$$

And substituting in the second equation,

$$\begin{aligned}Y(z) &= [C(zI - A)^{-1}B + D]U(z) \\ G(z) &= \frac{Y(z)}{U(z)} = C(zI - A)^{-1}B + D\end{aligned}$$

The inverse of a matrix is defined as, $M^{-1} = \frac{adj(M)^T}{|M|}$, $G(z)$ may be written as,

$$G(z) = C \frac{\text{adj}(zI - A)^{-1}}{|zI - A|} B + D$$

Where the poles of the system are the roots of the characteristic equation, $|zI - A| = 0$.

Example 7.12: Consider the system given by the following state equations:

$$x(k+1) = \begin{bmatrix} 0 & 1 \\ -2 & 3 \end{bmatrix} x(k) + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u(k)$$

$$y(k) = [0 \quad 1]x(k)$$

Calculate the pulse transfer function.

Solution:

$$(zI - A)^{-1} = \begin{bmatrix} z & -1 \\ 2 & z - 3 \end{bmatrix}^{-1} = \frac{1}{z^2 - 3z + 2} \begin{bmatrix} z - 3 & 1 \\ -2 & z \end{bmatrix}$$

Then,

$$G(z) = C \frac{\text{adj}(zI - A)^{-1}}{|zI - A|} B + D = [0 \quad 1] \frac{1}{z^2 - 3z + 2} \begin{bmatrix} z - 3 & 1 \\ -2 & z \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \frac{z}{z^2 - 3z + 2}$$

The same example using MATLAB:

$$A = [0 \ 1; -2 \ 3]; B = [0; 1]; C = [0 \ 1]; D = 0;$$

$$[G_{num}, G_{den}] = ss2tf(A, B, C, D)$$

The command `ss2tf` transforms the state space system into a transfer function system.

7.10 Sampled data control systems.

The sampled-data control systems include clock-driven elements and reflect the current trends in the design of feedback control systems. In control systems technology, data acquisition cards (DAQ) are commonly used to sense, sample, and process variables of interest. The controller is digitally implemented as a software routine on a programmable logic controller (PLC), microcontroller, digital signal processor (DSP), or Field Programmable Gate Array (FPGA).

The sampled-data control systems employ software-based controllers that work with discretized time. The discrete-time system models are represented by difference equations, with input and output variables represented by number sequences. The analog-to-digital (ADC) and digital-to-analog (DAC) converters are modeled as sampler and hold devices (Figure 7.12).

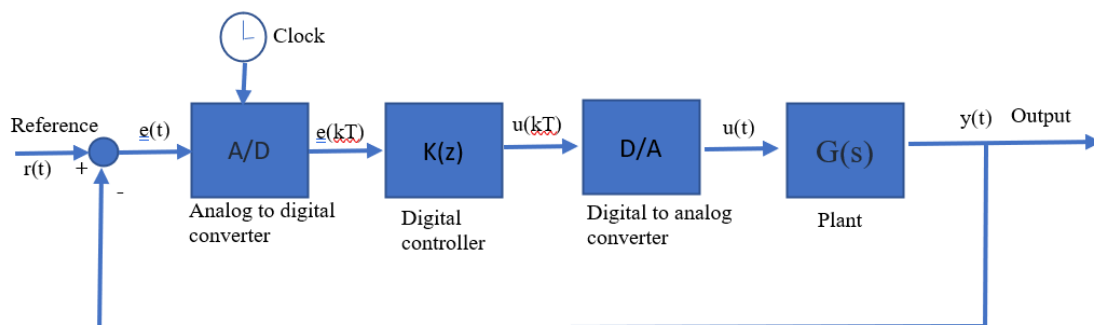


Figure 7.12: Discrete-time closed-loop system.

A continuous-time system model may be converted to a discrete system model by assuming a piece-wise constant input generated by a zero-order hold (ZOH) in general, but other kinds of holds exist. In MATLAB, the continuous-to-discrete conversion is handled by the 'c2d' function in the Control Systems Toolbox. The function allows a variety of input methods.

The Z-transform for discrete-time systems serves as the equivalent of the Laplace transform for continuous-time systems. Discrete system models are represented by pulse transfer functions that are valid at sampling instances. The added phase angle due to sampling adversely affects the dynamic stability of the closed-loop system. The discrete system stability, which is depicted by the roots of the characteristic polynomial being restricted to the inside of a unit circle, is indicated.

Analog controllers designed for transfer function models of continuous-time systems can be approximated for their application to sampled-data systems. Assuming a high enough sampling rate (five to ten times the system bandwidth), the digital controller obtained by emulation gives comparable performance to the analog controller it mimics.

There is a technique called root locus [3] that can be used for controller design in the case of discrete systems. The design is performed on the z-plane, keeping in view the stability boundary, i.e., the unit circle. The performance criteria, defined in terms of settling time, damping ratio, and so on, can be reflected on the z-plane.

In this section, models of sampled-data systems, their properties, stability characterization, and the analysis and design of controllers for such systems are discussed.

a) A/D Converter. Sampled data device.

An ADC converts a continuous-time and continuous-amplitude analog signal to a discrete-time and discrete-amplitude digital signal. The conversion involves quantization of the input, so it necessarily introduces a small amount of error or noise. Furthermore, instead of continuously performing the conversion, an ADC does the conversion periodically, sampling the input, limiting the allowable bandwidth of the input signal.

The performance of an ADC is primarily characterized by its bandwidth and signal-noise ratio (SNR). The bandwidth of an ADC is characterized primarily by its sampling rate. The SNR of an ADC is influenced by many factors, including resolution, linearity, accuracy (how well the quantization levels match the true analog signal), aliasing, and jitter. The SNR of an ADC is often summarized in terms of its effective number of bits (ENOB), the number of bits of each measure it returns that are, on average, not noise. An ideal ADC has an ENOB equal to its resolution. ADCs are chosen to match the bandwidth and required SNR of the signal to be digitized. If an ADC operates at a sampling rate greater than twice the bandwidth of the signal, then, per the Nyquist-Shannon sampling theorem, perfect reconstruction is possible. The presence of quantization error limits the SNR of even an ideal ADC. However, if the SNR of the ADC exceeds that of the input signal, its effects may be neglected, resulting in an essentially perfect digital representation of the analog input signal.

The resolution of the converter indicates the number of different, i.e., discrete, values it can produce over the allowed range of analog input values. Thus, a particular resolution determines the magnitude of the quantization error and therefore determines the maximum possible signal-noise ratio for an ideal ADC without the use of oversampling. The input samples are usually stored electronically in binary form within the ADC, so the resolution is usually expressed as the audio bit depth. In consequence, the number of

discrete values available is usually a power of two. For example, an ADC with a resolution of 8 bits can encode an analog input to one of 256 different levels ($2^8 = 256$). The values can represent the ranges from 0 to 255 (i.e., as unsigned integers) or from -128 to 127 (i.e., as signed integers), depending on the application.

b) Sampling theorem

The sampling theorem specifies the minimum sampling rate at which a continuous-time signal needs to be uniformly sampled so that the original signal can be completely recovered or reconstructed by these samples alone. This is usually referred to as Shannon's sampling theorem in the literature.

The **Shannon's sampling theorem** says that if a continuous time signal contains no frequency components higher than W Hz, then it can be completely determined by uniform samples taken at a rate of f_s samples per second where:

$$f_s \geq 2W$$

or, in terms of the sampling period:

$$(T \leq 1/2W).$$

A signal with no frequency component above a certain maximum frequency is known as a bandlimited signal.

The minimum sampling rate allowed by the sampling theorem ($f_s = 2W$) is called the Nyquist rate.

c) D/A Converter. Reconstruction filter.

In a mixed-signal system (analog and digital), a reconstruction filter, sometimes called an anti-imaging filter, is used to construct a smooth analog signal from a digital input, as in the case of a digital-analog converter (DAC) or other sampled data output device.

The sampling theorem describes why the input of an ADC requires a low-pass analog electronic filter, called the anti-aliasing filter: the sampled *input* signal must be bandlimited to prevent aliasing (here meaning waves of higher frequency being *recorded* as a lower frequency).

For the same reason, the output of a DAC requires a low-pass analog filter, called a reconstruction filter - because the *output* signal must be bandlimited to prevent imaging (meaning Fourier coefficients being reconstructed as spurious high-frequency 'mirrors'). This is an implementation of the Shannon interpolation formula.

Ideally, both filters should be brick wall filters, with a constant phase delay in the bandpass and a constant flat frequency response, and zero response from the Nyquist frequency. This can be achieved by a filter with a 'sinc' impulse response.

d) Implementation

While in theory a DAC outputs a series of discrete Dirac impulses, in practice, a real DAC outputs pulses with a finite bandwidth and width. Both idealized Dirac pulses have zero-order holding steps, and other output pulses, if unfiltered, would contain spurious high-frequency replicas, "*or images*" of the original bandlimited signal. Thus, the reconstruction filter smooths the waveform to remove image frequencies (copies) above the Nyquist limit. In doing so, it reconstructs the continuous time signal (whether originally sampled or modeled by digital logic) corresponding to the digital time sequence.

Practical filters have a non-flat frequency or phase response in the pass band and incomplete suppression of the signal elsewhere. The ideal sinc waveform has an infinite response to a signal, in both the positive and negative time directions, which is impossible to perform in real time as it would require infinite delay. Consequently, real reconstruction filters typically either allow some energy above the Nyquist rate, attenuate some in-band frequencies, or both. For this reason, oversampling may be used to ensure that frequencies of interest are accurately reproduced without excess energy being emitted out of band.

In systems that have both, the anti-aliasing filter and a reconstruction filter may be of identical design. For example, both the input and the output of audio equipment may be sampled at 44.1 kHz. In this case, both audio filters block as much as possible above 22 kHz and pass as much as possible below 20 kHz.

Alternatively, a system may have no reconstruction filter and simply tolerate some energy being wasted reproducing higher frequency images of the primary signal spectrum.

7.11 Analysis of closed-loop discrete-time systems

In this section, it is discussed converting continuous-time models into discrete-time (or difference equation) models. It also introduces the Z-transform and show how to use it to analyze and design controllers for discrete-time systems.

The MATLAB commands which will be used in this example are the following: `c2d`, `pzmap`, `zgrid`, `step`, and `rlocus`.

7.11.1 Introduction

Figure 7.13 below shows the typical continuous-time feedback system that we have been considering so far in this document. Almost all continuous-time controllers can be implemented using analog electronics.

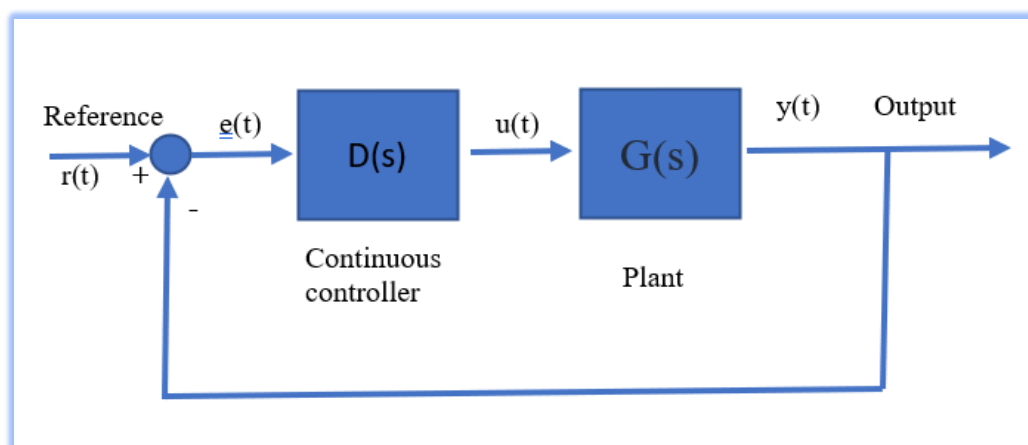


Figure 7.13: Closed-loop continuous-time control scheme.

The continuous controller, enclosed in the shaded rectangle (Figure 7.13), can be replaced by a digital controller, shown below, that performs the same control task as the continuous controller. The basic difference between these controllers is that the digital system

operates on discrete signals (samples of the sensed signals) rather than on continuous signals. Such a change may be necessary if it is required to implement the control algorithm in software on a digital computer, which is frequently the case.

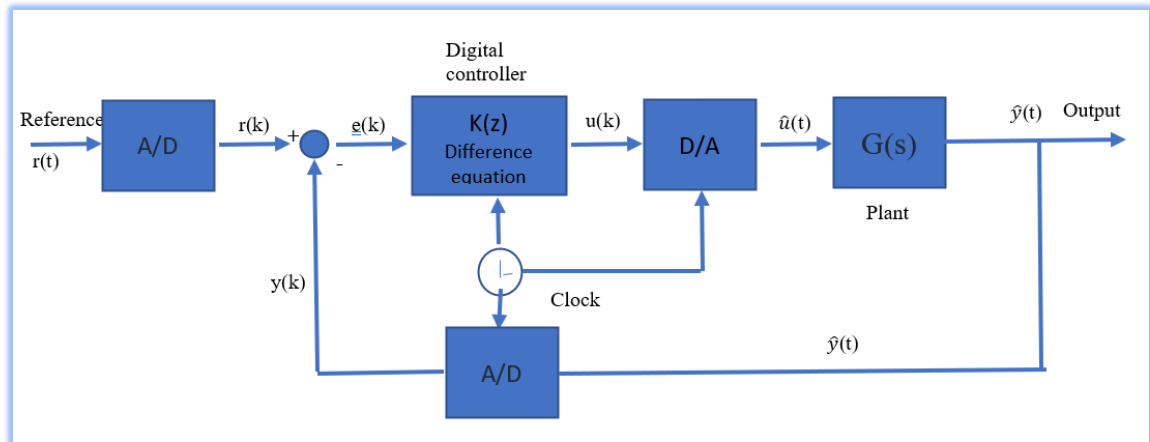


Figure 7.14: Closed-loop discrete-time control scheme.

The various signals in the above digital system schematic can be represented by the following plots (Figure 7.15):

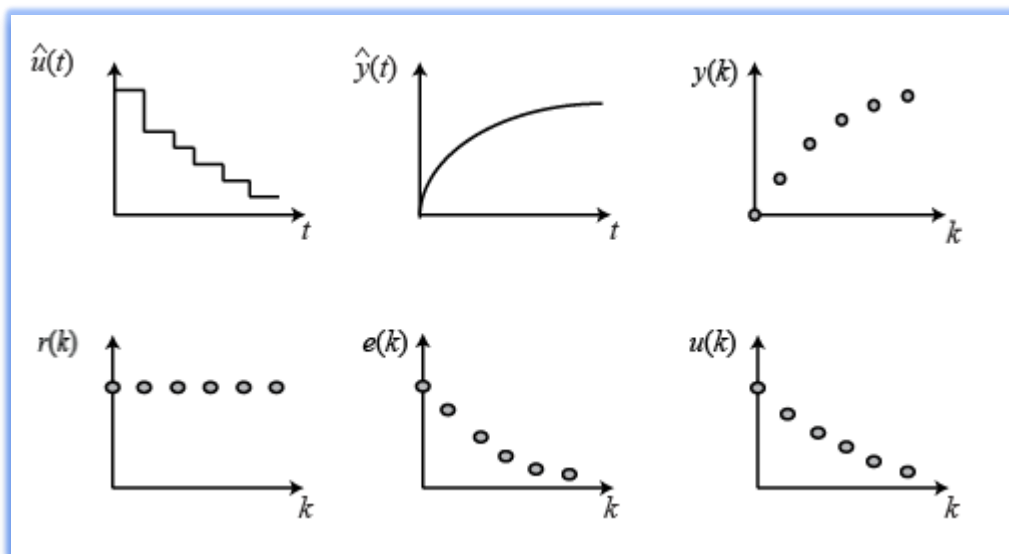


Figure 7.15: Examples of signals, $\hat{u}(t)$, continuous control signal, $\hat{y}(t)$, output signal, $y(k)$, discrete-time output signal, $r(k)$, discrete-time reference signal, $e(k)$, discrete-time error signal, and $u(k)$, discrete-time control signal corresponding to the block diagram of figure 6.10.

The purpose of this digital control example is to demonstrate how to use MATLAB to work with discrete functions, either in transfer function or state-space form, to design digital control systems.

7.11.2 Zero-hold equivalence

In the above schematic of the digital control system, it is seen that the system contains both discrete and continuous portions. Typically, the system being controlled is in the physical world and generates and responds to continuous-time signals, while the control

algorithm may be implemented on a digital computer. When designing a digital control system, it is first necessary to find the discrete equivalent of the continuous portion of the system.

For this technique, it is considered the following portion, see figure 7.16 of the digital control (Figure 7.14) system and rearrange as follows:

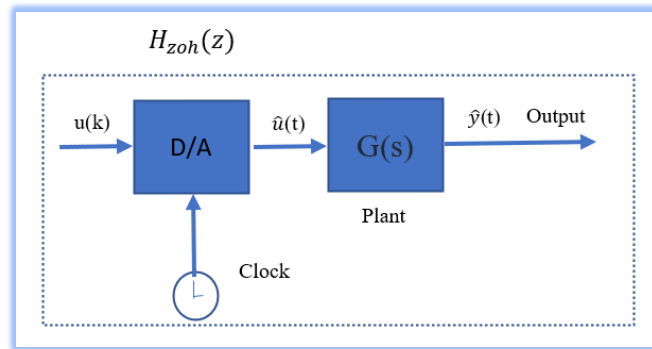


Figure 7.16: Portion of the digital controller considered in this section.

The **clock** connected to the **D/A and A/D converters** supplies a pulse every T seconds and each D/A and A/D sends a signal only when the pulse arrives. The purpose of having this pulse is to require that $H_{zoh}(z)$ (equivalent discrete transfer function of $G(s)$) acts only on periodic input samples $u(k)$, and produces periodic outputs $y(k)$ only at discrete intervals of time; thus, $H_{zoh}(z)$ can be realized as a discrete function.

The idea behind the design is the following: it is required to find a discrete function $H_{zoh}(z)$ so that for a piecewise constant input to the continuous system $G(s)$, the sampled output of the continuous system equals the discrete output. Suppose the signal $u(k)$ represents a sample of the input signal. There are techniques for taking this sample $u(k)$ and holding it to produce a continuous signal $\hat{u}(t)$. The sketch below shows one example where the continuous signal $\hat{u}(t)$ is held constant at each sample $u(k)$ over the interval kT to $(k+1)T$. This operation of holding $\hat{u}(t)$ constant over the sample period is called a **zero-order hold**.

A zero-order hold (ZOH) reconstructs a piece-wise constant signal from a number sequence and represents a model of the digital-to-analog converter (DAC). Assuming that the input to the ZOH is a sampled signal, $r(kT) = r(t)|_{t=kT}$, its output is given as:

$$r(t) = r((k-1)T) \text{ for } (k-1)T \leq t < kT$$

The output of the ZOH to an arbitrary input, $r(kT)$, is a staircase reconstruction of the analog signal, $r(t)$. The impulse response of the ZOH is a square pulse (Figure 7.17): $g_{zoh}(t) = 1, 0 < t < 1$.

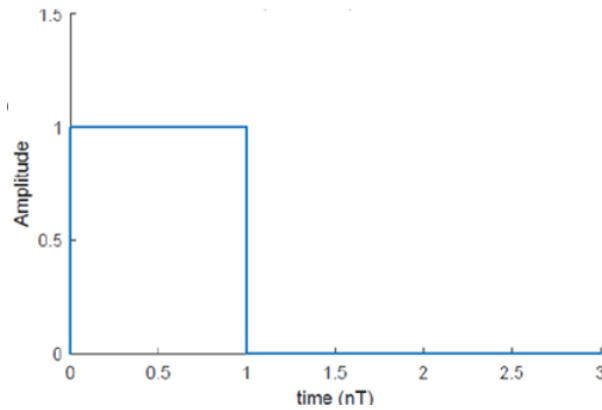


Figure 7.17: Unit impulse response of a ZOH.

By applying the Laplace transform to the ZOH impulse response, its transfer function is obtained as:

$$G_{zoh}(s) = \frac{1}{s} - \frac{e^{-sT}}{s} = \frac{1 - e^{-sT}}{s}$$

As a result, the pulse transfer function of a continuous-time plant, $G(s)$, is obtained as:

$$G(z) = Z \left\{ \frac{1 - e^{-sT}}{s} G(s) \right\} = (1 - z^{-1}) Z \left\{ \frac{G(s)}{s} \right\}$$

where $Z \left\{ \frac{G(s)}{s} \right\}$ denotes the z-transform of a sequence obtained by sampling $\int g(t)dt$.

Example 7.13: Let $G(s) = \frac{a}{s+a}$; then $G(z) = (1 - z^{-1}) Z \left\{ \frac{a}{s(s+a)} \right\}$.

By using the z-transform table: $G(z) = \frac{z-1}{z} \left(\frac{z}{z-1} - \frac{z}{z-e^{-aT}} \right) = \frac{1-e^{-aT}}{z-e^{-aT}}$.

For numerical purposes, suppose $a=1$, $T=0.2s$. Then, the pulse transfer function is obtained as follows: $G(z) = \frac{1-e^{-0.2}}{z-e^{-0.2}} = \frac{0.181}{z-0.819}$.

Figure 7.18 shows the reconstruction of a signal from a ZOH.

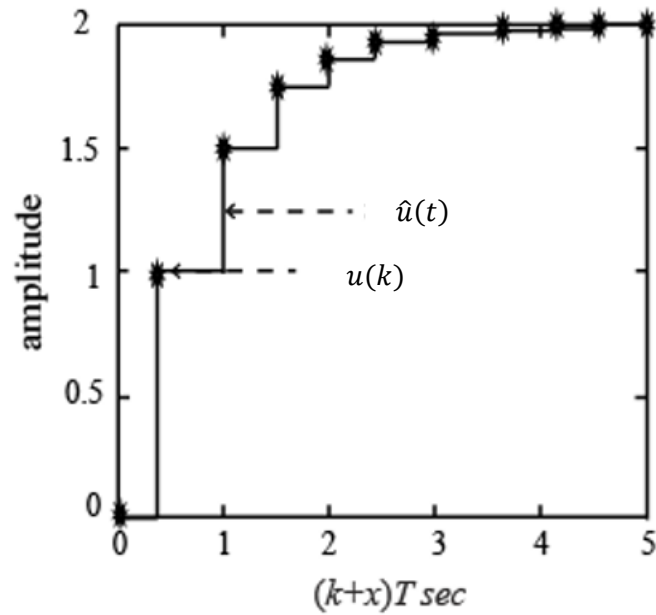


Figure 7.18: Reconstruction of the continuous time signal $\hat{u}(t)$ ($uhat(t)$) using a ZOH from the discrete time sequence $u(k)$.

The held signal $\hat{u}(t)$ then is passed to $H_2(s)$ and the A/D produces the output $y(k)$, which will be the same piecewise signal as if the discrete signal $u(k)$ had been passed through $H_{zoh}(z)$ to produce the discrete output $y(k)$ (see Figure 7.19)

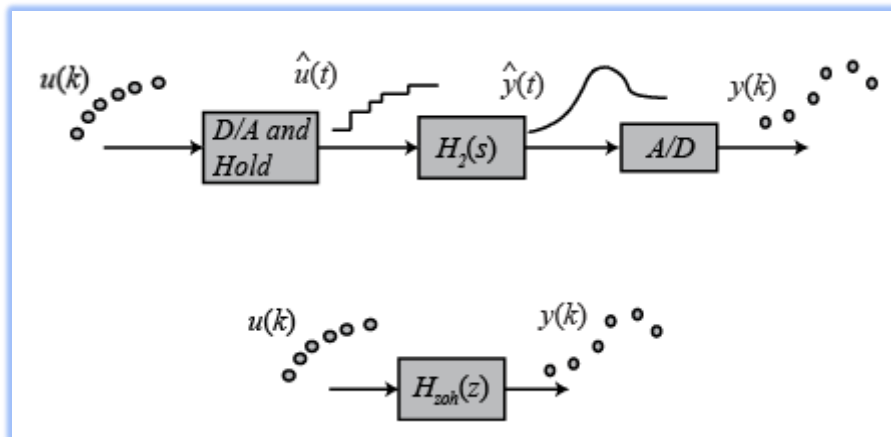


Figure 7.19: Analogy to visualize the conversion of a continuous transfer function into a discrete time and the processing of the signals under a ZOH.

Now it will be redrawing the schematics, replacing the continuous portion of the system with $H_{zoh}(z)$ (Figure 7.20)

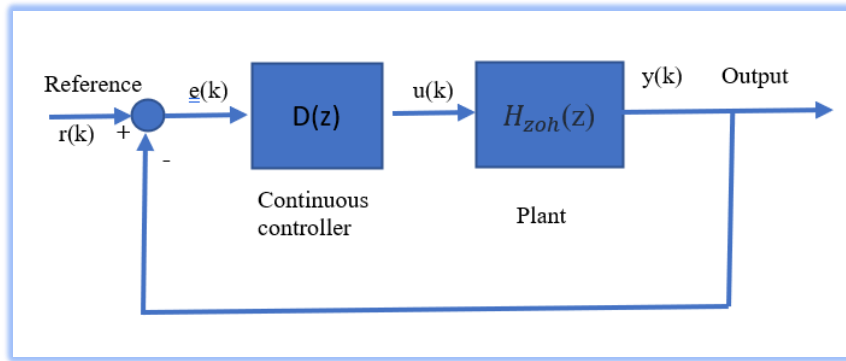


Figure 7.20: A digital filter or controller acting on a discrete transfer function.

Now it is possible to design a digital control system dealing with only discrete functions. **Note:** There are certain cases where the discrete response does not match the continuous response generated by a hold circuit implemented in a digital control system.

There is a MATLAB function `c2d` that converts a given continuous system (either in transfer function or state-space form) to a discrete system using the zero-order hold operation explained above. The basic syntax for this in MATLAB is `sys_d = c2d(sys,Ts,'zoh')`.

The sampling time (T_s in sec/sample) should be smaller than $1/(30BW)$, where BW is the system's closed-loop bandwidth frequency.

Example 7.14: From continuous to discrete transfer function for mass-spring-damper system with MATLAB

The following continuous **transfer function** models the mass-spring-damper system as shown in previous chapters:

$$\frac{X(s)}{F(s)} = \frac{1}{ms^2 + bs + k}$$

Assuming the closed-loop bandwidth frequency is greater than 1 rad/sec, the sampling time (T_s) is chosen to be equal to 1/100 sec. Now, a new m-file is created. The system is defined with the following values for the mass-spring system.

```
% Define input parameters
m = 1;
b = 10;
k = 20;

% Define continuous transfer function
s = tf('s');
sys = 1/(m*s^2+b*s+k);

% Define sampling time
Ts = 1/100;

% Define discrete transfer function from continuous transfer function
```

```
sys_d = c2d(sys,Ts,'zoh')
```

```
% Output of the discrete transfer function in MATLAB
```

```
sys_d =
```

```
4.837e-05 z + 4.678e-05
-----
z^2 - 1.903 z + 0.9048
```

```
Sample time: 0.01 seconds.
Discrete-time transfer function.
```

State-Space model.

A continuous-time state-space model of this system is the following:

$$\dot{\mathbf{x}} = \begin{bmatrix} \dot{x} \\ \dot{\ddot{x}} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\frac{k}{m} & -\frac{b}{m} \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{m} \end{bmatrix} F(t)$$

$$y = [1 \quad 0] \begin{bmatrix} x \\ \dot{x} \end{bmatrix}$$

All constants are the same as before. The following m-file converts the above continuous-time state-space model to a discrete-time state-space model.

```
% Define the matrixes A, B, C and D
```

```
A = [0 1; -k/m -b/m];
```

```
B = [0; 1/m];
```

```
C = [1 0];
```

```
D = [0];
```

```
% Sampling time is the same as before
```

```
Ts = 1/100;
```

```
% State-space model in continuous-time
```

```
sys = ss (A, B, C, D);
```

```
% State-space model in discrete-time
```

```
sys_d = c2d (sys, Ts, 'zoh')
```

```
% Output values of the state-space model in discrete-time
```

```
sys_d =
```

```
A =
    x1    x2
x1  0.999 0.009513
x2 -0.1903 0.9039
```

```
B =
```

$$\begin{array}{l} u1 \\ x1 \ 4.837e-05 \\ x2 \ 0.009513 \end{array}$$

$$C = \begin{array}{l} \\ x1 \ x2 \\ y1 \ 1 \ 0 \end{array}$$

$$D = \begin{array}{l} u1 \\ y1 \ 0 \end{array}$$

Sample time: 0.01 seconds.

From these matrices, the discrete state-space can be written as

$$\begin{bmatrix} x(k) \\ v(k) \end{bmatrix} = \begin{bmatrix} 0.9990 & 0.0095 \\ -0.1903 & 0.9039 \end{bmatrix} \begin{bmatrix} x(k-1) \\ v(k-1) \end{bmatrix} + \begin{bmatrix} 0 \\ 0.0095 \end{bmatrix} F(k-1)$$

$$y(k-1) = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} x(k-1) \\ v(k-1) \end{bmatrix}$$

And the discrete-time state-space model is obtained.

Example 7.15: Discrete-time transfer function in Z-domain: connection with Laplace domain

For continuous systems, it is known that certain behaviors result from different pole locations in the s-plane. For instance, a system is unstable when any pole is located to the right of the imaginary axis. For discrete systems, the system behavior is analyzed from different pole locations in the z-plane. The characteristics in the z-plane can be related to those in the s-plane by the following expression.

$$z = e^{sT}$$

- T = sampling time (sec/sample)
- s = location in the s-plane
- z = location in the z-plane

It is supposed that the following discrete transfer function models the system:

$$\frac{Y(z)}{F(z)} = \frac{1}{z^2 - 0.3z + 0.5}$$

The following m-file program collocates the poles in the z-plane:

```
numDz = 1;
denDz = [1 -0.3 0.5];
sys = tf(numDz, denDz,-1); % the -1 indicates that the sample time is undetermined
```

```
pzmap(sys)
axis ([-1 1 -1 1])
zgrid
```

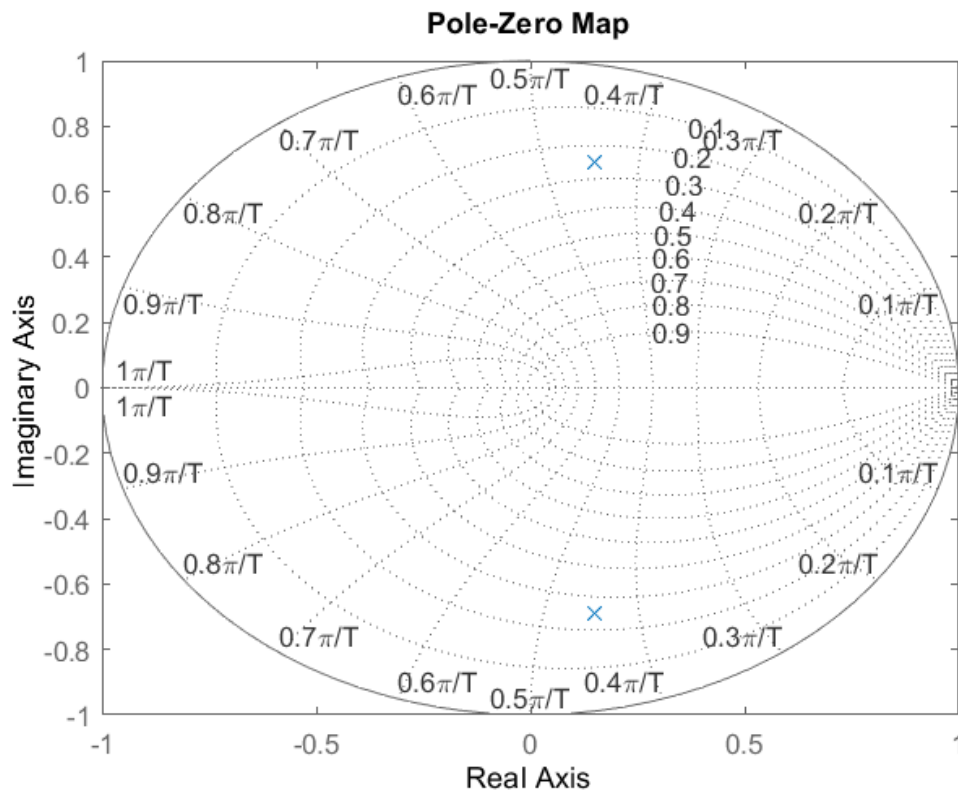


Figure 7.21: Pole colocation in the Z-plane of Example 7.15.

From this plot, we see the poles are located approximately at a natural frequency of $9\pi/20T$ (rad/sample) and a damping ratio of 0.25.

Figure 7.21 shows the mapping of lines of constant damping ratio (ζ) and natural frequency (ω_n) from the s-plane to the z-plane using the expression shown above.

It is noticed that in the z-plane, the stability boundary is no longer the imaginary axis but rather the circle of radius one centered at the origin, $|z| = 1$, referred to as the **unit circle**. A discrete system is stable when all poles are located inside the unit circle and unstable when any pole is located outside the circle.

If the sampling time of $1/20$ sec (which leads to $\omega_n = 28.2$ rad/sec) and using the equations shown in Chapter 4 for second order systems, it can be determined that this system should have a rise time of 0.06 sec, a settling time of 0.65 sec, and a maximum percent overshoot of 45% (0.45 times more than the steady-state value). The step response is obtained to check these values. The following commands are added to the above m-file and rerun it in the command window. Figure 7.22 shows the obtained step response.

```
sys = tf(numDz, denDz, 1/20);
step(sys, 2.5);
```

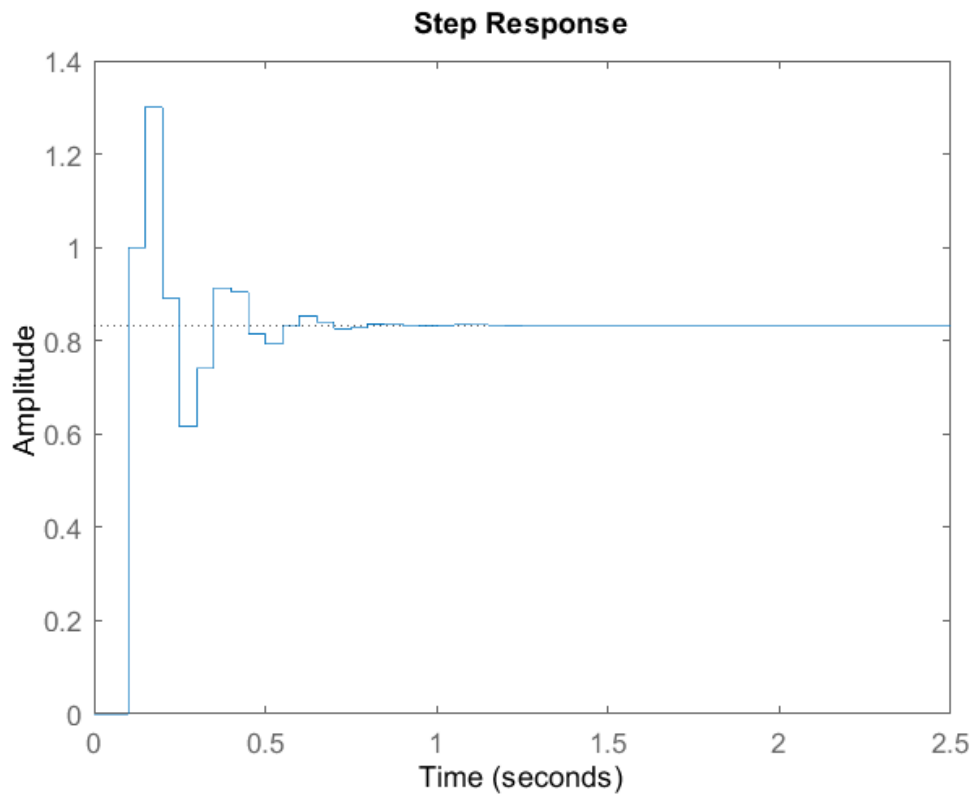



Figure 7.22: Discrete-time step response.

As it is seen from figure 7.22, the rise time, settling time, and overshoot came out to be what it is expected. This explains how the locations of poles and the equations of overshoot, rise time, and settling time can be used to analyze the transient response of a discrete system.

Important: The natural frequency (ω_n) in the z-plane has units of rad/sample, but when it is used in the equations shown above, ω_n must be represented in units of rad/sec.

Example 7.16: Discrete root locus using MATLAB

The root locus is the locus of points where roots of the characteristic equation can be found as a single parameter is varied from zero to infinity. The characteristic equation of our unity-feedback system with simple proportional gain, K , is:

$$1 + KG(z)H_{zoh}(z) = 0$$

where $G(z)$ is the digital controller and $H_{zoh}(z)$ is the plant transfer function in the z-domain (obtained by implementing a zero-order hold).

The mechanics of drawing the root-loci are the same in the z-plane as in the s-plane. Recalling the continuous root-locus example (Section 6.3 Root Locus), it was used with the MATLAB function *sgrid* to find the root-locus region that gives an acceptable gain (K) for specifications in the damping ratio and natural frequency, or, in other words, specifications in the time domain, such as overshoot, settling time, and rise time. For the

discrete root-locus analysis, the function **zgrid** has the same function as **sgrid**. The syntax **zgrid** (ζ, ω_n) draws lines of constant damping ratio (ζ) and natural frequency (ω_n).

Supposing the following discrete transfer function is analyzed:

$$\frac{Y(z)}{F(z)} = \frac{z - 0.3}{z^2 - 1.6z + 0.7}$$

the requirements are a damping ratio greater than 0.6 and a natural frequency greater than 0.4 rad/sample (these can be found from the design requirements, the sampling period (sec/sample), and the three equations shown in the previous section). The following commands draw the root-locus with the lines of constant damping ratio and natural frequency. Running the following m-file, it generates the root-locus plot of figure 7.23.

```
% Define discrete closed loop transfer function
numDz = [1 -0.3];
denDz = [1 -1.6 0.7];
sys = tf(numDz, denDz,-1);
% Calculate root locus and define axis of plot
rlocus(sys)
axis ([-1 1 -1 1])
% Define specifications of the controller
zeta = 0.4;
omega_n = 0.3;
zgrid (zeta, omega_n)
```

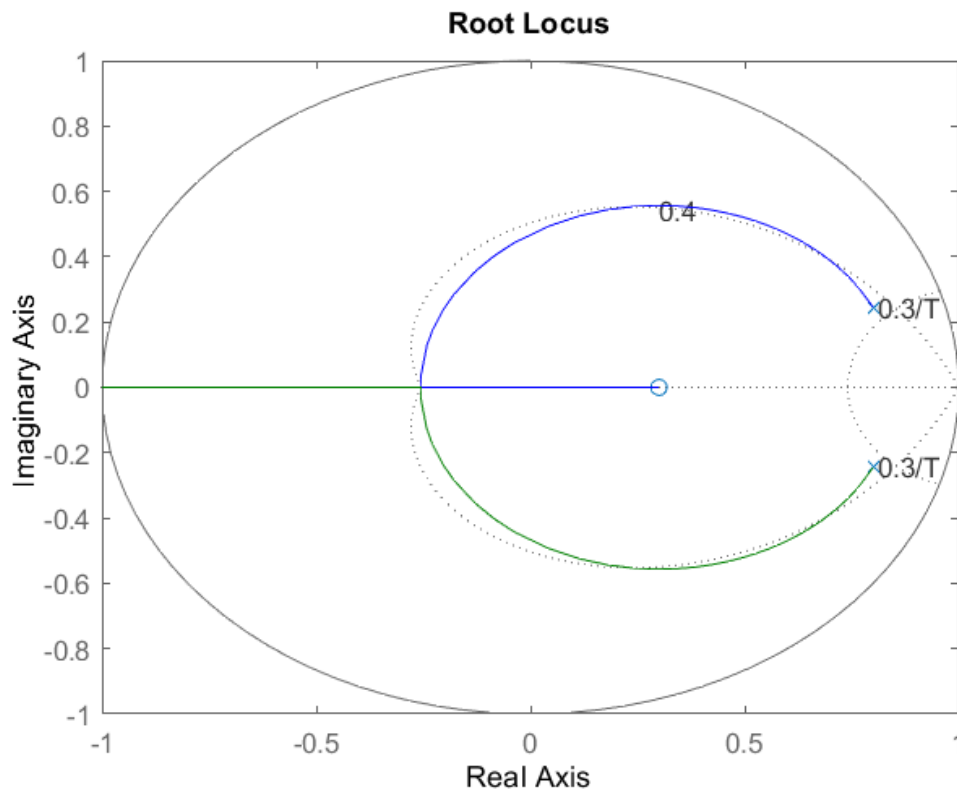


Figure 7.23: Root-locus with the lines of constant damping ratio and natural frequency.

From this plot, it can be observed that the system is stable for some values of K since there are parts of the root locus where both branches are located inside the unit circle. Also, it is observed that two dotted lines represent the constant damping ratio and natural frequency. The natural frequency is greater than 0.3 outside the constant- ω_n line, and the damping ratio is greater than 0.4 inside the constant-zeta line. In this example, parts of the generated root locus are within the desired region. Therefore, the gain (K) chosen to place the two closed-loop poles on the loci within the desired region should provide a response that satisfies the given design requirements[4].

References:

- [1] Ifeachor, E.C, and Jervis, B.W. Digital Signal Processing, a practical approach. Prentice-Hall, New Jersey, 2nd. Ed., 2002
- [2] Benjamin C. Kuo and Farid Golnaraghi. 2002. Automatic Control Systems (8th. ed.). John Wiley & Sons, Inc., USA.
- [3] Nise, Norman S. Control Systems Engineering. 4th ed. Hoboken, NJ: John Wiley, 2004.
- [4] Rakesh Patel (2024). Root Locus design for Discrete time system (<https://www.mathworks.com/matlabcentral/fileexchange/28777-root-locus-design-for-discrete-time-system>), MATLAB Central File Exchange. Retrieved June 26, 2024.

Chapter 8: PID Control

This chapter addresses some different approaches to control systems. These are control structures and components. According to the natural evolution of control theory, they may be classified as open and closed loop control. Open-loop control is a straightforward strategy that requires complete knowledge of the system to be successful. A closed-loop controller is a more sophisticated strategy and relies on feedback from the signal of the output to the input. Once the error of the output with respect to the input or reference signal is identified, the controller is designed based on that error. There are several methodologies for closed-loop controller which will be reviewed in this chapter. Some of the most significant are PID controllers based. The PID controller is presented in more detail.

In an **open-loop controller**, also called a **non-feedback controller**, the control action from the controller is independent of the "process output," which is the process variable that is being controlled. It does not use feedback to determine if its output has achieved the desired goal of the input command or process "set point."

There are many open-loop controls, such as on-off switching of valves, machinery, lights, stepping motors, or heaters, where the control result is known to be approximately sufficient under normal conditions without the need for feedback. The advantage of using open-loop control in these cases is the reduction in component count and complexity. However, an open-loop system cannot correct any errors that it makes or correct for outside disturbances, and it cannot engage in machine learning.

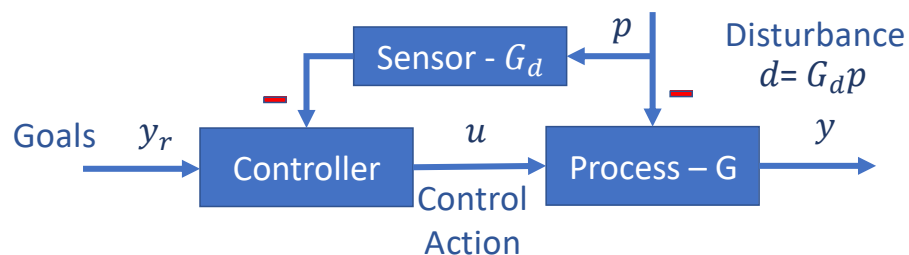


Figure 8.1: Open-loop control architecture.

Figure 8.1 shows a typical open loop control architecture. It is represented by the plant of the process, $G(s)$, disturbances, d , that influence the process, a sensor that measures disturbance signals, and the controller, which proposes actions to command the goals set by the designer or manager, avoiding, or minimizing disturbances. Mathematically, the loop is composed of:

$$\begin{aligned} \text{Plant process model: } y &= Gu + d = Gu + G_d p \\ \text{Actual commanded control action: } u &= G^{-1}[y_r - G_p d] \end{aligned}$$

$$\text{Measured disturbance: } \hat{d} = G_d \hat{p}$$

The process model is represented as the relation between the input, u , and output, y , adding the disturbances affecting the system. The control action should invert the dynamics of the process, G , and remove disturbance, $\hat{d} = G_d \hat{p}$ (estimation of the disturbance). In this way, the reference signal, y_r , is achieved, which means goals are achieved. Please, note that the mathematical model of the process might be invertible, i.e., $\exists G^{-1}, s \in \mathbb{C}$, where \mathbb{C} represents the complex numbers.

As a result, for a successful controller, perfect knowledge about the plant/process/model of the system is required. Moreover, the feasibility and stability of the inverse transfer function of the plant/system/process G^{-1} are needed. Finally, knowledge about the measurement system and its model is involved in cases of disturbances affecting the system for effective estimation.

If the objective of the controller is to track the reference signal, then the control objective is such as: $u = G_r y_r \cong G^{-1} y_r$. However, if the objective of the controller becomes the rejection of the disturbance, the design of the controller turns out to be $u = -G_d \hat{p} \cong -G^{-1} G_d \hat{p}$, which requires the measurement of the disturbance or having access to the disturbance and the feasibility of the transfer function of the measurement system multiplied by the disturbance itself $-G_d \hat{p}$.

As a review of the open loop controllers: 1) It is required knowledge of the plant/process/system, i.e., the transfer function of the system must be modeled. 2) The measurement system and its model must be known to avoid the effects of the disturbances. Otherwise, no disturbances must affect the system if adequate control action is expected. 3) A track reference signal and rejection of the disturbance may be achieved at the same time. Feasibility and stability of the inverse transfer function of the plant/system/process G^{-1} is needed for these purposes, or, in other words, there will not be stability problems in control action. 4) With previous characteristics, zero error may be achieved. Note that this characteristic is not possible in closed loop arrangements.

A **feedback control system** can be improved by combining the feedback (or closed-loop) control of a controller with feed-forward (or open-loop) control. Knowledge about the system (such as the desired acceleration and inertia) can be fed forward and combined with the controller output to improve the overall system performance. The feed-forward value alone can often provide most of the controller output. The controller primarily must compensate for whatever difference or *error* remains between the setpoint (SP) and the system response to the open-loop control. Since the feedforward output is not affected by the process feedback, it can never cause the control system to oscillate, thus improving the system response without affecting stability. Feedforward can be based on the setpoint and on extra measured disturbances. Setpoint weighting is a simple form of feedforward.

For example, in most motion control systems, to accelerate a mechanical load under control, more force is required from the actuator. If a velocity loop controller is being used to control the speed of the load and command the force being applied by the actuator, then it is beneficial to take the desired instantaneous acceleration, scale that value appropriately, and add it to the output of the velocity loop controller. This means that whenever the load is being accelerated or decelerated, a proportional amount of force is commanded from the actuator, regardless of the feedback value. The loop in this situation uses the feedback information to change the combined output to reduce the remaining difference between the process setpoint and the feedback value. Working together, the

combined open-loop feed-forward controller and closed-loop controller can provide a more responsive control system in some situations.

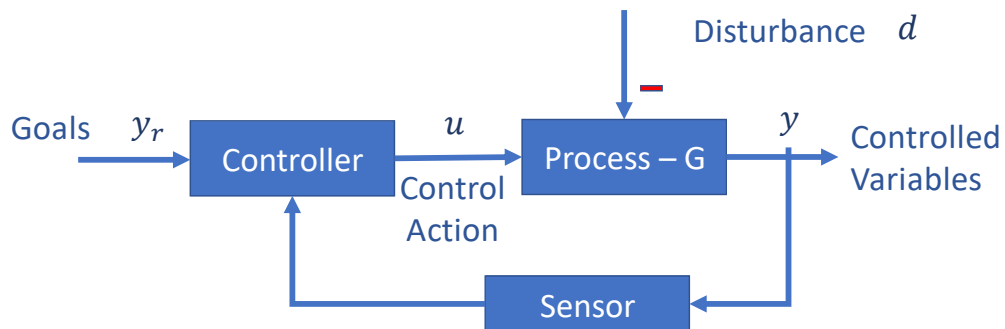


Figure 8.2: Feedback control loop

Figure 8.2 shows the closed-loop controller. The controller measures the model of the plant process, the absence of disturbances, and the set goals. The basic closed-loop control system shown in Figure 8.2 can describe a variety of control systems, including those driving elevators, thermostats, and cruise control.

Closed-loop control systems typically operate at a fixed frequency. The frequency of changes to the *drive signal* is usually the same as the sampling rate (for discrete-time systems), and certainly not any faster. After reading each new sample from the sensor, the software reacts to the plant's changed state by recalculating and adjusting the drive signal. The plant responds to this change, another sample is taken, and the cycle repeats. Eventually, the plant should reach the desired state, and the software will cease making changes. Please, note the negative feedback loop of the retro-alimentation (Figure 8.3) which gives the control loop the error signal to design the controller to minimize it.

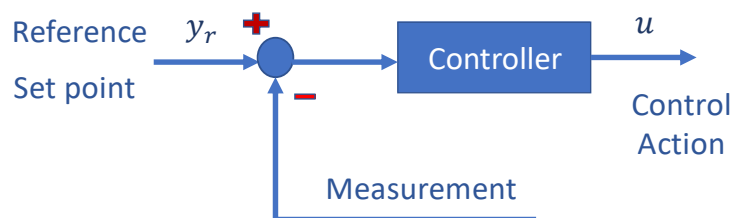


Figure 8.3: Detailed explanation of feedback control loop

On-off closed loop structure

On-off control is the simplest form of feedback control. An on-off controller simply drives the manipulated variable from fully closed to fully open, depending on the position of the controlled variable relative to the setpoint. A common example of on-off control is the temperature control in a domestic heating system. When the temperature is below the thermostat setpoint, the heating system is switched on, and when the temperature is above the setpoint, the heating switches off.

There is, however, a bit of subtlety applied in practical on-off systems. If the heating switches on and off the instant the measured temperature crosses the setpoint, then the system would *chatter*, repeatedly switching on and off at very high frequency. If this

happened, the boiler wouldn't last very long. To avoid chattering, practical on-off controllers usually have a *deadband* around the setpoint. When the measured value lies within this dead band the controller does nothing: it is only when the value moves outside that action is taken. The effect of this is to introduce continuous oscillation in the value of the controlled variable; the larger the deadband the higher the amplitude and lower the frequency.

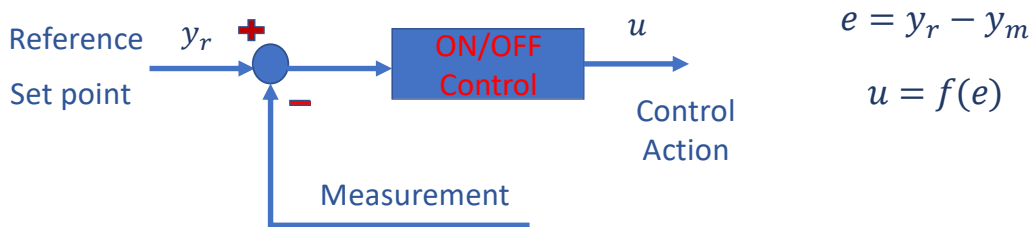


Figure 8.4: Detailed explanation of the ON/OFF feedback control loop.

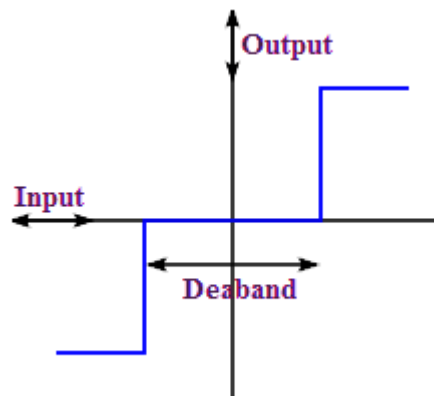


Figure 8.5: ON/OFF control loop with dead band.

As a summary of the ON/OFF control, it is possible to account for: 1) In an on/off control, the control signal is either 0% or 100% or -V or +V. 2) If the control at the setpoint is not achievable, a dead-band must be incorporated. 3) It is useful for large, sluggish systems, particularly those incorporating electric heaters.

PID closed-loop controller

The **PID circuit** is often utilized as a control loop feedback controller and is very commonly used for many forms of servo circuits. The letters making up the acronym PID correspond to Proportional (P), Integral (I), and Derivative (D), which represent the three control settings of a PID circuit. The purpose of any servo circuit is to hold the system at a predetermined value (set point) for long periods of time. The PID circuit actively controls the system to hold it at the set point by generating an error signal that is essentially the difference between the set point and the current value. The three controls relate to the time-dependent error signal; at its simplest, this can be thought of as follows: Proportional is dependent upon the present error; Integral is dependent upon the accumulation of past errors (please note that the integral part sums previous errors and accumulates them); and Derivative is the prediction of future errors (the derivative part is represented by the slope of the controller that directs the error signal). The results of each of the controls are then fed into a weighted sum, which then adjusts the output of the

circuit, $u(t)$. This output is fed into a control device, its value is fed back into the circuit, and the process is allowed to actively stabilize the circuit's output to reach and hold at the set point value. The block diagram below illustrates very simply the action of a PID in a parallel circuit. One or more combinations of the PID control parameters can be utilized in any servo circuit depending on system demand and requirements (i.e., P, I, PI, PD, or PID).

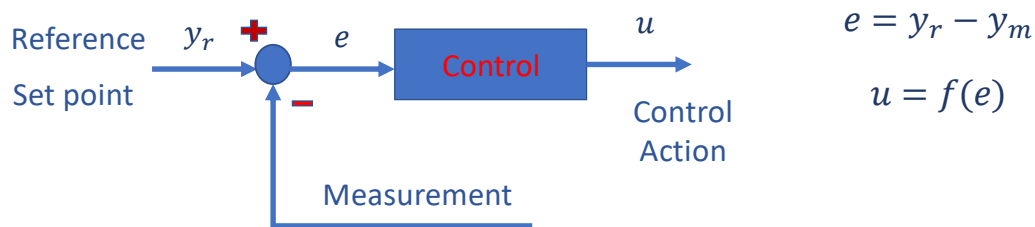


Figure 8.6: Detailed explanation of PID feedback control loop

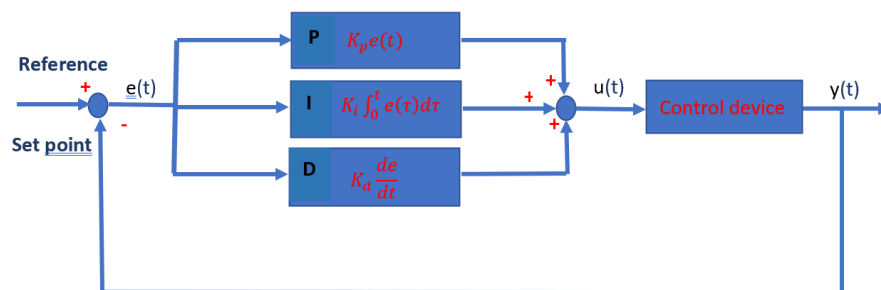


Figure 8.7: Detailed explanation of PID in parallel feedback control loop

Through proper setting of the controls in a PID circuit, relatively quick response with minimal overshoot (passing the set point value) and ringing (oscillation about the set point value) can be achieved.

Improper setting of the PID controls can cause the circuit to oscillate significantly and lead to instability in control. It is up to the user to properly adjust the PID gains to ensure proper performance.

Hereby, the PID controller in parallel is explained in more detail from physical and mathematical points of view. The output of the PID control circuit, $u(t)$, is given as:

$$u = K_p e + K_I \int_0^t e dt + K_d \frac{de}{dt} \quad (8.1)$$

where K_p, K_I, K_d are proportional, integral, and derivative gains. e is the error signal, which is defined as the difference between the set point or reference signal and the process variable or output signal. From here, the control units through their mathematical definitions can be defined and discussed each in a little more detail.

Proportional control is proportional to the error signal and the current values of the signals, denoted by the proportionality; as such, it is a direct response to the error signal generated by the closed loop system:

$$u = K_p e$$

Larger proportional gain results in larger changes in response to the error, and thus affects the speed at which the controller can respond to changes in the system. As a result, increasing the value of K_p results in a faster but more oscillatory response, reducing the steady-state error. Too low values of K_p mean the controller cannot efficiently respond to changes in the system. The proportional control sees the current state of the error signal, i.e., it evaluates the actual error signal.

Integral control goes a step further than proportional gain, as it is proportional to not just the magnitude of the error signal but also the duration of the error, past values of the signals, denoted by the integral.

$$u = K_I \int_0^t e dt$$

Integral control is highly effective at increasing the response time of the system along with eliminating or canceling the steady-state error associated with purely proportional control. In essence integral control sums over the previous error, which was not corrected, and then multiplies that error by K_I to produce the integral response. Thus, for even small, sustained error, larger values of integral value, K_I , response may achieve zero steady-state error. However, due to the fast response of integral control, high gain values can cause significant overshoot of the reference signal or set point value and lead to oscillation and instability. Too low and the system will be significantly slower in responding to changes in the system. An integral part of the control sees at past values of the error signal as it integrates over past errors, i.e., it evaluates past actions of the system.

Derivative control attempts to reduce the overshoot and has potential from the use of proportional and integral control. It determines how quickly the response of the system is changing over time (by looking at the derivative of the error signal, i.e., future values of the signal) and multiplies it by K_d to produce the derivative response.

$$u = K_d \frac{de}{dt}$$

Unlike proportional and integral control, derivative control will slow the response of the system. It considers the error signal trend. It may compensate overshoot and damp out oscillations caused by integral and proportional parts of the control. High gain values cause the system to respond very slowly, and it can place and amplify noise and high frequency oscillation (as the system becomes too slow to respond quickly). Too low and the system is prone to overshooting the reference signal, goals or set point value. However, in some cases overshooting the set point value by any significant amount must be avoided and thus a higher derivative gain (along with lower proportional gain) may be used. The derivative part of the PID controller gives the slope which the error trajectory is forwarded, i.e., it evaluates future tendencies in the control system.

The use of a derivative part works well for errors that show low dynamic changes. A temperature control, for example, is not subject to fast changes. But if the value it is controlled is subject to very high dynamic changes or if the retrieved error is very noisy, its differentiation can result in even higher amplitude than the desired signal, leading to an unstable system.

As result, to implement PID controllers; it is not required a good model of the process. As the PID controller directly acts to the output signal the PID parameters may be tuned to accommodate the output signal with low dependency on the dynamics of the system. Moreover, the PID control may be used to filter disturbances while achieving specific output signal, but as closed loop system zero tracking error is not achievable. Finally, some extra instrumentation may be required to close the system (output signal measurement) and, if necessary, to quantify disturbances.

When using the control law given by Equation 8.1, it follows that a step change in the reference signal will result in an impulse in the control signal. This is often highly undesirable: therefore, derivative action is frequently not applied to the reference signal. This problem can be avoided by filtering the reference value before feeding it to the controller. Another possibility is to let proportional action act only on part of the reference signal. This is called set point weighting.

The transfer function of Eq. 8.1 is known as the ideal transfer function because, in practice, it is not possible to calculate the derivative term, and it is substituted by a pseudo-derivative. In real practical cases, the Eq. 8.1 in the Laplace domain occurs as follows:

$$U(s) = K_p \left(1 + \frac{1}{T_i s} + \frac{T_d}{\alpha T_d s + 1} s \right) E(s) \quad (8.1)$$

Where T_i and T_d are time constants associated with the integral and derivative terms of the controller, and α ($\alpha < 1$) is a derivative filter factor, that takes typical values $0.05 < \alpha < 0.1$. This modification can be interpreted as the ideal derivative action is filtered by a first-order system with a time constant αT_d . The new derivative action:

$$U_d(s) = \frac{K_p T_d s}{\alpha T_d s + 1} E(s)$$

will act as a true derivative just at low frequencies, and its gain at high frequencies is limited by K_p/α . Then the noise at high frequencies will be amplified by at most for this value and not at high values as the ideal case.

The chart below (Table 8.1) explains the effects of increasing the gain of any one of the parameters independently.

Table 8.1: Tuning PID control parameters.

Parameter Increased	Rise Time	Overshoot	Settling Time	Steady-State Error	Stability
K_p	Decrease	Increase	Small Change	Decrease	Degrade
K_i	Decrease	Increase	Increase	Decrease Significantly	Degrade
K_d	Minor Decrease	Minor Decrease	Minor Decrease	No Effect	Improve (for small K_d)

To tuning the parameters of the PID controller, in general, the gains of P, I, and D will need to be adjusted by the user to best serve the system. While there is not a static set of rules for what the values should be for any specific system, following the general procedures should help in tuning a circuit to match one's system and environment. In general, a PID circuit will typically overshoot the set point value slightly and then quickly damp out to reach the set point value.

Manual tuning of the gain settings is the simplest method for setting the PID controls. However, this procedure is done actively (the PID controller turned on and properly attached to the system) and requires some amount of experience to fully integrate. To tune your PID controller manually, first the integral and derivative gains are set to zero. Increase the proportional gain until you observe oscillation in the output. Your proportional gain should then be set to roughly half this value. After the proportional gain is set, increase the integral gain until any offset is corrected for on a time scale appropriate for your system. If you increase this gain too much, you will observe significant overshoot of the SP value and instability in the circuit. Once the integral gain is set, the derivative gain can then be increased. Derivative gain will reduce overshoot and damp the system quickly to the SP value. If you increase the derivative gain too much, you will see large overshoot (due to the circuit being too slow to respond). By playing with the gain settings, you can maximize the performance of your PID circuit, resulting in a circuit that quickly responds to changes in the system and effectively damps out oscillation about the SP value.

While manual tuning can be very effective at setting a PID circuit for your specific system, it does require some amount of experience and understanding of PID circuits and response. The Ziegler-Nichols method for PID tuning offers a bit more structured guide to setting PID values.

Anti-wind up PID controller

Integral windup, also known as **integrator windup** or **reset windup**, refers to the situation in a PID controller where a large change in setpoint occurs (say a positive change) and the integral term accumulates a significant error during the rise (windup), thus overshooting and continuing to increase as this accumulated error is unwound (offset by errors in the other direction). The specific problem is the excess overshooting.

This problem can be addressed by:

- Initializing the controller integral to a desired value, for instance to the value before the problem.
- Increasing the setpoint in a suitable ramp.
- Conditional Integration: disabling the integral function until the to-be-controlled process variable (PV) has entered the controllable region.
- Preventing the integral term from accumulating above or below pre-determined bounds.
- Back calculating the integral term to constrain the process output within feasible bounds.
- Clegg integrator: Zeroing the integral value every time the error is equal to or crosses zero. This avoids having the controller attempt to drive the system to have the same error integral in the opposite direction as was caused by a perturbation.

Integral windup particularly occurs as a limitation of physical systems, compared with ideal systems, due to the ideal output being physically impossible (process saturation: the output of the process being limited at the top or bottom of its scale, making the error constant). For example, the position of a valve cannot be any more open than fully open and cannot be closed any more than fully closed. In this case, anti-windup can involve the integrator being turned off for periods of time until the response falls back into an acceptable range.

This usually occurs when the controller's output can no longer affect the controlled variable, or if the controller is part of a selection scheme and it is selected right.

Integral windup was more of a problem in analog controllers. Within modern distributed control systems and programmable logic controllers, it is much easier to prevent integral windup by either limiting the controller output, limiting the integral to produce feasible output, or by using external reset feedback, which is a means of feeding back the selected output to the integral circuit of all controllers in the selection scheme so that a closed loop is maintained. Figure 8.8 represents PID controller with anti-wind-up action.

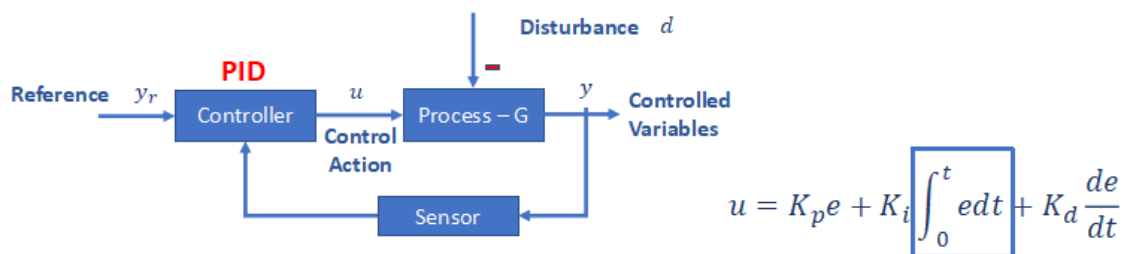


Figure 8.8: Schematic representation of PID controller with anti-wind up.

Example 8.1: PID control of a generic transfer function by tuning PID parameters manually

In this example, it is shown how to tune the PID parameters by insight and understanding gained over previous experiences. If this is the case, for any transfer function it is possible to apply this methodology when the transfer function of the system is simple enough to account for every physical parameter which is involved in the process. The open loop transfer function of the system is the following:

$$G(s) = \frac{1}{s^2 + 3s + 1}$$

```
hold on
grid on
%PID example
```

```
clear all
clc
```

% The transfer function variables. We have the values for the numerator.

```
% and the denominator
```

```
num = [ 1 ];
den = [ 1 3 1];
```

```
% We denote the transfer function as tf.
```

```
Gp = tf(num, den); % setting the transfer function.
```

At this point, it is possible to observe the output of the system in open loop against step unit input. This view gives insight on how the output of the system will increase the performance when the control is added.

```
step(Gp) % step response of the open loop system
```

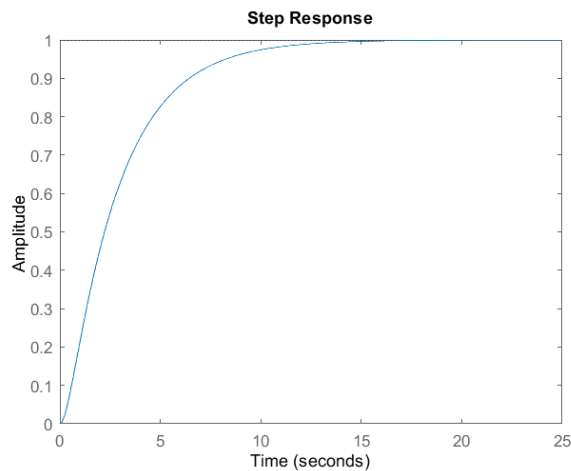


Figure 8.9: Step output of the open loop transfer function $G(s) = \frac{1}{s^2+3s+1}$

It is observed (figure 8.9) that the output signal reaches the reference in about 15 seconds, but the signal looks smooth and feasible.

In this example, the sensor transfer function, feedback transfer function represented by H in this case, is modeled as unity.

```
H = 1; % feedback transfer function. It shows the expected.
      % unit response.
```

```
Gcl = feedback(Gp, H); % feedback of the plant.
step(Gcl) % step response
```

The response of the closed loop system can be plotted without control and now, the control is initiated.

```
%%
Kp = 1; % proportional constant
Ki = 0; % integral constant
Kd = 0; % derivative constant

Gc = pid(Kp,Ki,Kd); % the controller.
```

```
Gclcontrol = feedback(Gc*Gp, H); % transfer function with control.
step(Gclcontrol) % step response of the transfer function with control.
grid on
```

The effects of changing P, I, and D parameters are shown against an input step response, and PID controller parameters are tuned to check the influence in the closed loop system.

If the values of K_p are varied in the program and it can be seen how the increase of these values influences the output of plotted results after running the program for every change. The initial value of K_p is 1 but it can be changed to different values, for example, 2, 5, 10, 15, 20, 25, 30, 40 in the command line of the program. To see the effect of these variations in the K_p values in one graph, it is added the 'hold on' function immediately after step M. This is to be able to see each plot in one figure after changing the value.

When the program is run after adding the hold on function, the output is shown in figure 8.10.

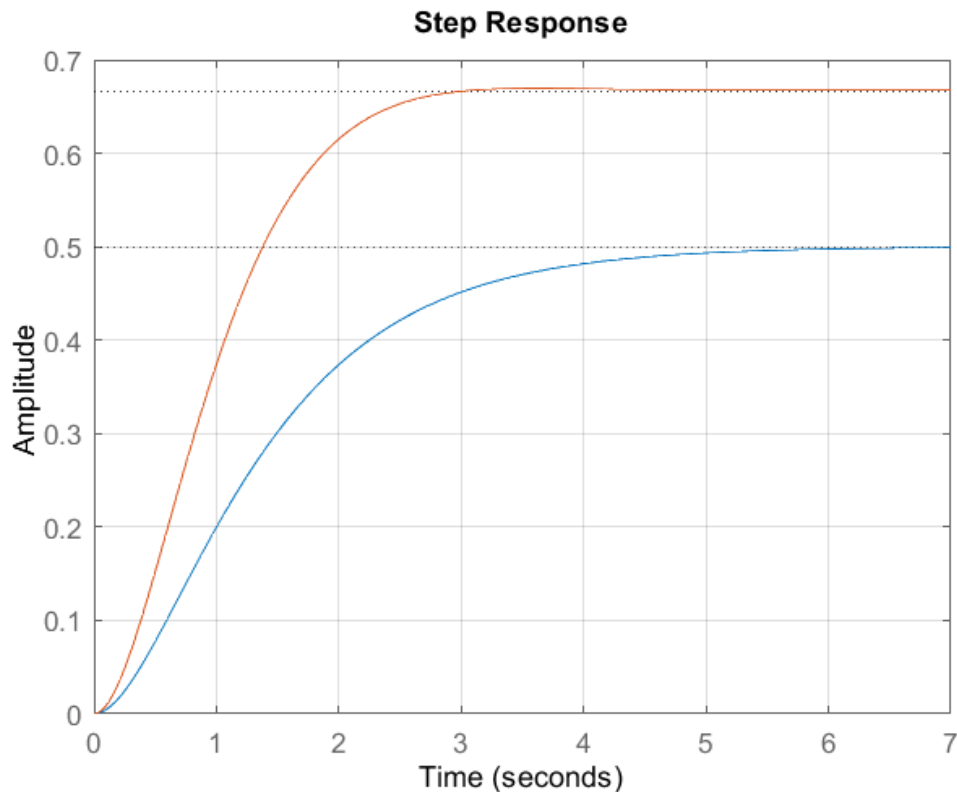


Figure 8.10: Plot of step responses when K_p is changed from 1 to 2.

In the figure above, it is obtained a plant with the feedback and K_p equal to 1 which is the blue plot, and it goes asymptotic to 0.5 in about 5 secs. The red plot corresponds to K_p equal to 2. .

It is introduced a peak controller. The peak controllers make the system go faster and to change the steady-state error without changing other characteristics of the output signal.

To see the steady-state characteristics visible in our plot, it is possible if it is right-clicked any point within the graph (Figure 8.11).

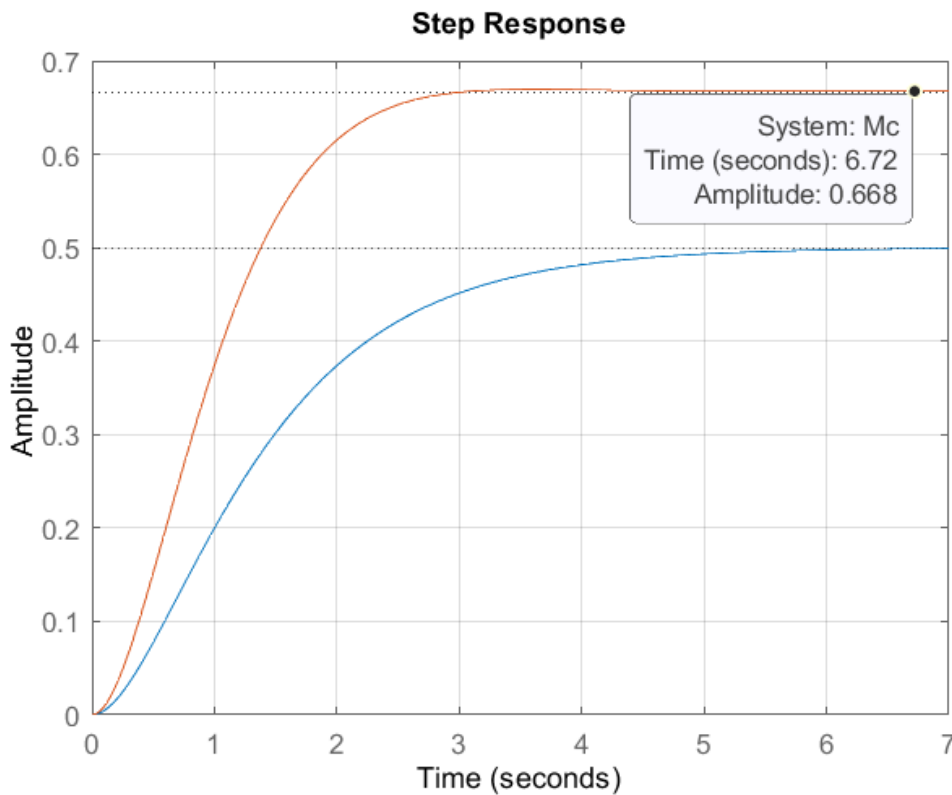


Figure 8.11: Steady state characteristics of plots.

When it is hovered over the dots, it is seen that as time goes to infinity, the blue plot goes to 0.5 and the red plot to 0.668. It is required the steady state of the red plot go to 1, for this purpose the parameters of the controller are tuned for. In this case, there exists some steady state-state error which is necessary to fix. To do so, it is increased or tuned the K_p until it is attained.

In Figure 8.12, the outputs of the system corresponding to the values of K_p previously mentioned are plotted in the same graph. The increased of K_p allows higher frequencies into the systems, but the system becomes more oscillatory.

If the derivative part of the control is introduced, i.e., the K_d value is tuned, the effect will make the oscillations to be squared in the edges as it can be shown in the output (Figure 8.13). It is possible to tune both parameters with the ideas mentioned within this chapter.

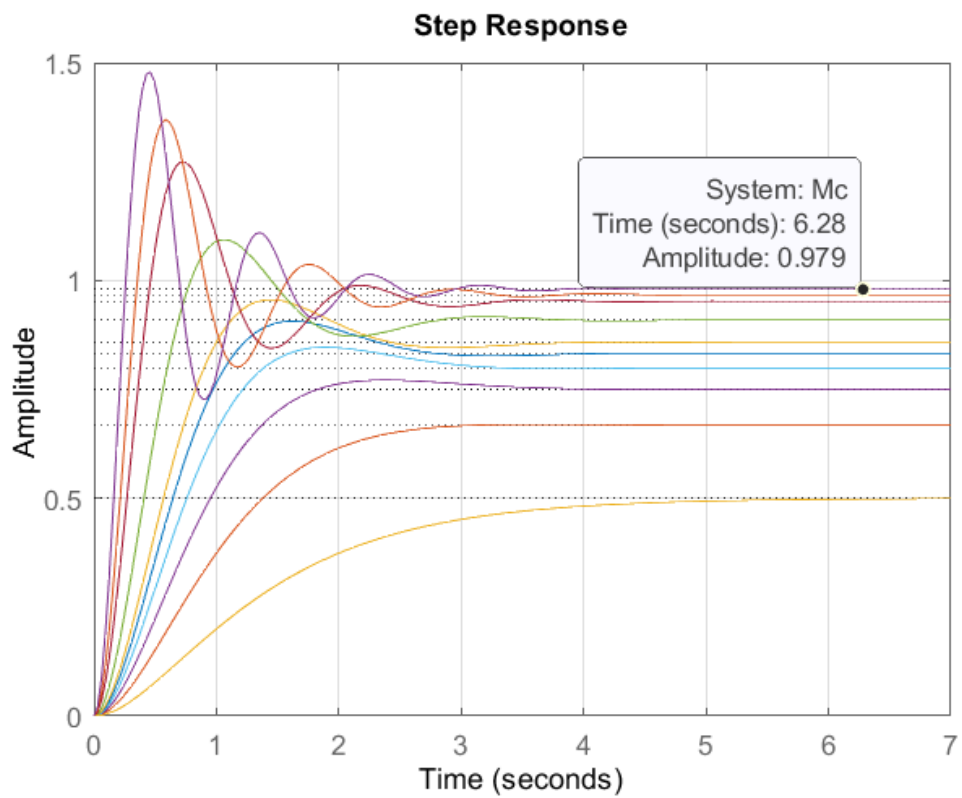


Figure 8.12: Effect of changing K_p value in amplitude 0.979 for $K_p = 50$ and plot of previous mentioned K_p parameters.

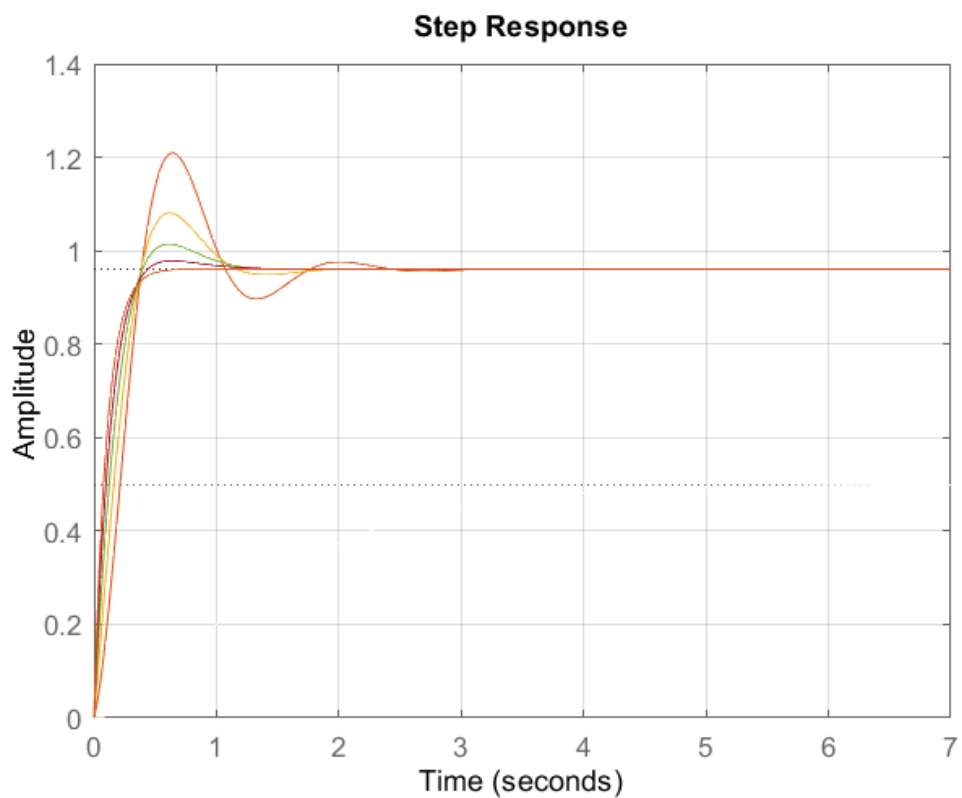


Figure 8.13: Effect of changing K_d values for a fixed K_p .

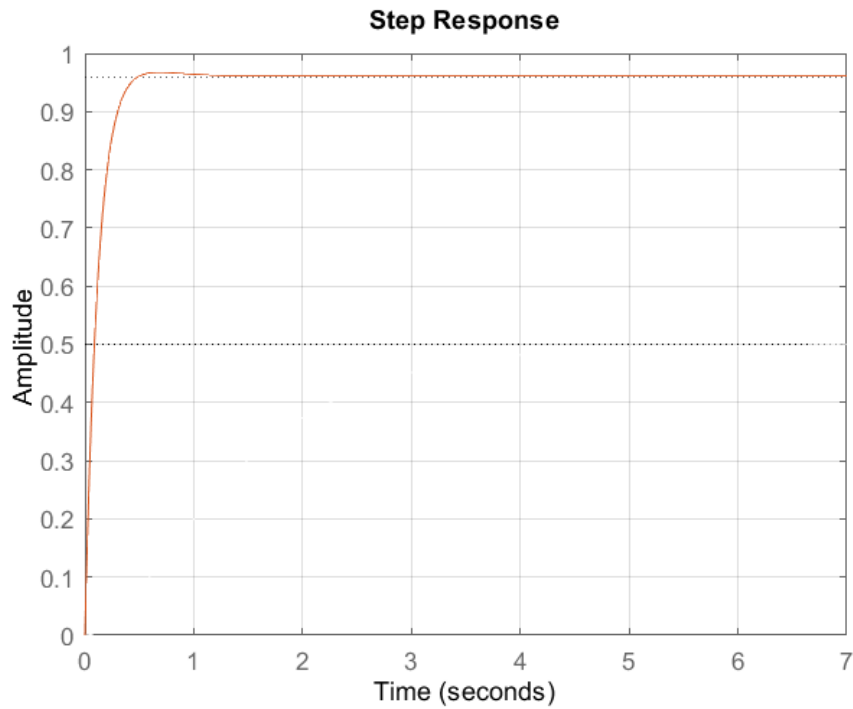


Figure 8.14: Final plot with required output response $K_d = 8$ and the $K_p = 24$

The system stabilizes when the K_d is 8 and the K_p is 24. When the step response is outputted, the control response went to 0.96 but we require it to be at 1 since it is a unit response. The integration part enhances these characteristics, so adding and tuning the K_i value less steady error can be achieved as observed in Figures 8.13 and 8.14.

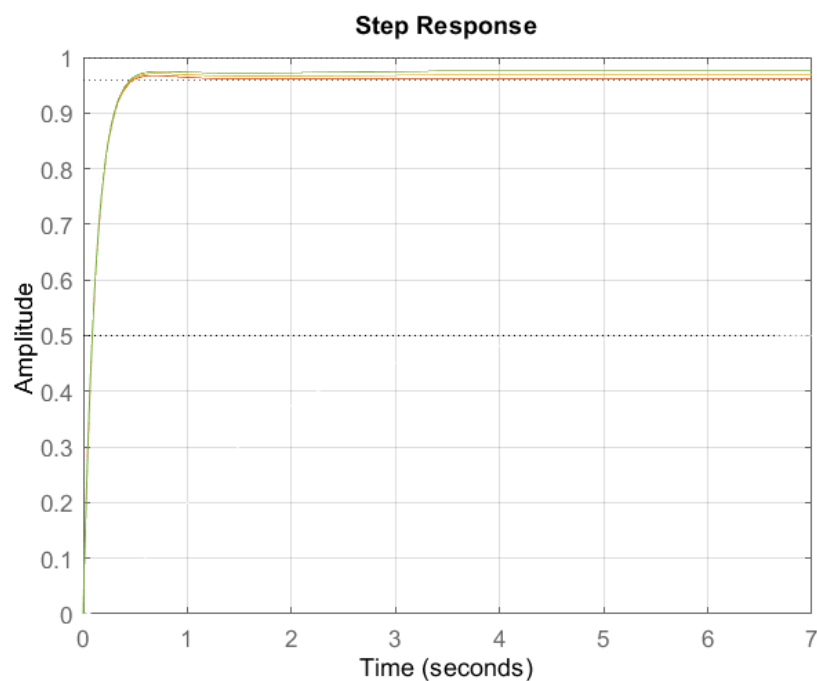


Figure 8.15: Effect of changing K_i with constants K_p and K_d .

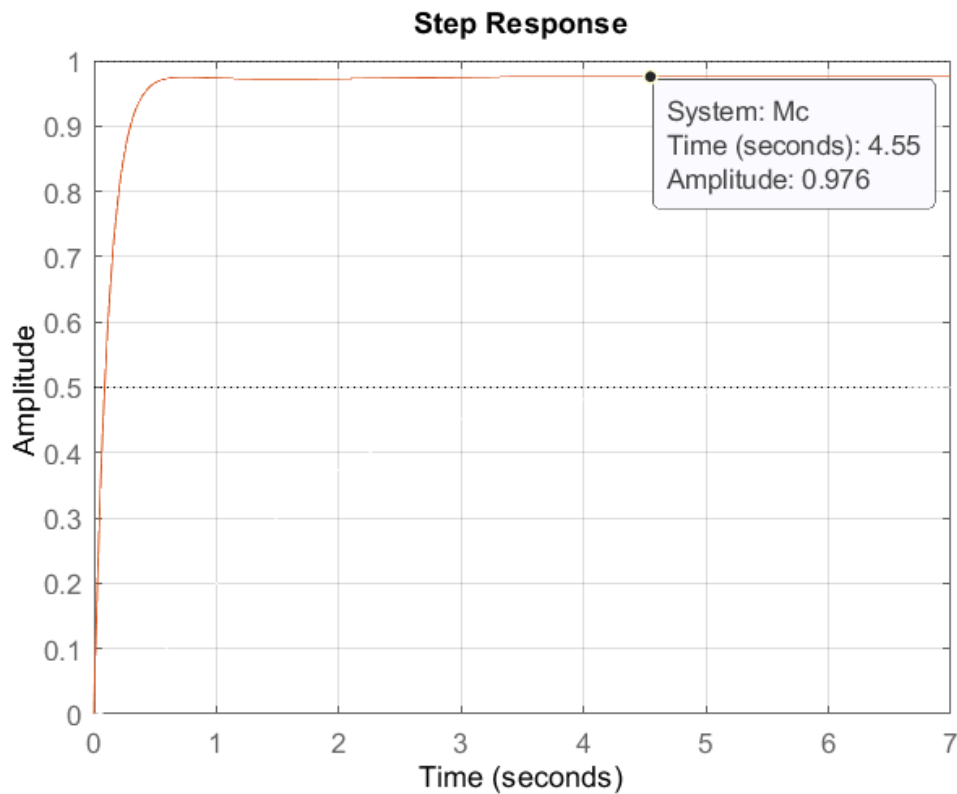


Figure 8.16: Final plot with required output responses $K_p=24$, $K_d=8$ and $K_i=2$.

The PID controller is the final values of K_p , K_d , and K_i . Adding the PID function to the closed loop system the resulted transfer function which represents the closed loop system is obtained. With the command *Gclcontrol* the resulted transfer function is taken.

Example 8.2: PID Control design of a DC motor by root locus in MATLAB [1]

The transfer function of a small armature-controlled DC motor may be modelled as (see chapter 2 to know how to get the model):

$$H(s) = \frac{\theta(s)}{V(s)} = \frac{K_T}{s((Js + B)(Ls + R) + K_T K_e)} \left[\frac{\text{rad}}{V} \right]$$

To define an m-file in MATLAB (the program is shown at the end of this chapter, please follow it at the same time with the explanations) to model the system and design the controller, it is first defined typical parameters of the motor, which are:

% DC motor Transfer Function

```
J = 3.2284e-6; % Rotacional inertia [Nms2/rad]
b = 3.5077e-6; % Viscous friction [Nms/rad]
KT = 0.0274; % Torque constant [Nm/A]
Ke = 0.0274; % back emf constant [Vs/rad]
R = 4; % Armature resistance [Ω]
L = 2.75e-6; % Armature inductance [H]
```

$$s = \text{tf('s');}$$

$$P_motor = K/(s*((J*s+b)*(L*s+R)+K^2));$$

The structure of the system is as follows (figure 8.17):

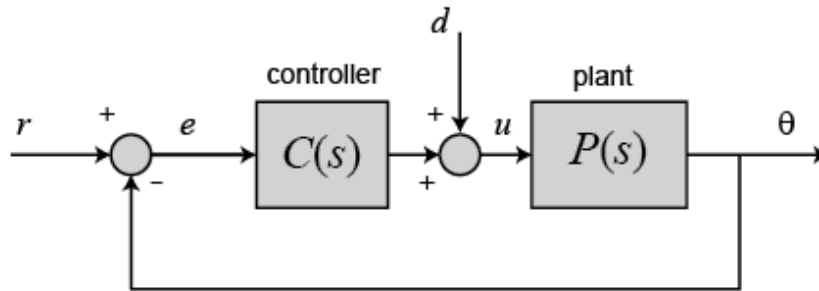


Figure 8.17: Feedback control loop for an armature-controlled DC motor.

The specifications of the system output for controller design purposes are the following:

With a 1-radian step reference, the design criteria are the following:

- Settling time less than 0.040 seconds; $t_s = \frac{4}{\sigma} \rightarrow \sigma < 100$.
- Overshoot less than 16%; $M_p = e^{\frac{-\zeta\pi}{\sqrt{1-\zeta^2}}} \rightarrow \zeta > 0.5$.
- No steady-state error, even in the presence of a step disturbance input.

The variables in the “sgrid” command are the damping ratio ($\zeta = 0.5$) and the natural frequency ($\omega_n = 0$). The variable in the sgrid command is the damping ratio term (ζ). The ζ and ω_n used above correspond to an overshoot of 16% and a settling time of 0.040 seconds, respectively, for a canonical second-order system. Even though the transfer function of the motor is third order, it will be explained that these second order-based definitions will work well. No value is given for ω_n since we have no requirement on rise time.

`% Drawing the open-loop root locus K*P_motor`

```
rlocus(P_motor)
title('Root Locus - P Control')
sgrid(.5, 0)
sigrid(100)
```

The function sigrid needs to be incorporate to the library, it is possible to be downloaded from MATLAB exchange files on the internet.

Drawing the open-loop root locus K*P_motor:

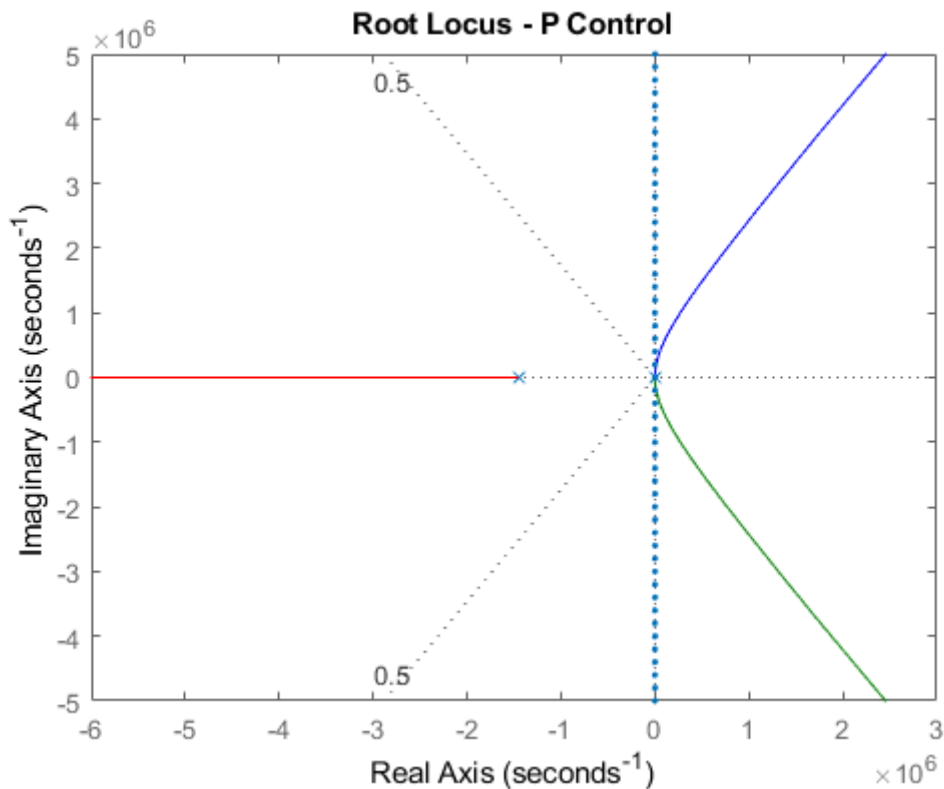


Figure 8.18: Root locus P-control for an armature-controlled DC motor.

From the above figure, the two open-loop poles near the origin cannot be distinguished because the scale of the axes is set to show the third pole which is much farther to the left than the other two poles. The MATLAB command `pole` can be employed to determine the exact values of the open-loop poles.

```
% Poles TF
```

```
poles = pole(P_motor)
```

```
poles =
 1.0e+06 *
 0
 -1.4545
 -0.0001
```

The open-loop pole located very far to the left (further than $1e+06$) does not affect the closed-loop dynamics unless very large gains are used. These large gains place two of the closed-loop poles in the right-half complex s -plane where the system becomes unstable. Since it is not used gains that will make the closed-loop system unstable, the pole can be neglected by performing a model reduction.

Model reduction

In general, the real part of a pole indicates how quickly the transient portion of the corresponding mode decays to zero (assuming negative real part). Therefore, a transfer function which has one (or more) poles much farther to the left in the complex plane (more negative) than the other poles, their effect on the dynamic response will be hidden by the slower, more dominant poles. In the case of the motor position example, the transient closed-loop response for small gains will not be affected much by the open-loop pole at $-1.45e6$. The correct way to neglect this pole to maintain its steady-state contribution is to keep the DC gain of the transfer function the same, as follows:

$$H(s) = \frac{G(s)}{\frac{s}{p} + 1} \approx G(s) \text{ for } s \ll p$$

As shown above, the poles of the open-loop transfer function can be identified using the MATLAB command `pole`. The two poles that dominate are difficult to identify from above because of the notation, but they can be seen more clearly by recognizing that they are the first and third elements of the resulting vector which we have named `poles`.

```
% Model reduction

poles(1), poles(3)
poles = pole(P_motor);
rP_motor = minreal(P_motor*(s/max(abs(poles)) + 1))
pole(rP_motor)

poles(1), poles(3)

ans = 0

ans = -59.2260
```

To construct the reduced transfer function to neglect the pole at $-1.45e6$ without affecting the steady-state behavior of the system, the MATLAB command “`minreal`” is introduced, adding the following commands to the m-file, and re-running the program.

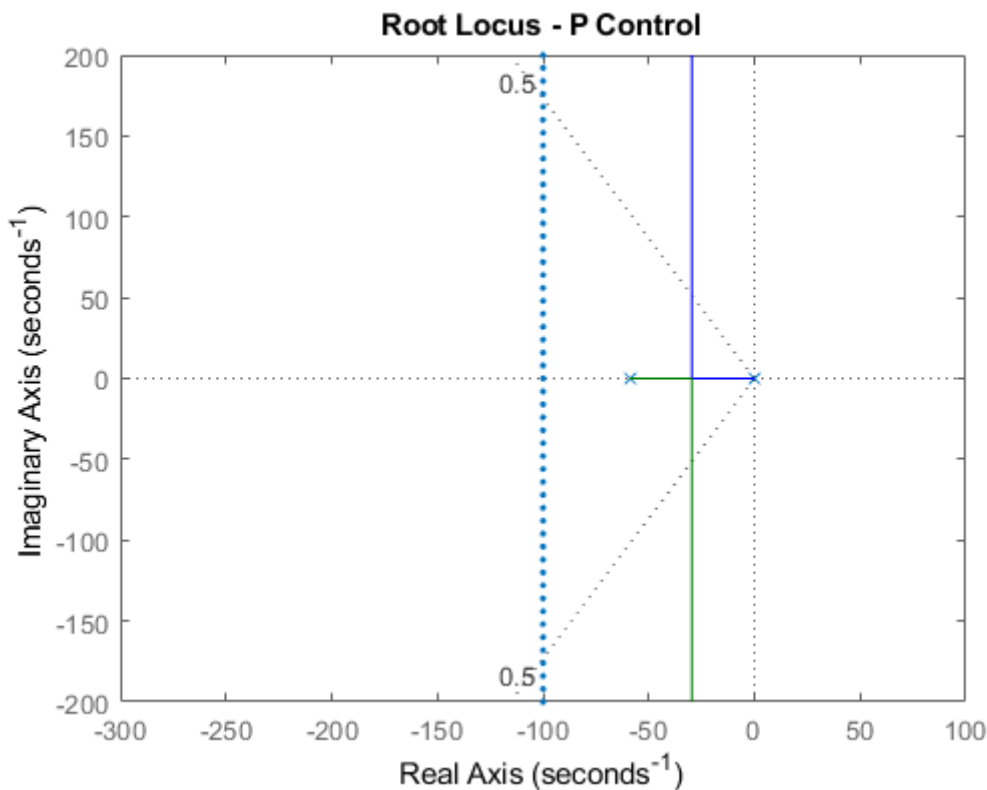


Figure 8.19: P-control for an armature-controlled DC motor with model reduction.

When the original root locus near the origin is examined, it would closely approximate the locus of the reduced transfer function shown above. It can be seen from this new plot that the closed-loop poles are never fast enough to meet the settling time requirement (that is, they never move to the left of the $\sigma = 100$ vertical line). Also, it is needed an integrator in the controller (not just in the system) to remove the steady-state error due to a constant disturbance.

```
% rlocus

rlocus(rP_motor)
title('Root Locus - P Control')
axis([-300 100 -200 200])
sgrid(.5, 0)
sigrid(100)
```

Integral control

It is using integral control to remove the steady-state error due to a constant disturbance. Note that this adds a $1/s$ term to the forward path of the system. When this m-file is run and a plot like the one shown in figure 7.21 is obtained. From this root locus it is observed that the closed-loop system under integral control is never stable, therefore, a different controller must be employed.

```
% Integral control
```

```
C = 1/s;
```

```

rlocus(C*rP_motor)
title('Root Locus - I Control')
axis([-300 100 -200 200])
sgrid(.5, 0)
sigrid(100)

```

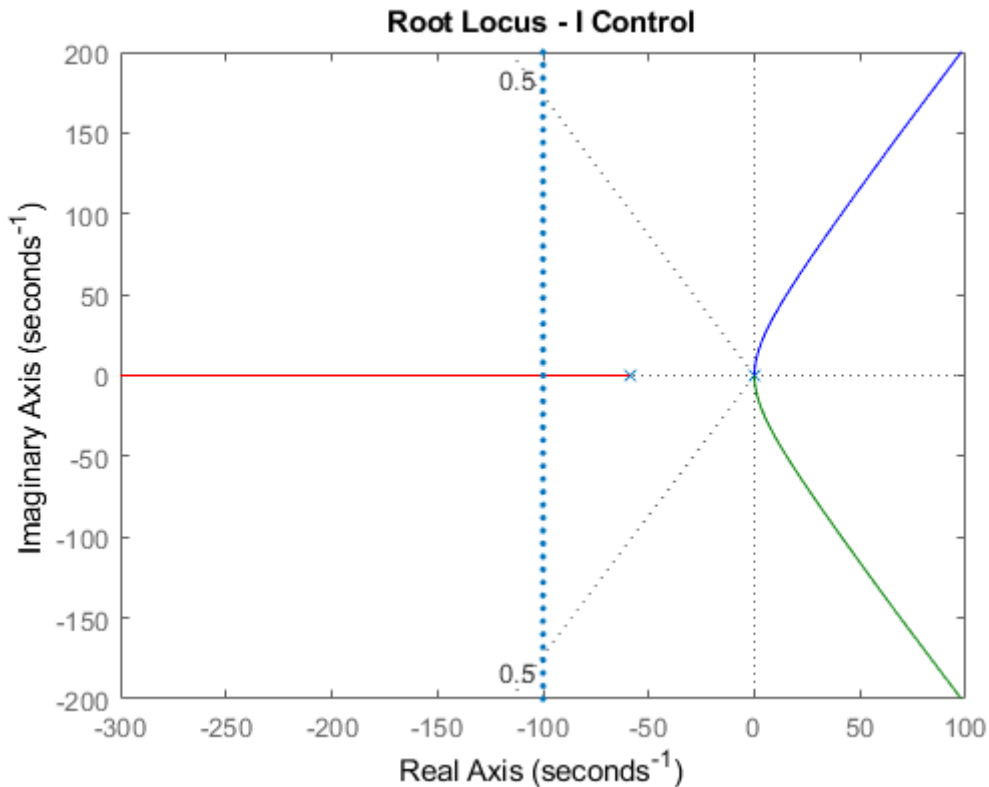


Figure 8.20: Root locus I-control for an armature-controlled DC motor.

PI Controller

At this point, the integral controller is switched to a PI controller. Using PI instead of I control adds a zero in the transfer function of the closed loop respect to the open-loop system. This zero is placed at $s = -20$. It must lie between the open-loop poles of the system in this case so that the closed-loop system will be stable. Changing the commands to define the controller in the m-file to the following. If the program is re-run and a plot like the one shown in figure 8.21 is obtained. The system is stabilized, and it is achieved zero steady-state error to a constant disturbance, but the system is still not fast enough.

```

% PI Control

C = (s + 20) / s;
rlocus(C*rP_motor)
title('Root Locus - PI Control')
axis([-300 100 -200 200])
sgrid(.5, 0)
sigrid(100)

```

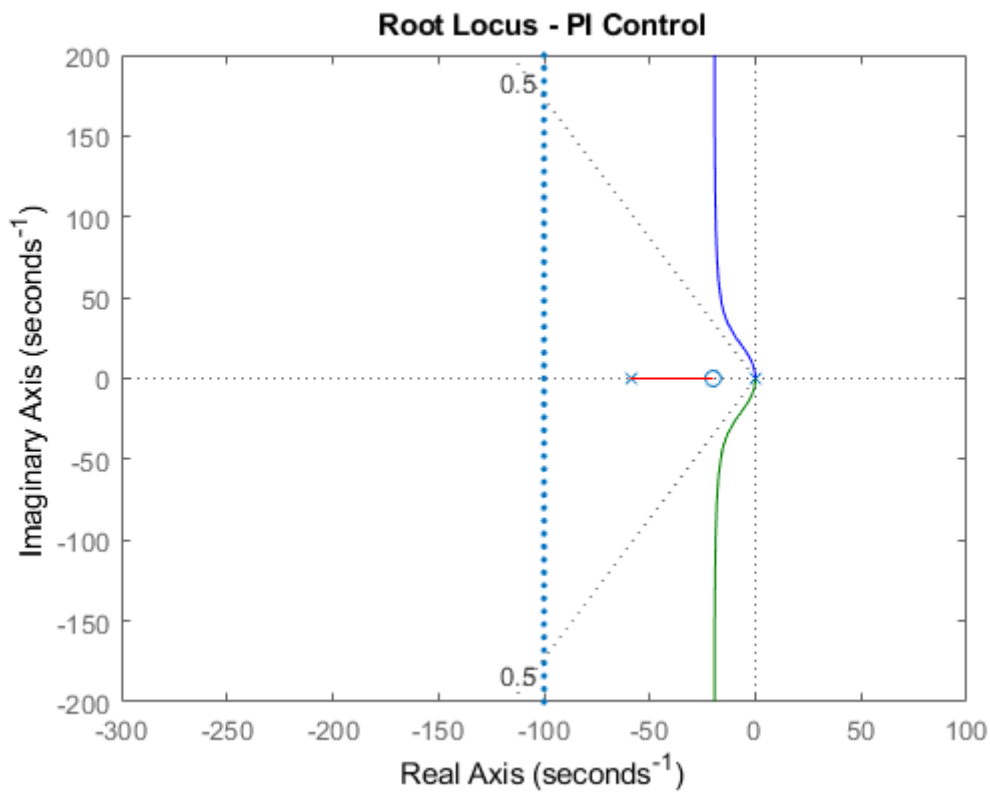


Figure 8.21: Root locus PI-control for an armature-controlled DC motor.

PID Controller

To pull the root locus further to the left, to make it faster, it is needed to place a second open-loop zero, resulting in a PID controller. After some experimentation, it is placed the two PID zeros at $s = -60$ and $s = -70$. The program is updated with the corresponding commands and re-running it and it is generated a plot like the one shown in figure 8.22.

```
% PID Control
```

```
C = (s + 60)*(s + 70) / s;
rlocus(C*rP_motor)
title('Root Locus - PID Control')
axis([-300 100 -200 200])
sgrid(.5, 0)
sigrid(100)
```


The root locus for this further reduced system with controller is shown in figure 8.23. Note how closely it resembles the root locus without the pole-zero cancellation. Even though the one open-loop zero was cancelled, the other open-loop zero remains in the closed-loop transfer function and cannot be neglected. The effect of an additional zero (if there is no cancellation) is in general to speed up the response and add overshoot.

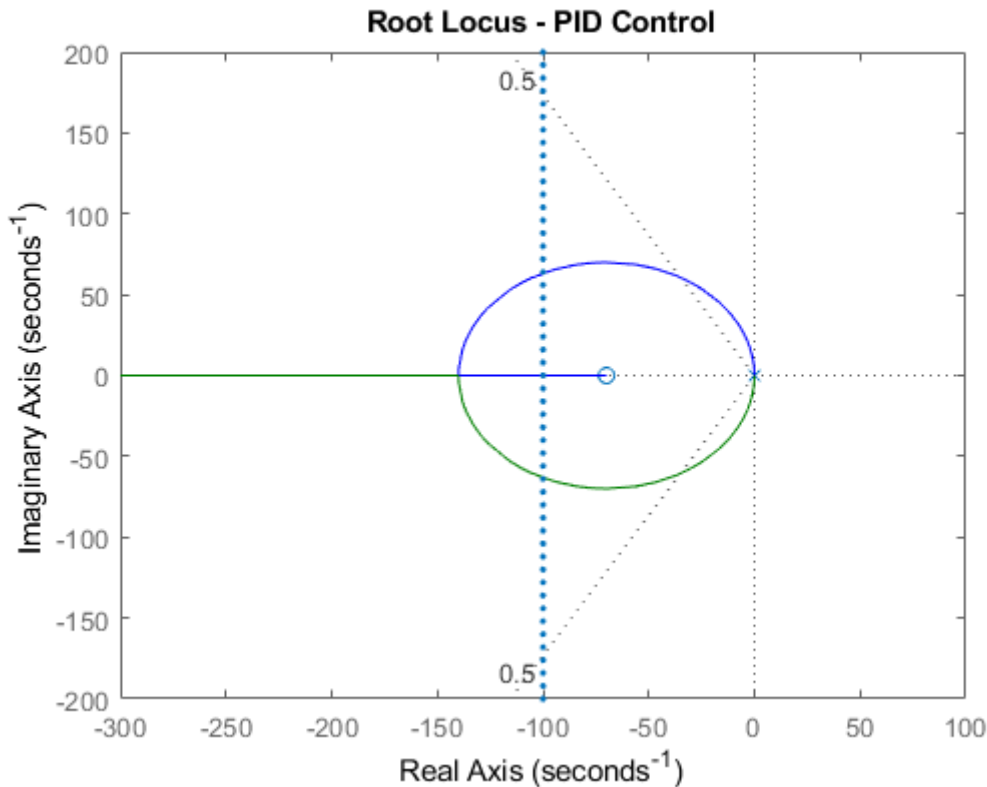


Figure 8.23: Root locus PID-control for an armature-controlled DC motor with model reduction.

Determining gain using rlocfind command

As control specifications, it is required the settling time and the overshoot to be as small as possible, particularly because of the effect of the extra zero. Large damping corresponds to points on the root locus near the real axis. A fast response corresponds to points on the root locus far to the left of the imaginary axis. To find the gain corresponding to a point on the root locus, the `rlocfind` command provides that functionality.

```
% Determining gain using rlocfind command
```

```
[k,poles] = rlocfind(rsys_ol)
```

Then on the plot and it is possible to select a point on the root locus on left side of the loop, close to the real axis as shown in figure 7.25 with the small + marks. This will ensure that the response will be nearly as fast as possible with minimal overshoot. These pole locations indicate that the response would have almost no overshoot if the system were a canonical second-order system. However, the presence of the zero will add some overshoot.

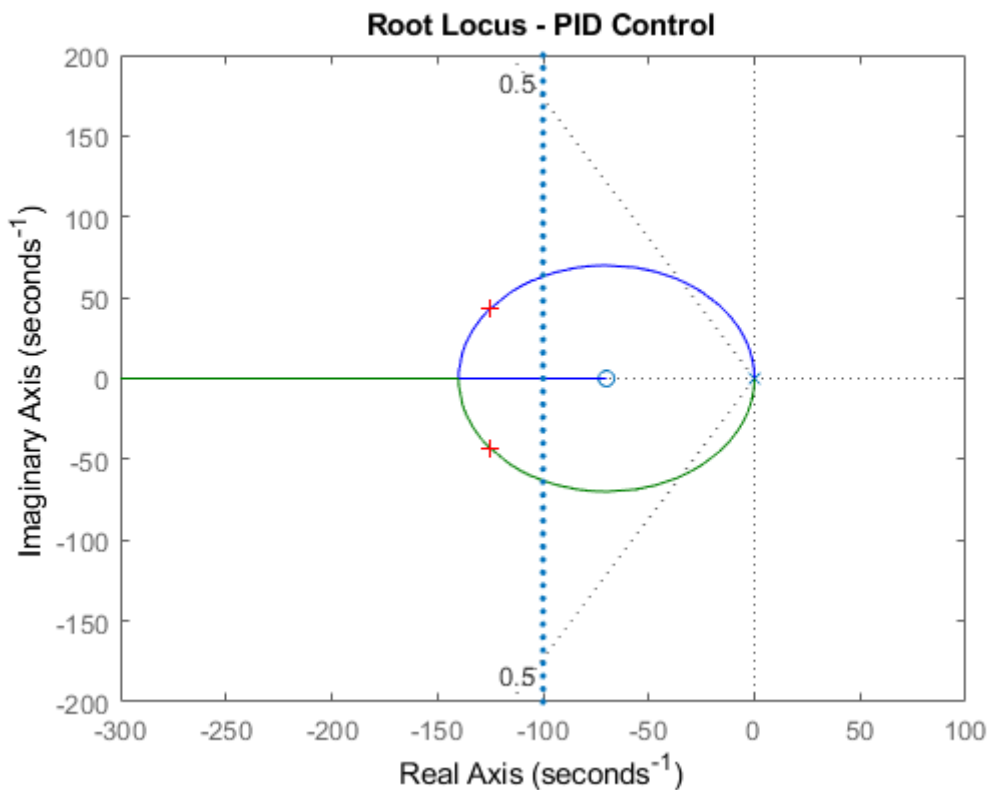


Figure 8.24: Root locus PID-control for an armature-controlled DC motor.

After doing this, in the command window appears the following sentence:

Select a point in the graphics window

Choosing the point in the plot, the selected point appears in the command window:

```
selected_point = -1.2417e+02 + 4.3344e+01i
k = 0.1175
poles = 1.0e+02 *
-1.2466 + 0.4373i
-1.2466 - 0.4373i
```

Note that the values returned in the MATLAB command window may not be the same but should at least have the same order of magnitude. The step response plot for the reference and the output to external unity step disturbance with this specific controller and loop gain by executing the following code.

```
% step response plots for the reference and disturbance with this
% specific controller and loop gain
```

```
sys_cl = feedback(k*rsys_ol,1);
t = 0:0.0001:0.1;
step(sys_cl, t)
grid
ylabel('Position, \theta (radians)')
```

```

title('Response to a Step Reference with PID Control')

% Response step disturbance

dist_cl = feedback(P_motor,k*C);
figure;step(dist_cl, t)
grid
ylabel('Position, \theta (radians)')
title('Response to a Step Disturbance with PID Control')

```

These commands should produce the plots in figures 8.25 and 8.26 where the annotations to the figures are added by choosing Characteristics from the right-click menu of each of the plots.

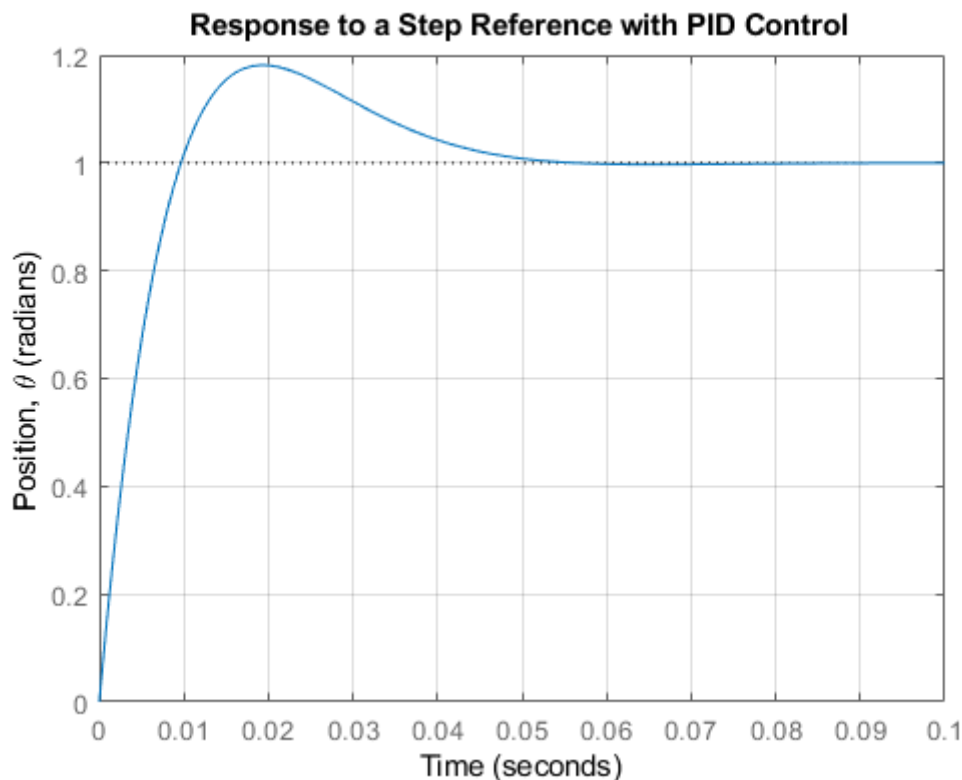


Figure 8.25: Step response of PID-control for an armature-controlled DC motor.

Step response system with disturbance.

It can be appreciated that in response to a step reference the system has an overshoot of approximately 14%, a settling time just under 0.04 seconds, and no steady-state error. Also, the response to a step disturbance reaches a steady-state value of zero. Therefore, all the design requirements have been met.

In this example the zeros of our compensator are placed such as to reshape the root locus so that the closed-loop poles could be placed in the region of the complex plane that would achieve the given design requirements. It is in general helpful to understand the principles

of how the root locus is drawn, however, MATLAB can be very helpful in refining the design and verifying the resulting performance.

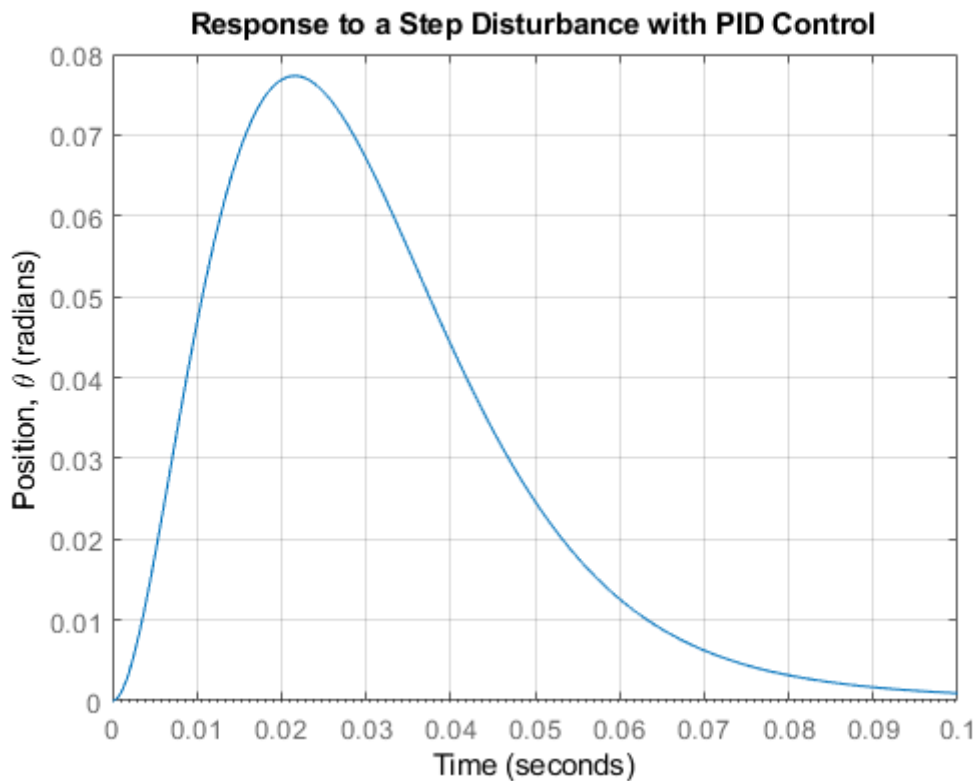


Figure 8.26: Step response PID-control for an armature-controlled DC motor with disturbances.

Exercises:

1. Suppose that at $t=0.5s$ the disturbance considered in Example 2 with the PID controller is acquired. Show the step response of the system considering the disturbance and how it is damped by the PID controller.

References:

[1]<https://ctms.engin.umich.edu/CTMS/index.php?example=MotorPosition§ion=ControlRootLocus>

[2] Wang, Liuping. (2020). PID Control System Design and Automatic Tuning using MATLAB/Simulink. 10.1002/9781119469414.

MATLAB program of PID control of a DC motor [1]

```

% DC motor Transfer Function

J = 3.2284E-6;
b = 3.5077E-6;
K = 0.0274;
R = 4;
L = 2.75E-6;
s = tf('s');
P_motor = K/(s*((J*s+b)*(L*s+R)+K^2));

% Drawing the open-loop root locus K*P_motor

rlocus(P_motor)
title('Root Locus - P Control')
sgrid(.5, 0)
sigrid(100)

% Poles TF

poles = pole(P_motor)

% Model reduction

poles(1), poles(3)

poles = pole(P_motor);
rP_motor = minreal(P_motor*(s/max(abs(poles)) + 1))

pole(rP_motor)

% rlocus

rlocus(rP_motor)
title('Root Locus - P Control')
axis([-300 100 -200 200])
sgrid(.5, 0)
sigrid(100)

% Integral control

C = 1/s;
rlocus(C*rP_motor)
title('Root Locus - I Control')
axis([-300 100 -200 200])
sgrid(.5, 0)
sigrid(100)

% PI Control

C = (s + 20) / s;
rlocus(C*rP_motor)
title('Root Locus - PI Control')
axis([-300 100 -200 200])
sgrid(.5, 0)

```

```

sigrid(100)

% PID Control

C = (s + 60)*(s + 70) / s;
rlocus(C*rP_motor)
title('Root Locus - PID Control')
axis([ -300 100 -200 200])
sgrid(.5, 0)
sigrid(100)

% Reduced PID Control

rsys_ol = minreal(C*rP_motor, 0.1);
rlocus(rsys_ol)
title('Root Locus - PID Control')
axis([ -300 100 -200 200])
sgrid(.5, 0)
sigrid(100)

% Determining gain using rlockfind command

[k,poles] = rlocfind(rsys_ol)

% step response plots for the reference and disturbance with this
% specific controller and loop gain

sys_cl = feedback(k*rsys_ol,1);
t = 0:0.0001:0.1;
step(sys_cl, t)
grid
ylabel('Position, \theta (radians)')
title('Response to a Step Reference with PID Control')

% Response step disturbance

dist_cl = feedback(P_motor,k*C);
figure;step(dist_cl, t)
grid
ylabel('Position, \theta (radians)')
title('Response to a Step Disturbance with PID Control')

```

Chapter 9: Control Structures

This chapter covers some control structures that provide more refined but complex solutions to certain system configurations. These are the following:

- **Feed-forward control.**
- **Two degrees of freedom control.**
- **Supervisory control.**
- **Hierarchical control.**
- **Cascade control.**

9.1 Feed-forward control

A **feed-forward** is an element or pathway within a control system that passes a controlling signal from a source in its external environment to a load elsewhere in its external environment. This is often a command signal from an external operator.

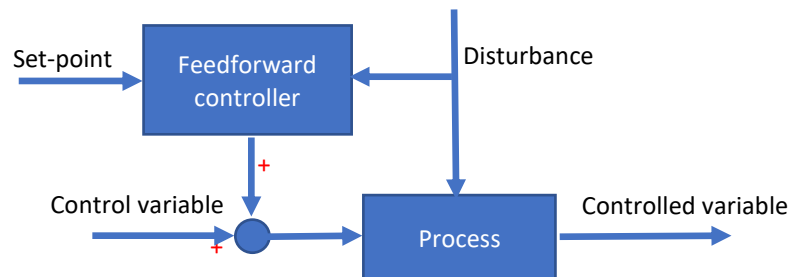


Figure 9.1: Feedforward Controller

Figure 9.1 shows a Feedforward Controller, which manipulates two input variables for the system. The feedforward controller is entered with a set point, an input variable or goal (note the three are the same also, reference and command denote the identical concept), and outputs signals to the manipulated variables and disturbances, which are the objectives of the control, and the output-controlled variable.

A feed-forward control system of the type shown in figure 9.1 responds to its control signal in a pre-defined way without responding or noticing how the load reacts (does not have feedback from the output, where normally the load is situated); in contrast, a system that also has feedback, which adjusts the input to take account of how it affects the load (output signal and physical variable that interacts with the system), and how the load itself may vary unpredictably, is considered to belong to the external environment of the system.

In a feed-forward system, the control variable adjustment is not error-based. Instead, it is based on knowledge about the process in the form of a mathematical model of the process and knowledge about, or measurements of, the process disturbances.

Some prerequisites are needed for a control scheme to be reliable by pure feed-forward without feedback: the external command or controlling signal must be available, and the effect of the output of the system on the load should be known (that usually means that the load must be predictably unchanging with time). So, normally, a feedforward controller may be found associated with a feedback loop.

9.2 Two-degrees of freedom control

Two-degree-of-freedom (2DOF) controller is a 2DOF controller composed of a serial compensator and feedforward compensator. In general terms, the serial compensator deals with tracking the reference signal, while the feedforward compensator deals with tracking the reference value. Also, a noise filter may be incorporated. Any kind of combination of controllers may be considered to deal with the structure, including PID or advanced controllers. The rationale of operation associated with both the inner and outer controllers determines the need for good performance for disturbance attenuation (regulation) as well as set-point following (tracking) [1].

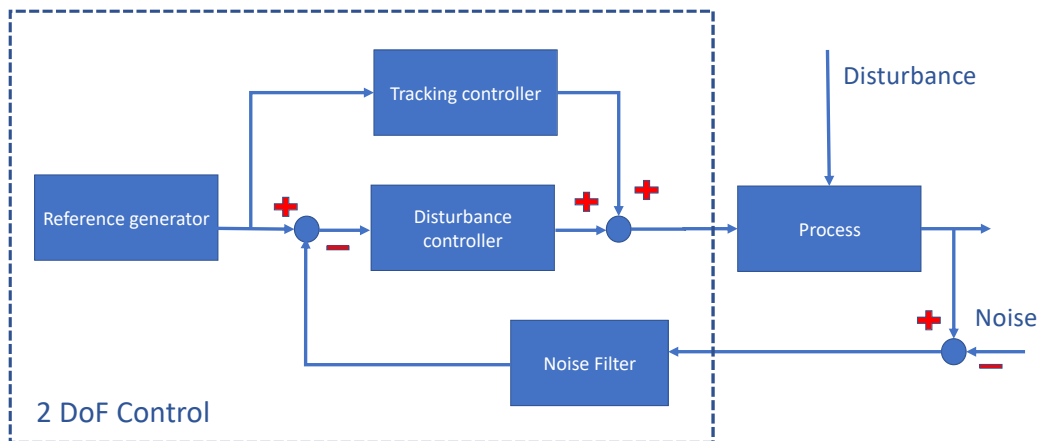


Figure 9.2: 2DOF control loop.

9.3 Supervisory control

Supervisory control is a usual term for controlling several individual controllers or control loops, namely, distributed control systems, which may be implemented in standalone [1] or distributed [2] systems. It describes a high level of overall monitoring of individual process controllers, which is not necessary for the operation of each individual controller, but gives the operator a complete plant process view, and allows integration of operation between controllers.

A more specific use of the term is for a Supervisory Control and Data Acquisition System, or SCADA, which refers to a specific class of system for use in process control, often on small and remote applications such as pipeline transport, water distribution, or wastewater utility system stations [2].

Supervisory control relies on a main CPU to provide setpoint values while one or more ‘regulatory’ controllers perform the continuous, real-time feedback and output loops for the final control elements.

Supervisory control often proceeds in one of the following two forms: On one hand, the controlled machine or process continues autonomously. It is observed from time to time by a human who, when seeing it necessary, intervenes to modify the control algorithm in some way. On the other hand, the process accepts an instruction, carries it out autonomously, reports the results, and awaits further commands. With manual control, the operator interacts directly with a controlled process or task using switches, levers, screws, valves, etc., to control actuators [3].

This concept was incorporated into the earliest machines, which sought to extend the physical capabilities of operators and managers. In contrast, with automatic control, the machine adapts to changing circumstances and makes decisions in pursuit of some goal, which can be as simple as switching a heating system on and off to maintain a room temperature within a specified range [3]. In [4], supervisory control is defined as follows: "In the strictest sense, supervisory control means that one or more human operators are intermittently programming and continually receiving information from a computer that itself closes an autonomous control loop through artificial effectors to the controlled process or task environment."

As mentioned before, supervisory control may be applied to standalone systems or distributed systems. In this case, a **control closed loop with supervisory control** (Figure 9.3) is introduced.

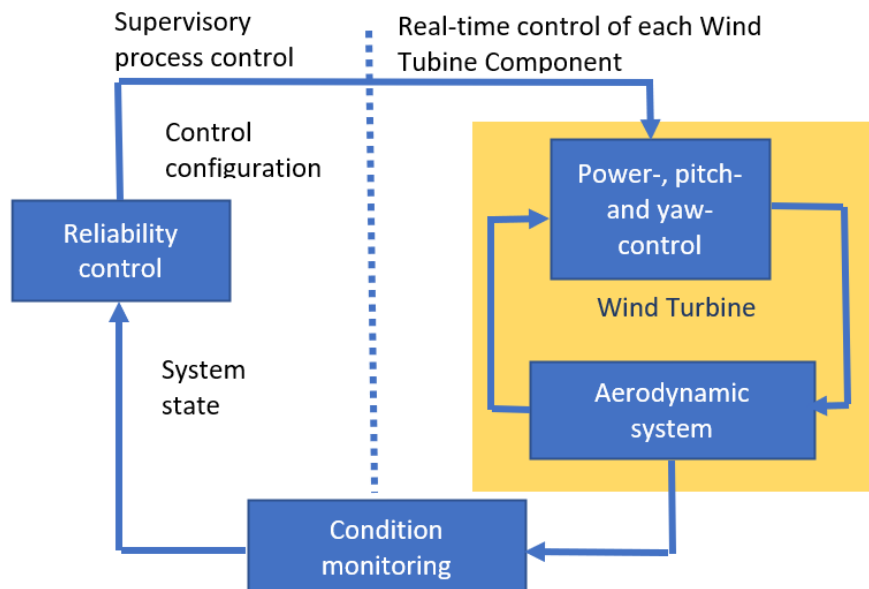


Figure 9.3: General scheme of supervisory control of standalone wind turbine components.

Figure 9.3 shows a standalone control structure. It is composed of the control of the different physical variables of the wind turbine, such as the pitch and yaw angles of the blades and the power system. These variables are monitored and passed to the central unit, called reliable control, which gives an overview of the system and checks its reliability.

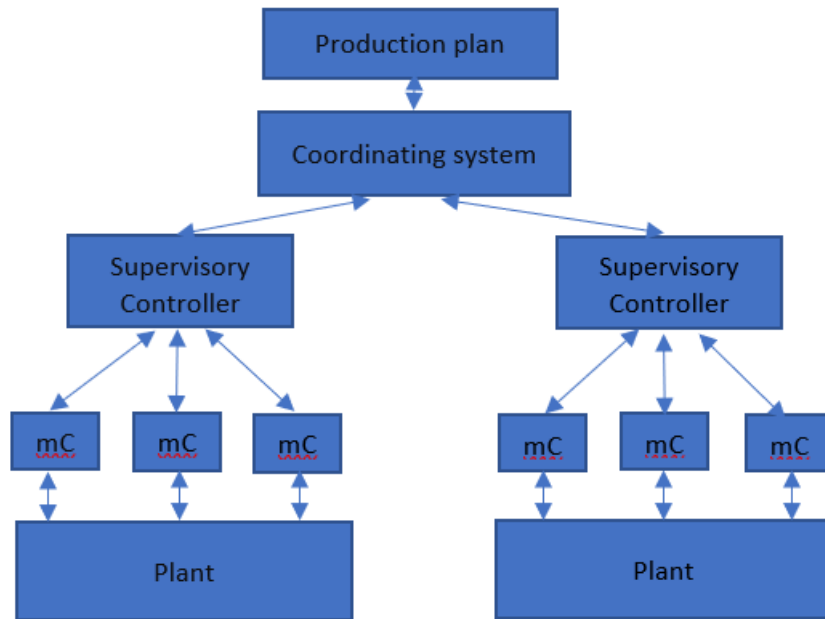


Figure 9.4: Supervisory control of distributed manufacturing plants.

Moreover, distributed systems may propose supervisory controllers naturally. Typical examples are production plants, such as assembly systems, where several subprocesses are carried out. In figure 9.4, the architecture of a distributed plan is designed as a supervisory control system. Each different station of the shop floor is controlled by several microprocessors or microcomputers (μC , in figure 8.4) that, at the same time, are supervised and coordinated accordingly to the production requirements or scheduling of the whole plant. The architecture also presents a hierarchical control structure.

9.4 Hierarchical control

A **hierarchical control system (HCS)** is a form of control system in which a set of devices and governing software is arranged in a hierarchical tree. When the links in the tree are implemented by a computer network, then that hierarchical control system is also a form of networked control system.

Figure 9.5 represents a network of interrelated subsystems that belong to the considered system under study. Each subsystem holds local control and adaptive supervision, which means they can control individual processes with the ability to adapt the parameters of the controller accordingly to a certain supervision requirement, i.e., possess an interchange of information according to a dynamic system.

Hierarchical control can be interpreted as an attempt to handle complex problems by decomposing them into smaller subproblems and reassembling their solutions into a "functioning" hierarchical structure. So far, heuristic approaches have been prevalent. However, they cannot guarantee that the overall solution does indeed meet the specifications. In contrast, our project aims at a formal synthesis method that can provide such a guarantee. A **hierarchical control system (HCS)** is a form of control system in which a set of devices and governing software is arranged in a hierarchical tree. When

the links in the tree are implemented by a computer network, then that hierarchical control system is also a form of networked control system [6]. Figure 9.6 shows a standard hierarchical control system.

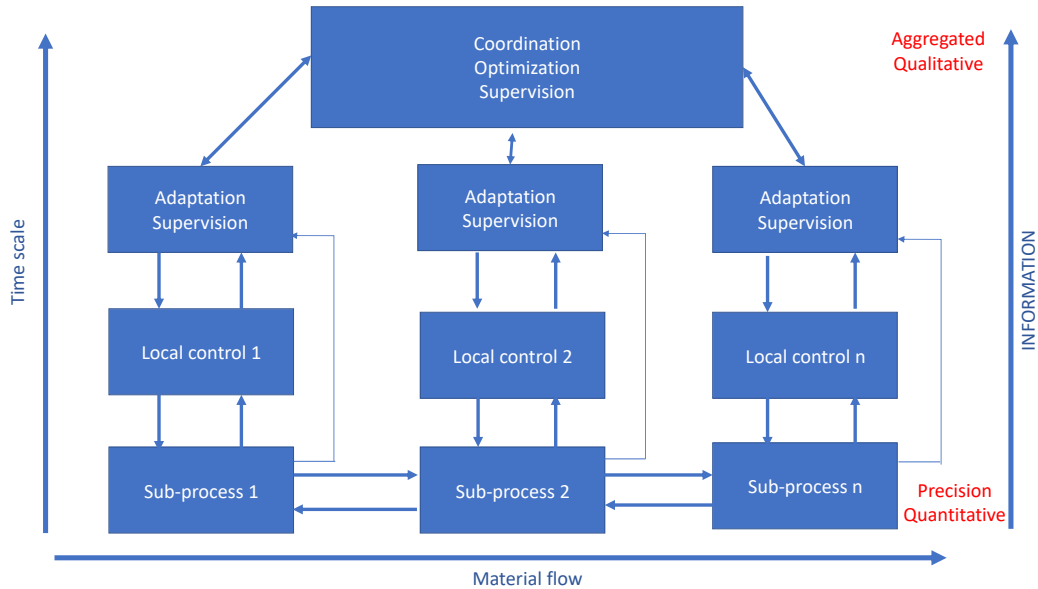


Figure 9.5: Hierarchical control architecture.

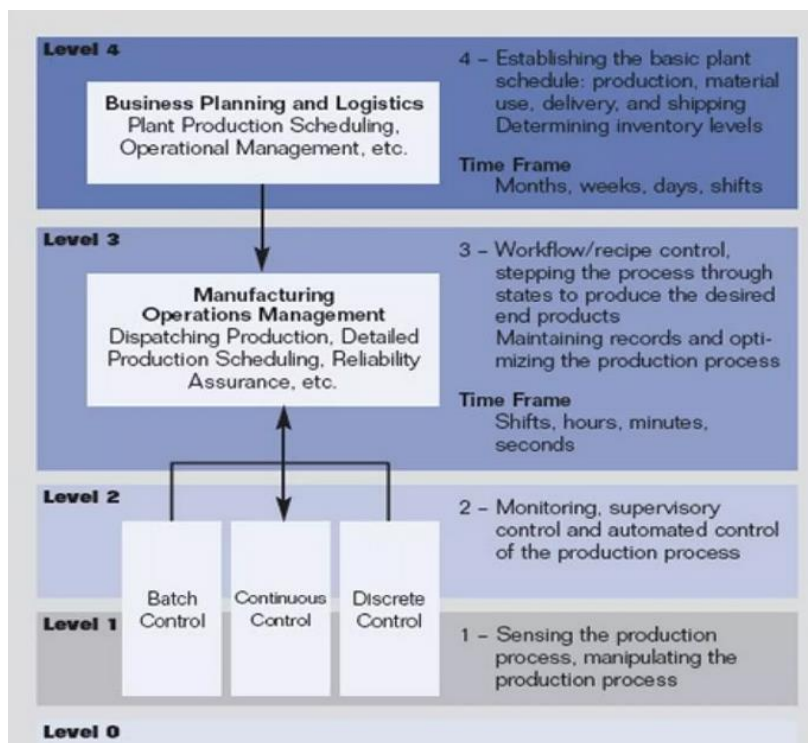


Figure 9.6: Standard ANSI/ISA-95 hierarchical control architecture.

9.5 Virtual sensors, estimators, observers, or filters. Soft sensors.

Virtual sensing techniques, also called **soft sensing**, are used to provide feasible and economical alternatives to costly or impractical physical measuring instruments. A **virtual sensing** system uses information available from other measurements and process parameters to calculate an estimate of the quantity of interest.

In the field of **gas sensors**, an **array of virtual sensors** can substitute electronic noise. Virtual gas sensors can be obtained by using a single sensor working in dynamic mode, i.e., working in repeated cycles that include a customized range of temperature, voltage, or both, which is equivalent to an array of real sensors. The choice of the temperature or voltage range depends on the gas type and its concentration.

In statistics, an **estimator** is a rule for calculating an estimate of a given quantity based on observed data; thus, the rule (the estimator), the quantity of interest (the “estimand”), and its result (the estimate) are distinguished. For example, the sample mean is a commonly used estimator of the population mean [7].

There are point and interval estimators. The point estimators yield single-valued results. This contrasts with an interval estimator, where the result would be a range of plausible values. "Single value" does not necessarily mean "single number," but includes vector-valued or function-valued estimators.

Estimation theory is concerned with the properties of estimators, that is, with defining properties that can be used to compare different estimators (different rules for creating estimates) for the same quantity based on the same data. Such properties can be used to determine the best rules to use under given circumstances. However, in robust statistics, statistical theory goes on to consider the balance between having good properties, if tightly defined assumptions hold, and having fewer good properties that hold under wider conditions [7].

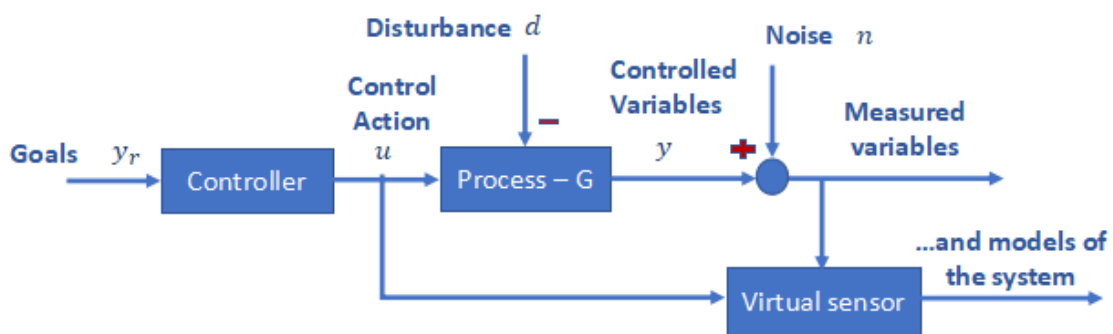


Figure 9.7: Closed-loop scheme with virtual sensor.

Figure 9.7 shows a closed-loop scheme incorporating a virtual sensor. In this case, the virtual sensor is taking signals from the process control and output of the physical system. They may be used to filter outputs from noise, estimate disturbances, and consider other internal variables. Within this scenario, a virtual sensor may be used to reduce the effect of measurement noise, estimate disturbances, and get richer information from the plant

process. As a result, virtual sensors include filters (noise), estimators (disturbances), and observers (internal variables).

Optimal estimators (or observers or filters) for nonlinear systems are, in general, difficult to derive or implement. The common approach is to use approximate solutions such as extended Kalman filters, ensemble filters, or particle filters. However, no optimality properties can be guaranteed by these approximations, and even the stability of the estimation error cannot often be ensured. Another relevant issue is that, in most practical situations, the system whose variables must be estimated is not known, and a two-step procedure is adopted based on model identification from data and filter design from the identified model.

9.6 Cascade structure/master-slave control.

Master/slave is a model of asymmetric communication or control where one device or process (the "master") controls one or more other devices or processes (the "slaves") and serves as their communication hub. In some systems, a master is selected from a group of eligible devices, with the other devices acting in the role of slaves.

The master/slave terminology was coined in 1904. Since the early 21st century, it has become a subject of controversy for its association with slavery, and organizations and products have begun replacing them with alternative terms, such as **cascade control**.

In automotive engineering, the master cylinder is a control device that converts force into hydraulic pressure in the brake system. This device controls slave cylinders located at the other end of the hydraulic system.

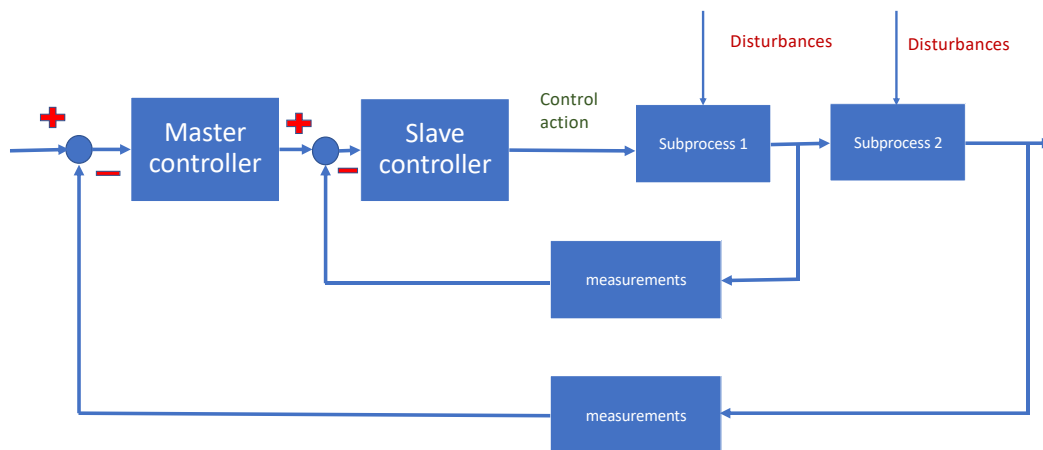


Figure 9.8: Cascade control loop.

The control structure, which represents the master/slave controller, is shown in figure 9.8. Following the structure of cascade control, a secondary loop (slave loop) usually responds fast and can overcome the effect of secondary disturbances on the primary loop efficiently. The secondary loop may reduce the nonlinearity of control valves and secondary processes. The control scheme or structure may be as sophisticated as subprocesses of the system.

Cascade control setpoints are sent to a secondary controller from a primary control system, each monitoring a different variable, with the secondary controller monitoring the data with the most time-sensitive response.

Information from the measuring device (sensor or transmitter) goes to the controller, then to the final control device (solenoid coil, motor drive, or control valve), influencing the process, which is sensed again by the measuring device. The controller's task is to inject the proper amount of negative feedback so that the process variable stabilizes over time. This flow of information is collectively referred to as a feedback control loop.

To cascade controllers means to connect the output signal of one controller to the setpoint of another controller, with each controller sensing a different aspect of the same process. The first controller (called the *primary*, or archaically, *master*) essentially “gives orders” to the second controller (called the *secondary*, or archaically, *slave*) via a *remote setpoint* signal. Thus, a cascade control system consists of two feedback control loops, one nested inside the other:

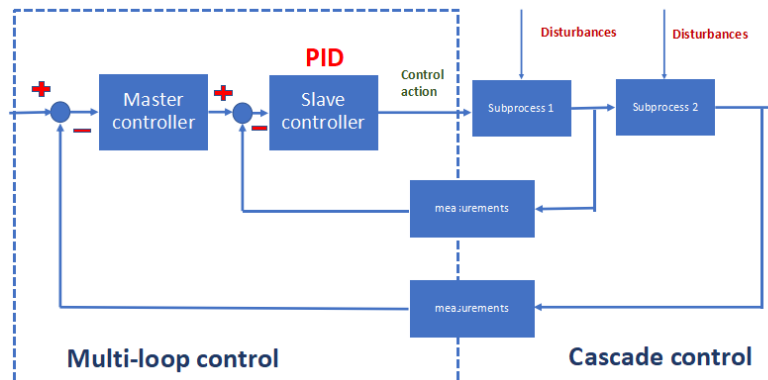


Figure 9.9: Cascade control loop.

In general terms, a **multi-loop scheme** (Figure 9.9) uses an electronic device as the controller of several subprocesses. It can read multiple input variables from several measuring devices, providing the outputs to reach and maintain separate desired setpoints, reference signals, and commands values for each one. The controlled variables are measured by suitable sensors and converted to various signals acceptable to the multiloop controller. The controller compares those measured values to the command values or set points and actuates the control devices as needed. The control devices alter the controlled variable (control signal) by changing the appropriate properties to reach and maintain accurate and stable levels of the signal related to the control behavior at or near the setpoint.

The purpose of cascade control is to achieve greater stability of the primary process variable by regulating a secondary process variable in accordance with the needs of the first. An essential requirement of cascaded control is that the secondary process variable be faster-responding (shorter lag and dead times) than the primary process variable. This allows the secondary controller to be more immediately responsible for fine-tuning setpoint adjustments because of its proximity to the final control element. The primary controller may then be left to handle numerous other variables and remote setpoints for other secondary controllers. The cascade controller makes the internal subprocess faster, reducing the effects of the inner disturbance. Furthermore, the desired closed-loop behavior is achieved, reducing the effect of disturbances in both loops. It provides a suitable solution for tracking.

A very common example of cascade control is a *valve positioner*, which sends a command signal from a regular process controller and, in turn, works to ensure the valve stem position precisely matches that command signal. The control valve's stem position is the process variable (PV) for the positioner, just as the command signal is the positioner's setpoint (SP). Valve positioners therefore act as “slave” controllers to “master” process controllers controlling pressure, temperature, flow, or some other process variable.

Typical schemes are the P-PI controllers for electrical drives [8, 9] shown in figure 9.10.

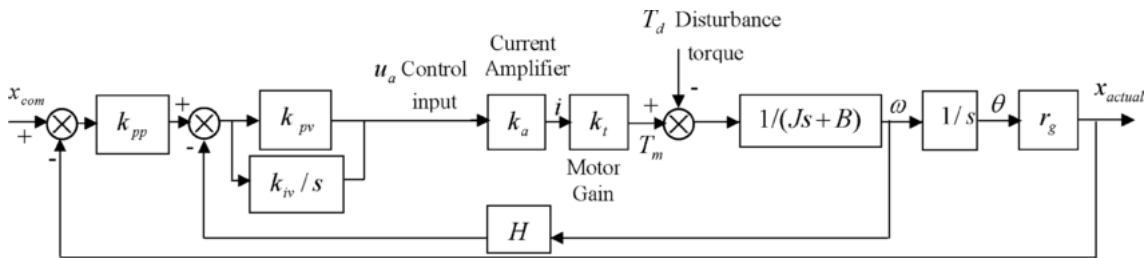


Figure 9.10: P-PI Control for electrical drives [9].

References:

- [1] Taguchi, H. and Araki, M., (2000), Two-Degree-of-Freedom PID Controllers — Their Functions and Optimal Tuning IFAC Proceedings Volumes, Vol. 33 (4), pp. 91-96, IFAC Workshop on Digital Control: Past, Present and Future of PID Control, Terrassa, Spain, 5-7 April 2000, [https://doi.org/10.1016/S1474-6670\(17\)38226-5](https://doi.org/10.1016/S1474-6670(17)38226-5).
- [2] Landers, R.G., (1997). Supervisory Machining Control: A Design Approach Plus Chatter Analysis and Force Control Components, 9780591307450, University of Michigan.
- [3] Landers, Robert & Ulsoy, A. (1998). Supervisory Machining Control: Design Approach and Experiments. CIRP Annals - Manufacturing Technology. 47. 301-306. 10.1016/S0007-8506(07)62838-8.
- [4] Sharma, K.L.S. (2011). Overview of Industrial Process Automation. 10.1016/C2011-0-04273-4, 2nd ed.
- [5] G. Pritschow *et al.* “Open Systems Controllers - A Challenge for the Future of the Machine Tool Industry,” CIRP Annals (1993).
- [6] Senehi, M.K., Kramer, T.R., Ray, S.R., Quintero, R., Albus, J.S. (1994). Hierarchical control architectures from shop level to end effectors. In: Joshi, S.B., Smith, J.S. (eds) Computer control of flexible manufacturing systems. Springer, Dordrecht. https://doi.org/10.1007/978-94-011-1230-7_2
- [7] Fortuna, Luigi & Graziani, Salvatore & Rizzo, Alessandro & Xibilia, M.G. (2007). Soft Sensors for Monitoring and Control of Industrial Processes. 10.1007/978-1-84628-480-9.
- [8] L. Rubio, A. Ibeas and X. Luo, "P-PI and super twisting sliding mode control schemes comparison for high-precision CNC machining," *2016 24th Iranian Conference on Electrical Engineering (ICEE)*, Shiraz, Iran, 2016, pp. 1825-1830, doi:10.1109/IranianCEE.2016.7585818.
- [9] Hecker, Rogelio & Flores, G. & Xie, Q. & Haran, R. (2008). Servocontrol of machine-tools: A review. Latin American applied research. 38. 85-94.

Chapter 10: Advanced Control Methods

This chapter covers some advanced control methods that provide more elegant and sophisticated solutions to certain system configurations.

- 1. Model predictive control.**
- 2. Adaptive control.**
- 3. Neural network control.**
- 4. Sliding mode control.**
- 5. Optimal control.**
- 6. Intelligent control.**
 - a. Learning control.**
 - b. Expert control.**
 - c. Fuzzy control.**

The previous control structures or architectures may be implemented with advanced control methods. Advanced control methods have been proven to be more beneficial and profitable than elementary control methods. Some claim that applying advanced control has resulted in cost savings or product quality improvements from 2% to tens of percent. Advanced control methods involve more complex calculations than the conventional PID controller algorithm.

Advanced control requires modeling the process and identifying the parameter, either offline or online. Moreover, the process behavior can be predicted using the model of the process. The system is subjected to an evaluation performance criterion subject to process constraints, and its optimization may be required. In multivariable control, state variables appear, and matrix calculations are needed.

Often, advanced control is a high-level control procedure that takes care of subprocesses controlling low-level unit control loops such as, PID controllers. In this case, the advanced control strategy aims to fulfill economic objectives by providing appropriate set points for the lower level.

As a class of control methods, advanced control is rather vague— not because of the large number of methods that can be included but the indistinct classification criteria that control loops use to minimize a given performance criterion.

Fundamentally, advanced control does not differ from any other control strategy in the sense that it is also based on feedback control. Yet, it is the intelligence behind advanced control that makes the difference when compared to conventional controllers.

The use of process models is a common characteristic of advanced control methods. Advanced control relies strongly on process models that describe the process behavior. The models try to capture the essence of the process information. The most important use of the model is to enable prediction of the process output behavior when facing changes in either set point or load disturbance variables. For control design purposes, the more accurate the process model is, the better the control performance that can be achieved by a successful control design. Small process model uncertainties can be allowed because of the feedback control that corrects deviations due to, not only process disturbances but also modeling errors.

Process models can be divided into qualitative, mathematical, and statistical models.

Qualitative models are basically rule-based tables of process behavior laws. Fuzzy-based models can be regarded as qualitative.

10.1 Model predictive control

Model Predictive Control (MPC) is referred to as a family of controllers in which there is a direct use of an explicit model of the system. Control design methods based on the MPC concept have found wide acceptance in industrial applications and have been studied by academia. The reason for such popularity is the ability of MPC designs to yield high-performance control systems capable of operating without expert intervention for long periods of time.

Model predictive control (MPC) is an optimal control-based method to select control inputs by minimizing an objective function. The objective function is defined in terms of both present and predicted system variables and is evaluated using an explicit model to predict future process outputs. Model predictive control incorporates ideas from systems theory, system identification, and optimization. MPC is implemented with digital computers; it carries discrete-time formulations [1].

10.2 Adaptive control

Adaptive control is the methodology used by a controller that must adapt to a system with parameters that vary or are initially uncertain. An adaptive control system utilizes on-line identification of either a system parameter or a controller parameter, which does not require a priori information about the bounds on these uncertain or time-varying parameters. The adaptive control approaches consider their control design in the sense of Lyapunov. An adaptive control system can also be defined as a feedback control system intelligent enough to adjust its characteristics in a changing environment to operate in an optimal manner according to some specified criteria.

Adaptive control systems have achieved great success in aircraft, missile, and spacecraft control applications. Traditional adaptive control methods are mainly suitable for (1) mechanical systems that do not have significant time delays and (2) systems that have been designed so that their dynamics are well understood. In industrial process control applications [2], however, traditional adaptive control has not been very successful. Traditional adaptive control methods, either **model reference** or **self-tuning**, usually require some kind of identification of the process dynamics. This contributes to a few fundamental problems, such as (1) the amount of offline training required, (2) the tradeoff between the persistent excitation of signals for correct identification and the steady system response for control performance, (3) the assumption of the process structure, and (4) the model convergence and system stability issues in real applications. Traditional adaptive control methods assume knowledge of the process structure. They have major difficulties in dealing with nonlinear, structure-variant, or large-time-delayed processes. The field of electrical drives and some manufacturing processes seem to be the most promising areas for the application of adaptive control. The dynamics of such systems are well understood, and the limitations of theory are less restrictive.

In general, it is recommended that more experimental work be done in connection with proven theoretical methods. These methods, however, must provide a systematic design procedure that can be understood and also implemented by industrial engineers.

The classification of adaptive control schemes can be found in [3], while the theory behind them is well addressed, for instance, in the series of books and articles by Åström and Wittenmark [4] and further research by the authors in the field.

Example 10.1: As a practical example, model reference adaptive control schemes are shown and implemented in [5, 6] where the cutting forces in milling processes are kept constant despite sudden variations in the cutting conditions, in that case, coming from the axial depth of cut changes in the workpiece, as figure 10.1 shows.

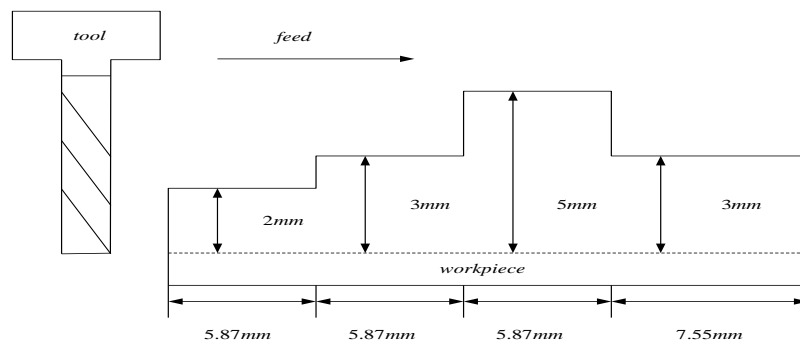


Figure 10.1: Workpiece profile when adaptive control is tested in milling processes for aeronautic applications.

The model reference adaptive control scheme in Figure 10.2 estimates the parameters of the transfer function in real-time, and the controller is adapted to keep the forces under a prescribed upper limit.

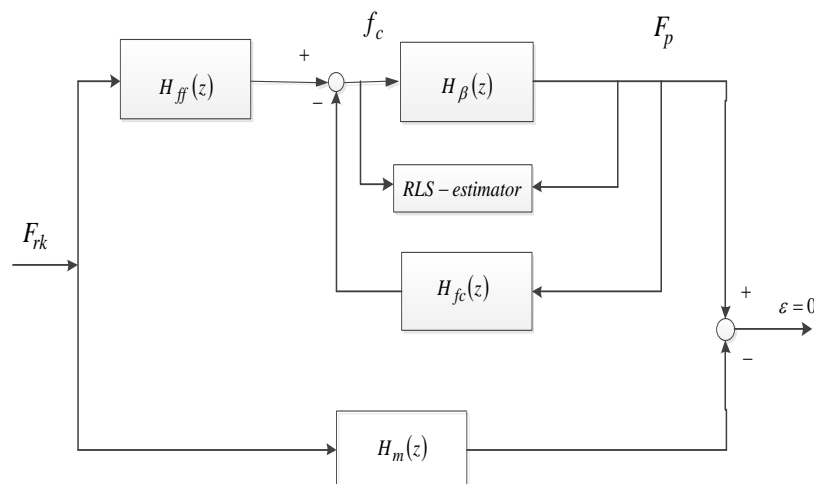


Figure 10.2: Model reference adaptive control applied in the milling cutting force control.

Figure 10.3 shows the results of the system, with discrete and continuous force responses keeping constant except at the point where the axial depth of cut is changed, the error signal, and the control law that adapts the system.

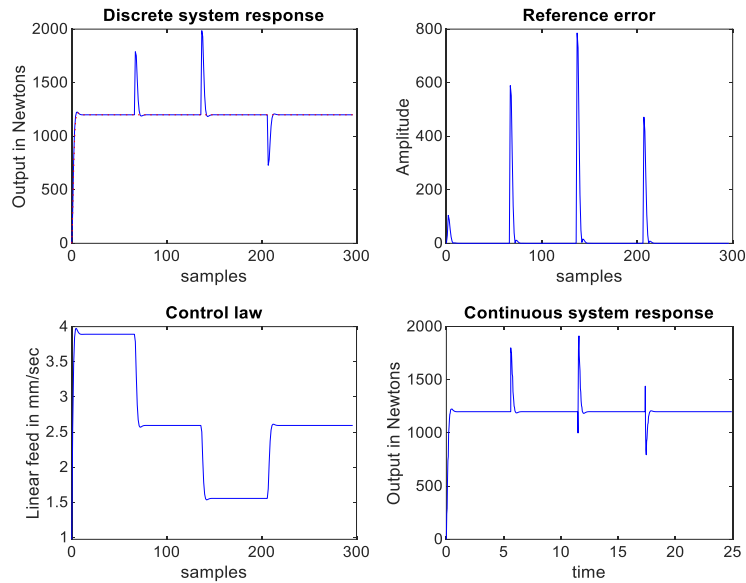


Figure 10.3: Responses of the system, discrete and continuous time, error, and control signal.

10.3 Neural Network Control

Neural network (NN) control involves using neural networks to design controllers for dynamic systems. These controllers power the learning capabilities of neural networks to manage complex, nonlinear, or time-varying systems that are difficult to model using traditional methods. Neural networks are computational models inspired by the human brain, composed of interconnected layers of neurons. Neural networks can approximate complex functions and learn from data. They can be used to identify the model of the system or to adapt the control law. NN can optimize the control law and effectively learn the dynamics of the system. As a result, NN controllers can identify the system's dynamics, design the controller, and be trained using data from previous experiments. On the other hand, NN requires large amounts of data to train effectively, the generalization of unconsidered situations is not straightforward, and the stability of the system is not ensured.

One class of NN is deep learning. Deep learning is a subset of machine learning that involves neural networks with many layers (hence "deep") that can learn and model complex patterns in large amounts of data. They include some NN architectures such as feedforward neural networks, convolutional neural networks, or generative adversarial networks. Deep learning models typically require large amounts of labeled data. Training deep learning models is computationally intensive and often requires specialized hardware. Deep learning models are often seen as "black boxes," making it difficult to understand how they make decisions. Choosing the right model architecture and training parameters can be complex and time-consuming.

10.4 Sliding mode control

To overcome the influences of disturbances and model parameter uncertainties, a natural control approach is adopted by the sliding mode controller (SMC), which was first proposed by Utkin [5]. These controllers have some desirable closed-loop properties,

including invariance, dynamic order reduction, and robustness against parameter variations and disturbances. The design of SMC consists of two stages, i.e., an equivalent control and a switching control. The equivalent control is derived from the definition of sliding surface to which the controlled system trajectories must belong. The switching control forces the system to slide along the sliding surface despite being influenced by parameters or external disturbances. As switching control usually takes a discontinuous form, it often results in undesirable *chattering* in the control signals of the close-loop system [5].

Sliding mode control is widely used in the speed control of electric drive systems. It provides attractive features such as fast dynamic response, insensitivity to variations in plant parameters, and external disturbance.

Example 10.2: One example of this approach is given to show how the linear motor feed drive (movement of the axes of the system in milling processes) is controlled against disturbances inherently acquired [6, 7]. Figure 10.4 shows the presented algorithm.

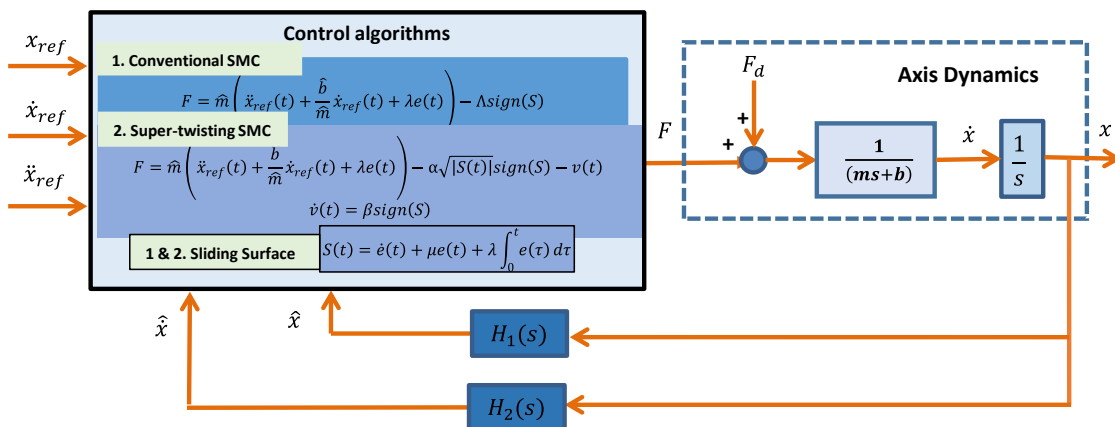


Figure 10.4: Closed-loop description, including feed drive machine tools and SMC algorithms.

The results show four figures. The upper left figure plots the reference position in dashed red and the actual position in blue; the second one represents the position error; the third one outputs velocity error; and the fourth one is the control signal.

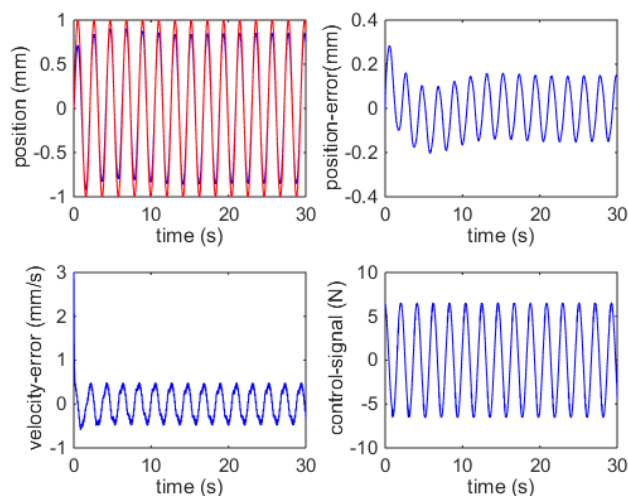


Figure 10.5: Outputs of the SMC with $\Lambda = 40$.

Figure 10.5 represents the SMC with the control parameter $\Lambda = 40$ under the influence of the cutting force acting as an external disturbance. It can be shown that the control signal is smooth, but the tracking error is significant, leading to the system not following pre-established tool path references.

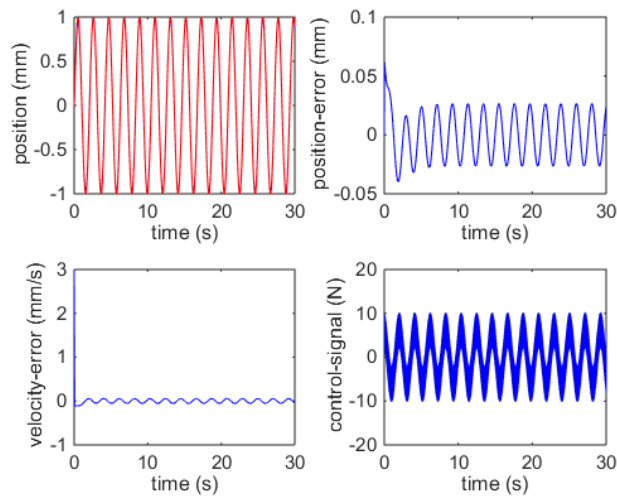


Figure 10.6: Outputs of the SMC with $\Lambda = 400$.

To deal with error, the control parameter Λ is increased to 400, as can be seen in figure 10.6. In this case, adequate tracking performance is obtained; however, chattering due to the high frequencies introduced by the sign function in the control signal appears, as can be seen in the control signal plot in Figure 10.6. The trade-off between precision in tracking the reference signal and chattering limits the control algorithm.

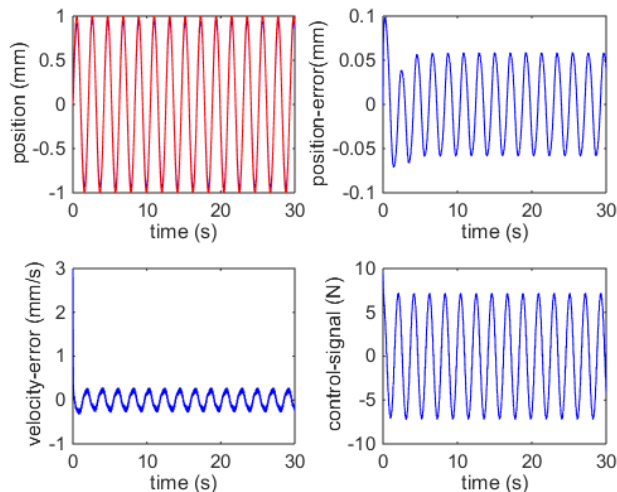


Figure 10.7: Outputs of Super-twisting SMC with $(\alpha, \beta) = (200, 200)$.

To deal with the chattering problem, super-twisting SMC has been applied. Figure 10.7 shows that better performance in tracking the reference signal can be achieved without introducing chattering in the control signal. A balance between tuning control parameters to reach adequate tracking and chattering is required to achieve satisfactory performance indexes.

High values of control parameters make the algorithms more demanding in terms of control power, i.e., amplifiers, and more sensitive to introducing noise to the system.

10.5 Robust Control

Robust control is a controller design methodology that focuses on the reliability (robustness) of the control algorithm. Robustness is usually defined as the minimum requirement a control system must satisfy to be useful in a practical environment. Once the controller is designed, its parameters do not change, and control performance is guaranteed.

Robust control methods, either in the time domain or the frequency domain, usually assume knowledge of process dynamics and their variation ranges. Some algorithms may not need a precise process model but then require some kind of off-line identification. The design of a robust control system is typically based on the worst-case scenario, so the system usually does not work at optimal status in terms of control performance under normal circumstances.

Some approaches to robust control are H_2 and H_∞ . H_2 optimized control aims to minimize the energy of the system's response to disturbances, quantified by the H_2 norm of the transfer function from the disturbance to the controlled output. This approach is particularly useful when the system is subject to stochastic disturbances or when minimizing the average performance over all frequencies is desired. H_∞ optimized control aims to minimize the worst-case gain (the H_∞ norm) from disturbances to the controlled output, providing robustness against model uncertainties and disturbances. It is particularly useful in scenarios where the system may face unexpected or unmodeled disturbances.

Robust control methods are well suited to applications where control system stability and reliability are the top priorities, process dynamics are known, and variation ranges for uncertainties can be estimated. Aircraft and spacecraft controls are some examples of these systems.

In process control applications, some control systems can also be designed with robust control methods, especially for those processes that are mission-critical and naturally have large uncertainty ranges and small stability margins. However, the design of a robust control system requires high-level expertise. Once the design is properly accomplished, the system should work well without the need for much operator attention. But on the other hand, if upgrades or major modifications are required, the system must be redesigned [9].

10.6 Optimal Control

The statement of a typical optimal control problem can be expressed in the following paragraph: The state equation and its initial condition of a system to be controlled are given. The defined objectives or requirements are also provided. Find a feasible control such that the system, starting from the given initial condition, transfers its state to the objective set and minimizes a performance index.

In principle, optimal control problems belong to the calculus of variations. Pontryagin's Maximum Principle and Bellman's Dynamic Programming are two powerful tools to

solve closed-set constrained variation problems, which are related to the most optimal control problems [10].

Optimal control is an important component in modern control theory. It has great success in space, aerospace, and military applications such as the moon landing of a spacecraft, the flight control of a rocket, and the missile blocking of a defense missile. In industrial systems, most process control problems are related to the control of flow, pressure, temperature, and level.

10.7 Intelligent Control

Intelligent control is another major field in modern control technology. There are different definitions regarding intelligent control, but it is referred to as a control paradigm that uses various artificial intelligence techniques, which may include the following methods: (1) learning control, (2) expert control, and (3) fuzzy control.

10.7.1 Learning Control

Learning control uses pattern recognition techniques to obtain the status of the control loop; and then makes control decisions based on the loop status as well as the knowledge or experience stored previously. Since learning control is limited by its stored knowledge, its application has never been popular.

10.7.2 Expert Control

Expert control, based on expert system technology, uses a knowledge base to make control decisions. The knowledge base is built by human expertise, system data acquired online, and inference machines designed. Mathematical models of the system are also a widely accepted approach to tackling the knowledge base of expert systems. Since the knowledge in expert control is represented symbolically and is always in discrete format, it is suitable for solving decision making problems such as production planning, scheduling, and fault diagnosis.

Example 10.3: The expert system developed in [11] to choose cutting parameters in milling processes according to some specifications of the process to carry out is briefly introduced in this section.

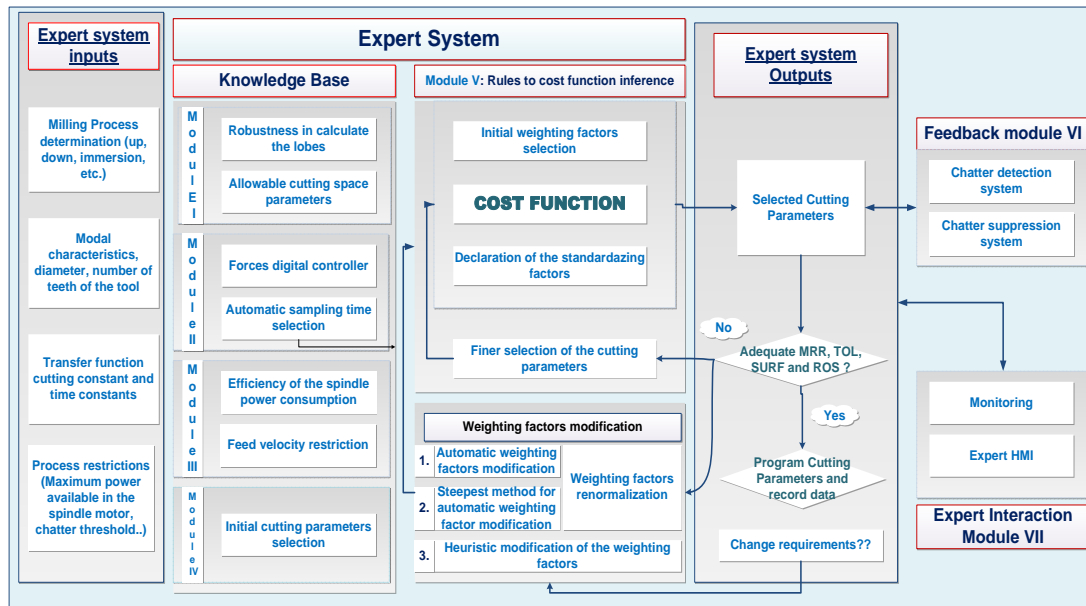


Figure 10.8: Schematic representation of the expert system.

The approach consists of a series of rules split into seven modules (Figure 10.8). Each module can interact independently, leading to a universal system in the sense that it can be applied to every machine. The first module gives robustness to the system and presents the rough allowable input cutting parameters. The second module includes the model reference adaptive control of milling forces functionality, keeping the forces of the system under the prescribed upper limit despite variations in system parameters. Moreover, the possibility of manipulating the sampling time of the system is added to preserve computer resources if necessary. The third module covers the spindle and feed-drive motor constraints. The fourth module gives initial computational input parameters subjected to constraints on the motors and suggests potential initial spindle speed candidates. The fifth module presents a novel multi-objective cost function to evaluate the performance of the system. It has been devised from first principles and depends on the material to be removed, tool life, surface roughness, and a stability margin. Weighting factors indicate the importance of each term in the cost function. Each term is then modulated by a weighting factor, where the most important term is associated with the largest weighting factor to obtain Pareto optimal cutting parameters. Pareto optimal fronts can be obtained by two methods. First, by refining the searching cutting parameters around the selected one, and secondly, by modulating the weighting factors automatically if new production requirements are required or by interacting with system engineers or operators to take advantage of their experience. This information is stored in a database to register and modify previous data. Module 6 gives automatic feedback to the system if chatter vibrations are experienced due to non-modelled nonlinear effects such as wear or run out of the tool. Finally, module 7 proposes to monitor key parameters for better interaction with expert engineers and operators and to infer with them, providing in this way the best qualities of traditional expert systems and model based expert systems. Nevertheless, since the presented expert system is based on mathematical models of the system, its output selection is dominated by the modal parameters of the tool, tool and workpiece material properties, the linear transfer function, which gives the relationship between the resultant force and the feed velocity, stability and robustness constants, the determination of the process, and the process restrictions.

The inference rules are presented in Table 10.1.

Table 10.1: Application of the expert rules

<i>Rule</i>	Knowledge Base
1.	Since stability lobes are calculated from a linear approximation, stability region inexactitude is added. Also, a robustness factor that influences the axial depth of the cut is considered.
2.	Cutting space parameters extracted from lobes, restrictions in spindle power availability, and cutting force controllers.
3.	Considerations about controlling the forces; using model reference adaptive control for keeping forces under the prescribed upper limit.
4.	Sampling period considerations to maximize computer resources.
5.	Spindle power consumption restrictions.
6.	Feed drive limitations.
7.	Giving the appropriate initial cutting space parameter to decrease searching time.
Rules for inference with the cost function	
8.	Initial weighting factor selection: the selection of the right initial weighting factors plays an important role in achieving good solutions in a short time.
9.	Declaration of standardizing factors.
10.	Selection of cutting parameters criteria; coarse and fine criteria.
11.	Automatic rule of weighting factors modification.
12.	Automatic weighting factor modification through the gradient descent method. An alternative way to automatically modify the weighting factors is proposed.
13.	Renormalization of the weighting factors if they are re-programmed automatically through rules 11 and 12.
14.	Re-parameterization of weighting factors.
Feedback and expert HMI rules	
15.	A chatter vibration detection algorithm is added to be more reliable.
16.	Chatter suppression algorithm to lead the system to stable and reliable cutting conditions.
17.	Monitoring signals
18.	Interaction with expert engineers and operators

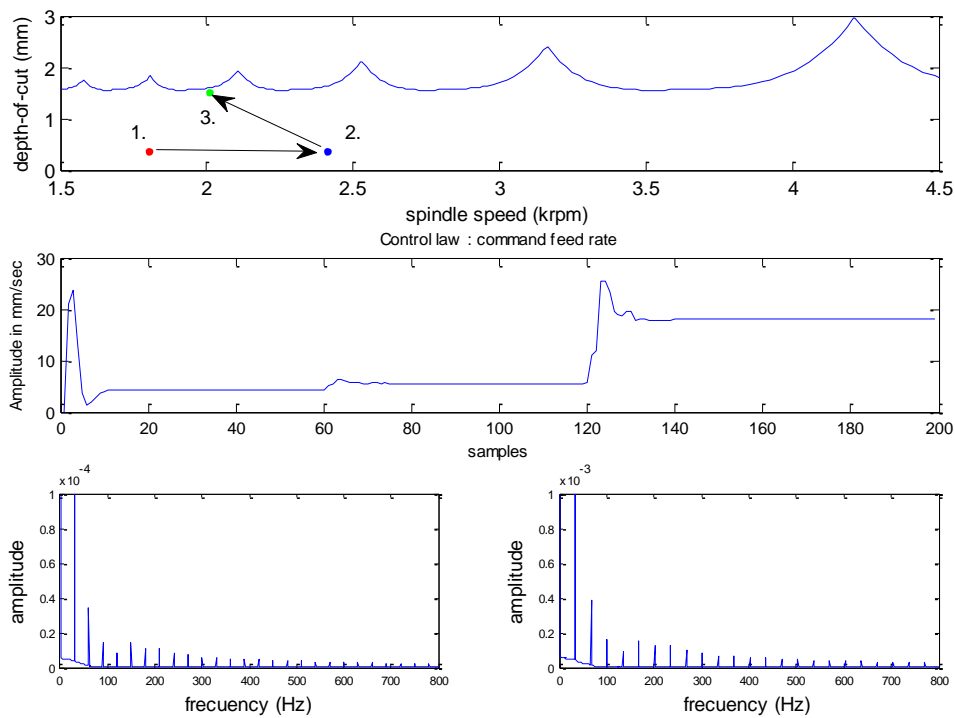


Figure 10.9: Programmed cutting parameters in the lobes chart, programmed feed rates, and frequencies.

Figure 10.9 shows the represented cutting parameters on the stability charts: spindle speed and axial depth of cut for the three working points, namely 1, 2, and 3, and the programmed feed velocity for the three cases, represented by changes in the feed velocity. It can be observed that the control signal (feed velocity) is smooth and feasible except in the transient response, when cutting parameters change and the pairs of spindle speed and axial depth of cuts are both under the border line in the stable zone. The control signal has a peak at the transitory due to the selection of initial conditions that are not close to the real values. It also experiments with a not smooth transition when changing from working point 2 to 3. The frequency response of points 1 and 3 gives the tooth pass frequency and its harmonics in each case, as with case 2, which has not been considered in the graphs.

The expert system can move around the cutting parameter space subject to different production states or requirements. This fact can be achieved by programming easy and intuitive c_i –parameters, which leads to giving adequate cutting parameters to the milling system according to production requirements. In this way, the expert system provides an intuitive but intelligent plan for cutting parameters, giving a fast solution if changeable situations happen. Finally, through rule 14, the obtained c_i parameters are stored and subjected to learning and adaptive skills to help improve proven solutions.

10.7.3 Fuzzy Control

Fuzzy control, unlike learning control and expert control, is built on mathematical foundations with fuzzy set theory. It represents knowledge or experience in a mathematical format that processes, and system dynamic characteristics can be described by fuzzy sets and fuzzy relational functions. Control decisions can be generated based on fuzzy sets and functions with rules.

Although fuzzy control has great potential for solving complex control problems, its design procedure is complicated and requires a great deal of specialty. In addition, fuzzy math does not belong to the field of mathematics since many basic mathematical operations do not exist. For instance, the inverse addition is not available in fuzzy math. Then, it is very difficult to solve a fuzzy equation, yet solving a differential equation is one of the basic practices in traditional control theory and applications. Therefore, a lack of good mathematical tools is a fundamental problem for fuzzy control to overcome.

References:

- [1] Model predictive control. Edited by Eduardo F. Camacho and Carlos Bordons. Berlin Heidelberg: Springer-Verlag, 2007.
- [2] Parks, P. & Schaufelberger, W. & Schmid, Chr & Unbehauen, H. (2006). Applications of adaptive control systems. 10.1007/BFb0003264.
- [3] Åström, K. J., & Wittenmark, B. (1995). Adaptive Control (2 ed.). Addison-Wesley. <https://doi.org/10.2307/1269433>.
- [4] Altintas, Y. (2012). Manufacturing Automation: Metal Cutting Mechanics, Machine Tool Vibrations, and CNC Design (2nd ed.). Cambridge: Cambridge University Press.
- [5] Rubio, L. & De la Sen, M. & Bilbao-Guillerna, A. (2007). Intelligent adaptive control of forces in milling processes. 2007 Mediterranean Conference on Control and Automation, MED. 1 - 6. 10.1109/MED.2007.4433842.
- [6] Utkin, V.I., Sliding modes in control and optimization, NY, Berlin: Springer-Verlag, 1992.
- [7] Y. Altintas, K. Erkorkmaz, and W-H. Zhu, Sliding mode controller for high-speed feed drives, Annals of the CIRP 49, 2000, pp. 265-270, [https://doi.org/10.1016/S0007-8506\(07\)62943-6](https://doi.org/10.1016/S0007-8506(07)62943-6).
- [8] Rubio, L., Ibeas, A. and Luo, X., P-PI and super twisting sliding mode control schemes comparison for high-precision CNC machining, 2016 24th Iranian Conference on Electrical Engineering (ICEE), Shiraz, 2016, pp. 1825-1830, doi: 10.1109/IranianCEE.2016.7585818.
- [9] Mackenroth, U., Robust Control Systems: Theory and Case Studies, 2013, Springer Berlin Heidelberg.
- [10] Lewis, F.L. and Syrmos, V.L., Optimal Control, 2012 Wiley.
- [11] Rubio, L., De la Sen, M., Longstaff, A.P., Fletcher, S., Model-based expert system to automatically adapt milling forces in Pareto optimal multi-objective working points, Expert Systems with Applications, Volume 40, Issue 6, 2013, Pages 2312-2322, ISSN 0957-4174, <https://doi.org/10.1016/j.eswa.2012.10.034>.

Acknowledgments

The author is thankful to Professor Asier Ibeas for the feedback given to correct and clarify the structure and contents of the booklet.

The author wants to extend the acknowledgments to the Control Theory and Control Practice communities for the work they are carrying out in every part of the world which facilitates the learning and teaching of the subject to the students and professors and, as consequence, this work.

The author is also grateful for the opportunity given by the Savaria Institute of Technology and Eötvös Loránd University to make this manuscript true.

Note from the author: This booklet is the first version of a work in progress whose unique intention is to provide students with an easy-going reference when dealing with techniques as an introductory topic. It is intended to continue working on it, introducing examples and exercises with more pedagogical content, structuring the document, if possible, in a better structure way to facilitate the interiorization of the contents, improving the written in a way where the mathematics and the language merge, introducing more refined references, and including some which they deserve, but at this point, for any reason, I could not insert, decrease the typo errors, which will also depend on the feedback given by the readers, etc.