Lesson 1

Introduction

This lesson establishes some basic definitions and provides examples of signals and signal processing systems.

1.1 Learning Outcomes

By the end of this lesson, you should ...

1. **Understand** that there are different classes of signals.
2. **Understand** the meaning of signal processing and be aware of its applications.

1.2 Signals and Signal Classification

So what are signals? A **signal** is a quantity that can be varied in order to convey information. If a signal does not contain useful information (at least not in the current context), then the signal is regarded as **noise**. You may have a useful audio signal for your neighbour in a lecture, but this could be noise to anyone nearby that is trying to listen to the instructor!

Practically any physical phenomena can be understood as a signal (e.g., temperature, pressure, concentration, voltage, current, impedance, velocity, displacement, vibrations, colour). Immaterial quantities can also be signals (e.g., words, stock prices, module marks). Signals are usually described over time, frequency, and/or spatial domains. Time and frequency will be the most common in the context of this module, but our brief introduction to image processing will treat images as two-dimensional signals.

There are several ways of classifying signals. We will classify according to how they are defined over time and in amplitude. Over *time* we have:

1. **Continuous-time signals** – signals that are specified for every value of time t (e.g., sound level in a classroom).

2. **Discrete-time signals** – signals that are specified at discrete values of time (e.g., the average daily temperature). The times are usually denoted by the integer n.

In *amplitude* we have:

1. **Analogue signals** – signals can have any value over a continuous range (e.g., body temperature).

2. **Digital signals** – signals whose amplitude is restricted to a finite number of values (e.g., the result of rolling a die).

While we can mix and match these classes of signals, in practice we most often see **continuous-time analogue signals** (i.e., many physical phenomena) and **discrete-time digital signals** (i.e., how signals are most easily represented in a computer); see Fig. 1.1. However, digital representations of data are often difficult to analyse mathematically, so we will usually treat them as if they were analogue. Thus, the *key distinction is actually continuous-time versus discrete-time*, even though for convenience we will refer to these as analogue and digital. The corresponding mathematics for continuous-time and discrete-time signals are distinct, and so they also impose the structure of this module.

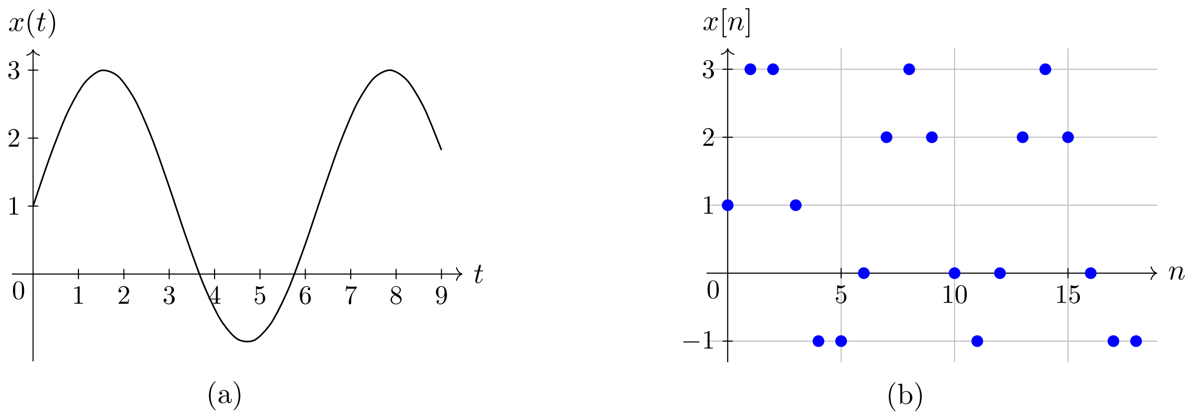


Figure 1.1: Comparison of classes of signals. (a) A continuous-time analogue signal $x(t)$. (b) A discrete-time digital signal $x[n]$ where amplitudes must be a whole integer.

Although many signals are not naturally in an electrical form, we can convert them to electrical form using **transducers**. We do this because most physical signal processing is done electrically (whether in analogue electronics or digital electronics).

1.3 What is Signal Processing?

In short, **signal processing** applies mathematical operations to a signal. Signal processing is applied in many disciplines in practice. Here are some top-level examples:

(a) **Image and video processing**. Used in industrial machine vision, target tracking, media compression, social media photo filters, etc.

(b) **Communication systems**. Used to package information for transmission over a noisy channel (wired or wireless) and recover at a destination.

(c) **Audio mixing**. Used to amplify sounds at different frequencies, noise cancellation, karaoke, introduce effects such as reverb, distortion, and delay, etc.

(d) **Biomedical systems**. Used to monitor vital signs, diagnose diseases, guide surgical procedures, etc.

(e) **Artificial intelligence**. Self-driving cars, speech/pattern recognition, smart homes (heating, appliances), video games, etc.

(f) **Financial markets**. Predict future prices (of currencies, stocks, options, houses, etc.) and optimise portfolio asset allocations.

We will not be covering all of these applications in this module, particularly as some of them rely on more advanced methods than what we will learn about. But we will use a diverse range of applications for our examples.

1.4 Summary

- **Signals** are quantities that vary to convey information.
- This module will focus on **continuous-time analogue signals**, which are continuous in both time and amplitude, and **discrete-time analogue signals**, which are discrete in both time. For convenience we will usually refer to these as just **analogue signals** and **digital signals**.
- **Signal processing** applies mathematical operations to a signal. There are many engineering fields that make use of it.

1.5 Further Reading

- Section 1.1 of "Essentials of Digital Signal Processing," B.P. Lathi and R.A. Green, Cambridge University Press, 2014.

1.6 End-of-Lesson Problems

We haven't covered enough material yet to provide meaningful problems, but consider the following reflective questions to review and prepare for the module:

1. Refer to the top-level examples of signal processing in Section 1.3. Can you infer what the signals might be and how they would be classified?

2. Refer to the background mathematical material in Appendix A. Are you comfortable with the skills needed for this module?

Part I

Analogue Signals and Systems

Lesson 2

Laplace Transforms and LTI Systems

The first part of this module is the study of **Analogue systems and signals**. As discussed in Lesson 1, most practical physical phenomena can be represented as continuous-time analogue signals. In this lesson, we present the notion of Laplace transforms (which should be a review from previous modules) and how they can be applied to linear time invariant (LTI) systems. This forms the basis for all of the analysis of analogue signal processing that we will perform in this part of the module.

2.1 Learning Outcomes

By the end of this lesson, you should be able to ...

1. **Apply** the Laplace transform and the inverse Laplace transform.
2. **Apply** properties of the Laplace transform and the inverse Laplace transform.
3. **Understand** what a Linear Time-Invariant (LTI) system is.
4. **Understand** the definitions of a transfer function and convolution.
5. **Understand** the common simple input signals.
6. **Apply** the Laplace transform and inverse Laplace transform to find the system dynamic response to an input.
7. **Analyse** common electrical circuits with RLC components.

2.2 Linear Time Invariant Systems

This is a module on signal processing, and in this context we perform signal processing through **systems**, which take a signal as an input and then return a signal as an output. We will focus on systems that we can design as engineers, i.e., with particular system processing goals in mind. For example, in communication problems, there is a natural system that distorts our communication signal, and we design a receiver system to help us recover the original signal.

We will focus our study of analogue systems in this part of the module on a particular class of systems: those that are **Linear Time Invariant** (LTI). LTI systems have particular properties when acting on input signals. Given an LTI system that is defined by the functional (i.e., function of a function) $\mathcal{F}\{\cdot\}$ acting on time-varying input signals $x_1(t)$ and $x_2(t)$, where t is time, the properties are as follows:

1. The system is **linear**, meaning that:

   (a) The system is **additive**, i.e.,

   $$\mathcal{F}\{x_1(t) + x_2(t)\} = \mathcal{F}\{x_1(t)\} + \mathcal{F}\{x_2(t)\} \tag{2.1}$$

   (b) The system is **scalable** (or **homogeneous**), i.e.,

   $$\mathcal{F}\{ax_1(t)\} = a\mathcal{F}\{x_1(t)\} \tag{2.2}$$

   for any real or complex constant a.

2. The system is **time-invariant**, i.e., if output $y(t) = \mathcal{F}\{x_1(t)\}$, then

   $$y(t-\tau) = \mathcal{F}\{x_1(t-\tau)\}\,. \tag{2.3}$$

   In other words, delaying the input by some constant time τ will delay the output and make no other changes.

Part of the convenience of working with LTI systems is that we can derive the output $y(t)$ given the input $x(t)$, if we know the system's **impulse response** $h(t)$. The impulse response is the system output when the input is a Dirac delta, i.e.,

$$h(t) = \mathcal{F}\{\delta(t)\}\,. \tag{2.4}$$

Given the impulse response $h(t)$ of a system, the output is the **convolution** of the input signal with the impulse response, i.e.,

$$y(t) = \int_0^t x(\tau)\,h(t-\tau)\,d\tau = x(t) * h(t) = h(t) * x(t)\,. \tag{2.5}$$

Convolution can be described as "**flip-and-shift**", because in effect it flips one function about the horizontal time axis (around $t = 0$) and then measures the area as that function is shifted past the other one.

While it is nice that the output of convolution is still a time-domain signal, it is often cumbersome to use in practice and it doesn't offer much intuition to help us design $h(t)$ or interpret it for a particular system. To help us out, we will find it helpful to take advantage of the **Laplace transform**.

2.3 The Laplace Transform

2.3.1 Laplace Transform Definition

The **Laplace transform** $\mathcal{L}\{\cdot\}$ is a very widely used signal transformation in the physical sciences and engineering. It converts a time-domain function $f(t)$, which we will assume is causal, i.e., $f(t) = 0, \forall t < 0$, into a complex domain function $F(s)$ that we say is in the Laplace domain or the s-domain. It is defined as follows:

$$\mathcal{L}\{f(t)\} = F(s) = \int_{t=0}^{\infty} f(t)\, e^{-st} dt, \tag{2.6}$$

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

EXTRA 2.1: The Bilateral Laplace Transform

The definition above for the Laplace transform is formally for the *unilateral* Laplace transform and is specifically for causal signals. But what if our function $f(t)$ is not causal, i.e., what if it is non-zero for some negative values of t? In this case, we should use the *bilateral* Laplace transform, which is defined as

$$F(s) = \int_{t=-\infty}^{\infty} f(t)\, e^{-st} dt. \tag{2.7}$$

Why would we not always use this form, since it is more general? The bilateral transform is more involved than the unilateral transform, particularly when evaluating its inverse, because we need to consider its **Region of Convergence**. Since we are generally more interested in causal signals, and your Engineering data book only provides a table for unilateral Laplace transforms anyway, we will just assume that all Laplace transforms in this module are unilateral. For additional details, refer to Section 1.10 of Lathi and Green.

You will likely have already seen the Laplace transform in previous modules (hopefully!). The notation may have been slightly different, but we will present it consistently here. While we will formally write the transform as $F(s) = \mathcal{L}\{f(t)\}$, we will also show conversion between the time and Laplace domains with a double arrow, i.e., by writing

$$f(t) \Longleftrightarrow F(s). \tag{2.8}$$

The double arrow suggests that we can also reverse the transform to go from the Laplace domain back to the time domain, and this is true. The **inverse Laplace transform** $\mathcal{L}^{-1}\{\cdot\}$ is defined as follows:

$$\mathcal{L}^{-1}\{F(s)\} = f(t) = \frac{1}{2\pi j} \lim_{T\to\infty} \int_{s=\gamma-jT}^{\gamma+jT} F(s)\, e^{st} ds, \tag{2.9}$$

which converts the complex function $F(s)$ into the causal ($t > 0$) time-domain signal $f(t)$.

Example 2.1: Laplace Transform of the Step Function

Find the Laplace transform of the step function

$$f(t) = u(t) = \begin{cases} 1, & t \geq 0, \\ 0, & t < 0. \end{cases} \tag{2.10}$$

Using the definition of the Laplace transform, we can write

$$\begin{aligned} F(s) &= \int_{t=0}^{\infty} f(t)\, e^{-st} dt \\ &= \int_{t=0}^{\infty} e^{-st} dt \\ &= -\frac{e^{-st}}{s}\bigg|_{0}^{\infty} = \boxed{\frac{1}{s}}, s \neq 0. \end{aligned} \tag{2.11}$$

Example 2.2: Laplace Transform of an Exponential Function

Find the Laplace transform of the exponential function

$$f(t) = e^{at}, \tag{2.12}$$

for real constant a. Using the definition of the Laplace transform, we can write

$$\begin{aligned} F(s) &= \int_{t=0}^{\infty} f(t)\, e^{-st} dt \\ &= \int_{t=0}^{\infty} e^{at} e^{-st} dt = \int_{t=0}^{\infty} e^{(a-s)t} dt \\ &= \frac{1}{a-s} e^{(a-s)t} \Big|_0^{\infty} = \boxed{\frac{1}{s-a}}, s \neq a. \end{aligned} \tag{2.13}$$

So far, it is not easy to appreciate the benefit of using the Laplace transform over a convolution to study systems. However, the Laplace transform is almost entirely *bijective* (and for this module we will always assume that it is bijective), which means there is a one-to-one mapping between the time-domain function $f(t)$ and its transform $F(s)$, i.e., a time-domain function has a unique Laplace domain function and vice versa. Common function pairs are published in tables and there is a Laplace transform table in the Engineering Data Book; a brief list of common transform pairs is presented here in Example 2.3.

Example 2.3: Sample Laplace Transform Pairs

$$\begin{array}{ccc} f(t) & & F(s) \\ 1 & \Longleftrightarrow & \frac{1}{s} \\ e^{at} & \Longleftrightarrow & \frac{1}{s-a} \\ t^n, n \in \{1, 2, 3, \ldots\} & \Longleftrightarrow & \frac{n!}{s^{n+1}} \\ \cos(at) & \Longleftrightarrow & \frac{s}{s^2+a^2} \\ \sin(at) & \Longleftrightarrow & \frac{a}{s^2+a^2} \end{array}$$

We can now use a Laplace transform table to simplify an example.

Example 2.4: Inverse Laplace Example

Find the inverse Laplace transform of the following equation:

$$F(s) = \frac{s+4}{s(s+2)}. \tag{2.14}$$

Before applying the transform table, we need to decompose $F(s)$ using partial fractions.

$$F(s) = \frac{2}{s} - \frac{1}{s+2}$$
$$\implies f(t) = \boxed{u(t)(2 - e^{-2t})}$$

In fact, our standard approach to find an inverse Laplace transform is to apply partial fractions and then use a table to invert each of the components.

2.3.2 Laplace Transform Properties

The Laplace transform has several properties that we will find relevant in this module. They are as follows:

1. **Linearity** - the Laplace transform of a sum of scaled functions is equal to the sum of scaled Laplace transforms of the individual functions, i.e.,

$$af_1(t) + bf_2(t) \Longleftrightarrow aF_1(s) + bF_2(s), \tag{2.15}$$

which we have already applied in Example 2.4.

2. **Derivatives** - the Laplace transform of the derivative of a smooth function (i.e., no discontinuities) is as follows:

$$\mathcal{L}\{f'(t)\} = s\mathcal{L}\{f(t)\} - f(0) = \boxed{sF(s) - f(0)}. \tag{2.16}$$

We have a proof in the following example.

Example 2.5: Proof of the Laplace Transform of the Derivative

Use integration-by-parts to find the Laplace transform of the derivative of a function $f(t)$.

$$\begin{aligned}\mathcal{L}\{f'(t)\} &= \int_{t=0}^{\infty} f'(t)\, e^{-st} dt \\ &= e^{-st} f(t)\Big|_0^{\infty} + \int_{t=0}^{\infty} s e^{-st} f(t)\, dt \\ &= 0 - f(0) + s\int_{t=0}^{\infty} e^{-st} f(t)\, dt \\ &= sF(s) - f(0)\end{aligned}$$

We can also use recursion to find higher order derivatives, e.g.,

$$\mathcal{L}\{f''(t)\} = \mathcal{L}\left\{\frac{df'(t)}{dt}\right\} = \boxed{s^2F(s) - sf(0) - f'(0)}$$

In general and ignoring initial conditions (i.e., at time $t = 0$), the Laplace transform of the nth order derivative is

$$f^n(t) \Longleftrightarrow \boxed{s^nF(s)}. \tag{2.17}$$

3. **Integrals** - the Laplace transform of the integral of a smooth function is as follows:

$$\int_{t=0}^{\infty} f(t)\, dt \Longleftrightarrow \boxed{\frac{F(s)}{s}}. \tag{2.18}$$

   We omit the proof because we will generally find the Laplace transform of the derivative to be more useful than that of the integral.

4. **Delay (in both domains)** - delaying a signal in one domain corresponds to multiplying by an exponential in the other domain. More precisely, we have the following two transforms:

$$f(t-T) \Longleftrightarrow e^{-sT}F(s) \tag{2.19}$$

$$e^{at}f(t) \Longleftrightarrow F(s-a), \tag{2.20}$$

   for constant time shift T or Laplace shift a.

2.4 Transfer Functions

Now that we are caught up on Laplace transforms, we can proceed to apply them to LTI systems. This is because there is another very important property about the

Laplace transform. The Laplace transform of the convolution of two functions is the product of the Laplace transforms of the two functions, i.e.,

$$\mathcal{L}\left\{x\left(t\right)*h\left(t\right)\right\}=\mathcal{L}\left\{x\left(t\right)\right\}\mathcal{L}\left\{h\left(t\right)\right\}=X\left(s\right)H\left(s\right). \tag{2.21}$$

In other words, **convolution in the time domain is equal to multiplication in the Laplace domain**. Recalling our convention that $x\left(t\right)$ is the system input and $h\left(t\right)$ is the system impulse response, we can combine this property with our understanding of LTI systems by writing the Laplace transform of the system output as

$$\mathcal{L}\left\{y\left(t\right)\right\}=Y(s)=X\left(s\right)H\left(s\right). \tag{2.22}$$

So, we can find the time-domain output $y\left(t\right)$ of an LTI system by

1. Transforming $x\left(t\right)$ and $h\left(t\right)$ into the Laplace domain.
2. Finding the product $Y(s)=X\left(s\right)H\left(s\right)$.
3. Taking the inverse Laplace transform of $Y(s)$.

Although this series of steps may seem difficult, it is often easier than doing convolution in the time domain. Furthermore, as we will see in the following lectures, we can gain a lot of insight about a system by analysing it in the Laplace domain. We refer to the Laplace transform $H\left(s\right)$ of the system impulse response as the system's **transfer function**.

Example 2.6: System Responses with Simple Inputs

There are two special cases of system response that we will regularly see.

1. When the input is a delta function, i.e., $x\left(t\right)=\delta(t)$, then we know that $X\left(s\right)=1$ and so we immediately have $Y(s)=H\left(s\right)$. This can also be proven in the time domain, i.e., $y\left(t\right)=h\left(t\right)$, using convolution.
2. When the input is a step function, i.e., $x\left(t\right)=u(t)$, then we know that $X\left(s\right)=1/s$ and so $Y(s)=H\left(s\right)/s$. This can be proven in the time domain from convolution and the integral property of the Laplace transform.

EXTRA 2.2: Proof that Convolution in the Time Domain is Equivalent to Multiplication in the Laplace Domain

We want to prove that $\mathcal{L}\{x(t) * h(t)\} = X(s) H(s)$. We will start with the definition of the Laplace transform of $y(t)$ and apply a change of variables.

$$\begin{aligned}
\mathcal{L}\{x(t) * h(t)\} = Y(s) &= \int_{t=0}^{\infty} e^{-st} \left(\int_{\tau=0}^{t} x(\tau) h(t-\tau) d\tau \right) dt \\
&= \int_{\tau=0}^{\infty} \int_{t=\tau}^{\infty} e^{-st} x(\tau) h(t-\tau) dt d\tau \\
&= \int_{\tau=0}^{\infty} \int_{\alpha=0}^{\infty} e^{-s(\alpha+\tau)} x(\tau) h(\alpha) d\alpha d\tau \\
&= \left(\int_{\tau=0}^{\infty} e^{-s\tau} x(\tau) d\tau \right) \left(\int_{\alpha=0}^{\infty} e^{-s\alpha} h(\alpha) d\alpha \right) \\
&= \boxed{X(s) H(s)},
\end{aligned}$$

where the swapping of integrals in the second line requires application of Fubini's theorem, and in the third line we define $\alpha = t - \tau$.

We can also re-arrange the product $Y(s) = X(s) H(s)$ to write the definition of the system transfer function $H(s)$ from the time-domain input and output, i.e.,

$$H(s) = \frac{\mathcal{L}\{y(t)\}}{\mathcal{L}\{x(t)\}}, \tag{2.23}$$

which is valid for a linear system when the initial conditions are zero. We will commonly see this kind of formulation in circuit analysis, where the input and output are currents or voltages associated with the circuit.

Example 2.7: System Response of Simple Circuit

Consider the capacitive circuit in Fig. 2.1. What is the transfer function of this circuit? Assume that the input is the voltage $v(t)$ and the output is the current $i(t)$.

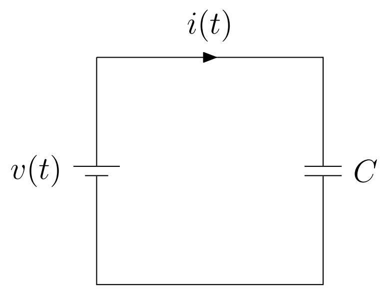


Figure 2.1: Simple capacitive circuit for Example 2.7.

From circuit theory we (should) know that:

$$v(t) = \frac{1}{C}\int_0^t i(\tau)d\tau$$
$$\implies V(s) = \frac{I(s)}{sC}$$
$$\implies H(s) = \frac{I(s)}{V(s)} = \boxed{sC}$$

Generally, we will find it helpful to know the voltage potential drops across the 3 common passive circuit elements in the Laplace domain. Given the current $I(s)$ through the element, the potential drop $V(s)$ across the element is:

1. $V(s) = RI(s)$ across resistance R.
2. $V(s) = sLI(s)$ across inductance L.
3. $V(s) = \frac{I(s)}{sC}$ across capacitance C.

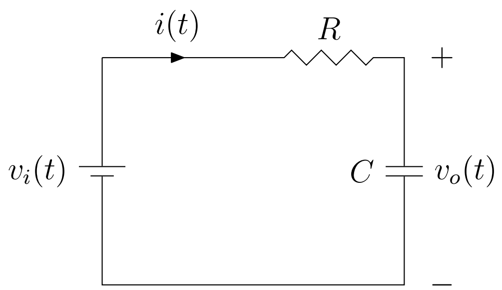


Figure 2.2: More complex capacitive circuit for Example 2.8.

Example 2.8: System Response of More Complex Circuit

Consider the capacitive circuit in Fig. 2.2. What is the transfer function of this circuit? Assume that the input is the voltage $v_i(t)$ and the output is the voltage $v_o(t)$.

Using result of the previous example we already have

$$V_o(s) = \frac{I(s)}{sC}.$$

From Kirchoff's circuit law we can also write

$$\begin{aligned} V_i(s) &= RI(s) + V_o(s) \\ &= RI(s) + \frac{I(s)}{sC} \\ \implies H(s) = \frac{V_o(s)}{V_i(s)} &= \frac{\frac{I(s)}{sC}}{RI(s) + \frac{I(s)}{sC}} \\ &= \boxed{\frac{1}{sRC + 1}} \end{aligned}$$

2.5 Summary

- The output of a **linear time invariant** (LTI) system can be found by convolving the input with the system **impulse response**.
- We can use the Laplace transform to convert an LTI system into the Laplace domain, where the impulse response is known as the **transfer function**. Convolution in the time domain is equivalent to multiplication in the Laplace domain, so we can more readily find the output of a system in the Laplace domain.
- Laplace domain analysis is particularly helpful when analysing circuits.

2.6 Further Reading

- Sections 1.5, 1.10 of "Essentials of Digital Signal Processing," B.P. Lathi and R.A. Green, Cambridge University Press, 2014.

2.7 End-of-Lesson Problems

1. Find the Laplace transform of the following functions:
   (a) $f(t) = 4t^2$
   (b) $f(t) = t\sin(2t)$
   (c) $f(t) = 2te^{2t} + t^2$
   (d) $f(t) = e^{3t}\cos(2t) + e^{3t}\sin(2t)$
   (e) $f(t) = e^{-3t}t\sin(2t)$
2. Find the inverse Laplace transform of the following functions:
   (a) $F(s) = \frac{1}{(s-2)^2+9}$
   (b) $F(s) = \frac{5s+10}{s^2+3s-4}$
3. Solve the differential equation $f''(t) - 3f'(t) + 2f(t) = 0$, given that $f(0) = 0$ and $f'(0) = 1$.
4. Use Euler's formula and the transform $e^{at} \Longleftrightarrow \frac{1}{s-a}$ to prove that the Laplace transform of $\cos(\omega t)$ is $\frac{s}{s^2+\omega^2}$
5. Use the Laplace transform of the first order derivative of a function to prove that the Laplace transform of the second order derivative is $\mathcal{L}\{f''(t)\} = s^2F(s) - sf(0) - f'(0)$
6. Find the time-domain output $y(t)$ of an LTI system defined by transfer function
$$H(s) = \frac{1}{1+\tau s}$$
when the input $x(t)$ is i) A step function, and ii) A ramp function, i.e., $x(t) = t$.
7. Find the transfer function $H(s) = \frac{V_o(s)}{V_i(s)}$ for the circuit in Figure 2.3
8. Find the transfer function $H(s) = \frac{V_o(s)}{V_i(s)}$ for the circuit in Figure 2.4
9. Assume that the operational amplifier in Fig. 2.5 is perfect. Find the transfer function $H(s) = \frac{V_o(s)}{V_i(s)}$ for the circuit.

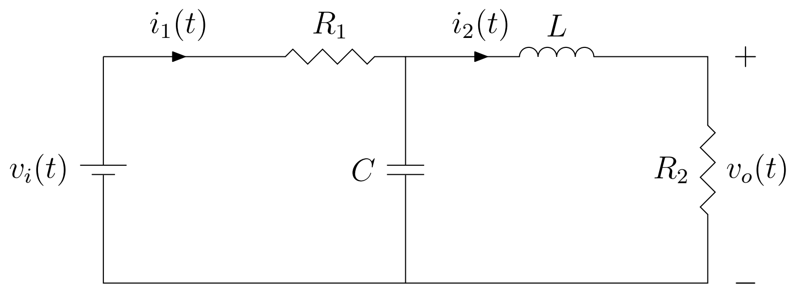


Figure 2.3: A two-loop circuit.

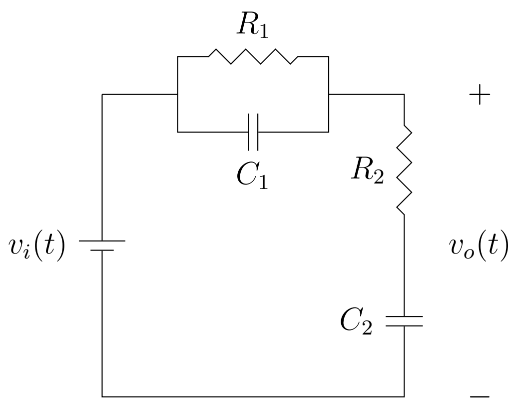


Figure 2.4: A second order filter circuit.

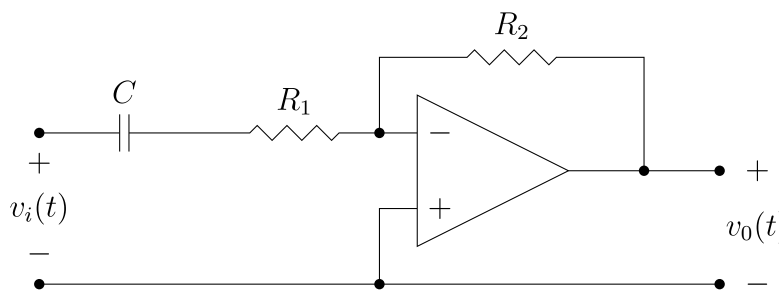


Figure 2.5: Op-amp circuit.

Lesson 3

Poles, Zeros, and Stability of Analogue Systems

System stability is a very important consideration in system design, as we will see in future lessons throughout this module. In this lesson, we consider the stability of analogue systems. We will see that Laplace domain analysis makes this possible by looking at the roots of the system's transfer function.

3.1 Learning Outcomes

By the end of this lesson, you should be able to ...

1. **Use** an analogue system's transfer function to identify its poles and zeroes.
2. **Apply** the criterion for analogue system stability.
3. **Analyse** the dynamic response of an analogue system with real poles.
4. **Analyse** the dynamic response of an analogue system with complex poles.

3.2 Poles and Zeros

Generally, a transfer function $H(s)$ in the Laplace domain can be written as a fraction of two polynomials, i.e.,

$$H(s) = \frac{b_0 s^M + b_1 s^{M-1} + \ldots + b_{M-1} s + b_M}{a_0 s^N + a_1 s^{N-1} + \ldots + a_{N-1} s + a_N}, \tag{3.1}$$

where the numerator is a Mth order polynomial with coefficients bs, and the denominator is a Nth order polynomial with coefficients as. For a system to be real, then the order of the numerator polynomial must be no greater than the order of the denominator polynomial, i.e., $M \leq N$.

Eq. 3.1 may look intimidating, but it is general. If we factorize Eq. 3.1 into its roots, then we can re-write $H(s)$ as a ratio of products, i.e.,

$$H(s) = K\frac{(s-z_1)(s-z_2)\ldots(s-z_M)}{(s-p_1)(s-p_2)\ldots(s-p_N)} \tag{3.2}$$

$$= K\frac{\prod_{i=1}^{M} s - z_i}{\prod_{i=1}^{N} s - p_i} \tag{3.3}$$

where we emphasise that the coefficient on each s is 1. Eq. 3.1 has several key definitions:

- The roots z of the numerator are called the **zeros**. When s is equal to any z_i, the transfer function $H(s) = 0$.
- The roots p of the denominator are called the **poles**. When s is equal to any p_i, the transfer function $H(s)$ will be infinite (and we will soon see that this relates to stability...)
- K is the overall transfer function **gain**.

Note: later in this module we will see the z-transform. It will be important to not confuse it with the zeros of a transfer function. Unfortunately, due to the names of the terms, re-use of variables is unavoidable here.

We will often find it useful to plot the poles and zeros of a transfer function on the complex domain of s, which we call a **pole-zero plot**. The convention is to plot zeros as circles (o for 0, i.e., don't fall in the hole!) and poles as "x"s (i.e., stay away from these!).

Example 3.1: First Pole-Zero Plot

Plot the pole-zero plot for the transfer function

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

First we need to figure out the poles and zeros, i.e., the roots of the transfer function. This transfer function is already in a form where the roots are factorized. We can see that the numerator is zero when $s = -1$, so $z - 1$ and the denominator is zero when $s = 0$, so $p = 0$. We also see that the gain is $K = 1$.

We need to draw a 2D axes, where the real components of s lie along the horizontal axis and the imaginary components of s lie along the vertical axis. The plot is shown in Fig. 3.1.

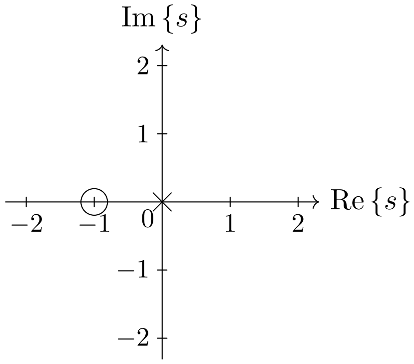


Figure 3.1: Pole-zero plot for Example 3.1.

More generally, poles and zeros are complex, i.e., they do not need to lie on the horizontal axis of the s-plane. In the following, we focus on the features of analogue systems with real and complex poles.

3.3 Poles and Dynamic Responses

3.3.1 Real Poles

We will see that the poles of a system determine the general form of its response. First, consider a system where all of the poles are *real*. If we consider a step input to the system, then from the previous lesson we know what we can write the output in the Laplace domain as

$$\begin{aligned} Y(s) &= \frac{1}{s}H(s) &(3.4)\\ &= \frac{1}{s}\times\frac{K\times\text{Zero Polynomial}}{(s-p_1)(s-p_2)(s-p_3)\ldots} &(3.5)\\ &= \frac{\text{some constant}}{s}+\frac{\text{some constant}}{s-p_1}+\frac{\text{some constant}}{s-p_2}+\ldots, &(3.6)\end{aligned}$$

where the numerators will all be constants that depend on the form of the numerator polynomial (i.e., zero polynomial). By partial fraction expansion, the output $Y(s)$ is a sum of fractions where each denominator is a linear function of s. Since we've assumed that each pole is real, we can apply the transforms $1 \Longleftrightarrow \frac{1}{s}$ and $e^{at} \Longleftrightarrow \frac{1}{s-a}$ and see that the time-domain output signal $y(t)$ is scaled constant plus a sum of (hopefully decaying!) exponentials. In this particular example, the scaled constant came from the input; more generally, we will see that the input dominates the first term of the output.

Thus, if we take the inverse Laplace transform of *only* the terms associated with the poles, then we have

$$y_{\text{poles}}(t) = [\text{some constant}]e^{p_1 t} + [\text{some constant}]e^{p_2 t} + \ldots \tag{3.7}$$

The main takeaways for systems with real poles are as follows:

- The system input will (hopefully) dominate the output.
- Each real pole corresponds to an exponential component in the time domain with a rate that depends on that pole.
- The magnitude of each exponential component depends on the zeros.

Example 3.2

Consider an LTI system with the impulse response

$$h(t) = \frac{1}{2}e^{-t}(e^{2t} - 1)$$

Answer the following:

1. Find the system's poles and zeros and sketch a pole-zero plot.
2. Find the dynamic response in the time domain, subject to a step input.
3. Sketch the dynamic response and comment on the role of the poles and zeros.

We can re-write the impulse response as

$$h(t) = \frac{e^t}{2} - \frac{e^{-t}}{2}.$$

The Laplace transform gives the transfer function:

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

Thus, there are no zeros and 2 poles at $p = \pm 1$. Also, the gain is $K = 1$. See the pole-zero plot in Fig. 3.2.

If there is a step input, then we know $X(s) = \frac{1}{s}$ and hence the output is

$$\begin{aligned} Y(s) &= \frac{1}{s}H(s) \\ &= \frac{1}{s}\frac{1}{(s-1)(s+1)} \\ &= \frac{1}{2(s-1)} - \frac{1}{s} + \frac{1}{2(s+1)} \\ \implies y(t) &= \boxed{-1 + \frac{e^{-t}}{2} + \frac{e^{t}}{2}} \end{aligned}$$

A plot of this output is shown in Fig. 3.3. We see that the +1 exponent dominates for time $t > 0$ (in fact this system is unstable; we will soon see why), and the −1 exponent dominates for $t < 0$ (just that this isn't actually a real system!).

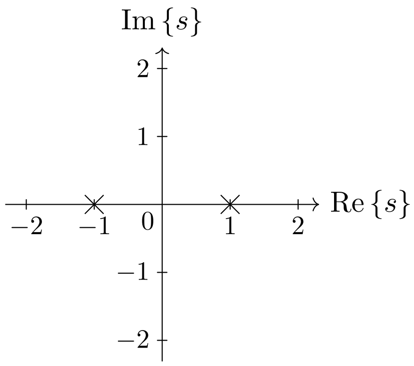


Figure 3.2: Pole-zero plot for Example 3.2.

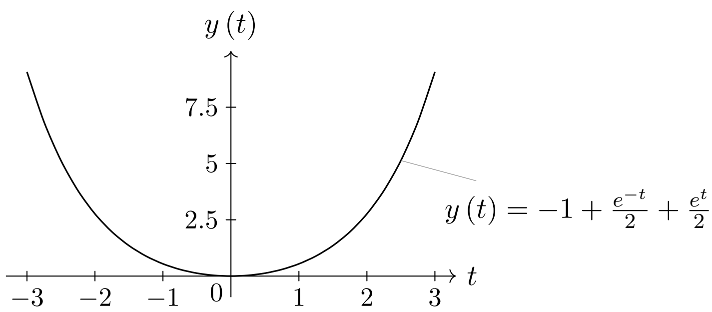


Figure 3.3: Dynamic response for Example 3.2.

3.3.2 Complex Poles

Now let us consider a system with complex poles. Rather than write the most general form, let us jump right to a more specific example and consider the transfer function

$$H\left(s\right) = \frac{As+B}{(s+\alpha)^2+\beta^2}, \tag{3.8}$$

for real constants α and β. In this case, the poles exist on the complex s-plane at $s = -\alpha \pm j\beta$. We see that there are two poles, and in fact they are a **conjugate pair**. In practice, any poles that we will see with an imaginary component will be part of a conjugate pair. What is more interesting, however, is the impact of complex poles on the system response. Let us find the inverse Laplace transform of $Y(s)$ by first finding suitable numerators:

$$H\left(s\right) = \frac{As+B}{(s+\alpha)^2+\beta^2} = \frac{A(s+\alpha)}{(s+\alpha)^2+\beta^2} + \frac{C\beta}{(s+\alpha)^2+\beta^2}, \tag{3.9}$$

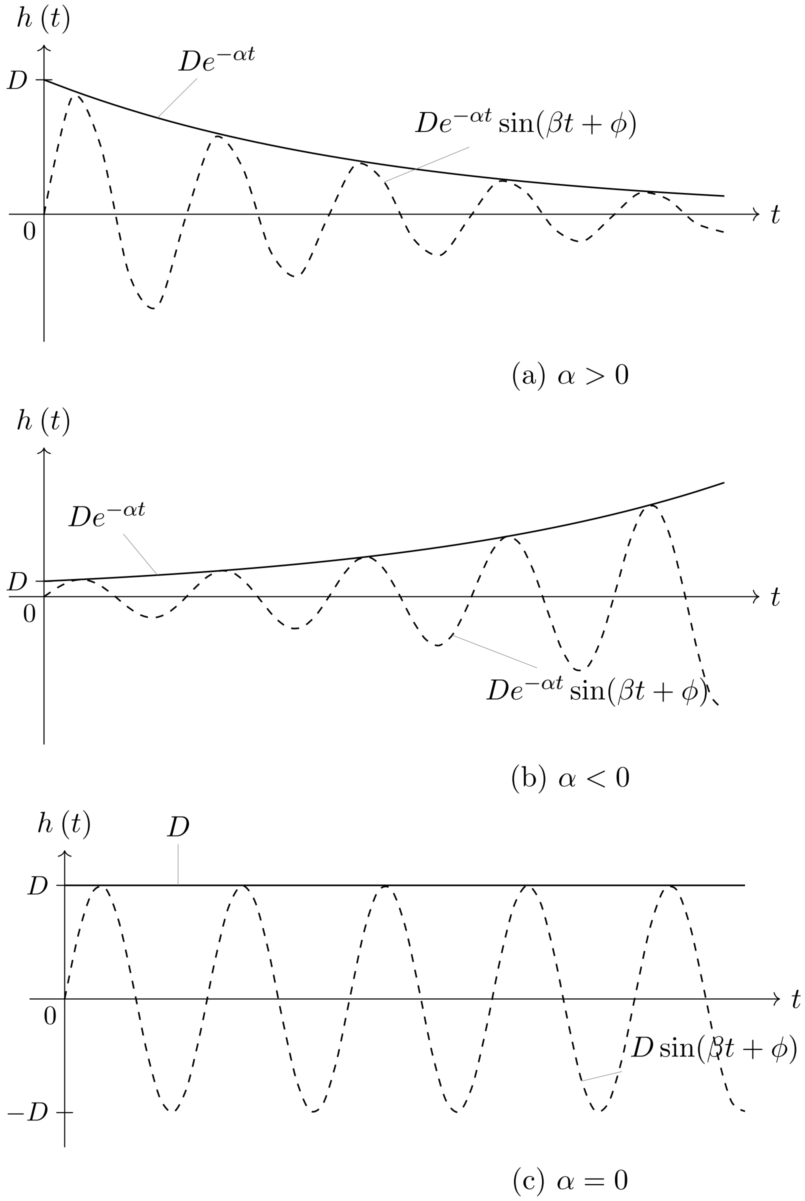
where we have introduced the constant $C = \frac{B-A\alpha}{\beta}$. We can now take the inverse Laplace transform as

$$h\left(t\right) = Ae^{-\alpha t}\cos(\beta t) + Ce^{-\alpha t}\sin(\beta t) = De^{-\alpha t}\sin(\beta t+\phi), \tag{3.10}$$

where $D = \sqrt{A^2+C^2}$ and $\tan\phi = \frac{A}{C}$. The important result here is that *complex poles lead to an oscillatory response* in the time domain. The real component of the pole ($-\alpha$) corresponds to the power of the exponent, and the imaginary component of the pole (β) corresponds to the frequency of oscillation. A critical parameter, then, is the value of α, or more specifically the *sign*. If $\alpha > 0$, then the exponent $e^{-\alpha t}$ will decay with time. If $\alpha < 0$, then the exponent $e^{-\alpha t}$ will grow with time. At the transition is where $\alpha = 0$ and the response stays the same. We visualize these three cases in Fig. 3.4.

3.3.3 Stability Criterion for Analogue Systems

The preceding discussion about poles brought us very close to discussing stability. In fact, we should now define it. A system is considered to be **stable** if its impulse response tends to zero in the time domain. If the impulse response does not tend to zero or some finite value, then the system is **unstable**. We have seen that this depends on the locations of the poles, so we can use the poles to define a stability criterion for a system. In particular, if all of a system's poles are on the left half of the complex s-plane (i.e., to the left of the imaginary s-axis), then the system is stable. We see this criterion visually in Fig. 3.5. Any pole satisfying this condition corresponds to a decaying exponential in the time domain.



(a) $\alpha > 0$

(b) $\alpha < 0$

(c) $\alpha = 0$

Figure 3.4: Impulse response of system with complex poles for different values of α.

It is worth mentioning what to do when a system has a pole that is *on* the imaginary s-axis. Is such a system stable? From Fig. 3.5, we might be inclined to say yes. However, as we can see in Fig. 3.4(c), such a system does not satisfy our definition of stability, as the impulse response does not tend to zero over time. Such a system is said to be **marginally stable**.

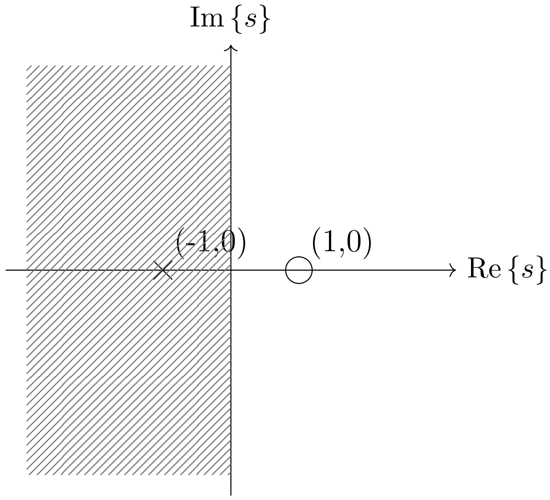


Figure 3.5: Pole-zero plot showing the stability region for poles with shaded cross hatches.

EXTRA 3.1: Alternative Definitions of Stability

Lathi and Green define stability from a slightly different perspective. They call a system **bounded-input bounded-output (BIBO) stable** if and only if every bounded (finite) input $x(t)$ results in bounded output $y(t)$. A system is **asymptotically stable** if bounded initial conditions result in bounded output. In practice for most systems, these two definitions are *equivalent* and also equal to our definition based on the impulse response. Under all 3 definitions, a system whose transfer function's poles are all to the left of the imaginary s-axis is stable and having poles to the right of the imaginary s-axis makes the system unstable. The details vary for poles *on* the imaginary s-axis.

3.4 Summary

- The **zeros** are the roots to the transfer function's numerator in the Laplace domain.
- The **poles** are the roots to the transfer function's denominator in the Laplace domain.
- A system is **stable** if the impulse response tends to zero with increasing time.

- Real components of poles correspond to exponential responses.
- Imaginary components of poles correspond to the angular frequency of oscillating responses.

3.5 Further Reading

- Sections 1.5.5 and 2.1.1 of "Essentials of Digital Signal Processing," B.P. Lathi and R.A. Green, Cambridge University Press, 2014. Note that Lathi and Green use a slightly different definition of stability than what we use in this module. Our definition for stability in analogue systems is more convenient to use in consideration of time-domain behaviour.

3.6 End-of-Lesson Problems

1. Determine the zeros and poles of the system with transfer function

$$H(s) = \frac{(1 + 0.04s)(1 + 11s)}{0.44s^2 + 13.04s + 1}.$$

2. Consider a system with transfer function

$$H(s) = \frac{20}{0.03s^2 + 1.4s + 20}.$$

Determine the poles of this system. Sketch the time-domain output of the system in response to a step input. Is this system stable?

3. Determine the s-domain transfer function $H(s)$ of the system with the pole-zero plot shown in Fig. 3.6. Is this system stable?

4. Determine the poles of each of the following transfer functions and state whether they represent a stable system:

   (a) $H(s) = \frac{1}{s-1}$

   (b) $H(s) = \frac{\omega}{s^2+\omega^2}$

   (c) $H(s) = \frac{3}{s^2+2s+10}$

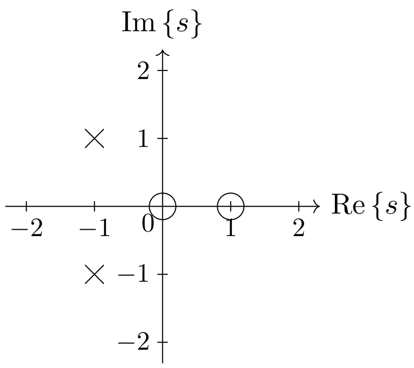


Figure 3.6: Pole-zero plot for Question 3.

Lesson 4

Analog Frequency Response

In Lesson 3, we saw how a system's pole-zero plot enabled us to describe time-domain characteristics of the system, including its stability. In this lesson, we will focus on the behaviour of systems in response to periodic inputs of a particular frequency. To do so, we will find it convenient to translate between the Laplace transform and the Fourier transform. Fortunately, this translation is straightforward and we can also use the Laplace-domain pole-zero plot to determine the frequency response of a system. Being able to analyse the frequency response of a system is a prerequisite for studying and designing analogue filters, as we will see in Lesson 5.

4.1 Learning Outcomes

By the end of this lesson, you should be able to ...

1. **Apply** knowledge of the Laplace transform to find the frequency response of an input to an analogue system.
2. **Understand** the Fourier transform and its relation to the Laplace transform.

4.2 Frequency Response

The **frequency response** of a system is its output in response to a sinusoid input of unit magnitude and some specified frequency. The frequency response is measured as 2 components: a **magnitude** and a **phase angle**. Typically, we show the frequency response in two plots (1 for magnitude; 1 for phase) as a function of the input frequency.

Generally, a system may be given inputs at different frequencies, but it is *common when we design a system to do so for a particular target frequency or range of frequencies.* Some examples:

- Telephone technologies are designed for processing the frequency ranges commonly used by a human voice.
- Household appliances are designed to operate with a mains frequency of about 50Hz (in Europe; 60Hz in North America, etc.)
- Different radio frequency bands are licensed (or open) to different technologies. Wifi commonly operates at 2.4GHz and 5GHz; other bands are reserved for specific use such as television transmissions or the military.

To find the frequency response, recall the definition of the Laplace transform:

$$F(s) = \int_{t=0}^{\infty} f(t)\, e^{-st} dt, \tag{4.1}$$

for $s = \sigma + j\omega$. Let us *ignore the real component* of s and only consider the Laplace transform on the imaginary s-axis, such that $s = j\omega$. We can then re-write the Laplace transform as

$$F(j\omega) = \int_{t=0}^{\infty} f(t)\, e^{-j\omega t} dt. \tag{4.2}$$

We emphasise that ω is the radial frequency, measured in $\frac{\text{rad}}{\text{s}}$, and we can convert to and from the frequency f in Hz via $\omega = 2\pi f$.

The special case of the Laplace transform in Eq. 4.2 is better known as the **continuous Fourier transform** and the result $F(j\omega)$ is referred to as the **spectrum** of $f(t)$ or as the **frequency response**. Similarly, we can also write the inverse of the continuous Fourier transform as

$$f(t) = \frac{1}{2\pi} \int_{\omega=-\infty}^{\infty} F(j\omega)\, e^{j\omega t} d\omega. \tag{4.3}$$

We recall from Lesson 3 that we wrote a system's transfer function as a ratio of products that included its poles and zeros. We can re-write with $s = j\omega$ as

$$H(j\omega) = K \frac{\prod_{i=1}^{M} j\omega - z_i}{\prod_{i=1}^{N} j\omega - p_i}. \tag{4.4}$$

This form of $H(j\omega)$ (which we may also simply write as $H(\omega)$) is incredibly useful for determining the frequency response of a system. We can treat $H(j\omega)$ as a function of vectors from the system's poles and zeros to the imaginary $j\omega$-axis (i.e., each pole and zero corresponds to 1 vector; effectively a line from that pole or zero to the $j\omega$-axis at frequency ω). Thus, from a pole-zero plot, we can *geometrically* determine the magnitude and phase of the frequency response.

Recall that the phasor form of a complex number separates it into its magnitude and phase, i.e.,

$$H(j\omega) = |H(j\omega)|e^{j\angle H(j\omega)}, \tag{4.5}$$

such that the magnitude component has a phase of 0 and the phase component has a magnitude of 1. First we consider the magnitude. We can take the magnitude of both sides of Eq. 4.4 as follows:

$$|H(j\omega)| = |K|\frac{\prod_{i=1}^{M}|j\omega - z_i|}{\prod_{i=1}^{N}|j\omega - p_i|}. \tag{4.6}$$

In words, the **magnitude of the frequency response** (MFR) $|H(j\omega)|$ is equal to *the gain multiplied by the magnitudes of the vectors corresponding to the zeros, divided by the magnitudes of the vectors corresponding to the poles.*

The phase depends on the sign of K. If $K > 0$, then

$$\angle H(j\omega) = \sum_{i=1}^{M}\angle(j\omega - z_i) - \sum_{i=1}^{N}\angle(j\omega - p_i). \tag{4.7}$$

If $K < 0$, then it effectively adds a phase of π and we have

$$\angle H(j\omega) = \sum_{i=1}^{M}\angle(j\omega - z_i) - \sum_{i=1}^{N}\angle(j\omega - p_i) + \pi. \tag{4.8}$$

In words, the **phase angle of the frequency response** (PAFR) $\angle H(j\omega)$ is equal to *the sum of the phases of the vectors corresponding to the zeros, minus the sum of the phases of the vectors correspond to the poles, plus* π *if the gain is negative.* Each phase vector is measured from the positive real s-axis (or a line parallel to the real s-axis if the pole or zero is not on the real s-axis).

We will now look at examples of frequency responses.

4.3 Frequency Response Examples

Example 4.1: Low Pass Filter

We will consider filters in greater detail in Lesson 5, but for now consider this transfer function of a simple low pass filter:

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

We first re-write $H(s)$ to match the form of Eq. 4.4 so that the coefficients of all roots are equal to 1:

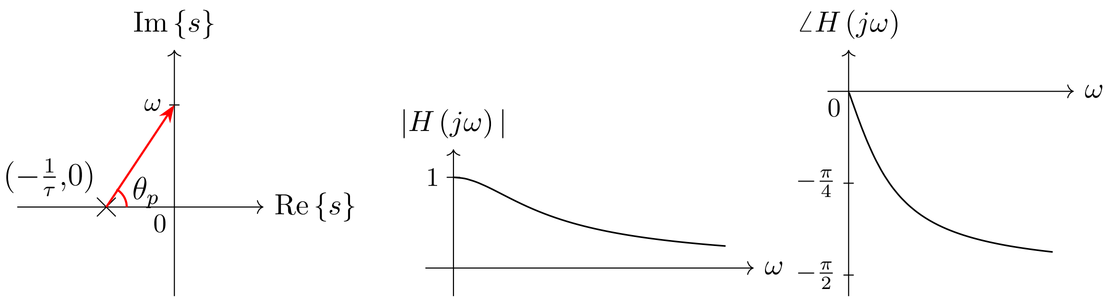
$$H(s) = \frac{1/\tau}{1/\tau + s},$$

so the gain is $K = \frac{1}{\tau}$ and there is a pole at $s = -\frac{1}{\tau}$. We sketch the pole-zero plot and draw a vector from the pole to an arbitrary (usually positive) ω in Fig. 4.1(a).

Consider what happens to the vector as we vary ω. The vector has a minimum magnitude when $\omega = 0$ and rises *monotonically* (i.e., sign doesn't change) with increasing (or decreasing) ω. So, the magnitude of this vector *increases* with ω, but since it is a pole the magnitude of the *system decreases* with ω.

The magnitude of the frequency response is shown in Fig. 4.1(b). Note that the magnitude of the pole vector at frequency $\omega = 0$ is $1/\tau$, but $K = \frac{1}{\tau}$, so the overall system magnitude is 1. This system is a **low pass filter** because it "*passes*" a strong signal (i.e., higher magnitude) at lower frequencies and passes a weak signal (i.e., *attenuates*) at higher frequencies.

The phase of the pole vector also increases monotonically with the frequency, but it converges to a maximum of $\pi/2$. Thus, the phase of the system is a negative monotonic function and is shown in Fig. 4.1(c).



(a) Pole-zero plot. (b) Magnitude plot. (c) Phase plot.

Figure 4.1: Plots for Example 4.1

Example 4.2: Band Pass Filter

Consider a system with the transfer function

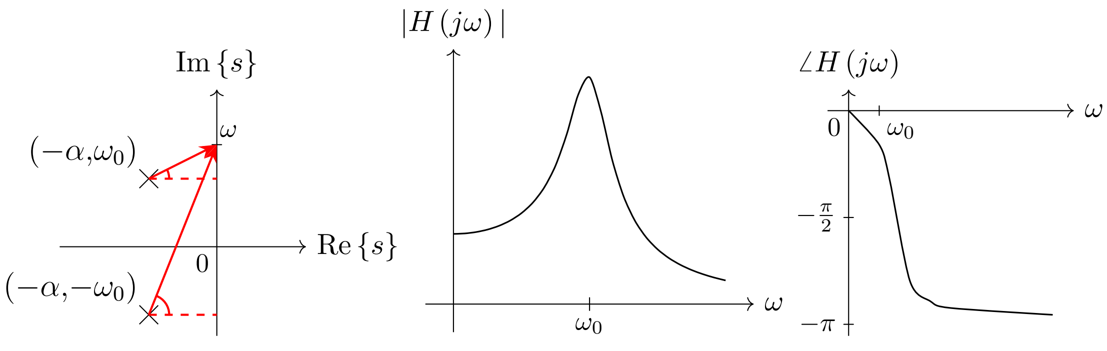
$$H(s) = \frac{1}{s^2 + 2\alpha s + (\alpha^2 + \omega_0^2)},$$

for positive real constants α and ω_0. This transfer function has no zeros, but a pair of complex conjugate poles at $s = -\alpha \pm j\omega_0$. There are now two pole vectors to deal with and they do not lie on the real s-axis. The corresponding pole-zero plot is in Fig. 4.2(a).

The magnitude of the 2 pole vectors has a minimum value at $\omega = \pm\omega_0$ (or just $\omega = \omega_0$ when assuming $\omega > 0$). The magnitude rises monotonically as the frequency moves away from ω_0. The frequency response magnitude of the system will do the opposite and decrease as we move away from ω_0, as shown in Fig. 4.2(b). This is referred to as a **band pass filter** because it passes signals with frequencies that are near the frequency ω_0 and attenuates signals at higher and lower frequencies.

By symmetry, the phase of the frequency response is 0 at $\omega = 0$. As ω increases, the angle associated with one pole vector increases while the other decreases, which results in a slow net decrease of the system phase, until both vector angles increase and the system phase decreases to $-\pi$, as shown in Fig. 4.2(c).

We have not used actual numbers for the pole coordinates in this example, but you will see if you try substituting different values of α that a smaller α results in a more pronounced peak in the magnitude of the frequency response.



(a) Pole-zero plot. (b) Magnitude plot. (c) Phase plot.

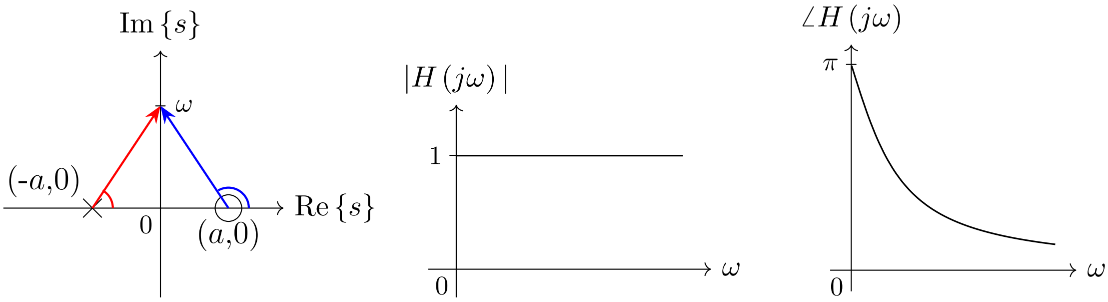
Figure 4.2: Plots for Example 4.2

Example 4.3: All Pass Filter

Consider the following filter transfer function:

$$H\left(s\right) = \frac{s-a}{s+a},$$

for positive real constant a. The pole-zero plot is shown in Fig. 4.3(a). By symmetry of the pole and zero, the pole vector will always have the same magnitude as the zero vector for any ω. Thus, the magnitude of the frequency response is always 1, as shown in Fig. 4.3(b). This is known as an **all pass filter**. While the magnitude does not change, the phase does; the zero vector has a phase of π when $\omega = 0$ whereas the pole vector has a phase of 0, then the phases of both vectors tend to $\pi/2$ with increasing ω such that their difference tends to 0. This is shown in Fig. 4.3(c).



(a) Pole-zero plot. (b) Magnitude plot. (c) Phase plot.

Figure 4.3: Plots for Example 4.3

4.4 Summary

- The **frequency response** of a system is its response to an input with unit magnitude and a fixed frequency. The response is given by the magnitude and phase of the system output as a function of the input frequency.
- The **continuous Fourier transform** is the Laplace transform evaluated on the imaginary s-axis at some frequency $s = j\omega$.
- The frequency response can be found geometrically by assessing the vectors made by the poles and zeros of the system's transfer function to the imaginary

s-axis.

4.5 Further Reading

- Section 2.1 of "Essentials of Digital Signal Processing," B.P. Lathi and R.A. Green, Cambridge University Press, 2014.

4.6 End-of-Lesson Problems

Note: For additional practice, you can try plotting the frequency response for any Laplace-domain transfer function in this part of the notes. As we will see in Lesson 7, all results can be readily verified in MATLAB.

1. Sketch the pole-zero plot, magnitude of the frequency response, and phase of the frequency response of the system with the transfer function:

$$H(s) = \frac{s}{s+1},$$

2. Sketch the pole-zero plot, magnitude of the frequency response, and phase of the frequency response of the system with the transfer function:

$$H(s) = \frac{1}{s^2+3s+2},$$

3. Consider the all pass filter from Example 4.3. The transfer function is

$$H(s) = \frac{s-a}{s+a},$$

for real positive constant a. Prove mathematically that the magnitude of the frequency response is always 1, and that the phase of the frequency response is given by

$$\angle H(j\omega) = \pi - 2\tan^{-1}\frac{\omega}{a}.$$

Lesson 5

Analogue Filter Design

In Lesson 4 we started to talk about some types of simple filters and their frequency responses. Filtering is a very common use of signal processing and practically any signal processing system will include some form of filtering. In this lesson, we focus on some of the practical details of filter design and how to design an analogue filter to meet target specifications (i.e., how to *engineer* a filter ... did you wonder when engineering would appear in this module?).

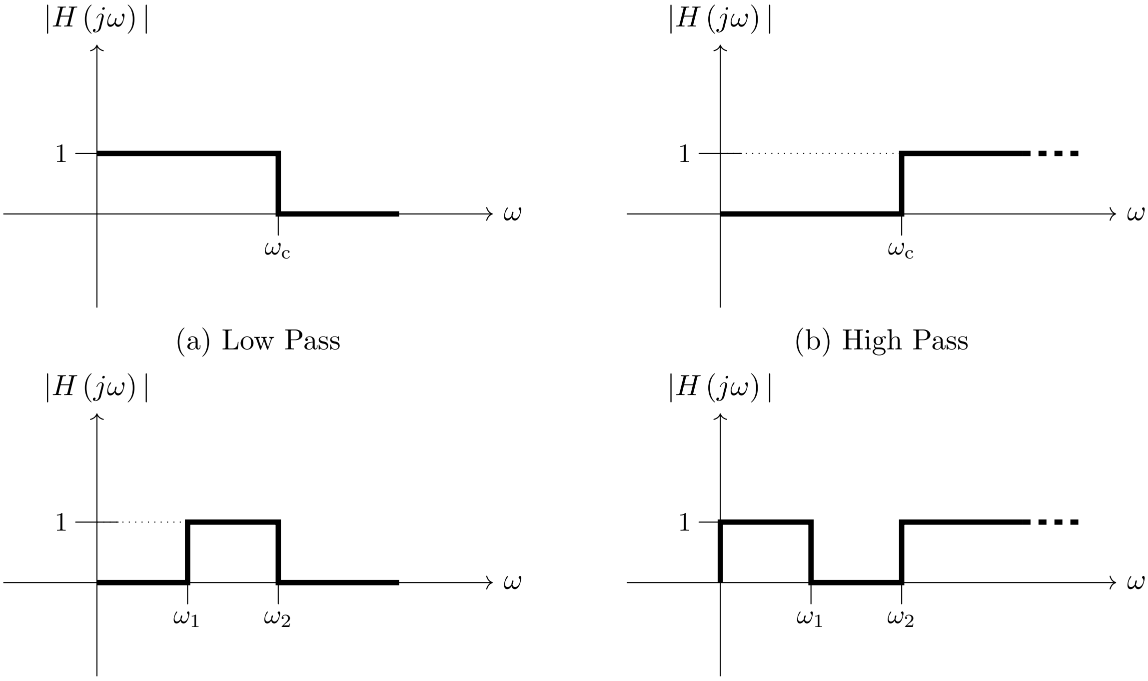
5.1 Learning Outcomes

By the end of this lesson, you should be able to ...

1. **Understand** the basic classifications of analogue filters and common filter designs.
2. **Understand** the constraints on filter realisability.
3. **Design** analogue filters to meet specified performance criteria.

5.2 Ideal Filter Responses

Our discussion of analogue filter design begins with ideal filter responses, as we show in Fig. 5.1. This enables us to define the different types of filters and helps us to establish the targets that we will try to realise with practical filters. Each ideal filter has unambiguous **pass bands**, which are ranges of frequencies that pass through the system without distortion, and **stop bands**, which are ranges of frequencies that are rejected and do not pass through the system at all. The **transition band** between stop and pass bands in ideal filters has a size of 0; transitions occur at single frequencies.



(a) Low Pass (b) High Pass (c) Band Pass (d) Band Stop

Figure 5.1: Frequency response magnitudes of common ideal analogue filters.

The four main types of filter magnitude responses are as follows:

- **Low pass** filters pass frequencies less than cutoff frequency ω_c and reject frequencies greater than ω_c.
- **High pass** filters reject frequencies less than cutoff frequency ω_c and pass frequencies greater than ω_c.
- **Band pass** filters pass frequencies that are within a specified range, i.e., between ω_1 and ω_2, and reject frequencies that are either below or above the band.
- **Band stop** filters reject frequencies that are within a specified range, i.e., between ω_1 and ω_2, and pass all other frequencies.

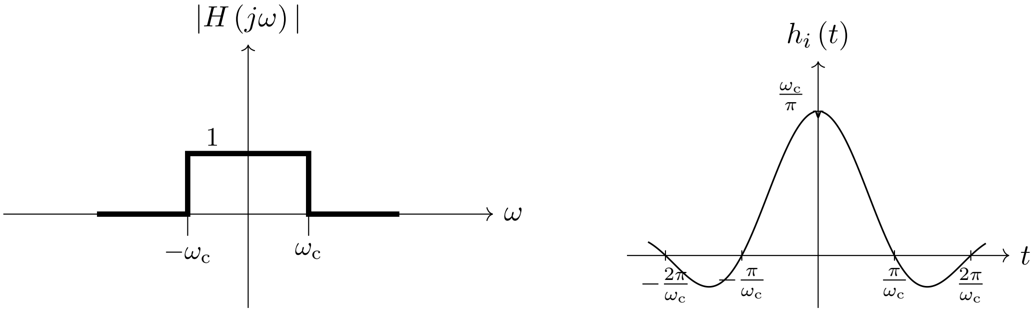
Unfortunately, none of these filters are realisable. We can show this mathematically in a very important example.

Example 5.1: Realisability of an Ideal Filter

Consider an ideal low pass filter, which acts as a rectangular pulse in the Laplace domain. We will focus on its double-sided frequency response, so we consider that it passes signals within the range $-\omega_c < \omega < \omega_c$. Is it realisable? Let's apply the inverse Fourier transform to determine the impulse response.

$$\begin{aligned} h(t) &= \frac{1}{2\pi}\int_{\omega=-\infty}^{\infty} H(j\omega)\, e^{j\omega t} d\omega \\ &= \frac{1}{2\pi}\int_{\omega=-\omega_c}^{\omega_c} e^{j\omega t} d\omega \\ &= \frac{1}{2\pi j t} e^{j\omega t}\Big|_{\omega=-\omega_c}^{\omega=\omega_c} \\ &= \frac{1}{2\pi j t}\left[e^{j\omega_c t} - e^{-j\omega_c t}\right] \\ &= \frac{1}{\pi t}\sin(\omega_c t)\frac{\omega_c t}{\omega_c t} \\ &= \frac{\omega_c}{\pi}\,\mathrm{sinc}(\omega_c t). \end{aligned}$$

Thus the impulse response is a scaled sinc function, which has non-zero values for $t < 0$ as shown in Fig. 5.2. This means that the system starts to respond to an input *before* that input is applied, so this filter is *unrealisable*. In fact, none of the ideal filters are realisable.



(a) Frequency response magnitude. (b) Impulse Response.

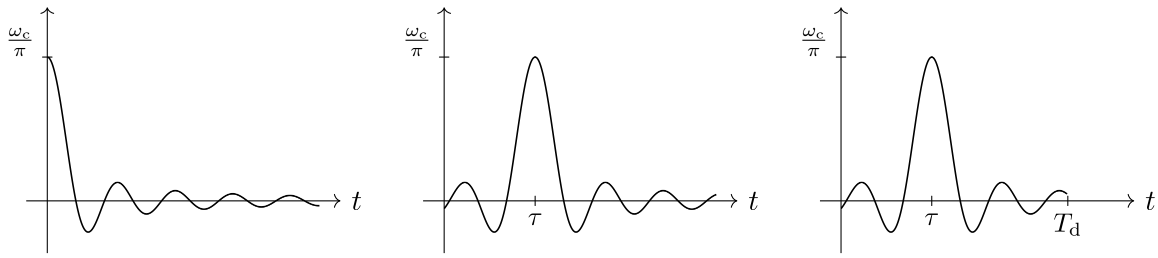
Figure 5.2: Responses of ideal low pass filter in Example 5.1.

5.3 Realising Filters

So, how to realise a practical filter? Even if we cannot implement an ideal filter, we seek smooth behaviour in the pass bands and steep transitions to the stop bands. We have seen simple realisations of filters in Lesson 4, but their frequency responses deviated quite significantly from the ideal filter behaviour. It turns out that we can do much better with filters that are not only (theoretically) realisable but also practical to implement with circuits.

First, let's consider what we could do to the impulse response in Fig. 5.2(b) to make it realisable. If we just dropped the portion of $h_i(t)$ for $t < 0$, i.e., used $h_i(t)\,u(t)$ then we would not get suitable behaviour in the frequency domain because we have discarded 50% of the system energy. However, if we are able to tolerate delays, then we can shift the sinc function to the right so that more of the energy is in causal time, i.e., use $h_i(t-\tau)\,u(t)$. As we recall from Laplace transforms, a shift in the time domain corresponds to scaling by a complex exponential in the Laplace domain. This is also true for the Fourier transform. So, *a delay in time maintains the magnitude of the frequency response* but it changes the phase.

Is waiting sufficient to realise a filter? If we can wait forever for our filter output, then yes. But in practice we will have a **delay tolerance** T_d that we are willing to wait for signal output, which means that we need to further **truncate** our impulse response $h_i(t)$ so that it is finite, i.e., $h_i(t-\tau)\,u(t) = 0$ for $t > T_\mathrm{d}$. We show the combined effects of causality, shifting, and truncating in Fig. 5.3.



(a) Causal response $h_i(t)\,u(t)$

(b) Shifted response $h_i(t-\tau)\,u(t)$.

(c) Truncation of shifted response.

Figure 5.3: Realising a filter by transforming unrealisable behaviour.

Are we done? Not quite. Truncation can cause unexpected problems in the frequency domain, so we also usually apply **windowing**, which scales the non-truncated impulse response to improve the frequency response behaviour. We won't worry about windowing for analogue systems, but we will re-visit this idea for digital systems. A more serious issue is the actual physical implementation of a filter with a truncated impulse response, which do not generally produce a *rational* transfer function $H(s)$. However, fortunately, there are families of practical filters that have rational transfer functions, which make them implementable with standard electric circuit components. We will see some examples of these in the following section.

5.4 Practical Filter Families

Before we discuss specific practical filters, we need to define terminology that is used to set and describe practical filter performance. Our description here is from the perspective of a low pass filter, but the same ideas apply to the other types of filter as well. A practical filter tends to have a frequency response magnitude in the form of that in Fig. 5.4. The main characteristics are the following:

- There tends to not be a true stop band, as there is always some non-zero magnitude for any finite frequency. So, we define a small maximum **stop band gain** G_s, such that the gain in a stop band is no greater than this value. The threshold frequency that needs to satisfy this constraint is the **stop band frequency** ω_s.
- The magnitude of the frequency response is not actually flat over the entire pass band. So, we define a minimum **pass band gain** G_p such that the gain in a pass band is no less than this value. The threshold frequency that needs to satisfy this constraint is the **pass band frequency** ω_p.
- There is a non-zero gap between the pass band frequency and the stop band frequency that we call the **transition band**. We typically want this to be as steep as possible, but in generally to do so also increases the complexity of the filter and its implementation.

Typically, the gains G_p and G_s are defined on a logarithmic scale in decibels (dB). We convert between a linear gain and dB gain as follows:

$$G_\mathrm{dB} = 20\log_{10} G_\mathrm{linear} \tag{5.1}$$

$$G_\mathrm{linear} = 10^{\frac{G_\mathrm{dB}}{20}}. \tag{5.2}$$

Note: the transfer function gain K that appears in the pole-zero factorization form of $H(s)$ is *different* from the filter gain G that refers to the magnitude of the frequency response at a particular frequency. Unfortunately, naming conventions mean that we need to refer to both of them as gains.

5.4.1 Butterworth Filters

A common filter design is the **Butterworth filter**. Butterworth filters are *maximally flat* in the pass band, i.e., they are as flat as possible for the given order of

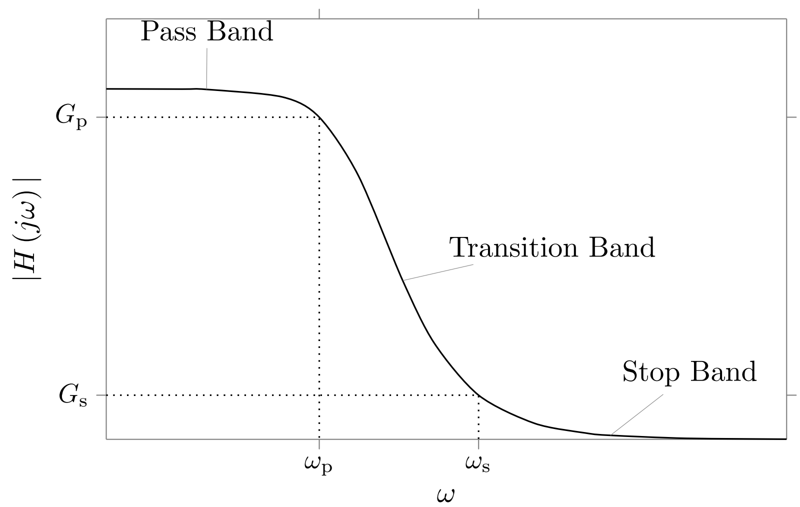


Figure 5.4: Frequency response magnitude of realizable analogue low pass filter.

the filter. The transfer function of an Nth order Butterworth low pass filter is

$$H(s) = \frac{\omega_c^N}{\prod_{n=1}^{N}(s - p_n)}, \tag{5.3}$$

where p_n is the nth pole and ω_c is the **half-power cutoff frequency**, i.e., the frequency at which the filter gain is $G_{\text{linear}} = 1/\sqrt{2}$ or $G_{\text{dB}} = -3dB$. The poles are located at

$$p_n = j\omega_c e^{\frac{j\pi}{2N}(2n-1)} \tag{5.4}$$

$$= -\omega_c \sin\left(\frac{\pi(2n-1)}{2N}\right) + j\omega_c \cos\left(\frac{\pi(2n-1)}{2N}\right), \tag{5.5}$$

for $n = \{1, 2, \ldots, N\}$. So, given a desired filter order N and cutoff frequency ω_c, we can use Eq. 5.5 to find the poles and then Eq. 5.3 to determine the transfer function. You would find that the poles form a semi-circle to the left of the imaginary s-axis.

In practice the magnitude of the Butterworth filter frequency response is more common. It is

$$|H(j\omega)| = \frac{1}{\sqrt{1 + \left(\frac{\omega}{\omega_c}\right)^{2N}}}. \tag{5.6}$$

To standardise design, we usually assume a normalised cutoff frequency $\omega_c = 1$ radian per second. The frequency response magnitude is then just

$$|H(j\omega)| = \frac{1}{\sqrt{1 + \omega^{2N}}}. \tag{5.7}$$

A sample of frequency response magnitudes for Butterworth filters with normalised frequency is shown in Fig. 5.5. We see that increasing the order improves the approximation of ideal behaviour.

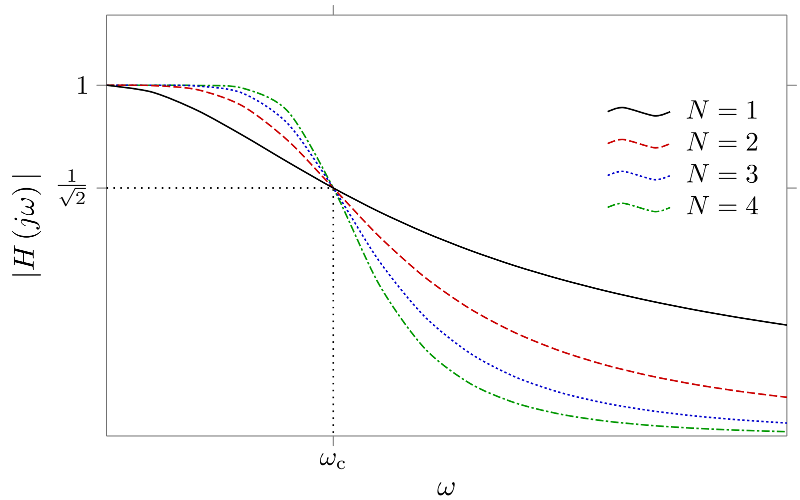


Figure 5.5: Frequency response magnitude of Butterworth low pass filter for different orders.

EXTRA 5.1: Converting Filter Designs

Our design of practical filters has focused on low pass filters and converting between different filter specifications. This is not an inherent limitation because we can also convert between different types of filters. For example, we can replace s with $1/s$ to convert a low pass filter transfer function into a corresponding high pass filter transfer function. More complex conversions exist for band pass and band stop filters. More details can be found in Section 2.6 of Lathi and Green.

For a given Butterworth filter specification, we have the following steps to find a corresponding filter that meets the specification:

1. Translating pass band and stop band requirements (via $G_p, \omega_p, G_s, \omega_s$) to a suitable order N.

2. Determine the cut-off frequency ω_c

3. Scaling the normalised frequency $\omega_c = 1$.

It can be shown that the minimum order N is the following:

$$N = \left\lceil \frac{\log\left(\frac{10^{-G_\mathrm{s}/10}-1}{10^{-G_\mathrm{p}/10}-1}\right)}{2\log\left(\frac{\omega_\mathrm{s}}{\omega_\mathrm{p}}\right)} \right\rceil, \tag{5.8}$$

where $\lceil \cdot \rceil$ mean that we have to round up (i.e., so that we over-satisfy and not under-satisfy the specification) and the gains are in dB. Given the order, we have two ways to determine the cut-off frequency ω_c:

$$\omega_\mathrm{c} = \frac{\omega_\mathrm{p}}{\left(10^{-G_\mathrm{p}/10}-1\right)^{\frac{1}{2N}}} \tag{5.9}$$

$$\omega_\mathrm{c} = \frac{\omega_\mathrm{s}}{\left(10^{-G_\mathrm{s}/10}-1\right)^{\frac{1}{2N}}}, \tag{5.10}$$

such that we meet the pass band specification exactly in the first case and we meet the stop band specification exactly in the first case. Due to component imperfections, we may choose a cut-off frequency that is in between these two cases.

To scale the normalised frequency in Eq. 5.7, we just replace ω with ω/ω_c.

EXTRA 5.2: Proof of Butterworth Filter Specification Equations

Let us demonstrate the validity of Eqs. 5.8, 5.9, and 5.10. We can write G_p (in dB) as a function of ω_p by converting to linear gain and using the definition of the Butterworth filter frequency response, i.e.,

$$\begin{aligned} G_\mathrm{p} &= 20\log_{10}\left|H\left(j\omega_\mathrm{p}\right)\right| \\ &= 20\log_{10}\left(\frac{1}{\sqrt{1+\left(\frac{\omega_\mathrm{p}}{\omega_\mathrm{c}}\right)^{2N}}}\right) \\ &= 10\log_{10}\left(1+\left(\frac{\omega_\mathrm{p}}{\omega_\mathrm{c}}\right)^{2N}\right) \end{aligned}$$

which we can re-arrange to solve for ω_c and arrive at Eq. 5.9. Similarly, we can write G_s (in dB) as a function of ω_s and re-arrange to solve for ω_c and arrive at Eq. 5.10. By setting Eqs. 5.9 and 5.10 equal to each other, we can re-arrange and arrive at Eq. 5.8. The ceiling function in Eq. 5.8 comes from the fact that we need an integer order and rounding down will make the filter unable to meet the specification.

Given Eqs. 5.8, 5.9, and 5.10 (which you would *not* be expected to memorise), we can design Butterworth filters to meet target specifications. We see this in the following example.

Example 5.2: Butterworth Filter Design

Design a Butterworth filter with a pass band gain no less than $-2\,\text{dB}$ over $0 \leq \omega < 10\,\frac{\text{rad}}{\text{s}}$, and a stop band gain no greater than $-20\,\text{dB}$ for $\omega > 20\,\frac{\text{rad}}{\text{s}}$.

From the specification, we have $G_\text{p} = -2\,\text{dB}$, $\omega_\text{p} = 10\,\frac{\text{rad}}{\text{s}}$ $G_\text{s} = -20\,\text{dB}$, and $\omega_\text{s} = 20\,\frac{\text{rad}}{\text{s}}$. From Eq. 5.8, we find that $N = 3.701$. We need an integer order, so we choose $N = 4$. We are then free to choose whether to exactly meet the pass band or stop band specification. If we choose to satisfy the pass band specification, then we solve Eq. 5.9 and get $\omega_\text{c} = 10.69\,\frac{\text{rad}}{\text{s}}$. If we choose to satisfy the stop band specification, then we solve Eq. 5.10 and get $\omega_\text{c} = 11.26\,\frac{\text{rad}}{\text{s}}$. We could leave a "buffer" on both bands and choose $10.69\,\frac{\text{rad}}{\text{s}} < \omega_\text{c} < 11.26\,\frac{\text{rad}}{\text{s}}$.

EXTRA 5.3: Other Filter Families

We could do a whole module on practical analogue filters, but we've just covered the Butterworth filter to give you some sense of practical filter design. Here we briefly summarise a few other common filter families to give you an idea of some of the trade-offs available. You can find more details in Lathi and Green.

The **Chebyshev** filter has "ripples" in the pass band (i.e., there are oscillations instead of being smooth like a Butterworth filter), but the transition band behaviour is steeper than the Butterworth filter. In practice, a lower order N is needed to meet a specification with Chebyshev filter than a Butterworth one.

While the pass band of a Chebyshev filter has ripples, its stop band is smooth. Since pass band behaviour is usually more important than the stop band, the **inverse Chebyshev** filter does the reverse and can be derived from a Chebyshev response.

Elliptic filters have ripples in both the pass band and stop band, in exchange for an even sharper transition band.

We haven't considered the phase of the frequency response in our discussion, but some filters are designed specifically for phase characteristics. **Bessel-Thomson** filters are designed for a maximally flat time delay over a specified frequency band, such that the phase response is close to linear over that band.

We are not covering time domain circuits implementations of practical analogue filters in this module, but you should be aware that they exist and commonly include op-amps. We will focus more on time-domain implementations when we consider digital filters.

5.5 Summary

- The common classifications of filters are **low pass**, **high pass**, **band pass**, and **band stop**. They are defined according to ideal transitions between **pass bands** and **stop bands**.
- Ideal filter magnitude responses are not realisable but we can obtain practical filter designs to approximate ideal responses.
- Butterworth filters can be readily designed to meet specified performance criteria.

5.6 Further Reading

- Chapter 2 of "Essentials of Digital Signal Processing," B.P. Lathi and R.A. Green, Cambridge University Press, 2014.

5.7 End-of-Lesson Problems

1. Explain in words why the ideal low pass filter frequency response is unrealisable.
2. If a low pass Butterworth filter is 12th order with a 3 dB cut-off frequency at 500 Hz, calculate the gain of the filter at 750 Hz.
3. Design the lowest order low pass Butterworth filter that meets the specification $\omega_\mathrm{p} \geq 10\,\frac{\mathrm{rad}}{\mathrm{s}}$, $G_\mathrm{p} \geq -2\,\mathrm{dB}$, $\omega_\mathrm{s} \leq 30\,\frac{\mathrm{rad}}{\mathrm{s}}$, $G_\mathrm{s} \leq -20\,\mathrm{dB}$. Find the order and feasible range of cut-off frequencies.

4. A subwoofer is a loudspeaker designed for low-pitch audio frequencies. Professional subwoofers typically produce sounds below 100 Hz. Design an audio filter for a speaker input that produces $G_\mathrm{p} \geq -1\,\mathrm{dB}$ for frequencies below 100 Hz and $G_\mathrm{p} \leq -30\,\mathrm{dB}$ for frequencies above 250 Hz. Find a suitable order and cut-off frequency (in Hz) for a Butterworth filter that meets these requirements.

Lesson 6

Periodic Analogue Functions

As we saw in Lesson 4, the Fourier transform is used to determine the spectra of signals. This is fine when we focus on the response of a system to a particular input frequency, but it isn't convenient when the input signal of interest has multiple components with different frequencies. We complete the theoretical material on analogue systems and signals with a brief look at the Fourier Series, which is useful for representing more complicated periodic signals.

6.1 Learning Outcomes

By the end of this lesson, you should be able to ...

1. **Understand** and **Apply** the Fourier Series to model complex periodic waveforms in the frequency domain.

2. **Understand** how periodic signals are affected by linear systems.

6.2 Representing Periodic Analogue Signals

There are multiple ways to represent periodic signals. We have been relying on Euler's formula to convert between an exponential representation and a trigonometric representation:

$$e^{jx} = \cos x + j \sin x \tag{6.1}$$

From Eq. 6.1, we can convert trigonometric functions into exponentials:

$$\cos x = \mathrm{Re}\left\{e^{jx}\right\} = \frac{e^{jx} + e^{-jx}}{2} \tag{6.2}$$

$$\sin x = \mathrm{Im}\left\{e^{jx}\right\} = \frac{e^{jx} - e^{-jx}}{2j} \tag{6.3}$$

A key point here is that a trigonometric function has an exponential form, and vice versa. Generally, the exponential form is more compact, and we have seen from Laplace and Fourier transforms that system responses to exponential signals are much easier to determine and manipulate than responses to trigonometric signals. However, there is a trade-off; complex exponential signals are more difficult to visualise. Thus, it is often more convenient to use exponential forms for mathematical manipulations and trigonometric forms for plots and intuition.

The exponential form of the **Fourier series** represents the period signal $x(t)$ as a sum of complex exponentials. There is an underlying fundamental frequency f_0, such that *all* frequencies contained in the signal are multiples of f_0. The corresponding fundamental period is $T_0 = 1/f_0$. The Fourier series is written as

$$x(t) = \sum_{k=-\infty}^{\infty} X_k e^{jk\omega_0 t}, \tag{6.4}$$

where $\omega_0 = 2\pi f_0 = 2\pi/T_0$ and the **Fourier coefficients** are

$$X_k = \frac{1}{T_0}\int_{T_0} x(t)\, e^{-jk\omega_0 t} dt, \tag{6.5}$$

which integrates the signal over its fundamental period.

One important property of the Fourier series is how it represents signals $x(t)$ that are real. A real $x(t)$ has an even magnitude spectrum and an odd phase spectrum, i.e.,

$$|X_k| = |X_{-k}| \quad \text{and} \quad \angle X_k = -\angle X_{-k} \tag{6.6}$$

Example 6.1: Fourier Series of Periodic Square Wave

A very important Fourier series is that for the periodic square wave. Consider that shown in Fig. 6.1, which is centred around $t = 0$, has amplitude A, and within a period remains high for τ seconds. We wish to write the function as a Fourier series and plots its magnitude. The period of the wave is by definition the fundamental T_0. We can then find the Fourier coefficients as follows:

$$\begin{aligned}X_k &= \frac{1}{T_0}\int_{T_0} x\left(t\right)e^{-jk\omega_0 t}dt\\ &= \frac{1}{T_0}\int_{-\frac{\tau}{2}}^{\frac{\tau}{2}} Ae^{-jk\omega_0 t}dt\\ &= \frac{A}{T_0}\left[\frac{-e^{-jk\omega_0 t}}{jk\omega_0}\right]_{-\frac{\tau}{2}}^{\frac{\tau}{2}}\\ &= \frac{A}{T_0}\left(\frac{e^{jk\omega_0\frac{\tau}{2}}-e^{-jk\omega_0\frac{\tau}{2}}}{jk\omega_0}\right)\\ &= \frac{2A}{T_0}\frac{\sin\left(k\omega_0\frac{\tau}{2}\right)}{k\omega_0}\frac{\frac{\tau}{2}}{\frac{\tau}{2}}\\ &= \frac{A\tau}{T_0}\operatorname{sinc}\left(k\omega_0\frac{\tau}{2}\right),\end{aligned}$$

which we note is an even function as expected for a real $x\left(t\right)$. In this particular case, the Fourier coefficients are also real (though generally they can be complex).

The signal $x\left(t\right)$ as a Fourier series is then

$$x\left(t\right) = \sum_{k=-\infty}^{\infty}\frac{A\tau}{T_0}\operatorname{sinc}\left(k\omega_0\frac{\tau}{2}\right)e^{jk\omega_0 t},$$

which we could also equivalently write by converting the sinc back to exponential form or converting the complex exponential to trigonometric form. More importantly, let us consider the magnitudes of the Fourier coefficients for different values of T_0 relative to τ, i.e., for different fundamental frequencies. For $T_0 = 2\tau$, we note that $\omega_0 = \pi/\tau$, and thus

$$X_k = \frac{A\tau}{2\tau}\operatorname{sinc}\left(k\frac{\pi}{\tau}\frac{\tau}{2}\right) = \frac{A}{2}\operatorname{sinc}\left(\frac{k\pi}{2}\right),$$

so we are sampling a sinc function at multiples of $\pi/2$. Similarly, for $T_0 = 5\tau$, we are sampling a (different) sinc function at multiples of $\pi/5$. The Fourier coefficients in both cases are plotted in Fig. 6.2.

It is important to emphasise that Fourier spectra only exist at the **harmonic frequencies**, i.e., at integer multiples of the fundamental frequency. The overall shape made the data points at these frequencies, i.e., the envelope, depends on the

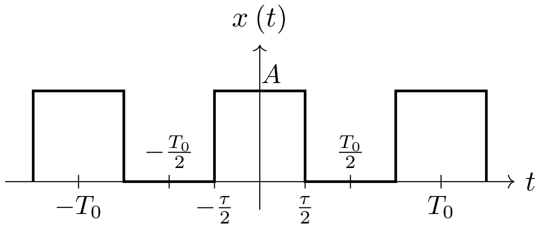
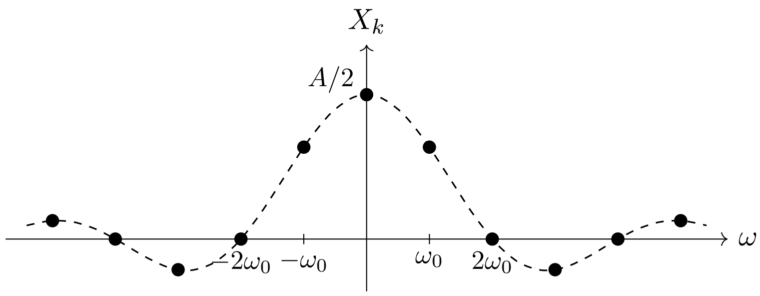
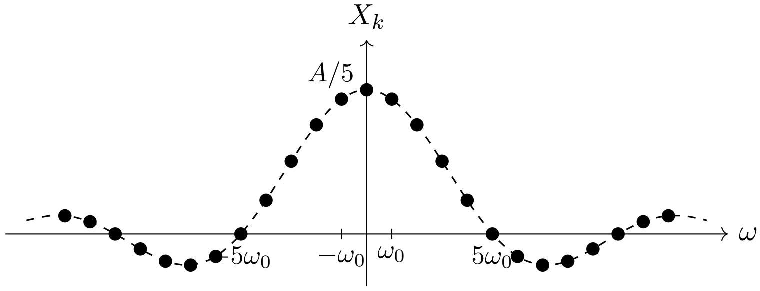


Figure 6.1: Periodic square wave in Example 6.1.



(a) $T_0 = 2\tau$.



(b) $T_0 = 5\tau$.

Figure 6.2: Fourier coefficients for periodic square wave in Example 6.1.

shape of the time-domain signal. It is as if we are sampling in the frequency domain, which results in a repeated signal in the time domain. In some sense, this is the opposite of what happens in analogue-to-digital conversion, where a time-domain signal is sampled and this results in repeated spectra. We will see this in more detail in the digital part of these notes.

6.3 Processing Periodic Signals in Linear Systems

We recall that for the Fourier transform we considered a single test sinusoid as the input to an LTI system. With the Fourier series, we have a (potentially infinite) sum of sinusoids for the system input. Fortunately, besides the increase in complexity, there is no fundamental difference in the analysis. Due to the superposition property of LTI systems, the system will produce an output for each corresponding input, based on the frequency response at the corresponding frequency. In other words, the system will change the amplitude and the phase of each frequency in the input. Thus, we can write

$$y(t) = \sum_{k=-\infty}^{\infty} H(jk\omega_0) X_k e^{jk\omega_0 t}, \tag{6.7}$$

or in other words

$$Y_k = H(jk\omega_0) X_k. \tag{6.8}$$

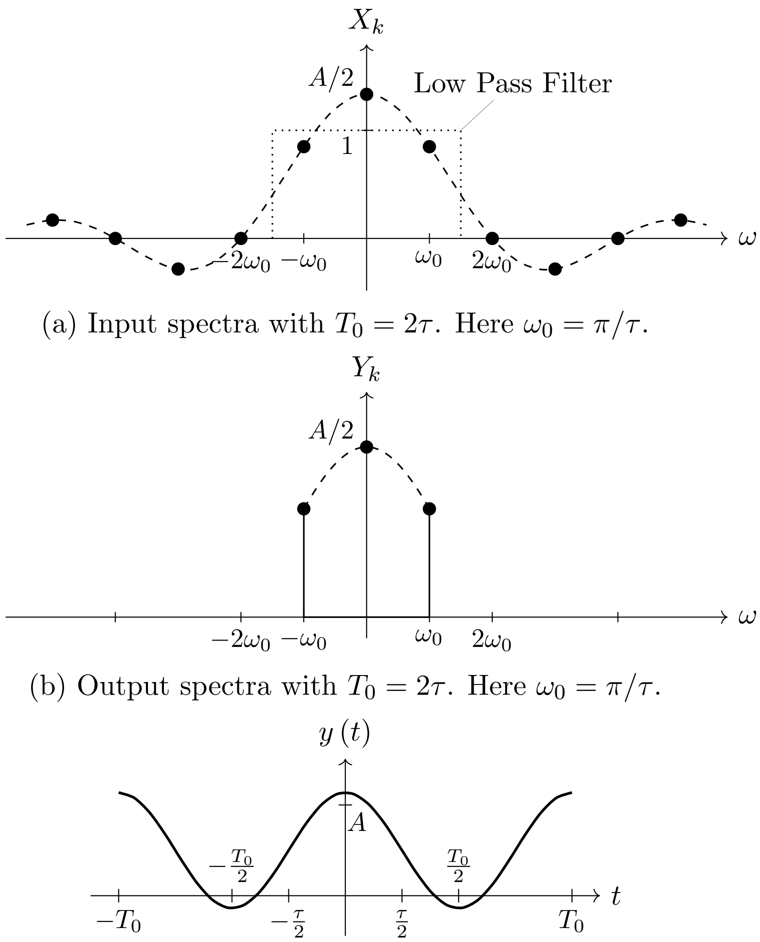
Example 6.2: Filtering a Periodic Signal

Consider passing the periodic square wave in Fig. 6.1, with $T_0 = 2\tau$, as the input into an ideal low pass filter with gain 1 for $0 \leq \omega < 1.5\frac{\pi}{\tau}$. What is the output of this filter in trigonometric form?

If $T_0 = 2\tau$, then the fundamental frequency is $\omega_0 = \pi/\tau$. The filter will attenuate any sinusoids with frequencies outside of $-1.5\pi/\tau < \omega_c < 1.5\pi/\tau$, so the only terms remaining in the summation are $k \in \{-1, 0, 1\}$. Therefore, the output is

$$\begin{aligned} y(t) &= \sum_{k=-1}^{1} H(jk\omega_0) X_k e^{jk\omega_0 t} \\ &= X_0 + X_{-1} e^{-jk\frac{\pi}{\tau}t} + X_1 e^{jk\frac{\pi}{\tau}t} \\ &= \frac{A}{2}\left[1 + \operatorname{sinc}\left(\frac{-\pi}{2}\right) e^{-\frac{j\pi t}{\tau}} + \operatorname{sinc}\left(\frac{\pi}{2}\right) e^{\frac{j\pi t}{\tau}}\right] \\ &= \left[1 + \frac{2}{\pi} e^{-\frac{j\pi t}{\tau}} + \frac{2}{\pi} e^{\frac{j\pi t}{\tau}}\right] \\ &= \frac{A}{2}\left[1 + \frac{4}{\pi}\cos\left(\frac{\pi t}{\tau}\right)\right]. \end{aligned}$$

We determined the output in trigonometric form because this is easier to plot (see Fig. 6.3). We also see that the output has 2 components: a constant plus a sinusoid with frequency π/τ. These match the frequencies of the input signal components that passed through the filter without being attenuated.



(a) Input spectra with $T_0 = 2\tau$. Here $\omega_0 = \pi/\tau$.

(b) Output spectra with $T_0 = 2\tau$. Here $\omega_0 = \pi/\tau$.

(c) Time-domain filter output $y(t)$.

Figure 6.3: Filter input and output in Example 6.2.

EXTRA 6.1: Fourier Series Properties

We can show that the Fourier series has many properties that are similar to those for the Fourier transform. However, since we will place a bigger emphasis on digital analysis, we will not elaborate on these here. You can find more discussion of these in Section 1.9 of Lathi and Green.

6.4 Summary

- The **Fourier Series** is used to model signals with multiple frequency components in the frequency domain.
- The output of an LTI system due to a signal with multiple frequency components can be found by superposition of the outputs due to the individual frequency components.

6.5 Further Reading

- Section 1.7 of "Essentials of Digital Signal Processing," B.P. Lathi and R.A. Green, Cambridge University Press, 2014.

6.6 End-of-Lesson Problems

1. Consider passing the periodic square wave in Fig. 6.1, with $T_0 = 5\tau$, as the input into an ideal band pass filter with gain 1 for $1.5\frac{\pi}{\tau} < \omega < 2.5\frac{\pi}{\tau}$ (and gain 0 elsewhere). How many sinusoidal terms will there be at the output? What will the frequencies of those terms be?
2. Consider passing the periodic square wave in Fig. 6.1, with $T_0 = 2\tau$, as the input into a low pass Butterworth filter of the 3rd order and cut-off frequency $\omega_c = 3/\tau$. What is the output amplitude of the complex exponential component of frequency π/τ?

Lesson 7

Computing with Analogue Signals

This lesson is the end of Part 1 of these notes. Our time to focus on purely analogue signals and systems is coming to a close. Here we shift our attention from primarily mathematical analysis to computing and applications. Analogue systems are by definition not digital systems, but we still find it useful to represent them on computers to help with design and analysis. In this lesson, we will see how MATLAB can be used as a computing tool for analogue systems. We should emphasise that MATLAB is not the only tool available for such purposes, as there are comparable features available in other platforms such as Mathematica, and also in programming languages. We use MATLAB since you may have used it in other modules and it has a relatively low learning curve for programming.

The computing examples are implementations and demonstrations of tasks that we've done "by hand" in previous lessons, so they should be helpful as a learning tool, e.g., to help verify your calculations and plots, and to investigate systems that are too complex to readily do so manually. Furthermore, the MATLAB functions used for digital systems are very similar (or sometimes *the same*) as those for analogue systems, so understanding the code here will offer some additional return later.

7.1 Learning Outcomes

By the end of this lesson, you should be able to ...

1. **Understand** how analogue signals and systems are represented in digital hardware.

2. **Analyse** analogue signals and systems with a computing package.

3. **Design** and **Implement** analogue filters with a computing package.

7.2 Analogue Signals in MATLAB

Computers are inherently digital systems. Information is stored in discrete containers such as arrays and matrices, and individual numbers are restricted to take on a finite number of values. Thus, they are ideal for implementing discrete-time digital signals and systems, and in fact they are why we have digital signal processing to begin with. So how can computers effectively model analogue systems?

While it is not possible to *precisely* implement an analogue system in a computer, we are able to store precise information about an analogue system via **symbolic math** (also known as **computer algebra**). In other words, we can store the *functions* that describe the analogue system, e.g., the impulse response, the transfer function, the input signal, and the output signal. For example, we have seen that we can write the Laplace-domain transfer function $H(s)$ of an LTI system as a ratio of polynomials. Therefore, to represent this system as a computer, we can use arrays to store the polynomial coefficients. This is the approach used in MATLAB.

Even though digitally-represented numbers have a finite number of values, the number of values is sufficiently large that granularity is not a problem for most practical systems. For example, a double-precision floating point number, which is the standard precision for most modern computing systems, uses 64 bits and has 15-17 decimal digits of precision over its range. This is *far more precision than what we usually ever need.* Thus, we can approximate double-precision numbers as being over a continuous range. When we want to know the behaviour of an analogue system at a particular time, we **sample** at the time of interest within the resolution of a double-precision number, and behaviour of the system at that time is calculated and returned within the resolution of a double-precision number.

EXTRA 7.1: Aside on Double-Precision Granularity

Strictly speaking, as we perform more and more calculations with double-precision numbers, we lose more and more accuracy as the precision errors accumulate. This is usually not an issue in practice, but it can be useful to keep in mind if you get unexpected behaviour from code where you expect two calculated values to be identical. For example, it is often helpful to round values to the nearest integer to mitigate such effects.

Let's now discuss how to create and study analogue systems in MATLAB. These is not a definitive discussion, as there is often more than one way to do something in MATLAB, but this will give you some ways to get started. Some particularly relevant toolboxes are the **Symbolic Math Toolbox** the **Signal Processing Tool-**

box, and the **Control Systems Toolbox**, although they can be used independently from each other.

7.2.1 Analogue Systems in the Symbolic Math Toolbox

With the Symbolic Math Toolbox, you can create symbolic variables. These are not variables in the traditional programming sense, but in the more abstract mathematical sense as they can be used to define functions and solve them analytically. Symbolic variables can be created using the `syms` function, and for an analogue system we usually find it convenient to create symbolic variables for time t and for s, i.e., enter

```
syms t s
```

We can now write expressions using s or t and they will be stored as symbolic functions that can be evaluated analytically. For example, we can create a time-domain input signal as a function of t and a Laplace-domain transfer function as a function of s. We can translate between the two domains using the `laplace` and `ilaplace` functions. Similar, we can use the `fourier` and `ifourier` functions for the Fourier and inverse Fourier transforms, respectively. There are a large number of pre-built functions available, such as `rectangularPulse`, `dirac`, `heaviside` (step function), `cos` (and other trigonometric functions as you might expect), and `sign`. You can also write your own functions.

Example 7.1: Finding Expression for System Output

Find the output of a simple low pass filter with transfer function

$$H(s) = \frac{1}{s+1},$$

when the input is a cosine $x(t) = \cos(\omega t)$. Using our knowledge of LTI systems, we can find the answer in a short function using symbolic math.

```
function y = lowpassOutput(omega)
% Find output to my low pass filter

% Create symbolic variables
syms t s
% Create symbolic functions for input and system
x = cos(omega*t);
H = 1/(s+1);
```

```
% Convert input to Laplace, solve, and convert output to time
X = laplace(x); Y = H*X; y = ilaplace(Y);
```

If you call this function with frequency $\omega = 5$, then the output will be `y = cos(5*t)/26 - exp(-t)/26 + (5*sin(5*t))/26`.

We can visualise a symbolic function using `fplot`, where an optional second argument specifies the range of coordinates of the domain, e.g.,

```
y = lowpassOutput(5);
figure; % New figure
fplot(y,[0,10])
```

This works for either the time or s-domain, but only for real values of s.

7.2.2 Analogue Systems in the Signal Processing Toolbox

The Signal Processing toolbox provides the `freqs` function for plotting frequency responses for analogue transfer functions (the "s" in `freqs` stands for s-domain). The first two input arguments for this function are two arrays, one describing the coefficients of the numerator polynomial and one describing the coefficients of the denominator polynomial. The standard naming convention for these arrays matches the form that we used in Lesson 3; a `b` vector lists the numerator coefficients in decreasing order and an `a` vector lists the denominator coefficients in decreasing order. For the system in Example 7.1, we have `b=1` and `a=[1,1]`. We can then plot the frequency response of this filter:

```
b = 1; a = [1, 1];
figure; % New figure
w = 0.1:0.1:10;
freqs(b,a,w)
```

The figure is shown in Fig. 7.1. The argument `w` is a vector of (radial) frequencies, but if you make it scalar instead then it defines the number of frequency points are determined automatically. Importantly, `freqs` plots the magnitude and phase of the frequency response on a logarithmic scale (for both the frequencies and for the magnitude). If you want to evenly space the data points along the domain of each plot, then you can use the `logspace` function to create these for you (e.g., `logspace(1,2,10)` will create 10 logarithmically spaced points between 10^1 and 10^2).

If you assign `freqs` to an output argument, then it will store the complex frequency response values for each of the frequencies. If you append a semi-colon `;`

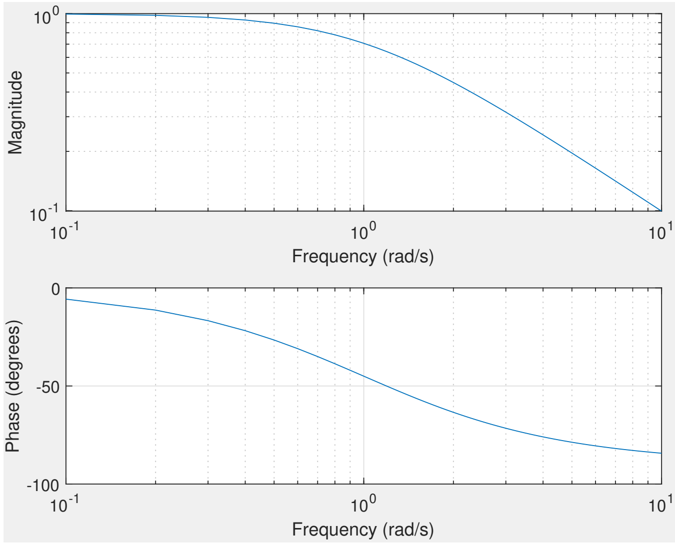


Figure 7.1: Frequency response of the system in Example 7.1.

after the function then it will omit the plot.

7.2.3 Analogue Systems in the Control Systems Toolbox

If you also have the Control Systems Toolbox then you can also have functions that can automatically generate pole-zero plots from a transfer function. With this toolbox, transfer functions are defined using the `tf` function, where the input is of the form `sys = tf(b,a)`, where `b` and `a` are once again vectors that describe the descending coefficients of the numerator and denominator polynomials. You can then create a pole-zero plot using the `pzplot` function:

```
b = 1; a = [1, 1];
figure; % New figure
H = tf(b,a);
pzplot(H)
```

The figure is shown in Fig. 7.2. For this particular system the result is rather boring as there is a single pole and no zeros.

7.3 Analogue Filter Design

A particular strength of the Signal Processing Toolbox is that it has practical filter families included, even for analogue systems. In fact, all of the filter families described in Lesson 5 are available, i.e., Butterworth, Chebyshev, elliptic, Bessel, and

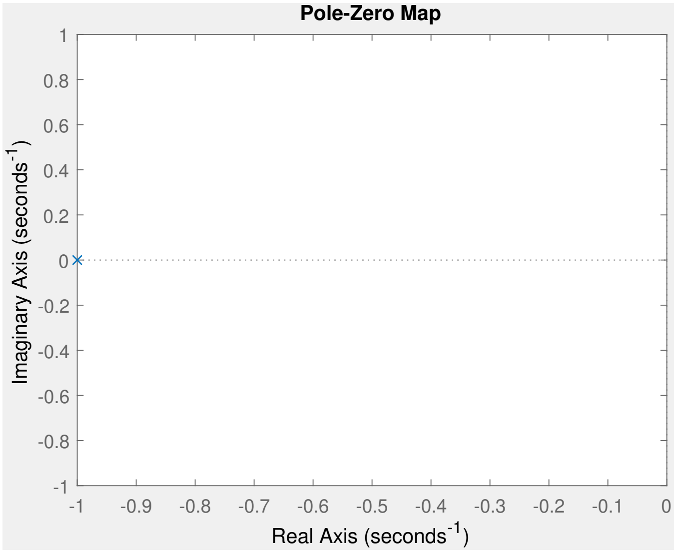


Figure 7.2: Frequency response of the system in Example 7.1.

more. The corresponding function names are somewhat intuitive but we list them here:

- Butterworth: `butter`
- Chebyshev: `cheby1` (i.e., type 1)
- Inverse Chebyshev: `cheby2` (i.e., type 2)
- Elliptic: `ellip`
- Bessel (i.e., Bessel-Thomson): `besself`

The syntax and use of these functions are consistent, so we will focus on using the `butter` function. The default input arguments are the order of the filter and the cut-off frequency. By default the filter will be low pass, but a third optional argument can indicate whether the filter is to be a different type, i.e., one of `'low'`, `'high'`, `'bandpass'`, or `'stop'`. To actually make this an *analogue* filter instead of a digital one is to make the final argument `'s'`. For the output, we will consider one of 2 formats. If you define two input arguments then they will be of the form `[b,a]`, i.e., the coefficients of the transfer function's numerator and denominator polynomials, respectively. If you define three input arguments then they will be of the form `[z,p,k]`, i.e., you will get the poles, zeros, and the transfer function gain K. From our knowledge of transfer functions, either of these two forms are sufficient to fully define the transfer function. You can also convert between the two forms using the `zp2tf` or `tf2zp` functions.

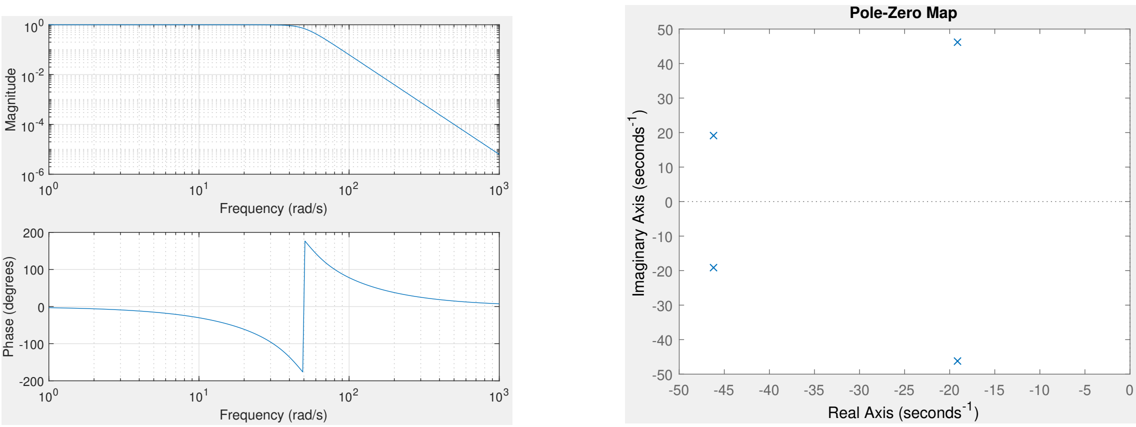
Example 7.2: Analgoue Butterworth Filter in MATLAB

Design a 4th order analogue low pass Butterworth filter with cutoff frequency $\omega_c = 50\,\frac{\text{rad}}{\text{s}}$. What are its poles and zeros? Plot its frequency response and a pole-zero plot.

We can complete all of these tasks directly in a script.

```
[z,p,k] = butter(4,50,'low','s') % Omitting semi-colon will
    print output
[b,a] = zp2tf(z,p,k); % Convert to [b,a] form for freqs
figure; freqs(b,a) % Plot frequency response
H = tf(b,a);
figure; pzplot(H) % Pole-zero plot
```

The output states that there are no zeros and 4 poles. The generated plots are shown in Fig. 7.3.



(a) Frequency response. (b) Pole-zero plot.

Figure 7.3: Output plots of script in Example 7.2.

7.4 Summary

- We can analytically represent analogue systems in computing platforms, either formally using **symbolic math** or indirectly with vectors that represent system components.

- MATLAB has toolboxes with functions that enable us to implement most of the tasks that we have completed mathematically and "by hand" throughout these notes.

7.5 Further Reading

- The MATLAB documentation is an excellent resource, whether online or within the MATLAB help browser.
- Chapter 17 of "Essential MATLAB for engineers and scientists," B. Hahn and D. Valentine, Academic Press, 7th Edition, 2019.

7.6 End-of-Lesson Problems

1. Find the output of a filter with transfer function

$$H(s) = \frac{R_2}{s^2 LCR_1 + s(L + CR_1R_2) + R_1 + R_2},$$

where $R_1 = R_2 = 2\,\Omega$, $C = 1\,\mathrm{F}$, and $2\,\mathrm{H}$. The input is the sinusoid $x(t) = \sin(10t)$. Do *not* try to solve this by hand!

2. The `butter` function requires the target order N and cut-off frequency ω_c. Write a function that will calculate these from the more general low pass filter specifications (i.e., from the pass band and stop band gains and frequencies).