EEET 3046 Control Systems (2020)

Lecture 2: System dynamics and modeling

Dr. Yee Wei Law ⟨yeewei.law@unisa.edu.au⟩

Contents

1 Introduction 1

2 Differential equations 2
- 2.1 Examples 3

3 Laplace transform 6
- 3.1 Important properties 9
- 3.2 Transfer function 13
- 3.3 Introduction to discrete-time modeling and the z-transform 16

4 State-space equations 19
- 4.1 Realization* 23

5 An elaborate example 25

6 Summary 27

1 Introduction

As mentioned in Lecture 1, model-based control techniques[1] require a model of the process to be controlled, and this model comes in the form of a set of *differential equations*. The act of deriving these differential equations is part of *system modeling*, which is typically the first step of any control engineering process. System modeling requires an in-depth understanding of the physics of the system; it is hence typically the most time consuming, if not the most challenging part of the process.

In Lecture 1, we have seen how control can be and have been applied to a wide range of systems, whether they are electrical, mechanical, biological, or any mixture of these. It is impossible for the course to give you all the training you need to model all kinds of systems. For this course, we will learn and practise modeling

- **RLC circuits** (electrical): RLC circuit models are widely used in the analysis and design of analog filters, electrical machine drives, power electronics, and power systems.
- **Rigid-body systems** (mechanical): Among the most important rigid-body systems are the *mass-spring-damper* systems, because they can be used to model a wide range of mechanical systems, such as car suspension (see Supplementary Lecture A), harmonic drives (see later), aircrafts, ships, and buildings.

Modeling systems with differential equations is not enough. In order to design a controller for a system, we need to be able to solve the differential equations, because the solution will give us the

[1] For a whizbang presentation on how model-based design works, see `https://au.mathworks.com/solutions/model-based-design.html` or `https://www.youtube.com/user/3dsCATIA`.

output for any given input. The output expressed as a function of time is called the *output response*, to emphasize the fact that the output is the system's response to the input stimulus. For controller design, there are two standard solution approaches: either solve the equations in the Laplace domain using **Laplace transform** (Sect. 3), or the time domain using **state-space equations** (Sect. 4). While Laplace transformation is a core part of classical control, the coverage of state-space equations here serves only as an initiation to modern control.

This lecture follows the directions of [Nis15, Chs. 2 and 3], with supplements.

2 Differential equations

In the most general context, the term *dynamical system* refers to a system that evolves with time, but it is typically used to refer to a system that can be described using differential equation(s). Differential equations can be ordinary or partial.

- **Ordinary differential equations** (ODEs): In these equations, the only independent variable is time. Systems describable using ODEs alone are called **lumped-parameter systems**, since dependent variables are not "spatially distributed". We have seen examples of ODEs since high school:
  - In electrical circuits, Faraday's law of inductance: $v = L\dfrac{\mathrm{d}i}{\mathrm{d}t}$.
  - In mechanics, Newton's second law: $F = m\dfrac{\mathrm{d}^2x}{\mathrm{d}t^2}$.
  - In physics and chemistry, law of radioactive decay: $\dfrac{\mathrm{d}N}{\mathrm{d}t} = -kN$, where N is the number of atoms.
  - In biology, the logistic equation: $\dfrac{\mathrm{d}A}{\mathrm{d}t} = cA(P - A)$, where A is the number of infected people, and P is the total population.

  When the ODEs are written in the state-space form, the state vector has a finite dimension (i.e., finite number of elements), so lumped-parameter systems are also **finite-dimensional systems**.

  Detail: Notation

  There are three main types of notations for derivatives. For the first-, second-, and nth-order derivatives, we have
  - Leibniz's notation: $\dfrac{\mathrm{d}x}{\mathrm{d}t}, \dfrac{\mathrm{d}^2x}{\mathrm{d}t^2}, \ldots, \dfrac{\mathrm{d}^nx}{\mathrm{d}t^n}$.
  - Newton's notation: $\dot{x}, \ddot{x}, \dddot{x}, \ldots, x^{(n)}$.
  - Lagrange's notation: $x', x'', x''', \ldots, x^{(n)}$.

  In the control literature, Newton's notation is the mainstream notation.

- **Partial differential equations** (PDEs): In these equations, time is not the only independent variable; some variables may depend on space or some other variable, in addition to time. Systems modeled with PDEs are **distributed-parameter systems**. For example, in a diffusion process where heat diffuses from one end of a narrow rectangular strip to another end of the strip (see Figure 1) can be modeled with the PDE:

$$C\rho\frac{\partial z(x,t)}{\partial t} = K\frac{\partial^2 z(x,t)}{\partial x^2}, \tag{1}$$

where $z(x,t)$ is the temperature of the strip at location x and time t; C, ρ, K are constants [Lev11, Example 67.1]. Systems with flexible bodies are another example of distributed-parameter systems. PDEs cannot be written in the finite-dimensional state space form, so distributed-parameter systems are also **infinite-dimensional systems**. Whenever you see such keywords as "flexible structures" (as opposed to "rigid structures"), "vibration", "noise" and "flow", you can immediately think of PDEs / distributed-parameters systems / infinite-dimensional systems.

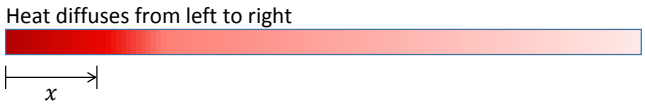


Figure 1: Diffusion of heat from the left end to the right end of a narrow rectangular strip. x marks the location from the left.

For this course, we only deal with ODEs, because lumped-parameter systems are common, and distributed-parameter systems require more advanced maths. Furthermore, we only deal with *linear* ODEs of the form

$$a_n y^{(n)}(t) + a_{n-1} y^{(n-1)}(t) + \cdots + a_0 y(t) = b_m x^{(m)}(t) + \cdots + b_0 x(t), \tag{2}$$

where $x(t)$ and $y(t)$ represent the input and output respectively; the coefficients $a_0, \ldots, a_n, b_0, \ldots, b_m$ are time-independent constants. Eq. (2) describes a linear time-invariant (LTI) systems.

If you have taken EEET 3041 Signals and Systems, then you should already be familiar with linear ODEs. Otherwise, linear ODEs are covered in the EEET 5148 Control Systems M workshops. In any case, in-depth knowledge of linear ODEs is not necessary.

2.1 Examples

Let us look at some sample systems describable using linear ODEs. We will look at an electrical system (RC circuit) in Example 1, a mechanical system (mass-spring-damper) in Example 2, and an electromechanical system (DC motor) in Example 3. Additionally, we will see how a rotational mass-spring-damper (governed by the rotational version of Newton's second law) can be used to model a specific kind of gear in Example 4.

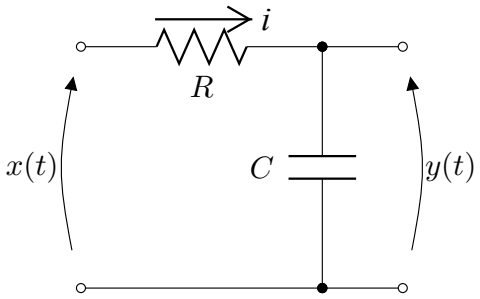


Figure 2: RC circuit for Example 1.

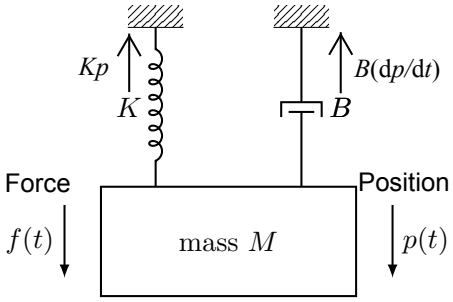


Figure 3: Mass-spring-damper system for Example 2.

Example 1

Consider the RC circuit in Figure 2, with input voltage $x(t)$ and output voltage $y(t)$. For those not familiar with capacitors, the current flowing the capacitor C is

$$i(t) = C\frac{\mathrm{d}y(t)}{\mathrm{d}t}.$$

By Kirchoff's voltage law and Ohm's law, we have the system model:

$$x = iR + y = RC\,\frac{\mathrm{d}y}{\mathrm{d}t} + y = RC\dot{y} + y. \tag{3}$$

Example 2

Consider the mass-spring-damper system in Figure 3, with input force $f(t)$ and output displacement/position $p(t)$. For those not familiar with springs and viscous dampers:

- When a spring is stretched/compressed, it resists the stretching/compression with a reaction force of $Kp(t)$, as per Hooke's law, where K is the spring constant, $p(t)$ is the amount of stretching/compression.
- When a damper is stretched/compressed, is resists the stretching/compression with a reaction force of $B\dot{p}(t)$, where B is the coefficient of viscous damping, $\dot{p}(t)$ is the speed of stretching/compression.

The net force on mass M is thus

$$f(t) - Kp(t) - B\dot{p}(t). \tag{4}$$

Note that there are no gravity terms in the equation above, because $p(t)$ is measured from the point where only gravity acts on the system. This is as explained in detail at `http://lpsa.swarthmore.edu/Systems/MechTranslating/TransMechSysModel.html`.

By Newton's second law, we have the system model:

$$M\ddot{p} = f - Kp - B\dot{p}. \tag{5}$$

In the future, we can write down Eq. (5) simply by inspection:

$$M\ddot{p} + B\dot{p} + Kp = f, \tag{6}$$

as long as positive displacement is in the same direction as the external force.

Example 3

Consider the DC motor in Figure 4 with input voltage $v(t)$ and output motor speed $\dot{\theta}(t)$. Let

- $T(t)$ be the torque,
- K_t be the torque constant,

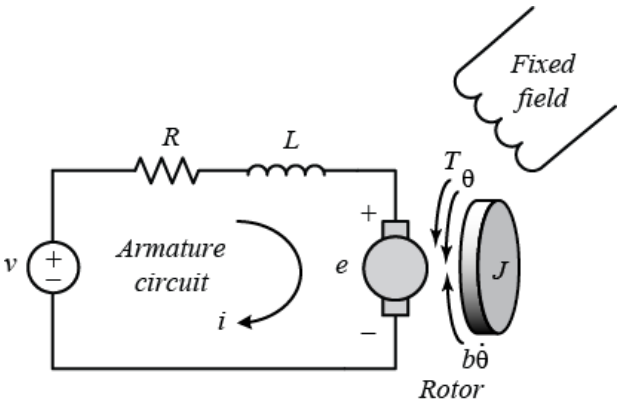


Figure 4: A simplified model of a DC motor for Example 3. Source: `http://ctms.engin.umich.edu/CTMS/index.php?example=MotorSpeed&section=SystemModeling`.

- $i(t)$ be the armature current.
- J be the moment of inertia of the rotor,
- $\theta(t)$ be the rotor angle,
- b be the coefficient of viscous friction,
- K_b be the back emf constant,
- R be the armature resistance,
- L be the armature inductance.

The torque is proportional to the armature current:

$$T = K_t i. \tag{7}$$

Using this fact and Newton's second law, we have

$$J\ddot{\theta} = T - b\dot{\theta} = K_t i - b\dot{\theta}. \tag{8}$$

The back electromotive force (emf) is proportional to the rotor speed:

$$e = K_b \dot{\theta}. \tag{9}$$

Using this fact and Kirchhoff's voltage law, we have

$$v = Ri + L\frac{\mathrm{d}i}{\mathrm{d}t} + e = Ri + L\frac{\mathrm{d}i}{\mathrm{d}t} + K_b\dot{\theta}. \tag{10}$$

Now we have a system model comprising Eqs. (8) and (10) relating input $v(t)$ to output $\dot{\theta}(t)$.

Example 4

Consider the *harmonic drive* in Figure 5, which is a gear mechanism[a] designed to provide low backlash[b], high torque transmission in a compact assembly [SHV06, Sect. 6.5]. When the wave generator rotates under the action of a motor, it causes the some of the teeth of the flexispline and the circular spline to mesh. The way the teeth mesh has the advantage of producing little backlash. The YouTube video at `https://youtu.be/bzRh672peNk` shows a harmonic drive in action. Harmonic drives are useful for robotic applications, and in general applications that

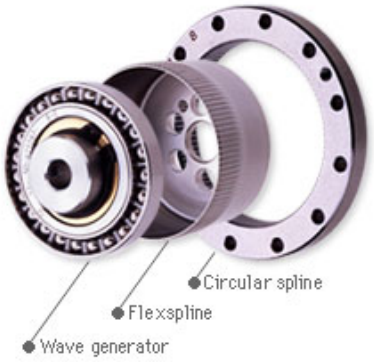


Figure 5: A harmonic drive consists of a circular spline, a flexispline, and an elliptical wave generator. The flexispline takes on the shape of the wave generator when the latter is inserted into it. Image source: `https://www.hds.co.jp`.

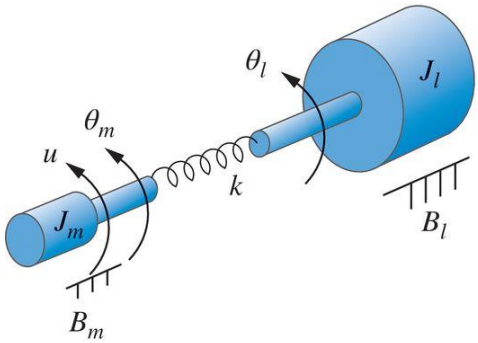


Figure 6: An idealized model of a harmonic drive, where the flexipline is modeled as a torsional spring of stiffness k. Subscripts m and l refer to the motor and the load respectively. Image source: [Nis15, FIGURE P8.18].

require low backlash and high precision.

To model a harmonic drive, we can represent its flexispline as a torsional spring of stiffness k (see Figure 6). J_m, B_m and θ_m characterize the motor, representing its the moment of inertia, damping coefficient, and angle respectively. J_l, B_l and θ_l characterize the load, with apparent meaning. Suppose we take the motor torque $T(t)$ as input and the load angle $\theta_l(t)$ as output.

Applying Newton's second law for rotation to the motor side, we have

$$J_m\ddot{\theta}_m + B_m\dot{\theta}_m + k(\theta_m - \theta_l) = T. \tag{11}$$

Note the amount of torsional stretching/compression is $\theta_m - \theta_l$ rather than θ_m.

Applying Newton's second law for rotation to the load side, we have

$$J_l\ddot{\theta}_l + B_l\dot{\theta}_l + k(\theta_l - \theta_m) = 0. \tag{12}$$

Now we have a system model comprising Eqs. (11) and (12) relating input $T(t)$ to output $\theta_l(t)$.

[a] One may wonder why it is not called a "harmonic gear" instead.

[b] Loss of motion between inter-meshing machine parts.

3 Laplace transform

This section shows how to solve differential equations by *Laplace transform*, which essentially converts linear ODEs in the time domain into simple algebraic equations in the Laplace domain. Additionally, this section introduces the concept of *transfer function*. A transfer function relates the system output to the system input in the Laplace domain. Transfer functions are not necessary for solving differential equations, but they make modeling so much easier, by allowing us to represent a large system using a network of transfer functions.

Definition: Laplace transform

The Laplace transform of function $f(t)$, denoted $F(s)$, is

$$F(s) = \mathcal{L}\{f(t)\} \stackrel{\text{def}}{=} \int_0^\infty e^{-st} f(t)\,\mathrm{d}t, \tag{13}$$

where $\mathcal{L}$ is the Laplace transform operator, and s is a complex variable known as the Laplace variable.

Detail: Unilateral vs. bilateral Laplace transform

The standard definition of the Laplace transform is "unilateral" (integration from 0 to ∞). The bilateral/two-sided definition of the Laplace transform, namely $\mathcal{L}_{\text{bilateral}}\{f(t)\} = \int_{-\infty}^{\infty} e^{-st} f(t)\,\mathrm{d}t$, is not as commonly used. See `http://mathworld.wolfram.com/BilateralLaplaceTransform.html`.

The Laplace transform is an integral. Clearly if $f(t)$ tends to infinity over time, the integral becomes undefined, so a Laplace transform only exists provided the conditions in the following theorem are satisfied:

Theorem 1: [BB11, Theorem 5.1.6]

If function f is piecewise continuous on the interval $0 \leq t \leq T$ for any positive T, and f is of **exponential order** a, then the Laplace transform $\mathcal{L}\{f(t)\}$ exists for $s > a$.

Quiz 1

Can you sketch an example of a piecewise continuous function?

Definition: Exponential order [BB11, Definition 5.1.5]

A function $f(t)$ is of **exponential order** a if there exists real constants $T \geq 0$, $K > 0$, and a, such that when $t \geq T$,

$$|f(t)| \leq Ke^{at}.$$

In other words, a function of exponential order a is guaranteed to grow *no faster* than some exponential function Ke^{at}. For a function of exponential order a, the magnitude of the integrand in Eq. (13) is bounded for $s > a$, since

$$|e^{-st} f(t)| \leq |e^{-st}||f(t)| = e^{-st}|f(t)| \leq Ke^{-(s-a)t}.$$

Note when $s > a$, $e^{-(s-a)t}$ is a decaying exponential, and $|e^{-st} f(t)|$ is guaranteed to decay with t. Therefore, the Laplace transform of $f(t)$ exists for $s > a$. The values of s for which the Laplace transform exists constitute the **region of convergence**.

Example 5

$$\mathcal{L}\{1\} = \int_0^\infty e^{-st}\,\mathrm{d}t = -\lim_{t\to\infty}\frac{e^{-st}}{s} + \frac{1}{s}.$$

For the above to be finite, we need $\Re\{s\} > 0$, hence the region of convergence is $\Re\{s\} > 0$.

Some popular Laplace transforms are provided in Table 1 on p. 29. Let us see how we can use this table to calculate Laplace transforms.

Example 6

Use Table 1 to determine the Laplace transform of $2t\sin(t)$.

Solution: From the table, we know $\mathcal{L}\{tf(t)\} = -F'(s)$. Clearly, we can define $f(t) = 2\sin(t)$, then based on the table

$$F(s) = \mathcal{L}\{2\sin(t)\} = \frac{2}{s^2+1} \implies F'(s) = \frac{\mathrm{d}}{\mathrm{d}s}\left(\frac{2}{s^2+1}\right) = -\frac{4s}{(s^2+1)^2}.$$

Therefore,

$$\mathcal{L}\{t\cdot 2\sin(t)\} = -\left(-\frac{4s}{(s^2+1)^2}\right) = \frac{4s}{(s^2+1)^2}. \tag{14}$$

We can verify Eq. (14) with just a web browser:

1. Visit website `www.wolframalpha.com`.
2. Type in "`Laplace 2*t*sin(t)`" in loose language, or "`LaplaceTransform[2*t*Sin[t], t, s]`" in proper Mathematica syntax. Note that both Wolfram Alpha and Mathematica are made by the same company, Wolfram.
3. Read the result.

Note that there is ABSOLUTELY NO expectation for the students to use Wolfram Alpha or Mathematica in this course.

Alternatively, we can verify Eq. (14) using the following MATLAB code:

```
syms t s
laplace(2*t*sin(t), t, s)
```

Above, the keyword `syms` declares symbolic (as opposed to numeric) variables or functions. We will be using more functions from MATLAB's Symbolic Math Toolbox, like `syms`, as we go along.

The inverse Laplace transform of $F(s)$ is the contour integral called the **Bromwich inversion integral**:

$$\mathcal{L}^{-1}[F(s)] = \frac{1}{2\pi j}\int_{\sigma-j\infty}^{\sigma+j\infty} e^{st}F(s)ds = f(t)u(t), \tag{15}$$

where $u(t)$ is the Heaviside/unit step function ($u(t) = 1$ for $t > 0$, $u(t) = 0$ for $t < 0$). For most purposes though, it is easier to apply Laplace transform formulas in reverse than to calculate the contour integral directly.

3.1 Important properties

The Laplace transform has several important properties that make it extremely useful for system modeling and controller design. First and foremost, the Laplace transform is a linear operator, because

$$\mathcal{L}\{af(x) + bg(x)\} = \int_0^\infty e^{-st}(af(t) + bg(t))\,\mathrm{d}t = a\,\mathcal{L}\{f(x)\} + b\,\mathcal{L}\{g(x)\}. \tag{16}$$

We can similarly show the inverse Laplace transform is also a linear operator.

The key property of the Laplace transform that enables the conversion of differential equations into algebraic equations is this:

Theorem 2: [BB11, Corollary 5.2.3]

Given continuous function f, if
- $f', \cdots, f^{(n-1)}$ are continuous and $f^{(n)}$ is piecewise continuous on any interval $0 \le t \le T$,
- $f', f'', \cdots, f^{(n-1)}, f^{(n)}$ are of exponential order a,

then $\mathcal{L}\{f^{(n)}(t)\}$ exists for $s > a$, and is given by

$$\mathcal{L}\{f^{(n)}(t)\} = s^n\,\mathcal{L}\{f(t)\} - s^{n-1}f(0) - \cdots - f^{(n-1)}(0). \tag{17}$$

Example 7

Let us see how we can use Theorem 2 to determine the output response of the RC circuit in Example 1. To both sides of Eq. (3):

$$x = RC\dot{y} + y,$$

applying Laplace transformation and Theorem 2, we get

$$X(s) = RCsY(s) - RCy(0) + Y(s) \implies Y(s) = \frac{1}{RCs+1}(X(s) + RCy(0)). \tag{18}$$

If $x(t)$ is a sinusoid, e.g., $x(t) = \sin(\omega t)$, then $X(s) = \dfrac{\omega}{s^2+\omega^2}$, and

$$\begin{aligned} Y(s) = \frac{1}{RCs+1}\left(\frac{\omega}{s^2+\omega^2} + RCy(0)\right) &= \frac{\omega}{(RCs+1)(s^2+\omega^2)} + \frac{y(0)}{s+1/(RC)} \\ &= \frac{\omega\omega_c}{(s+\omega_c)(s^2+\omega^2)} + \frac{y(0)}{s+\omega_c}, \end{aligned} \tag{19}$$

where we have defined $\omega_c \overset{\text{def}}{=} 1/(RC)$. At this point, we can perform inverse Laplace transform on Eq. (19) to obtain $y(t)$, but this term

$$\frac{\omega\omega_c}{(s+\omega_c)(s^2+\omega^2)}$$

is a third-order fraction (since the highest-order term is s^3), whereas the Laplace transform table only has entries up to the second order. To solve this problem, we need to apply **partial**

fraction expansion, i.e., a way of expressing a high-order fraction as a sum of first-order and second-order fractions. In this case, we would like to find constants A, B and C such that

$$\frac{\omega\omega_c}{(s+\omega_c)(s^2+\omega^2)} = \frac{A}{s+\omega_c} + \frac{Bs+C}{s^2+\omega^2}. \tag{20}$$

The numerators on both sides must be equal, i.e.,

$$\omega\omega_c = A(s^2+\omega^2) + Bs(s+\omega_c) + C(s+\omega_c). \tag{21}$$

Equating the coefficients of s terms on both sides, we get

$$\begin{aligned} A + B = 0 &\implies A = -B, \\ B\omega_c + C = 0 &\implies C = A\omega_c, \\ A\omega^2 + C\omega_c = \omega\omega_c &\implies A\omega^2 + A\omega_c^2 = \omega\omega_c. \end{aligned}$$

Therefore,

$$A = \frac{\omega\omega_c}{\omega^2+\omega_c^2}, \qquad B = -\frac{\omega\omega_c}{\omega^2+\omega_c^2}, \qquad C = \frac{\omega\omega_c^2}{\omega^2+\omega_c^2}. \tag{22}$$

Substituting Eqs. (22) into Eq. (20), and substituting Eq. (20) into Eq. (19), we get

$$Y(s) = \frac{\omega\omega_c}{(\omega^2+\omega_c^2)(s+\omega_c)} - \frac{\omega\omega_c s}{(\omega^2+\omega_c^2)(s^2+\omega^2)} + \frac{\omega\omega_c^2}{(\omega^2+\omega_c^2)(s^2+\omega^2)} + \frac{y(0)}{s+\omega_c}. \tag{23}$$

Taking inverse Laplace transform of the above, we get the output response

$$y(t) = \frac{\omega\omega_c}{\omega^2+\omega_c^2}\left[e^{-\omega_c t} - \cos(\omega t) + \frac{\omega_c}{\omega}\sin(\omega t)\right] + y(0)e^{-\omega_c t}. \tag{24}$$

We can readily verify Eq. (24) using the MATLAB code below:

```
syms y(t) x(t) t omega_c omega y_0
x = sin(omega*t);
dsolve(diff(y,t)==-omega_c*y+omega_c*x,y(0)==y_0)
```

Since ω_c is positive, the exponentials in Eq. (24) decay with time, so in steady state, $y(t)$ is a sinusoid, just as $x(t)$ is a sinusoid.

- The last term $y(0)e^{-\omega_c t}$ is the only term that depends on the initial state, and represents the *free response* or *natural response* of the system. The free response is also called the *zero-input response*, since it equals the output response at zero input (i.e., $x(t) = 0$).
- The other terms represent the *forced response* of the system. The forced response is also called the *zero-state response* since it equals the output response at zero initial state (i.e., $y(0) = 0$).

Note:

- Partial fraction expansion, as the preceding example suggests, is often needed during manual analysis. An e-reading on partial fraction expansion [Nis11, pp. 37–44] is available on learnonline for those who want to learn more than the technique demonstrated in Example 7.

- In classical control, we typically assume zero initial conditions, i.e., $f(0) = 0, \cdots, f^{(n-1)}(0) = 0$, in which case we end up with $\mathcal{L}\{f^{(n)}(t)\} = s^n \mathcal{L}\{f(t)\}$. Therefore, remember this: at zero initial conditions, differentiation in the time domain is equivalent to multiplication with s in the Laplace domain.

Symmetrical to Theorem 2, we have the following theorem which will become useful in Lecture 4.

Theorem 3: [BB11, Theorem 5.2.4]

Given continuous function f, if
- f is piecewise continuous on any interval $0 \leq t \leq T$,
- f is of exponential order a,

then for any positive integer n and $s > a$,

$$\mathcal{L}\left\{t^n f(t)\right\} = (-1)^n F^{(n)}(s). \tag{25}$$

The next key property has to do with time shift (Second Shifting Theorem) and frequency shift (First Shifting Theorem):

Theorem 4: First Shifting Theorem / First Translation Theorem

If $F(s) = \mathcal{L}\{f(t)\}$ and $a \in \mathbb{R}$, then $\mathcal{L}\{e^{-at} f(t)\} = F(s + a)$.

Example 8

The First Shifting Theorem is especially useful for inverse-transforming denominators with complex roots, for example,

$$\mathcal{L}^{-1}\left\{\frac{1}{s^2 + 2s + 5}\right\} = \mathcal{L}^{-1}\left\{\frac{1}{(s+1)^2 + 2^2}\right\} = e^{-t}\,\mathcal{L}^{-1}\left\{\frac{1}{s^2 + 2^2}\right\} = \frac{1}{2}e^{-t}\sin(2t).$$

Theorem 5: Second Shifting Theorem / Second Translation Theorem

If $a > 0$ and $F(s) = \mathcal{L}\{f(t)\}$, then $\mathcal{L}\{f(t-a)u(t-a)\} = e^{-as}F(s)$.

The Second Shifting Theorem provides a way to model systems with *delay*. This delay, also called *transport lag*, *transport delay* or *dead time*, is the time between when an input is applied, and when the system starts responding to the input.

The next key property is *convolution*, which enables the block diagram approach to control system design.

Definition: Convolution [HS14, Definition 4.38]

Suppose $f(t)$ and $g(t)$ are piecewise continuous functions. The convolution $f * g$ of the functions f and g is

$$f * g \stackrel{\text{def}}{=} \int_0^t f(\tau)g(t-\tau)\,\mathrm{d}\tau. \tag{26}$$

The physical interpretation of convolution can be understood by approximating the integral in Eq. (26) with a sum:

$$f * g \approx f[0]g[t]\Delta t + f[\Delta t]g[t-\Delta t]\Delta t + \cdots + f[t]g[0]\Delta t.$$

Think of $f[0], f[\Delta t], \cdots, f[t-\Delta t], f[t]$ as input impulses at time instants $0, \Delta t, \cdots, t-\Delta t, t$. Also, think of $g[t], g[t-\Delta t], \cdots, g[\Delta t], g[0]$ as the system's output impulses at time instants $t, t-\Delta t, \cdots, \Delta t, 0$. The term "impulse response" simply means the system's output in response to an input impulse.

- In response to $f[0]$, the system produces response $f[0]g[0], f[0]g[\Delta t], \cdots, f[0]g[t], \cdots$.
- In response to $f[\Delta t]$, the system produces response $0, f[\Delta t]g[0], \cdots, f[\Delta t]g[t-\Delta t], \cdots$.
- $\cdots$
- In response to $f[t]$, the system produces response $0, 0, \cdots, f[t]g[0], \cdots$.

Therefore,

- At time t, the system's response to $f[0]$ is $f[0]g[t]$.
- At time t, the system's response to $f[\Delta t]$ is $f[\Delta t]g[t-\Delta t]$.
- $\cdots$
- At time t, the system's response to $f[t]$ is $f[t]g[0]$.

The convolution $f*g$ is thus the sum of output impulses responding to input impulses $f[0], f[\Delta t], \cdots, f[t]$, where $\Delta t \to 0$. By approximation, we assume the system input stays at $f[0]$ for the time between 0 and Δt, so the contribution of $f[0]$ to the system response is $f[0]g[t]\Delta t$; by the same reasoning, we multiply Δt with $f[\Delta t]g[t-\Delta t]$, ..., $f[t]g[0]$.

For the convolution, we have a really convenient theorem:

Theorem 6: Convolution Theorem [HS14, Theorem 4.39]

If $\mathcal{L}\{f(t)\} = F(s)$ and $\mathcal{L}\{g(t)\} = G(s)$, then

$$\mathcal{L}\{f * g\} = F(s)G(s).$$

The Convolution Theorem means that if $F(s)$ is the Laplace transform of an input $f(t)$, and $G(s)$ is the Laplace transform of the system's impulse response, then the Laplace transform of the system's response to $f(t)$ is simply the product $F(s)G(s)$. This theorem paves way for the introduction of "transfer functions", and makes the block diagram approach of controller design feasible.

The last key property in this subsection is the *final value theorem*, which is the standard tool for calculating the steady-state value (e.g., response, error) of a system.

Theorem 7: Final value theorem [BtMvdB03]

Let $f(t)$ be a piecewise smooth function with Laplace transform $F(s)$. When $f(\infty) = \lim_{t\to\infty} f(t)$ exists, then

$$f(\infty) = \lim_{s\to 0} sF(s), \tag{27}$$

where the limit $s \to 0$ has to be taken s.t. $\Re\{s\}$ approaches 0 from the positive side.

Above, the reason for emphasizing taking the limit from the positive side is to ensure the operation is performed within the region of convergence.

Quiz 2

Is the Final Value Theorem applicable to $f(t) = \sin(t)$? What about $f(t) = e^{-t}\sin(t)$?

3.2 Transfer function

A transfer function relates the system output to the system input in the Laplace domain; this is why a transfer function is called an *input-output representation*. If the input function and output function have Laplace transforms $U(s)$ and $Y(s)$ respectively, then the Convolution Theorem says there exists $G(s)$ such that $Y(s) = G(s)U(s)$. $G(s)$ is the system's impulse response, and equivalently the system's *transfer function*. Assuming zero initial conditions, the transfer function $G(s)$ is defined as

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

Therefore, given zero initial conditions, the output response in the Laplace domain is the transfer function multiplied with the input.

Example 9

Revisiting the RC circuit of Example 7, we have Eq. (18):

$$Y(s) = \frac{1}{RCs+1}(X(s) + RCy(0)).$$

Setting $y(0) = 0$, we have the transfer function of the RC circuit:

$$G(s) = \frac{Y(s)}{X(s)} = \frac{1}{RCs+1},$$

which represents a first-order system.

The Laplace-domain output response to a unit step input is

$$Y_{\text{step}}(s) = \frac{1}{RCs+1}\left(\frac{1}{s}\right) = \frac{1}{s} - \frac{1}{s+1/(RC)}.$$

The time-domain output response to a unit step unit is thus

$$y_{\text{step}}(t) = \mathcal{L}^{-1}\left\{\frac{1}{s} - \frac{1}{s+1/(RC)}\right\} = 1 - e^{-t/(RC)}.$$

Notice the free response term is a decaying exponential only because the root of the denominator of $G(s)$ is negative. Had $1/(RC)$ been a negative value, the free response would have been an increasing exponential — this kind of response is *unstable*. This example is a preview of the important concepts of *poles* and *stability*.

Definition: Poles and Zeros

A *rational* transfer function is a transfer function that can be written as a polynomial of s divided by another polynomial of s, where the polynominals cannot be degenerate (i.e., con-

taining just a constant term) at the same time. Any rational transfer function of the form

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

can be factorized into the *zero-pole-gain form*

$$G(s) = K \frac{(s - z_1) \cdots (s - z_m)}{(s - p_1) \cdots (s - p_n)}. \tag{28}$$

- The polynomial $D(s)$ is a *characteristic polynomial*, and the equation $D(s) = 0$ is a *characteristic equation*.
- The **poles** of $G(s)$ are the roots of $D(s)$, i.e., the values of s that make $D(s) = 0$. See Figure 7.
- The **zeros** of $G(s)$ are the roots of $N(s)$, i.e., the values of s that make $N(s) = 0$.

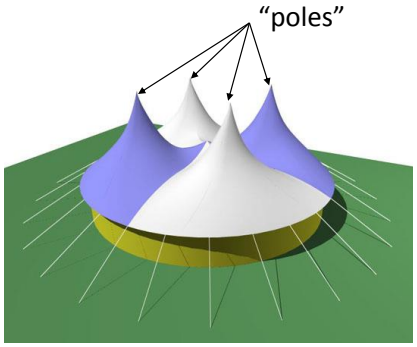


Figure 7: The poles of a transfer function are like the poles of a tent, that are infinitely tall.

Poles and zeros can be real or complex, but for real-world systems, complex poles or zeros must come in conjugate pairs. The *fundamental theorem of algebra* states that any n-degree polynomial has n roots. Hence, any nth-order transfer function has n poles. This means, we can expand any nth-order rational transfer function by partial fraction expansion into n fractions:

$$G(s) = \frac{c_1}{s - p_1} + \cdots + \frac{c_n}{s - p_n},$$

where the poles p_i $(i = 1, \ldots, n)$ can be real or complex. For all the fractions to be inverse-transformable to decaying exponentials, all the poles p_i $(i = 1, \ldots, n)$ must have negative real parts. As long as the input-dependent forced response does not diverge, and the natural response dies off, the output response is stable. Since the poles determine the natural response of the system, we conclude

For a system to be stable, all its poles must have negative real parts.

We will discuss stability more rigorously in a later lecture.

Figure 8 shows various system configurations, where each block represents a transfer function. The Convolution Theorem tells us

- For the open-loop configuration in Figure 8(a), the transfer function

$$T(s) = \frac{Y(s)}{R(s)} = G(s)H(s).$$

$T(s)$ is an open-loop transfer function, since there is no feedback.

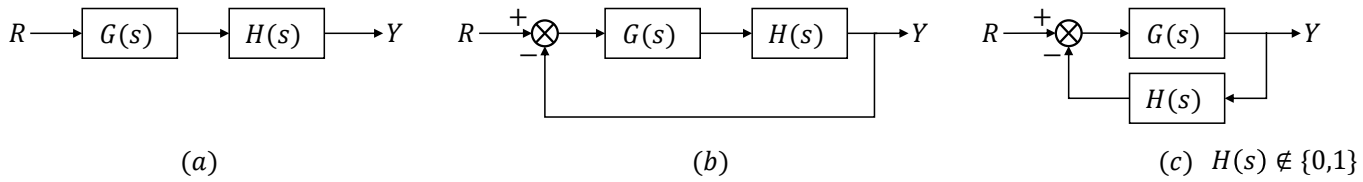


Figure 8: Various control configurations. Each block represents a transfer function. No feedback is employed in (a). Unity feedback is employed in (b). Nonunity feedback is employed in (c).

- For the unity feedback configuration in Figure 8(b),

$$Y(s) = G(s)H(s)(R(s) - Y(s)) \implies Y(s) = \frac{G(s)H(s)}{1 + G(s)H(s)}R(s),$$

so the transfer function

$$T(s) = \frac{Y(s)}{R(s)} = \frac{G(s)H(s)}{1 + G(s)H(s)}. \tag{29}$$

$T(s)$ is a closed-loop transfer function.

- For the nonunity feedback configuration in Figure 8(c),

$$Y(s) = G(s)(R(s) - H(s)Y(s)) \implies Y(s) = \frac{G(s)}{1 + G(s)H(s)}R(s),$$

so the transfer function

$$T(s) = \frac{Y(s)}{R(s)} = \frac{G(s)}{1 + G(s)H(s)}. \tag{30}$$

$T(s)$ is a closed-loop transfer function.

Eqs. (29) and (30) are very useful equations to remember. Unity feedback is more common than nonunity feedback because of its robustness [FPEN02, p. 236].

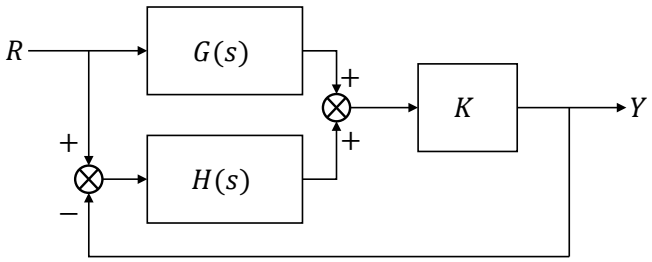


Figure 9: Block diagram for Example 10.

Example 10

Referring to the block diagram in Figure 9, determine the transfer function $Y(s)/R(s)$.

Solution:

$$Y(s) = K[G(s)R(s) + H(s)(R(s) - Y(s))] \implies Y(s)[1 + KH(s)] = [KG(s) + KH(s)]R(s)$$
$$\implies \frac{Y(s)}{R(s)} = \frac{KG(s) + KH(s)}{1 + KH(s)}.$$

3.3 Introduction to discrete-time modeling and the z-transform

Recall that in a sampled-data control system (see Figure 10), the plant operates in continuous time, but the controller operates in discrete time. We can either design the controller in continuous time, or discretize the plant model and design the controller in discrete time. Suppose we try the latter approach. The goal here is to determine the transfer function of a plant in discrete time if we know the Laplace transform of the plant. For this, we need to investigate the effects of sampling and holding (see Figure 11) on any given signal.

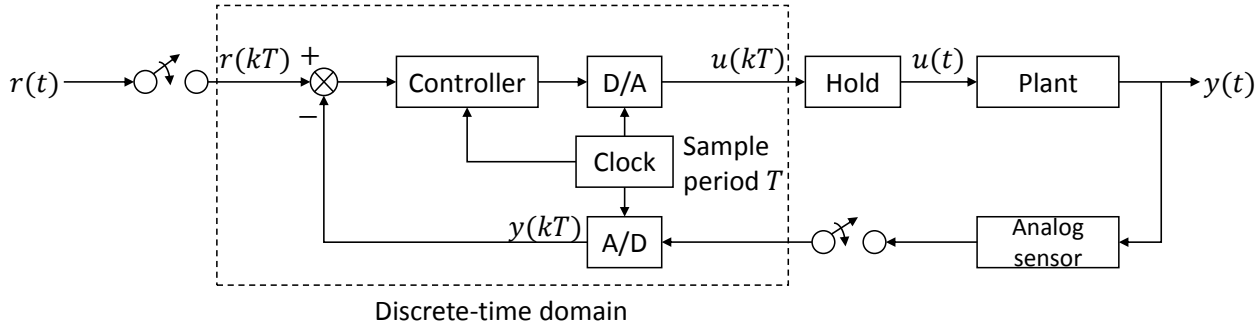


Figure 10: A sampled-data control system.

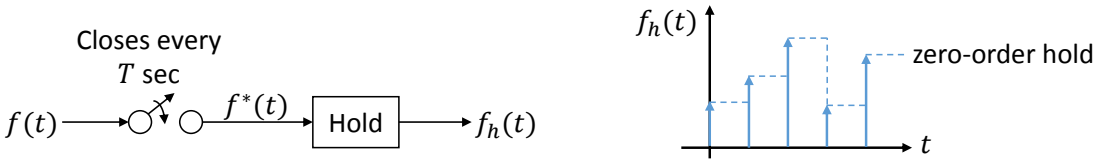


Figure 11: A sampler device and a hold device. We call the sampler "ideal" when we assume it closes for an infinitesimally short amount of time.

Consider the sampling device in Figure 11. We can express the sampled version of $f(t)$, $f^*(t)$, as

$$f^*(t) = f(0)\delta(t) + f(T)\delta(t - T) + f(2T)\delta(t - 2T) + \cdots = \sum_{k=0}^{\infty} f(kT)\delta(t - kT), \tag{31}$$

where T is the sample period, and δ is the Dirac delta function. Applying Laplace transformation and the Second Shifting Theorem (see Theorem 5),

$$F^*(s) = \sum_{k=0}^{\infty} f(kT)e^{-kTs}. \tag{32}$$

Define $z \stackrel{\text{def}}{=} e^{Ts}$, and $F(z) \stackrel{\text{def}}{=} F^*(s)$, then we have the definition of the z-transform for $f(kT)$:

$$F(z) = \sum_{k=0}^{\infty} f(kT) z^{-k}. \tag{33}$$

- Notice by definition $z^{-1} = e^{-Ts}$ is a time-delay operator, i.e.,

$$Y(z) = z^{-1} R(z) \implies y(kT) = r((k-1)T).$$

- A list of popular $f(kT)$ and $F(z)$ pairs can be found in Table 1.
- Table 1 shows the z-transform satisfies the Convolution Theorem, meaning we can treat z-transforms as blocks in a block diagram, like how we treat Laplace transforms. Therefore in the discrete-time domain, we use z-transforms rather than Laplace transforms.
- Table 1 shows the z-transform of an impulse is 1, so a system's transfer function in the z domain, or z-transfer function in short, is equivalent to its impulse response.

For the hold device in Figure 11, we almost always use *zero-order hold* (ZOH), which means given an impulse signal of amplitude A, the ZOH device outputs a pulse of height A and width T. In other words, given an input of A, the output response in the time domain is

$$Au(t) - Au(t-T),$$

which has the Laplace transform $A\left(\frac{1}{s} - \frac{e^{-Ts}}{s}\right)$. Note that we are working in the Laplace domain rather than the z domain, because $f_h(t)$ is a continuous-time signal. Thus, the Laplace transform of the ZOH device is

$$G_{ZOH}(s) = \frac{1 - e^{-Ts}}{s}. \tag{34}$$

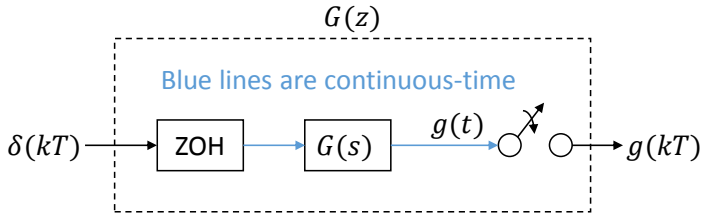


Figure 12: A cascade of a ZOH, a continuous-time plant, and an ideal sampler.

Armed with the knowledge of the z-transform, and the Laplace transform of the ZOH, we can now derive the z-transfer function of the cascade system in Figure 12. The result will be useful because this cascade system is a common part of a sampled-data system, as can be observed in Figure 10. The output response before sampling is given by

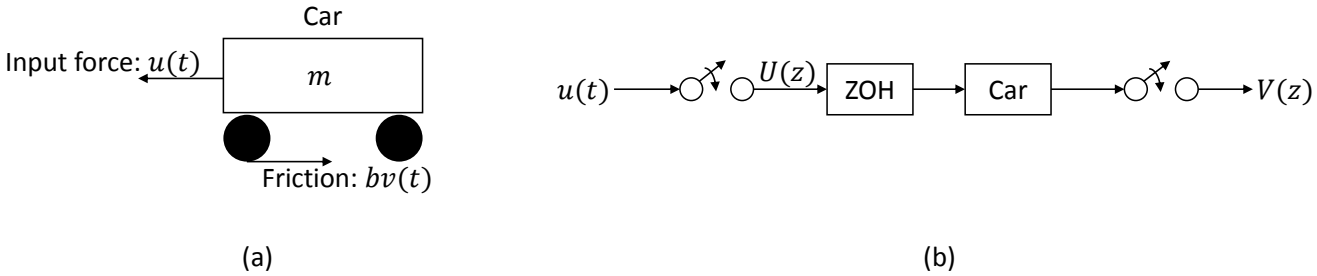
$$g(t) = \mathcal{L}^{-1}\left\{\left(1 - e^{-Ts}\right)\frac{G(s)}{s}\right\} = g_1(t)u(t) - g_1(t-T)u(t-T),$$

where $g_1(t) \stackrel{\text{def}}{=} \mathcal{L}^{-1}\{G(s)/s\}$. The sampled output response is thus $g(kT) = g_1(kT)u(kT) - g_1(kT - T)u(kT - T)$. Using Table 1, we can get the z-transform of $g(kT)$ as

$$G(z) = \mathcal{Z}\{g_1(kT)\} - z^{-1}\,\mathcal{Z}\{g_1(kT)\}.$$

$$\therefore G(z) = (1 - z^{-1})\,\mathcal{Z}\left\{\mathcal{L}^{-1}\left\{\frac{G(s)}{s}\right\}^*\right\}, \tag{35}$$

where $\mathcal{L}^{-1}\{G(s)/s\}^*$ represents the sampled version of $\mathcal{L}^{-1}\{G(s)/s\}$.



(a) (b)

Figure 13: For Example 11: (a) Simplified model for cruise control. (b) Block diagram.

Example 11

This example is adapted from [FV13, Example 3.2]. Consider the model of a cruise control system in Figure 13. Determine the z-transfer function, $G(z) = V(z)/U(z)$.

Solution: Applying Newton's second law, we get the differential equation describing cruise control:

$$m\dot{v} + bv = u.$$

Applying Laplace transformation to both sides, we get the transfer function of the "Car" block in Figure 13(b):

$$G(s) = \frac{1}{ms + b}.$$

The z-transfer function is thus

$$\begin{aligned} G(z) = \frac{V(z)}{U(z)} &= (1 - z^{-1})\,\mathcal{Z}\left\{\mathcal{L}^{-1}\left\{\frac{1}{s(ms + b)}\right\}^*\right\} = \frac{1}{b}(1 - z^{-1})\,\mathcal{Z}\left\{\mathcal{L}^{-1}\left\{\frac{1}{s} - \frac{m}{ms + b}\right\}^*\right\} \\ &= \frac{1}{b}(1 - z^{-1})\,\mathcal{Z}\left\{u(kT) - e^{-bkT/m}u(kT)\right\} \\ &= \frac{1}{b}(1 - z^{-1})\left(\frac{1}{1 - z^{-1}} - \frac{1}{1 - e^{-bT/m}z^{-1}}\right) \\ &= \frac{1}{b}\left(1 - \frac{z - 1}{z - e^{-bT/m}}\right) = \frac{1}{b}\left(\frac{1 - e^{-bT/m}}{z - e^{-bT/m}}\right). \end{aligned}$$

The derived z-transfer function can be validated using the following MATLAB code:

```
syms s m b t k T z
tmp = subs(ilaplace(1/(s*(m*s+b))), t, k*T)
G = simplify((z-1)/z*ztrans(tmp,k,z))
```

At this point, we have caught a glimpse of discrete-time modeling using the z-transform. There is much much more to system modeling, analysis and controller design using the z-transform than this course can cover, but these topics are easier to learn once you know the continuous-time techniques and not vice versa. Next, we will return to the continuous-time domain, but on a different approach than transfer function.

4 State-space equations

An alternative to the *input-output representation* provided by transfer functions is the *state-space representation*. In this representation, the *system state* plays a major role. Whereas an input-output representation is unique (since there is only one way the input affects the output), a state-space representation is not (since some states can be mutually dependent).

To see how state-space equations arise from a set of differential equations, notice the differential equations of Examples 1–4 can be written in the form:

$$\begin{aligned}
\dot{x}_1(t) &= a_{11}(t)x_1(t) + a_{12}(t)x_2(t) + \cdots + a_{1n}(t)x_n(t) + f_1(t),\\
\dot{x}_2(t) &= a_{21}(t)x_1(t) + a_{22}(t)x_2(t) + \cdots + a_{2n}(t)x_n(t) + f_2(t),\\
&\vdots\\
\dot{x}_n(t) &= a_{n1}(t)x_1(t) + a_{n2}(t)x_2(t) + \cdots + a_{nn}(t)x_n(t) + f_n(t),
\end{aligned}$$

or in vector-matrix form,

$$\dot{\boldsymbol{x}}(t) = \mathbf{A}(t)\boldsymbol{x}(t) + \boldsymbol{f}(t), \tag{36}$$

where

$$\dot{\boldsymbol{x}} = \begin{bmatrix} \dot{x}_1 \\ \vdots \\ \dot{x}_n \end{bmatrix}, \quad \mathbf{A} = \begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{n1} & \cdots & a_{nn} \end{bmatrix}, \quad \boldsymbol{x} = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix}, \quad \boldsymbol{f} = \begin{bmatrix} f_1 \\ \vdots \\ f_n \end{bmatrix}.$$

Note my notation: bold italic lower case for vectors (e.g., $\boldsymbol{x}$); bold upright upper case for matrices (e.g., $\mathbf{A}$). The reason for using the first-order form of Eq. (36) is that there is a well-known solution for it.

The solution for

$$\dot{\boldsymbol{x}}(t) = \mathbf{A}\boldsymbol{x}(t) + \boldsymbol{f}(t), \qquad \text{subject to initial state } \boldsymbol{x}(0) = \boldsymbol{x}_0, \tag{37}$$

consists of a homogeneous part and a particular part:

$$\boldsymbol{x}(t) = \boldsymbol{x}_h(t) + \boldsymbol{x}_p(t) = e^{\mathbf{A}t}\boldsymbol{x}_0 + \int_0^t e^{\mathbf{A}(t-\tau)}\boldsymbol{f}(\tau)\,\mathrm{d}\tau. \tag{38}$$

The homogeneous part $\boldsymbol{x}_h(t) = e^{\mathbf{A}t}\boldsymbol{x}_0$ is the solution to the homogeneous equation

$$\dot{\boldsymbol{x}}(t) = \mathbf{A}\boldsymbol{x}(t), \qquad \boldsymbol{x}(0) = \boldsymbol{x}_0.$$

The matrix $e^{\mathbf{A}t}$ is called a **matrix exponential**:

Definition: Matrix exponential

$$e^{\mathbf{X}} \stackrel{\text{def}}{=} \sum_{k=0}^{\infty} \frac{\mathbf{X}^k}{k!} = \mathbf{I} + \mathbf{X} + \frac{1}{2!}\mathbf{X}^2 + \frac{1}{3!}\mathbf{X}^3 + \cdots \tag{39}$$

- It is exactly the same as the regular exponential when $\mathbf{A}$ is a 1-by-1 matrix (i.e., a scalar).
- In EEET 4071 Advanced Control, we will learn multiple ways of computing the matrix exponential.
- Property that matters for now: $\dfrac{\mathrm{d}}{\mathrm{d}t} e^{\mathbf{A}t} = \mathbf{A}e^{\mathbf{A}t} = e^{\mathbf{A}t}\mathbf{A}$.

Example 12

In Example 7, we applied the Laplace approach to solving the ODE describing the RC circuit of Example 1. Show that the time-domain solution is equivalent to the Laplace-domain solution (see Eq. (24)).

Solution: From Example 1, we have Eq. (3):

$$x = RC\dot{y} + y \implies \dot{y} = -\omega_c y + \omega_c x, \tag{40}$$

where we have used the definition $\omega_c \stackrel{\text{def}}{=} 1/(RC)$. Applying Eq. (38) to Eq. (40), we have

$$y(t) = e^{-\omega_c t}y(0) + \int_0^t e^{-\omega_c(t-\tau)}\omega_c x(\tau)\,\mathrm{d}\tau. \tag{41}$$

If $x(t)$ is a sinusoid, e.g., $x(t) = \sin(\omega t)$, then the previous equation becomes

$$y(t) = \omega_c e^{-\omega_c t}\int_0^t e^{\omega_c \tau}\sin(\omega\tau)\,\mathrm{d}\tau + y(0)e^{-\omega_c t}. \tag{42}$$

Performing integration by parts, we have

$$\begin{aligned}\int_0^t e^{\omega_c\tau}\sin(\omega\tau)\,\mathrm{d}\tau &= \int_0^t e^{\omega_c\tau}\,\mathrm{d}\left(-\frac{1}{\omega}\cos(\omega\tau)\right)\\ &= \left[-\frac{1}{\omega}e^{\omega_c\tau}\cos(\omega\tau)\right]_0^t - \int_0^t -\frac{1}{\omega}\cos(\omega\tau)\omega_c e^{\omega_c\tau}\,\mathrm{d}\tau\\ &= \frac{1}{\omega} - \frac{1}{\omega}e^{\omega_c t}\cos(\omega t) + \frac{\omega_c}{\omega}\int_0^t e^{\omega_c\tau}\cos(\omega\tau)\,\mathrm{d}\tau.\end{aligned} \tag{43}$$

We have to perform another integration by parts:

$$\begin{aligned}\int_0^t e^{\omega_c\tau}\cos(\omega\tau)\,\mathrm{d}\tau &= \int_0^t e^{\omega_c\tau}\,\mathrm{d}\left(\frac{1}{\omega}\sin(\omega\tau)\right)\\ &= \frac{1}{\omega}e^{\omega_c t}\sin(\omega t) - \frac{\omega_c}{\omega}\int_0^t e^{\omega_c\tau}\sin(\omega\tau)\,\mathrm{d}\tau.\end{aligned} \tag{44}$$

Substituting Eq. (44) into Eq. (43) gives us

$$\int_0^t e^{\omega_c \tau} \sin(\omega\tau)\,\mathrm{d}\tau = \frac{1}{\omega} - \frac{1}{\omega} e^{\omega_c t}\cos(\omega t) + \frac{\omega_c}{\omega^2} e^{\omega_c t}\sin(\omega t) - \frac{\omega_c^2}{\omega^2}\int_0^t e^{\omega_c \tau}\sin(\omega\tau)\,\mathrm{d}\tau.$$

$$\therefore \int_0^t e^{\omega_c \tau} \sin(\omega\tau)\,\mathrm{d}\tau = \frac{\omega}{\omega^2+\omega_c^2}\left[1 - e^{\omega_c t}\cos(\omega t) + \frac{\omega_c}{\omega} e^{\omega_c t}\sin(\omega t)\right]. \tag{45}$$

After much effort, we finally have a closed-form formula for the integral, which we can substitute back into Eq. (42) and get

$$\begin{aligned} y(t) &= \omega_c e^{-\omega_c t}\int_0^t e^{\omega_c\tau}\sin(\omega\tau)\,\mathrm{d}\tau + y(0)e^{-\omega_c t} \\ &= \frac{\omega\omega_c}{\omega^2+\omega_c^2}\left[e^{-\omega_c t} - \cos(\omega t) + \frac{\omega_c}{\omega}\sin(\omega t)\right] + y(0)e^{-\omega_c t}. \end{aligned} \tag{46}$$

Eq. (46) is exactly the same as Eq. (24) in Example 7. Thus solution in the time domain is equivalent to solution in the Laplace domain.

Although the Laplace approach seems to be easier, state-space solutions

- are necessary for state-space control methods, which are in turn necessary for multivariable control;
- can be efficiently performed using computers; in fact, to solve any set of ODEs in MATLAB numerically, we have to express it in the first-order form first[2].

In this course, we will focus on setting up so-called *linear state-space equations*, and leave state-space controller design to EEET 4071 Advanced Control. Looking back at Eq. (36), if we can express $\boldsymbol{f}$ in the form $\mathbf{B}\boldsymbol{u}$, where $\boldsymbol{u}$ is the input vector (i.e., the vector of input variables), then we get a *state equation*. In general, any continuous-time finite-dimensional LTI system with n states, m inputs and p outputs can be described by

linear state-space equations

$$\dot{\boldsymbol{x}}(t) = \mathbf{A}\boldsymbol{x}(t) + \mathbf{B}\boldsymbol{u}(t), \tag{47}$$

$$\boldsymbol{y}(t) = \mathbf{C}\boldsymbol{x}(t) + \mathbf{D}\boldsymbol{u}(t), \tag{48}$$

where

- $\boldsymbol{x} \in \mathbb{R}^n$ is the *state vector*, $\boldsymbol{u} \in \mathbb{R}^m$ is the *input vector*, $\boldsymbol{y} \in \mathbb{R}^p$ is the *output vector*;
- Eq. (47) is the *state equation*;
- Eq. (48) is the *observation / output / measurement equation*;
- $\mathbf{A} \in \mathbb{R}^{n\times n}$ is the *system / dynamic / evolution matrix*;
- $\mathbf{B} \in \mathbb{R}^{n\times m}$ is the *input / control matrix*;
- $\mathbf{C} \in \mathbb{R}^{p\times n}$ is the *output / observation matrix*;
- $\mathbf{D} \in \mathbb{R}^{p\times m}$ is the *coupling / direct feedthrough / direct transmission / direct transfer / feedforward matrix*.

To obtain the output response for system (47)–(48), we apply Eq. (38) to the state equation (47), and

[2] https://www.mathworks.com/help/releases/R2017b/matlab/math/choose-an-ode-solver.html

substitute the solution into the observation equation (48):

$$\boldsymbol{y}(t) = \mathbf{C}e^{\mathbf{A}t}\boldsymbol{x}_0 + \mathbf{C}\int_0^t e^{\mathbf{A}(t-\tau)}\mathbf{B}\boldsymbol{u}(\tau)\,\mathrm{d}\tau + \mathbf{D}\boldsymbol{u}(t). \tag{49}$$

Example 13

Revisiting the mass-spring-damper system of Example 2, we have Eq. (6):

$$M\ddot{p}(t) = -B\dot{p}(t) - Kp(t) + f(t) \implies \ddot{p}(t) = -\frac{B}{M}\dot{p}(t) - \frac{K}{M}p(t) + \frac{1}{M}f(t), \tag{50}$$

where $f(t)$ is the input, $p(t)$ is the output.

To rewrite the above as state-space equations, we have to determine the states that make up the state vector $\boldsymbol{x}$:

- $\dot{\boldsymbol{x}}$ contains the highest-order derivatives, so $\dot{\boldsymbol{x}}$ must contain $\ddot{p}$, and thus $\boldsymbol{x}$ must contain $\dot{p}$.
- Eq. (50) shows $\ddot{p}$ equals something times $\dot{p}$ plus something times p plus something times f, so $\boldsymbol{x}$ must contain p in addition to $\dot{p}$.

The reasoning above suggests we can define $\boldsymbol{x} \overset{\text{def}}{=} [\dot{p} \quad p]^\top$ ($^\top$ is the transpose operator). Now we can write the state equation as

$$\begin{bmatrix}\ddot{p}\\ \dot{p}\end{bmatrix} = \begin{bmatrix}-\frac{B}{M} & -\frac{K}{M}\\ 1 & 0\end{bmatrix}\begin{bmatrix}\dot{p}\\ p\end{bmatrix} + \begin{bmatrix}\frac{1}{M}\\ 0\end{bmatrix} f, \tag{51}$$

and the observation equation as

$$p = \begin{bmatrix}0 & 1\end{bmatrix}\begin{bmatrix}\dot{p}\\ p\end{bmatrix}. \tag{52}$$

Since there are two variables in the state vector, this is a second-order system.

Example 14

Revisiting the DC motor of Example 3, we have Eqs. (8) and (10):

$$J\ddot{\theta}(t) = -b\dot{\theta}(t) + Ki(t), \tag{53}$$

$$Li' = -K\dot{\theta}(t) - Ri(t) + v(t), \tag{54}$$

where $v(t)$ is the input, $\dot{\theta}(t)$ is the output.

To rewrite the above as state-space equations, we have to determine the states that make up the state vector $\boldsymbol{x}$:

- $\dot{\boldsymbol{x}}$ contains the highest-order derivatives, so $\dot{\boldsymbol{x}}$ must contain $\ddot{\theta}$ and i', and thus $\boldsymbol{x}$ must contain $\dot{\theta}$ and i.
- Eqs. (53)–(54) show that there are no other derivatives of θ and i other than $\dot{\theta}$ and i (zero-order derivative) on the right-hand side.

The reasoning above suggests we can define $\boldsymbol{x} \overset{\text{def}}{=} [\dot{\theta} \quad i]^\top$. Now we can write the state equation

as

$$\begin{bmatrix} \ddot{\theta} \\ i' \end{bmatrix} = \begin{bmatrix} -b/J & K/J \\ -K/L & -R/L \end{bmatrix} \begin{bmatrix} \dot{\theta} \\ i \end{bmatrix} + \begin{bmatrix} 0 \\ 1/L \end{bmatrix} v, \tag{55}$$

and the observation equation as

$$\dot{\theta} = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} \dot{\theta} \\ i \end{bmatrix}. \tag{56}$$

Since there are two variables in the state vector, this is a second-order system.

In general, given an ODE of the form

$$a_n z^{(n)} + a_{n-1} z^{(n-1)} + \cdots + a_1 \dot{z} + a_0 z = bu, \qquad (a_n \neq 0), \tag{57}$$

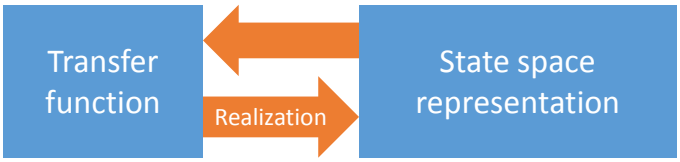
we can define the state vector as $\boldsymbol{x} = [z^{(n-1)} \;\; z^{(n-2)} \;\; \cdots \;\; z]^\top$, and write the state equation accordingly as

$$\dot{\boldsymbol{x}} = \begin{bmatrix} -a_{n-1}/a_n & \cdots & -a_1/a_n & -a_0/a_n \\ 1 & \cdots & 0 & 0 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & 1 & 0 \end{bmatrix} \boldsymbol{x} + \begin{bmatrix} b/a_n \\ 0 \\ \vdots \\ 0 \end{bmatrix} u. \tag{58}$$

If we take z as the output variable, then we can write the observation equation as

$$y = [0 \;\; \cdots \;\; 0 \;\; 1]\boldsymbol{x}. \tag{59}$$

4.1 Realization*



We can convert a state-space representation to a transfer function, and vice versa. Applying Laplace transform to the state equation and initial state

$$\dot{\boldsymbol{x}}(t) = \mathbf{A}\boldsymbol{x}(t) + \mathbf{B}\boldsymbol{u}(t), \qquad \boldsymbol{x}(0) = \boldsymbol{x}_0,$$

we get

$$\begin{aligned} s\boldsymbol{X}(s) - \boldsymbol{x}_0 &= \mathbf{A}\boldsymbol{X}(s) + \mathbf{B}\boldsymbol{U}(s) \\ \implies \boldsymbol{X}(s) &= (s\mathbf{I} - \mathbf{A})^{-1}\boldsymbol{x}_0 + (s\mathbf{I} - \mathbf{A})^{-1}\mathbf{B}\boldsymbol{U}(s) \\ &= \Phi(s)\boldsymbol{x}_0 + \Phi(s)\mathbf{B}\boldsymbol{U}(s), \end{aligned} \tag{60}$$

where $\Phi(s) \overset{\text{def}}{=} (s\mathbf{I} - \mathbf{A})^{-1}$ is called the *resolvent matrix* [HJS08, p.64]. Notice how convolution in the time domain corresponds to multiplication in the s-domain, i.e.,

$$\mathcal{L}\left\{ \int_0^t e^{\mathbf{A}(t-\tau)} \mathbf{B}\boldsymbol{u}(\tau)\, \mathrm{d}\tau \right\} = (s\mathbf{I} - \mathbf{A})^{-1}\mathbf{B}\boldsymbol{U}(s).$$

Similarly, applying Laplace transformation to the observation equation

$$\boldsymbol{y}(t) = \mathbf{C}\boldsymbol{x}(t) + \mathbf{D}\boldsymbol{u}(t),$$

and using Eq. (60), we get

$$\begin{aligned}\boldsymbol{Y}(s) &= \mathbf{C}\boldsymbol{X}(s) + \mathbf{D}\boldsymbol{U}(s) = \mathbf{C}[\Phi(s)\boldsymbol{x}_0 + \Phi(s)\mathbf{B}\boldsymbol{U}(s)] + \mathbf{D}\boldsymbol{U}(s) \\ &= \mathbf{C}\Phi(s)\boldsymbol{x}_0 + (\mathbf{C}\Phi(s)\mathbf{B} + \mathbf{D})\boldsymbol{U}(s).\end{aligned} \tag{61}$$

Setting $\boldsymbol{x}_0 = \mathbf{0}$ gives us the *transfer matrix*

$$\mathbf{G}(s) \stackrel{\text{def}}{=} \mathbf{C}(s\mathbf{I} - \mathbf{A})^{-1}\mathbf{B} + \mathbf{D}. \tag{62}$$

The tuple $(\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D})$ is called a **realization** of $\mathbf{G}(s)$. For SISO systems, $\mathbf{G}(s)$ is scalar and is a transfer function. As a reminder, a transfer function is a unique representation, because there is only one way to relate the output to the input.

It is not always possible to realize a transfer function, because:

Theorem 8: [Che99, Theorem 4.2]

A transfer function is *realizable* if and only if it is *rational* and *proper*.

Definition: Rational and proper transfer function

Suppose $G(s)$ is a transfer function. Furthermore, suppose $G(s)$ is rational, i.e., can be expressed as $\frac{N(s)}{D(s)}$, where $N(s)$ and $D(s)$ are polynomial functions of s.

- $G(s)$ is **strictly proper** if $D(s)$ has a higher degree than $N(s)$.
- $G(s)$ is **semiproper** or **biproper** if $D(s)$ has the same degree as $N(s)$.
- $G(s)$ which is strictly proper or semiproper or biproper is **proper**, i.e., $D(s)$ has the same degree or a higher degree than $N(s)$.
- $G(s)$ is **improper** if $D(s)$ has a lower degree than $N(s)$.

Quiz 3

Suppose transfer function $G(s)$ is strictly proper, as $s \to \infty$, $|G(s)| \to ?$

- Even when we can realize a transfer function in state space, the state-space representation is not unique, since we can pick different states for the state-space equations.
- The most important realization is the *minimal realization* (or *minimum realization*), i.e., a state-space representation of the system in the lowest order; this is covered in EEET 4071 Advanced Control.

In summary, state-space representations have certain advantages over transfer functions:

- A transfer function represents *input-output behavior*, whereas a state-space representation reflects *internal system behavior*, providing deeper insights into system internals.

  - Such insights include *controllability* and *observability*, which form the basis for state-feedback controller designs; these topics are covered in EEET 4071 Advanced Control.

- SISO and multivariable systems can be handled equally in state space.
- The initial state is inherently taken into account.

5 An elaborate example

Supplementary Lecture A has some information on *active suspension*, which uses control to achieve good ride and handling quality. The controller is to counter disturbances due to road roughness, and the forces and moments associated with the various inertial and aerodynamic loadings caused by braking, turning, wind gusts, etc. For designing an active suspension controller, the standard practice is to use the quarter-car model in Figure 14. Here, we will look at the detail of this model [LLGS12]. The relevant variables and constants are as follows.

- Mass m_s is the sprung mass representing the car chassis.
- Mass m_u is the unsprung mass representing the wheel assembly.
- $u(t)$ represents the force exerted by the active suspension controller. For m_s, $u(t)$ is an upward force. For m_u, $u(t)$ is a downward force.
- The k_* constants are stiffness coefficients.
- The c_* constants are damping coefficients.
- The $z_*(t)$ variables measure displacement. Notice $z_r(t)$ measures tire deflection/compression, rather than how far it bounces off the ground.

The input to the plant is $u(t)$. It may not be obvious, but $z_r(t)$ and $\dot{z}_r(t)$ are *disturbances*, and can be thought of as inputs imposed by the road condition. The output is $\ddot{z}_s(t)$, because the control objective is to minimize the vertical acceleration of the chassis to maximize ride comfort.

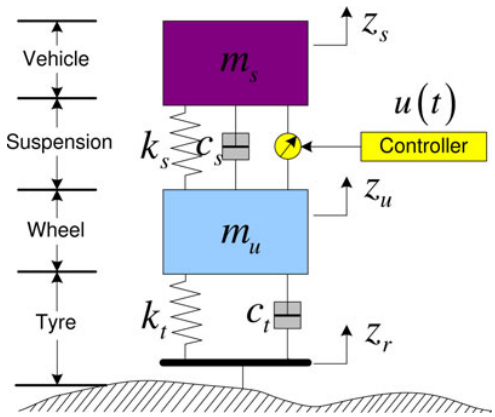


Figure 14: A quarter-car model comprising mass-spring-dampers used for designing active suspension [LLGS12, Fig. 1]. To see generic active suspension in action, see `https://youtu.be/NELQQgRyOjE`. To see electromagnetic active suspension in action, see `https://youtu.be/IQ1eKddstxM`.

The following is similar to what we have done for Example 2. We look at each mass in turn and apply Newton's second law. For mass m_s,

$$m_s\ddot{z}_s + c_s(\dot{z}_s - \dot{z}_u) + k_s(z_s - z_u) = u. \tag{63}$$

For mass m_u,

$$m_u\ddot{z}_u + c_s(\dot{z}_u - \dot{z}_s) + k_s(z_u - z_s) + c_t(\dot{z}_u - \dot{z}_r) + k_t(z_u - z_r) = -u. \tag{64}$$

Our usual practice suggests the state vector should contain $\dot{z}_s$, z_s, $\dot{z}_u$, z_u, $\dot{z}_r$ and z_r, but since $\dot{z}_r$ and z_r are disturbances, we only need $\dot{z}_s$, z_s, $\dot{z}_u$, and z_u. In practice, it is easier to measure the *suspension deflection* $x_{su} \stackrel{\text{def}}{=} z_s - z_u$, and the *tire deflection* $x_{ur} \stackrel{\text{def}}{=} z_u - z_r$, for example using high-resolution encoders, so it makes sense to replace the state variables z_s and z_u with x_{su} and x_{ur}. Substituting x_{su} and x_{ur} into Eqs. (63)–(64), we get

$$\begin{aligned} & m_s\ddot{z}_s + c_s(\dot{z}_s - \dot{z}_u) + k_s x_{su} = u \\ \Longrightarrow\ & \ddot{z}_s = -\frac{c_s}{m_s}\dot{z}_s + \frac{c_s}{m_s}\dot{z}_u - \frac{k_s}{m_s}x_{su} + \frac{1}{m_s}u, \end{aligned} \tag{65}$$

$$\begin{aligned} & m_u\ddot{z}_u + c_s(\dot{z}_u - \dot{z}_s) - k_s x_{su} + c_t(\dot{z}_u - \dot{z}_r) + k_t x_{ur} = -u \\ \Longrightarrow\ & \ddot{z}_u = \frac{c_s}{m_u}\dot{z}_s - \frac{c_s}{m_u}\dot{z}_u + \frac{k_s}{m_u}x_{su} - \frac{c_t}{m_u}\dot{z}_u + \frac{c_t}{m_u}\dot{z}_r - \frac{k_t}{m_u}x_{ur} - \frac{1}{m_u}u. \end{aligned} \tag{66}$$

Now, define $\boldsymbol{x} \stackrel{\text{def}}{=} [x_{su}\ x_{ur}\ \dot{z}_s\ \dot{z}_u]^\top$, then we can write Eqs. (65)–(66) into state-space equations

$$\underbrace{\begin{bmatrix} \dot{x}_{su} \\ \dot{x}_{ur} \\ \ddot{z}_s \\ \ddot{z}_u \end{bmatrix}}_{\dot{\boldsymbol{x}}} = \underbrace{\begin{bmatrix} 0 & 0 & 1 & -1 \\ 0 & 0 & 0 & 1 \\ -\frac{k_s}{m_s} & 0 & -\frac{c_s}{m_s} & \frac{c_s}{m_s} \\ \frac{k_s}{m_u} & -\frac{k_t}{m_u} & \frac{c_s}{m_u} & -\frac{c_s+c_t}{m_u} \end{bmatrix}}_{\mathbf{A}} \underbrace{\begin{bmatrix} x_{su} \\ x_{ur} \\ \dot{z}_s \\ \dot{z}_u \end{bmatrix}}_{\boldsymbol{x}} + \underbrace{\begin{bmatrix} 0 & 0 \\ 0 & -1 \\ \frac{1}{m_s} & 0 \\ -\frac{1}{m_u} & \frac{c_t}{m_u} \end{bmatrix}}_{\mathbf{B}} \begin{bmatrix} u \\ \dot{z}_r \end{bmatrix}, \tag{67}$$

$$\underbrace{\ddot{z}_s}_{\boldsymbol{y}} = \underbrace{\begin{bmatrix} -\frac{k_s}{m_s} & 0 & -\frac{c_s}{m_s} & \frac{c_s}{m_s} \end{bmatrix}}_{\mathbf{C}} \boldsymbol{x} + \underbrace{\begin{bmatrix} \frac{1}{m_s} & 0 \end{bmatrix}}_{\mathbf{D}} \begin{bmatrix} u \\ \dot{z}_r \end{bmatrix}. \tag{68}$$

Using the code in Listing 1, we can determine the transfer matrix to be

$$\mathbf{G}(s) = \begin{bmatrix} \frac{s^2(m_u s^2 + c_t s + k_t)}{m_s m_u s^4 + (c_s m_s + c_t m_s + c_s m_u)s^3 + (k_s m_s + k_t m_s + k_s m_u + c_s c_t)s^2 + (c_s k_t + c_t k_s)s + k_s k_t} \\ \frac{s(k_s + c_s s)(k_t + c_t s)}{m_s m_u s^4 + (c_s m_s + c_t m_s + c_s m_u)s^3 + (k_s m_s + k_t m_s + k_s m_u + c_s c_t)s^2 + (c_s k_t + c_t k_s)s + k_s k_t} \end{bmatrix}^\top. \tag{69}$$

Listing 1: MATLAB code for Sect. 5

```
syms s cs ks ct kt ms mu
A = [0       0       1       -1;
     0       0       0       1;
     -ks/ms  0       -cs/ms  cs/ms;
     ks/mu   -kt/mu  cs/mu   -(cs+ct)/mu];
B = [0      0;
     0      -1;
     1/ms   0;
     -1/mu  ct/mu];
C = [-ks/ms 0       -cs/ms  cs/ms];
D = [1/ms    0];
G = simplify(C*inv(s*eye(size(A))-A)*B+D);
pretty(G)
```

6 Summary

- A "dynamical system" is a system that evolves with time. Typically, they can be described with ODEs or PDEs. In this course, we focus on dynamical systems that can be described with linear ODEs; more specifically, RLC circuits (electrical domain), and mass-spring-dampers (mechanical domain).
- By solving a set of ODEs describing a system, we are essentially predicting its behavior over time, given the input and initial conditions. Solution methods include the Laplace transform (frequency-domain, typical of classical control), and state-space methods (time domain, typical of modern control).
- The Laplace transform is an integral transform, and hence a linear operator.
  - Theorem 2 implies, given zero initial conditions, differentiation in the time domain is equivalent to multiplication with s in the Laplace domain. This theorem and the First Shifting Theorem (see Theorem 4) are indispensable for deriving transfer functions.
  - The Second Shifting Theorem (see Theorem 5) provides a way to model systems with delay. It can also be used to derive the z-transform.
  - The existence of the Convolution Theorem (see Theorem 6) enables the block diagram approach to control system design, where each block in a diagram is a transfer function.
- Difference equations and z-transforms are the equivalents of differential equations and Laplace transforms in the discrete-time domain.
- Some commonly used Laplace transforms and z-transforms can be found in Table 1.
- In state-space methods, a system is modeled with state-space equations. Unlike input-output representations such as transfer functions, state-space representations are not unique because (i) the choice of state variables is not unique, (ii) the order of the states in the state vector is not unique.
  - A state-space representation can always be expressed as a transfer function/matrix.
  - However, not all transfer functions/matrices can be *realized* in state space.

References

[BB11] J.R. Brannan and W.E. Boyce. *Differential equations: an introduction to modern methods and applications*. John Wiley & Sons, Inc., 2011.

[BtMvdB03] R. J. Beerends, H. G. ter Morsche, and J. C. van den Berg. *Fourier and Laplace transforms*. Cambridge University Press, 2003.

[Che99] Chi-Tsong Chen. *Linear System Theory and Design*. Oxford University Press, 3rd edition, 1999.

[FPEN02] G.F. Franklin, J.D. Powell, and A. Emami-Naeini. *Feedback Control of Dynamic Systems*. Prentice-Hall, Inc., 4th edition, 2002.

[FV13] M. Sami Fadali and Antonio Visioli. *Digital Control Engineering: Analysis and Design*. Elsevier, Inc., 2nd edition, 2013.

[HJS08] Elbert Hendricks, Ole Jannerup, and Paul Haase Sorensen. *Linear Systems Control: Deterministic and Stochastic Methods*. Springer-Verlag Berlin Heidelberg, 2008.

[HS14] Martin Hermann and Masoud Saravi. *A First Course in Ordinary Differential Equations: Analytical and Numerical Methods*. Springer India, 2014.

[Lev11] William S. Levine, editor. *The Control Handbook: Advanced Methods*. CRC Press, 2nd edition, 2011.

[LLGS12] Hongyi Li, Honghai Liu, Huijun Gao, and Peng Shi. Reliable fuzzy control for active suspension systems with actuator delay and fault. *IEEE Trans. Fuzzy Syst.*, 20(2):342–357, April 2012.

[Nis11] Norman S. Nise. *Control Systems Engineering*. Wiley, 6th edition, 2011.

[Nis15] Norman S. Nise. *Control Systems Engineering*. Wiley, 7th edition, 2015.

[SHV06] Mark W Spong, Seth Hutchinson, and Mathukumalli Vidyasagar. *Robot modeling and control*. John Wiley & Sons, 2006.

Table 1: Table of Laplace transforms and z-transforms.

- By definition, $f(t) = 0$ for $t \leq 0$.
- $F(s) = \mathcal{L}\{f(t)\}$ is the Laplace transform of $f(t)$.
- $F(z) = \mathcal{Z}\{f[k]\}$ is the z-transform of $f(kT)$ (a pulse train), where T is the sample period.

	$f(t)$	$F(s)$	$f(kT)$	$F(z)$
Impulse	$\delta(t)$	1	$\delta(kT)$	1
Step	$u(t)$	$\frac{1}{s}$	$u(kT)$	$\frac{z}{z-1}$
Ramp	t	$\frac{1}{s^2}$	kT	$\frac{Tz}{(z-1)^2}$
Parabola	t^2	$\frac{2}{s^3}$	k^2T^2	$\frac{T^2z(z+1)}{(z-1)^3}$
Power	t^n	$\frac{n!}{s^{n+1}}$	k^nT^n	$\lim_{a\to 0}(-1)^n \frac{\mathrm{d}^n}{\mathrm{d}a^n}\left(\frac{z}{z-e^{-aT}}\right)$
Sine	$\sin(\omega t)$	$\frac{\omega}{s^2+\omega^2}$	$\sin(\omega kT)$	$\frac{\sin(\omega T)z}{z^2-2\cos(\omega T)z+1}$
Cosine	$\cos(\omega t)$	$\frac{s}{s^2+\omega^2}$	$\cos(\omega kT)$	$\frac{z[z-\cos(\omega T)]}{z^2-2\cos(\omega T)z+1}$
Freq. shift	$e^{-at}f(t)$	$F(s+a)$	$e^{-akT}f(kT)$	$F(e^{aT}z)$
Time shift	$f(t-a)u(t-a)$	$e^{-as}F(s)$	$f(kT-aT)u(kT-aT)$	$z^{-a}F(z)$
Freq. scale			$a^{-k}f(kT)$	$F(az)$
Time scale	$f(at)$	$\frac{1}{\lvert a\rvert}F\left(\frac{s}{a}\right)$		
Freq. diff.	$t^nf(t)$	$(-1)^n\frac{\mathrm{d}^n F(s)}{\mathrm{d}s^n}$	$k^nf(kT)$	$(-z)^n\frac{\mathrm{d}^n F(z)}{\mathrm{d}z^n}$
Time diff.	$f'(t)$	$sF(s)-f(0)$	$f(kT)-f((k-1)T)$	$(1-z^{-1})F(z)$
Convolution	$(f*g)(t)$	$F(s)G(s)$	$(f*g)(kT)$	$F(z)G(z)$
Final value	$f(\infty)$	$\lim_{s\to 0} sF(s)$	$f(\infty)$	$\lim_{z\to 1}(z-1)F(z)$