This homework is due on Tuesday, July 28, 2020, at 11:59PM.
Self-grades are due on Tuesday, August 4, 2020, at 11:59PM.

1 Controllability in circuits

Consider the circuit in Figure 1, where V_s is an input we can control:

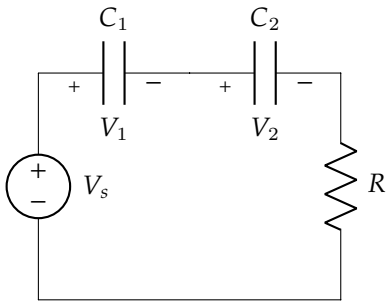


Figure 1: Controllability in circuits

a) Write a state space model for this circuit with V_1 and V_2 as the state variables.

b) Show that this system is not controllable.

c) Explain, in terms of circuit currents and voltages, why this system isn't controllable. (Hint: think about what currents/voltages of the circuit we are controlling with V_s)

d) Draw an equivalent circuit of this system that is controllable. What quantity can you control in this system?

2 Controllability in 2D

Consider the control of some two-dimensional linear discrete-time system

$$\vec{x}(k+1) = A\vec{x}(k) + Bu(k)$$

where A is a 2×2 real matrix and B is a 2×1 real vector.

a) Let $A = \begin{bmatrix} a & 0 \\ c & d \end{bmatrix}$ with $a, c, d \neq 0$, and $B = \begin{bmatrix} f \\ g \end{bmatrix}$. Find a B such that the system is controllable no matter what nonzero values a, c, d take on, and a B for which it is not controllable no matter what nonzero values are given for a, c, d. You can use the controllability rank test, but please explain your intuition as well.

b) Let $A = \begin{bmatrix} a & 0 \\ 0 & d \end{bmatrix}$ with $a, d \neq 0$. and $B = \begin{bmatrix} f \\ g \end{bmatrix}$ with $f, g \neq 0$. Is this system always controllable? If not, find configurations of nonzero a, d, f, g that make the system uncontrollable.

c) We want to see if controllability is preserved under changes of coordinates. To begin with, let $\vec{z}(k) = V^{-1}\vec{x}(k)$, please write out the system equation with respect to $\vec{z}$.

d) Now show that controllability is preserved under change of coordinates. (Hint: use the fact that $\text{rank}(MA) = \text{rank}(A)$ for any invertible matrix M.)

3 Controllability and discretization

In this problem, we will use the car model

$$\frac{d}{dt}x(t) = v(t)$$
$$\frac{d}{dt}v(t) = u(t)$$

that was discussed in class.

a) Assuming that the input $u(t)$ can be varied continuously, is this system controllable?

b) Now assume that we can only change our control input every T seconds. Derive a discrete-time state space model for the state updates, assuming that the input is held constant between times nT and $nT + T$.

c) Is the discrete-time system controllable?

4 The Moore-Penrose pseudoinverse for "wide" matrices

Say we have a set of linear equations given by $A\vec{x} = \vec{y}$. If A is invertible, we know that the solution for $\vec{x}$ is $\vec{x} = A^{-1}\vec{y}$.

However, what if A is not a square matrix? In 16A, you saw how to set up a least squares problem for tall matrices A which gave us a reasonable answer that asks for the "best" match in terms of reducing the norm of the error vector.

Now we will consider the case where A is a wide matrix with more columns than rows. To solve this system, we will walk you through the **Moore-Penrose pseudoinverse** that generalizes the idea of the matrix inverse and is derived from the singular value decomposition.

To find the Moore-Penrose pseudoinverse we start by calculating the SVD of A.

a) Say you have the matrix

$$A = \begin{bmatrix} 1 & -1 & 1 \\ 1 & 1 & -1 \end{bmatrix}.$$

Calculate U, Σ, V such that

$$A = U\Sigma V^T \tag{1}$$

Let us now think about what the SVD does. The matrix A acts on a vector $\vec{x}$ to give the result $\vec{y}$. We have

$$A\vec{x} = U\Sigma V^T\vec{x} = \vec{y}.$$

Observe that V^T rotates the vector, Σ scales the result, and U rotates it again. We will try to "reverse" these operations one at a time and then put them together to construct the Moore-Penrose pseudoinverse.

If U "rotates" the vector $\left(\Sigma V^T\right)\vec{x}$, what operator will undo the rotation?

b) **Derive a matrix that will "unscale", or undo the effect of Σ where it is possible to undo.** Recall that Σ has the same dimensions as A. Ignore any division by zeros (that is to say, let it stay zero).

c) **Derive an operator that would "unrotate" by V^T.**

d) **Try to use this idea of "unrotating" and "unscaling" to derive an "inverse", denoted as $A^\dagger$.** That is to say,

$$\vec{x} = A^\dagger\vec{y}$$

The reason why the word inverse is in quotes (or why this is called a pseudo-inverse) is because we're ignoring the "divisions" by zero.

e) **Use $A^\dagger$ to solve for a vector $\vec{x}$ in the following system of equations.**

$$\begin{bmatrix} 1 & -1 & 1 \\ 1 & 1 & -1 \end{bmatrix}\vec{x} = \begin{bmatrix} 2 \\ 4 \end{bmatrix}$$

Feel free to use the SVD of A from part (a) and numpy.

We will now show that $\vec{x} = A^\dagger\vec{y}$ satisfies the minimality property that $\|\vec{x}\| \leq \|\vec{z}\|$ for all other vectors $\vec{z}$ satisfying $A\vec{z} = \vec{y}$. To do this, suppose A is a generic matrix of rank $m < n$ and let us look at the coordinates of $\vec{x}$ in the V basis.

f) **Find the coordinates of $\vec{x}$ in the basis formed by the columns of V.**

g) Now let $\vec{z}$ be a solution to $A\vec{z} = \vec{y}$. Write out the SVD of A and undo the rotation of U. If $\vec{z}|_V$ is the representation of $\vec{z}$ in the V basis, **what happens to its last $n - m$ entries when left multiplied by Σ? Use this result to show that $\|\vec{x}\| \leq \|\vec{z}\|$.**

5 Weighted Minimum Norm

From the previous problem, given a wide matrix A, we solved $A\vec{x} = \vec{y}$ such that $\vec{x}$ is a minimum norm solution:

$$\|\vec{x}\| \leq \|\vec{z}\|$$

for all $\vec{z}$ such that $A\vec{z} = \vec{y}$. However, what if you weren't interested in minimizing just the norm of $\vec{x}$? What if you instead cared about minimizing $C\vec{x}$? For example, suppose that controls were more or less costly at different times.

The problem can be written out mathematically as:

$$\min_{\vec{z}\in\mathbb{R}^n} \|C\vec{z}\| \text{ subject to } A\vec{z} = \vec{y} \tag{2}$$

To give a concrete example, we define the following matrices

$$A = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \end{bmatrix}, \quad C = \begin{bmatrix} 0 & 0 & 2 \\ 0 & 1 & 0 \\ 2 & 0 & 0 \end{bmatrix}, \quad \vec{y} = \begin{bmatrix} 2 \\ 1 \end{bmatrix}$$

Let's start with the case of C being invertible. We will reformulate the minimum norm problem into one we already now how to solve.

a) Define the change of basis $\vec{w} = C\vec{z}$. Show that the original minimum norm problem (2) can be reformulated as

$$\min_{\vec{w}\in\mathbb{R}^n} \|\vec{w}\| \text{ subject to } AC^{-1}\vec{w} = \vec{y} \tag{3}$$

b) Explain why our new problem is one that we already know how to solve. Then give the optimal solution $\vec{x}$ such that $\|C\vec{x}\| \leq \|C\vec{z}\|$.

c) Now what if C were a tall $p \times n$ matrix with linearly independent columns? This would mean we are no longer able to invert C. Show that the original problem (2) can be reformulated as

$$\min_{\vec{z}\in\mathbb{R}^n} \|B\vec{z}\| \text{ subject to } A\vec{z} = \vec{y} \tag{4}$$

where B is an invertible matrix. You must also show what B is.

Hint: Try using the compact SVD of C to find what B is.

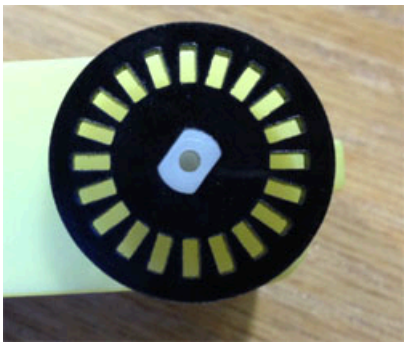
d) Let

$$A = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \end{bmatrix}, \quad C = \begin{bmatrix} 2 & 0 & 1 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 0 & 0 \end{bmatrix}, \quad \vec{y} = \begin{bmatrix} 2 \\ 1 \end{bmatrix}$$

Based on the reformulation from the previous part, solve for $\vec{x}$ such that $\|C\vec{x}\| \leq \|C\vec{z}\|$.
Note: You will most likely need to use numpy to do these calculations

6 Understanding the SIXT33N Car Control Model

As we continue along the process of making the SIXT33N cars awesome, we'd like to better understand the car model that we will be using to develop a control scheme. As a wheel on the car turns, there is an encoder disc (see below) that also turns as the wheel turns. The encoder shines a light though the encoder disc, and as the wheel turns, the light is continually blocked and unblocked, allowing the encoder to detect how fast the wheel is turning by looking at the number of times that the light "ticks" between being blocked and unblocked over a specific time interval.



The following model applies separately to each wheel (and associated motor) of the car:

$$v[k] = d[k+1] - d[k] = \theta u[k] - \beta$$

Meet the variables at play in this model:

- k - The current timestep of the model. Since we model the car as a discrete system, this will advance by 1 on every new sample in the system.
- $d[k]$ - The total number of ticks advanced by a given encoder (the values may differ for the left and right motors–think about when this would be the case).
- $v[k]$ - The discrete-time velocity (in units of ticks/timestep) of the wheel, measured by finding the difference between two subsequent tick counts ($d[k+1] - d[k]$).
- $u[k]$ - The input to the system. The motors that apply force to the wheels are driven by an input voltage signal. This voltage is delivered via a technique known as pulse width modulation (PWM), where the average value of the voltage (which is what the motor is responsive to) is controlled by changing the duty cycle of the voltage waveform. The duty cycle, or percentage of the square wave's period for which the square wave is HIGH, is mapped to the range $[0, 255]$. Thus, $u[k]$ takes a value in $[0, 255]$ representing the duty cycle. For example, when $u[k] = 255$, the duty cycle is 100 %, and the motor controller just delivers a constant signal at the system's HIGH voltage, delivering the maximum possible power to the motor. When $u[k] = 0$, the duty cycle is 0 %, and the motor controller delivers 0 V.
- θ - Relates change in input to change in velocity: if the wheel rotates through n ticks in one timestep for a given $u[k]$ and m ticks in one timestep for an input of $u[k]+1$, then $\theta = m - n = \frac{\Delta v[k]}{\Delta u[k]} = \frac{v_{u_1[k]}[k] - v_{u_0[k]}[k]}{u_1[k] - u_0[k]}$. **Its units are ticks/(timestep · duty cycle).** Since our model is linear, we assume that θ is the same for every unit increase in $u[k]$. This is empirically measured using the car: θ depends on many physical phenomena, so for the purpose of this class, we will not attempt to create a mathematical model based on the actual physics. However, you can conceptualize θ as a "sensitivity factor",

representing the idiosyncratic response of your wheel and motor to a change in power (you will have a separate θ for your left and your right wheel).

- β - Similarly to θ, β is dependent upon many physical phenomena, so we will empirically determine it using the car. β represents a constant offset in velocity, and hence **its units are ticks/timestep**. Note that you will also have a different β for your left and your right wheel.

In this problem (except parts (c) and (e)) we will assume that the wheel conforms perfectly to this model to get an intuition of how the model works.

a) If we wanted to make the wheel move at a certain target velocity v^*, what input $u[k]$ should we provide to the motor that drives it? Your answer should be symbolic, and in terms of v^*, $u[k]$, θ, and β.

b) Even if the wheel and the motor driving it conform perfectly to the model, our inputs still limit the range of velocities. Given that $0 \leq u[k] \leq 255$, determine the maximum and minimum velocities possible for the wheel. How can you slow the car down?

c) Our intuition tells us that a wheel on a car should eventually stop turning if we stop applying any power to it. Find $v[k]$ assuming that $u[k] = 0$. Does the model obey your intutition? What does that tell us about our model?

d) In order to characterize the car, we need to find the θ and β values that model your left and right wheels and their motors: θ_l, β_l, θ_r, and β_r. How would you determine θ and β empirically? What data would you need to collect? *Hint:* keep in mind we also know the input $u[k]$ for all k.

e) How can you use the data you collected in part (d) to mitigate the effect of the system's nonlinearity and/or minimize model mismatch? *Hint:* you will only wind up using a small range of the possible input values in practice. There are several reasons this is true, but one is that each motor has a characteristic attainable velocity range, and for your car to drive straight, we need the wheels to rotate with the same velocity.

7 Open-Loop Control of SIXT33N

Last time, we learned that the ideal input PWM for running a motor at a target velocity v^* is:

$$u(t) = \frac{v^* + \beta}{\theta}$$

In this problem, we will extend our analysis from one motor to a two-motor car system and evaluate how well our open-loop control scheme does.

$$v_L(t) = d_L(t+1) - d_L(t) = \theta_L u_L(t) - \beta_L$$
$$v_R(t) = d_R(t+1) - d_R(t) = \theta_R u_R(t) - \beta_R$$

a) In reality, we need to "kickstart" electric motors with a pulse in order for them to work. That is, we can't go straight from 0 to our desired input signal for $u(t)$, since the motor needs to overcome its initial inertia in order to operate in accordance with our model.

Let us model the pulse as having a width (in timesteps) of t_p. In order to model this phenomenon, we can say that $u(t) = 255$ for $t \in [0, t_p - 1]$[1]. In addition, the car initially (at $t = 0$) hasn't moved, so we can also say $d(0) = 0$.

Firstly, let us examine what happens to d_L and d_R at $t = t_p$, that is, after the kickstart pulse has passed. Find $d_L(t_p)$ and $d_R(t_p)$. (*Hint:* If it helps, try finding $d_L(1)$ and $d_R(1)$ first and then generalizing your result to the t_p case.)

Note: It is very important that you distinguish θ_L and θ_R as the motors we have are liable to vary in their parameters, just as how real resistors vary from their ideal resistance.

b) Let us define $\delta(t) = d_L(t) - d_R(t)$ as the difference in positions between the two wheels. If both wheels of the car are going at the same velocity, then this difference δ should remain constant since no wheel will advance by more ticks than the other. As a result, this will be useful in our analysis and in designing our control schemes.

Find $\delta(t_p)$. For both an ideal car ($\theta_L = \theta_R$ and $\beta_L = \beta_R$) where both motors are perfectly ideal and a non-ideal car ($\theta_L \neq \theta_R$ and $\beta_L \neq \beta_R$), did the car turn compared to before the pulse?

Note: Since $d(0) = d_L(0) = d_R(0) = 0$, $\delta(0) = 0$.

c) We can still declare victory though, even if the car turns a little bit during the initial pulse (t_p will be very short in lab), so long as the car continues to go straight afterwards when we apply our control scheme; that is, as long as $\delta(t \to \infty)$ converges to a constant value (as opposed to going to $\pm\infty$ or oscillating).

Let's try applying the open-loop control scheme we learned last week to each of the motors independently, and see if our car still goes straight.

$$u_L(t) = \frac{v^* + \beta_L}{\theta_L}$$
$$u_R(t) = \frac{v^* + \beta_R}{\theta_R}$$

Let $\delta(t_p) = \delta_0$. Find $\delta(t)$ for $t \geq t_p$ in terms of δ_0. (*Hint:* As in part (a), if it helps you, try finding $\delta(t_p + 1)$, $\delta(t_p + 2)$, etc., and generalizing your result to the $\delta(t)$ case.)

Does $\delta(t \to \infty)$ deviate from δ_0? Why or why not?

[1] $x \in [a, b]$ means that x goes from a to b inclusive.

d) Unfortunately, in real life, it is hard to capture the precise parameters of the car motors like θ and β, and even if we did manage to capture them, they could vary as a function of temperature, time, wheel conditions, battery voltage, etc. In order to model this effect of **model mismatch**, we consider model mismatch terms (such as $\Delta\theta_L$), which reflects the discrepancy between the model parameters and actual parameters.

$$v_L(t) = d_L(t+1) - d_L(t) = (\theta_L + \Delta\theta_L)u_L(t) - (\beta_L + \Delta\beta_L)$$
$$v_R(t) = d_R(t+1) - d_R(t) = (\theta_R + \Delta\theta_R)u_R(t) - (\beta_R + \Delta\beta_R)$$

Let us try applying the open-loop control scheme again to this new system. Note that **no model mismatch terms appear below** – this is intentional since our control scheme is derived from the model parameters for θ and β, not from the actual $\theta + \Delta\theta$, etc.[2]

$$u_L(t) = \frac{v^* + \beta_L}{\theta_L}$$
$$u_R(t) = \frac{v^* + \beta_R}{\theta_R}$$

As before, let $\delta(t_p) = \delta_0$. Find $\delta(t)$ for $t \geq t_p$ in terms of δ_0.

Does $\delta(t \to \infty)$ change from δ_0? Why or why not, and how is it different from the previous case of no model mismatch?

You may have noticed that open-loop control is insufficient in light of non-idealities and mismatches. Next time, we will analyze a more powerful form of control (closed-loop control) which should be more robust against these kinds of problems.

[2]Why not just do a better job of capturing the parameters, one may ask? Well, as noted above, the mismatch can vary as a function of an assortment of factors including temperature, time, wheel conditions, battery voltage, and it is not realistic to try to capture the parameters under every possible environment, so it is up to the control designer to ensure that the system can tolerate a reasonable amount of mismatch.

8 Homework Process and Study Group

Citing sources and collaborators are an important part of life, including being a student! We also want to understand what resources you find helpful and how much time homework is taking, so we can change things in the future if possible.

a) **What sources (if any) did you use as you worked through the homework?**

b) **If you worked with someone on this homework, who did you work with?** List names and student ID's. (In case of homework party, you can also just describe the group.)

c) **How did you work on this homework?** (For example, *I first worked by myself for 2 hours, but got stuck on problem 3, so I went to office hours. Then I went to homework party for a few hours, where I finished the homework.*)

d) **Do you have any feedback on this homework assignment?**

e) **Roughly how many total hours did you work on this homework?**