EECS 16A Designing Information Devices and Systems I
Fall 2017 Homework 3

This homework is due September 18, 2017, at 23:59.
Self-grades are due September 21, 2017, at 23:59.

Submission Format

Your homework submission should consist of **two** files.

- `hw3.pdf`: A single PDF file that contains all of your answers (any handwritten answers should be scanned) as well as your IPython notebook saved as a PDF.

  If you do not attach a PDF of your IPython notebook, you will not receive credit for problems that involve coding. Make sure that your results and your plots are visible.

- `hw3.ipynb`: A single IPython notebook with all of your code in it.

  In order to receive credit for your IPython notebook, you must submit both a "printout" and the code itself.

Submit each file to its respective assignment on Gradescope.

1. Mechanical Inverses

For each of the following matrices, state whether the inverse exists. If so, find the inverse. If not, show why no inverse exists. **Solve these by hand. Do NOT use IPython for this problem.**

(a) $\begin{bmatrix} 1 & 1 \\ 2 & 0 \end{bmatrix}$

(b) $\begin{bmatrix} 3 & 2 \\ 1 & -1 \end{bmatrix}$

(c) $\begin{bmatrix} -\frac{1}{2} & -\frac{\sqrt{3}}{2} \\ \frac{\sqrt{3}}{2} & -\frac{1}{2} \end{bmatrix}$

(d) $\begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 2 \end{bmatrix}$

(e) $\begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 1 \\ 0 & 1 & 0 \end{bmatrix}$

(f) $\begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & 2 \\ 1 & 4 & 4 \end{bmatrix}$

(g) **PRACTICE:** $\begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 1 \\ 0 & 1 & -1 \end{bmatrix}$

(h) **PRACTICE:** $\begin{bmatrix} -1 & 1 & -\frac{1}{2} \\ 1 & 1 & -\frac{1}{2} \\ 0 & 1 & 1 \end{bmatrix}$

(i) $\begin{bmatrix} 3 & 0 & -2 & 1 \\ 0 & 2 & 1 & 3 \\ 3 & 1 & 0 & 4 \\ 1 & 0 & 0 & 1 \end{bmatrix}$

2. Pumps Properties Proofs

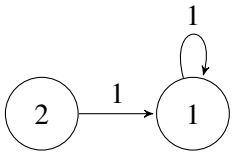


Figure 1: Pump system.

(a) Suppose that we have a pump setup as in Figure 1 with the associated transition matrix **A**. Write out the state transition matrix **A**.

$$\vec{x}[n+1] = \mathbf{A}\vec{x}[n]$$

(b) Suppose that the reservoirs are initialized to the following water levels: $x_1[0] = 0.5, x_2[0] = 0.5$. In a completely alternate universe, the reservoirs are initialized to the following water levels: $x_1[0] = 0.3, x_2[0] = 0.7$. For both initial states, what are the water levels at timestep 1 ($\vec{x}[1]$)?

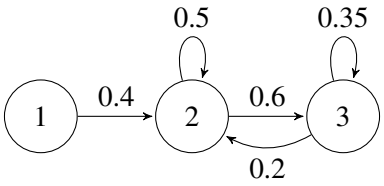
(c) If you observe the reservoirs at timestep 1, can you figure out what the initial ($\vec{x}[0]$) water levels were? Why or why not?

(d) Now generalize: if there exists a state transition matrix where two different initial state vectors lead to the same water levels/state vectors at a timestep in the future, can you recover the initial water levels? Prove your answer.
(*Hint: What does this say about the matrix* $\boldsymbol{A}$*?*)

(e) Suppose that there is a state transition matrix, where every initial state is guaranteed to have a unique state vector in the next timestep. Consider what this statement implies about the system of linear equations represented by the state transition matrix **A**. Can you recover the initial state? Prove your answer, and explain what this implies about the system.

(f) Suppose that there is a state transition matrix such that the entries of each column vector sum to one. What is the physical interpretation about the total amount of water in the system?

(g) Set up the state transition matrix **A** for the system of pumps shown below. Explain what this **A** matrix physically implies about the total amount of water in this system.
Note: If there is no "self-arrow/self-loop," then the water does not return.

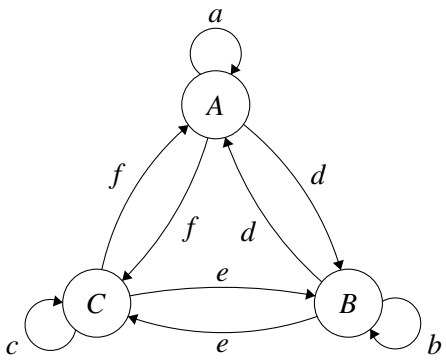


(h) There is a state transition matrix where the entries of its rows sum to one. Prove that applying this system to a uniform vector will return the same uniform vector. A uniform vector is a vector whose elements are all the same.

$$\begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & a_{ij} & \vdots \\ a_{n1} & \cdots & a_{nn} \end{bmatrix} \begin{bmatrix} x \\ \vdots \\ x \end{bmatrix} = \begin{bmatrix} x \\ \vdots \\ x \end{bmatrix}$$

3. Reservoirs That Give and Take

Consider a network of three water reservoirs A, B, and C. At the end of each day, water transfers among the reservoirs according to the directed graph shown below.



The parameters a, b, and c—which label the self-loops—denote the *fractions* of the water in reservoirs A, B, and C, respectively, that stay in the same reservoir at the end of each day n. The parameters d, e, and f denote the fractions of the reservoir contents that transfer to adjacent reservoirs at the end of each day, according to the directed graph above.

Assume that the reservoir system is conservative—no water enters or leaves the system, which means that the total water in the network is constant. Accordingly, *for each node*, the weights on its self-loop and its two outgoing edges sum to 1; for example, for node A, we have

$$a+d+f=1,$$

and similar equations hold for the other nodes. Moreover, assume that all the edge weights are positive numbers—that is,

$$0 < a,b,c,d,e,f < 1.$$

The state evolution equation governing the water flow dynamics in the reservoir system is given by $\vec{s}[n+1] = \mathbf{A}\vec{s}[n]$, where the 3×3 matrix $\mathbf{A}$ is the state transition matrix, and $\vec{s}[n] = \begin{bmatrix} s_A[n] & s_B[n] & s_C[n] \end{bmatrix}^T \in \mathbb{R}^3$ is the nonnegative state vector that shows the water distribution among the three reservoirs at the end of day n, as fractions of the total water in the network.

(a) Determine the state transition matrix $\mathbf{A}$.

(b) For some systems, there exists an equilibrium state—that is, a state for which the following is true:

$$\vec{s}[n+1] = \vec{s}[n] = \vec{s}^*$$

Determine if for the systems described above, there exists a equilibrium state. If such a state exists, find it.

Hint: What's special about the rows of this matrix?

(c) Suppose the state transition matrix for the network is given by

$$\mathbf{A} = \begin{bmatrix} \frac{1}{5} & \frac{2}{5} & \frac{2}{5} \\ \frac{2}{5} & \frac{1}{5} & \frac{2}{5} \\ \frac{2}{5} & \frac{2}{5} & \frac{1}{5} \end{bmatrix}.$$

Is it possible to determine the state $\vec{s}[n]$ from the subsequent state $\vec{s}[n+1]$? You do not have to find $\vec{s}[n]$ from $\vec{s}[n+1]$, just determine whether it is possible to do so.

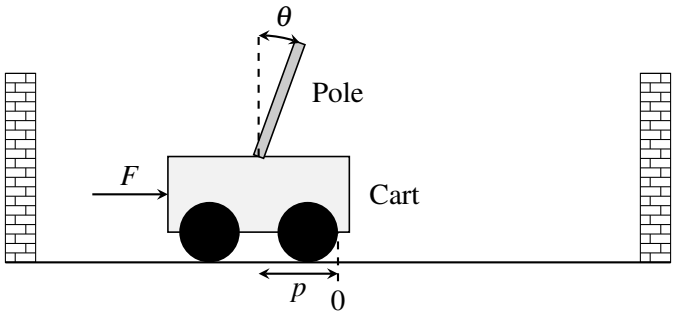
4. Bieber's Segway

After one too many infractions with the law, J-Biebs decides to find a new mode of transportation, and you suggest he get a segway.

He becomes intrigued by your idea and asks you how it works.

You let him know that a force (through the spinning wheel) is applied to the base of the segway, and this in turn controls both the position of the segway and the angle of the stand. As you push on the stand, the segway tries to bring itself back to the upright position, and it can only do this by moving the base.

J-Biebs is impressed, to say the least, but he is a little concerned that only one input (force) is used to control two outputs (position and angle). He finally asks if it's possible for the segway to be brought upright and to a stop from any initial configuration. J-Biebs calls up a friend who's majoring in mechanical engineering, who tells him that a segway can be modeled as a cart-pole system:



A cart-pole system can be fully described by its position p, velocity $\dot{p}$, angle θ, and angular velocity $\dot{\theta}$. We write this as a "state vector":

$$\vec{x} = \begin{bmatrix} p \\ \dot{p} \\ \theta \\ \dot{\theta} \end{bmatrix}$$

The input to this system u will just be the force applied to the cart (or base of the segway).[1]

[1] Some of you might be wondering why it is that applying a force in this model immediately causes a change in position. You might have been taught in high school physics that force creates acceleration, which changes velocity (both simple and angular), which in turn causes changes to position and angle. Indeed, when viewed in continuous time, this is true instantaneously. However, here in this engineering system, we have discretized time, i.e. we think about applying this force constantly for a given finite duration and we see how the system responds after that finite duration. In this finite time, indeed the application of a force will cause changes to all four components of the state. But notice that the biggest influence is indeed on the two velocities, as your intuition from high school physics would predict.

At time step n, we can apply scalar input $u[n]$. The cart-pole system can be represented by a linear model:

$$\vec{x}[n+1] = \mathbf{A}\vec{x}[n] + \vec{b}u[n], \tag{1}$$

where $\mathbf{A} \in \mathbb{R}^{4\times 4}$ and $\vec{b} \in \mathbb{R}^{4\times 1}$. The model tells us how the the state vector will evolve over (discrete) time as a function of the current state vector and control inputs.

To answer J-Biebs' question, you look at this general linear system and try to answer the following question: Starting from some initial state $\vec{x}_0$, can we reach a final desired state, $\vec{x}_f$, in N steps?

The challenge seems to be that the state is 4-dimensional and keeps evolving and that we can only apply a one dimensional control at each time. Is it possible to control something 4-dimensional with only one degree of freedom that we can control?

You will solve this problem by walking through several steps.

(a) Express $\vec{x}[1]$ in terms of $\vec{x}[0]$ and the input $u[0]$.

(b) Express $\vec{x}[2]$ in terms of *only* $\vec{x}[0]$ and the inputs, $u[0]$ and $u[1]$. Then express $\vec{x}[3]$ in terms of *only* $\vec{x}[0]$ and the inputs, $u[0]$, $u[1]$, and $u[2]$, and express $\vec{x}[4]$ in terms of *only* $\vec{x}[0]$ and the inputs, $u[0]$, $u[1]$, $u[2]$, and $u[3]$.

(c) Now, derive an expression for $\vec{x}[N]$ in terms of $\vec{x}[0]$ and the inputs from $u[0], \ldots, u[N-1]$. (*Hint: You can use a summation from* 0 *to* $N-1$.)

For the next four parts of the problem, you are given the matrix $\mathbf{A}$ and the vector $\vec{b}$:

$$\mathbf{A} = \begin{bmatrix} 1 & 0.05 & -0.01 & 0 \\ 0 & 0.22 & -0.17 & -0.01 \\ 0 & 0.10 & 1.14 & 0.10 \\ 0 & 1.66 & 2.85 & 1.14 \end{bmatrix}$$

$$\vec{b} = \begin{bmatrix} 0.01 \\ 0.21 \\ -0.03 \\ -0.44 \end{bmatrix}$$

The state vector $\vec{0}$ corresponds to the cart-pole (or segway) being upright and stopped at the origin. Assume the cart-pole starts in an initial state $\vec{x}[0] = \begin{bmatrix} -0.3853493 \\ 6.1032227 \\ 0.8120005 \\ -14 \end{bmatrix}$, and you want to reach the desired state $\vec{x}_f = \vec{0}$ using the control inputs $u[0], u[1], \ldots$. (Reaching $\vec{x}_f = \vec{0}$ in N steps means that, given that we start at $\vec{x}[0]$, we can find control inputs, such that we get $\vec{x}[N]$, the state vector at the Nth time step, equal to $\vec{x}_f$.)

Note: You may use IPython to solve the next four parts of the problem. We have provided a function `gauss_elim(matrix)` to help you find the reduced row echelon form of matrices.

(d) Can you reach $\vec{x}_f$ in *two* time steps? (*Hint: Express* $\vec{x}[2] - \mathbf{A}^2\vec{x}[0]$ *in terms of the inputs* $u[0]$ *and* $u[1]$.)

(e) Can you reach $\vec{x}_f$ in *three* time steps?

(f) Can you reach $\vec{x}_f$ in *four* time steps?

(g) If the answer to the previous part is yes, find the required correct control inputs using IPython and verify the answer by entering these control inputs into the code in the IPython notebook. The code has been written to simulate this system, and you should see the system come to a halt in four time steps! *Suggestion: See what happens if you enter all four control inputs equal to 0. This gives you an idea of how the system naturally evolves!*

(h) Let's return to a general matrix $\mathbf{A}$ and a general vector $\vec{b}$. What condition do we need on

$$\text{span}\left\{\vec{b}, \mathbf{A}\vec{b}, \mathbf{A}^2\vec{b}, \ldots, \mathbf{A}^{N-1}\vec{b}\right\}$$

for $\vec{x}_f = \vec{0}$ to be "reachable" from $\vec{x}_0$ in N steps?

(i) What condition would we need on span $\left\{\vec{b}, \mathbf{A}\vec{b}, \mathbf{A}^2\vec{b}, \ldots, \mathbf{A}^{N-1}\vec{b}\right\}$ for *any* valid state vector to be reachable from $\vec{x}_0$ in N steps?
Wouldn't this be cool?

5. Write Your Own Question And Provide a Thorough Solution.

Writing your own problems is a very important way to really learn material. The famous "Bloom's Taxonomy" that lists the levels of learning is: Remember, Understand, Apply, Analyze, Evaluate, and Create. Using what you know to create is the top level. We rarely ask you any homework questions about the lowest level of straight-up remembering, expecting you to be able to do that yourself (e.g. making flashcards). But we don't want the same to be true about the highest level. As a practical matter, having some practice at trying to create problems helps you study for exams much better than simply counting on solving existing practice problems. This is because thinking about how to create an interesting problem forces you to really look at the material from the perspective of those who are going to create the exams. Besides, this is fun. If you want to make a boring problem, go ahead. That is your prerogative. But it is more fun to really engage with the material, discover something interesting, and then come up with a problem that walks others down a journey that lets them share your discovery. You don't have to achieve this every week. But unless you try every week, it probably won't ever happen.

6. Homework Process and Study Group

Who else did you work with on this homework? List names and student ID's. (In case of homework party, you can also just describe the group.) How did you work on this homework?