EECS 16A Designing Information Devices and Systems I
Spring 2019 Homework 3

This homework is due February 15, 2019, at 23:59.
Self-grades are due February 19, 2019, at 23:59.

Submission Format

Your homework submission should consist of **one** file.

- `hw2.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 "printout" of your IPython notebook, you will not receive credit for problems that involve coding. Make sure that your results and your plots are visible. Assign the IPython printout to the correct problem(s) on Gradescope.

Submit each file to its respective assignment on Gradescope.

1. Lab Grades and Discussion Attendance

Lab grades and discussion attendance will be posted on Gradescope. These will be updated periodically; check the Piazza post for the most recent updates.

For lab grades, please submit a file named `sid.txt` to the "Lab Grades" assignment. This text file should contain only your SID number and nothing else (e.g. no "" marks or extra whitespace). You can find a template in the iPython folder.

For discussion grades, submit the same file `sid.txt` to "Discussion Attendance."

If you use a strange text editor, the autograder will give a strange error message complaining about the unicode format. If this happens, please use a different editor and reupload.

Describe how to find your lab and discussion grades.

2. Elementary Matrices

In lecture, we learned about an important technique for solving systems of linear equations called Gaussian elimination. It turns out that each row operation in Gaussian elimination can be performed by multiplying the augmented matrix on the left by a specific matrix called an *elementary matrix*. For example, suppose we want to row reduce the following augmented matrix:

$$\mathbf{A} = \left[\begin{array}{cccc|c} 1 & -2 & 0 & -5 & 15 \\ 0 & 1 & 0 & 3 & -7 \\ -2 & -3 & 1 & -6 & 9 \\ 0 & 1 & 0 & 2 & -5 \end{array}\right] \tag{1}$$

What matrix do you get when you subtract the 4th row from the 2nd row of $\mathbf{A}$ (putting the result in row 2)?

(You don't have to include this in your solution.) Now, try multiplying the original $\mathbf{A}$ on the left by

$$\mathbf{E} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

(You don't have to include this in your solutions either.) Notice that you get the same thing.

$$\mathbf{EA} = \left[\begin{array}{cccc|c} 1 & -2 & 0 & -5 & 15 \\ 0 & 0 & 0 & 1 & -2 \\ -2 & -3 & 1 & -6 & 9 \\ 0 & 1 & 0 & 2 & -5 \end{array}\right]$$

$\mathbf{E}$ is a special type of matrix called an *elementary matrix*. This means that we can obtain the matrix $\mathbf{E}$ from the identity matrix by applying an elementary row operation – in this case, subtracting the 4th row from the 2nd row.

In general, any elementary row operation can be performed by left multiplying by an appropriate elementary matrix.

In other words, you can perform a row operation on a matrix $\mathbf{A}$ by first performing that row operation on the identity matrix to get an elementary matrix (see below), and then left multiplying $\mathbf{A}$ by the elementary matrix (like we did above).

$$\mathbf{I} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \xrightarrow{R_2 - R_4 \mapsto R_2} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \mathbf{E}$$

(a) Write down the elementary matrices required to perform the following row operations on a 4×5 augmented matrix.

i. $R_1 \mapsto R_3$
$R_3 \mapsto R_1$

ii. $-5R_3 \mapsto R_3$

iii. $3R_2 + R_4 \mapsto R_4$
$R_1 - R_2 \mapsto R_1$

Hint: For the last one, note that if you want to perform two row operations on the matrix $\mathbf{A}$, you can perform them both on the identity matrix and then left multiply $\mathbf{A}$ by the resulting matrix.

(b) In lecture we emphasized using Gaussian Elimination to reach an upper triangular form to determine the number of solutions for a given system of linear equations. When there is a unique solution, however, it is useful to determine exactly what that solution is by continuing Gaussian Elimination to reach a "fully reduced" form like that shown below. An example of this process can be found in Note 1, Example 1.7.

Compute a matrix $\mathbf{E}$ (by hand) that fully row reduces the augmented matrix $\mathbf{A}$ given in Equation (1)– that is, find $\mathbf{E}$ such that $\mathbf{EA}$ is a diagonal matrix with 1s along the diagonal. Show that this is true by multiplying out $\mathbf{EA}$. When an augmented matrix is in this final form, it will have the form

$$\mathbf{EA} = \left[\begin{array}{cccc|c} 1 & 0 & 0 & 0 & b_1 \\ 0 & 1 & 0 & 0 & b_2 \\ 0 & 0 & 1 & 0 & b_3 \\ 0 & 0 & 0 & 1 & b_4 \end{array}\right]$$

Once you have found the required elementary matrices,

i. use IPython to find the matrix **E**

ii. verify by hand that multiplying **E** and **A** gives you the identity matrix augmented with constants (as **EA** shown above).

Hint: As before, note that you can either ***apply a set of row operations to the same identity matrix*** *or* ***apply them to separate identity matrices and then multiply the matrices together****. Make sure, though, that you apply the row operations and multiply the matrices in the correct order.*

3. Mechanical Inverses

For each of the following matrices, state whether the inverse exists. If so, find the inverse, $\mathbf{A}^{-1}$. If not, show why no inverse exists. **Solve the inverses by hand. You may use IPython for parts (a)-(d) to visualize how the matrix A changes a vector.**

For parts (a)-(d), in addition to finding the inverse (if it exists), describe how the original matrix, **A**, changes a vector it's applied to. For example, if $\mathbf{A}\vec{b} = \vec{c}$, then **A** could scale $\vec{b}$ by 2 to get $\vec{c}$, or **A** could reflect $\vec{b}$ across the x axis to get $\vec{c}$, etc. *Hint:* It may help to plot a few examples to recognize the pattern.

(a) $\mathbf{A} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$

(b) **(PRACTICE)**
$\mathbf{A} = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix}$

(c) **(PRACTICE)**
$\mathbf{A} = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}$

(d) Hint: What does the result look like when you apply this matrix to $\begin{bmatrix} 1 \\ 0 \end{bmatrix}$ and $\begin{bmatrix} 0 \\ 1 \end{bmatrix}$? Draw it out. In the iPython notebook for this homework, we have provided you with a rotation function that may help to visualize what is happening.
$\mathbf{A} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$

(e) $\mathbf{A} = \begin{bmatrix} 1 & 1 \\ 2 & 0 \end{bmatrix}$

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

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

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

(i) **(PRACTICE)**

$$\mathbf{A} = \begin{bmatrix} 3 & 0 & -2 & 1 \\ 0 & 2 & 1 & 3 \\ 3 & 1 & 0 & 4 \\ 1 & 0 & 0 & 1 \end{bmatrix}$$

Hint 1: What do the linear (in)dependence of the rows and columns tell us about the invertibility of a matrix? Hint 2: We're reasonable people!

4. Proofs about Independence, Span, and Subspaces

In this problem, we will explore the connections between independence, span, and subspaces. We will start with a list of definitions. Using proofs, you will explore how these definitions interact with each other.

Recall the definition of linear independence: a set of vectors $(\vec{v}_1, ..., \vec{v}_n)$ is **linearly dependent** if one of the vectors can be written as a linear combination of the other vectors. In mathematical notation, this means that for some i, $\vec{v}_i = \sum_{j \neq i} a_j \vec{v}_j$. We saw that this definition is equivalent to an alternate definition: that there exists some linear combination $\sum_{i=1}^{n} a_i \vec{v}_i = 0$ where at least one a_i is non-zero.

A set of vectors is **linearly independent** if it is not dependent. In other words, the only time that the linear combination $\sum_{i=1}^{n} a_i \vec{v}_i$ is equal to 0 is when all the a_i are 0.

Also, recall the definition of span: the **span** of a set of vectors $(\vec{v}_1, ..., \vec{v}_n)$ is the set of all linear combinations of these vectors, or any vector that can be written $\sum_{i=1}^{n} a_i \vec{v}_i$ for some scalars a_i.

(a) First, suppose we have some set of linearly independent vectors $(\vec{v}_1, ..., \vec{v}_n)$, and we have an extra vector $\vec{u}$. Show that if $\vec{u}$ is in the span of $(\vec{v}_1, ..., \vec{v}_n)$, then the set $(\vec{v}_1, ..., \vec{v}_n, \vec{u})$ is dependent. Hint: This should be a simple application of the definition.

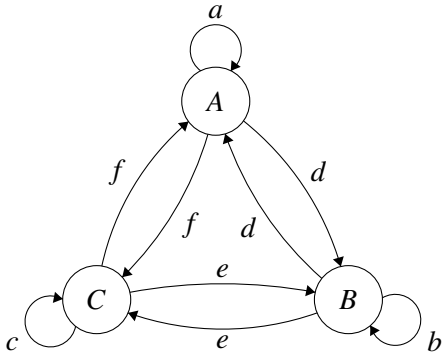
(b) Now, show that if $(\vec{v}_1, ..., \vec{v}_n)$ is independent but $(\vec{v}_1, ..., \vec{v}_n, \vec{u})$ is dependent, then $\vec{u}$ is in the span of $(\vec{v}_1, ..., \vec{v}_n)$.

(c) Recall that a **subspace** is a subset of vectors that contains the $\vec{0}$ vector and is closed under addition and scalar multiplication. In other words, if $\vec{u}$ and $\vec{v}$ are in the subspace, then $\vec{u} + \vec{v}$ is in the subspace. Also, if $\vec{u}$ is in the subspace, then $a\vec{u}$ is in the subspace for any scalar a.
Suppose we have a set of vectors $(\vec{v}_1, ..., \vec{v}_n)$. Show that the span of these vectors is a subspace.

(d) Consider the span of the set $(\vec{v}_1, ..., \vec{v}_n, \vec{u})$. Suppose $\vec{u}$ is in the span of $(\vec{v}_1, ..., \vec{v}_n)$. Then, show that any vector in $span(\vec{v}_1, ..., \vec{v}_n, \vec{u})$ is in $span(\vec{v}_1, ..., \vec{v}_n)$.

5. 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

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

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.

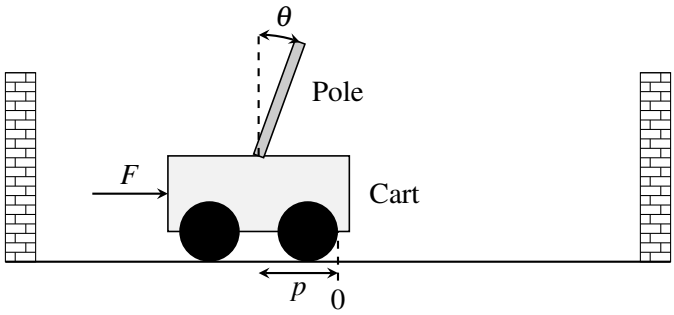
6. Segway Tours

Your friend has decided to start a new SF tour business, and you suggest he use segways.

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.

Your friend 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. He 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]

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

$$\vec{x}[n+1] = \mathbf{A}\vec{x}[n] + \vec{b}u[n], \tag{2}$$

where $\mathbf{A} \in \mathbb{R}^{4\times4}$ and $\vec{b} \in \mathbb{R}^{4\times1}$. 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 your friend's 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 $u[0]$. (*Hint:* This is easy.)

(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 $u[0],\ldots,u[N-1]$. (*Note: To obtain a compact expression, 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 $\vec{b}$:

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

[1] You might note that velocity and angular velocity are derivatives of position and angle respectively. Differential equations are used to describe continuous time systems, which you will learn more about in EE 16B. But even without these techniques, we can still approximate the solution to be a continuous time system by modeling it as a discrete time system where we take very small steps in time. We think about applying a force constantly for a given finite duration and we see how the system responds after that finite duration.

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

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 $u[0], u[1], \ldots$. The state vector $\vec{x}_f = \vec{0}$ corresponds to the cart-pole (or segway) being upright and stopped at the origin. (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 three parts of the problem. You may use the function we provided (`gauss_elim(matrix)`) to help you find the upper triangular form of matrices. An example of Gaussian Elimination using this code is provide in the iPython notebook. You may also use the function (`np.linalg.solve`) to solve the equations.

(d) Can you reach $\vec{x}_f$ in *two* time steps? (*Hint: Express* $\vec{x}[2] - \boldsymbol{A}^2\vec{x}[0]$ *in terms of the inputs* $u[0]$ *and* $u[1]$. *Then determine if the system of equations can be solved to obtain* $u[0]$ *and* $u[1]$. *If we obtain valid solutions for* $u[0]$ *and* $u[1]$*, then we can say we will reach* $\vec{x}_f$ *in two time steps.*)

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

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

(g) If you have found that you can get to the final state in 4 time steps, find the required correct control inputs using IPython and verify the answer by entering these control inputs into the *Plug in your controller* section of 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?

7. Audio File Matching

Each day a wide variety of quantities we interact with can be expressed as vectors. For example, an audio clip or a sound wave (continuous function in time) can be sampled at regular intervals to make a discrete sequence of values and represented in vector form.

This problem explores using inner products for measuring similarity between two sound signals represented as vectors. The same ideas here will be further developed in the third module of EE16A where we will learn about Locationing and GPS.

Let us consider a very simplified model for an audio signal; one that is composed of just two tones. One tone is represented by the value -1 and the other by the value $+1$. A vector of length n makes up the audio file.

(a) Say we want to compare two audio files of the same length n to decide how similar they are. First consider two vectors that are exactly identical $\vec{X}_1 = \begin{bmatrix}1 & 1 & \cdots & 1\end{bmatrix}^T$ and $\vec{X}_2 = \begin{bmatrix}1 & 1 & \cdots & 1\end{bmatrix}^T$. What is the inner product/dot product of these two vectors, i.e. $\vec{X}_1^T\vec{X}_2$? What if $\vec{X}_1 = \begin{bmatrix}1 & 1\cdots 1\end{bmatrix}^T$ and $\vec{X}_2 = \begin{bmatrix}1 & -1 & 1 & -1\cdots 1 & -1\end{bmatrix}^T$ (where the length of the vector is an even number)? For pairs of vectors of length n made of ± 1's, does a larger dot product imply that the vectors are more similar or less similar?

(b) Next suppose we want to search for a short audio clip in a longer one. We might want to do this for an application like *Shazam* to be able to identify a song from a signature tune. Consider the vector of length 8, $\vec{X} = \begin{bmatrix}1 & -1 & -1 & 1 & 1 & 1 & -1 & 1\end{bmatrix}^T$. Let us label the elements of $\vec{X}$ so that $\vec{X} = \begin{bmatrix}x_1 & x_2 & x_3 & x_4 & x_5 & x_6 & x_7 & x_8\end{bmatrix}^T$. Our goal is to find the short segment $\vec{Y} = \begin{bmatrix}1 & 1 & 1\end{bmatrix}^T$ in the longer vector (i.e. we want to find i, such that the sequence represented by $\begin{bmatrix}x_i & x_{i+1} & x_{i+2}\end{bmatrix}^T$ is the closest to $\vec{Y}$). Come up with an approach to do this. Can your approach be written as a matrix vector multiplication $\mathbf{A}\vec{x}$ where $\mathbf{A}$ is 6×8 and $\vec{x}$ is 8×1? Applying your technique, which i gives the best match for $\vec{Y} = \begin{bmatrix}1 & 1 & 1\end{bmatrix}^T$?

(c) **(PRACTICE)** Now suppose our vector was represented using integers and not just by 1 and -1. We want to find the subsequence of the longer vector that is most similar to that of a sample vector. Say we wanted to locate the sequence closest in direction to $\vec{Y} = \begin{bmatrix}1 & 2 & 3\end{bmatrix}^T$ within $\vec{X} = \begin{bmatrix}1 & 2 & 3 & 4 & 5 & 6 & 7 & 8\end{bmatrix}^T$. Can you explain why the approach in part (b) won't work? Consider using the norm/magnitude of the short segments (the norm/magnitude of a vector is defined as $\|\vec{z}\| = \sqrt{z_0^2 + z_1^2 + \ldots + z_n^2}$). How might you use this quantity to modify your approach for the new vectors to focus on the direction rather than the magnitude of the vectors?

(d) In the IPython notebook, `prob3.ipynb`, complete part 1.

(e) In the IPython notebook, `prob3.ipynb`, complete part 2.

8. 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?