Homework 01

Due: January 16, 9PM
Point total: 60

Instructions:

- Submit your PDF and/or .py file to Blackboard by the due date and time. Please do not zip your files together, as this interferes with Blackboard's preview functionality. Always show all your work, and for full credit, you must use the method that the problem instructs you to use (unless none is mentioned). Handwritten or typeset solutions are both acceptable, but unreadable submissions will be penalized. You may discuss problems with other students, but you may not write up solutions together, copy solutions from a common whiteboard, or otherwise share your written work or code. Do not use code or language that is copied from the Internet or other students; attribute the ideas *and* rephrase in your own words.

Problem 1 (15 points)

For each system of equations, write the corresponding augmented matrix. Then solve for the solution *using Gauss-Jordan elimination*, describing what you are doing with each step (for example, "row1 = row1 - 2*row2"). To avoid rewriting the same numbers over and over, you can perform multiple steps at a time as long as each step affects a different row. If there is no solution, say so; you can stop as soon as this fact is apparent from your work. If there are multiple solutions, give equations describing the solutions in terms of z.

i. $3x + 2y + z = 9$
$2x + 3y - 3z = -14$
$-x - y + z = 4$

ii. $3x + z = 0$
$x - y + z = -5$
$2x + y = 5$

iii. $w + 2x - y - z = 4$
$2w + x + 2y + z = 1$
$3w + 2y - z = -3$
$2w - 2x + 3y = -6$

Problem 2 (8 points)

For each system of equations in problem 1, determine whether the corresponding homogeneous matrix is singular or not, and explain how we know.

Problem 3 (10 points)

i. A particle at location (2,4) at time 0 is subject to a current that causes the following changes in position from timestep i (position (x_i, y_i)) to timestep $i+1$ (position (x_{i+1}, y_{i+1})):

$x_{i+1} = x_i - 0.2y_i$
$y_{i+1} = y_i + 0.2x_i$

(Notice that the order of the variables changed in the second equation.) Write the matrix that corresponds to this linear dynamical system, perform 3 matrix multiplications, and give the new location at time 3.

ii. Find a point in this system where a particle will stay still forever. Is this the only such point? Show some work to support your conclusion.

Problem 4 (12 points)

Find the angle between each pair of vectors in radians, using the formulas for the dot product and vector length. Give exact answers, not decimals.

i. $\begin{bmatrix}1\\1\end{bmatrix}, \begin{bmatrix}0\\1\end{bmatrix}$

ii. $\begin{bmatrix}1\\2\\1\end{bmatrix}, \begin{bmatrix}-1\\-1\\0\end{bmatrix}$

iii. $\begin{bmatrix}1\\1\\0\\0\end{bmatrix}, \begin{bmatrix}0\\0\\1\\1\end{bmatrix}$

iv. $\begin{bmatrix}1\\1\\-1\\-1\end{bmatrix}, \begin{bmatrix}-2\\3\\3\\-2\end{bmatrix}$

Problem 5 (15 points)

Later on in the semester, we will see that if you repeatedly multiply a vector by a matrix, the result in the long term may depend more on the matrix than the original vector. In this homework, you'll have a chance to see this behavior.

i. Write a Python function that will multiply any matrix A by any vector $\vec{b}$ n times, and report the results of running it for each combination of matrix and vector below (2*3 = 6 combinations) where $n = 100$.

You must use Python - part of the point is an introduction or refresher for Python, which we'll use more later. You can and should use numpy.matmul for the multiplication. Report values to two decimal places. Submit your code as an additional file when you submit the assignment.

Matrices:

$$A = \begin{bmatrix} 0.1 & 0.7 & 0.1 \\ 0.2 & 0.1 & 0.3 \\ 0.7 & 0.2 & 0.6 \end{bmatrix}, B = \begin{bmatrix} 0.2 & 0.3 & 0.5 \\ 0.4 & 0.4 & 0.2 \\ 0.4 & 0.3 & 0.3 \end{bmatrix}$$

Vectors:

$$\vec{c} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \vec{d} = \begin{bmatrix} 0.3 \\ 0.4 \\ 0.3 \end{bmatrix}, \vec{e} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

ii. Looking at the vector results above, do all three vectors result in the same final answer for each matrix? If not, are these final vectors pointing in the same direction for the same matrix?

Appendix 1: Python examples

Here are some examples that you may find useful in writing your first Python function. You can find instructions on how to set up Python on your machine online (we'll use the latest Python 3.x). I recommend installing from the python.org website even if you already have Python on your machine, both to get the latest version and to ensure that you'll be able to spawn windows for the `matplotlib` plotting utility later. You may need to install numpy separately (again, just follow instructions online).

If you run into trouble, you may want to have a glance at the separate *Python Pitfalls* handout, located in the Course Materials section of Blackboard.

Start up Python's interactive mode with `python3` (`python` starts version 2.x). In the following examples, things typed at a prompt are preceded by `">>>"`.

i. You can always just type expressions to be evaluated at the prompt:

```
>>> 1+1
2
```

ii. To import numpy and try out a single matrix multiplication $\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} \begin{bmatrix} 10 \\ 11 \\ 12 \end{bmatrix}$:

```
>>> import numpy as np
>>> A = [[1, 2, 3], [4, 5, 6], [7, 8, 9]]
>>> b = [10, 11, 12]
>>> np.matmul(A,b)
array([ 68, 167, 266])
```

Note that if you are writing code using numpy in a file, you must include "import numpy as np" at the top.

iii. To print "Hello, world!" 10 times:

```
for x in range(10):
        print("Hello, world!")
```

iv. To write a function that takes as an argument the number of times to print "Hello, world!":

```
def hello_lots(n):
    for i in range(n):
        print("Hello, world!")
```

v. To call the function defined in the previous example from a command line, if it is in a file called “hello.py”:

```
>>>import hello
>>>hello.hello_lots(5)
Hello, world!
Hello, world!
Hello, world!
Hello, world!
Hello, world!
```