CS2810 Mathematics of Data Models	Gold
Spring 2019	January 16, 2019

Homework 02

Due: January 23, 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 (12 points, 4 each)

In each case, find the matrix that performs the transformation described. (Note that you are looking for a *single* matrix, which you can find by multiplying out matrices that perform each individual operation. Be careful about which matrix goes first or last.) You can assume the direction of rotation is the one that is most convenient for our formulas.

i. In two dimensions: rotate 45 degrees about the origin, then swap the x and y coordinates.

ii. In three dimensions: rotate 30 degrees around the x axis, then 90 degrees around the z axis, then scale up by a factor of 2

iii. In three dimensions: rotate 60 degrees around the y-axis, then project onto the xy plane (zeroing the z coordinate)

Problem 2 (12 points, 4 each)

In each subproblem, you are given three vectors, $\vec{q}, \vec{r}$, and $\vec{s}$. Determine whether the three vectors are linearly independent. If they are not, find the coefficients a, b such that $a\vec{q} + b\vec{r} = s$. If they are linearly independent, find coordinates for the vector $\begin{bmatrix}1\\1\\1\end{bmatrix}$ using the three vectors $\vec{q}, \vec{r}$, and $\vec{s}$ as a basis.

i. $\vec{q} = \begin{bmatrix}1\\2\\3\end{bmatrix}, \vec{r} = \begin{bmatrix}3\\2\\1\end{bmatrix}, \vec{s} = \begin{bmatrix}10\\4\\-2\end{bmatrix}$

ii. $\vec{q} = \begin{bmatrix}1\\2\\7\end{bmatrix}, \vec{r} = \begin{bmatrix}6\\1\\5\end{bmatrix}, \vec{s} = \begin{bmatrix}3\\-2\\1\end{bmatrix}$

iii. $\vec{q} = \begin{bmatrix} \pi \\ 1 \\ 1 \end{bmatrix}, \vec{r} = \begin{bmatrix} -1 \\ \pi \\ 0 \end{bmatrix}, \vec{s} = \begin{bmatrix} -1 \\ 1 \\ 0 \end{bmatrix}$

Problem 3 (9 points, 3 each)

i. Is a vector that is not in the span of a set of vectors necessarily orthogonal to each of those vectors? Explain why or provide a counterexample.

ii. Does the span of n vectors always have n dimensions? Explain why or give a counterexample.

iii. Given a nonzero vector $\vec{v}$ in an n-dimensional space, what is the maximum number of (nonzero) vectors can we find that are both orthogonal to $\vec{v}$ and orthogonal to each other? (Don't count $\vec{v}$ itself.)

Problem 4 - 9 points (3 each)

Draw each of the following pairs of high-dimensional vectors by computing their lengths and the angle between them. (The plane of your homework page should be the same as the plane of their span. Their rotation doesn't matter. Your angles and lengths don't have to be exact on the page as long as we get the idea.)

i. $\begin{bmatrix} 1 \\ 2 \\ 3 \\ -6 \end{bmatrix}$ and $\begin{bmatrix} -1 \\ -1 \\ -1 \\ -1 \end{bmatrix}$

ii. $\begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$ and $\begin{bmatrix} -1 \\ -1 \\ -1 \\ -1 \\ -1 \end{bmatrix}$

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

Problem 5 (18 points)

For this part, download `transform.py`, and also install matplotlib (remember to install it for Python3!). Matplotlib is the main Python module that we will use for graphical output in this course.

I've borrowed an example from the matplotlib tutorial page that draws a spiral. Your job will be to rotate this spiral. The spiral consists of three lists of coordinates: x coordinates, y coordinates, and z coordinates. (Thus, for example, (x[0], y[0], z[0]) is a point on the spiral.) These points are

generated within the code, so you can see the spiral just by passing in any arguments at all to the plotspiral method. (Notice how you can click and drag to rotate!)

These lists will be handed to your function `transform()` along with 4 arguments: a rotation around the x axis, a rotation around the y axis, a rotation around the z axis, and a scaling factor. (The rotations should all be in radians.) Your job is to perform these transformations on the points of the spiral, returning new lists of x, y, and z coordinates. The operations should happen in the order of the arguments: the x-axis rotation happens first, then the y-axis, then the z-axis, then scaling.

We'd like the transformation to happen in a *very specific way*, so pay close attention to the following instructions.

- When you create a rotation matrix, you should calculate the elements of the matrix yourself; don't invoke any numpy methods or other methods that just hand you a rotation matrix. Use the formulas from lecture. (For the scaling, you can just multiply the matrix by a scalar.)
- You should multiply to create *one matrix* to perform the whole transformation, just like in problem 1 of this assignment. For the purpose of the next instruction, we'll call this T.
- For *three extra credit points*, transform *all* the points at once with *one matrix multiplication*: construct a matrix P such that the columns of $P' = TP$ are the points you want. (Think about how vector multiplication and matrix multiplication each work.) This approach has the advantage of taking less code, and could also be optimized by a GPU or a smart compiler. (If you don't want to go for the extra credit, you can iterate through the point coordinates with a "for" loop.)

Take a snapshot of what it looks like to rotate by $\pi/2$ for the x-axis then the y-axis (no need to scale) and include this snapshot in your PDF. (Don't play with the view before taking the snapshot.) You should also submit `transform.py`.