PSYCH-GA.2211/NEURL-GA.2201 – Fall 2017
Mathematical Tools for Cognitive and Neural Science

Homework 1

Due: 26 Sep 2017
(late homeworks penalized 10% per day)

See the course web site for submission details. Please: don't wait until the day before the due date... start *now*!

1. **Testing for (non)linearity.** Suppose, for each of the systems below, you observe the indicated input/output pairs of vectors (or scalars). Determine whether each system could possibly be a *linear* system. If so, provide an example of a matrix that is consistent with the observed input/output pairs, and state whether you think that matrix is unique (i.e., the only matrix that is consistent with the observations). If not, explain why.

System 1:	[2, 4]	$\longrightarrow$	[-6, -6]
	[-1, -2]	$\longrightarrow$	[3, 3]
System 2:	1	$\longrightarrow$	[1, 4]
	2	$\longrightarrow$	[2, 6]
System 3:	[1, 1]	$\longrightarrow$	[4, -1]
	[1, -0.5]	$\longrightarrow$	[1, 3]
	[3, 0]	$\longrightarrow$	[6, 2]
System 4:	[2, 4]	$\longrightarrow$	0
	[-2, 2]	$\longrightarrow$	3
System 5:	0	$\longrightarrow$	[1, 2]

2. **Geometry of linear transformations**

   (a) Write a function `plotVec2` that takes as an argument a matrix of height 2, and plots each column vector from this matrix on 2-dimensional axes. It should check that the matrix argument has height two, signaling an error if not. Vectors should be plotted as a line from the origin to the vector position, using circle or other symbol to denote the "head" (see help for function 'plot'). It should also draw the x and y axes, extending from -1 to 1. The two axes should be equal size, so that horizontal units are equal to vertical units (read the help for the function 'axis').

   (b) Write a second function `vecLenAngle` that takes two vectors as arguments and returns the length (magnitude, or Euclidean-norm, not *dimensionality*) of each vector, as well as the angle between them. Decide how you would like to hanldle cases when one (or both) vectors have zero length.

(c) Generate a random 2x2 matrix, and decompose it using the SVD. Now examine the action of this sequence of transformations (USV^T) on the two "standard basis" vectors, $\{\hat{e}_1, \hat{e}_2\}$. Specifically, use `vecLenAngle` to examine the lengths and angle between two basis vectors $\hat{e}_n$, the two vectors $V^T\hat{e}_n$, the two vectors $SV^T\hat{e}_n$, and the two vectors $USV^T\hat{e}_n$. Do these values change, and if so, after which transformation? Verify this is consistent with their visual appearance by plotting each pair using `plotVec2`.

(d) Generate a matrix P with 65 columns containing 2-dimensional unit-vectors $\hat{u}_n = [cos(\theta_n); sin(\theta_n)]$, and $\theta_n = 2\pi n/64, n = 0, 1, \ldots, 64$. [Note: Don't use a `for` loop! Create a vector containing the values of θ_n.] Plot a single blue curve through these points, and a red star (asterisk) at the location of the first point. As in the previous problem, apply the SVD transformations one at a time to full set of points (again, don't use a for loop!), plot them, and describe what geometric changes you see (and why).

3. **A simple linear system.** Suppose you have a retinal neuron whose response is a weighted sum of the intensities of light that land on a local collection of photoreceptors (note that these intensities are positive values). The weight vector is $[1, 3, 4, 2, 1]$. (a) What unit-length stimulus vector elicits the largest response in the neuron? Explain how you arrived at your answer. (b) Now generate a unit-length stimulus vector that elicits a zero response in the neuron (and verify that this is true). Is this a physically realizable stimulus? If not, is there *any* realizable stimulus (not necessarily unit length) that would elicit a zero response in the neuron? If not, explain why and if so, give an example.

4. **Gram-Schmidt.** A classic method for constructing an orthonormal basis is known as *Gram-Schmidt orthogonalization*. First, one generates an arbitrary unit vector (e.g., by normalizing a vector created with `randn`). Each subsequent basis vector is created by generating another arbitrary vector, subtracting off the projections of that vector along each of the previously created basis vectors, and normalizing the remaining vector. You should draw (by hand) the picture in 2D, assuming you have one unit vector, and you're creating the second, to make sure you understand the geometry of this construction!

   Write a MATLAB function `gramSchmidt` that takes a single argument, N, specifying the dimensionality of the basis. It should then generate an $N \times N$ matrix whose columns contain a set of orthogonal normalized unit vectors. Try your function for $N = 3$, and plot the basis vectors (you can use MATLAB's `rotate3d` to interactively examine these). Check your function numerically by calling it for an N larger than 3 and verifying that the resulting matrix is orthonormal. Extra credit: make your function *recursive* – instead of using a `for` loop, have the function call itself. To do this, you'll probably need to write two functions (a main function that initializes the problem, and a helper function that then calls itself to generate each vector).

5. **Null and Range spaces.** Load the file `mtxExamples.mat` into your MATLAB world. You'll find a set of matrices named `mtxN`, with $N = 1, 2....$ For each matrix, use the SVD to: (a) determine if there are non-trivial (i.e., not the zero vector!) vectors in the input space that the matrix maps to zero (i.e., determine if there's a nullspace), and if so, write a MATLAB expression that generates a random example of such a vector, and verify that the matrix maps it to the zero vector; (b) write a MATLAB expression to generate a random vector y that lies in the range space of the matrix, and then verify that it's in the range space by solving for an input vector, x, such that $Mx = y$.