Math 350 Final Exam, Spr 2025

Instructions: This is a take home final exam. You may use your notes and anything on our class website to assist you, **but you may not use anything else on the internet**. You may also use the help files available in Matlab or Python. **You may work in groups of two.** If you have questions, you may ask me (I may not be able to answer fully, but it is better to ask me than not to know).

For each problem below, upload the script or function file you created, together with a screen shot of the output. You might create a document that has both, and that creates a good summary of your work.

Important: I want to see your code **and** what your code produced.

Due: The deadline will be at 11:59PM on Monday, May 19th. Grades for seniors are due to the Registrar's office by Wednesday morning, so **no late solutions will be accepted- start early!**

Problem 1: Linear System

We want to analyze the matrix equation $A\mathbf{x} = \mathbf{b}$, where A and $\mathbf{b}$ are provided in the file `FinalExamData01.mat`. In this case, A is a "wide" matrix, so we run into the problem of not only having an exact solution, but since we have free variables, any least squares solution we get should have an infinite number of possibilities.

You may answer the questions below in your Matlab or Python script file. Turn in the script file and give a screen shot of the output (or create a document with both).

1. Using the SVD, compute a least squares solution, $\hat{x}$ by creating the pseudoinverse of A. You're going to need to decide on the rank of the matrix.
2. Show (using the SVD and an appropriate computation) that $\hat{x}$ is in the row space of A. Similarly, show that $A\hat{x}$ is in the column space of A.
3. We said that there were an infinite number of possible solutions- Use the SVD to come up with two more (distinct, linearly independent) solutions, and show that they're correct by computing $A\hat{x}$ for each of your answers.

Problem 2: An RBF classifier on mushroom data

The data will be measurements taken from mushrooms. We have 22 measurements taken from 4062 mushrooms, and we want to classify them as edible or poisonous.

The dataset `mushrooms.mat` is on the class website, and when you load the data, you'll see two arrays: X is 22×4062 (so each "point" is a column, and there are 4062 of them), and matrix T is 2×4062. The column is either $[1, 0]^T$ or $[0, 1]^T$ for poisonous/edible (it doesn't matter which is which until we actually have to taste one).

The RBF is built using $k-$means with 120 centers and try the thin plate spline for the transfer function.

Once trained, be sure and output the confusion matrix.

This problem is similar to the RBF homework. If you still have questions about the code for that, be sure to ask me about it.

Problem 3: A Feedforward Neural Network

This is similar to the homework problem on feedforward nets where we built the network to classify the iris data. If you still have questions about that homework, be sure to ask me about it.

We'll be working with the diabetes data from the RBF homework.

We'll have data from 768 patients, and from each patient, we'll have 8 measurements. We want to predict if the patient has diabetes or not.

If you download `diabetes1.mat` and load it into Matlab, you will see two matrices: Matrix P will be 768×8 and matrix T will be 768×2.

We're building the neural network from scratch (use your homework as a starting template). Here are the details:

- Leave the split for training and testing at 70-30.
- You might drop the epochs to something like 10 until you're sure your code runs, and then crank it up to around 50 or 100. (If you use Octave-online, be prepared to click the "extra time" button several times).
- You might choose to change the learning rate α, depending on what happens in training (for example, if your weights and biases blow up).
- Lastly, there are two important changes we want to make:
    1. Add an extra hidden layer, so that the network is $8 - 15 - 15 - 2$.
    2. Change the activation function to the ReLu function rather than the sigmoidal.
- Be sure to turn in the code you used and the result of the training (showing your final confusion matrices).