Lecture 2 Problem Set, MATLAB

Remember to use `help, doc,` and `lookfor` to help you figure out parts of Matlab that you're unfamiliar with.

Vectors and matrices

1. Use the Matlab function `rand` to write an n-sided "dice" function. That is, write a function that outputs an integer from 1 to n with equal probability. Allow the user to specify the value n.

2. Expand on your dice function to add a second input parameter that allows the user to specify the number of rolls of the dice to be returned. The rolls should be returned as a vector. For example, if you ask for 6 rolls of a 3-sided dice, you might get

   >> dice(3, 6)

   ans = [1 3 2 2 1 3]

   Matlab has the function `randi` that does this, but you should write your function using `rand` and one of the rounding functions (`round`, `ceil`, `floor`).

3. Use the function you just wrote to roll your Matlab dice K times, where K is some large number. Use `hist` to make a histogram the number of times you got dice=1, dice=2, dice=3, ... , dice=n. If the distribution is uniform, the fraction of rolls that came out in each of these bins should be 1/n. Are you close to that?

4. To assess the answer to the previous question quentitatively, compute E, the mean squared difference between the expected fraction (1/n) and the value you obtained, i.e., compute the mean, over all the histogram bins, of (1/n - (bin_height/K)).^2. To do this, you can use the `mean` function, as well as the fact that Matlab allows arithmetic operations on vectors. (For example, if vec = [1 2 3], vec+1 = [2 3 4].)

5. Compute E for many different Ks, and make a plot of E as a function of K. What do you observe?

6. Write a script that makes an N by N matrix, containing a checkerboard of 1s and 0 where each of the squares that form the checkerboard is of size 5x5 entries. That is, if N=2, the final result should be 10x10, in 5x5 blocks of all 1s or all 0s. Use `imagesc` to display your matrix.

Strings

7. Download the text file provided on the wiki, and unzip it into your working directory. You should find a file called “joyce.mat”. Within Matlab, load this file into your workspace by typing

    >> load joyce

   This will create a variable called “str” that contains some text. Define a new string called vowels,

    >> vowels = ‘aeiou’

   Using a loop through the elements of your “vowels” string, compute how many times each vowel appears in the text file (i.e., how many ‘a’, how many ‘e’, how many ‘i’, etc.).

8. Write a “first_sentence.m” function that given a string, extracts and returns the first sentence of the string (i.e., all the characters up to and including the first period; if the string contains no period, then return the original string).

9. Write a function “replacer.m” that takes in a variable representing a string, and replaces every letter “e” with an “X”, and then returns the new string. Use your first_sentence.m to test your new “replacer.m” function with the first sentence of the text. Then use replacer.m on the entire text.

10. Write a function “replacerD.m” that takes in a variable representing, a string, and then replaces every letter “e” with an “X”, **but only for those “e”s that are followed by a “d”**, and returns the new string. Once again, test it on the first sentence of the text, and then run it on the entire text.

11. We’re going to examine the text you loaded from the “joyce.mat” file again. Expand your function from problem 1 so that now you count the frequency of occurrence of every letter in the text file. To help you do this, the unique.m function will be useful. This function takes a vector (which may be a vector of letters, i.e., a string), and returns a unique list of each of the values in the vector. For example,

    >> unique(‘open sesame’)

        ans = ‘aemnops’