MAE 8 - Winter 2015
Homework 3

Instructions: Follow the homework solution template. Put all answers in a MATLAB script named **hw3.m**. For this homework, you will need to submit multiple files. Create a zip archive named **hw3.zip**. The zip archive should include the following files: **hw3.m**, **figure1.png**, **figure2.png**, **figure3.png**, **rainfall.dat**, **newrainfall.dat**, and **interest.m**. Submit **hw3.zip** through TED before 9 PM on 01/29/2015. Use double precision unless otherwise stated.

Problem 1: Perform the following exercises to compute eigenvalues and eigenvectors of a matrix.

(a) Use functions **ones** and **eye** to to create a 3 x 3 matrix **A** with each elements has a value of 0.5 except for the diagonal where the value is 2.5. Set **p1a=A**.

(b) Use function **det** to compute the determinant of **A** and store the answer in **p1b**.

(c) Use function **eig** to compute the eigenvalues of **A** and store them in a column vector **p1c**.

(d) Compute the eigenvectors of **A** and store them in matrix **p1d** where each column is a different eigenvector.

Problem 2: The built-in function **clock** returns a row vector that contains 6 elements: the first three are the current date (year, month, day) and the last three represent the current time in hours (24 hour clock), minutes, and seconds. The seconds is a real number, but all others are integers. Use **sprintf** to accomplish the following formatting exercises.

(a) Get the current time and store it in **p2a**. The current time should be the time when the grader calls your script.

(b) Using the format 'MM:DD:YYYY', write the current date to string **p2b**. Here, MM, DD, and YYYY corresponds to month, day, and year, respectively.

(c) Using the format 'HH:MM:SS.SS', write the current time to string **p2c**. Here, HH, MM, and SS corresponds to hour, minute and second, respectively.

(d) Remove the last 3 digits from the string in part (c) so that the format is now 'HH:MM:SS'. Put the answer into string **p2d**.

(e) Combine the strings in part (b) and part(d) together separated by a single space. Put the answer in string **p2e**.

Problem 3: Create the following matrix **C**:

$$\begin{bmatrix} \pi & \pi/2 & \pi/3 \\ \pi/2 & \pi & \pi/4 \\ \pi/3 & \pi/4 & \pi \end{bmatrix}$$

(a) Write the first column of matrix **C** into string **p3a** using %f with 4 spaces including 2 decimal points for each element. The string must show a column vector.

(b) Write the first row of matrix **C** into string **p3b** using %f with 4 spaces including 2 decimal points for each element. The string must show a row vector.

(c) Write matrix **C** into string **p3c** using %f with 6 spaces including 4 decimal places.

(d) Write matrix **C** into string **p3d** using %e with 6 spaces including 4 decimal places.

Problem 4: Download the file **rainfall.dat**. The file includes the San Diego rainfall data from 1850 to 2012. The data is comma delimited and the columns represent the following: Year, Jan, Feb, Mar, Apr, May, Jun, Jul, Aug, Sep, Oct, Nov, Dec, YearTotal, NextJan, NextFeb, NextMar, NextApr, NextMay, NextJun, SeasonTotal. The file is structured in this manner so that rainfall data can be looked at either by calendar year or by season (July-June). Load this data file into MATLAB.

(a) Extract the year into row vector **p4a**.

(b) Extract the total rainfall per year into row vector **p4b**.

(c) Use function **mean** to compute the average rainfall over all years. Create a row vector **p4c** with the same length as the vector in part (a) but each element has the average rainfall.

(d) Find the maximum and minimum rainfall over the year. Put the answers in a 2-element row vector **p4d** where the first element is the maximum and the second is the minimum.

(e) Find the years when the maximum and minimum rainfall occur. Put the answers a 2-element row vector **p4e** where the first element is the year with the maximum rainfall and the second is the year with the minimum rainfall.

(f) Create figure 1. Save the finished figure as **figure1.png** and set **p4f='See figure 1'**. The figure should include the following:

- A line plot showing the total rainfall per year, use black solid line.
- A line plot showing the average rainfall over the years, use blue solid line.
- The maximum rainfall, use red square marker.
- The minimum rainfall, use green diamond marker.
- Give the figure a title, label the axes and create a descriptive legend.
- Make the axes fit to the data range.

(g) Extract the rainfall for the year 2012 (from Jan to Dec) into row vector **p4g**.

(h) Create row vector **p4h** to indicate the month from Jan (1) to Dec (12).

(i) Create figure 2 with a bar graph showing the monthly rainfall for the year 2012. Save the finished figure as **figure2.png**. The figure should include a title and labels for the axes. Set **p4i='See figure 2'**.

(j) Extract the rainfall data from year 2000 to year 2010 (all columns) into matrix **p4j** and save the matrix into a new file named **newrainfall.dat**.

(k) Write the matrix in part (j) into string **p4k** using the same format as in the file **rainfall.dat**.

Problem 5: Simple interest only pays interest on the initial investment. A better investment strategy is to add interest paid to the principle investment, so that going forward interest is also paid on the interest. This is called compound interest. The interest can either be compounded at discrete times during the year, or continuously. The formulas for the interest earned on an initial principle **Po**, given an interest rate **r**, compounding frequency per year **n**, and duration **t**, are:

Simple Interest: $P(t) = Po\,(1 + t * r)$

Periodic Compounding: $P(t) = Po\,(1 + r/n)^{nt}$

Continuous Compounding: $P(t) = Po\,e^{rt}$

Write a function that will receive input arguments for **Po**, **r**, **n**, **t**, and a return the total worth of the investment as **psimple pperiodic pcontinuous**. The function should be capable of taking the argument **t** as a vector and returning the total worth and interest paid as row vectors. The function should be called **interest** and saved as **interest.m**. This file is to be turned in with your script.

(a) Your function should include a description so that the command **help interest** would tell the user what the function does and defines the input and output variables. Set **p5a=evalc('help interest')**.

(b-d) Create row vector **time=[1:100]** and call your function with an initial principle Po=$1,000, an interest rate of 5%, a compounding frequency of 4 periods/year. The output here should be 3 vectors **p5b**, **p5c**, and **p5d** corresponding to **psimple**, **pperiodic**, and **pcontinuous**, respectively.

(e) Create figure 3. Plot the returns in part (b-d) (normalized by the initial investment) versus time to compare the investments. Use different line colors for each type of returns. Give title and legend to the figure and label the axes. Save the finished figure as **figure3.png** and set **p5e='See figure 3'**.