CS1132 Fall 2013 Assignment 1b due 9/17 11:59pm

> *Adhere to the Code of Academic Integrity.* You may discuss background issues and general strategies with others and seek help from course staff, but the implementations that you submit must be your own. In particular, you may discuss general ideas with others but you may not work out the detailed solutions with others. It is never OK for you to see or hear another student's code and it is never OK to copy code from published/Internet sources. If you feel that you cannot complete the assignment on you own, seek help from the course staff.

When submitting your assignment, follow the instructions summarized in Section 4 of this document.

Do not use the `break` or `continue` statement in any homework or test in CS1132.

1 Two Dimensional Matrix

Suppose $f(x,y) = exp^{-(x^2+y^2)}$ and we want to compute a matrix F size $N_1 \times N_2$ with the property that

$$f_{ij} = e^{-(x^2+y^2)}$$

where $x_i = \dfrac{i-1}{N_1-1}$ and $y_j = \dfrac{j-1}{N_2-1}$. i and j are the row and column index.
Implement the following function: `CreateFunctionMatrix`.

```
function F = CreateFunctionMatrix(N1,N2)
% This function creates a matrix and each entry is f(x,y) = exp^{-(x^2+y^2)}
% Input: size of the matrix F is N1 X N2
% Output: Matrix F that satisfies the above requirement.
```

2 Plot Function Matrix

Create a script called `plotMyFunctionMatrix.m` which plots the function in problem 1. First make use of the function $F = CreateFunctionMatrix(N_1, N_2)$, where $N_1 = 4, N_2 = 20$. Plot $f_(x,y)$ for several values of x where x goes from $\dfrac{1-1}{N_1-1}, \dfrac{2-1}{N_1-1}, \cdots, \dfrac{4-1}{N_1-1}$. Therefore, there are four $x-y$ plots on the same graph and each plot represents a fixed value of x. Choose different colors or line types for each case. Use `legend` to indicate what each line represents and add a `title` to the final figure.

3 Toeplitz Matrix

Toeplitz matrix or diagonal-constant matrix, named after Otto Toeplitz, is a matrix in which each descending diagonal from left to right is constant[1]. This matrix is very helpful in solving a Toeplitz system in the form of $Ax = b$ where A is a Toeplitz matrix. The system has only $2n-1$ degrees

[1] Cited from Wikipedia

of freedom rather than n^2 and hence the solution of this system is much easier. For instance, the following matrix is a Toeplitz matrix:

$$\begin{bmatrix} a_0 & a_{-1} & a_{-2} & \cdots & \cdots & a_{-n+1} \\ a_1 & a_0 & a_{-1} & \ddots & \ddots & \vdots \\ a_2 & a_1 & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & a_{-1} & a_{-2} \\ \vdots & \ddots & \ddots & a_1 & a_0 & a_{-1} \\ a_{n-1} & \cdots & \cdots & a_2 & a_1 & a_0 \end{bmatrix}$$

Write a function called `CreateToeplitz` that sets up a Toeplitz matrix with input parameter `row` and `col`. For example, the following matrix is an example of Toeplitz matrix. the input parameters for this matrix is column vector `[a;f;g;h;i]` and row vector `[a,b,c,d,e]`

$$\begin{bmatrix} a & b & c & d & e \\ f & a & b & c & d \\ g & f & a & b & c \\ h & g & f & a & b \\ i & h & g & f & a \end{bmatrix}$$

Implement the following function:

```
function T = CreateToeplitz(row,col)
% This function creates a Toeplitz matrix
% Input (row,col):  r: the row vector ; col: the column vector.
% Output T: the final Toeplitz matrix created according to row and col.
```

4 Merging Two Airplane Seats Assignment

Two airplanes have the same departure city and destination city. When the smaller airplane is under booked, airlines decide to move all the passengers from the smaller aircraft to the bigger one (given that there are enough seats). The seat chart for the aircraft is represented as a 2D binary matrix. For example $seatChart(1,2) = 1$ represents row 1, seat 2 is available. When merging, we would like to keep the seat assignment for the the bigger aircraft the same and reassign seats for the passengers from the smaller aircraft (If two airplanes are the same size, keep the one that is more occupied the same and reassign seats for the other one). When reassigning seats, the goal is to let the passengers sit as close to their original seats as possible. To compute distances between seats, use the following definition:

$$Score = |Row_{old} - Row_{new}| + |Seat_{old} - Seat_{new}| \tag{1}$$

We would like to minimize the score for each customer. Consider passenger sequentially in the following order: $seatChart(1,1), seatChart(1,2) \cdots seatChart(1,n), seatChart(2,1) \cdots$. Break ties of scores by choosing the seat with the smaller row number. If the row number is the same, choose the one with the smaller seat number. For output, please return the new seat assignment chart and also the `changedAssignment` matrix which contains all the seats that were changed due to the merge. For example a row in `changedAssignment` $[2,3,4,5]$ means the person sitting on the smaller airplane at row 2, seat 3 moved to row 4 seat 5.
Implement the following function:

```
function [finalSeatChart,changedAssignment] = MergeAirplanes(seatChart1, seatChart2);
% This function merges the smaller underplane to the bigger airplane.
% Input seatChart1 is from airplanes one; seatChart2 is from airplane two.
% output finalSeatChart the merged final seatchart from the bigger aircraft
% changedAssignment is a matrix that each row shows the original seat and the new
% seat assignment
```

5 Self-check list

The following is a list of the minimum *necessary* criteria that your assignment must meet in order to be considered *satisfactory*. Failure to satisfy any of these conditions will result in an immediate request to resubmit your assignment. Save yourself and the graders time and effort by going over it before submitting your assignment for the first time. Although all these criteria are necessary, meeting all of them might still not be *sufficient* to consider your submission satisfactory. We cannot list everything that could be possibly wrong with any particular assignment!

- Δ Comment your code! Make sure your functions are properly commented, regarding function purpose and input/output parameters.
- Δ Suppress all unnecessary output by placing semicolons (;) appropriately. At the same time, make sure that all output that your program intentionally produces is formatted in a user-friendly way.
- Δ Make sure your functions names are *exactly* the ones we have specified, *including* case.
- Δ Check that the number and order of input and output parameters for each of the functions match exactly the specifications we have given.
- Δ Test each one of your functions independently, whenever possible, or write short scripts to test them.
- Δ Check that your scripts do not crash (i.e., end unexpectedly with an error message) or run into infinite loops. Check your script several times in a row. Before each test run, type the commands `clear all; close all;` to delete all variables in the workspace and close all figure windows.

6 Submission instructions

1. Upload files `CreateFunctionMatrix.m`, `CreateToeplitz.m`, `MergeAirplanes.m`, , and `plotMyFunctionMatrix.m` to CMS in the submission area corresponding to Assignment 1a in CMS before the deadline. Late submission is accepted up to 24 hours after the deadline with a 10% penalty.
2. *After grading:* If you resubmit the assignment, upload your corrected files and *be sure to select the* ***Regrade Request*** *option.*