Lab 10 – Matrix Algebra

1 Matrix algebra and Handling Matrices

1.1 Inverting a matrix typed by the user
Write an interactive Matlab program that takes a matrix typed in by the user, then displays the inverse of the given matrix.

Your program may assume that the given matrix always has an inverse.

Here is an example of how the program should look like when it runs:

```
Enter a matrix: [7 3 6; 1 2 4; 6 5 1]
0.1818   -0.2727   -0.0000
-0.2323    0.2929    0.2222
0.0707    0.1717   -0.1111
```

Whatever is in **black** above is what the program should print out, and whatever is in **red** is what the user should type. But when you run your program, everything will appear black.

The above is just an example. The user should be able to type in any matrix that can be inverted, and your code should still work. Use lecture 8-2 to help you.

1.2 Squaring a matrix and squaring each element in a matrix
Write an interactive Matlab program that takes a matrix typed in by the user (call it `matrix`), then displays `matrix^2` and `matrix.^2`. Make sure to write a nice, detailed comment describing the differences between using the dot and not using the dot.

But if `matrix` is not a square matrix, then `matrix^2` is undefined, so you should print “**Not a square matrix**”.

Here is an example of how your program should run:

```
Enter a matrix: [7 3 6; 1 2 4; 6 5 1]
88    57    60
33    27    18
53    33    57
49     9    36
 1     4    16
36    25    1
```
Here is another example of how your program should run:

Enter a matrix: [3 4 5]
Not a square matrix
9  16  25

1.3 Replacing some elements in a matrix
Write an interactive Matlab program that takes a matrix typed in by the user, then displays a matrix with the same values, except that every value less than 5 has been replaced by 0.

Here is an example of how your program should run:

Enter a matrix: [9.8271 1.0892 0.9382; 3.4599 8.3829 9.3823]
9.8271         0         0
0    8.3829    9.3823

1.4 Find minimum
Write an interactive Matlab program that takes a matrix typed in by the user, then displays the smallest value in the matrix.

Do not use the inbuilt Matlab function min.

Here is an example of how your program should run:

Enter a matrix: [4 5 6; 10 9 8; 6 2 8]
2

1.5 Create mirror image
Write an interactive Matlab program that takes a matrix typed in by the user, then displays the mirror image of the matrix along the vertical axis. That is, the 1st column in the given matrix now becomes the last column in the displayed matrix, the 2nd column in the given matrix now becomes the 2nd last column in the displayed matrix, etc.

Do not use any inbuilt functions in Matlab, except size.

Here is an example of how your program should run:

Enter a matrix: [1 2 3 4; 3 2 5 8; 1 7 4 0]
4  3  2  1
8  5  2  3
0  4  7  1
2 Handling Matrices

2.1 Create a special matrix

This Q was recently used in the Final Exam

Write a function called createMatrix which takes 2 parameters, and returns a square matrix, where:

- The 1st parameter is the number of rows and also the number of columns in the matrix to be returned
- The 2nd parameter is the value that should be found in the top-right half of the matrix, including the main diagonal from the top left to the bottom right
- All other elements in the matrix should contain the value which is the negative of the 2nd parameter

For example, if your function was given the values 5 and 3 (in this exact order), then your function should return the following matrix:

```
3  3  3  3  3
-3  3  3  3  3
-3 -3  3  3  3
-3 -3 -3  3  3
-3 -3 -3 -3  3
```

Your function should return an empty matrix if the 1st parameter is non-positive.

You may assume that you will always be given integer parameters.

3 Solving Linear Equations

3.1 Solving systems of linear equations

Write a Matlab program to solve the system of linear equations:

\[
\begin{align*}
2x + 5y + 7z &= 9 \\
3x + 2y + 3z &= 2 \\
x + 3y + 2z &= 5
\end{align*}
\]

a) by computing the inverse of a matrix. Check that the solution is correct (lecture 10-2)
b) by using the backslash operator “\". Check that the solution is the same as in a).
c) Write code to display an appropriate message depending on whether there exists a unique solution or not

(This Q is based on Exercise 3, Chapter 1 from “Engineering Computation with Matlab”, D. Smith, 2008)
4 Linear Regression

4.1 Temperature scales with linear algebra

The relationship between the Celsius and Fahrenheit temperature scales is linear and is given by the equation $C = sF + t$, where $C$ is the temperature in degrees Celsius and $F$ is the temperature in degrees Fahrenheit.

The line described by the equation above must pass through: (1) the melting point of water which has coordinates $F=32$ and $C=0$ and (2) the boiling point of water which has coordinates $F=212$ and $C=100$.

Use Matlab to determine the values of the coefficients $s$ and $t$ by solving a system of linear equations with matrix algebra.

(This Q is based on an exercise from "Matlab for Engineers", A. Biran and M. Breiner, 1999)

In the Final Exam, you will be assessed on matrix algebra and the various operations that can be performed on matrices in lectures 10-1, 10-2, and also lecture 3-1.

For example, you need to know how to perform cross-product multiplication by hand (using just pen and paper, not using Excel or Matlab), as shown in lecture 3-1 and lab 3. So make sure you are familiar with all these operations.

The only operation that you do NOT need to know how to do by hand is finding the inverse of a matrix.

5 Matrix Q from Final Exam

5.1 Find and explain what is incorrect

This question is based on an actual question that was recently used in the Final Exam

Download deleteRowsColsWithZeroBad1.m from the course website.

This code is attempting to do Lab 7, Exercise 6-1, which was a question that was recently used in the Final Exam. However, it does not work correctly.

Find out what is wrong, and explain the main reason why it’s wrong. There are actually many reasons why, but just focus on the main reason for now.

Many students did this in the final exam, and therefore lost a lot of marks (it was worth 16 marks, or 8% of the final result)!
5.2 Find and explain what is incorrect
This question is based on an actual question that was recently used in the Final Exam

Download deleteRowsColsWithZeroBad2.m from the course website.

This code fixes the main reason the code from Exercise 5-1 was not working. However, it still does not work correctly.

Find out what is wrong, and explain why it’s wrong.

There are actually at least 3 reasons why, but one of these reasons is very subtle (we will try to find it in Exercise 5-3), so just try to find the 2 main reasons for now.

Remember, debugging is where you spend 80% of your time. Therefore, you should take these kinds of questions especially seriously.

5.3 Find and explain what is incorrect
This question is based on an actual question that was recently used in the Final Exam

Download deleteRowsColsWithZeroBad3.m from the course website.

This code fixes the 2 main reasons the code from Exercise 5-2 was not working. However, it still does not work correctly.

Find out what is wrong, and explain why it’s wrong.

5.4 Find and explain what is incorrect
This question is based on an actual question that was recently used in the Final Exam

Consider Lab 7, Exercise 6-2, which described 3 different ways to do the above Final Exam Q. After this week’s lectures, you should now know a few more ways. Use all these methods, each in a separate file:

Method 4
- Make 2 arrays, containing the row and column numbers of the 2-D matrix that contain zeroes (i.e. the arrays contain integers, not boolean values)
- Sort the above 2 arrays, using the sort function, which takes a 1-D array and returns a new 1-D array which has the same elements as the given array, except that all the elements are sorted.
- Use the above 2 arrays and delete the rows and columns. Hint: You don’t even need to know whether the sort function sorts in ascending or descending order.
Method 5

- Write an auxiliary function that takes a 2-D matrix, and returns a new matrix without all the rows (but not the columns) that contain zeroes. Do this by only copying those rows that don’t have a zero into a new matrix, rather than by actually deleting rows from the old matrix.
- Use this function twice: once giving the matrix, and then the second time using the transpose of the resultant matrix (with unwanted rows deleted; we transpose so that it will now delete the unwanted columns in the original matrix)
- Then transpose again the result from this second function call

Method 6

- First, copy the 2-D matrix into another variable, so that you have oldMatrix and newMatrix
- Go through oldMatrix, determine which rows and columns have a zero, and delete the corresponding rows and columns from newMatrix

Method 7

- Write an auxiliary function that takes a 2-D matrix, and returns a 1-D array that contains the rows of the matrix that we want to keep (i.e. that don’t contain a zero).
- Use this function twice: once giving the matrix, and then the second time using the transpose of the matrix
- Use this information to then copy only the wanted elements from the oldMatrix into a newMatrix

6 Additional Questions

After completing the above exercises, if you have not finished them, please go on to the Lab 8 Exercises on Text & File I/O and Strings.

Lab 8 is very important – remember that you are guaranteed that an entire Q in Lab Exam 3 will be based on the material in week 8, which is worth 10 marks (including comments and coding style); that’s 5% of the entire course.

Please open all your exercises before calling the tutor to get marked off and leave early.