

Psych 254 (Poldrack)
Exercise 1.

For each exercise you should report the MATLAB code used to perform each operation, along with the results of that operation.

Basic matrix operations:

1. Create a new matrix variable with 2 columns and 10 rows called data containing the following values:

10	19
21	29
13	29
28	36
15	29
23	22
21	28
14	19
6	25
19	25

Save these data to a file called ex1_data. Use the size() function to find out the size of data.

2. Create two new variables called colmean and rowmean, containing the marginal column and row means for data, using the mean () function.

Matrix Algebra:

MATLAB is built around matrix algebra, and most operations in MATLAB can be done most efficiently using matrix algebra. In this section, we will review some basic matrix algebra operations. For additional tutorials, go to the following web site:

http://people.hofstra.edu/faculty/Stefan_Waner/RealWorld/tutorialsf1/frames3_1.html

3. First, create the following variables in MATLAB:

$$X = \begin{bmatrix} 1 & 5 \\ 3 & 7 \end{bmatrix}$$

$$Y = \begin{bmatrix} 2 & 8 \\ 4 & 2 \end{bmatrix}$$

$$Z = \begin{bmatrix} 3 & 6 & 3 \\ 2 & 5 & 2 \end{bmatrix}$$

4. Adding matrices. For two matrices to be added together, they must be the same size. Confirm this by trying to add $X+Y$ and $X+Z$; which one works, and what is the result?

A scalar can also be added to a matrix. What is the result of $X + 10$?

5. Scalar multiplication. A matrix can be multiplied by a scalar, which is the same as multiplying each entry in the matrix by that scalar. What is the result of $X * 2$?

6. Matrix multiplication. Unlike matrix addition, standard matrix multiplication does not directly multiply the individual entries in the matrix. Let's look at X and Y .

If we were to simply multiply X and Y in an entry-by-entry fashion, we would get:

$$\begin{array}{ccc} 1 & 5 & 2 & 8 & 2 & 40 \\ 3 & 7 & 4 & 2 & 12 & 14 \end{array} + =$$

However, this is not how matrix multiplication works. Instead, we multiply rows by columns. Let's take a simple example first. Let's say we have a 1×3 matrix $[10 \ 15 \ 20]$ and a 3×1 matrix $[3; 4; 1]$. To multiply these, we do as follows:

$$\begin{array}{ccc} 10 & 15 & 20 & * & \begin{array}{c} 3 \\ 4 \\ 1 \end{array} & = & [10*3 + 15*4 + 1*20] = 110 \end{array}$$

That is, you multiply each corresponding entry in rows versus columns, and then add them up to get the result. For more complex matrices, you get the a particular entry by multiplying the corresponding rows and columns, just as in the previous example. For example, entry 1,1 in $A*B$ is computed by multiplying row 1 of A with column 1 of B , and so on. Thus, the matrix multiplication version of $X * Y$ is:

$$\begin{array}{ccc} 1 & 5 & 2 & 8 & 22 & 18 \\ 3 & 7 & 4 & 2 & 34 & 38 \end{array} + =$$

confirm this for yourself. What happens if you do $Y*X$ rather than $X*Y$? Is the answer different?

One of the most important rules of matrix algebra has to do with the dimensions of the matrices you are multiplying. The inner dimensions of the two matrices must match in order to multiply them, and the size of the result is determined by the outer dimensions. For example, if you are multiplying a 3×5 matrix with a 5×2 matrix, the inner dimensions are 5 and 5, and the outer dimensions are 3 and 2. Thus, you could multiply

these matrices, and the result would be a 3 X 2 matrix. Confirm this in MATLAB by trying to multiply some matrices of various sizes.

7. The identity matrix. In regular algebra, if you want something to stay the same you can multiply it by one. In matrix algebra, the equivalent is the identity matrix (often denoted by I). This is a matrix with ones down the diagonal and zeros everywhere else:

$$I = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Create a 2 X 2 identity matrix and multiply it by X (you can either create it by hand or use a special function in MATLAB to create it – use “lookfor identity” to find this function). What is the result? Does the order of the multiplication matter?

Useful matrix operations: Computing the standard deviation

8. In statistics we often want to compute the sum of squares for a set of numbers, which is simply the sum of each element multiplied by itself. This can be accomplished by pre-multiplying a matrix by its transpose (often denoted with a ‘, e.g., the transpose of X is X’). For example, to get the sum of squares of the matrix [3;5;8] you would do the following:

$$\begin{bmatrix} 3 & 5 & 8 \end{bmatrix} * \begin{bmatrix} 3 \\ 5 \\ 8 \end{bmatrix} = [3*3 + 5*5 + 8*8] = 98$$

In this exercise, you will compute the standard deviation of the first column in data. If data is not in memory, then load it in from the file that you saved earlier. Create a new variable containing just the first column of data, calling it d.

The formula for the variance of a set of data is:

$$\text{Var} = (\text{SumSqrs} - \text{Sum}*\text{Sum}/n)/n-1$$

where SumSqrs is the sum of squares and Sum is the sum of the data. The standard deviation is simply the square root of the variance.

Compute the standard deviation of d using this formula. Check your answer by computing the standard deviation of d using the built-in standard deviation function (which you can find using the lookfor command – remember that if you are looking for multiple terms you have to put them in single quotes, e.g., lookfor ‘standard deviation’).

Name: _____

Take-home exercise 1 (turn in Oct 3)

1. Match the code on the left with the appropriate outputs on the right:

$$X = [1 \ 2 \ 3] \quad Y = [4 \ 5 \ 6]$$

Code	Your Answer (A-E)
X*5	
X+3	
X+Y	
X*Y	
X*Y'	

	Output
A	Error: inner dimensions must agree
B	[5 7 9]
C	32
D	[4 5 6]
E	[5 10 15]

2. What type of variable would be most appropriate to hold each of the following pieces of data (be as specific as possible):

- _____ a. "UCLA"
- _____ b. 12
- _____ c. 3.65
- _____ d. 399204
- _____ f. -14

3. True or false: For each line, determine whether the resulting Boolean value is true or false:

- true/false a. mean([-1 0 1])
- true/false b. ~false
- true/false c. true - false
- true/false d. (3*5)<=12
- true/false e. pi==3.14
- true/false f. (3==3) | false
- true/false g. (1~2) & (3/5<1)