Lecture 12

1. Turn in HW 6, Quiz 4
2. Review & Try:

Chapter 3.1 – Math & Trig functions

Chapter 3.2 – Data Analysis functions

3. New topics:

Chapter 4.1 – Matrix Operations

Chapter 4.2 – Solutions to Linear Equations

4. Next class – Read Chapter 4.
   - 20 mins - Review types of matrix multiplication
   - Rest of class - lab on solving linear equations using matrix operations
5. Monday - Quiz on Chapter 3.1 - 3.2, Chapter 4

3.2 - Data Analysis functions– see list p.61

- max (x) returns largest value in a row vector

if x is a matrix, returns vector with max of each column

- max (x,y) returns matrix with maximums after comparing matrix x and y element by element.
- mean (x) calculates average of a row vector
if x is a matrix, returns vector with average of each column
- sum (x) computes sum of elements in a vector
if x is a matrix, returns vector with sum of each column
- sort (x) sorts vector into ascending order
if matrix, sorts each column into ascending order

Also:

min(x) min(x,y) prod(x) median(x) cumsum(x) cumprod

3.2 – Standard deviation & histograms

- **std (x) computes standard deviation of the values in a vector x**
- **hist (x) generates a histogram**

Creating an m-file – (review)

% before starting insert diskette

cd a:

% start an m-file (1st click file → new → m-file)

% display program purpose

disp('This program will generate a plot.')

% ask user for value z

Z=input('Enter value for Z integer between 10 and 100: ')

% create two arrays

X= 0 : 0.5 : 50;

Y= 5 * X .^2;

% plot

plot(X,Y)

% save m-file .. then run it (click debug → run)

3.1 - Math & Trig functions – (review)

- **Math functions.**

% 1. remainder or mod function

```
rem(10,3) or mod(10,3)
```

% 2. “nested” functions

```
round (sqrt(143))
```

% 3. functions with a matrix as an argument

```
abs(-10:2:10)
```

- **Trig functions (use radians).**

% 4. tangent of 45 degrees

```
tan(45*pi/180)
sin(45*pi/180)
cos(45*pi/180)
```

3.2.1 - Vector & matrix data analysis– (review)

```
% Try #1. maximum and minimum
w=[0, 3, -2, 7]; x=[3, -1, 5, 7]; y=[1,3,7; 2,8,4; 6,-1,-2]
max(w)
min (x)
max (y)
min (w,x)
% Try #2. mean and median
mean(w)
mean(y)
z=1:5
median(z)
median(w)
% Try #3. cumsum & cumprod
cumsum(z)
y
cumprod(y)
```

3.2.1 - Vector & matrix data analysis– (review)

```
% Try #1 - sorting
w              % display w to remind yourself what is in w
sort(w)
w
% note: result of sort command is in answer (not in w)
ans
% let's put result of sort in w
w=sort(w)
% Try #2 -  sort a matrix
y              % display w to remind yourself what is in w
sort (y)
% #3 - remember that arguments can include calculations
% Before you do last sort… try to predict the answer
x              % display x to remind yourself what is in x
w              % display w to remind yourself what is in w
2*x+w          % display 2*x+w to see the result before sorting
sort(2*x+w)
```

Chapter 4.1 – Matrix Operations

Transpose: i'th row becomes i'th column. Elements (a,b) become (b,a). => If A = m x n matrix, then A^T = n x m.

```
% Try #1 - transpose a vector
X=2 : 2 : 10
Y=X'
% Try #2 - transpose a square matrix
A=[1, 2, 3;  4, 5, 6;  7, 8, 9]
size(A)
diag(A)
B=A'
size(B)                  % should get same as A  3x3
diag(B)                  % should get same as A ->  1  5  9
% Try #3 - transpose a non-square matrix
C=[A; 10, 11, 12]
D=C'
% predict sizes and diagonals of C and D before doing them
```

Chapter 4.1 – Matrix Dot Product

Definition: A dot product is a scalar computed from two vectors (must be same size)

$$\rightarrow \; a_1b_1 + a_2b_2 + a_3b_3 + \ldots a_ib_i$$

```
% Try dot product by hand first to predict results
X=[4, -1, 3]
Y=[-2, 5, 2]
X.*Y                    % this is element by element multiplication
sum(X.*Y)               % this is the dot product
% or use the built in function in MATLAB
%    to calculate the dot product of the two vectors
dot(X,Y)
```

Chapter 4.1 – Matrix Multiplication

•**NOT** ELEMENT BY ELEMENT MULTIPLICATION THAT WE DID IN CHAPTER 2 → A.*B

C = A*B is calculated by taking the dot products of each ROW in A with each COLUMN in B.

C(i,j) = dot product of row i in A with col j in B

This is why the # columns in A (i.e., length of rows in A) must be same as the # of rows in B (i.e., length of cols in B)

So if A is (m x p) and B is (p x n), then C is (m x n):

```
C=[ C(1,1)  C(1,2)   C(1,3) …  C(1,n)
    C(2,1)  C(2,2)   C(2,3) …  C(2,n)
    C(3,1)  C(3,2)   C(3,3) …  C(3,n)
    …              …             …
    C(m,1) C(m,2)   C(m,3) … C(m,n)]
```

Chapter 4.1 – Matrix Multiplication

- When can you do it?
  - → inner dimensions must be equal
- What are dimensions of results?
  - → outer dimensions

OK to do:

Matrix X is:	Matrix Y is:	Result will be:
2 by 3	3 by 2	2 by 2
4 by 2	2 by 4	4 by 4
5 by 5	5 by 5	5 by 5
m by n	n by p	m by p

CAN'T DO:

4 by 3	4 by 3	Error!

Chapter 4.1 – Matrix Multiplication

*** NOT ELEMENT BY ELEMENT MULTIPLICATION THAT WE DID IN CHAPTER 2 → A.*B**

```
% Try this where A is (2 x 3) and B is (3 x 3)
% result of matrix multiplication is (2 x 3) matrix C
A=[2, 5, 1;  0, 3, -1]
B=[1,  0,  2;   -1, 4, -2;   5, 2, 1]
% calculate each element in resulting matrix C
C(1,1)=A(1,1)*B(1,1)+A(1,2)*B(2,1)+A(1,3)*B(3,1)
C(1,2)=A(1,1)*B(1,2)+A(1,2)*B(2,2)+A(1,3)*B(3,2)
C(1,3)=A(1,1)*B(1,3)+A(1,2)*B(2,3)+A(1,3)*B(3,3)
C(2,1)=A(2,1)*B(1,1)+A(2,2)*B(2,1)+A(2,3)*B(3,1)
C(2,2)=A(2,1)*B(1,2)+A(2,2)*B(2,2)+A(2,3)*B(3,2)
C(2,3)=A(2,1)*B(1,3)+A(2,2)*B(2,3)+A(2,3)*B(3,3)
% easy way to do matrix multiplication in MATLAB
D= A*B
```

Chapter 4.1 – Matrix Multiplication

* **For Quiz, should be able to compute (by hand):**

A=[2, 1; 0 -1; 3 0];
B=[1 3; -1 5];
C=[3 2; -1 -2; 0 2];
D=[1 2]

DB^2 (Matlab command = D*B^2) (What if D*B.^2?)

BC^T (Matlab command = B*C')

$(CB)D^T$ (Matlab command = (C*B)*D')

AC^T (Matlab command = A*C')

Practice on your own (later) - check using Matlab.

Chapter 4.1 – Identity Matrix

*** An identity matrix is composed of:**

1's on the main diagonal and 0's everywhere else.

If A is a square matrix and I is an identity matrix of the same size… then A*I = A …. also A*I = I*A

```
% try this
% generate identity matrix
I=eye(3)
A=[1,  0,  2;   -1, 4, -2;   5, 2, 1]
A*I
I*A
%    note that the answer to A*I is the same as I*A
(A*I)-(I*A)
```

Chapter 4.1 – Matrix Inverse

* If A is a square matrix …

…..then the inverse of A is A^{-1}…

Definition: $A * A^{-1} = I$ (identity matrix)

…. also $A * A^{-1} = A^{-1} * A$

MATLAB COMMAND IS: inv(A)

```
% try creating a matrix inverse
A=[2,  1;   4, 3]
A_inverse=inv(A)
A*A_inverse
A_inverse*A
%  note: answer to both of the above is identity matrix
```

Chapter 4.2 – Solutions to Linear Equations

* Let's solve the following system of equations:

$$3x_1 + 5x_2 = -7$$

$$2x_1 - 4x_2 = 10$$

Linear Algebra Format:

* Put the coefficients on left side into matrix A
* Put the numbers on the right side into column B
* Unknowns are column X

$$A = \begin{pmatrix} 3 & 5 \\ 2 & -4 \end{pmatrix} \quad X = \begin{pmatrix} x1 \\ x2 \end{pmatrix} \quad B = \begin{pmatrix} -7 \\ 10 \end{pmatrix}$$

AX= B (to solve for X → multiply both sides by the inverse of A) …. so **X=inv(A) * B**

Chapter 4.2 – Solutions to Linear Equations

```
% Try solving  it in MATLAB
% put coefficients from left side of the equations into A
A=[3   5;   2   -4]
% put coefficients from right side of equations into
%               column vector B
B=[-7;  10]
% find solutions (X) by multiplying the inverse of A by B
X=inv(A)*B
```

```
% Another way to solve Linear Equations
% ….. using Matrix Left Division
%  uses Gaussian elimination without forming the inverse
A \  B
% this is a direct method => quicker and more precise
```

Chapter 4 – Summary

Two methods for solving a system of equations that are in the format: A * X = B

1. using inverse: X = inv(A) * B
2. using matrix left divisions: X = A \ B

Rules for doing matrix multiplication:

1. "inner dimensions" of matrices must be equal
2. resulting matrix will equal "outer dimensions"

Chapter 4.1 – Matrix Determinant

*** If A is a square 2 by 2 matrix = [a_{11} a_{12}; a_{21} a_{22}]**

…..then the determinant of A is a scalar

$$| A | = (a_{11}*a_{22}) - (a_{21}*a_{12})$$

MATLAB COMMAND IS: det(A)

```
% try calculating the determinant of a 2 by 2 matrix
% first do it by hand to predict results
A=[1,  3;  -1, 5]
det(A)
%  try determinant of a 3 by 3 matrix – see page 92
B=[1  3   0;   -1  5  2;   1  2  1]
det(B)
% try determinant of singular matrix (doesn't have inverse)
C=[2, 4;  -1, -2]
det(C)
inv(C)
```