ARRAY VARIABLES (ROW VECTORS)

% Variables in addition to being singular valued can be set up as AN ARRAY of numbers. If we have an array variable as a row of numbers we call it a **ROW VECTOR**. You can also think of array variables as physics vectors with each member being a component. But row vectors we can have many components not just 3 as in physics.

,
% note: IN GENERAL USE remember FOR VARIABLE NAMES, MATLAB IS CASE SENSITIVE!
%we usual use lower case for basic variables and upper case for Matrices we will study later.
% We use the brackets [ ] to indicate the range of values in the row vector.
For examples
>> A=[3,5]
A = 3 5

>> velocity = [23.4 45.3 -9.8]
velocity = 23.4000 45.3000 -9.8000

% MATLAB shows us these **row vectors** as a row of their values.

% A four component array follows
>> r = [1 2 3 4]
r = 1 2 3 4

ARRAY OPERATORS

SIMILAR TO SCALAR BUT WITH DIFFERENT RULES FOR ROW VECTORS AND MATRICES
We first explore Element by Element operations ON ROW VECTORS
That is, when performed each member of the array is acted on with the operator.
These are
+ - .* ./ .^

>>A =[3 6 8] ;
>> B =[2 3 4];

If just using a scalar on an array its an element by element operation and scalar operators are ok as in
>>C = 2 *A
C = 6 12 16

>>C= A / 2
C = 1.5000 3.0000 4.0000

Equivalent to using the array operators which is redundant in this case

>>C= 2 .* A
C = 6 12 16

>>C =A ./ 2
C = 1.5000 3.0000 4.0000

But operation between arrays have specific meaning if an element by element effect is needed than one must use the array operators as follows
>>D= A + B
D = 5 9 12

>>D= B- A
D = -1 -3 -4

>>E= A .* B
E = 6 18 32
NOTE: if one summed the three values then this is equivalent to the DOT PRODUCT of two vectors in Math
Or sum(E) =56

MATLAB also provides the sum of the dot product from the function dot()
>>SumAB = dot(A,B)
SumAB = 56

We know from math A ○ B = |A| |B| cosθ

SUPPOSE WE WANT THE ANGLE? Then we need
acos θ = (**A**○ **B**)/ **|A| |B|** A**|A|** and B **|A| |B|** in divisor are magnitudes.

The magnitude of a vector is gotten by the Pythagorean technique which means we would have to square the components and then add them and take the square root.
This is easily done by first consider the array operator .^ which acts on each component

>>G =A.^2
G = 9 36 64
Summing the components which are the squares of the original vector and taking the square root gives us the magnitude of the vector. Using MATLAB functions.

>>MagG=sqrt(sum(G))
MagG = 10.4403

So putting all this together gives us the angle (in radians) as follows

>>Angle =acos(sum(A.*B) / (sqrt(sum(A.^2)) * sqrt(sum(B.^2))))
Angle = 0.0890 (radians)

Or in degrees
>>angledeg =Angle*180/pi
angledeg = 5.1022 degrees

Think about these dot calculations and be sure you understand the approach.
MATLAB offers other ways to do the latter.

IF YOU TRY SCALAR OPERATIONS ON VECTORS YOU GET AN ERROR MESSAGE THAT WILL MAKE MORE SENSE LATER (UNLESS YOU UNDERSTAND MATRIX MULTIPICAITON).

>> E=A*B
Error using * Inner matrix dimensions must agree
E= A * B is not ok with two row vectors here (later with matrices it will be, provide the dimensions of the matrices are proper)

Errors will also happen if you use the other scalar operator A/B A^2 on row vectors
Which can be used when we define the array variables called matrices and even then there are special matrix operations that are invalid. The latter is because the area in mathematics that was developed for matrices has very specific operation which MATLAB provides.

Finally we can also do element by element division as follows. Note again the use of " ./ "

>>F = A ./ B

F = 1.5000 2.0000 2.0000 an element by element division!
Be sure to check simple examples when using these operators!

Some additional Illustrations
% if we want to raise each value of the vector to a power, we can also use power (a,b) a is vector, b the power.
Consider again
% the four component row vector
>> r = [1 2 3 4]
r = 1 2 3 4

z=power(r,3)
z= 1 8 27 64

% Try z=r.^3

% In addition to generating a row vector with the brackets [ ] we can have MATLAB generate a multivalued vector useful for representing an independent variable or even a dependent one.
For example: A row vector all the x axis values we will use to solve some function to get our y axis values.
% We illustrate with an x range of from -4 TO +4 WITH A STEP OF 0.1 between each of the x values.
% NOTE use of **" : "** (colon) in the definition of this multivalued vector
>> x = -4:.1:4
x =
Columns 1 through 7
-4.0000 -3.9000 -3.8000 -3.7000 -3.6000 -3.5000 -3.4000
... Columns 22 through 28
-1.9000 -1.8000 -1.7000 -1.6000 -1.5000 -1.4000 -1.3000
... Columns 57 through 63
1.6000 1.7000 1.8000 1.9000 2.0000 2.1000 2.2000
... Columns 78 through 81
3.7000 3.8000 3.9000 4.0000
% NOTE ALL 81 COMPONENTS WERE NOT SHOWN HERE

% MATLAB shows us the components in one long row. Since all values cannot be seen in one line, each position is labeled as a Column.
% Once we have the independent variable we can use MATLAB supplied
functions such as sin(),sinh(),log10() etc., to obtain corresponding y = f(x) values for simple functions or functions we need that are combinations of the basic ones.
% For Example, using the above x values we can generate y = sin(x) values
% for each x value by simply stating sin(x) which is an element by element evaluation of the sin()
>> y = sin(x)
y =
Columns 1 through 7
0.7568 0.6878 0.6119 0.5298 0.4425 0.3508 0.2555
... Columns 22 through 28
-0.9463 -0.9738 -0.9917 -0.9996 -0.9975 -0.9854 -0.9636

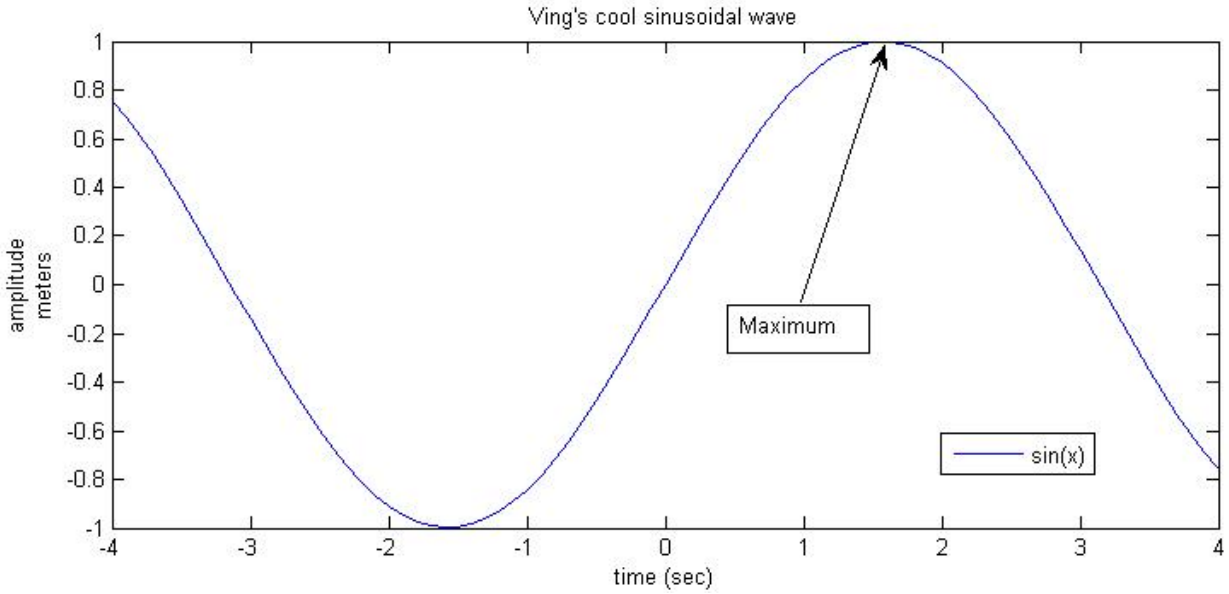
... Columns 57 through 63
0.9996 0.9917 0.9738 0.9463 0.9093 0.8632 0.8085
... Columns 78 through 81
-0.5298 -0.6119 -0.6878 -0.7568

% We have just generated another vector variable whose 81 components
are the given function (sine in this case) of each of the components of the independent variable 'x'.
% As we generate values they are kept in memory by MATLAB.

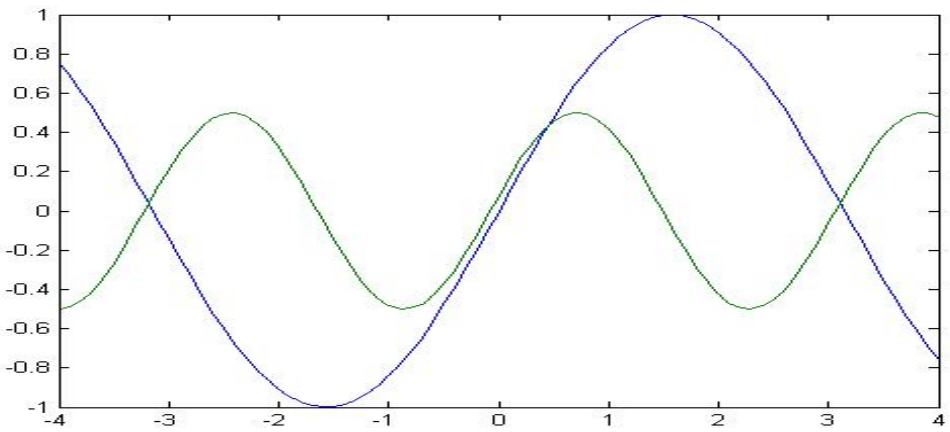
GRAPHING

We now have y=sin(x) and x values in memory at this point. Now we can use the MATLAB plot() function to get a graph of x versus our f(x) (= 'y' here).
% The plot appears in another window called the graphics window. Tapping any key (space bar is convenient) will exit back to the command window.
% Doing the following.

>> plot(x,y)
% Generates the plot (graph) below **In a FIGURE WINDOW that includes numerous graphical tools in the bar at the top.**
The figure window can be printed by clicking on “FILE” and choosing print options;
Numerous options to label and modify the graph are in the menus in the Figure window.
Try in the Insert menu to figure out what options were used below to “polish” the image.
Check out some of the other options, by just exploring them for fun!
% We note that MATLAB automatically scales the axis. We will explore later more controls of graphic options.



%MATLAB provides a number of different plotting options for engineers and scientist that are well known in the various STEM fields. (examples: log vs linear, log vs log plots).
% Multple functions can be plotted in the same graph with the plot() function. An example follows.
>> z= cos(2*x+30);.
>> plot(x,y,x,z); creates the following:



CREATING A TABLE WITH THE TRANSPOSE OPERATOR

%Since tables are columns of data we can use the transpose operator to transform how data is seen and also treated in memory. This operator plays an important part in Matrix theory which we will see later on.

Given a row vector we transform it with the transpose operator as a column vector. These latter vectors also play an important role in Matrix theory. So when we output from a column vector the data lines up as a column which forms the basis of creating a table. Consider the row vector

```
>> z=1:5
z=     1     2     3     4     5                THIS IS A MATRIX OF 0NE ROW AND 5 COLUMNS
```

The transpose is just using a single quote on the row vector variable, as x'

```
>> z'
ans =
     1
     2
     3
     4
     5                    THIS IS A MATRIX OF 1 COLUMN AND FIVE ROWS
```

Now consider the row vector w

```
 >> w=-10:5:10
w =   -10    -5     0     5    10
```

A table of both z and w can be created using braces and the transpose of the row vectors as follows M =[z' w'] which is useful when we want to create a of tabulate data in our programs. We are creating this way a matrix, M, in this case has five row and two columns whose output will give us a table of values. We say M is a 5 by 2 or 5 X 2 matrix that looks like…

```
 >> M=[z' w']
M =
     1   -10
     2    -5
     3     0
     4     5
     5    10
```

MEMORY INQUIRY

% It is useful to know at times what variables are being kept in the memory. We obtain this answered with the 'who' command

```
>> who
Your variables are:
 r  velocity     x    y          .
```

% More detail about the variables in memory is gotten with 'whos'

```
>> whos
             Name          Size            Bytes  Class     Attributes
                r             1x4               32  double
             velocity      1x3               24  double
              x            1x81             648  double
              y            1x81             648  double
```

% We note that the Size of a variable is expressed in **matrix form**. Namely all of the variables above are on one row with variable column numbers corresponding to the number of components of our row vectors. We can, in addition, think of our row vectors (called arrays in MATLAB) as a **Matrix** with one row.

Recall a **Byte** is the number of 8 bit binary units that mean something, numbers in memory are represented by zero's and ones in **binary numerical** value (8 at a time or 1 bite ex. 10101101 and alphabet and numbers not used for calculations, called **alphanumerics,** have a code for each letter and number. The code is called **ASCII as was seen in C++** on most computers except Main Frame ones.
In the table Bytes: is the number of bytes used per number so for example r has 4 numbers whose total bytes used are 32. So it must be that 8 bytes are a double Class, the more bytes used the greater number of significant digits kept.

%REMINDER: IN WRITING EXPRESSION FOR ROW VECTORS(ARRAYS) WE MUST USE A PERIOD '.' WITH OPERATORS AS IN .* AND ./ AND .^ FOR ELEMENT BY ELEMENT MULTIPLICATION, DIVISION AND EXPONENTIATION.

%

LABORATORY TASKS

Notes: command "clear" will clean the memory of all values and is useful for the following
Placing a semi colon ";" at the end of a line suppresses MATLAB echo of all values when you define a variable or vector so it is easier to collect the source code to hand in with each task.

PRINT THE GRAPH in tasks 8-14 AND SUCCESSFUL COMMANDS (**SOURCE CODES**) OF EACH OF THE FOLLOWING TASKS but before you print a graph after you generate it use the **title** option in tools to put your name as a unique heading on the graph. This is especially important when many students are printing graphs in the laboratory. **Label the x and y axis with meaningful names that are appropriate to the functions you are graphing.**
Use ; on commands to suppress the MATLAB echo (unless needed) of many values when you print up your work on the functions to support your graph outputs. Hand in the graphs and the commands and functions and variables you defined with the graphs.

8. Generate the curve for $y(t)=3t^2 + 2t -4.5$ for t from 0 to 10 step .1
NOTE: The notation y(t) is mathematical not a MATLAB variable.

9. Generate the curve $y1(t)=4\sin(5\pi t)$ with t from -.4 to +.4 steps .01
Use the constant 'pi' for π which is well known to MATLAB when you write your expression.

10.(2 PTS) Generate the function $y2(t)=8\sin(3\pi t +\pi/2)$ with the same t vector as above (TASK #9) then plot simultaneously the two curves y1 AND y2.
SOLUTION: This is done by noting that the MATLAB 'plot' function can be used as plot(x1,y1,x2,y2) for two functions y1 and y2 over the ranges x1 and x2. Both y1 and y2 use the same range there. See 'help plot'.

11. **DIGITAL SIGNAL PROCESSING** uses the summing of various sinusoidal curves to simulate real signals. Obtain the graph for the sum of the two sinusoids in the function $y3(t)= 8\sin20\pi t +8\sin22\pi t$ for t from 0 to 2 with intervals of .01. Use 'pi'.

12. As a variation of 11 multiply the expression in 11 by the exponential function $e^{-1.2t}$. The resulting plot will be an illustration of a signal that dampens out. (Reminder: 'e' is the exponential function 'exp' and each part is a vector and when multiplying the sum of sin()'s and the exp().

13. (3 PTS) Review the weather balloon problem we did with C++ where we had time going from 0 to 48 hours and generated the altitude and velocity of the balloon. Using the polynomial solutions generate the two curves you saw illustrated in the power point notes along with titles and labeled axis. Where t is in hours
$alt(t)= -0.12t^4 +12t^3 -380t^2 +4100t +220$ in meters

$v(t)= -0.48t^3 +36t^2 -760t +4100$ m/hours but convert to m/sec for the graph values
also generate a table of values from the row vectors you have created for alt and v
only include a copy of the first 10 values of the table to show you did it as well as the work that was used.

14. In the illustrated graph of y=sin(x) above (the graph in these notes). List the names of the options used to polish the graph in the menus found in the figure window?

15.(3 PTS) Given the vectors H=[34 -56 23] and J =[9 12 26] fully set up and solve for the magnitude of each and the angle in degrees between the two vectors using dot product discussed in this lesson.

16. (2 PTS) Using the vectors H and J above find
a. H^2 H' H'^2
b. Pushing our knowledge MATRIX MATH (Comment here). try J*H H*J J* H' H'*J (NO DOTS LAST TWO)