**MATLAB Techniques Summary – Scripts, Matrix & Linear Systems, Complex Numbers**

**Scripts**

Reminder: start a script by clicking on the “New Script” icon on the far left of the Home tab. Start with comments that include: the name of the file, its purpose, your name, the date. Include definition of all variables and units in comments. Scripts use commands exactly as they are used in the command window.

Script Input Options

1. Hard Code in an input section in the code – have a separate section where you define the value of starting variables
2. Read from the workspace – variables can be defined at the command line. Once in the workspace they are available to the script
3. Interactive input commands:

a = input(‘text’) Displays the “text” in the command window and waits for a user response.

Returns the user response as output (i.e., stores it in variable ***a*** in this case).

a = input(‘text’,’s’) Similar to above except it treats the user input as text instead of as a number

Script Output

; a semicolon at the end of a executable expression will suppress the echo printing of the result of that expression. Use to hide intermediate results. Leave off final results to allow them to show.

disp(‘text’) a simple way to add text to the command line output. Use before your final output to label

**Vectors & Matrices**

Creating Row Vectors (vectors that form a horizontal row of numbers)

[x1, x2, x3 …] Creates a vector (a) of the elements listed in the square brackets. This form can also be used to combine two or more row vectors into one new vector.

b = m:s:n Three numbers separated by colons will create a list that starts at the first number n, counts by the increment in the middle (s) and stops when the last number (n) is reached

c = linspace(m,n,p) This function creates a vector that starts at m, stops when n and includes p equally spaced samples (take not of the order of the arguments).

Creating Column Vectors (vectors that form vertical column of numbers)

a = [x1; x2; x3; …] Separating the numbers in square brackets by semicolons will create a column vector.

b' Using an apostrophe will “Transpose” an existing row vector into a column vector.

Matrix Creation

[x11, x12,…; x21, x22 …] Creates a matrix, starting a new row at the semicolon (;)

all rows must have the same number of elements

d =[a;b;c] If you have a series of equal length row vectors they can be combined (concatenated) into a matrix by listing in square brackets separated by semicolons.

Variable editor Double clicking on a variable in the workspace will open a spreadsheet like editor (to edit existing values or add new ones).

**Vectors & Matrices (cont.)**

Addressing (spacing added for clarity)

x (a) Returns the ath element in the vector x

x (a : b) Returns the ath through the bth elements in vector x

A(a, b) Returns the element in the ath row and the bth column of matrix A

A( : , b) Returns the bth column in its entirety of matrix A

A( a , : ) Returns the ath row in its entirety of matrix A (a colon **as an address =** all elements)

A( a : b, c) Returns rows a through b of the cth column of matrix A

**Matrix Solutions**

Systems of Equations in Matrix form

Example: for the system of two equations: 3x + 4 y = 3 and

5 x + 3y = 2

the matrix equation would be: **A X** = **b**

where:

Functions

linsolve(A,b) Returns (solves for) **X** in the matrix equation **AX** = **b**

inv(A) Returns the inverse of a matrix A; the matrix must be square

det(A) Returns the determinant of a matrix A; the matrix must be square

Operators

A’ Takes the transpose of a matrix A (i.e., reverses rows and columns)

\ Left division can be used to solve linear equations in matrix form using **b**\**A**, where A is the coefficient matrix and b is the column vector of constants. Preferred method.

Use the help command to get additional information on any of these topics.

**Complex Number Summary**

Definitions for variables used in this section

x = some complex number a = the real part of a complex number

i or j = the square root of -1 b = the imaginary part of a complex number

M = the magnitude of the complex number, i.e., its length in the complex plane

θ = the angle (in radians) of a complex number, in the complex plane

Cartesian Form: *x = a + bi or a + bj* Polar Form: *x = M ‚ θ* Eulerian Form: *x = Me θi*

Entering Complex numbers in MATLAB

Cartesian: >> x = 3 + 4j % this stores the complex number 3 + 4i in the variable x

Eulerian: >> x = 5\*exp(0.9273i) % this stores the complex number 5*e 0.9273i*

Complex Functionsabs(x) returns the magnitude (i.e., M) of the complex number x

angle(x) returns the angel (i.e., θ) of the complex number x

real(x) returns the real component (i.e., a) of the complex number x

imag(x) returns the imaginary component (i.e., b) of the complex number x