MATLAB NOTES

Matlab designed for numerical computing.

Strongly oriented towards use of arrays, one and two dimensional.

Excellent graphics that are easy to use.

Powerful interactive facilities; and programs can also be written in it.

It is a procedural language, not an object-oriented language.

It has facilities for working with both Fortran and C language programs.
USING MATLAB

At the prompt in Unix, type *Matlab*. Or use the pull down menu accessed with the right button.

Run the *Demo* program (simply type *demo*).

To seek help on any command, simply type

```
help command
```

To seek information on Matlab commands that involve a given *word* in their description, type

```
lookfor word
```

Look at the various online manuals available thru the course web page.
MATLAB is an interactive computer language. For example, to evaluate
\[ y = 6 - 4x + 7x^2 - 3x^5 + \frac{3}{x + 2} \]
use
\[ y = 6 - 4x + 7x^2 - 3x^5 + \frac{3}{x + 2}; \]
There are many built-in functions, e.g.
\[ \text{exp}(x), \cos(x), \text{sqrt}(x), \log(x) \]
The default arithmetic used in MATLAB is double precision and real. However, complex arithmetic appears automatically when needed. \text{sqrt}(-4) results in an answer of 2i.
The default output to the screen is to have 4 digits to the right of the decimal point. To control the formatting of output to the screen, use the command `format`. The default formatting is obtained using

```
format short
```

To obtain the full accuracy available in a number, you can use

```
format long
```

The commands

```
format short e
format long e
```

will use ‘scientific notation’ for the output. Other `format` options are also available.
MATLAB works very efficiently with arrays, and many tasks are best done with arrays. For example, plot $\sin x$ and $\cos x$ on the interval $0 \leq x \leq 10$.

\[
\begin{align*}
t &= 0:.1:10; \\
x &= \cos(t); \quad y = \sin(t); \\
\text{plot}(t,x,t,y)
\end{align*}
\]

The statement

\[
t = a:h:b;
\]

with $h > 0$ creates a row vector of the form

\[
t = [a, a + h, a + 2h, \ldots]
\]
giving all values $a +jh$ that are $\leq b$.

When $h$ is omitted, it is assumed to be 1. Thus

\[
n = 1:5
\]

creates the row vector

\[
n = [1, 2, 3, 4, 5]
\]
ARRAYS

\[ b = [1, 2, 3] \]

creates a row vector of length 3.

\[ A = [1 2 3; 4 5 6; 7 8 9] \]

creates the square matrix

\[
A = \begin{bmatrix}
1 & 2 & 3 \\
4 & 5 & 6 \\
7 & 8 & 9
\end{bmatrix}
\]

Spaces or commas can be used as delimiters in giving the components of an array; and a semicolon will separate the various rows of a matrix. For a column vector,

\[ b = [1 \ 3 \ -6] \]

results in the column vector

\[
\begin{bmatrix}
1 \\
3 \\
-6
\end{bmatrix}
\]
ARRAY OPERATIONS

**Addition**: Do componentwise addition.

\[
A = \begin{bmatrix} 1, & 2; & 3, & -2; & -6, & 1 \end{bmatrix}; \\
B = \begin{bmatrix} 2, & 3; & -3, & 2; & 2, & -2 \end{bmatrix}; \\
C = A + B;
\]

results in the answer

\[
C = \begin{bmatrix} 3 & 5 \\ 0 & 0 \\ -4 & -1 \end{bmatrix}
\]

**Multiplication by a constant**: Multiply the constant times each component of the array.

\[
D = 2*A;
\]

results in the answer

\[
D = \begin{bmatrix} 2 & 4 \\ 6 & -4 \\ -12 & 2 \end{bmatrix}
\]
Matrix multiplication: This has the standard meaning.

\[
E = \begin{bmatrix} 1, & -2; & 2, & -1; & -3, & 2 \end{bmatrix}; \\
F = \begin{bmatrix} 2, & -1, & 3; & -1, & 2, & 3 \end{bmatrix}; \\
G = E \times F;
\]

results in the answer

\[
G = \begin{bmatrix} 1 & -2 \\ 2 & -1 \\ -3 & 2 \end{bmatrix} \begin{bmatrix} 2 & -1 & 3 \\ -1 & 2 & 3 \end{bmatrix} = \begin{bmatrix} 4 & -5 & -3 \\ 5 & -4 & 3 \\ -8 & 7 & -3 \end{bmatrix}
\]

A nonstandard notation:

\[
H = 3 + F;
\]

results in the computation

\[
H = 3 \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} + \begin{bmatrix} 2 & -1 & 3 \\ -1 & 2 & 3 \end{bmatrix} = \begin{bmatrix} 5 & 2 & 6 \\ 2 & 5 & 6 \end{bmatrix}
\]
Matlab has component-wise operations for multiplication, division and exponentiation. These three operations are denoted by using a period to precede the usual symbol for the operation. With

\[
a = [1 \ 2 \ 3]; \quad b = [2 \ -1 \ 4];
\]

we have

\[
a.*b = [2 \ -2 \ 12]
\]
\[
a./b = [0.5 \ -2.0 \ 0.75]
\]
\[
a.^3 = [1 \ 8 \ 27]
\]
\[
2.^a = [2 \ 4 \ 8]
\]
\[
b.^a = [2 \ 1 \ 64]
\]

The expression

\[
y = 6 - 4x + 7x^2 - 3x^5 + \frac{3}{x + 2}
\]

can be evaluated at all of the elements of an array \(x\) using the command

\[
y = 6 - 4*x + 7*x.*x - 3*x.^5 + 3./(x+2);
\]

The output \(y\) is then an array of the same size as \(x\).
OTHER COMMANDS

clear: To remove the current variables from use.

cls: To clear the output screen.

help command_name: Brief description of command_name.

    help sqrt

results in the output

SQRT Square root.

SQRT(X) is the square root of the elements of X. Complex results are produced if X is not positive.
Special arrays:

\[ A = \text{zeros}(2,3) \]

produces an array with 2 rows and 3 columns, with all components set to zero,

\[
\begin{bmatrix}
0 & 0 & 0 \\
0 & 0 & 0
\end{bmatrix}
\]

\[ B = \text{ones}(2,3) \]

produces an array with 2 rows and 3 columns, with all components set to 1,

\[
\begin{bmatrix}
1 & 1 & 1 \\
1 & 1 & 1
\end{bmatrix}
\]

\( \text{eye}(3) \) results in the \( 3 \times 3 \) identity matrix,

\[
\begin{bmatrix}
1 & 0 & 0 \\
0 & 1 & 0 \\
0 & 0 & 1
\end{bmatrix}
\]
ARRAY FUNCTIONS

There are many *MATLAB* commands that operate on arrays, we include only a very few here. For a vector \( x \), row or column, of length \( n \), we have the following functions.

\[
\begin{align*}
\text{max}(x) &= \text{maximum component of } x \\
\text{min}(x) &= \text{minimum component of } x \\
\text{abs}(x) &= \text{vector of absolute values of components of } x \\
\text{sum}(x) &= \text{sum of the components of } x \\
\text{norm}(x) &= \sqrt{|x_1|^2 + \cdots + |x_n|^2}
\end{align*}
\]
SCRIPT FILES

A list of interactive commands can be stored as a **script file**. For example, store

\[
\begin{align*}
t &= 0:.1:10; \\
x &= \cos(t); \quad y = \sin(t); \\
plot(t,x,t,y)
\end{align*}
\]

with the file name *plot_trig.m*. Then to run the program, give the command

```
plot_trig
```

The variables used in the script file will be stored locally, and parameters given locally are available for use by the script file.
FUNCTIONS

To create a function, we proceed similarly, but now there are input and output parameters. Consider a function for evaluating the polynomial

\[ p(x) = a_1 + a_2 x + a_3 x^2 + \cdots + a_n x^{n-1} \]

MATLAB does not allow zero subscripts for arrays. The following function would be stored under the name polyeval.m. The coefficients \( \{a_j\} \) are given to the function in the array named coeff, and the polynomial is to be evaluated at all of the components of the array \( x \).
function value = polyeval(x,coeff);
%
% function value = polyeval(x,coeff)
%
% Evaluate a polynomial at the points given
% in x. The coefficients are to be given in
% coeff. The constant term in the polynomial
% is coeff(1).

n = length(coeff)
value = coeff(n)*ones(size(x));
for i = n-1:-1:1
    value = coeff(i) + x.*value;
end