

Notes on using MATLAB

MATLAB is an interactive program for numerical methods, with graphing capability. These notes describe some useful functions and syntax. The following sites have more extensive tutorials:

http://www.engin.umich.edu/caen/technotes/CAEN705_Matlab.html

<http://www.math.mtu.edu/~msgocken/intro/intro.html>

<http://www.math.unh.edu/mathadm/tutorial/software/matlab/>

http://www.mines.utah.edu/gg_computer_seminar/matlab/matlab.html

The command for starting MATLAB depends on your system configuration (you can often start MATLAB on UNIX systems by typing **matlab**). To obtain help from within MATLAB, type **help**; this provides a list of available functions. Supply the function name for information about a particular item (e.g. **help plot**). For demonstration of a few commands, type **demo**. To terminate a MATLAB session, type **quit**.

Formats for printing numbers.

format short 3.1416

format short e 3.1416e+00

format long 3.14159265358979

format long e 3.141592653589793e+00

There is only one data type in MATLAB, complex matrices. Vectors and scalars are special cases. Matrices can be created as follows, **A = [1, 1, 1, 1; 1, 2, 3, 4]**. This creates a 2×4 matrix A whose first row is (1,1,1,1) and whose second row is (1,2,3,4). The dimensions of a matrix A can be found by typing **size A**.

To create a vector, type **x=[1,2,3,4]**. The system responds with:

x =

1 2 3 4

The commas are optional, **x=[1 2 3 4]** gives the same result. If an assignment statement ends with a semicolon, then the result is not displayed. Thus if you type **x=[1 2 3 4];**, nothing will be displayed. You can then type **x** to display the vector. The length of a vector **x** is obtained from **length(x)**. Indices for vectors and matrices must be positive integers. Thus, A(1.5,2) and x(0) are not allowed. There is a special syntax for creating a vector whose components differ by a fixed increment. Thus, **x=[0 .2 .4 .6 .8 1]** can be created by typing **x=0:.2:1**.

Built-in functions.

pi 3.1415....

zeros(3,3) 3×3 matrix of zeros

eye(5) 5×5 identity matrix

ones(10) vector of length 10 with all entries =1

abs(x) absolute value

sqrt(x) square root, e.g. **i=sqrt(-1)**

real(z), imag(z) real, imaginary parts of a complex number

| | |
|-------------------------|--|
| conj(z) | complex conjugate |
| atan2(y,x) | polar angle of the complex number $x + iy$ |
| sin(x), cos(x) | trig functions |
| sinh(x), cosh(x) | hyperbolic functions |
| exp(x) | exponential function |
| log(x) | natural logarithm |
| gamma(n) | gamma function = $(n-1)!$ |
| bessel (a,x) | bessel function of order a at x |

Example of a loop.

```
for i = 1:4
    x(i) = i;
end
```

Example of a conditional.

```
if a==0;
    x = a+1;
elseif a < 0;
    x = a-1;
else;
    x = a+1;
end
```

Plotting.

| | |
|----------------------------|--|
| plot(x,y) | linear plot, uses defaults limits, x and y are vectors |
| grid | draw grid lines on graphics screen |
| title('text') | prints a title for the plot |
| xlabel('text') | prints a label for the x-axis |
| ylabel('text') | prints a label for the y-axis |
| axis([0, 1, -2, 2]) | overrides default limits for plotting |
| hold on | superimpose all subsequent plots |
| hold off | turns off a previous hold on |
| clg | clear graphics screen |
| mesh | 3-d plot; type help mesh for details |
| contour | contour plot; type help contour for details |
| subplot | several plots in a window; type help subplot for details |

Example. To plot a Gaussian function, type the following lines:

```
x = -3:.01:3;
y=exp(-x.*x);
plot(x,y)
```

Matrix functions.

| | |
|-------------------------|-------------------------------------|
| x = A\b | gives the solution of $Ax=b$ |
| [l,u] = lu(A) | LU decomposition of A |
| [v,d] = eig(A) | eigenvalues in d, eigenvectors in v |
| [u,s,v] = svd(A) | singular value decomposition |

| | |
|-----------------------|---|
| chol(A) | Cholesky factorization |
| inv(A) | inverse of a square matrix |
| rank(A) | matrix rank |
| cond(A) | condition number |
| * , + | matrix product and sum |
| .* , .+ | element by element product and sum |
| ' | transpose, e.g. A' |
| ^ | power, e.g. A ^ 2 |
| .^ | element by element power, e.g. A.^ 2 |

m-files.

An m-file is a file that contains a sequence of MATLAB commands. Some m-files are built into MATLAB. A user can create a new m-file using an editor. For example, an m-file called `fourier.m` could be created containing the lines:

```
%
% Plot a trigonometric function.
%
x = 0:.01:1;
y=sin(2*pi*x);
plot(x,y)
```

In this case, typing `fourier` would produce a plot of a sine curve. (Note: `%` in an m-file denotes a comment line.) In order to pass arguments to and from an m-file, the word “function” must be on the first line. For example:

```
function [x,y] = fourier(n,xmax)
%
% Plot a trigonometric function.
%
x=0:.01:xmax;
y=sin(n*pi*x);
plot(x,y)
```

Typing `[x,y] = fourier(2,7);` plots a sine curve. After execution, the vectors `x` and `y` are available for further calculations.

Useful commands.

| | |
|------------------|--|
| type fft | lists the contents of the m-file <code>fft.m</code> |
| save A | stores a matrix in a file called <code>A.mat</code> |
| save | saves all variables in a file called <code>matlab.mat</code> |
| load temp | retrieves all the variables from file <code>temp.mat</code> |
| print | prints the current graphics window |