

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/matlab.pdf>

<http://www.cyclismo.org/tutorial/matlab/>

The command for starting MATLAB depends of your system configuration (you can often start MATLAB on Unix by typing **matlab**). To obtain help from within MATLAB, type **help**; this provides a list of available functions. 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+000
format long	3.14159265358979
format long e	3.141592653589793e+000

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 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

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

Example of a loop.

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

Example of a conditional statement.

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

Plotting.

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

<code>x=A\b</code>	gives the solution of $Ax=b$
<code>[l,u]=lu(A)</code>	LU decomposition of A
<code>[v,d]=eig(A)</code>	eigenvalues in d , eigenvectors in v
<code>[u,s,v]=svd(A)</code>	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 (componentwise)
'	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:.01:1;
y=sin(2*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 dft	lists the content of the m-file <code>dft.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