Matlab Instructions

You can reach Matlab in the MIT server by clicking on the main menu (lower left icon with the foot- print), then Math/Plotting, then Matlab. Be patient – it takes Matlab several seconds to load. Wait for the prompt symbol: `>>`. You can leave Matlab by typing `exit` [return], or by closing the window. You can also access Matlab from a terminal window by typing:

`% add matlab` [return] `% matlab &` [return].

(The `&` is helpful but not necessary. With it you can use the xterm window from which Matlab was launched for other things.)

Matlab calculates with matrices and vectors and draws graphs in 2D and 3D. Skip the Introduction and Help documents; as preliminary practice, just read and carry out the following.

Entering matrices and vectors. In Matlab the variables represent matrices and vectors. The symbol `=` assigns the value on the right side of the equation to the symbol on the left. Type each of these lines in order, and see what you get. (Always hit [return] to end a line or command.)

`A = [1 2 3; 4 5 6; 7 8 9]` *(you can use commas instead of spaces: 1,2,3;)*
`b = [5 2 1]`
`b'` *(transpose: gives the column vector which Matlab calls* `[5;2;1]` *)*
`eye(3)` *(eye = I, the identity matrix)*

Try making a mistake: `C = [1,2,3; 4,5]`. To edit the mistake, press any of the four arrow keys to get the line back. (You can also prepare your commands in a text editor such as emacs and copy them with the mouse onto the Matlab command line.)

Operations with matrices and vectors

Sum, difference `A+B`, `A-B` (matrices must be same size)
Product `A*B` (matrices must be compatibly sized)
Powers `A^n` (A times itself n times; A must be square)
Transpose `A'`
Inverse `inv(A)` (or `A^-1`)

Try typing (use the values of A and b above): `A+eye(3)` `A*b` `A*(b')` `A*b'` `b*A`

Graphing with Matlab

Array operations. Recall that `*` and `^` are product and power operations *for matrices*. Adding a dot before `*` or `^` makes these operations act component-wise. So, if `x = [`$x_1\ x_2\ \ldots\ x_n$`]`, then

`exp(x) = [`$\exp(x_1)\ \ldots\ \exp(x_n)$`]` (similarly with `sin`, `cos`, `log`, etc.)
`x+y = [`$x_1+y_1\ \ldots\ x_n+y_n$`]` (similarly with `-`)
`x.*y = [`$x_1y_1\ \ldots\ x_ny_n$`]`
`x.^m = [`$x_1^m\ \ldots\ x_n^m$`]` (m can be zero)

Colon operator. This generates a vector with equally spaced entries; for example,

$[0:2:12] = [0\ 2\ 4\ 6\ 8\ 10\ 12]$; $[2:-.1:1.6] = [2.0\ 1.9\ 1.8\ 1.7\ 1.6]$

2D plot directions. Given $\mathbf{x} = [x_1\ x_2\ \ldots\ x_n]$, $\mathbf{y} = [y_1\ \ldots\ y_n]$,

`plot`(x, y) plots the n points (x_i, y_i), joined by solid line segments.

`plot`$(x, y, '--')$ plots the n points, joined by dashed line segments.

`plot`$(x, y, '*')$ plots the n points as individual stars (or dots or circles, etc).

`hold` toggles between on and off (at the start it's off); when off, a new plot erases the previous one; when on, the new plot is superimposed on the old one.

`print` gives a print-out of the current screen plot.

Try in order (press [return] after each command):

```
x=[0:.1:2]
plot(x,sin(x))
plot(x,cos(x),'*')
hold
plot(x,sin(x),'--')
hold
plot(x,4*x.^3)
```
(this plots $y = 4x^3$; note the need for the array operator)

You can also put graphs and scatter plots together without the hold command. The commands below graph the three functions $10x$, $10x^{1/2}$, $2x^{5/3}$. (With the semicolon at the end of each command Matlab won't print out all the numbers. The semicolon also permits you to put several commands on one line.)

```
x = [2:40:400];   w = [1:1:500];   b = 10*(w.^.5);   c = 2*(w.^(5/3));
plot(x,10*x, '*',w,b,w,c, '--');
```

Graphing with Matlab (continued)

3D Plot directions. To plot $z = f(x, y)$, you specify:

the grid (x_i, y_j) of lattice points: give the vectors $x = [x_1 \dots x_n]$ and $y = [y1 \dots y_n]$.

Example: To make a grid with spacing .1, over the interval $[-2, 2]$ on both axes, type (in what follows, $\gg$ is the matlab prompt; don't type it — type the semicolon at the end so Matlab won't print out all the numbers — remember [return] at the end)

$\gg x = [-2 : .1 : 2];$
$\gg y = [-2 : .1 : 2];$
$\gg [x, y] = \text{meshgrid}(x, y);$

the function $z = f(x, y)$ For example, to graph the function $f(x, y) = y^2 - x^2$, type

$\gg z = y.\hat{}\ 2 - x.\hat{}\ 2;$

plot the graph either as a mesh of lines, or as a filled-in surface (the color indicates the value of z, i.e., the height of the graph above the xy-plane); type first

$\gg \text{mesh}(x, y, z)$ then $\gg \text{surf}(x, y, z)$

change the viewpoint The default picture is shown at the right; to change the viewpoint (rotate left-right, or up-down), type

$\gg$ rotate3d

then place the mouse cursor in the graph region, hold down left button, move mouse, release button. The two numbers on the screen are the *azimuth*: angle in degrees from the negative y-axis to the line of sight, and the *elevation*, the angle in degrees from the xy-plane to the line of sight. To turn off rotation, type again: $\gg$ rotate3d

hidden lines Try typing: $\gg$ hidden (type it again to change back)

changing scale To change the x-axis scale to $[-4, 4]$, the y-axis to $[-5, 5]$, and the z-axis to $[-20, 20]$, type

$\gg \text{axis}([-4 \;\; 4 \;\; -5 \;\; 5 \;\; -20 \;\; 20])$

contour curves To get a 2D plot of level curves or a 3D plot with 20 contour curves, type

$\gg \text{contour}(x, y, z, 20)$ $\gg \text{contour3}(x, y, z, 20)$