PC-MATLAB PRIMER

This is intended as a guided tour through PCMATLAB. Type as you go and watch what happens.

```
>> 2*3
ans = 6
```
PCMATLAB uses several lines for the answer, but I've edited this to save space.

```
>> a=2*3
a = 6
```
Now I've saved the result in the variable "a"

```
>> a=2*3;
```
The semicolon ";" saves the result but without showing the answer.

```
>> j = sqrt (-1);
```
PCMATLAB doesn't have j stored as a variable. If you'll need it, define it this way. Now you can do complex multiplication:

```
>> (1+j) * (2 + j)
ans =   1.0000 + 3.0000i
```
and even mix multiplication and addition:

```
>> (1+j) * (2+j) + (3+j)
ans = 4.0000 + 4.0000i
```

Let's do some work with vectors. First define a vector going from 0 to 2pi (pi is stored in PCMATLAB in steps of 0.01 sec:

```
>> t = [0: 0.01: 2*pi];
```

Now let's compute cos (t) and plot t vs. cos (t):

```
>> c = cos (t);
>> plot (t, c)
```
Hit ENTER key to stop viewing plot. Type "shg" (show graphics to return. Try "grid" at the >> prompt to get an x-y grid.

What happens if you just say plot (c) like this:

```
>> plot (c)
```
The x-axis is i and y-axis is c_i.

Now let's define two vectors a and b and try several kinds of multiplication:

```
>> a = [1, 2]; b= [3, 4];
>> a.*b
ans =     3     8
```
a.*b is a vector c with $c_i = a_i\, b_i$

a*b is matrix multiplication but requires that dimensions match.

```
>> a*b
??? Error using ==> *
Inner matrix dimensions must agree.
```

So try using the transpose operator (')

```
>> a*b'
ans =    11
```

Of course! A is a row vector and b' is a column vector, so a*b' is the dot product of a and b. Same as sum (a.*b).

Copyright © Electrical Engineering Department, KFUPM.

If you've had linear algebra, you know that a'*b (a column vector times a row vector) should produce a 2x2 matrix:

```
>> a'*b
ans =      3     4
           6     8
```

It does!

Now let's define a 2x2 matrix (; separates rows):

```
>> a= [1 2; 3 4]

a =   1     2
      3     4
```

And take its inverse:

```
>> inv (a)
ans =   -2.0000    1.0000
         1.5000   -0.5000
```

Couldn't be simpler!

And (for linear algebra devotees) find its eigenvectors and eigenvalues:

```
>> [u, v] = eig (a)
u =     -0.8246   -0.4160
         0.5658   -0.9094

 v =    -0.3723         0
              0    5.3723
```

These are the eigenvectors

The diagonal entries are the eigenvalues

```
>> eig (a)
ans = -0.3723
       5.3723
```

returns just the eigenvalues:

To list all the variables in the current workspace, together with information about their size, bytes, class, type "whos":

```
>> whos
```

Name	Size	Bytes	Class
a	2x2	32	double array
ans	2x1	16	double array
b	1x2	16	double array
c	1x629	5032	double array
j	1x1	16	double array (complex)
t	1x629	5032	double array
u	2x2	32	double array
v	2x2	32	double array

```
Grand total is 1275 elements using 10208 bytes
```

Now, let's get rid of all these variables and check:

```
>> clear
>> whos
>>
```

Copyright © Electrical Engineering Department, KFUPM

Let's get the t vector back and see what happens if we do:

```
>> t = [0: 0.01: 2*pi]
>> plot (sin (t), cos(t))
>> plot (sin (3*t), cos (5*t))
```

To get the axes scaled as you want, type

```
axis([-1.5, 1.5, -1, 1)
```
sets scaling for the x- and y-axes on the current plot.

```
>> axis auto
```
return to autoscaling

Now let's generate a half-wave rectified sine wave and find its DC component (average value):

```
>> w0=120*pi;                (60 Hz)
>> t =     [0: 0.0001: 1/60]; delta t chosen for 100-200 points per period.
>> whos                      check:
```

Name	Size	Total	Complex
t	1 by 167	167	No
w0	1 by 1	1	No

```
>> f = sin (t);
>> f = [f (1:84), zeros (1, 83)];      Kills negative half:
>> plot (t, f)                         Plotted as a check. Uh-oh, I forgot w0! Easy to fix:

>> f = sin (w0*t);
>> f = [f (1:84), zeros (1, 83)];
>> plot (t, f)                         Now it checks. (Note the value of checking).
```

Here's how to simulate $1/T\int (f.dt)$

```
>> T = 1/60;                 T and t are different variables.
>> dt=0.0001;
>> a0=sum (f.*dt)/T

>> a0 =0.3183
```

PCMATLAB remembers your last input line and you can edit it using the up arrow key, INS (insert), DEL (delete), and typing over. This really speeds up corrections and repetitive calculations.

```
>> quit                Leaves MATLAB
```

Try typing the following at the >> prompt:

Help.	help roots	help	help poly
demo	help Isim	help nyquist	help bode
help impulse	help step		

Copyright © Electrical Engineering Department, KFUPM

Suggested exercise: Generate a full-wave rectified sine wave using f = abs (sin (w_0t)) with t as above. Plot it and check that it looks right. Then calculate the DC component and see if it is twice that found above.

For most problems you'll find it a lot easier to write .m files, programs executable by PCMATLAB. Look at the sample program below to see the basic syntax. Refer to the PCMATLAB manual for details. This program calculates finite approximations to

V(t) = 4/pi [cos (w0*t) – cos(3*w0*t) /3 + cos (5*w0*t) /5 …]

Try running it and seeing what it looks like for n=1, 3, 11, 21, 101. You call the program by saying "[t, v] = sq(11).

```
function [t, v] = sq(n)
% Fourier series to n-th harmonic
titl='Fourier approximation to square wave to';
titl=[titl, num2str(n)];
titl=[titl, '-th harmonic'];
%
%
t  =[-.25 : 0.005 : 1.25] ;
w0=2*pi ;
v  =cos(w0*t) ;
factor = 1 ;
for i= 3 : 2 :n
       factor = factor*(-1) ;
       v = v + factor*(1/i) *cos(i*w0*t) ;
       end
v = (4/pi) *v ;
clg
% clg Clears (erases) Graphics memory. Shg Shows Graphics.
plot (t, v)
grid
title (titl)
shg
pause
```

To help you in understanding, note the following:

% denotes a comment line

[start : increment : end] generates a vector starting at "start", ending at "end" and incrementing by "increment".
[0 : 0.25 : 1] generates the vector
(0, .25 .5, .75 1).

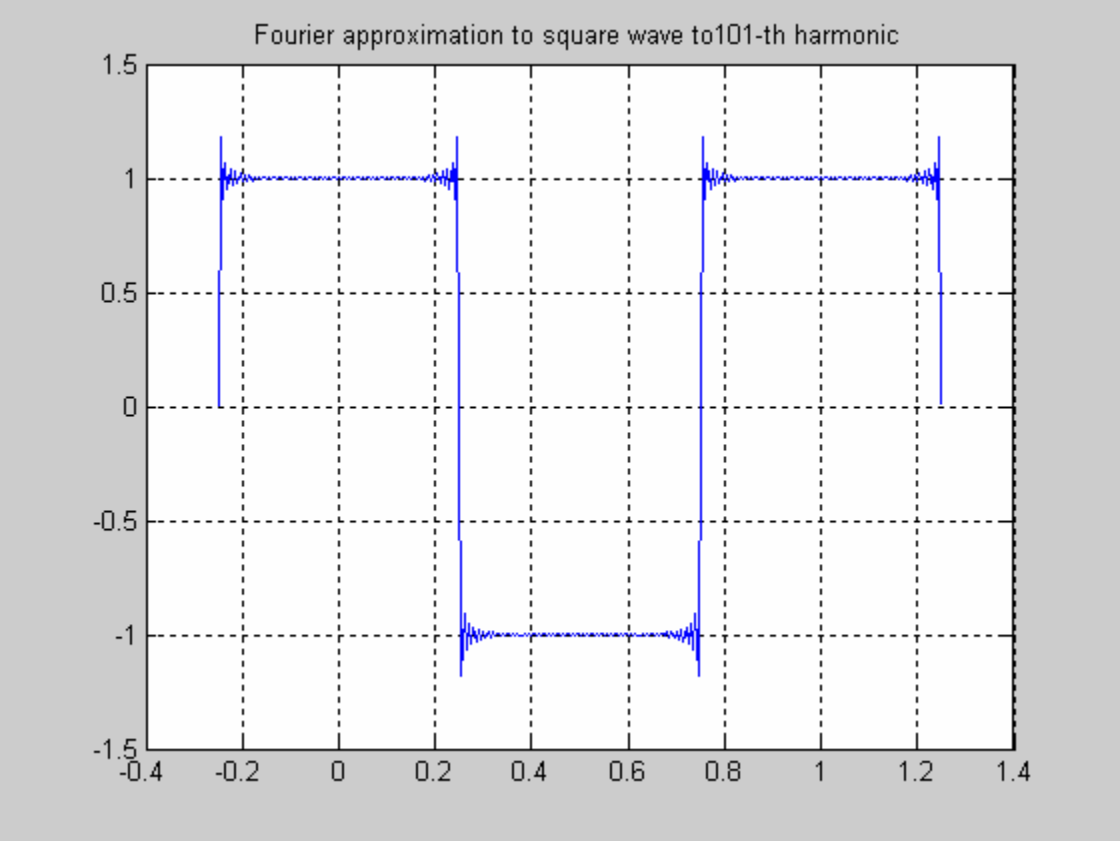
All of the lines involving the variable titl just generate the title for the graph. Look in reference section of PCMATLAB if you want to understand it.

Copyright © Electrical Engineering Department, KFUPM

v = cos (w0*t) does the following: each component of the vector t is multiplied by the scalar w0 and then the cos function is applied to each element of the resultant vector.

"for i = vector" executes the lines of code up to the "end" statement for i equal to each value of vector.

Fourier approximation to square wave to 101.000-th harmonic



Copyright © Electrical Engineering Department, KFUPM

Plotting Facilities

PLOT

Plot vectors or matrices. PLOT (X, Y) plots vector X versus Y. If X or Y is a matrix, then the vector is plotted versus the rows or columns of the matrix, whichever lines up. PLOT (X1, Y1, X2, Y2) is another way of producing multiple lines on the plot. PLOT (X1, Y1,':', X2, Y2, '+') uses a dotted line for the first curve and the point symbol + for the second curve. Other line and point types are:

solid	__	point	.	red	r
dashed	--	plus	+	green	g
dotted	:	star	*	blue	b
dashdot	-.	circle	o	white	w
		x-mark	x	invisible	i

PLOT (Y) plots the columns of Y versus their index. PLOT (Y) is equivalent to PLOT (real(Y), imag(Y) if Y is complex. In all other uses of PLOT, the imaginary part is ignored.

See SEMI, LOGLOG, POLAR, GRID, SHG, CLC, CLG, TITLE, XLABEL, YLABEL, AXIS, HOLD, MESH, CONTOUR, SUBPLOT.

POLAR

POLAR (THETA, RHO), makes a plot using polar coordinates of the angle THETA, in radians, versus the radius RHO.
See GRID for polar grid lines and PLOT for how to obtain multiple lines and different line-types.

XLABEL

XLABEL text, writes the text on the current plot beneath the x-axis.

YLABEL

YLABEL ('text') writes the text on the current plot beside the y-axis.

AXIS

Manual axis scaling on plots. Tying AXIS by itself freezes the current axis scaling for subsequent plots. Typing AXIS again resumes auto-scaling. AXIS returns a 4 element row vector containing the [x–min, x–max, y–min, y–max] used on the last plot.
AXIS (V) where V is a 4 element vector sets the axis scaling to the prescribed limits.
AXIS ('square') sets the plot to an approximately square ratio. In this mode, a line with slope 1 will be at a true 45 degrees, instead of skewed by the irregular shape of the screen. Thus PLOT (sin (t), cos (t)) will look like a circle instead of an oval. AXIS ('normal') sets the aspect ratio back to normal.

Copyright © Electrical Engineering Department, KFUPM

HOLD — Holds the current graph on the screen. Subsequent PLOT commands will add to the plot, using the already established axis limits, and retaining the previously plotted curves. HOLD ON turns on holding, HOLD OFF turns it off, and HOLD, by itself, toggles the HOLD state.

SUBPLOT — Graphing windows control. SUBPLOT(mnp), where, 'mnp' is a three digit number, breaks the graph window into an m-by-n matrix of small graph windows, and selects the p-th window for the current plot. For Example,

SUBPLOT (211), PLOT (1:3), SUBPLOT (212), PLOT (SIN (1 : 3))

Plots 1 : 3 on the top half of the screen and SIN (1 : 3) on the bottom half. After the first SUBPLOT command, subsequent plotting commands automatically advance to the next subplot window. So SUBPLOT (211), PLOT (1 : 3), PLOT (SIN (1 : 3)) achieves the same effect as above. CLG, SUBPLOT (111) or just SUBPLOT returns to the default single plot configuration.

CLC — Clear command window. See also HOME and CLG.

TITLE — TITLE ('text') writes the text at the top of the current plot as a title.

CONTOUR — CONTOUR surface. CONTOUR (Z) produces a contour plot of matrix using the values in Z as heights above a plane.
CONTOUR (Z, N) draws N contour lines, overriding the default automatic value.
CONTOUR (Z, V) draws LENGTH (V) contour lines at the locations specified in vector V.
CONTOUR (Z, N, X, Y) where X and Y are vectors, specifies the X- and Y- axes used on the plot. See also MESH.

MESH — Mesh surface. MESH (Z) produces a 3-dimensional mesh surface "picture" of matrix Z using the values in Z as heights above a plane. See MESHDOM to plot functions of two variables.

MESH (Z, M) specifies a view-point. The two-element vector M = [AS EL] contains AZ, the azimuth or horizontal rotation, and EL, the vertical elevation (both in degrees). Azimuth revolves about the z-axis, with positive values indicating counter-clockwise rotation of the view-point (clockwise rotation of the object). Positive values of elevation correspond to moving above the object; negative values move below. Here are some examples:

EL = 90 is directly overhead.
M = [0 0] looks directly up the first column of Z,
From the Z(m, l) element.
AZ = 180 is behind the matrix.
M = [-37.5 30] is the default

MESH (Z, S) and MESH (Z, M, S) control the scale factors used to set the X, Y and Z axes. The vector S is defined as S = [sx sy sz], where the three scalars, relative to each other, set the size of the object in each of the three dimensions. See also CONTOUR.

Copyright © Electrical Engineering Department, KFUPM