APPENDIX A

This section is intended to take you through a step-by-step procedure for plotting functions in MATLAB®. Try the commands in the command window of MATLAB®. A specific example is taken for this purpose. This example was introduced in chapter 2 of the thesis as example 2.1. The 2 signals are repeated here for convenience.

$$x1(t) = \sin(2*\pi*100*t) \qquad 0 <= t < 0.1 \text{ sec}$$

$$= \sin(2*\pi*500*t) \qquad 0.1 <= t < 0.2 \text{ sec}$$

$$x2(t) = \sin(2*\pi*500*t) \qquad 0 <= t < 0.1 \text{ sec}$$

$$= \sin(2*\pi*100*t) \qquad 0.1 <= t < 0.2 \text{ sec}$$

Solution :

MATLAB® always plots one vector (a vector is a 1xN matrix, 1-d array), say vector ‘y’ against another vector, say ‘t’. This vector ‘t’ is usually the independent variable and ‘y’ is the dependent variable. We may choose any number of points for the independent variable, and then plot the value of the dependent variable at these points.

Thus, in the given example, where ‘t’ ranges from 0 to 0.2 seconds, the first thing that needs to be done is to select some values for ‘t’ at which we need to compute and plot x1(t) and x2(t).

Note that here we need to plot analog signal, but in a digital computer , we can compute only at discrete values.

Thus, to give it a look of analog signal, we divide the interval of 0 to 0.2 seconds into a large number of points. I have chosen 10,000 points, on intuition. The following command is used for the purpose.

```
t = linspace(0,0.2,10000);
```

This creates a vector (an array) of size [1 x 10000]. Note that the index of this array ranges from 1 to 10000 , and not from 0 to 9999 as would have been the case in C-language.

Now, we need to compute the vectors x1 and x2 corresponding to x1(t) and x2(t) respectively. Since these functions have different mathematical relation at different intervals, we first need to divide the ‘t’ vector into different intervals.

For the given example, we divide 't' into 2 intervals ranging from (1 to 5000) and (5001 to 10000). This is done by the following commands.

```
t1 = t(1:5000);
t2 = t(5001:10000);
```

Now, we will compute the the signal x1(t) and x2(t) in the following 2 subsections

A.1 plotting x1(t):

x1(t) will be computed separately for 2 different intervals. Let us call these 2 values of x1(t) as 'x1a' and 'x1b'.

```
x1a = sin(2*pi*100*t1) ;
x1b = sin(2*pi*500*t2) ;
```

These 2 commands will create 2 vectors of size 5000 each, and they repersent the values of 'x1' for the intervals 0-0.1 seconds and 0.1-0.2 seconds respectively.

Now all that remains is to append 'x1a' and 'x1b' and plot the resulting vector against the vector 't'/

Following steps will do the job.

```
x1=[x1a x1b];

plot(t,x1);
```

If these values may be needed for future use or if this varible needs to be called from a different function / GUI, it is advisable to save this variable.

MATLAB® saves variables in a MAT file. (See the MATLAB® HELP for more information on MAT files).

The command to save variables in a MAT file is

```
save('c:\myvar1.mat','x1');
```

This command creates a MAT file called 'myvar1.mat' in the root directory of 'C' drive. You may specify any path of your choice.

A.2 plotting x2(t):

The procedure is same as in section A.1. I enlist here only the sequence of steps.

```
x2a = sin(2*pi*500*t1) ;
x2b = sin(2*pi*100*t2) ;

x2=[x2a x2b];

plot(t,x2);


save('c:\myvar2.mat','x2');
```

A.3 Computing and plotting FFT of these 2 signals.

In this section, we see how to compute FFT of signals in MATLAB® . FFT of x1(t) and x2(t) are computed and plotted, along with the original signal.

Since these 2 signals are of length 10000, we need to find the power of 2 which is greater than 10000. 16384 is such a number and we plot 16384-point FFT of these 2 signals.

The following code listing represents the sequence of steps to be followed. Note that in MATLAB the **'%'** denotes the user-comments and will be ignored. Read these comments and understand the procedure.

```
%this code will plot the 2 signals x1(t) , x2(t) and their FFTs
%note that here, instead of x1(t) and x2(t) we will call them
%x1(n) and x2(n)
%make sure that you have the MAT-files myvar1.mat  and myvar2.mat with the proper data

 load('c:\myvar1.mat');
%note that we saved the vector 'x1' in myvar1.mat
%when you give this load command, 'x1' is made  available in the workspace

%now compute the 16384-point fft of x1, store the result in x1fft
 x1fft=fft(x1,16384);

%for plotting the signal and its fft, we need discrete values on the x-axis
%time varies from 0 to  0.2 seconds, but for discrete data,
%we plot the values from n=1 to 10000
 n=1:10000;
 N=1:16384;

 subplot(2,1,1); %check the MATLAB help for details on subplot

 plot(n,x1)  %plot the signal x1(n)
 title('t=0 to 0.2 seconds. sinusoidal components are sin(2*pi*100t) and sin(2*pi*500t)');
 subplot(2,1,2);
 plot(N,abs(x1fft));
 title('Frequency response using FFT');

 figure;    % plot the signal x2(n) in a different window

%the procedure is now repeated for x2(n)
 load('c:\myvar2.mat');
 x2fft=fft(x2,16384);

 subplot(2,1,1);

 plot(n,x2);
 title('t=0 to 0.2 seconds . sinusoidal components are sin(2*pi*100t)  and sin(2*pi*500t)');
 subplot(2,1,2);
 plot(N,abs(x2fft));
 title('Frequency response using FFT');
```

PLEASE NOTIFY ANY INCONSISTENCY IN THIS APPENDIX AT THE FOLLOWING EMAIL
shivramangiri@yahoo.co.uk   shivpap@hotmail.com