Scilab Manual for
Digital Signal Processing
by Mr Rajesh B Raut
Electronics Engineering
Shri Ramdeobaba College of Engg & Mgmt[1]

Solutions provided by
Mr. Vipul S Lande
Electronics Engineering
Shri Ramdeobaba College of Engg & Mgmt, Nagpur

August 2, 2025

[1]Funded by a grant from the National Mission on Education through ICT, http://spoken-tutorial.org/NMEICT-Intro. This Scilab Manual and Scilab codes written in it can be downloaded from the "Migrated Labs" section at the website http://scilab.in

Contents

List of Experiments

List of Figures

Experiment: 1

Basic operations on sequences and plotting them in continuous/discrete form.

Scilab code Solution 1.0 Basic operations on sequences and plotting them in continuous form

```
//Experiment-1.1
//Basic operations on sequences and plotting them in
    continuous form.

clear;
clc;
close;

//unit step
N=input('enter time of unit step sequence: ');//N=20
    (only intiger value)
n=0:0.5:N-1;
NL=length(n)
y=ones(1,NL);
subplot(2,2,1);
plot(n,y);
```

```
title('UNIT STEP');
xlabel('Time-->'); ylabel('AMPL-->');

//ramp
N=input('enter time of ramp sequence: ');//N=20 (only intiger value)
n=0:0.5:N-1;
subplot(2,2,2);
plot(n,n);
title('RAMP');
xlabel('Time-->'); ylabel('AMPL-->');

//exponential exp(a*t)
N=input('enter time of expo. sequence: ');//N=20 (only intiger value)
n=0:0.5:N-1;
a=input('enter value of a= ');//a=0.03
y=exp(a*n);
subplot(2,2,3);
plot(n,y);
title('EXPONENTIAL');
xlabel('Time-->'); ylabel('AMPL-->');

//sine wave
f=input('enter frequency of sine sequence: ');//f=5 (only intiger value)
t=0:0.001:1;
y=sin(2*%pi*f*t);
figure(1);
subplot(3,2,1);
plot(t,y);
title('SINE');
xlabel('Time-->'); ylabel('AMPL-->');

//cosine wave
f2=input('enter frequency of cosine sequence: ');//f=5 (only intiger value)
t=0:0.001:1;
```

```
y2=cos(2*%pi*f2*t);
subplot(3,2,2);
plot(t,y2);
title('COSINE');
xlabel('Time—>'); ylabel('AMPL—>');

//sine + cosine wave
y3=cos(2*%pi*f2*t)+sin(2*%pi*f*t);
subplot(3,2,3);
plot(t,y3);
title('SINE+COSINE');
xlabel('Time—>'); ylabel('AMPL—>');

//sine x sine wave
y4=y.*y2;
subplot(3,2,4);
plot(t,y4);
title('SINE x COSINE');
xlabel('Time—>'); ylabel('AMPL—>');
```

Scilab code Solution 1.1 Basic operations on sequences and plotting them in discrete form

```
//Experiment -1.2
//Basic operations on sequences and plotting them in
    discrete form.

clear;
clc;
close;

//impulse
N=-3:1:3;
y=[zeros(1,3),ones(1,1),zeros(1,3)];
subplot(2,2,1);
```

```
plot2d3(N,y);
title('IMPULSE');
xlabel('N'); ylabel('AMPL-->');

//unit step
N=input('enter length of unit step sequence: '); //N=20 (only intiger value)
n=0:1:N-1;
y=ones(1,N);
subplot(2,2,2);
plot2d3(n,y);
title('UNIT STEP');
xlabel('N'); ylabel('AMPL-->');

//ramp
N=input('enter length of ramp sequence: ');//N=20 (only intiger value)
n=0:1:N-1;
subplot(2,2,3);
plot2d3(n,n);
title('RAMP');
xlabel('N'); ylabel('AMPL-->');

//exponential exp(a*t)
N=input('enter length of expo. sequence: ');//N=20 (only intiger value)
n=0:1:N-1;
a=input('enter value of a= ');//a=0.03
y=exp(a*n);
subplot(2,2,4);
plot2d3(n,y);
title('EXPONENTIAL');
xlabel('N'); ylabel('AMPL-->');

//sine wave
N=input('enter length of sine sequence: ');//N=60 (only intiger value)
n=0:1:N-1;
```

```
y=sin(0.3*%pi*n);
figure(1);
subplot(3,1,1);
plot2d3(n,y);
title('SINE');
xlabel('N'); ylabel('AMPL-->');

//cosine wave
N=input('enter length of cosine sequence: ');//N=60 (only intiger value)
n=0:1:N-1;
y=cos(0.2*%pi*n);
subplot(3,1,2);
plot2d3(n,y);
title('COSINE');
xlabel('N'); ylabel('AMPL-->');

//sine + cosine wave
N=input('enter length of cosine/sine sequence: ');//N=80 (only intiger value)
n=0:1:N-1;
y=cos(0.2*%pi*n)+sin(0.3*%pi*n);
subplot(3,1,3);
plot2d3(n,y);
title('SINE+COSINE');
xlabel('N'); ylabel('AMPL-->');
```

Scilab code Solution 1.2 Basic operations on sequences Delaying and Advancing

```
//Experiment-1.3
//Basic operations on sequences and plotting them in continuous form.
clc;
clear;
```

```
close;
x = input('Enter the input sequence:=')//x =[1 2 3 -2 3]
m = length(x);
lx = input('Enter the starting index of original signal:=')//lx = -2
hx = lx+m-1;
n = lx:1:hx;
subplot(3,1,1)
a = gca();
a.x_location = "origin";
a.y_location = "origin";
a.data_bounds = [-10,0;10,10];
plot2d3('gnn',n,x);
xlabel('n===>')
ylabel('Amplitdue---->')
title('Original Sequence')
//
d = input('Enter the delay:=')// d = 1
n = lx+d:1:hx+d;
subplot(3,1,2)
a = gca();
a.x_location = "origin";
a.y_location = "origin";
a.data_bounds = [-10,0;10,10];
plot2d3('gnn',n,x)
xlabel('n===>')
ylabel('Amplitude---->')
title('Delayed Sequence')
//
a = input('Enter the advance:=')// a = 2
n = lx-a:1:hx-a;
subplot(3,1,3)
a = gca();
a.x_location = "origin";
a.y_location = "origin";
a.data_bounds = [-10,0;10,10];
plot2d3('gnn',n,x)
```

```
xlabel('n===>')
ylabel('Amplitude--->')
title('Advanced Sequence')
```

Experiment: 2

Analyzing the effect of Sampling of continuous time signal to avoid aliasing.

Scilab code Solution 2.0 Analyzing the effect of sampling of continuous time signal to avoid aliasing

```
//Experiment - 2
//Analyzing the effect of sampling of continuous
   time signal to avoid aliasing.

clear;
clc;
close;


// Aliasing

f = 60; // signal freq. in Hz
tmin = -0.05;
tmax = 0.05;
t = linspace(tmin, tmax, 400);
alpha = 1;
```

```
x_c = cos(alpha * 2 * %pi * f * t);
figure;
plot(t, x_c)

Fs = 200 // for 400, 200, 120, 70, 40 Hz

T = 1/Fs;
nmin = ceil(tmin/T);
nmax = floor(tmax/T);
n = nmin:nmax;
x_s = cos(2*%pi*f*n*T);
figure;
plot(t, x_c)
mtlb_hold("on")
plot(n*T, x_s, '.')
mtlb_hold("off")

figure;
plot(n*T, x_s, '-.');

//viewing spectrum of signals
t=soundsec(0.5);
s1=sin(2*%pi*60*t);// signal frequency=60Hz
s2=sin(2*%pi*200*t);// sampling frequency=200Hz
analyze(s1,10,600,22050)
analyze(s2,10,600,22050)
title('Spectrum of signals')
```

Experiment: 3

Convolving two sequences in discrete time domain in discrete frequency domain.

Scilab code Solution 3.0 Convolving two sequences in discrete time domain

```
//Experiment - 3
//Convolving two sequences in discrete time domain.


clc;
clear ;
close ;
x=input('enter i/p x(n):');// x=[1 2 3 4 5 6]
m=length(x);
h=input('enter i/p  h(n):');// h=[1 -1 1]
n=length(h);

subplot(2,2,1),
p=0:1:m-1;

a = gca ();
```

```
a.x_location = "origin";
a.y_location = "origin";
plot2d3('gnn',p,x)

title('i/p sequencce x(n)is:');
xlabel('-->n');


x=[x,zeros(1,n)];

subplot(2,2,2),
q=0:1:n-1;

a = gca ();
a.x_location = "origin";
a.y_location = "origin";
plot2d3('gnn',q,h)
title('i/p sequencce h(n)is:');
xlabel('-->n');


h=[h,zeros(1,m)];

disp('convolution of x(n) & h(n) is y(n):');
y=zeros(1,m+n-1);
for i=1:m+n-1
    y(i)=0;
    for j=1:m+n-1
        if(j<i+1)
            y(i)=y(i)+x(j)*h(i-j+1);
        end
    end
end
y
subplot(2,2,3)
r=0:1:m+n-2;

a = gca ();
```

```
a.x_location = "origin";
a.y_location = "origin";
plot2d3('gnn',r,y)


title('convolution of x(n) & h(n) is :');
xlabel('—>n');
```

Experiment: 4

Evaluating circular convolution using DFT-IDFT method.

Scilab code Solution 4.0 Evaluating circular convolution using DFT IDFT method

```
//Experiment - 4
//Evaluating circular convolution using DFT-IDFT
   method.

clear ;
clc;
close ;
x=input('enter 1st seq:');// x=[1 2 3 4 5 6]
h=input('enter 2nd seq:');// h=[1 -1 1]


dif =abs( max(size(x))- max(size(h)));
if(dif>0)
  h(max(size(h))+dif) = 0;
else
  x(max(size(x))+dif) = 0;
end

```

```

X=fft(x,-1);
H=fft(h,-1);
Y=X.*H;
y=ifft(Y);

subplot(3,2,1);
n=0:1:length(x)-1
plot2d3(n,x);
ylabel('amplitude——>');
xlabel('x(n)      n——>');

subplot(3,2,2);
plot2d3(n,abs(X));
ylabel('amplitude——>');
xlabel('X(k)      k——>');

subplot(3,2,3);
n=0:1:length(h)-1
plot2d3(n,h);
ylabel('amplitude——>');
xlabel('h(n)      n——>');

subplot(3,2,4);
plot2d3(n,abs(H));
ylabel('amplitude——>');
xlabel('H(k)      k——>');

subplot(3,2,5);
n=0:1:length(y)-1
plot2d3(n,y);
ylabel('amplitude——>');
xlabel('y(n)      n——>');

subplot(3,2,6);
plot2d3(n,abs(Y));
ylabel('amplitude——>');
xlabel('Y(k)      k——>');
```

```
disp(y,'the circular conv. resultant signal is');
```

Experiment: 5

Designing of FIR filters for low pass, high pass and band reject response.

Scilab code Solution 5.1 Design an Low Pass FIR Filter

```
//Experiment - 5
//Designing of FIR filters for low pass, high pass,
   band pass and band reject response.

// To Design an Low Pass FIR Filter
//Filter Length =5, Order = 4
//Window = Rectangular Window

clc;
clear;
xdel(winsid());
fc = input("Enter Analog cutoff freq. in Hz=")//250
fs = input("Enter Analog sampling freq. in Hz=")//
   2000
M = input("Enter order of filter =")//4
w = (2*%pi)*(fc/fs);
disp(w,'Digital cutoff frequency in radians.cycles/
```

```
samples');
wc = w/%pi;
disp(wc,'Normalized digital cutoff frequency in cycles/samples');
[wft,wfm,fr]=wfir('lp',M+1,[wc/2,0],'re',[0,0]);
disp(wft,'Impulse Response of LPF FIR Filter:h[n]=');
//Plotting the Magnitude Response of LPF FIR Filter
subplot(2,1,1)
plot(2*fr,wfm)
xlabel('Normalized Digital Frequency w——>')
ylabel('Magnitude |H(w)|=')
title('Magnitude Response of FIR LPF')
xgrid(1)
subplot(2,1,2)
plot(fr*fs,wfm)
xlabel('Analog Frequency in Hz f ——>')
ylabel('Magnitude |H(w)|=')
title('Magnitude Response of FIR LPF')
xgrid(1)
```

Scilab code Solution 5.2 To Design an High Pass FIR Filter

```
//Experiment - 5
//Designing of FIR filters for low pass, high pass, band pass and band reject response.

// To Design an High Pass FIR Filter
//Filter Length =5, Order = 4
//Window = Rectangular Window

clc;
clear;
xdel(winsid());
fc = input("Enter Analog cutoff freq. in Hz=")//250
```

```
fs = input("Enter Analog sampling freq. in Hz=")//2000
M = input("Enter order of filter =")//4
w = (2*%pi)*(fc/fs);
disp(w,'Digital cutoff frequency in radians.cycles/samples');
wc = w/%pi;
disp(wc,'Normalized digital cutoff frequency in cycles/samples');
[wft,wfm,fr]=wfir('hp',M+1,[wc/2,0],'re',[0,0]);
disp(wft,'Impulse Response of HPF FIR Filter:h[n]=');
//Plotting the Magnitude Response of HPF FIR Filter
subplot(2,1,1)
plot(2*fr,wfm)
xlabel('Normalized Digital Frequency w——>')
ylabel('Magnitude |H(w)|=')
title('Magnitude Response of FIR HPF')
xgrid(1)
subplot(2,1,2)
plot(fr*fs,wfm)
xlabel('Analog Frequency in Hz f ——>')
ylabel('Magnitude |H(w)|=')
title('Magnitude Response of FIR HPF')
xgrid(1)
```

Scilab code Solution 5.3 To Design Band Pass FIR Filter

```
//Experiment - 5
//Designing of FIR filters for low pass, high pass, band pass and band reject response.

// To Design an Band Pass FIR Filter
//Filter Length =5, Order = 4
//Window = Rectangular Window
```

```
clc;
clear;
xdel(winsid());
fc1 = input("Enter Analog lower cutoff freq. in Hz=")//250
fc2 = input("Enter Analog higher cutoff freq. in Hz=")//600
fs = input("Enter Analog sampling freq. in Hz=")//2000
M = input("Enter order of filter =")//4
w1 = (2*%pi)*(fc1/fs);
w2 = (2*%pi)*(fc2/fs);
disp(w1,'Digital lower cutoff frequency in radians.cycles/samples');
disp(w2,'Digital higher cutoff frequency in radians.cycles/samples');
wc1 = w1/%pi;
wc2 = w2/%pi;
disp(wc1,'Normalized digital lower cutoff frequency in cycles/samples');
disp(wc2,'Normalized digital higher cutoff frequency in cycles/samples');
[wft,wfm,fr]=wfir('bp',M+1,[wc1/2,wc2/2],'re',[0,0]);
disp(wft,'Impulse Response of BPF FIR Filter:h[n]=');
//Plotting the Magnitude Response of HPF FIR Filter
subplot(2,1,1)
plot(2*fr,wfm)
xlabel('Normalized Digital Frequency w--->')
ylabel('Magnitude |H(w)|=')
title('Magnitude Response of FIR BPF')
xgrid(1)
subplot(2,1,2)
plot(fr*fs,wfm)
xlabel('Analog Frequency in Hz f --->')
ylabel('Magnitude |H(w)|=')
```

```
title('Magnitude Response of FIR BPF')
xgrid(1)
```

Scilab code Solution 5.4 To Design Band Stop FIR Filter

```
//Experiment - 5
//Designing of FIR filters for low pass, high pass, band pass and band reject response.

// To Design an Band Stop FIR Filter
//Filter Length =5, Order = 4
//Window = Rectangular Window

clc;
clear;
xdel(winsid());
fc1 = input("Enter Analog lower cutoff freq. in Hz=")//250
fc2 = input("Enter Analog higher cutoff freq. in Hz=")//600
fs = input("Enter Analog sampling freq. in Hz=")//2000
M = input("Enter order of filter =")//4
w1 = (2*%pi)*(fc1/fs);
w2 = (2*%pi)*(fc2/fs);
disp(w1,'Digital lower cutoff frequency in radians.cycles/samples');
disp(w2,'Digital higher cutoff frequency in radians.cycles/samples');
wc1 = w1/%pi;
wc2 = w2/%pi;
disp(wc1,'Normalized digital lower cutoff frequency in cycles/samples');
disp(wc2,'Normalized digital higher cutoff frequency in cycles/samples');
```

```
[wft,wfm,fr]=wfir('sb',M+1,[wc1/2,wc2/2],'re',[0,0])
    ;
disp(wft,'Impulse Response of BSF FIR Filter:h[n]=')
    ;
//Plotting the Magnitude Response of HPF FIR Filter
subplot(2,1,1)
plot(2*fr,wfm)
xlabel('Normalized Digital Frequency w——>')
ylabel('Magnitude |H(w)|=')
title('Magnitude Response of FIR BSF')
xgrid(1)
subplot(2,1,2)
plot(fr*fs,wfm)
xlabel('Analog Frequency in Hz f ——>')
ylabel('Magnitude |H(w)|=')
title('Magnitude Response of FIR BSF')
xgrid(1)
```

Experiment: 6

Designing of Butterworth IIR filters and its realisation to filter out noise from a given signal.

Scilab code Solution 6.0 Designing of butterworth IIR filters and its realization to filter our noise from a given signal

```
//Experiment - 6
//Designing of butterworth IIR filters and its
   realization to filter our noise from a given
   signal.



t = 0:0.001:4;
signal = sin(2*%pi*20*t);
noise = sin(2*%pi*50*t);
compound = signal + noise;

playsnd(compound);

```

```
myfilter = iir(15,'lp','butt',[0.025 0],[0 0]); //
    butt, cheb1/2, ellip

output = flts(compound,myfilter);

halt('Press Enter');

playsnd(output);

// Sound difference in filtered sound
xsetech([0,0,1,1/2]);
plot2d(t,compound);
xtitle('input signal');
xsetech([0,1/2,1,1/2]);
plot2d(t,output);
xtitle('output signal');
```

Experiment: 7

Designing of Chebychev/inverse Chebychev/elliptical filter from a given transfer function.

Scilab code Solution 7.0 Designing of chebychev inverse chebyshev elliptical filter from a given transfer function

```
//Experiment - 7
//Designing of chebychev/inverse chebyshev/elliptical filter from a given transfer function.

clc;
clear all;

cheblpf = iir(2,'lp','cheb1',[0.3 0],[0.1 0]);
[hzm,fr]=frmag(cheblpf,256);
figure;
plot2d(fr',hzm')
title('Chebysheve Lowpass Filter')


```

```
[cells,fact,zzeros,zpoles]=eqiir('lp','ellip',[%pi*3/8,%pi*3.5/8],0.1,0.01);
h=fact*poly(zzeros,'z')/poly(zpoles,'z');
[hzm,fr]=frmag(h,256);
figure;
plot2d(fr',hzm')
title('Elliptical Lowpass Filter')
[valcoeff, filtamp, filtfreq] = wfir ('lp', 20, [.2 0], 'hm', [0 0]);
figure;
plot2d(filtfreq,filtamp)
title('FIR-Hamming window Lowpass Filter')
hn=eqfir(33,[0 .2;.25 .35;.4 .5],[0 1 0],[1 1 1]);
[hm,fr]=frmag(hn,256);
figure;
plot2d(fr,hm)
title('FIR-Multiband Filter response')
```

Experiment: 8

Application of sound effect on wave file like Flanging, Echo and Equiliser.

Scilab code Solution 8.0 Application of sound effect on wave file like Flanging Echo and Equalizer

```

//Experiment - 8
//Application of sound effect on wave file like
    Flanging, Echo and Equalizer.


// The "dsp01.wav" file should be placed in current
    working directory
// The code can also be executed with any other
    small .wav file of duration 3-5 seconds.
clc;
clear;
clear all;

[y,fs] = wavread('dsp01.wav');
playsnd(y,fs);
```

```
halt('Original');

//Flanging
z = 0;
n = 1:length(y);
z(n) = y(n);
m = 11:length(y);
x(m) = z(m) + 0.8*z(m-10).*cos(2*%pi*m/fs);
playsnd(x,fs);
halt('Flange');


//Echo
clc;
clear all;
a = 0.5;
[l,fs] = wavread('dsp01.wav');
len = length(l);
delay = 0.4;
D = ceil(fs*delay);
m = zeros(max(size(l)));
for i = D+1:len
  m(i) = l(i) + a*l(i-D);
end

playsnd(m,fs);


halt('Echo');

//Equalizer
//For Low frequencies
clc;
clear all;

[h,fs] = wavread('dsp01.wav');
playsnd(h,fs);
```

```
halt('Original');

j = iir(3,'bp','butt',[.01 .05],[0.03 0.03]);

k = flts(h(1,:),j);
playsnd(k,fs);
halt('Low');

// For Mid frequencies
clc;
clear all;

[g,fs] = wavread('dsp01.wav');
playsnd(g,fs);
halt('Original');

j = iir(3,'bp','ellip',[.05 .10],[.08 .03]);

k = flts(g(2,:),j);
sound(k,fs);

halt('Mid');

//For High frequencies
clc;
clear all;

[h,fs] = wavread('dsp01.wav');
playsnd(h,fs);
halt('Original');

j = iir(3,'bp','butt',[.15 .25],[0.03 0.03]);

k = flts(h(1,:),j);
playsnd(k,fs);

halt('High');
```

Experiment: 9

Polyphase decomposition by decimation method for a given decimation factor.

Scilab code Solution 9.0 Polyphase decomposition by decimation method for a given decimation factor

```
//Experiment - 9
//Polyphase decomposition by decimation method for a
    given decimation factor.

clear;
clc;
close;
M =input('Enter the filter length(M)= '); //Filter
   Length = 30
D =input('Decimation Factor(D) ='); //Decimation
   Factor = 2
Ws =input('Enter stopband frequency(Ws)= '); //
   Stopband Edge Frequency = 0.31
Wc = %pi/2; //Cutoff Frequency
Wp = Wc/(2*%pi); //Passband Edge Frequency

```

```
hn=eqfir(M,[0 Wp;Ws .5],[1 0],[2 1]);
disp(hn,'The LPF Filter Coefficients are:')
//Obtaining Polyphase Filter Coefficients from hn
p = zeros(D,M/D);
for k = 1:D
  for n = 1:(length(hn)/D)
    p(k,n) = hn(D*(n-1)+k);
  end
end
disp(D,'For the given Decimation factor = ')
disp(p,'The Polyphase Decimator are:')
```

Experiment: 10

Spectral estimation of a sequence using Periodogram method.

Scilab code Solution 10.0 Spectral estimation of a sequence using Periodogram method

```
//Experiment - 10
//Spectral estimation of a sequence using Periodogram method.

clear;
clc;
close;
x=input('Input the signal x(n) = ');// x=[0.1 2.6 -1.1 3.2 2 -8 1.2 6]
N =length(x);
X = dft(x,-1);
Pxx = (1/N)*(abs(X).^2); //Peridogram Estimate
disp(X,'DFT of x(n)is X(k)=')
disp(Pxx,'Peridogram of x(n) is Pxx(k/N)=')
figure(1)
a = gca();
```

```
a.data_bounds =[0,0;8,11];
plot2d3('gnn',[1:N],Pxx)
a.foreground = 5;
a.font_color = 5;
a.font_style = 5;
title('Peridogram Estimate')
xlabel('Discrete Frequency Variable K ——>')
ylabel('Periodogram Pxx (k /N) ——>')
```

Experiment: 11

Real-time data acquisition and plotting through external hardware interfaced through serial port.

Scilab code Solution 11.0 Real time Data acquisition and plotting through external hardware interfaced through serial port

```
//Experiment - 11
//Real time Data acquisition and plotting through
   external hardware interfaced through serial port.

// This code requires serial toolbox installed on
   scilab.
// The code will execute only when Data acquisition
   hardware is connected on serial port(COM) of
   computer.

clear;
clc;
```

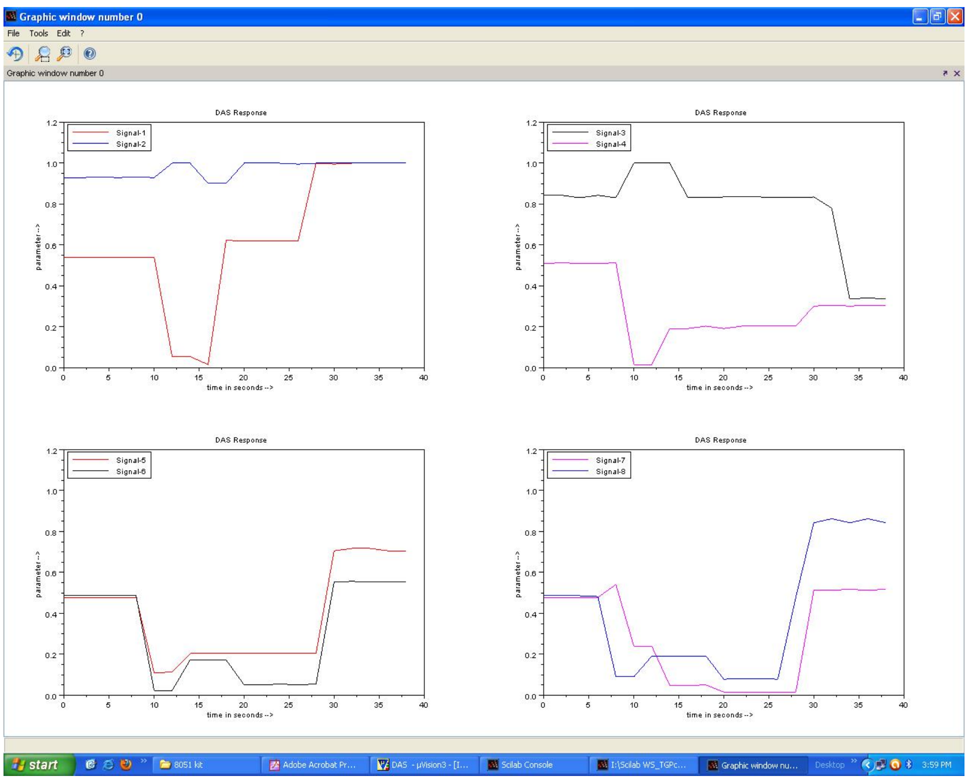

Figure 11.1: Real time Data acquisition and plotting through external hardware interfaced through serial port

```
close;

//s = openserial(1,"9600,n,8,1","binary","dtrdsr");
s = openserial(1,"9600,n,8,1","binary","none");
ns=input('Samples to be plotted: ');// 50
ts=input('Sampling time(sec): ');//2 sec
xpause(20000);

signal1=zeros(1,ns);
signal2=zeros(1,ns);
signal3=zeros(1,ns);
signal4=zeros(1,ns);
signal5=zeros(1,ns);
signal6=zeros(1,ns);
signal7=zeros(1,ns);
signal8=zeros(1,ns);

t=0:1*ts:ts*ns-1;
j=0;
clf()

    result = writeserial(s,"OK");//
    buf = readserial(s, [17]);
    xpause(200000);

for i=1:1:ns,
    result = writeserial(s,"OK");//
  //  xpause(6000);//66941usec after tx will begin
    buf = readserial(s, [17]);
    rbuf = str2code(buf);

    signal1(i)=abs((1/255)*(rbuf(2)-100));
    signal2(i)=abs((1/255)*(rbuf(4)-100));
    subplot(2,2,1)
    plot(t(i-j:i),signal1(i-j:i),'-r',t(i-j:i),signal2(i-j:i),'-b')
    a=gca();
    a.data_bounds=[0 -0.1;ts*ns-1 1.2];
```

```
    h = legend('Signal-1','Signal-2',2);
    xlabel('time in seconds -->')
    ylabel('parameter -->')
    title('DAS Response')
    set(gca(),"auto_clear","off")

    signal3(i)=abs((1/255)*(rbuf(6)-100));
    signal4(i)=abs((1/255)*(rbuf(8)-100));
    subplot(2,2,2)
    plot(t(i-j:i),signal3(i-j:i),'-k',t(i-j:i),
       signal4(i-j:i),'-m')
    a=gca();
    a.data_bounds=[0 -0.1;ts*ns-1 1.2];
    h = legend('Signal-3','Signal-4',2);
    xlabel('time in seconds -->')
    ylabel('parameter -->')
    title('DAS Response')
    set(gca(),"auto_clear","off")

    signal5(i)=abs((1/255)*(rbuf(10)-100));
    signal6(i)=abs((1/255)*(rbuf(12)-100));
    subplot(2,2,3)
    plot(t(i-j:i),signal5(i-j:i),'-r',t(i-j:i),
       signal6(i-j:i),'-k')
    a=gca();
    a.data_bounds=[0 -0.1;ts*ns-1 1.2];
    h = legend('Signal-5','Signal-6',2);
    xlabel('time in seconds -->')
    ylabel('parameter -->')
    title('DAS Response')
    set(gca(),"auto_clear","off")

    signal7(i)=abs((1/255)*(rbuf(14)-100));
    signal8(i)=abs((1/255)*(rbuf(16)-100));
    subplot(2,2,4)
    plot(t(i-j:i),signal7(i-j:i),'-m',t(i-j:i),
       signal8(i-j:i),'-b')
    a=gca();
```

```
    a.data_bounds=[0 -0.1;ts*ns-1 1.2];
    h = legend('Signal-7','Signal-8',2);
    xlabel('time in seconds -->')
    ylabel('parameter -->')
    title('DAS Response')
    set(gca(),"auto_clear","off")
    sleep(ts*1000);//wait for ts sec.
    j=j+1;
    clear buf
end
closeserial(s);
clear all
clc
disp("Required Data Plotted")
```