Scilab Manual for Digital Signal Processing by Prof K Jeevan Reddy Electronics Engineering Sreenidhi Institute Of Science And Technology[1]

Solutions provided by
Prof K Jeevan Reddy
Electronics Engineering
Sreenidhi Institute Of Science And Technology

November 24, 2026

[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 Scilab Solutions 3

1 Analog To Digital Filter Conversion using impulse invariance method 4

2 Analog To Digital Filter Conversion using bilinear transformation method 6

3 Prototype Analog Low Pass Filter to High Pass Filter Conversion 8

4 Prototype Analog Low Pass Filter to Band Pass Filter Conversion 11

5 Prototype Analog Low Pass Filter to Band Rejection/Stop Filter Conversion 14

6 Comparision of FIR filter design using windowing method 18

7 Detection of DTMF signal 24

8 Implementation of Decimation 28

9 Implementation of Interpolation 29

10 FIR Filter using Frequency Sampling Method 31

List of Experiments

Solution 1.1	Analog to Digital Filter Conversion	4
Solution 2.2	Analog to Digital Filter Conversion	6
Solution 3.1	Analog LPF to HPF Conversion	8
Solution 4.1	Analog LPF to BPF Conversion	11
Solution 5.1	Analog LPF to BSF Conversion	14
Solution 6.1	Rectangular Window	18
Solution 6.2	Kaiser Window	19
Solution 6.3	Hamming Window	21
Solution 6.4	Hanning Window	22
Solution 7.1	DTMF Decoding	24
Solution 8.1	Decimation	28
Solution 9.1	Interpolation	29
Solution 10.1	Frequency Sampling Method	31

Experiment: 1

Analog To Digital Filter Conversion using impulse invariance method

Scilab code Solution 1.1 Analog to Digital Filter Conversion

```
1  //Filter Conversion using Impulse Invariance Method
2  //OS:Windows 10
3  //Scilab 5.5.2
4
5  clear all;
6  clc;
7  close;
8
9  s=%s;
10 T =1;
11 HS =(2)/(s^2+3*s+2) ;
12 elts = pfss(HS);
13 disp(elts,'Factorized HS=');
14
15 //The poles associated are -2 and -1
16 p1 = -2;
17 p2 = -1;
```

```
18 z=%z;
19 HZ =(2/(1-%e^(p2*T)*z^(-1)))-(2/(1-%e^(p1*T)*z^(-1))
      )
20 disp(HZ,'HZ=');
```

Experiment: 2

Analog To Digital Filter Conversion using bilinear transformation method

Scilab code Solution 2.2 Analog to Digital Filter Conversion

```
1 //Filter Conversion using Bilinear Transformation
    Method
2 //OS:Windows 10
3 //Scilab 5.5.2
4
5 clear all;
6 clc;
7 close;
8
9 ap = input('Enter value of ap in dB')
10 as = input('Enter value of as in dB')
11 fp = input('Enter value of fp in Hz')
12 fs = input('Enter value of fs in Hz')
13 f = input('Enter value of f')
14
15 T=1/f;
16 wp =2*%pi*fp;
```

```
17 ws =2*%pi*fs;
18 op =2/T*tan(wp*T/2);
19 os =2/T*tan(ws*T/2);
20 N= log(sqrt((10^(0.1*as)-1)/(10^(0.1*ap)-1)))/log(op
   /os);
21 disp(ceil(N),' Order of the filter , N =');
22 s=%s;
23 HS =1/(s+1) // Transfer Function for N=1
24 oc=op //rad/sec
25 HS1 = horner(HS,oc/s);
26 disp(HS1,'Normalized Transfer Function , H(s) =');
27 z=%z;
28 HZ= horner(HS,(2/T)*(z-1)/(z+1));
29 disp(HZ,'H(z)=');
30
31 //Example Input
32 //Enter value of ap in dB 3
33 //Enter value of as in dB 10
34 //Enter value of fp in Hz 1000
35 //Enter value of fs in Hz 350
36 //Enter value of f 5000
37 //Order of the filter , N =   1.
38 // Normalized Transfer Function , H(s) =
39 //          s
40 //   ---------------
41 //
42 //    7265.4253 + s
43 // H(z)=
44 //         1 + z
45 //   ----------------
46 //
47 //    -9999 + 10001z
```

Experiment: 3

Prototype Analog Low Pass Filter to High Pass Filter Conversion

Scilab code Solution 3.1 Analog LPF to HPF Conversion

```
1  // To Convert Analog LPF to High Pass
2  //Using Analog Filter Transformations
3  //For the cutoff frequency Wc = 500 Hz
4  //OS:Windows 10
5  //Scilab 5.5.2
6
7  clear all;
8  clc;
9  close;
10
11 omegap =  500;
12 omegas =  1000;
13 delta1_in_dB = -3;
14 delta2_in_dB = -40;
15 delta1 = 10^(delta1_in_dB/20)
16 delta2 = 10^(delta2_in_dB/20)
17
```

```
18 //Calculation of Filter Order
19 N = log10((1/(delta2^2))-1)/(2*log10(omegas/omegap))
20 N = ceil(N)
21 omegac = omegap;
22
23 //Poles and Gain Calculation
24 [pols,gain]=zpbutt(N,omegac);
25 N =1;
26
27 omega_LPF = omegap;  //Analog LPF Cutoff frequency
28 omega_HPF = omega_LPF;  //Analog HPF Cutoff
      frequency
29 omega2 = 600;    //Upper Cutoff frequency
30 omega1 = 300;     //Lower Cutoff Frequency
31 omega0 = (omega2*omega1);
32 BW =  omega2 -  omega1;  //Bandwidth
33 disp('Analog LPF Transfer function')
34 [hs,pols,zers,gain] = analpf(N,'butt',[0,0],
      omega_LPF)
35 hs_LPF = hs;
36 hs_LPF(2) = hs_LPF(2)/500;
37 hs_LPF(3)= hs_LPF(3)/500;
38 s =poly(0,'s');
39 disp('Analog HPF Transfer function')
40 h_HPF = horner(hs_LPF,omega_LPF*omega_HPF/s)
41 num = (s^2)+omega0
42 den = BW*s
43 h_BPF = horner(hs_LPF,omega_LPF*(num/den))
44
45 //Plotting Low Pass Filter Frequency Response
46 figure
47 fr=0:.1:1000;
48 hf=freq(hs_LPF(2),hs_LPF(3),%i*fr);
49 hm=abs(hf);
50 plot(fr,hm)
51 xgrid(1)
52 xtitle('Magnitude Response of LPF Filter Cutoff
      frequency = 500Hz','Analog Frequency---->','
```

```
    Magnitude');
53
54 //Plotting High Pass Filter Frequency Response
55 figure
56 fr=0:.1:1000;
57 hf_HPF=freq(h_HPF(2),h_HPF(3),%i*fr);
58 hm_HPF=abs(hf_HPF);
59 plot(fr,hm_HPF)
60 xgrid(1)
61 xtitle('Magnitude Response of HPF Filter Cutoff
    frequency = 500Hz','Analog Frequency-->','
    Magnitude');
```

Experiment: 4

Prototype Analog Low Pass Filter to Band Pass Filter Conversion

Scilab code Solution 4.1 Analog LPF to BPF Conversion

```
1 // To Convert Analog LPF to Band Pass IIR
     Butterworth Filter
2 //Using Analog Filter Transformations
3 //For the cutoff frequency Wc = 500 Hz
4 //OS:Windows 10
5 //Scilab 5.5.2
6
7 clear all;
8 clc;
9 close;
10
11 omegap =  500;
12 omegas =  1000;
13 delta1_in_dB = -3;
14 delta2_in_dB = -40;
15 delta1 = 10^(delta1_in_dB/20)
16 delta2 = 10^(delta2_in_dB/20)
```

```
17
18 //Calculation of Filter Order
19 N = log10((1/(delta2^2))-1)/(2*log10(omegas/omegap))
20 N = ceil(N)
21 omegac = omegap;
22
23 //Poles and Gain Calculation
24 [pols,gain]=zpbutt(N,omegac);
25 N =1;
26
27 omega_LPF = omegap;  //Analog LPF Cutoff frequency
28 omega_HPF = omega_LPF;  //Analog HPF Cutoff
      frequency
29 omega2 = 600;    //Upper Cutoff frequency
30 omega1 = 300;     //Lower Cutoff Frequency
31 omega0 = (omega2*omega1);
32 BW =  omega2 -  omega1;  //Bandwidth
33 disp('Analog LPF Transfer function')
34 [hs,pols,zers,gain] = analpf(N,'butt',[0,0],
      omega_LPF)
35 hs_LPF = hs;
36 hs_LPF(2) = hs_LPF(2)/500;
37 hs_LPF(3)= hs_LPF(3)/500;
38 s =poly(0,'s');
39 h_HPF = horner(hs_LPF,omega_LPF*omega_HPF/s)
40 disp('Analog BPF Transfer function')
41 num = (s^2)+omega0
42 den = BW*s
43 h_BPF = horner(hs_LPF,omega_LPF*(num/den))
44
45 //Plotting Low Pass Filter Frequency Response
46 figure
47 fr=0:.1:1000;
48 hf=freq(hs_LPF(2),hs_LPF(3),%i*fr);
49 hm=abs(hf);
50 plot(fr,hm)
51 xgrid(1)
52 xtitle('Magnitude Response of LPF Filter Cutoff
```

```
    frequency = 500Hz','Analog Frequency---->','
    Magnitude');
53
54 //Plotting Band Pass Filter Frequency Response
55 figure
56 fr=0:.1:1000;
57 hf_BPF=freq(h_BPF(2),h_BPF(3),%i*fr);
58 hm_BPF=abs(hf_BPF);
59 plot(fr,hm_BPF)
60 xgrid(1)
61 xtitle('Magnitude Response of BPF Filter Upper
    Cutoff frequency = 600Hz & Lower Cutoff frequency
     = 300Hz','Analog  Frequency---->','Magnitude');
```

Experiment: 5

Prototype Analog Low Pass Filter to Band Rejection/Stop Filter Conversion

Scilab code Solution 5.1 Analog LPF to BSF Conversion

```
1 //Analog Low Pass Filter to Band Stop Filter
    Conversion
2 //OS:Windows 10
3 //Scilab 6.0.0
4
5 clear ;
6 clc ;
7 close ;
8
9 fc1 = input("Enter Analog lower cutoff freq. in Hz="
    )
10 fc2 = input("Enter Analog higher cutoff freq. in Hz=
    ")
11 fs = input("Enter Analog sampling freq. in Hz=")
12 M = input("Enter order of filter =")
13 w1 = (2*%pi)*(fc1/fs);
14 w2 = (2*%pi)*(fc2/fs);
```

```
15 disp(w1,'Digital lower cutoff frequency in radians.
     cycles/samples');
16 disp(w2,'Digital higher cutoff frequency in radians.
     cycles/samples');
17 wc1 = w1/%pi;
18 wc2 = w2/%pi;
19 disp(wc1,'Normalized digital lower cutoff frequency
     in cycles/samples');
20 disp(wc2,'Normalized digital higher cutoff frequency
      in cycles/samples');
21 [wft1,wfm1,fr1]=wfir('lp',M+1,[wc1/2,0],'re',[0,0]);
22 disp(wft1,'Impulse Response of LPF FIR Filter:h[n]='
     );
23
24 //Plotting the Magnitude Response of LPF FIR Filter
25 figure
26 subplot(2,1,1)
27 plot(2*fr1,wfm1)
28 xlabel('Normalized Digital Frequency w-->')
29 ylabel('Magnitude |H(w)|=')
30 title('Magnitude Response of FIR LPF')
31 xgrid(1)
32 subplot(2,1,2)
33 plot(fr1*fs,wfm1)
34 xlabel('Analog Frequency in Hz f -->')
35 ylabel('Magnitude |H(w)|=')
36 title('Magnitude Response of FIR LPF')
37 xgrid(1)
38
39 [wft,wfm,fr]=wfir('sb',M+1,[wc1/2,wc2/2],'re',[0,0])
     ;
40 disp(wft,'Impulse Response of BSF FIR Filter:h[n]=')
     ;
41
42 //Plotting the Magnitude Response of HPF FIR Filter
43 figure
44 subplot(2,1,1)
45 plot(2*fr,wfm)
```

```
46 xlabel('Normalized Digital Frequency w-->')
47 ylabel('Magnitude |H(w)|=')
48 title('Magnitude Response of FIR BSF')
49 xgrid(1)
50 subplot(2,1,2)
51 plot(fr*fs,wfm)
52 xlabel('Analog Frequency in Hz f -->')
53 ylabel('Magnitude |H(w)|=')
54 title('Magnitude Response of FIR BSF')
55 xgrid(1)
56
57
58 //Example Input
59 //Enter Analog lower cutoff freq. in Hz=250
60 //Enter Analog higher cutoff freq. in Hz=600
61 //Enter Analog sampling freq. in Hz=2000
62 //Enter order of filter =4
63 //Digital lower cutoff frequency in radians
64 // .cycles/samples
65 //    0.7853982
66
67 // Digital higher cutoff frequency in radian
68 // s.cycles/samples
69 //    1.8849556
70
71 // Normalized digital lower cutoff frequency
72 //  in cycles/samples
73 //    0.25
74
75 // Normalized digital higher cutoff frequenc
76 // y in cycles/samples
77 //    0.6
78 // Impulse Response of LPF FIR Filter:h[n]=
79 //          column 1 to 3
80 //    0.1591549   0.2250791   0.25
81 //          column 4 to 5
82 //    0.2250791   0.1591549
83
```

```
84 // Impulse Response of BSF FIR Filter:h[n]=
85 //          column 1 to 3
86 //    0.2527039   -0.0776516    0.65
87 //          column 4 to 5
88 //   -0.0776516    0.2527039
```

Experiment: 6

Comparision of FIR filter design using windowing method

Scilab code Solution 6.1 Rectangular Window

```
1 //Plot Magnitude Response of L.P.F.
2 //N=7 , fc =1000Hz , F=5000Hz
3 //FIR filter design using Windowing Method-
     Rectangular Window
4 //OS:Windows 10
5 //Scilab 6.0.0
6
7 clear ;
8 clc ;
9 close ;
10
11 N =7;
12 U =4;
13 h_Rect =window('re',N);
14 for n= -3+U :1:3+ U
15     if n ==4
16         hd(n) =0.4;
17     else
18         hd(n)=( sin (2* %pi *(n-U) /5) )/( %pi *(n-U
```

```
            ));
19      end
20      h(n)=hd(n)* h_Rect (n);
21  end
22  [hzm,fr]=frmag(h,256) ;
23  hzm_dB=20*log10(hzm)./max(hzm);
24  figure
25  xgrid(2);
26  plot(2*fr,hzm_dB)
27  a=gca();
28  xlabel('Frequency w*pi ');
29  ylabel('Magnitude in dB ');
30  title('Frequency Response of FIR LPF with N=7');
31  xgrid(2)
32  disp(h,"Filter Co efficients , h(n)=");
```

Scilab code Solution 6.2 Kaiser Window

```
1  //Plot Magnitude Response of L.P.F .
2  //FIR filter design using Windowing Method- Kaiser
       Window
3  //OS:Windows 10
4  //Scilab 6.0.0
5
6  clear ;
7  clc ;
8  close ;
9
10 wsf = input("Enter the value of wsf in rad/sec");
11 ws = input("Enter the value of ws in rad/sec");
12 wp = input("Enter the value of wp in rad/sec");
13 as = input("Enter the value of as in dB");
14 ap = input("Enter the value of ap in dB");
15
16 B=ws -wp;
```

```
wc =0.5*(ws+wp);
wc1 =wc*2*%pi/ wsf ;
delta1 =10^(-0.05* as);
delta2 =(10^(0.05* as) -1)/(10^(0.05* as)+1);
delta = min(delta1,delta2);
alphas = -20*log10(delta);
alpha =0.5842*(alphas -21)^0.4+0.07886*(alphas -21)
D=(alphas -7.95)/14.36;
N1=wsf*D/B +1;
N= ceil(N1);
U= ceil(N/2) ;
win_l = window('kr',N,alpha);
for n=- floor(N/2)+U :1: floor(N/2) +U
    if n== ceil(N /2) ;
        hd(n) =0.5;
    else
        hd(n)=(sin(%pi*(n-U)/2))/(%pi*(n-U));
    end
    h(n)=hd(n)* win_l(n);
end
[hzm,fr]=frmag(h,256) ;
hzm_dB = 20*log10(hzm)./max(hzm);
figure
plot(2*fr,hzm_dB )
a= gca();
xlabel('Frequency w*pi ');
ylabel('Magnitude in dB ');
title('Frequency Response of LPF');
xgrid(2);
disp(h,"Filter Co efficients , h(n)=");

//Example Input
//Enter the value of wsf in rad/sec 100
//Enter the value of ws in rad/sec 30
//Enter the value of wp in rad/sec 20
//Enter the value of as in dB 44
//Enter the value of ap in dB 0.1
//Filter Co efficients , h(n)=
```

```
55 //    0.002441   -3.172D-18   -0.0068491    6.235D-18
      0.0145007   -9.826D-18   -0.027237    1.347D-17
      0.0494369   -1.661D-17   -0.0970495    1.874D-17
      0.3152014    0.5    0.3152014    1.874D-17
      -0.0970495   -1.661D-17    0.0494369    1.347D-17
      -0.027237   -9.826D-18    0.0145007    6.235D-18
      -0.0068491   -3.172D-18    0.002441
```

Scilab code Solution 6.3 Hamming Window

```
1  //Plot Magnitude Response of ideal H.P.F.
2  //wc1=0.25*pi , N=11
3  //FIR filter design using Windowing Method-Hamming
      Window
4  //OS:Windows 10
5  //Scilab 6.0.0
6
7  clear;
8  clc;
9  close;
10
11 N =11;
12 U =6;
13 h_hamm=window('hm',N);
14 for n= -5+U :1:5+ U
15     if n ==6
16         hd(n) =0.75;
17     else
18         hd(n)=(sin(%pi*(n-U))-sin(%pi*(n-U)/4))/(%pi
              *(n-U));
19     end
20     h(n)= h_hamm(n)*hd(n);
21 end
22 [hzm,fr]=frmag(h,256);
23 hzm_dB=20*log10(hzm)./max(hzm);
```

```
figure
plot(2*fr,hzm_dB)
a=gca();
xlabel('Frequency w*pi');
ylabel('Magnitude in dB');
title('Frequency Response of FIR HPF with N=11 using
    Hamming Window');
xgrid(2);
```

Scilab code Solution 6.4 Hanning Window

```
//Plot Magnitude Response of ideal H.P.F.
//N=11 ,wc1=0.25* pi
//FIR filter design using Windowing Method-Hanning
   Window
//OS:Windows 10
//Scilab 6.0.0

clear;
clc;
close;

N =11;
U =6;
h_hann = window('hn',N);
for n= -5+U:1:5+U
    if n == 6
        hd(n) = 0.75;
    else
        hd(n)=(sin(%pi*(n-U))-sin(%pi*(n-U)/4))/(%pi
           *(n-U));
    end
    h(n)= h_hann(n)*hd(n);
end
[hzm,fr]= frmag(h,256) ;
```

```
23 hzm_dB =20*log10(hzm)./max(hzm);
24 figure
25 plot(2*fr,hzm_dB)
26 a=gca();
27 xlabel('Frequency w*pi ');
28 ylabel('Magnitude in dB ');
29 title('Frequency Response of FIR HPF with N=11 using
       Hanning Window ');
30 xgrid(2);
```

Experiment: 7

Detection of DTMF signal

Scilab code Solution 7.1 DTMF Decoding

```
1  //DTMF Signal Detection
2  //OS:Windows 10
3  //Scilab 5.5.2
4
5  clear all;
6  clc;
7  close;
8
9  row_f1=[800 870 950 990];          // Row Frequency
10 colum_f1=[1340 1440 1540];         // Column Frequency
11 fs=8000;                           // Sampling Frequency
12 N=1:800;                           // Total No. of
       Samples for each Digit
13 mobile=[5 6 7 8 9 0 1 2 3 4];
14 total_signal=[];
15
16 figure;
17
18 for i=1:length(mobile)
19     select mobile(i)
20     case 1
```

```
21          row_f=1;
22          colum_f=1;
23      case 2
24          row_f=1;
25          colum_f=2;
26      case 3
27          row_f=1;
28          colum_f=3;
29      case 4
30          row_f=2;
31          colum_f=1;
32      case 5
33          row_f=2;
34          colum_f=2;
35      case 6
36          row_f=2;
37          colum_f=3;
38       case 7
39          row_f=3;
40          colum_f=1;
41      case 8
42          row_f=3;
43          colum_f=2;
44      case 9
45          row_f=3;
46          colum_f=3;
47       case 0
48          row_f=4;
49          colum_f=2;
50      else
51          row_f=4;
52          colum_f=1;
53      end
54       y=sin(2*3.14*(row_f1(row_f)/fs)*N)+sin(2*3.14*(
            colum_f1(colum_f)/fs)*N);  //Time Domain
            Signal Generation for each Digit
55       total_signal=[total_signal y zeros(1,8800)];
56       temp(:,:,i)=y(:,:);
```

```
57      end
58      plot(total_signal);
59      title('DTMF Signal','color','blue');
60      xlabel("Samples","color", "blue");
61      ylabel("Amplitude","color", "blue");
62      sound(total_signal,fs);
63
64  row_f=[];
65  col_f=[];
66
67  for i=1:10
68      n=length(temp(:,:,i));
69      p=abs(fft(temp(:,:,i))); // FFT of Signal of
            respective Digit
70      f=(0:n-1)*fs/n;          // Total Frequency
            Range
71      //plot(f,p);
72      row=p(2:100);            // Row Frequency
            separation
73      col=p(101:200);          // Column Frequency
            separation
74      [r1 c1]=find(row==max(row));   // Finding the
            location of peak for Row Frequency
75      [r2 c2]=find(col==max(col));   // Finding the
            location of peak for Column Frequency
76      row_f=[row_f 10*c1];            // Array
            containing peak of Row Freqiency
77      col_f=[col_f (10*(c2+100))-10]; // Array
            containing peak of Column Freqiency
78  end
79
80  mobile_find=[];  // Blank Array to Store Mobile
      Number
81  for i=1:10      // Loop forFinding the Number form
      the Row and Column Frequency
82      if(row_f(i)==800 & col_f(i)==1340)
83          n0=1;
84          elseif(row_f(i)==800 & col_f(i)==1440)
```

```
85      n0=2;
86      elseif(row_f(i)==800 & col_f(i)==1540)
87      n0=3;
88      elseif(row_f(i)==870 & col_f(i)==1340)
89      n0=4;
90      elseif(row_f(i)==870 & col_f(i)==1440)
91      n0=5;
92      elseif(row_f(i)==870 & col_f(i)==1540)
93      n0=6;
94      elseif(row_f(i)==950 & col_f(i)==1340)
95      n0=7;
96      elseif(row_f(i)==950 & col_f(i)==1440)
97      n0=8;
98      elseif(row_f(i)==950 & col_f(i)==1540)
99      n0=9;
100     elseif(row_f(i)==990 & col_f(i)==1440)
101     n0=0;
102 end
103 mobile_find=[mobile_find n0];  // Array containing
      Decoded Digit of Mobile Number.
104 end
105
106 disp(mobile_find,"Decoded Mobile Number :");
```

Experiment: 8

Implementation of Decimation

Scilab code Solution 8.1 Decimation

```
1  //Implementation of Decimation
2  //OS:Windows 10
3  //Scilab 5.5.2
4
5  clear all;
6  clc;
7  close;
8
9  t=0:0.00025:1;
10 x=sin(2*%pi*30*t)+sin(2*%pi*60*t);    //original
       signal
11 y = x(1:4:length(x)); //downsampled by a factor of 4
12 subplot(2,1,1)
13 plot2d3(0:120,x(1:121),-1);
14 xgrid(1);
15 xtitle('Original singal')
16 subplot(2,1,2)
17 plot2d3(0:30,y(1:31),-1);
18 xgrid(1);
19 xtitle('Downsampled Signal');
```

Experiment: 9

Implementation of Interpolation

Scilab code Solution 9.1 Interpolation

```
1  //Implementation of Interpolation
2  //OS:Windows 10
3  //Scilab 5.5.2
4
5  clear all;
6  clc;
7  close;
8
9  t=0:0.00025:1;
10 x=sin(2*%pi*30*t)+sin(2*%pi*60*t);    //original
       signal
11 upsampling_x = zeros(1,2*length(x)); //upsampled by
       a factor of 2
12 upsampling_x(1:2:2*length(x)) = x;
13 subplot(2,1,1)
14 plot2d3(0:120,x(1:121),-1);
15 xgrid(1);
16 xtitle('Original singal')
17 subplot(2,1,2)
```

```
18 plot2d3(0:250,upsampling_x(1:251),-1);
19 xgrid(1);
20 xtitle('Upsampled Signal');
```

Experiment: 10

FIR Filter using Frequency Sampling Method

Scilab code Solution 10.1 Frequency Sampling Method

```
1  //FIR LPF Filter using Frequency Sampling Method
2  //OS:Windows 10
3  //Scilab 5.5.2
4
5  clc;
6  clear;
7  close;
8
9  N = input("Enter the value of N:");
10 U = input("Enter the value of U:");
11 for n =0+ U :1: N -1+ U
12     h(n)=(1+cos(2*%pi*(7-n)/N))/N;
13 end
14 [hz,f]=frmag(h,256) ;
15 hz_dB=20*log10(hz)./max(hz);
16 figure ;
17 plot(2*f,hz_dB);
18 a=gca();
19 xlabel('Frequency w      pi ') ;
```

```
20 ylabel('Magnitude in dB ') ;
21 title('Frequency Response of FIR LPF') ;
22 xgrid(2)
23
24 //Example input
25 //Enter the value of N:15
26 //Enter the value of U:1 , Wc=pi/4
```