

Scilab Manual for
Digital Signal Processing
by Dr Prarthan Mehta
Others
Dharmsinh Desai University¹

Solutions provided by
Prof Pinkesh Patel
Others
Dharmsinh Desai University

February 24, 2021

¹Funded 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	4
1 To generate basic discrete signal used in Digital Signal Processing	6
2 To perform basic signal operation (addition, multiplication, shifting, folding) on the discrete sequences.	10
3 To perform Convolution & Correlation Operation on Two Discrete Sequences	16
4 To perform Circular Shifting Operation on Discrete Sequence and prove phase reversal property of DFT.	20
5 Applications of Correlation	23
6 Musical Tone Generation [sa re ga ma pa dh ni sa] with each Tone has time duration 0.5 sec.	30
7 DTMF Signal Generation and Decoder according to the given Mobile Number.	32
8 Design Low Pass Filter as per the given specification and plot the Frequency Response.	39
9 Design High Pass Filter as per the given specification and plot the Frequency Response.	41

10 Design Band Pass Filter as per the given specification and plot the Frequency Response.	43
11 Design Equillizer (LOW + HIGH + BANDPASS) as per the given specification	45
12 To Generation of ECG signals used in Medical Processing	50
13 Design the following Low Pass analog filters with the given specification. (1) Butter Worth (2) Chebyshev-I (3) Chebyshev-II	52
14 Design the following IIR filters with the given specification. (1) Butter Worth (2) Chebyshev-I (3) Chebyshev-II (4) Elliptic	55
15 Design the following FIR filters with the given specification. (1) Low Pass (2) High Pass (3) Band Pass (4) Band Stop	58

List of Experiments

Solution 1.01	Basic Discreate Signal Generation	6
Solution 2.01	Signal Addition and Multiplication	10
Solution 2.02	Signal Shifting Operation	12
Solution 2.03	Signal Folding Operation	14
Solution 3.01	Convolution Operation	16
Solution 3.02	Correlation Operation	17
Solution 4.01	Circular Shifting Operation and Phase reversal Prop- erty	20
Solution 5.01	Radar Signal Processing	23
Solution 5.02	Binary Sequence Detection at Receiver	27
Solution 6.01	Musical Tone Generation	30
Solution 7.01	DTMF Signal Generation	32
Solution 7.02	DTMF Signal Generation and Decoder	34
Solution 8.01	Low Pass Filter Design	39
Solution 9.01	High Pass Filter design	41
Solution 10.01	Band Pass Filter	43
Solution 11.01	Equilizer Design	45
Solution 12.01	ECG Wave Generation	50
Solution 13.01	Low Pass Filter Design	52
Solution 14.01	IIR filter Design	55
Solution 15.01	FIR Filter	58

List of Figures

13.1 Low Pass Filter Design	54
---------------------------------------	----

Experiment: 1

To generate basic discrete signal used in Digital Signal Processing

Scilab code Solution 1.01 Basic Discrete Signal Generation

```
1 // Exp-1 To generate basic discrete signal used in
  Digital Signal Processing
2
3 // Version : Scilab 5.4.1
4 // Operating System : Window-xp, Window-7
5
6 clc;
7 clear;
8 xdel(winsid());
9 t=0:0.1:20;
10 f=0.2;
11 pi=3.14;
12
13
14 ////////////////////////////////////// SINEWAVE
  //////////////////////////////////////
15 x1=sin(2*pi*f*t);
```

```

16 //scf();
17 subplot(231);
18 plot(t,x1,'cya+','marker','d','markerfac','green','
    markeredg','red');
19 title('Sinewave','color','red','fontsize',4);
20 //xtitle('Sinewave','Index','Amplitude');
21 xlabel("Index","fontsize",2,"color","blue");
22 ylabel("Amplitude","fontsize",2,"color","blue");
23
24
25 /////////////////////////////////////////////////// Cosine Wave
    //////////////////////////////////////
26 x2=cos(2*pi*f*t);
27 //scf();
28 subplot(232);
29 plot(t,x2,'cya+','marker','d','markerfac','red','
    markeredg','yel');
30 title('Cosinewave','color','red','fontsize',4);
31 xlabel("Index","fontsize",2,"color","blue");
32 ylabel("Amplitude","fontsize",2,"color","blue");
33
34
35 /////////////////////////////////////////////////// Impulse Wave
    //////////////////////////////////////
36 t1=-10:10;
37 x3=[zeros(1,10) 1 zeros(1,10)];
38 //scf();
39 subplot(233);
40 plot(t1,x3,'cya+','marker','d','markerfac','green','
    markeredg','red');
41 title('Impulse','color','red','fontsize',4);
42 xlabel("Index","fontsize",2,"color","blue");
43 ylabel("Amplitude","fontsize",2,"color","blue");
44
45
46 /////////////////////////////////////////////////// Ramp Wave
    //////////////////////////////////////
47 t4=0:10;

```



```

48 x4=t4;
49 //scf();
50 subplot(234);
51 plot(t4,x4,'cya+', 'marker','d','markerfac','green','
    markeredg','red');
52 title('Ramp Wave','color','red','fontsize',4);
53 xlabel("Index","fontsize",2,"color","blue");
54 ylabel("Amplitude","fontsize",2,"color","blue");
55
56
57 ////////////////////////////////////// Exponetial Wave
    //////////////////////////////////////
58 t5=0:10;
59 x5=exp(t5);
60 //scf();
61 subplot(235);
62 plot(t5,x5,'cya+', 'marker','d','markerfac','green','
    markeredg','red');
63 title('Exponetial Wave','color','red','fontsize',4)
    ;
64 xlabel("Index","fontsize",2,"color","blue");
65 ylabel("Amplitude","fontsize",2,"color","blue");
66
67
68 ////////////////////////////////////// Random Wave
    //////////////////////////////////////
69
70 x6=rand(1,100);
71 //scf();
72 subplot(236);
73 plot(1:length(x6),x6,'cya+', 'marker','d','markerfac'
    ,'green','markeredg','red');
74 title('Random Wave','color','red','fontsize',4);
75 xlabel("Index","fontsize",2,"color","blue");
76 ylabel("Amplitude","fontsize",2,"color","blue");
77
78
79

```

```

80 ///////////////////////////////////////////////////////////////////Impulse Sequence ///////////////////////////////////////////////////////////////////
81 n1=1,n0=50,n2=100;
82 if((n0<n1)|(n0>n2)|(n1>n2))
83     error('arugument incorrect');
84 end
85 n=[n1:n2];
86 x7=[(n-n0)==0,1];
87 scf()
88 subplot(121);
89 plot(n,x7(n1:n2),'cya+', 'marker','d','markerfac','green',
      'markeredg','red');
90 title('Impulse Sequence','color','red','fontsize',4);
91 xlabel("Index", "fontsize", 2,"color", "blue");
92 ylabel("Amplitude", "fontsize", 2, "color", "blue");
93
94
95 /////////////////////////////////////////////////////////////////// Step Sequence ///////////////////////////////////////////////////////////////////
96 n1=1,n0=50,n2=100;
97 if((n0<n1)|(n0>n2)|(n1>n2))
98     error('arugument incorrect');
99 end
100 n=[n1:n2];
101 x8=[(n-n0)>=0,1];
102 subplot(122);
103 plot(n,x8(n1:n2),'cya+', 'marker','d','markerfac','green',
      'markeredg','red');
104 title('Step Sequence','color','red','fontsize',4);
105 xlabel("Index", "fontsize", 2,"color", "blue");
106 ylabel("Amplitude", "fontsize", 2, "color", "blue");

```

Experiment: 2

To perform basic signal operation (addition, multiplication, shifting, folding) on the discrete sequences.

Scilab code Solution 2.01 Signal Addition and Multiplication

```
1 // Exp-2 Addition & Multiplication Operation on two
   Discrete Sequences
2
3 // Version : Scilab 5.4.1
4 // Operating System : Window-xp, Window-7
5
6 clc;
7 clear;
8 xdel(winsid());
9
10 x1=[1 6 7 4 5 2 3 7 8 9];
11 n1=[-3 -2 -1 0 1 2 3 4 5 6];
12 x2=[5 8 6 9 4 2 3 7 5 6 2 8 7];
13 n2=[4 5 6 7 8 9 10 11 12 13 14 15 16] ;
14 n=min(min(n1),min(n2)):max(max(n1),max(n2));
```

```

15 y1=zeros(1,length(n));
16 y2=y1;
17 y1(find((n>=min(n1))&(n<=max(n1))))=x1;
18 y2(find((n>=min(n2))&(n<=max(n2))))=x2;
19 y=y1+y2;
20 x=y1.*y2;
21 // Addition of Two Sequences //
22 scf();
23 subplot(311);
24 bar(n,y1,0.1,'red');
25 title('Sequence_1','color','red','fontsize',4);
26 xlabel("Index","fontsize",2,"color","blue",'
    position',[0.3 0.3]);
27 ylabel("Amplitude","fontsize",2,"color","blue");
28 subplot(312);
29 bar(n,y2,0.1,'yellow');
30 title('Sequence_2','color','red','fontsize',4);
31 xlabel("Index","fontsize",2,"color","blue",'
    position',[0.3 0.3]);
32 ylabel("Amplitude","fontsize",2,"color","blue");
33 subplot(313)
34 bar(n,y,0.1,'Green');
35 //plot(n,y,'cya+','marker','d','markerfac','red','
    markeredg','yel');
36 title('Addition of Sequences','color','red','
    fontsize',4);
37 xlabel("Index","fontsize",2,"color","blue",'
    position',[0.3 0.3]);
38 ylabel("Amplitude","fontsize",2,"color","blue");
39
40 // Multiplication of Two Sequences //
41 scf();
42 subplot(311);
43 bar(n,y1,0.1,'red');
44 title('Sequence_1','color','red','fontsize',4);
45 xlabel("Index","fontsize",2,"color","blue",'
    position',[0.3 0.3]);
46 ylabel("Amplitude","fontsize",2,"color","blue");

```

```

47 subplot(312);
48 bar(n,y2,0.1,'yellow');
49 title('Sequence_2','color','red','fontsize',4);
50 xlabel("Index","fontsize",2,"color","blue",'
    position',[0.3 0.3]);
51 ylabel("Amplitude","fontsize",2,"color","blue");
52 subplot(313)
53 bar(n,x,0.1,'Green');
54 //plot(n,y,'cya+','marker','d','markerfac','red','
    markeredg','yel');
55 title('Multiplication of Sequences','color','red','
    fontsize',4);
56 xlabel("Index","fontsize",2,"color","blue",'
    position',[0.3 0.3]);
57 ylabel("Amplitude","fontsize",2,"color","blue");

```

Scilab code Solution 2.02 Signal Shifting Operation

```

1 // Exp-3 Perform Shifting Operation on Discrete
  Sequences
2
3 // Version : Scilab 5.4.1
4 // Operating System : Window-xp, Window-7
5
6 clc;
7 clear;
8 xdel(winsid());
9
10 x2=input("Enter the Sequence :"); // [2 3 5 6 4
    8 6];
11 factor=input("Enter the Shifting Factor :"); //
    Example : -2 or 2
12
13
14 a=[];

```

```

15 t=[];
16 for i=1:length(x2)
17     t=[t (i-factor)];
18     a=[a x2(i)];
19 end
20 temp=[];
21 if(factor>0)
22     a=[a zeros(1,factor)];
23     for j=1:factor
24         temp=[temp t(length(t))+j];
25     end
26     t=[t temp];
27 end
28 if(factor<0)
29     a=[zeros(1,-factor) a];
30     for j=1:-factor
31         temp=[temp j];
32     end
33     t=[temp t];
34 end
35
36
37 scf();
38 subplot(2,1,1);
39 bar(x2,0.1,'red');
40 title('Original Sequence','color','red','fontsize',
41     4);
42 xlabel("Index", "fontsize", 2,"color", "blue",'
43     position',[0.3 0.3]);
44 ylabel("Amplitude", "fontsize", 2, "color", "blue");
45
46 title('Original Sequence');
47 subplot(2,1,2);
48 bar(t,a,0.1,'green');
49 title('Shfted Sequence','color','red','fontsize', 4)
50 ;
51 xlabel("Index", "fontsize", 2,"color", "blue",'
52     position',[0.3 0.3]);

```

```
49 ylabel("Amplitude", "fontsize", 2, "color", "blue");
```

Scilab code Solution 2.03 Signal Folding Operation

```
1 // Exp-4 Perform Signal Folding Operation on
  Discrete Sequences
2
3 // Version : Scilab 5.4.1
4 // Operating System : Window-xp, Window-7
5
6 clc;
7 clear;
8 xdel(winsid());
9
10 x2=input("Enter the Sequence :"); // [2 3 5 6 4
    8 6];
11 x3=mtlbfliplr(x2);
12 n3=1:length(x2);
13 n3=-mtlbfliplr(n3);
14 scf();
15 subplot(2,1,1);
16 bar(x2,0.1,'red');
17 title('Original Sequence','color','red','fontsize',
    4);
18 xlabel("Index", "fontsize", 2,"color", "blue",'
    position',[0.3 0.3]);
19 ylabel("Amplitude", "fontsize", 2, "color", "blue");
20
21 subplot(2,1,2);
22 bar(n3,x3,0.1,'green');
23 title('Folded Sequence','color','green','fontsize',
    4);
24 xlabel("Index", "fontsize", 2,"color", "blue",'
    position',[0.3 0.3]);
25 ylabel("Amplitude", "fontsize", 2, "color", "blue");
```


Experiment: 3

To perform Convolution & Correlation Operation on Two Discrete Sequences

Scilab code Solution 3.01 Convolution Operation

```
1 // Exp-5 Perform Convolution Operation on Two
  Discrete Sequences
2
3 // Version : Scilab 5.4.1
4 // Operating Syatem : Window-xp, Window-7
5
6
7 clc;
8 clear;
9 xdel(winsid());
10
11 x1=input("Enter the Sequence_1 :"); // [1 2 3 4
    5];
12 x2=input("Enter the Sequence_2 :"); // [5 4 8];
13 n = length(x1);
14 m = length(x2);
15 for k = 1:(m+n-1)
```

```

16     w(k) = 0;
17         for j =max(1,k+1-m) : min(k,n)
18             w(k)= w(k)+(x1(j)*x2(k+1-j));
19         end
20 end
21
22 scf();
23 subplot(3,1,1);
24 bar(x1,0.1,'red');
25 title('Sequence_1','color','red','fontsize',4);
26 xlabel("Index", "fontsize", 2,"color", "blue",'
    position',[0.6 0.3]);
27 ylabel("Amplitude", "fontsize", 2, "color", "blue");
28
29 subplot(3,1,2);
30 bar(x2,0.1,'yellow');
31 title('Sequence_2','color','red','fontsize',4);
32 xlabel("Index", "fontsize", 2,"color", "blue",'
    position',[0.6 0.3]);
33 ylabel("Amplitude", "fontsize", 2, "color", "blue");
34
35 subplot(3,1,3);
36 bar(w,0.1,'green');
37 title('Convoled Sequence','color','green','fontsize',
    4);
38 xlabel("Index", "fontsize", 2,"color", "blue",'
    position',[0.3 0.3]);
39 ylabel("Amplitude", "fontsize", 2, "color", "blue");

```

Scilab code Solution 3.02 Correlation Operation

```

1 // Exp-6 Perform Correlation Operation on Two
  Discrete Sequences
2
3 // Version : Scilab 5.4.1

```

```

4 // Operating System : Window-xp, Window-7
5
6 clc;
7 clear;
8 xdel(winsid());
9 x=[2 4 5 6];
10 y=[2 4 5];
11
12 m=length(x);
13 n=length(y);
14
15 for k=1:m+n-1
16     w(k)=0;
17     for j=max(1,k+1-n):min(k,m)
18         w(k)=w(k)+(x(j)*y(n-k+j));
19     end
20 end
21
22
23 scf();
24 subplot(3,1,1);
25 bar(x,0.1,'red');
26 title('Sequence_1','color','red','fontsize',4);
27 xlabel("Index","fontsize",2,"color","blue",'
    position',[0.6 0.3]);
28 ylabel("Amplitude","fontsize",2,"color","blue");
29
30 subplot(3,1,2);
31 bar(y,0.1,'yellow');
32 title('Sequence_2','color','red','fontsize',4);
33 xlabel("Index","fontsize",2,"color","blue",'
    position',[0.6 0.3]);
34 ylabel("Amplitude","fontsize",2,"color","blue");
35
36 subplot(3,1,3);
37 bar(w,0.1,'green');
38 title('Correlation of Sequences','color','green','
    fontsize',4);

```

```
39 xlabel("Index", "fontsize", 2, "color", "blue", '  
    position', [0.3 0.3]);  
40 ylabel("Amplitude", "fontsize", 2, "color", "blue");
```

Experiment: 4

To perform Circular Shifting Operation on Discrete Sequence and prove phase reversal property of DFT.

Scilab code Solution 4.01 Circular Shifting Operation and Phase reversal Property

```
1 // Exp-7 Perform Circular Shifting Operation on
  Discrete Sequence and prove
2
3 // Version : Scilab 5.4.1
4 // Operating System : Window-xp, Window-7
5
6 clc;
7 clear;
8 xdel(winsid());
9
10 n=0:10; // Index
11 x=10*(0.8).^n; // Input Sequence
12 y=x(pmodulo(-n,length(n))+1); // y is circular
  shifted sequence of x
```

```

13
14 scf();
15 subplot(2,1,1);
16 bar(x,0.1,'Green');
17 title('Original Sequence','color','red','fontsize',
    4);
18 xlabel("Index", "fontsize", 2,"color", "blue");
19 ylabel("Amplitude", "fontsize", 2, "color", "blue");
20
21 subplot(2,1,2);
22 bar(y,0.1,'yellow');
23 title('Circular Shifted Sequence','color','red','
    fontsize', 4);
24 xlabel("Index", "fontsize", 2,"color", "blue");
25 ylabel("Amplitude", "fontsize", 2, "color", "blue");
26
27
28 X=fft(x,-1); // Discrete Fourier Transform of
    Original Sequence
29 Y= fft(y,-1); // Discrete Fourier Transform of
    Circular Shifted Sequence
30
31 scf();
32 subplot(2,2,1);
33 bar(n,real(X),0.1,'Green');
34 title('Real{DFT[x(n)]}','color','red','fontsize', 4)
    ;
35 xlabel("Index", "fontsize", 2,"color", "blue");
36 ylabel("Amplitude", "fontsize", 2, "color", "blue");
37
38
39 subplot(2,2,2);
40 bar(n,imag(X),0.1,'Green');
41 title('Imag{DFT[x(n)]}','color','red','fontsize', 4)
    ;
42 xlabel("Index", "fontsize", 2,"color", "blue");
43 ylabel("Amplitude", "fontsize", 2, "color", "blue");
44

```

```
45 subplot(2,2,3);
46 bar(n,real(Y),0.1,'Green');
47 title('Real{DFT[x((-n))]}', 'color','red','fontsize',
48       4);
49 xlabel("Index", "fontsize", 2,"color", "blue");
50 ylabel("Amplitude", "fontsize", 2, "color", "blue");
51 subplot(2,2,4);
52 bar(n,imag(Y),0.1,'Green');
53 title('Imag{DFT[x((-n))]}', 'color','red','fontsize',
54       4);
55 xlabel("Index", "fontsize", 2,"color", "blue");
56 ylabel("Amplitude", "fontsize", 2, "color", "blue");
```

Experiment: 5

Applications of Correlation

Scilab code Solution 5.01 Radar Signal Processing

```
1 // Exp-8 Application of Correlation in RADAR Signal
  Processing to detect Object and it's Distance.
2
3 // Objective : Correlation finds application in
  RADAR Signal Processing, in which the objective
  is to find whether the object is presence or
  Absence. it also required to find the distance
  between object and radar.
4
5 //  $R[n]=\text{Alpha}*X[n-D]+\text{Noise}$  where Alpha=
  Attenuation Factor = 1 and D= Delay factor by
  which transmitted signal is received back to
  receiver (RADAR)
6
7 // Observation : In the presence of object the
  received signal is reflected signal from the
  object which can be given by  $R(n) = X(n-D)+\text{Noise}$ .
  if we find the correlation between received
  signal and transmitted signal, than the
  correlation value is higher at the delay index.
8
```



```

9 // Observation :In the Absence of object the
  received signal is only the Noise , which can be
  given by  $R(n) = \text{Noise}$ . if we find the correlation
  between received signal and transmitted signal ,
  than the correlation value is not higher at the
  delay index what we have get in the presence of
  object .
10
11 // Version : Scilab 5.4.1
12 // Operating System : Window-xp, Window-7
13
14 clc;
15 clear;
16 xdel(winsid());
17
18 x=[0 1 2 3 2 1 0]; //Triangle pulse
  transmitted by radar
19 n=[-3 -2 -1 0 1 2 3]; // Index of Triangular
  Pulse
20 D=10; // Delay amount
21 nd=n+D; // Index of Delayed
  Signal
22 y=x; // Delayed Signal
23
24 scf();
25 subplot(2,1,1);
26 bar(n,x,0.1, 'red ');
27 title('Original Transmitted Signal ', 'color ', 'red ', '
  fontsize ', 4);
28 xlabel("Index", "fontsize", 2, "color", "blue");
29 ylabel("Amplitude", "fontsize", 2, "color", "blue");
30
31 subplot(2,1,2);
32 bar(nd,y,0.1, 'yellow ');
33 title('Delayed Signal ', 'color ', 'red ', 'fontsize ', 4);
34 xlabel("Index", "fontsize", 2, "color", "blue");
35 ylabel("Amplitude", "fontsize", 2, "color", "blue");
36

```

```

37 w=rand(1,length(x));           // Noise Generation
38 nw=nd;
39 scf();
40 bar(nw,w,0.1,'red');
41 title('Noisy Signal','color','red','fontsize',4);
42 xlabel("Index", "fontsize", 2,"color", "blue");
43 ylabel("Amplitude", "fontsize", 2, "color", "blue");
44
45 // If object is present we receive the signal R(n) =
    x(n-D) + Noise
46 R=y+w;           // Original Signal+Noise
47 nr=nw;          // Index of received signal at RADAR
48
49 nr_fold=mtlbfliplr(nr);
50 R_fold=mtlbfliplr(R);
51 nmin=min(n)+min(nr_fold);      // Lowest index of y
    (n)
52 nmax=max(n)+max(nr_fold);     // Highest index of
    y(n)
53 n_received=nmin:nmax;
54 Received_Presence=conv(x,R_fold); // Convolution of
    Original signal and Received Signal in the
    Presence of Object (Equivalent to Correlation)//
55
56 scf();
57 subplot(2,1,1);
58 bar(n_received,Received_Presence,0.1,'red');
59 title('Correlation in the Presence of Object','color
    ','red','fontsize',4);
60 xlabel("Index", "fontsize", 2,"color", "blue");
61 ylabel("Correlation Value", "fontsize", 2, "color",
    "blue");
62
63 // If object is not present we receive the signal R(
    n) = Noise
64 R=w;           // only Noise
    Signal
65 nr=nw;

```

```

66 nr_fold=mtlbfliplr(nr);
67 R_fold=mtlbfliplr(R);
68 nmin=min(n)+min(nr_fold);           // Lowest index of y
   (n)
69 nmax=max(n)+max(nr_fold);           // Highest index of
   y(n)
70 n_received=nmin:nmax;
71 Received_Absence=conv(x,R_fold); // Convolution of
   Original transmitted signal and Received Signal
   in the Absence of Object (Equivalent to
   Correlation)//
72
73 subplot(2,1,2);
74 bar(n_received,Received_Absence,0.1,'Green');
75 title('Correlation in the Absence of Object','color',
   'red','fontsize',4);
76 xlabel("Index", "fontsize",2,"color","blue");
77 ylabel("Correlation Value", "fontsize",2,"color",
   "blue");
78
79
80 // INPUT Parameters
81 //D=10;           // Delay amount
82 //x=[0 1 2 3 2 1 0]; //Triangle pulse
   transmitted by radar
83 //n=[-3 -2 -1 0 1 2 3]; // Index of Triangular
   Pulse
84
85 // OUT PUT Parameter
86 //R=[0.2113249    0.7560439    0.0002211
   0.3303271    0.6653811 0.6283918    0.8497452 ]
87 //Received_Presence=[0.    0.8497452    3.3278823
   8.4714004    15.245755    20.763547 22.706621
   19.050111    13.021551    6.1462834
   2.1786936 0.2113249    0]
88 //Received_Absence=[0.    0.8497452    2.3278823
   4.4714004    5.2457551    4.7635474 3.7066214
   3.0501113    3.0215507    2.1462834    1.1786936

```

Scilab code Solution 5.02 Binary Sequence Detection at Receiver

```
1 // Exp-9 : Application of Correlation to detect
  binary sequence at receiver.
2
3 // Objective : To transmute a Binary Signal "0" and
  "1", we used  $\sin(n\pi/8)$  and  $\sin(n\pi/4)$ 
  respectively. while transmission noise has been
  added with these signals. this information will
  be available to both receiver and transmitter. by
  finding correlation value between received
  signal with noise and the two corresponding
  signals used to represent binary signal "0" or
  "1", at the receiver side we can detect whether
  transmitted signal is binary "0" or "1"
4
5 // Version : Scilab 5.4.1
6 // Operating System : Window-xp, Window-7
7
8
9 clc;
10 clear;
11 xdel(winsid());
12 N=30; // Length of the signal
13 n=0:N-1;
14 pi=3.14;
15 x0=sin(n*pi/8); // For '0' transmission
16 x1=sin(n*pi/4); // For '1' transmission
17
18 scf();
19 subplot(2,1,1);
20 bar(n,x0,0.1,'red');
21 title('Sin(n*pi/8) to Represent Binary 0','color','
```

```

        red', 'fontsize', 4);
22 xlabel("Index", "fontsize", 2, "color", "blue");
23 ylabel("Amplitude", "fontsize", 2, "color", "blue");
24
25 subplot(2,1,2);
26 bar(n,x1,0.1, 'yellow');
27 title('Sin(n*pi/4) to Represent Binary 1', 'color', '
        red', 'fontsize', 4);
28 xlabel("Index", "fontsize", 2, "color", "blue");
29 ylabel("Amplitude", "fontsize", 2, "color", "blue");
30
31 w=rand(1,N); // Noise Signal
32 y0=x0+w; // Original Signal + Noise Signal
33 y1=x1+w; // Original Signal + Noise Signal
34
35 // If received signal is y0(n)
36 rx0y0=xcorr(x0,y0); // crosscorrelation
    between x0(n) and y0(n)
37 rx1y0=xcorr(x1,y0); // crosscorrelation
    between x1(n) and y0(n)
38
39 scf();
40 subplot(2,1,1);
41 bar(rx0y0,0.1, 'red');
42 title('Correlation between x0 and y0', 'color', 'red',
    'fontsize', 4);
43 xlabel("Index", "fontsize", 2, "color", "blue");
44 ylabel("Amplitude", "fontsize", 2, "color", "blue");
45
46 subplot(2,1,2);
47 bar(rx1y0,0.1, 'green');
48 title('Correlation between x1 and y0', 'color', 'red',
    'fontsize', 4);
49 xlabel("Index", "fontsize", 2, "color", "blue");
50 ylabel("Amplitude", "fontsize", 2, "color", "blue");
51
52
53 // If received signal is y1(n)

```

```

54 rx0y1=xcorr(x0,y1);          // crosscorrelation
    between x0(n) and y1(n)
55 rx1y1=xcorr(x1,y1);          // crosscorrelation
    between x1(n) and y1(n)
56
57 scf();
58 subplot(2,1,1);
59 bar(rx0y1,0.1,'red');
60 title('Correlation between x0 and y1','color','red',
    'fontsize',4);
61 xlabel("Index", "fontsize",2,"color","blue");
62 ylabel("Amplitude", "fontsize",2, "color", "blue");
63
64 subplot(2,1,2);
65 bar(rx1y1,0.1,'green');
66 title('Correlation between x1 and y1','color','red',
    'fontsize',4);
67 xlabel("Index", "fontsize",2,"color","blue");
68 ylabel("Amplitude", "fontsize",2, "color", "blue");

```

Experiment: 6

Musical Tone Generation [sa re
ga ma pa dh ni sa] with each
Tone has time duration 0.5 sec.

Scilab code Solution 6.01 Musical Tone Generation

```
1 // Exp-11 : Musical Tone Generation [sa re ga ma pa
   dh ni sa] with each Tone has time duration 0.5
   sec .
2
3 // Version : Scilab 5.4.1
4 // Operating Syatem : Window-xp, Window-7
5
6 clc;
7 close;
8 clear;
9 frequency=[240 254 302 320 358.5 380 451 470]; //
   Corresponding Frequency
10 fs=8000; //
   Sampling Frequency
11 no=8;
12 N=1:4000; // Total
   No. of Samples for Each tone
```

```
13
14 temp=[];
15 for i=1:no
16     y=sin(2*3.14*(frequency(i)/fs)*N);
17     temp=[temp y];
18 end
19 length(temp);
20 sound(temp,fs);
21
22 //savewave('D:\temp_Original.wav',temp,[8000]);
```

Experiment: 7

DTMF Signal Generation and Decoder according to the given Mobile Number.

Scilab code Solution 7.01 DTMF Signal Generation

```
1 // Exp-10 : DTMF Signal Generation for the given
  Mobile Number.
2
3 // Version : Scilab 5.4.1
4 // Operating System : Window-xp, Window-7
5
6 ////////////////////////////////////// DTMF //////////////////////////////////////
7 row_f1=[700 770 850 941]; // Row
  Frequency
8 colum_f1=[1220 1350 1490]; // Column
  Frequency
9 fs=8000; // Sampling
  Frequency
10 N=1:4000; // Total No.
  of Sample
11 mobile=[9 9 7 8 3 7 4 2 5 3]; // Mobile
  Number
```

```

12 temp=[]; // Array that
    Contain total signal for each Digit
13
14 figure;
15
16 for i=1:length(mobile)
17     select mobile(i)
18     case 1
19         row_f=1;
20         colum_f=1;
21     case 2
22         row_f=1;
23         colum_f=2;
24     case 3
25         row_f=1;
26         colum_f=3;
27     case 4
28         row_f=2;
29         colum_f=1;
30     case 5
31         row_f=2;
32         colum_f=2;
33     case 6
34         row_f=2;
35         colum_f=3;
36     case 7
37         row_f=3;
38         colum_f=1;
39     case 8
40         row_f=3;
41         colum_f=2;
42     case 9
43         row_f=3;
44         colum_f=3;
45     case 0
46         row_f=4;
47         colum_f=2;
48     else

```

```

49         row_f=4;
50         colum_f=1;
51     end
52
53     y=sin(2*3.14*(row_f1(row_f)/fs)*N)+sin(2*3.14*(
        colum_f1(colum_f)/fs)*N);
54
55     temp=[temp y zeros(1,4000)]; // Append the
        Signal + zeros After each Number
56
57     end
58     plot(temp);
59     sound(temp,fs);

```

Scilab code Solution 7.02 DTMF Signal Generation and Decoder

```

1 // Exp-12 : DTMF Signal Generation and Decoder
    according to the given Mobile Number.
2
3 // Version : Scilab 5.4.1
4 // Operating System : Window-xp, Window-7
5
6 ////////////////////////////////////// DTMF //////////////////////////////////////
7 clc;
8 close;
9 clear;
10 row_f1=[700 770 850 940]; // Row Frequency
11 colum_f1=[1220 1350 1490]; // Column Frequency
12 fs=8000; // Sampling Frequency
13 N=1:800; // Total No. of
        Samples for each Digit
14 mobile=[9 9 0 8 4 5 0 6 5 0];
15 total_signal=[];
16
17 figure;

```

```

18
19 for i=1:length(mobile)
20     select mobile(i)
21     case 1
22         row_f=1;
23         colum_f=1;
24     case 2
25         row_f=1;
26         colum_f=2;
27     case 3
28         row_f=1;
29         colum_f=3;
30     case 4
31         row_f=2;
32         colum_f=1;
33     case 5
34         row_f=2;
35         colum_f=2;
36     case 6
37         row_f=2;
38         colum_f=3;
39     case 7
40         row_f=3;
41         colum_f=1;
42     case 8
43         row_f=3;
44         colum_f=2;
45     case 9
46         row_f=3;
47         colum_f=3;
48     case 0
49         row_f=4;
50         colum_f=2;
51     else
52         row_f=4;
53         colum_f=1;
54     end
55     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
56     total_signal=[total_signal y zeros(1,8800)];
57     temp(:,:,i)=y(:,:,i);
58     end
59     plot(total_signal);
60     title('DTMF Signal','color','red','fontsize',4)
        ;
61     xlabel("Samples","fontsize",2,"color","blue")
        ;
62     ylabel("Amplitude","fontsize",2,"color","
        blue");
63     sound(total_signal,fs);
64
65
66     row_f=[];
67     col_f=[];
68
69     for i=1:10
70         n=length(temp(:,:,i));
71         p=abs(fft(temp(:,:,i))); // FFT of Signal of
            respective Digit
72         f=(0:n-1)*fs/n; // Total Frequency
            Range
73         //plot(f,p);
74         row=p(2:100); // Row Frequency
            separation
75         col=p(101:200); // Column Frequency
            separation
76         [r1 c1]=find(row==max(row)); // Finding the
            location of peak for Row Frequency
77         [r2 c2]=find(col==max(col)); // Finding the
            location of peak for Column Frequency
78         row_f=[row_f 10*c1]; // Array
            containing peak of Row Frequency
79         col_f=[col_f (10*(c2+100))-10]; // Array
            containing peak of Column Frequency
80     end

```

```

81
82 mobile_find=[]; // Blank Array to Store Mobile
    Number
83 for i=1:10 // Loop for Finding the Number form
    the Row and Column Frequency
84     if(row_f(i)==700 & col_f(i)==1220)
85         n0=1;
86         elseif(row_f(i)==700 & col_f(i)==1350)
87     n0=2;
88     elseif(row_f(i)==700 & col_f(i)==1490)
89     n0=3;
90     elseif(row_f(i)==770 & col_f(i)==1220)
91     n0=4;
92     elseif(row_f(i)==770 & col_f(i)==1350)
93     n0=5;
94     elseif(row_f(i)==770 & col_f(i)==1490)
95     n0=6;
96     elseif(row_f(i)==850 & col_f(i)==1220)
97     n0=7;
98     elseif(row_f(i)==850 & col_f(i)==1350)
99     n0=8;
100    elseif(row_f(i)==850 & col_f(i)==1490)
101    n0=9;
102    elseif(row_f(i)==940 & col_f(i)==1350)
103    n0=0;
104 end
105 mobile_find=[mobile_find n0]; // Array containing
    Decoded Digit of Mobile Number.
106 end
107
108 disp("Decoded Mobile Number :");
109 disp(mobile_find);
110
111
112 //////////////// Out Put Parameters
    ////////////////
113 //row_f=[850.    850.    940.    850.    770.
    770.    940.    770.    770. 940.];

```

```
114 // col_f=[1490.    1490.    1350.    1350.    1220.
           1350.    1350.    1490.  1350.  1350.];
115 // mobile_find=[9.    9.    0.    8.    4.    5.
                  0.    6.    5.    0.];
```

Experiment: 8

Design Low Pass Filter as per the given specification and plot the Frequency Response.

Scilab code Solution 8.01 Low Pass Filter Design

```
1 // Exp-13 : Design Low Pass Filter as per the given
   // specification and plot the Frequency Response.
2
3 // Version : Scilab 5.4.1
4 // Operating System : Window-xp, Window-7
5
6 clc;
7 close;
8 clear;
9 delta1=0.1; // Atenuation
10 delta2=0.1;
11 fl=400; // Low Cut-Off Frequency
12 fh=500; // High Cut-Off Frequency
13 fs=8000; // Sampling Frequency
14 A=-20*log10(min(delta1:delta2));
15 w1=2*%pi*fl/fs;
16 w2=2*%pi*fh/fs;
```



```

17 temp=1+((A-8)/(2.285*((2*3.14*fh/fs)-(2*3.14*fl/fs))
    ));
18 N=ceil((temp-1)/2);
19 n=-N:N;
20 h=((w2+w1)/2)*(sinc(((w2+w1)/2)*n))/(3.14); //
    Response
21 //plot(n,h);
22
23 [xm1,fr1]=fsmag(h,8000); // Frequency
    Response
24 figure;
25 plot(fr1,xm1);
26 title('Frequency Response','color','red','fontsize',
    4);
27 xlabel("Frequency (Normalized)", "fontsize", 2,"
    color", "blue");
28 ylabel("Magnitude", "fontsize", 2, "color", "blue");

```

Experiment: 9

Design High Pass Filter as per the given specification and plot the Frequency Response.

Scilab code Solution 9.01 High Pass Filter design

```
1 // Exp-14 : Design High Pass Filter as per the given
   // specification and plot the Frequency Response.
2
3 // Version : Scilab 5.4.1
4 // Operating System : Window-xp, Window-7
5
6 clc;
7 close;
8 clear;
9 delta1=0.1;
10 delta2=0.1;
11 fl=2;
12 fh=2.115;
13 fs=8;
14 A=-20*log10(min(delta1:delta2));
15 w1=2*3.14*fl/fs;
16 w2=2*3.14*fh/fs;
```

```

17 temp=1+((A-8)/(2.285*((2*3.14*fh/fs)-(2*3.14*f1/fs))
    ));
18 N=ceil((temp-1)/2);
19 n=-N:N;
20 del=[zeros(1:N) 1 zeros(N+1:2*N)];
21 h=del-(((w2+w1)/2)*(sinc(((w2+w1)/2)*n)))/(3.14);
    // High_Pass=1-Low_Pass
22 //h=[-h(1:30) 0 -h(32:61)];
23 //figure;
24 //plot(n,h);
25
26 [xmh,frh]=frmag(h,8000);
27 figure;
28 plot(frh,xmh);
29 title('Frequency Response','color','red','fontsize',
    4);
30 xlabel("Frequency (Normalized)", "fontsize", 2, "
    color", "blue");
31 ylabel("Magnitude", "fontsize", 2, "color", "blue");

```

Experiment: 10

Design Band Pass Filter as per the given specification and plot the Frequency Response.

Scilab code Solution 10.01 Band Pass Filter

```
1 // Exp-15 : Design Band Pass Filter as per the given
   // specification and plot the Frequency Response.
2
3 // Version : Scilab 5.4.1
4 // Operating System : Window-xp, Window-7
5
6 clc;
7 close;
8 clear;
9 delta1=0.1;
10 delta2=0.1;
11 f12=600;
12 fh2=700;
13 fs=8000;
14 A=-20*log10(min(delta1:delta2));
15 w12=2*3.14*f12/fs;
16 w22=2*3.14*fh2/fs;
```

```

17 temp2=1+((A-8)/(2.285*((2*3.14*fh2/fs)-(2*3.14*f12/
    fs)))));
18 N=ceil((temp2-1)/2);
19 n=-N:N;
20 h2=((w22+w12)/2)*(sinc(((w22+w12)/2)*n))/(3.14);
21
22 // [xm2, fr2]=frmag(h2,8000);
23 // figure;
24 // plot(fr2 ,xm2);
25
26 //////////////////////////////////////
27 delta1=0.1;
28 delta2=0.1;
29 f11=200;
30 fh1=300;
31 fs=8000;
32 A=-20*log10(min(delta1:delta2));
33 w11=2*3.14*f11/fs;
34 w21=2*3.14*fh1/fs;
35 temp1=1+((A-8)/(2.285*((2*3.14*fh1/fs)-(2*3.14*f11/
    fs)))));
36 N=ceil((temp1-1)/2);
37 n=-N:N;
38 h1=((w21+w11)/2)*(sinc(((w21+w11)/2)*n))/(3.14);
39
40 h=h2-h1;
41
42 [xmb, frb]=frmag(h,8000);
43
44 figure;
45 plot(fr2 ,xm2);
46 title('Frequency Response','color','red','fontsize',
    4);
47 xlabel("Frequency", "fontsize", 2,"color", "blue");
48 ylabel("Gain", "fontsize", 2, "color", "blue");

```

Experiment: 11

Design Equillizer (LOW + HIGH + BANDPASS) as per the given specification

Scilab code Solution 11.01 Equilizer Design

```
1 // Exp-16 : Design Equillizer (LOW + HIGH + BANDPASS
   ) as per the given specification
2
3 // Version : Scilab 5.4.1
4 // Operating Syatem : Window-xp, Window-7
5
6 clc;
7 close;
8 clear;
9
10 function [xm1,fr1]=low(f1,fh)
11
12 delta1=0.1;
13 delta2=0.1;
14 // f1=200;
15 // fh=300;
16 fs=8000;
```

```

17 A=-20*log10(min(delta1:delta2));
18 w1=2*3.14*f1/fs;
19 w2=2*3.14*fh/fs;
20 temp=1+((A-8)/(2.285*((2*3.14*fh/fs)-(2*3.14*f1/fs))
    ));
21 N=ceil((temp-1)/2);
22 n=-N:N;
23 h=(((w2+w1)/2)*(sinc(((w2+w1)/2)*n)))/(3.14);
24 //plot(n,h);
25
26 [xm1,fr1]=frmag(h,8000);
27 //figure;
28 //plot(fr1,xm1);
29
30 endfunction
31
32 //////////////////////////////////////
33 function [xmh,frh]=highpass(f1,fh)
34
35
36
37 delta1=0.1;
38 delta2=0.1;
39 //f1=400;
40 //fh=500;
41 fs=8000;
42 A=-20*log10(min(delta1:delta2));
43 w1=2*3.14*f1/fs;
44 w2=2*3.14*fh/fs;
45 temp=1+((A-8)/(2.285*((2*3.14*fh/fs)-(2*3.14*f1/fs))
    ));
46 N=ceil((temp-1)/2);
47 n=-N:N;
48 del=[zeros(1:N) 1 zeros(N+1:2*N)];
49 h=del-(((w2+w1)/2)*(sinc(((w2+w1)/2)*n)))/(3.14);
50 //h=[-h(1:30) 0 -h(32:61)];
51 //figure;
52 //plot(n,h);

```

```

53
54 [xmh, frh]=frmag(h,8000);
55 // figure;
56 //plot(frh,xmh);
57 endfunction
58
59
60 //////////////////////////////////////
61 function [xmb, frb]=bandpass(fl1, fh1, fl2, fh2)
62
63 delta1=0.1;
64 delta2=0.1;
65 // fl2=350;
66 // fh2=450;
67 fs=8000;
68 A=-20*log10(min(delta1:delta2));
69 w12=2*3.14*f12/fs;
70 w22=2*3.14*fh2/fs;
71 temp2=1+((A-8)/(2.285*((2*3.14*fh2/fs)-(2*3.14*f12/
    fs)))));
72 N=ceil((temp2-1)/2);
73 n=-N:N;
74 h2=(((w22+w12)/2)*(sinc(((w22+w12)/2)*n)))/(3.14);
75
76
77 // [xm2, fr2]=frmag(h2,8000);
78
79 //////////////////////////////////////
80 delta1=0.1;
81 delta2=0.1;
82 // fl1=250;
83 // fh1=350;
84 fs=8000;
85 A=-20*log10(min(delta1:delta2));
86 w11=2*3.14*f11/fs;
87 w21=2*3.14*fh1/fs;
88 temp1=1+((A-8)/(2.285*((2*3.14*fh1/fs)-(2*3.14*f11/
    fs)))));

```



```

89 N=ceil((temp1-1)/2);
90 n=-N:N;
91 h1=((w21+w11)/2)*(sinc(((w21+w11)/2)*n))/(3.14);
92
93 h=h2-h1;
94
95 [xmb,frb]=frmag(h,8000);
96
97
98 //figure;
99 //plot(frb,xmb);
100 endfunction
101
102 //////////////////////////////////////////////////Main Programm //////////////////////////////////
103
104 [y,fs]=wavread('original.wav'); // User can read
    any .wav file
105
106 ///////////////      LOW PASS      //////////////////////////////////
107 fl=200;                // Lower Cut-Off Frequency
108 fh=300;                // Higher Cut-Off Frequency
109 [xm1,fr1]=low(fl,fh); // Function for Low Pass
    Filter
110 ///////////////      HIGH PASS      //////////////////////////////////
111 fl=400;                // Lower Cut-Off Frequency
112 fh=500;                // Higher Cut-Off Frequency
113 [xmh,frh]=highpass(fl,fh); // Function for high
    Pass Filter
114 ///////////////      BAND PASS      //////////////////////////////////
    //////////////////////////////////
115 fl1=250;                // Lower Cut-Off Frequency1
116 fh1=350;                // Higher Cut-Off Frequency1
117 fl2=350;                // Lower Cut-Off Frequency2
118 fh2=450;                // Higher Cut-Off Frequency2
119 [xmb,frb]=bandpass(fl1,fh1,fl2,fh2); // Function
    for Band Pass Filter
120 //
    //////////////////////////////////

```

```

121
122 gain_L=2;           // Gain for Low Pass Frequency
123 gain_B=5;           // Gain for band Pass Frequency
124 gain_H=2;           // Gain for High Pass Frequency
125 sig_L=conv(y*gain_L,xm1);
126 sig_B=conv(y*gain_B,xmb);
127 sig_H=conv(y*gain_H,xmh);
128 sig_T=sig_L+sig_H+sig_B;
129 figure;
130 plot(y);
131 title('Original Signal (DTMF)', 'color', 'red', '
      fontsize', 4);
132 xlabel("Time Index (Samples)", "fontsize", 2, "color"
      , "blue");
133 ylabel("Amplitude", "fontsize", 2, "color", "blue");
134
135 figure;
136 plot(sig_T);
137 title('Filtered Signal (DTMF)', 'color', 'red', '
      fontsize', 4);
138 xlabel("Time Index (Samples)", "fontsize", 2, "color"
      , "blue");
139 ylabel("Amplitude", "fontsize", 2, "color", "blue");
140
141 sound(sig_T,fs);
142
143 //sound(y, fs);

```

Experiment: 12

To Generation of ECG signals used in Medical Processing

Scilab code Solution 12.01 ECG Wave Generation

```
1 // Exp-18 : To Generation of ECG signals used in
  Medical Processing
2
3 // Version : Scilab 5.4.1
4 // Operating System : Window-xp, Window-7
5
6 clc;
7 clear;
8 close;
9 cycle=zeros(1,500);
10 y=[01 2 3 4 5 6 7 8 9 10 9 8 7 6 5 4 3 2 1 0]/4;
    // Triangle pulse segment
11 t=[0:1:40];
    // Local time axis
12 a=(.05*(t-20).*(t-20)-20)/50;
    // Parabolic recovery segment
13 cycle(1:61)=2*[y a];
    // Place pulse in 1sec. cycle
14 cyc=filter([1 1 1 1 1], [1], cycle);
```

```
    // Smooth the corners
15 x=[cyc cyc cyc cyc cyc]; //
    Repetition used for 5 cycle of trace
16 [idum, nsize]=size(x);
17 t=[0:1:nsize-1]/500;
    // Sampling frequency 500Hz
18 plot(t,x);
19 title('ECG Signal', 'color', 'red', 'fontsize', 4);
20 xlabel("Time Index", "fontsize", 2, "color", "blue");
21 ylabel("Amplitude", "fontsize", 2, "color", "blue");
```

Experiment: 13

Design the following Low Pass analog filters with the given specification. (1) Butter Worth (2) Chebyshev-I (3) Chebyshev-II

Scilab code Solution 13.01 Low Pass Filter Design

```
1 // Exp-19 : Design the following Low Pass analog
  filters with the given specification.
2 //          (1) Butter Worth   (2) Chebyshev-I   (3)
  Chebyshev-II   (4) Elliptical
3
4
5 //Evaluate magnitude response of the filter
6
7 // Version : Scilab 5.4.1
8 // Operating System : Window-xp, Window-7
9
10 clc;
11 close;
```

```

12 clear;
13
14 fcut=5; //hz
15 n=5; //filter order
16 hc1=analpf(n, 'cheb1', [0.1 0], fcut*2*%pi);
17 hc2=analpf(n, 'cheb2', [0 0.1], fcut*2*%pi);
18 he=analpf(n, 'ellip', [0.1 0.1], fcut*2*%pi);
19 hb=analpf(n, 'butt', [0 0], fcut*2*%pi);
20 //hc1.dt='c'; hc2.dt='c'; he.dt='c'; hb.dt='c';
21 clf();
22 [fr, hf]=repfreq(hc1,0,15);
23 plot(fr,abs(hf), 'b');
24 [fr, hf]=repfreq(hc2,0,15);
25 plot(fr,abs(hf), 'y');
26 [fr, hf]=repfreq(he,0,15);
27 plot(fr,abs(hf), 'r');
28 [fr, hf]=repfreq(hb,0,15);
29 plot(fr,abs(hf), 'c');
30
31 xgrid();
32 legend(["Chebyshev I", "Chebyshev II", "Elliptic", "
    Butterworth"]);
33 xlabel("Frequency (Hz)");
34 ylabel("Gain");
35 title("Analog filters of order 5");

```

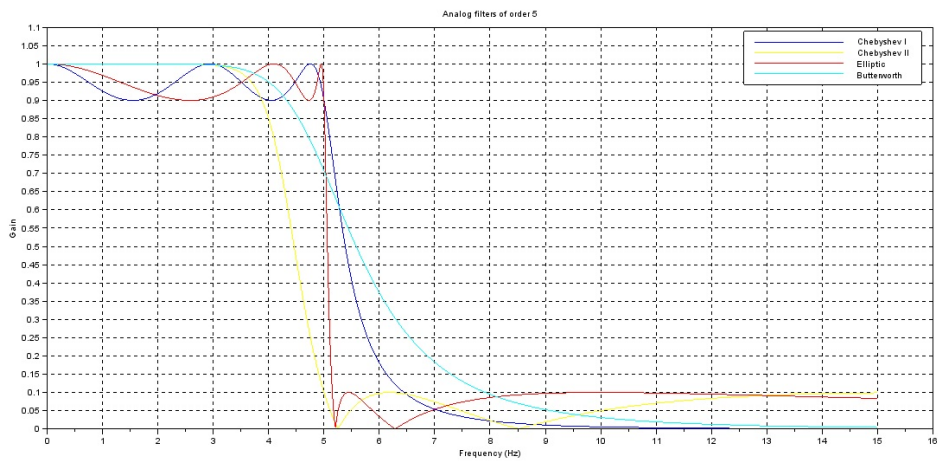


Figure 13.1: Low Pass Filter Design

Experiment: 14

Design the following IIR filters with the given specification. (1) Butter Worth (2) Chebyshev-I (3) Chebyshev-II (4) Elliptic

Scilab code Solution 14.01 IIR filter Design

```
1 // Exp-20 : Design the following IIR filters with
  the given specification.
2 //           (1) Butter Worth   (2) Chebyshev-I   (3)
  Chebyshev-II   (4) Elliptical
3
4 //
  ///////////////////////////////////////////////////////////////////
5 //   Example :
6 //           filter type ('lp', 'hp', 'sb', 'bp')
7 //           design approximation ('butt', '
cheb1', 'cheb2', 'ellip')
8 //           om=[om1,om2,om3,om4], 0 <= om1 <=
  om2 <= om3 <= om4 <= pi .When ftype='lp' or 'hp
', om3 and om4 are not used and may be set to 0.
```



```

9 //          0<= deltap <=1
10 //          0<= deltas <=1
11 //
//////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
12
13 //Evaluate magnitude response of the filter
14
15 // Version : Scilab 5.4.1
16 // Operating System : Window-xp, Window-7
17
18 clc;
19 close;
20 clear;
21 //ftype=input('Enter the Filter Type:', 's');
22 //approx=input('Enter the Filter Name:', 's');
23 //om=input('Enter Cut-Off Frequency Vector:');
24 //deltap=input('Enter Ripple in the Passband:');
25 //deltas=input('Enter ripple in the Stopband:');
26
27
28 ftype='lp';
29 approx='cheb2';
30 om=[0.3 0.5];
31 deltap=0.1;
32 deltas=0.4;
33
34
35 [cells, fact, zzeros, zpoles]=eqiir(ftype, approx, om,
    deltap, deltas);
36 h=fact*poly(zzeros, 'z')/poly(zpoles, 'z');
37
38 pole_real=[];
39 pole_imag=[];
40 for i=1:length(zpoles)
41     pole_real=[pole_real real(zpoles(i))];
42     pole_imag=[pole_imag imag(zpoles(i))];
43 end

```

```
44
45
46 mtlb_axis([-1 1 -1 1]);
47 xgrid();
48 plot(pole_real,pole_imag,'cya+', 'marker','d', '
      markerfac','red','markeredg','red');
49
50 title('Pole Location','color','red','fontsize',4);
51 xlabel("Real Axis", "fontsize",2,"color","blue");
52 ylabel("Imaginary Axis", "fontsize",2, "color", "
      blue");
```

Experiment: 15

Design the following FIR filters with the given specification.

(1) Low Pass (2) High Pass (3) Band Pass (4) Band Stop

Scilab code Solution 15.01 FIR Filter

```
1 // Exp-21 : Design the following FIR filters with
  the given specification.
2 //
3 //
4 //
  ///////////////////////////////////////////////////////////////////
5 //      Example:
6 //          filter type ('lp', 'hp', 'sb', 'bp')
7 //          Filter order (pos integer)(odd
  for ftype='hp' or 'sb')
8 //          cfreq=2-vector of cutoff
  frequencies (0<cfreq(1),cfreq(2)<.5) only cfreq
  (1) is used when ftype='lp' or 'hp'
9 //          wtype= Window type ('re', 'tr', 'hm
```

```

    ', 'hn', 'kr', 'ch')
10 //          fpar=2-vector of window
    parameters. Kaiser window fpar(1)>0 fpar(2)=0.
    Chebyshev window fpar(1)>0, fpar(2)<0 or fpar(1)
    <0, 0<fpar(2)<.5
11 //          wft=time domain filter
    coefficients
12 //          wfm=frequency domain filter
    response on the grid fr
13 //          fr=Frequency grid
14 //
    //////////////////////////////////////
15
16 //Evaluate magnitude response of the filter
17
18 // Version : Scilab 5.4.1
19 // Operating Syatem : Window-xp, Window-7
20
21 clc;
22 close;
23 clear;
24
25 //ftype=input('Enter the Filter Type:', 's');
26 //forder=input('Enter Order of Filter:');
27 //cfreq=input('Enter Cut-Off Frequency Vector:');
28 //wtype=input('Enter the Window Type:', 's');
29 //fpar=input('Enter Window Parameter:');
30
31 ftype='bp';
32 forder=33;
33 fs=8000;
34 cfreq=[(450/fs) (500/fs)];
35 wtype='kr';
36 fpar=[0.8 0];
37
38
39 [wft, wfm, fr]=wfir(ftype, forder, cfreq, wtype, fpar);

```

```
40
41 clf();
42 plot(fr,wfm,'b');
43 title('Frequency Responce','color','red','fontsize',
44       4);
44 xlabel("Frequency", "fontsize", 2,"color", "blue");
45 ylabel("Magnitude", "fontsize", 2, "color", "blue");
```
