

Scilab Manual for  
Digital Signal Processing  
by Mr Kaustubh Shivaji Sagale  
Electronics Engineering  
S.S.V.P.S's B.S.D. C.O.E.<sup>1</sup>

Solutions provided by  
Mr R. Senthilkumar  
Electrical Engineering  
IRTT

April 4, 2026

<sup>1</sup>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 Basic operations on sequences of equal and unequal lengths. | 4  |
| 2 Sampling of continuous time signal and aliasing effect.     | 11 |
| 3 Convolution of two sequence Impulse response.               | 13 |
| 4 Spectrum of signals using DFT.                              | 16 |
| 5 Designing of FIR Filter.                                    | 19 |
| 6 Designing of IIR Filter.                                    | 21 |

# List of Experiments

|              |   |    |
|--------------|---|----|
| Solution 1.1 | Program for addition of two sequences of unequal lengths . . . . .          | 4  |
| Solution 1.2 | Program for multiplication of two sequences of unequal lengths . . . . .    | 5  |
| Solution 1.3 | Program to Demonstrate the signal Folding . . . . .                         | 6  |
| Solution 1.4 | Program to demonstrate the Amplitude and Time Scaling of a signal . . . . . | 7  |
| Solution 1.5 | Program to demonstrate the shifting of the discrete time signal . . . . .   | 8  |
| Solution 1.6 | Program to Generate Sine Waveform for one cycle                             | 10 |
| Solution 2.1 | Program to demonstrate the Aliasing Effect . . . . .                        | 11 |
| Solution 3.1 | Program to Compute the Convolution of Two Sequences . . . . .               | 13 |
| Solution 4.1 | Program to find the spectral information of discrete time signal . . . . .  | 16 |
| Solution 5.1 | To Design an Low Pass FIR Filter . . . . .                                  | 19 |
| Solution 6.1 | To Design Digital IIR Butterworth LPF . . . . .                             | 21 |

# Experiment: 1

## Basic operations on sequences of equal and unequal lengths.

**Scilab code Solution 1.1** Program for addition of two sequences of unequal lengths

```
1 //Caption:Program for addition of two sequences of
   unequal lengths
2 clc;
3 clear;
4 close;
5 x = input('Enter Seq.= ');
6 y = input('Enter Seq.= ');
7 m = length(x);
8 n = length(y);
9 if m>n then
10     y = [y,zeros(1,m-n)];
11 elseif n>m
12     x = [x,zeros(1,n-m)];
13 end
14 z = x+y;
15 disp(z,'Addition result of two unequal length
   sequences:= ');
16 //Example
```

```

17 //Enter Seq.=[1,2,3]
18 //
19 //Enter Seq.=[1,1,1,1]
20 //
21 // Addition result of two unequal length sequences:=
22 //
23 //      2.      3.      4.      1.

```

---

**Scilab code Solution 1.2** Program for multiplication of two sequences of unequal lengths

```

1 //Caption:Program for multiplication of two
  sequences of unequal lengths
2 clc;
3 clear;
4 close;
5 x = input('Enter Seq.= ');
6 y = input('Enter Seq.= ');
7 m = length(x);
8 n = length(y);
9 if m>n then
10     y = [y,zeros(1,m-n)];
11 elseif n>m
12     x = [x,zeros(1,n-m)];
13 end
14 z = x.*y;
15 disp(z, 'Multiplication result of two unequal length
  sequences:= ');
16 //Example
17 //Enter Seq.=[1,2,3]
18 //
19 //Enter Seq.=[1,1,1,1]
20 //
21 // Multiplication result of two unequal length
  sequences:=

```

```
22 //
23 //      1.      2.      3.      0.
```

---

**Scilab code Solution 1.3** Program to Demonstrate the signal Folding

```
1 //Caption: Program to Demonstrate the signal Folding
2 clc;
3 clear;
4 x = input('Enter the input sequence:=');
5 m = length(x);
6 lx = input('Enter the starting point of original
      signal=');
7 hx = lx+m-1;
8 n = lx:1:hx;
9 subplot(2,1,1)
10 a = gca();
11 a.x_location = "origin";
12 a.y_location = "origin";
13 a.data_bounds = [-5,0;5,5];
14 plot2d3('gnn',n,x)
15 xlabel('n====>')
16 ylabel('Amplitude---->')
17 title('Original Sequence')
18 subplot(2,1,2)
19 a = gca();
20 a.x_location = "origin";
21 a.y_location = "origin";
22 a.data_bounds = [-5,0;5,5];
23 plot2d3(-n,x)
24 xlabel('n====>')
25 ylabel('Amplitude---->')
26 title('Folded Sequence')
```

---

**Scilab code Solution 1.4** Program to demonstrate the Amplitude and Time Scaling of a signal

```
1 //Caption: Program to demonstrate the Amplitude &
   Time Scaling of a signal
2 clc;
3 clear;
4 x = input('Enter input Sequence:= ');
5 m = length(x);
6 lx = input('Enter starting point of original signal
   := ');
7 hx = lx+m-1;
8 n = lx:1:hx;
9 subplot(2,2,1)
10 a = gca();
11 a.x_location = "origin";
12 a.y_location = "origin";
13 a.data_bounds = [-10,0;10,10];
14 plot2d3('gnn',n,x)
15 xlabel('n====>')
16 ylabel('Amplitude====>')
17 title('original sequence')
18 //Amplitude Scaling
19 a = input('Amplitude Scaling Factor:= ')
20 y =a*x;
21 subplot(2,2,2)
22 a = gca();
23 a.x_location = "origin";
24 a.y_location = "origin";
25 a.data_bounds = [-10,0;10,10];
26 plot2d3('gnn',n,y)
27 xlabel('n====>')
28 ylabel('Amplitdue====>')
29 title('Amplitude Scaled Sequence')
30 //Time Scaling-Compression
31 C = input('Enter Compression factor-Time Scaling
   factor ')
32 n = lx/C:1/C:hx/C;
```

```

33 subplot(2,2,3)
34 a = gca();
35 a.x_location = "origin";
36 a.y_location = "origin";
37 a.data_bounds = [-10,0;10,10];
38 plot2d3('gnn',n,x)
39 xlabel('n====>')
40 ylabel('Amplitude---->')
41 title('Compressed Sequence')
42 //Time Scaling-Expansion
43 d = input('Enter Extension factor-Time Scaling
           factor ')
44 n = lx*d:d:hx*d;
45 subplot(2,2,4)
46 a = gca();
47 a.x_location = "origin";
48 a.y_location = "origin";
49 a.data_bounds = [-10,0;10,10];
50 plot2d3('gnn',n,x)
51 xlabel('n====>')
52 ylabel('Amplitude---->')
53 title('Extended Sequence')
54 //Example
55 //Enter input Sequence:=[1,2,3,4,5]
56 //
57 //Enter starting point of original signal:= 2
58 //
59 //Amplitude Scaling Factor:= 2
60 //
61 //Enter Compression factor-Time Scaling factor 2
62 //
63 //Enter Extension factor-Time Scaling factor 2

```

---

**Scilab code Solution 1.5** Program to demonstrate the shifting of the discrete time signal

```

1 //Caption:Program to demonstrate the shifting of the
   discrete time signal
2 clc;
3 clear;
4 close;
5 x = input('Enter the input sequence:=')
6 m = length(x);
7 lx = input('Enter the starting point of original
   signal:=')
8 hx = lx+m-1;
9 n = lx:1:hx;
10 subplot(3,1,1)
11 a = gca();
12 a.x_location = "origin";
13 a.y_location = "origin";
14 a.data_bounds = [-10,0;10,10];
15 plot2d3('gnn',n,x);
16 xlabel('n====>')
17 ylabel('Amplitdue——>')
18 title('Original Sequence')
19 //
20 d = input('Enter the delay:=')
21 n = lx+d:1:hx+d;
22 subplot(3,1,2)
23 a = gca();
24 a.x_location = "origin";
25 a.y_location = "origin";
26 a.data_bounds = [-10,0;10,10];
27 plot2d3('gnn',n,x)
28 xlabel('n====>')
29 ylabel('Amplitude——>')
30 title('Delayed Sequence')
31 //
32 a = input('Enter the advance:=')
33 n = lx-a:1:hx-a;
34 subplot(3,1,3)
35 a = gca();
36 a.x_location = "origin";

```

```

37 a.y_location = "origin";
38 a.data_bounds = [-10,0;10,10];
39 plot2d3('gmn',n,x)
40 xlabel('n====>')
41 ylabel('Amplitude——>')
42 title('Advanced Sequence')
43 //Example
44 //Enter the input sequence:=[1,2,3,4,5]
45 //
46 //Enter the starting point of original signal:=0
47 //
48 //Enter the delay:=2
49 //
50 //Enter the advance:=3

```

---

**Scilab code Solution 1.6** Program to Generate Sine Waveform for one cycle

```

1 //Caption: Program to Generate Sine Waveform for one
  cycle
2 clc;
3 clear;
4 close;
5 t = 0:0.01:1;
6 f = 1; //frequency = 1Hz (i.e) one cycle
7 y = sin(2*%pi*f*t);
8 plot(2*%pi*t,y)
9 xlabel('time——>')
10 ylabel('Amplitude——>')
11 title('Sinusoidal Waveform')
12 xgrid(1)

```

---

## Experiment: 2

# Sampling of continuous time signal and aliasing effect.

Scilab code Solution 2.1 Program to demonstrate the Aliasing Effect

```
1 //Caption: Program to demonstrate the Aliasing
  Effect
2 clc;
3 clear;
4 close;
5 f = input('Enter the frequency of Continuous Time
  Signal:= ')
6 t = 0:0.00001:1/f;
7 xt = 3*sin(2*%pi*f*t);
8 subplot(2,2,1)
9 a = gca();
10 a.x_location = "origin";
11 a.y_location = "origin";
12 plot(t,xt)
13 title('Continuous Time Signal')
14 fs = input('Enter the Sampling frequency Fs='); // fs
  = 3000Hz
15 fd = f/fs;
16 n = 0:0.01:1/fd;
```

```

17 xn = 3*sin(2*pi*fd*n);
18 subplot(2,2,2)
19 a = gca();
20 a.x_location = "origin";
21 a.y_location = "origin";
22 plot2d3('gmn',n,xn)
23 title('Discrete Time Signal')
24 x1 = 3300;
25 x1n = 3*sin(2*pi*(x1/fs)*n);
26 subplot(2,2,3)
27 a = gca();
28 a.x_location = "origin";
29 a.y_location = "origin";
30 plot2d3('gmn',n,x1n)
31 title('Samples of 3300 Hz signal with fs = 3000 Hz')
32 x2 = 30300;
33 x2n = 3*sin(2*pi*(x2/fs)*n);
34 subplot(2,2,4)
35 a = gca();
36 a.x_location = "origin";
37 a.y_location = "origin";
38 plot2d3('gmn',n,x2n)
39 title('Samples of 30300 Hz signal with fs = 3000 Hz'
    )
40 //Example
41 //Enter the frequency of Continuous Time Signal:=
    1500
42 //
43 //Enter the Sampling frequency Fs= 3000

```

---

## Experiment: 3

# Convolution of two sequence Impulse response.

**Scilab code Solution 3.1** Program to Compute the Convolution of Two Sequences

```
1 //Caption: Program to Compute the Convolution of Two
   Sequences
2 clc;
3 clear;
4 close;
5 x = input('Enter the input Sequence:= ');
6 m = length(x);
7 lx = input('Enter the lower index of input sequence
   := ');
8 hx = lx+m-1;
9 n = lx:1:hx;
10 h = input('Enter impulse response sequence:= ');
11 l = length(h);
12 lh = input('Enter the lower index of impulse
   response:= ');
13 hh = lh+l-1;
14 g = lh:1:hh;
15 nx = lx+lh;
```

```

16 nh = nx+m+1-2;
17 y = convol(x,h)
18 r = nx:nh;
19 subplot(3,1,1)
20 a = gca();
21 a.x_location = "origin";
22 a.y_location = "origin";
23 plot2d3('gmn',n,x)
24 xlabel('n====>')
25 ylabel('Amplitude—>')
26 title('Input Sequence x[n]')
27 subplot(3,1,2)
28 a = gca();
29 a.x_location = "origin";
30 a.y_location = "origin";
31 plot2d3('gmn',g,h)
32 xlabel('n====>')
33 ylabel('Amplitude—>')
34 title('Impulse Response Sequence h[n]=')
35 subplot(3,1,3)
36 a = gca();
37 a.x_location = "origin";
38 a.y_location = "origin";
39 plot2d3('gmn',r,y)
40 xlabel('n====>')
41 ylabel('Amplitude—>')
42 title('Output Response Sequence y[n]=')
43 //Example
44 //Enter the input Sequence:=[1,2,3,1]
45 //
46 //Enter the lower index of input sequence:=0
47 //
48 //Enter impulse response sequence:=[1,2,1,-1]
49 //
50 //Enter the lower index of impulse response:=-1
51 //
52 //
53 //—>y

```

54 // y =  
55 //  
56 // 1. 4. 8. 8. 3. - 2. - 1.  
57 //

---

# Experiment: 4

## Spectrum of signals using DFT.

**Scilab code Solution 4.1** Program to find the spectral information of discrete time signal

```
1 //Caption: Program to find the spectral information
   of discrete time signal
2 clc;
3 close;
4 clear;
5 xn = input('Enter the real input discrete sequence x
   [n]= ');
6 N = length(xn);
7 XK = zeros(1,N);
8 IXK = zeros(1,N);
9 //Code block to find the DFT of the Sequence
10 for K = 0:N-1
11     for n = 0:N-1
12         XK(K+1) = XK(K+1)+xn(n+1)*exp(-%i*2*%pi*K*n/
           N);
13     end
14 end
15 [phase,db] = phasemag(XK)
16 disp(XK,'Discrete Fourier Transform X(k)=')
17 disp(abs(XK),'Magnitude Spectral Samples=')
```

```

18 disp(phase, 'Phase Spectral Samples=')
19 n = 0:N-1;
20 K = 0:N-1;
21 subplot(2,2,1)
22 a = gca();
23 a.x_location = "origin";
24 a.y_location = "origin";
25 plot2d3('gmn',n,xn)
26 xlabel('Time Index n——>')
27 ylabel('Amplitude xn——>')
28 title('Discrete Input Sequence')
29 subplot(2,2,2)
30 a = gca();
31 a.x_location = "origin";
32 a.y_location = "origin";
33 plot2d3('gmn',K,abs(XK))
34 xlabel('Frequency Sample Index K——>')
35 ylabel('|X(K)|---->')
36 title('Magnitude Spectrum')
37 subplot(2,2,3)
38 a = gca();
39 a.x_location = "origin";
40 a.y_location = "origin";
41 plot2d3('gmn',K,phase)
42 xlabel('Frequency Sample Index K——>')
43 ylabel('<X(K) in radians——>')
44 title('Phase Spectrum')
45 //Code block to find the IDFT of the sequence
46 for n = 0:N-1
47     for K = 0:N-1
48         IXK(n+1) = IXK(n+1)+XK(K+1)*exp(%i*2*%pi*K*n
49             /N);
49     end
50 end
51 IXK = IXK/N;
52 ixn = real(IXK);
53 subplot(2,2,4)
54 a = gca();

```

```

55 a.x_location = "origin";
56 a.y_location = "origin";
57 plot2d3('gnn',[0:N-1],ixn)
58 xlabel('Discrete Time Index n ---->')
59 ylabel('Amplitude x[n]---->')
60 title('IDFT sequence')
61 //Example
62 //
63 //Enter the real input discrete sequence x[n
    ]=[1,2,3,4]
64 //
65 // Discrete Fourier Transform X(k)=
66 //
67 //      10.   - 2. + 2.i   - 2. - 9.797D-16i   - 2. - 2.i
68 //
69 // Magnitude Spectral Samples=
70 //
71 //      10.      2.8284271      2.      2.8284271
72 //
73 // Phase Spectral Samples=
74 //
75 //      0.      135.      180.      225.
76 //

```

---

# Experiment: 5

## Designing of FIR Filter.

Scilab code Solution 5.1 To Design an Low Pass FIR Filter

```
1 //Caption: To Design an Low Pass FIR Filter
2 //Filter Length =5, Order = 4
3 //Window = Rectangular Window
4 //Ana
5 clc;
6 clear;
7 xdel(winsid());
8 fc = input("Enter Analog cutoff freq. in Hz=")
9 fs = input("Enter Analog sampling freq. in Hz=")
10 M = input("Enter order of filter =")
11 w = (2*%pi)*(fc/fs);
12 disp(w, 'Digital cutoff frequency in radians.cycles/
    samples');
13 wc = w/%pi;
14 disp(wc, 'Normalized digital cutoff frequency in
    cycles/samples');
15 [wft,wfm,fr]=wfir('lp',M+1,[wc/2,0], 're',[0,0]);
16 disp(wft, 'Impulse Response of LPF FIR Filter:h[n]=')
    ;
17 //Plotting the Magnitude Response of LPF FIR Filter
18 subplot(2,1,1)
```

```

19 plot(2*fr,wfm)
20 xlabel('Normalized Digital Frequency  $w \longrightarrow$ ')
21 ylabel('Magnitude  $|H(w)| =$ ')
22 title('Magnitude Response of FIR LPF')
23 xgrid(1)
24 subplot(2,1,2)
25 plot(fr*fs,wfm)
26 xlabel('Analog Frequency in Hz  $f \longrightarrow$ ')
27 ylabel('Magnitude  $|H(w)| =$ ')
28 title('Magnitude Response of FIR LPF')
29 xgrid(1)
30 //Example
31 //Enter Analog cutoff freq. in Hz= 250
32 //
33 //Enter Analog sampling freq. in Hz= 2000
34 //
35 //Enter order of filter = 4
36 //
37 // Digital cutoff frequency in radians.cycles/
    samples
38 //
39 //      0.7853982
40 //
41 // Normalized digital cutoff frequency in cycles/
    samples
42 //
43 //      0.25
44 //
45 // Impulse Response of LPF FIR Filter:h[n]=
46 //
47 //      0.1591549      0.2250791      0.25      0.2250791
    0.1591549

```

---

# Experiment: 6

## Designing of IIR Filter.

Scilab code Solution 6.1 To Design Digital IIR Butterworth LPF

```
1 //Caption: To Design Digital IIR Butterworth LPF
2 //Analog cutoff freq = 1000 Hz, Sampling Freq =
   10000 samples/sec
3 //Order of IIR filter N = 2
4 clc;
5 clear;
6 xdel(winsid());
7 fc = input('Enter cutoff freq in Hz fc =')
8 fs = input('Enter sampling freq in Hz fs =')
9 N = input('Enter order of Butterworth filterN =')
10 Fp = 2*fc/fs; //Pass band edge frequency in cycles/
   samples
11 [Hz]=iir(N, 'lp', 'butt', [Fp/2,0],[0,0]) //digital IIR
   Butterworth Filter
12 [Hw,w] = frmag(Hz,256);
13 subplot(2,1,1)
14 plot(2*w,abs(Hw));
15 xlabel('Normalized Digital Frequency w-->')
16 ylabel('Magnitude |H(w)|=')
17 title('Magnitude Response of IIR LPF')
18 xgrid(1)
```

```

19 subplot(2,1,2)
20 plot(2*w*fs,abs(Hw));
21 xlabel('Analog Frequency in Hz f ---->')
22 ylabel('Magnitude |H(w)|=')
23 title('Magnitude Response of IIR LPF')
24 xgrid(1)
25 //Example
26 //
27 //Enter cutoff freq in Hz fc =1000
28 //
29 //Enter sampling freq in Hz fs =10000
30 //
31 //Enter order of Butterworth filterN = 2
32 // -->Hz
33 // Hz =
34 //
35 //
36 // 
$$\frac{0.0674553 + 0.1349105z + 0.0674553z^2}{0.4128016 - 1.1429805z + z^2}$$

37 //
38 //
39 //

```

---