Scilab Manual for Random Signal Analysis by Mrs Ashwini S. Sawant Electronics Engineering VESIT[1]

Solutions provided by
Mrs Ashwini S. Sawant,Assistant Professor
Electronics Engineering
Mumbai/V.E.S.I.T.

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

Simulation of Discrete Random Variable and Estimation of its PDF and CDF.

Scilab code Solution 1.1 1

```
1 //Generate a Cumulative Distibution function and
     Probability Density Function
2 //plot of discrete random variables where x=1,2,3
     and their respective
3 //probabilities are given by 3/15,7/15,5/15
4
5 x
     =[1,2,3];.....................................................
     //Random Variable
6
7 px=[3/15 7/15
     5/15];.....................................
     //Probability of RV
8
9 y=cumsum(px)
     ;.....................................................
     //Cumulative sum of probabilities.
```

```
10
11 title("Simulation of Discrete RV and Estimation of
     CDF and PDF");//Title
12
13 subplot(121)
14
15 plot2d2(x,y,1,"111",rect=[0,0,4,1.25])
     ;.....................//plot step function
16
17 xlabel("RANDOM VARIABLE X","fontsize",5);
18
19 ylabel("CDF F(x)","fontsize",5);
20
21 title("Cumulative Distribution Function","fontsize"
     ,5);
22
23 subplot(122)
24
25 plot2d3(x,px,rect=[0,0,4,1])
     ;.................................//plot stem
     function
26
27 xlabel("Random Variable X","fontsize",5);
28
29 ylabel("PDF f(x)","fontsize",5);
30
31 title("Probability Density Function","fontsize",5);
```

Experiment: 2

Relation between Distribution(CDF) and Density Function(PDF).

Scilab code Solution 2.1 2

```
1 //Relation between Cumulative Distribution Function
    and Probability Density Function.
2
3 //Generate a set of 100 uniform random variable in
    the range of -2 to 2.
4 //Plot Histogram for the same.
5
6 stacksize("max");.....//Maximizing
    Size
7
8 x=grand(1,100,'unf',-2,2);.....//Generate
    random numbers
9
10 w=unique(x);.....//Arrange in
    ascending order
11
12 p=100^(-1)*ones(1,100);.....//Calculate
```

```
    probability
13
14 subplot(2,2,1.5);
15
16 histplot(10,w);.....//To plot
    Histogram
17
18 xlabel("Random Numbers");
19
20 ylabel("Amplitude");
21
22 title("Histogram","fontsize",4);
23
24 z=cumsum(p);.....//Take
    cumulative sum of probabilities
25
26 subplot(2,2,3);
27
28 plot2d2(w,z);.....//Plot CDF
29
30 xlabel("Random Numbers");
31
32 ylabel("Amplitude");
33
34 title("CDF","fontsize",4);
35
36 subplot(2,2,4);
37
38 plot2d3(w,z,p);.....//Plot PDF
39
40 title("PDF","fontsize",4);
41
42 xlabel("Random Numbers");
43
44 ylabel("Amplitude");
```

Experiment: 3

Study of Gaussian Distribution.

Scilab code Solution 3.1 3

```
1  //GAUSSIAN DISTRIBUTION
2
3  stacksize("max");
4
5  x=[-2:0.1:2];.....//
       defining X-axis
6
7  s1=0.5^0.5;.....//
       defining variance_1
8
9  s2=0.8^0.8;.....//
       defining variance_2
10
11 gd1=(1/((2*%pi)^0.5*s1)*%e^-((x/s1)^2/2));.....//
       Expression for Gaussian Distribution
12
13 plot2d1(x,gd1);
14
15 gd2=(1/((2*%pi)^0.5*s2)*%e^-((x/s2)^2/2));.....//
       Expression for Gaussian Distribution
16
```

```
17 plot2d1(x,gd2);
18
19 xlabel("X");
20
21 ylabel("f(x)");
22
23 title("Gaussian Distribution","fontsize",5);
24
25 e=gce();.....//
     Changing Line style
26
27 e1=e.children(1);.....//Call
     the Function
28
29 e1.thickness=1;.....//
     Thickness of the Line
30
31 e1.line_style=3;.....//Line
     style selection
32
33 h1=legend(['f1=var=0.5','f2=var=0.8']);.....//Plot
     for 2 variances.
```

Experiment: 4

Study of Power Spectral Density.

Scilab code Solution 4.1 4

```
1  //POWER SPECTRAL DENSITY
2
3  sample_rate
      =1000;.....................................
      //No. of bit intervals.
4
5  t=0:1/sample_rate
      :10;.....................................//time
      axis for pulse shape
6
7  N=size(t,'*');
8
9  x=cos(2*%pi*47*t)+cos((2*%pi*219*t)+(%pi/4));
10
11 f=sample_rate*(0:(N/2))/N;
12
13 n=size(f,'*');
14
15 sm=pspect(N/2,N,'hn',x)
```

```
    ;.....................................// two sided
    cross-spectral estimate
16                                                          //
                                                               between
                                                               2
                                                               discrete
                                                               time
                                                               signals
17
18  plot2d("gnl",f,sm(1:n));
19
20  xlabel('Frequency','fontsize',4);
21
22  title('POWER SPECTRAL DENSITY','fontsize',5);
23
24  ylabel('Power Content of the given signal','fontsize
    ',4);
```

Experiment: 5

Study of Gaussian Random Process.

Scilab code Solution 5.1 5

```
1  //GAUSSIAN RANDOM PROCESS
2
3  stacksize('max');
4
5  u=[50:50];
6
7  sig=[100,30;30,140];.....//
       defining covariance matrix....Cx(ti,tj)
8
9  a=1./((sqrt(det(sig)))*(sqrt(2*%pi)));
10
11 for(i=1:100)
12     for(j=1:100)
13         x=[i;j]
14         xn=x-u;
15         b(i,j)=exp(-(xn'*inv(sig)*xn*0.5));
16     end
17 end
18
```

```
19 f=a*b;.....//PDF
      =fx1,x2,....xn=[1/(sqrt(2*pi)*sigma)^n)*(exp^(-(
      xi-m)^2/(2*(sigma^2)))]
20
21 surf(f);
22
23 xlabel('x random variable','fontsize',2);
24
25 ylabel('y random variable','fontsize',2);
26
27 zlabel('PDF','fontsize',2);
28
29 title('Gaussian Random Process','fontsize',5);
```

Experiment: 6

M/M/1 queueing system

Scilab code Solution 6.1 6

```
1 //M/M/1 Queueing System,Average Arrival rate lambd
    =20customers per hour and avg. service time=1/u=2
    minutes.
2
3 a=20;
                                                    //
    Arrival rate(customers/hour)
4
5 b=30;
                                                    //
    Departure Rate(1/b=avg.service time)
6
7 N=25;
                                                    //
    Number of arrivals in simulation
8
9 x=-log(rand(1,N))/a;
                                          //Random
    interarrival time
10
11 x=cumsum(x);
```

```
                                                    //Random
    arrival time
12
13 y=-log(rand(1,N))/b;
                                                //Service times
    for each customer
14
15 ser_start=x(1);
                                                    //First
    customer starts service
16                                                              //
                                                                immediately
                                                                upon
                                                                arrival
                                                                .
17
18 z(1)=ser_start+y(1);
                                                //Departure time
    of first customer
19
20 for k=2:N
                                                            //kth
    customer
21     ser_start=max([z(k-1),x(k)]);
                                //beginning of service
       time
22     z(k)=ser_start+y(k);
                                                //end of service
       time.
23 end
24
25 //Construct date to plot graph of queue size vs.
    time.
26
27 xaxis=[0, x(1)];
```

```
                                                    //Vector of
    points for the M/M/1
28
29 yaxis=[0, 0];
                                                   //Vector
    of queue sizes at points in preceding vector.
30
31 qs=1;
                                                              //
    Current queue size.
32
33 x=x(2:length(x));
34
35 while length(x)>0
36     if x(1)<z(1)
                                                  //Next
        point is arrival.
37         qs=qs+1;
                                                          //
            Increase queue size
38         xaxis=[xaxis xaxis(length(xaxis)) x(1)];
39         yaxis=[yaxis qs qs];
40         x=x(2:length(x));
41     else

        //Next point is departure.
42         qs=qs-1;
                                                          //
            Decrease queue size
43         xaxis=[xaxis xaxis(length(xaxis)) z(1)];
44         yaxis=[yaxis qs qs];
45         z=z(2:length(z));
46     end
47 end
48
49 plot(xaxis,yaxis)
                                                 //Plot
    realization of birth-death process.
```

```
50
51 xlabel('time(hours)',fontsize',4);
52
53 ylabel('queue size','fontsize',4)
```