STOCHASTIC PROCESS LAB FILE

Submitted to:

Dr.Vivek Kumar Agarwal

Submitted by:

Ashish Gupta

2K16/MC/023

Department of Applied Mathematics

Delhi Technological University

INDEX

SNO.	EXPERIMENT	DATE	SIGNATURE
1.	Simulation of discrete parameter stochastic processes		
2.	Simulation of continuous parameter stochastic processes		
3.	Homogeneous and Non Homogenous Bernoulli Process		
4.	Homogenous, Non Homogenous Poisson Process and Renewal Process with Uniform Distribution		
5.	Simple Random Walk (Unrestricted)		
6.	Random walk with absorbing barriers		
7.	Random walk using Markov chains (absorbing and reflecting barriers)		
8.	Ergodic chain Steady state probabilities		
9.	M/M/1 Queuing Model		

EXPERIMENT # 1

QUESTION: Simulate the following discrete parameter stochastic processes

a Discrete State Space: No. of cars washed on n_{th} day in a car wash given minimum 20 cars are washed and maximum 30 cars are washed.

b Continuous State Space: Average time taken for a car to be washed on n_{th} day of month given time required is 2 minutes and maximum time taken is 4minutes.

CODE:

1) Discrete Parameter Discrete State Space –

```
x = [1:1:30];
y = 20 + randi([0 10],30,1);
p = scatter(x,y);
xlabel('Days of Month');
ylabel('No. of Car Washed on Nth Day');
title('Discrete Parameter - Discrete State Space');
```

2) Discrete Parameter Continuous State Space -

```
x = [1:1:30];
y = 120 + 120.*rand(30,1);
p = scatter(x,y);
xlabel('Days of Month');
ylabel('Avg Time Taken for Car Wash');
title('Discrete Parameter - Continuous State Space');
```

OUTPUT:

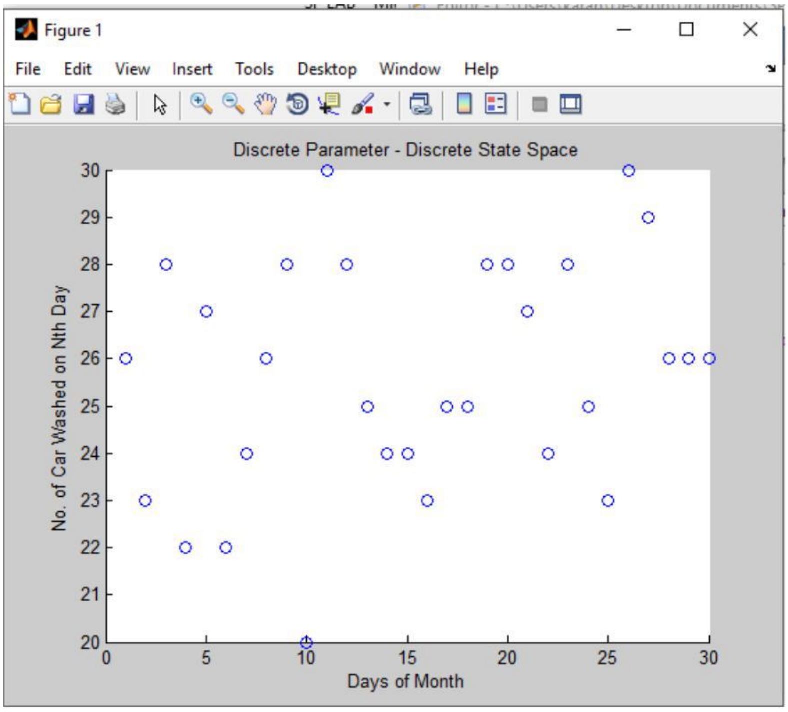

Figure 1
File Edit View Insert Tools Desktop Window Help
Discrete Parameter - Discrete State Space
No. of Car Washed on Nth Day
Days of Month
20
21
22
23
24
25
26
27
28
29
30
0
5
10
15
20
25
30

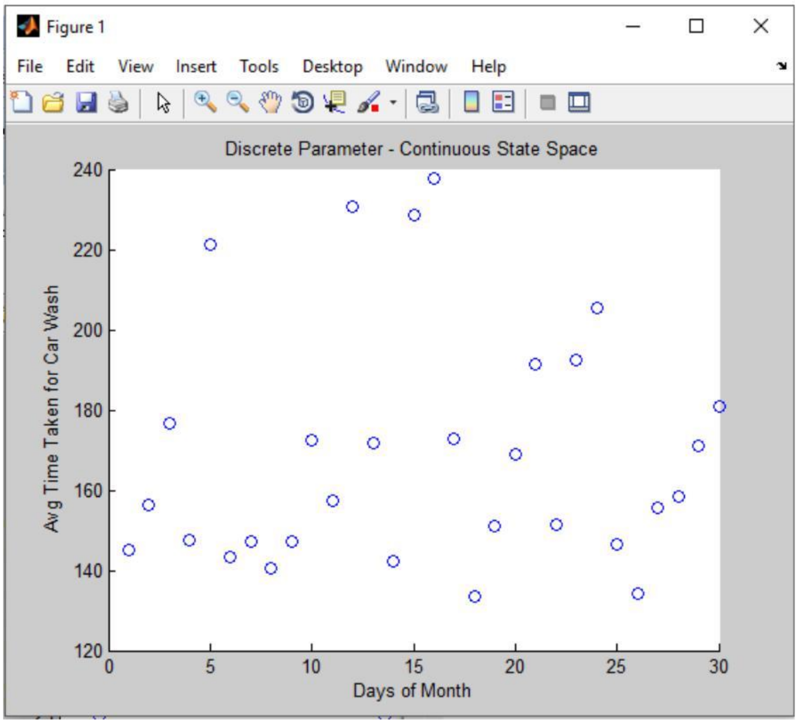

Figure 1
File Edit View Insert Tools Desktop Window Help
Discrete Parameter - Continuous State Space
Avg Time Taken for Car Wash
Days of Month
120
140
160
180
200
220
240
0
5
10
15
20
25
30

EXPERIMENT # 2

QUESTION: Simulate the following continuous parameter stochastic processes –

\>> Continuous State Space: Variation in temperature in a time period of 5 minutes given that temperature can vary between 22 and 26 C.
\>> Discrete State Space: No. of times temperature changed within 5 minutes and when it changed given temperature can change a maximum of 10 times.

CODE:

1) Continuous State Space:

```
x = [0.01:0.05:5];
y = 22 + 4.*rand(100,1);
p = plot(x,y);
xlabel('Time in minutes');
ylabel('Temp Change');
title('Continuous Parameter - Continuous State Space');
```

2) Discrete State Space:

```
x = [0.01:0.1:5];
y = randi([0 10],50,1);
p = stairs(x,y);
xlabel('Time in minutes');
ylabel('No. of times Temp Changed');
title('Continuous Parameter - Discrete State Space');
```

OUTPUT:

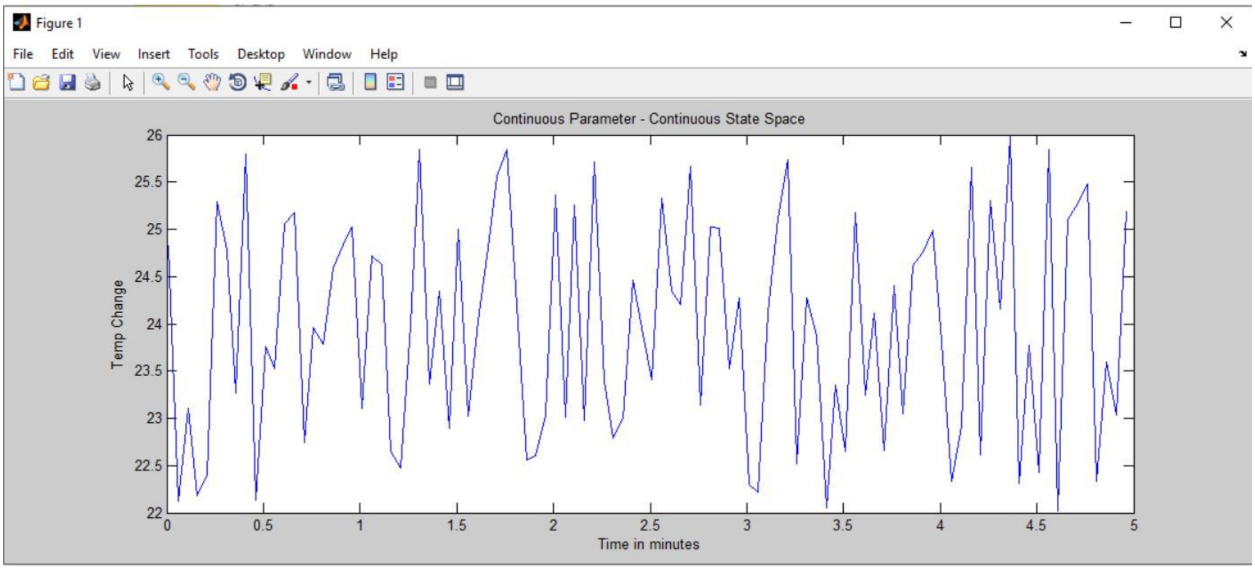
Figure 1
File Edit View Insert Tools Desktop Window Help
Continuous Parameter - Continuous State Space
Temp Change
26
25.5
25
24.5
24
23.5
23
22.5
22
0 0.5 1 1.5 2 2.5 3 3.5 4 4.5 5
Time in minutes

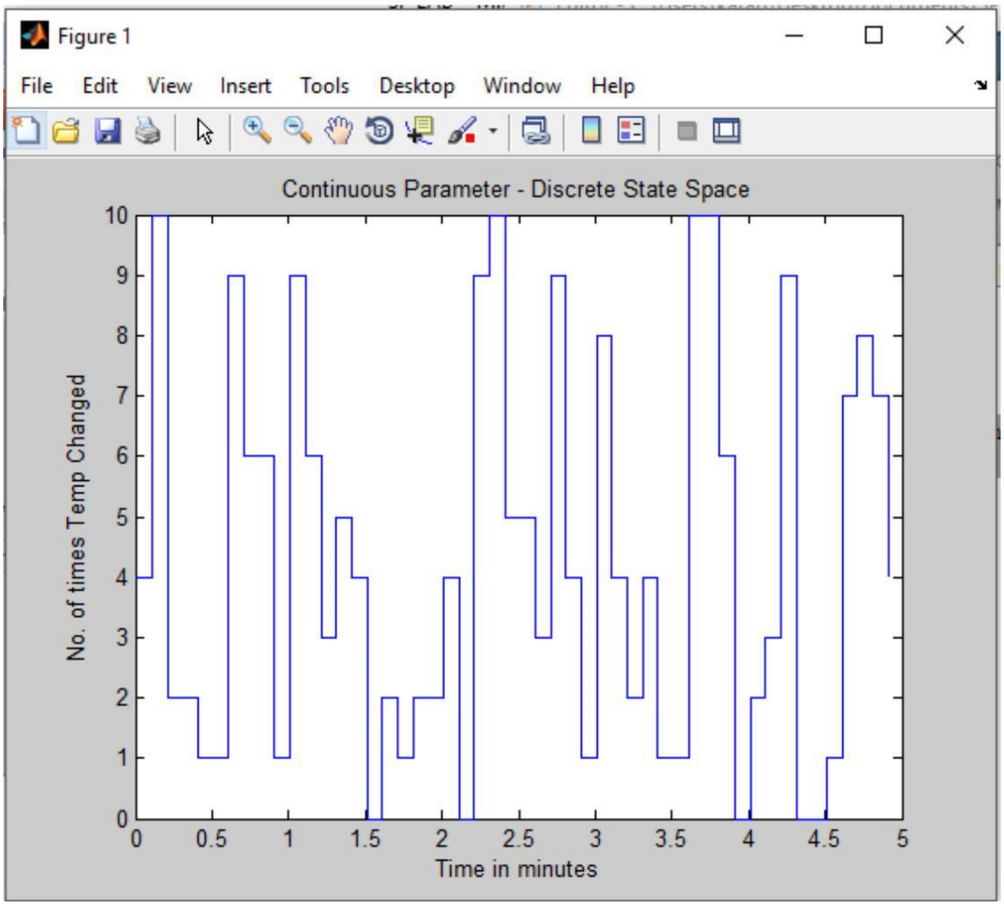
Figure 1
File Edit View Insert Tools Desktop Window Help
Continuous Parameter - Discrete State Space
No. of times Temp Changed
10
9
8
7
6
5
4
3
2
1
0
0 0.5 1 1.5 2 2.5 3 3.5 4 4.5 5
Time in minutes

EXPERIMENT # 3

QUESTION: It has been observed that a fuse designed by a company follows bernoulli distribution when being manufactured. If the company sells fuses in a box of 20 and gives guarantee that box will be replaced if no. of defective fuses is greater than 2. Find expected no. of replacements in a lot of 1000 boxes. Given –

1) Pr[fuse is defective] = 0.01 = p
2) Pr[n_{th} fuse is defective] = 0.01n = p_n

CODE:

Function file :

1) Homogenous Bernoulli

```
function [prob] = bernoulli(x,n)
p = 0.01;
>>   =
1-p;
prob =
0;
    for i=x+1:n
        prob = prob +
(factorial(n)/(factorial(n-
i)*factorial(i)))*(p^i)*(q^(n-i));
    end
end
```

2) Non Homogenous Bernoulli

```
function [prob] = nbernoulli(x,n)
prob = 0;
p=1;
q=1;
    for i=x+1:n
        for j=1:i
            p = p*(0.01*j);
        end
        for j=i+1:n
            q = q*(1-(0.01*j));
        end
prob = prob + (factorial(n)/(factorial(n-
        i)*factorial(i)))*p*q; p=1;
        q=1;
```

```
    end
end
```

OUTPUT:

Command Window:

```
>> ans =

1000*bernoulli(2,20) ans =

    1.0036

>> ans = 1000*nbernoulli(2,20)

ans =

    0.9193
```

EXPERIMENT # 4

QUESTION: Assuming that a circuit has an IC whose time to failure is exponentially distributed with expected lifetime of 3 months. If there are 10 spare IC's and time from failure to replacement is negligible. What is the probability that circuit is operational for a year? For non homogenous case consider expected lifetime to be 3/n. and for renewal consider time to failure is uniformly distributed with b = 3 and a = 0.

CODE:

Function file

1) Homogenous Poisson

```
function [p] = poisson(param,n)
    p = 0;
    for i=0:n
            p = p+((exp(-param)*((param)^i))/factorial(i));
    end
end
```

2) Non Homogenous Poisson

```
function [p] = npoisson(param,n)
    p = 0;
    for i=0:n
            p = p+((exp(-param/i)*((param/i)^i))/factorial(i));
    end
end
```

3) Uniform Renewal Process

```
function[prob] = uniformrenewal(n,t,a,b)
    c = 1/(b-a);prob =
    0; for i=0:n
        prob = prob +  c^i*t^i*((i+1 - c*t)/(factorial(i+1)));
    end
end
```

OUTPUT:

Command Window:

>> ans = poisson(4,10)

ans =

```
    0.9972


>> ans =

npoisson(4,10) ans =

    0.4647

>> ans = uniformrenewal(10,12,0,3)

ans =

    0.8949
```

EXPERIMENT # 5

QUESTION: A simple unrestricted random walk with

3) P = 0.4, q = 0.6
4) P = 0.4, q = 0.5

Find the probability that after 100 steps at n = 100 the particle lies between -15 and 20 in both cases. Find the probability that particle is away from 25 i.e. position at n = 100 >= 25.

CODE:

Function file:

```
function [p] = rndwalk(p,q,j,k,n)
if p+q < 1
    r = 1- p - q;
    c = 0.5;
else
    c = 1;
end
    mean = p - q;
    sd = sqrt(p + q - (p-q)^2);
    p = normcdf((k + c - n*mean)/(sd*sqrt(n))) - normcdf((j - c
- n*mean)/(sd*sqrt(n)));
end
```

OUTPUT:

Command Window:

>> ans = rndwalk(0.4,0.6,-

15,20,100) ans =

0.3415

>> ans = rndwalk(0.4,0.5,-

15,20,100) ans =

0.7194

```
>> ans =

rndwalk(0.4,0.6,25,inf,100) ans =

   3.5490e-06
>> ans =

rndwalk(0.4,0.5,25,inf,100) ans =

   1.2760e-04
```

EXPERIMENT # 6

QUESTION: Consider a random walk with two absorbing barriers and 1 absorbing barrier. Take any values of p,q such that

1) $p < q$ with 1 absorbing barrier
2) $p \geq q$ with 1 absorbing barrier
3) $p \neq q$ with 2 absorbing barriers
4) $p = q$ with 2 absorbing barriers

CODE:

Function file :

```
function [ans] = absbarrierrndwalk(p,q,a,b)
    if b == inf
        if p < q
            ans = (p/q)^a;
        else
            ans = 1;
        end
    else
        if p == q
            ans = b/(a+b);
        else
            ans = p^a*((p^b - q^b)/(p^(a+b) - q^(a+b)));
        end
    end

end
```

OUTPUT:

Command Window:

```
>> ans = absbarrierrndwalk(0.4,0.5,3,inf)

ans =

   0.5120
```

```
>> ans = absbarrierrndwalk(0.5,0.3,3,inf)

ans =
```

```
     1

>> ans =

absbarrierrndwalk(0.4,0.5,3,4) ans =

    0.3825

>> ans =

absbarrierrndwalk(0.5,0.5,3,4) ans =

    0.5714
```

EXPERIMENT # 7

QUESTION: Find the n step transitional probability matrix for following random walks using Markov chains-

1) two absorbing barriers
2) one absorbing barrier and one reflecting barrier
3) one reflecting barrier and one absorbing barrier
4) two reflecting barriers

CODE:

Function File:

```
function [answer] = markovchain(p,q,r,n,c)
    tpm1 = zeros(n,n);
    for i=1:n
        for j=1:n
            if i-j == -1 && i~=1 && i~=n
                tpm1(i,j) = p;
            elseif i-j == 0 && i~=1 && i~=n
                tpm1(i,j) = r;
            elseif i-j == 1 && i~=1 && i~=n
                tpm1(i,j) = q;
            end
        end
    end
    switch c
        case 1
            tpm1(1,1) = 1;
            tpm1(n,n) = 1;
        case 2
            tpm1(1,1) = 1;
            tpm1(n,n) = 1-q;
            tpm1(n,n-1) = q;
        case 3
            tpm1(1,1) = 1-p;
            tpm1(1,2) = p;
            tpm1(n,n) = 1-q;
            tpm1(n,n-1) = q;
        case 4
            tpm1(1,1) = 1-p;
            tpm1(1,2) = p;
            tpm1(n,n) = 1;
    end
    answer = tpm1;
```

```
    for i=1:n
        answer = answer*tpm1;
    end
end
```

OUTPUT:

Command Window:

```
>> ans = markovchain(0.5,0.4,0.1,5,1)

ans =

    1.0000         0         0         0         0
    0.6020    0.0443    0.0520    0.0554    0.2463
    0.3191    0.0416    0.0886    0.0520    0.4986
    0.1261    0.0354    0.0416    0.0443    0.7526
         0         0         0         0    1.0000

>> ans = markovchain(0.5,0.4,0.1,5,2)

ans =

    1.0000         0         0         0         0
    0.6020    0.0523    0.0740    0.1304    0.1413
    0.3255    0.0592    0.1566    0.1870    0.2716
    0.1517    0.0834    0.1496    0.2696    0.3457
    0.0794    0.0723    0.1738    0.2765    0.3980

>> >> ans = markovchain(0.5,0.4,0.1,5,3)

ans =

    0.2474    0.2439    0.2249    0.1525    0.1313
    0.1951    0.2322    0.1860    0.2079    0.1788
    0.1439    0.1488    0.2186    0.2070    0.2816
    0.0781    0.1330    0.1656    0.2776    0.3457
    0.0538    0.0915    0.1802    0.2765    0.3980
```

```
>> ans = markovchain(0.5,0.4,0.1,5,4)
>> ans =

   0.2474   0.2439   0.2149   0.1150   0.1787

   0.1951   0.2242   0.1640   0.1229   0.2938

   0.1375   0.1312   0.1506   0.0720   0.5086

   0.0589   0.0786   0.0576   0.0523   0.7526

        0        0        0        0   1.0000
```

EXPERIMENT # 8

QUESTION: Given one step transitional probability matrix find the long term or steady state or long term probabilities of visiting each city in the long run.

PROBLEM: A traveler visits 4 cities A, B, C, D if he visits A then he is equally likely to visit B, C but not D. If he visits B then he is twice as likely to go to C than A or D.If he visits C then he is 2 times as likely to go to A than B but he will not go to D. If he visits D then he is equally likely to go to A, B, C.

CODE:

Function File:

```
function [ answer ] = ergodic(tpm,n)
    tpm = -tpm;
    for i=1:n
        for j=1:n
            if i == j
                tpm(i,j) = 1;
            end
        end
    end
    A = [tpm';ones(1,n)];
    B = zeros(n+1,1);
    B(n+1) = 1;
    answer = linsolve(A,B);
end
```

OUTPUT:

Command Window:

```
>> t = [0 1/2 1/2 0;1/4 0 1/2 1/4;2/3 1/3 0 0;1/3 1/3 1/3 0]
t =

        0    0.5000    0.5000         0
   0.2500         0    0.5000    0.2500
   0.6667    0.3333         0         0
   0.3333    0.3333    0.3333         0
```

```
>> ans =

ergodic(t,4)

ans =

    0.3133
    0.2892
0.3253
0.0723
```