**Que 1**

**Aim**: To obtain realization on following processes up to the time t=15 and give the graphical representation of realization obtained. To estimate the in each case by simulation.

**Poisson Process with rate parameter 0.5**

Solution:

#realizations

t=0

l={}

g={}

i=1

event=0

while(t<15){

x=rexp(1,0.5)

t=t+x

l[i]=t

g[i]=i

i=i+1

event=event+1

# print(c(x,t,event))

}

real=data.frame(time=l[-length(l)],event=g[-length(g)]);real

output:

time event

1 0.7100508 1

2 3.1125049 2

3 3.3655420 3

4 6.7778493 4

5 7.2155643 5

6 7.7089502 6

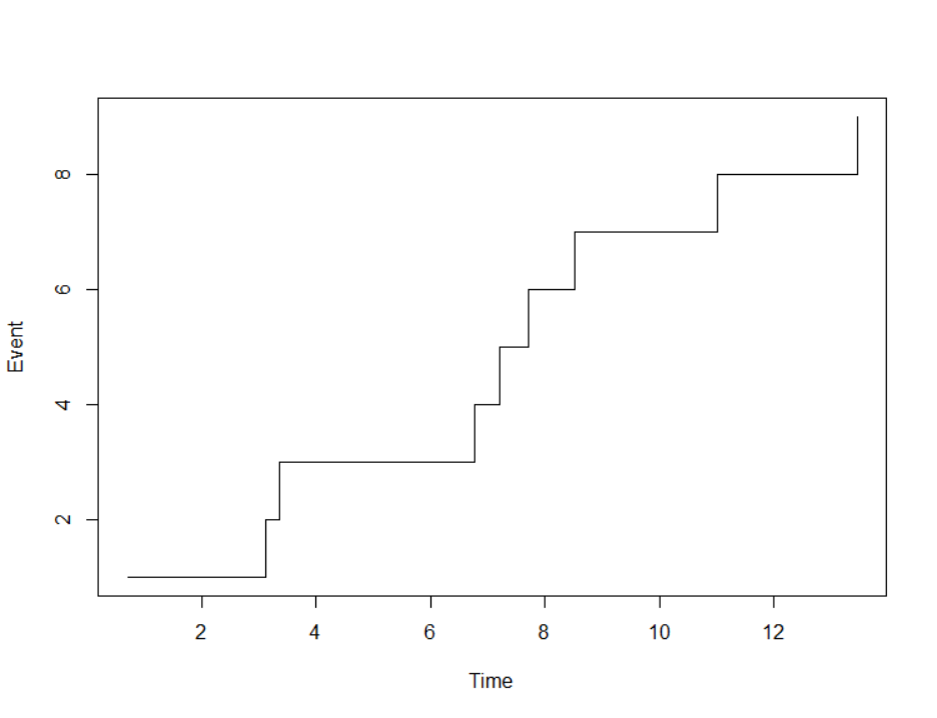
7 8.5154127 7

8 11.0125903 8

9 13.4523408 9

plot(l[-length(l)],g[-length(g)],xlab="Time",ylab="Event",type="s")

output:



real\_poi=function(n){

y={}

N=1000

for(i in 1:N){

t=0

event=0

while(t<n){

x=rexp(1,0.5)

t=t+x

event=event+1

#print(c(x,t,event))

}

y[i]=event-1

}

return(y)

}

# p(x(15)=5)

g=real\_poi(15)

mean(g==5)

output: 0.131

# p(x(9)<4)

g=real\_poi(9)

mean(g<4)

output: 0.326

# p(x(5)=3,x(9)=4)

y={}

l={}

N=1000

for(i in 1:N){

t=0

event=0

event1=0

while(t<9){

x=rexp(1,0.5)

t=t+x

event=event+1

if(t<5){event1=event1+1}

#print(c(x,t,event))

}

l[i]=event1

y[i]=event-1

}

mean(l==3 & y==4)

output: 0.063

#E(x(20))

g=real\_poi(20)

mean(g)

output: 10.124

**Renewal Process with Gamma(2,2) distributed inter arrival times**

#Q1\_renewal\_process

t=0

l={}

g={}

i=1

event=0

while(t<15){

x=rgamma(1,2,2)

t=t+x

l[i]=t

g[i]=i

i=i+1

event=event+1

# print(c(x,t,event))

}

real=data.frame(time=l[-length(l)],event=g[-length(g)])

real

output:

time event

1 0.2580931 1

2 0.4772589 2

3 1.4773894 3

4 1.7659967 4

5 2.8504136 5

6 4.3831894 6

7 6.2609942 7

8 6.5155592 8

9 7.0372390 9

10 8.2149527 10

11 8.9573423 11

12 10.0331091 12

13 10.2357307 13

14 10.6984725 14

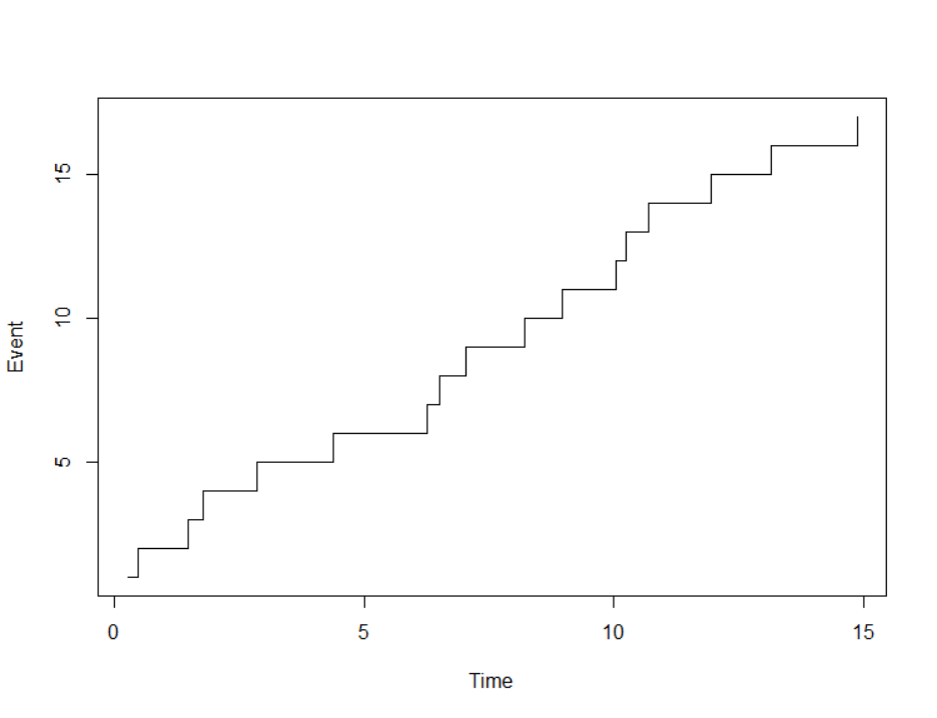
15 11.9340945 15

16 13.1374251 16

17 14.8724934 17

plot(l[-length(l)],g[-length(g)],xlab="Time",ylab="Event",type="s")

output:



real\_renewal=function(n){

y={}

N=1000

for(i in 1:N){

t=0

event=0

while(t<n){

x=rgamma(1,2,2)

t=t+x

event=event+1

#print(c(x,t,event))

}

y[i]=event-1

}

return(y)

}

#p(x(15)=5)

g=real\_renewal(15)

mean(g==5)

output: 0

#p(x(9)<4)

g=real\_renewal(9)

mean(g<4)

output: 0.003

#p(x(5)=3,x(9)=4)

y={}

l={}

N=1000

for(i in 1:N){

t=0

event=0

event1=0

while(t<9){

x=rgamma(1,2,2)

t=t+x

event=event+1

if(t<5){event1=event1+1}

#print(c(x,t,event))

}

l[i]=event1

y[i]=event-1

}

mean(l==3 & y==4)

output: 0.006

#E(x(20))

g=real\_renewal(20)

mean(y)

output: 8.847

**Pure Birth Process: with X(0)=1 and birth rate**

λ1 = 0.4, λ2 = 0.5, λ3=0.7, λ4 = 0.5, λk = 0.8, k ≥ 5

lambda=c(0.4,0.5,0.7,0.5,rep(0.8,15))

t=0

l={}

g={}

l[1]=0

g[1]=1

i=j=1

event=1

while(t<15){

x=rexp(1,lambda[i])

t=t+x

l[i+1]=t

event=event+1

g[i+1]=event

i=i+1

#print(c(x,t,event))

}

real3=data.frame(time=l[-length(l)],event=g[-length(g)])

real3

output:

time event

1 0.0000000 1

2 0.1853627 2

3 1.1517246 3

4 1.9318024 4

5 4.4922862 5

6 6.0971589 6

7 8.6990431 7

8 10.2252779 8

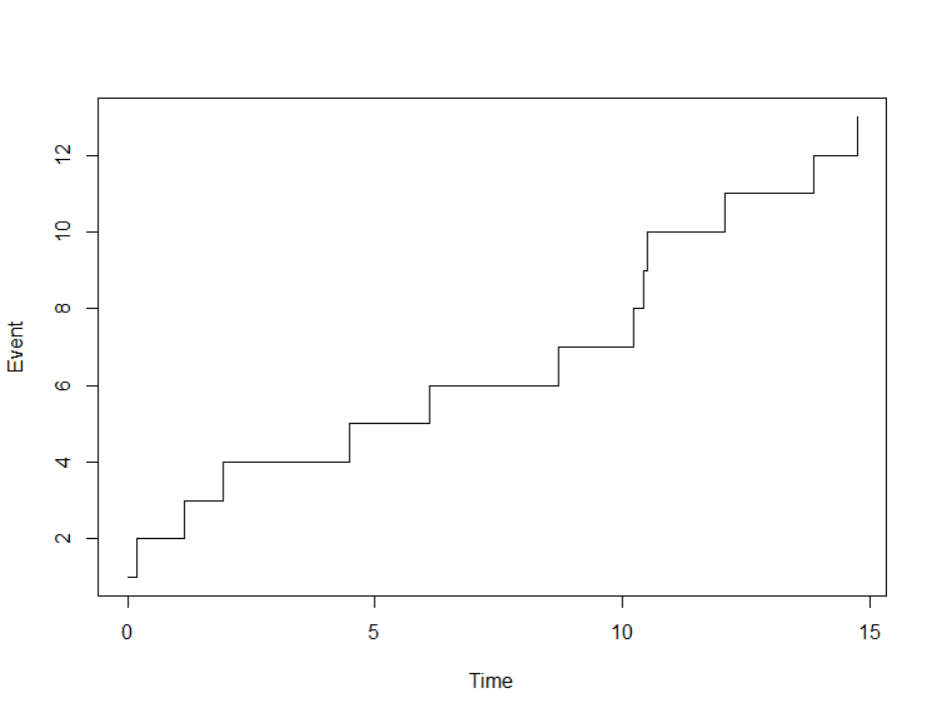
9 10.4380885 9

10 10.5123147 10

11 12.0683264 11

12 13.8635241 12

13 14.7519565 13

plot(g[-length(g)],l[-length(l)],xlab="Event",ylab="Time",type="s") 

real\_pb=function(n){

y={}

N=10000

y[1]=1

for(i in 2:N){

t=0

event=1

lambda=c(0.4,0.5,0.7,0.5,rep(0.8,30))

j=1

while(t<n){

x=rexp(1,lambda[j])

t=t+x

event=event+1

#print(c(x,t,event))

j=j+1

}

y[i]=event-1

}

return(y)

}

#p(x(15)=5)

g=real\_pb(15)

mean(g==5)

output: 0.0381

#p(x(9)<4)

g=real\_pb(9)

mean(g<4)

output: 0.1688

#p(x(5)=3,x(9)=4)

y={}

l={}

N=10000

l[1]=1

y[1]=1

for(i in 2:N){

t=0

event=event1=1

lambda=c(0.4,0.5,0.7,0.5,rep(0.8,30))

j=1

while(t<9){

x=rexp(1,lambda[j])

t=t+x

event=event+1

if(t<5){event1=event1+1}#print(c(x,t,event))

j=j+1

}

l[i]=event1

y[i]=event-1

}

mean(l==3 & y==4)

output: 0.0488

#E(x(20))

g=real\_pb(20)

mean(g)

output: 14.6283

**Yule’s Process X(0)=2 and λ=0.5**

#Yule\_process

lambda=0.5

t=0

l={}

g={}

l[1]=0

g[1]=2

i=1

event=2

while(t<15){

x=rexp(1,lambda\*i)

t=t+x

l[i+1]=t

event=event+1

g[i+1]=event

#print(c(x,t,event))

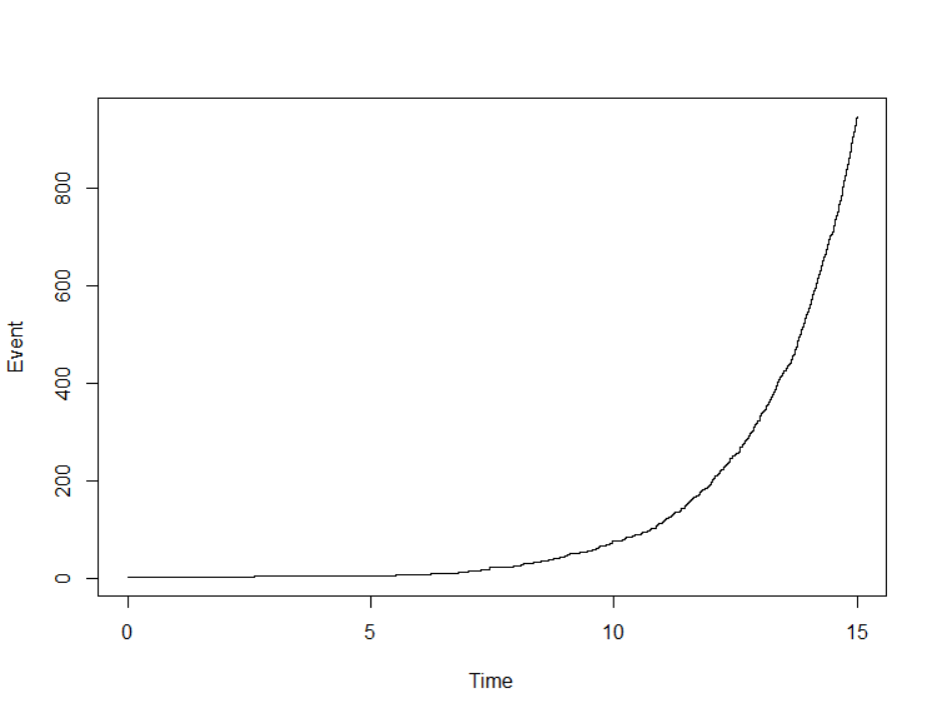
i=i+1

}

real3=data.frame(time=l[-length(l)],event=g[-length(g)])

plot(l[-length(l)],g[-length(g)],xlab="Time",ylab="Event",type="s")

output:



real\_YP=function(n){

y={}

N=10000

y[1]=1

for(i in 2:N){

t=0

event=2

lambda=0.5

j=1

while(t<n){

x=rexp(1,lambda\*j)

t=t+x

event=event+1

#print(c(x,t,event))

j=j+1

}

y[i]=event-1

}

return(y)

}

#p(x(15)=5)

g=real\_YP(15)

mean(y==5)

output: 0.1181

#p(x(9)<4)

g=real\_YP(9)

mean(g<4)

output: 0.0232

#p(x(5)=3,x(9)=4)

y={}

l={}

N=10000

l[1]=1

y[1]=1

for(i in 2:N){

t=0

event=event1=1

lambda=0.5

j=1

while(t<9){

x=rexp(1,lambda\*j)

t=t+x

event=event+1

if(t<5){event1=event1+1}#print(c(x,t,event))

j=j+1

}

l[i]=event1

y[i]=event-1

}

mean(l==3 & y==4)

output: 5e-04

#E(x(20))

g=real\_YP(20)

mean(g)

output: 22102.7

**Birth and Death process**

#Births rates: λ0 =1 λ1= 0.5, λ2= 0.6, λ3= 0.9, λk=1.4, k ≥ 4.

#Death rates: μ1= 0, μ2= 0.5, μ3= 0.7, μk = 1.5 k ≥ 4.

rm(list=ls())

lam=c(0.5,0.6,0.9,rep(1.4,100))

mu=c(0,0.5,0.7,rep(1.5,100))

t=0

event=0

i=1

y={}

g={}

while(t<15){

if(event==0){

b=rexp(1,lam[1])

t=t+b

event=event+1

}else{

b=rexp(1,lam[i])

d=rexp(1,mu[i])

t=t+min(b,d)

if(min(b,d)==b){

event=event+1

}else{

event=event-1

}

}

y[i]=event

g[i]=t

i=i+1

}

realbd=data.frame(time=g[-length(g)],event=y[-length(y)])

realbd

output:

time event

1 1.990207 1

2 2.809815 2

3 2.987382 3

4 3.281960 2

5 3.293179 3

6 4.065120 4

7 4.272700 3

8 4.306452 2

9 5.575895 1

10 6.412266 2

11 6.418639 1

12 7.556691 0

13 7.767147 1

14 7.989814 2

15 8.690017 3

16 8.769813 2

17 8.926985 3

18 9.044775 2

19 9.217865 3

20 9.344340 4

21 9.597193 3

22 9.632960 2

23 9.686841 3

24 9.976070 2

25 10.606404 3

26 10.933652 4

27 11.595927 5

28 11.898204 4

29 11.915733 3

30 12.283353 2

31 12.402653 1

32 12.800872 0

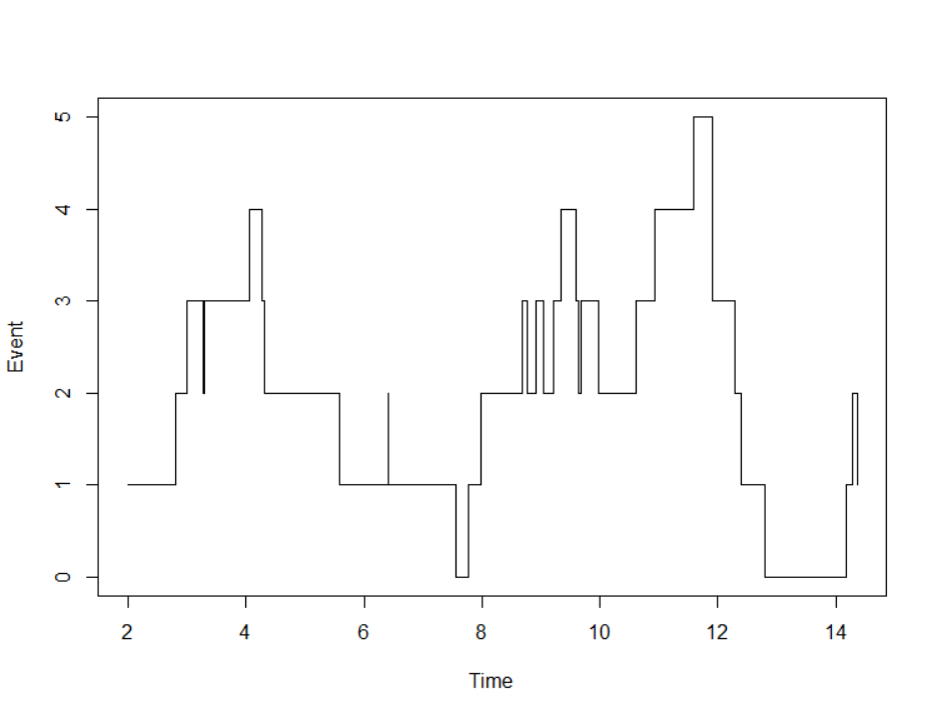
33 14.180821 1

34 14.279950 2

35 14.365978 1

plot(g[-length(g)],y[-length(y)],type="s",xlab="Time",ylab="Event")

output:



real\_bd=function(n){

z={}

N=1000

for(j in 1:N){

lam=c(0.5,0.6,0.9,rep(1.4,100))

mu=c(0,0.5,0.7,rep(1.5,100))

t=0

event=0

i=1

y={}

g={}

while(t<n){

if(event==0){

b=rexp(1,lam[1])

t=t+b

event=event+1

}else{

b=rexp(1,lam[i])

d=rexp(1,mu[i])

t=t+min(b,d)

if(min(b,d)==b){

event=event+1

}else{

event=event-1

}

}

y[i]=event

i=i+1

}

z[j]=y[length(y)-1]

}

return(z)

}

#p(x(15)=5)

g=real\_bd(15)

mean(y==5)

output: 0

#p(x(9)<4)

g=real\_bd(9)

mean(g<4)

output: 0.676

#p(x(5)=3,x(9)=4)

y={}

l={}

N=10000

lam=c(0.5,0.6,0.9,rep(1.4,100))

mu=c(0,0.5,0.7,rep(1.5,100))

for(j in 1:N){

t=0

i=1

event=0

event1=0

while(t<9){

if(event==0){

b=rexp(1,lam[1])

t=t+b

event=event+1

}else{

b=rexp(1,lam[i])

d=rexp(1,mu[i])

t=t+min(b,d)

if(min(b,d)==b){

event=event+1

}else{

event=event-1

}

if(t<5)

{event1=event}

}

}

y[i]=event

l[i]=event1

i=i+1

}

mean(l==3 & y==4)

output: 0

#E(x(20))

g=real\_bd(20)

mean(g)

output: 3.422

Q.2

**Aim** :To estimate the probability that machine will be working at time t = 10 given that it is in working condition at time 0.

**Solution**:

lam=1

mu=0.5

t=0

z={}

N=1000

for(j in 1:N){

lam=1

mu=0.5

t=0

event=1 #working

i=1

y={}

g={}

while(t<10){

b=rexp(1,lam)

d=rexp(1,mu)

t=t+min(b,d)

if(min(b,d)==b){

event=1

}else{

event=0

}

y[i]=event

i=i+1

}

z[j]=y[length(y)-1]

}

mean(z==1)

output: 0.674

**Q.4**

**Aim**: To obtain realization on Branching process. To estimate the mean size of the 5th generation and also estimate the P(X3=0) and P(X10=10).

**Solution**:

real\_brach=function(n){

y={}

y[1]=1

x=c(0,1,2,3,4)

p=c(0.1,0.3,0.3,0.2,0.1)

for(i in 2:n){

z=sample(x,y[i-1],replace=TRUE,prob=p)

y[i]=sum(z)

}

y

}

real\_brach(11)

output: 1 1 4 2 6 15 26 58 122 255 486

#E(x5)

h={}

for(j in 1:1000){

g=real\_brach(6)

h[j]=g[6]

}

mean(h)

output: 24.134

#P(X3=0)

h={}

for(j in 1:1000){

g=real\_brach(4)

h[j]=g[4]

}

mean(h==0)

output: 0.14

#P(X10=10)

h={}

for(j in 1:1000){

g=real\_brach(11)

h[j]=g[11]

}

mean(h==0)

output: 0.147

Q.5

**Aim**: To obtain the probability distribution of X5,.average size of 10th generation, probability that size of 7th generation is less than or equal to 10.

**solution** :

#p0 =0.4, p1 =0.4, p2 =0.2

real\_brach1=function(n){

y={}

y[1]=1

x=c(0,1,2)

p=c(0.4,0.4,0.2)

for(i in 2:n){

z=sample(x,y[i-1],replace=TRUE,prob=p)

y[i]=sum(z)

}

return(y)

}

real\_brach1(6)

output: 1 0 0 0 0 0

#probability distribution of 5th generation

h={}

for(j in 1:1000){

g=real\_brach1(6)

h[j]=g[6]

}

f=0:10

p={}

for(k in 1:11){

p[k]=mean(h==f[k])

}

dist=data.frame(x5=f,p=p)

dist

output:

x5 p

1 0 0.824

2 1 0.081

3 2 0.056

4 3 0.023

5 4 0.010

6 5 0.003

7 6 0.003

8 7 0.000

9 8 0.000

10 9 0.000

11 10 0.000

#average size of 10th generation

h={}

for(j in 1:1000){

g=real\_brach1(11)

h[j]=g[11]

}

mean(h)

output: 0.124

#p(y7<=10)

h={}

for(j in 1:1000){

g=real\_brach1(8)

h[j]=g[8]

}

mean(h<=10)

output:1