**Q1) DESCRIPTION**

A non-homogeneous Poisson process is similar to an ordinary Poisson process, except that the average rate of arrivals is allowed to vary with time. Many applications that generate random points in time are modeled more faithfully with such non-homogeneous processes. In our case, we have a Single Server Queuing system, where customers arrive according to a non-homogeneous Poisson process with intensity function lambda(t), t>=0. Server handles customer if free else joins queue. When server completes serving a customer, it begins handling longest waiting customer( First come first served) or remains free if queue is empty. The amount of time to service a customer follows a particular probability distribution. Fixed time T after which no new arrivals occur. But server continues to handle all customers in queue. The quantities of interest to us here is

* The average time customer spends in queue
* The average time past T that last customer departs.

**CODE DESCRIPTION**

In the code we use an algorithm for non homogeneous Poisson process which is roughly

Suppose that λ(t) is the bounded intensity function (arrival function) for a non-homogenous Poisson process. To generate a sample Ts that is the time of the first arrival after time s:

Algorithm: Choose λ so that λ(t)<λ for all t. Given λ(t), t>0, and λ:

Let t=s.

Generate U1∼U[0,1].

Let t=t−1/λ⋅logU1.

Generate U2∼U[0,1].

If U2≤(λ(t)/λ) set Ts=t and stop.

Goto step 2.

For the calculation of lambda\_t there are 2 cases. In the first case it linearly increases from 4 to 19 hours for the first 5 hours, which is given by the expression 3\*t+4. In the second case it linearly decreases from 19 hours to 4 hours in the next 5 hours. It is given by the expression -3\*t-4.

The service‐time distribution is exponential with rate 25 jobs per hour. The function R = exprnd(MU) returns an array of random numbers chosen from the exponential distribution with mean parameter MU. Here MU=25. The waiting time is uniformly distributed on (0,0.3). The variable ‘idle’ in the program represents this. So for 100 hours we increment the break time if there are no jobs waiting in the queue. We get the answer to be 11.2251 which means the server is idle for that many hours in 100 hours.

**Code**

N=100; %number of hours%

t=0;

x=0;

j=1;

lambda=20;

break\_time=0;

%Algorithm for non-homogeneous Poisson process where lambda\_t<lambda

while t<100

U1=rand();

t=t-(log(U1)/lambda);

if(mod(t,10) < 5)

lambda\_t=3\*t+4; %for the first 5 hours the arrival rate linearly increases

ratio=lambda\_t/lambda;

elseif(mod(t,10)>5)

lambda\_t=-3\*t-4; %for the next 5 hours the arrival rate linearly decreases

ratio=lambda\_t/lambda;

end

U2=rand();

if(U2<=ratio)

Ts(j)=t; %the time of the first arrival after some time%

j=j+1;

end

end

%Break time calculation

for i=1:length(Ts)-1

s=exprnd(25); %service time distribution with a rate of 25 jobs per hour

while((s + Ts(i)) < Ts(i+1)) %if there are no jobs waiting

idle=0.3\*rand(); %uniformly distributed break

break\_time=break\_time+idle; %total break time for 100 hours

s=s+idle;

end

end

disp(break\_time)

**OUTPUT**

The break time is 11.2251 hours.