Ma5

by Sigbjørn Fjelland

Date: '2022-11-04'

Due to issues with knit in r, and RCall package, the code wil be in Julia. I shal comment out so that it is fully understandable

Libraries in use

```
using Dates    ,DataFrames    ,Plots    , PlutoUI    , Random    , LaTeXStrings    ,
Statistics    , LinearAlgebra    , StatsPlots    , Distributions
```

Problem 5.1 a)

```
n = 100
```

```
λ = 0.6
```

```
μ = 0.7
```

```
p = 0.46153846153846156
```

```
#Proportion Entering in to the system
p=λ/(λ+μ)
```

```
q = 0.5384615384615385
```

```
#Proportion being served
q=μ/(λ+μ)
```

```
jump (generic function with 2 methods)
```

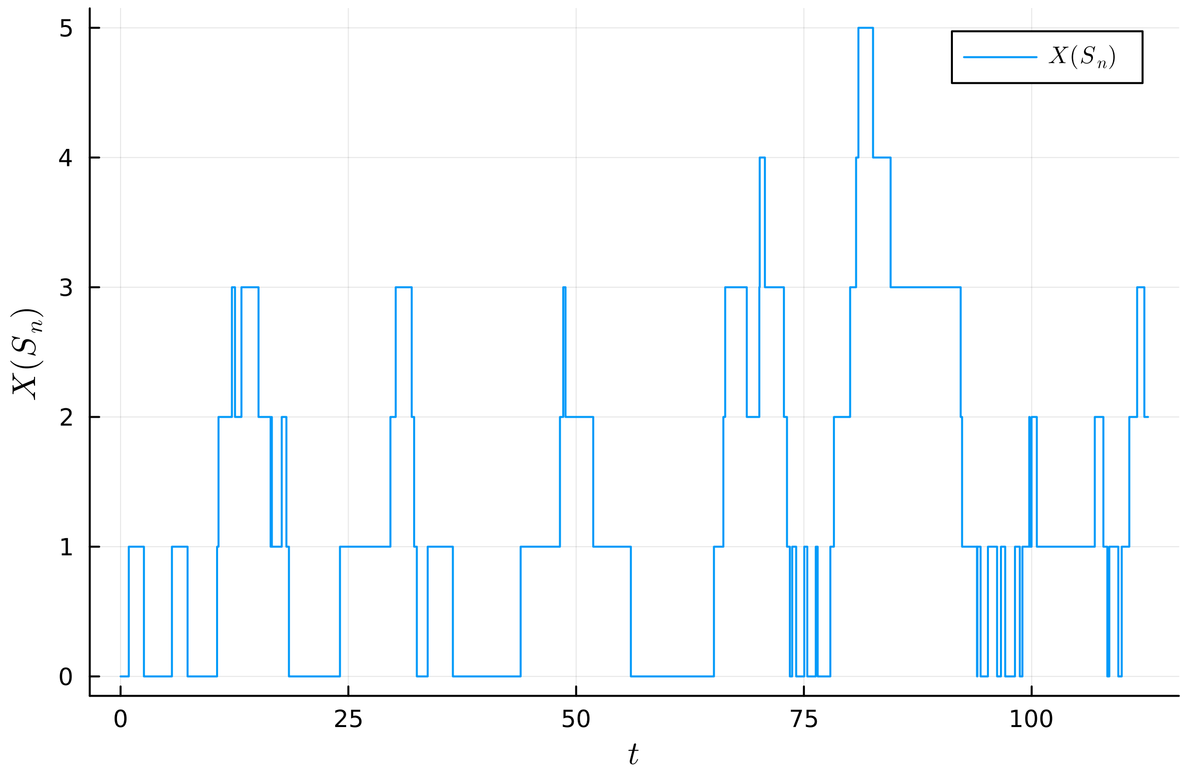
```
function jump(p,n, initialvalue=0)
        U = rand(n-1)
        queue = [initialvalue]                   #vector for storing each transition
        for i in 1:n-1
            if U[i] < p
                push!(queue,queue[i]+1)          #appends +1 to queue vector
            else
                push!(queue,max(0,queue[i]-1))   #appends -1 to queue vector with 0
                                                   as  floor
            end
        end
    return queue
end
```

```
jumptime (generic function with 1 method)
```

```
function jumptime(n)
    T=randexp(n-1)
    S=[0.00]
    for t in 1:(n-1)
        push!(S,S[t]+T[t]) #appends next cumulative sum (in lack of a cumsum function)
    end
    return S
end
```

```
[0.0, 0.629407, 0.902088, 2.56134, 2.79124, 5.63567, 7.35259, 10.5916, 10.7477, 12.2186, 1:
```

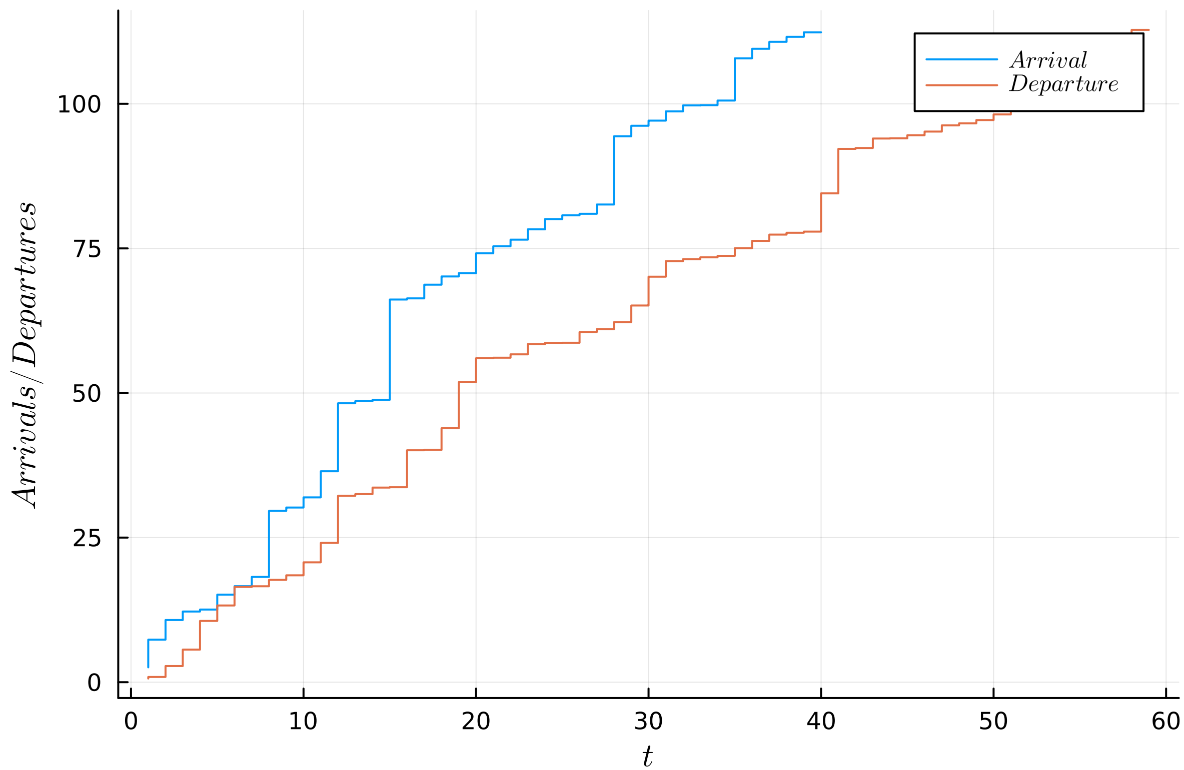
```julia
begin
Y_100=jump(p, 100)
S_100=jumptime(100)
end
```




```julia
plot(S_100, Y_100,
     titel = "simulation M/M/1 BD size n=100", label = L"X(S_n)",
     xlabel=L"t",  ylabel=L"X(S_n)", line = (:steppre))
```

Problem 5.1 b)

```
begin
arr = []
dep = []
for j in 2:n
    if Y_100[j]>Y_100[j-1]
        push!(arr, S_100[j])
    else
        push!(dep, S_100[j])
    end
end
end
```

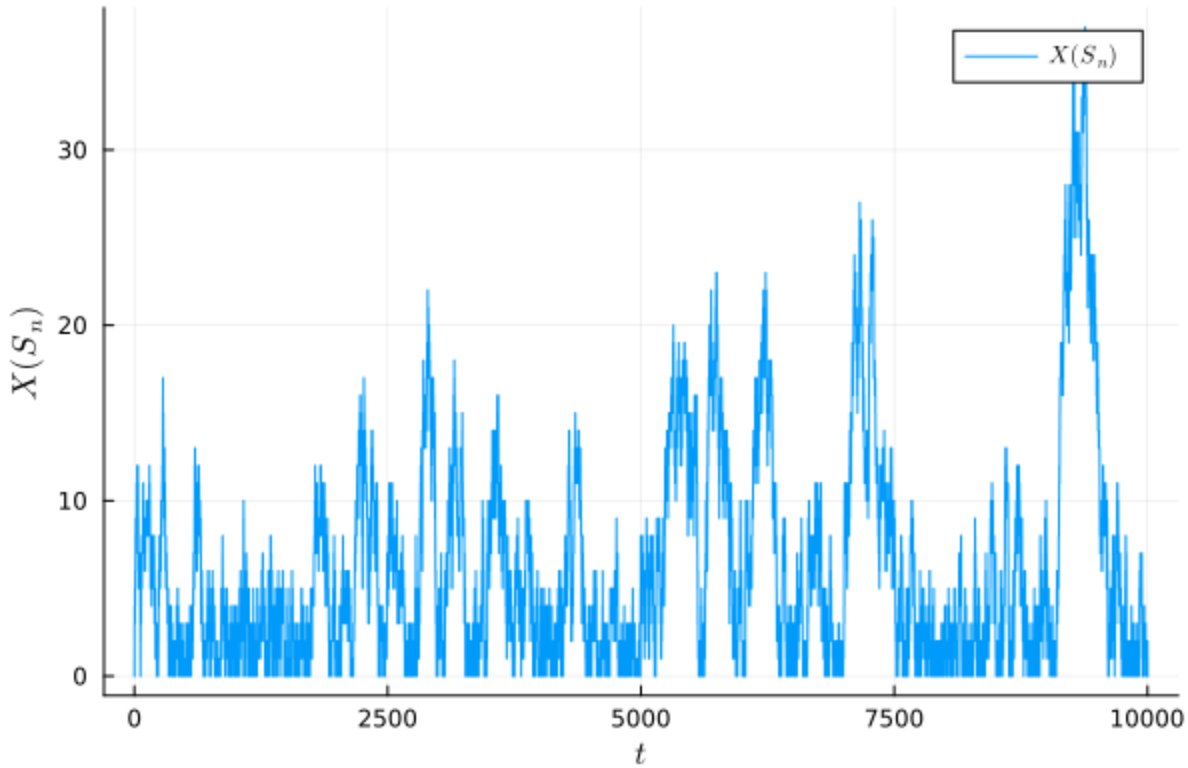



```
begin
plot(arr, label = L"Arrival",
    xlabel=L"t",  ylabel=L"Arrivals/Departures", linetype=:steppre)
plot!(dep, label = L"Departure", linetype=:steppre)
end
```

Problem 5.1 c)

```
[0.0, 0.124638, 0.206821, 0.753812, 0.820263, 0.979639, 2.64335, 2.99001, 4.38293, 5.1349,
```

```
begin
Y_10000=jump(p, 10000)
S_10000=jumptime(10000)
end
```

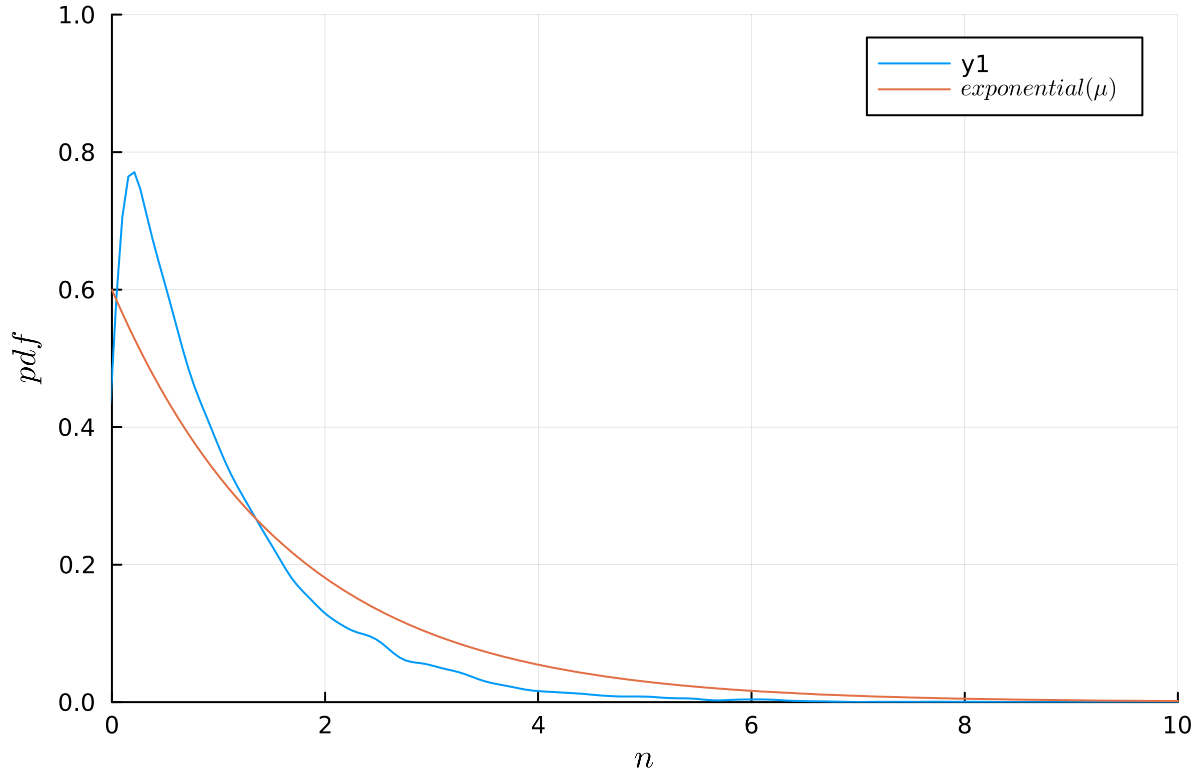



```
plot(S_10000, Y_10000,
    titel = "simulation M/M/1 BD size n=100", label = L"X(S_n)",
    xlabel=L"t",  ylabel=L"X(S_n)", linetype=:steppre)
```

holdtime (generic function with 1 method)

```
function holdtime(S)
T = []
for t in 2:length(S)
    push!(T, (S[t]-S[t-1]))
end
    return T
end

```




```julia
begin
    #density plot of holding times
    plot(density(holdtime(S_10000)),
        xlims= (0,10),
        ylims= (0,1),
        label = L"Holdtimes",
        titel = "Density of holdtimes",
        xlabel=L"n",
        ylabel=L"pdf")

        #Plot of distribution function
    plot!(Exponential((1/λ)),
        label = L"exponential(\mu)")
end
```

Problem 5.1 d)

```
π_0 = 0.1428571428571428
```

```
π_0 = 1-(λ/μ) #Stationary distribution
```

From the Embeded Markov chain we sort out the visits to state zero

```
S₀ =
 [1, 3, 4, 72, 73, 74, 240, 241, 243, 245, 246, 252, 356, 357, 358, 382, 383, 403, 407, 409,
```

```
S₀ = findall(Y_10000 .== 0) #argument of vector S₀=0 (simlar to which() in R)
```

From this we have to do the cumbersome way of extracting the holding time. This is due to som bad decition of keeping simulation of waiting time within the jumptime function..

```
begin
Times=holdtime(S_10000)
v = 0                          #Total time chain is in State zero
for i in S₀[1:length(S₀)-1]
      v = v + Times[i]
end
end
```

```
π̂₀ˣ = 0.13992245769115
```

```
π̂₀ˣ = (v+last(Times)) / last(S_10000)
```

```
π̂₀ʸ = 0.1355
```

```
π̂₀ʸ=length(S₀) / 10000
```

As we se the embedded chain focuses on jumps and increment while the chain it self is all about time and time spent in a state.

$$\hat{\pi}_0^X = \frac{\text{Time spent i 0}}{\text{Total time}}, \hat{\pi}_0^Y = \frac{\text{Jumps to 0}}{\text{Total number of jumps}}$$