Project 1 - Code

By: Sanne Jamila Razmara Olsen & Einride Brodahl Osland

Problem 1c

The following code consists of the function *simulate()* which simulates the Markov chain given by the probability matrix **P** for iter amount of days.

```r
simulate <- function(P, iter){
  #Finding the number of rows of P
  n <- nrow(P)
  #Initializing vector of the states, iter states
  states <- numeric(iter)

  #Assumption X_0 = 0
  states[1] <- 1

  #Simulating Markov chain
  for(t in 2:iter){
    #Probability vector to simulate next state (X_t+1)
    p <- P[states[t-1],]
    #Draw from multinomial distrubution to choose next state
    states[t] <- which(rmultinom(1,1,p) == 1)
  }
  return(states)
}
```

Constructing **P** by the values given in the project description we test our implementation of the simulation for iter = 7300 days.

```r
beta = 0.01
gamma = 0.1
alpha  = 0.005

P = matrix(c(1-beta,beta,0,0,1-gamma,gamma,alpha,0,1-alpha),nrow = 3, byrow = TRUE)

#State 0,1,2 transformed to state 1,2,3
n = 7300

states <- simulate(P,n)
```

We want to estimate the limiting probabilities by the function *limiting_probs()*.

```r
limiting_probs <- function(states){
  n = length(states)
  p1=0
  p2 = 0
  p3 = 0

  for(i in states){
    if(i == 1){
      p1 = p1 +1
    }
    else if(i == 2){
      p2 = p2 +1
    }
    else{
      p3 = p3 +1
    }
  }
  p1 = p1/n
  p2 = p2/n
  p3 = p3/n

  pi <- c(p1,p2,p3)
  return(pi)
}
```

The following code of the implementation *confidence_interval()* constructs a 95% confidence interval for the j´th limiting probability.

```
confidence_interval <- function(P,N,j){
  pi = numeric(N)

  for(i in 0:N){
    states = simulate(P,7300)
    states = states[3650:7300]

    pi[i] <- limiting_probs(states)[j]

  }
  n = N

  t = qt(0.95,df = n-1)

  variance = var(pi)
  mean = mean(pi)

  S = sqrt(n/(n-1)*variance)

  a = mean + t*S/sqrt(n)
  b = mean - t*S/sqrt(n)

  return(c(a,b))
}
```

Problem 2a

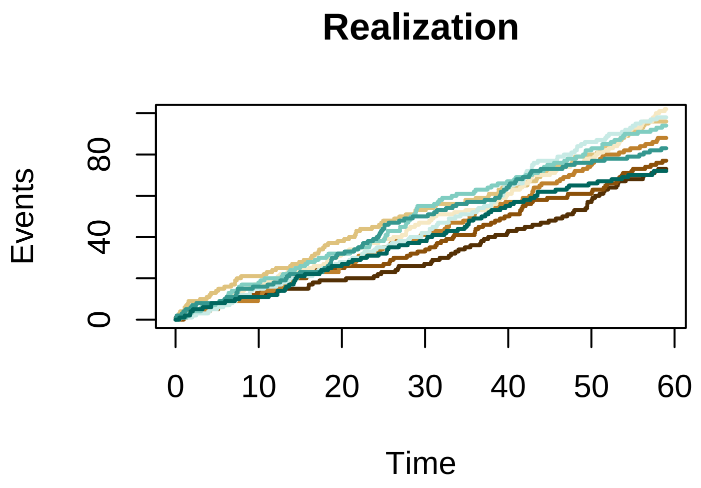
The function *Poisson_Process()* calculates n realizations of the Poisson Process.

```r
Poisson_Process <- function(lambda,n,t){
  #Creating empty vector of size n+1
  v<-numeric(n+1)
  v[1] <-0


  for (k in 1:n){
    x <- rpois(1,lambda*t)
    if(x > 100){
      v[k] = 1
    }
    else{
      v[k] = 0
    }
  }
  return(v)
}

t <- 59
n <- 1000
lambda <-1.5

v = Poisson_Process(lambda,n,t)
mean(v)
```

```
## [1] 0.1098901
```

The mean is a representation of the probability that there are more than 100 claims on day 59.

```r
#install.packages("RColorBrewer")
library(RColorBrewer)


plot_poisson <-function(t_value,lambda,N){
  #Creating colour palette to visualize simulations
  colour_palette = brewer.pal(n=10, name = "BrBG")
  #Generating plot to fill with simulation plots
  plot(NULL, NULL, xlim = c(0, t_value), ylim = c(0, 100), xlab = "Time", ylab = "Events", main = "Reali



  #Simulating N Poisson processes and plotting them by using lines() function
  for (n in 0:N) {
    #Picking from Poisson distribution
    X <- rpois(1, lambda * t_value)
    x = c(0:X, X)

    t <- runif(X, 0, t_value)
    t = c(0, sort(t), t_value)
    l = length(x)

    for (i in 1:(l - 1)){
      lines(t[i:(i + 1)], rep(x[i], 2), lwd = 2, col = colour_palette [n]) #Adding simulation to plot
    }
  }
}


#Initializing given values
t_value <- 59
lambda <- 1.5
N <- 9

plot_poisson(t_value,lambda,N)
```


Realization
Events
Time