Proyecto Procesos Estocásticos II

López Canché Luis Brayan

2024-05-04

1. Código que simule trayectorias de un proceso de renovación. El siguiente código nos simula la trayectoria de un proceso de renovación, en particular uno con tiempo de interrarribo que se distribuyen $T_n \sim \Gamma(\alpha, \beta)$.

```
RenovacionWn <- function(t, alpha, beta){
  #La funcion recibe 3 parametros y regresa uno
  #t es el tiempo final hasta el que queremos observar el proceso
  #alpha y beta son los parametros alpha y beta de una gama respectivamente
  #se regresa un vector con los tiempos de arribo que ocurren antes de t (Wn)

  Wn <- c() #Creamos el vector que regresa los Wn
  TempWn <- rgamma(1,shape = alpha,rate = beta) #tempWn es una variable auxiliar que guarda el
  #tiempo de arribo Wn que se va agregando a la lista siempre y cuando sea menor que t

  while(TempWn <= t){
    #el ciclo while va agregando un tiempo de arribo cada vez que este
    #no supere al tiempo t, si lo hace no se agrega.
    Wn <- c(Wn, TempWn)
    TempWn <- Wn[length(Wn)] + rgamma(1,shape = alpha,rate = beta)
  }

  return(Wn)
}
```

Pongamos a prueba el programa simulando un proceso Poisson de parámetro $\lambda = \frac{1}{2}$ a tiempo 10.

```
RenovacionWn(10, 1, 1/2) #En promedio tienen que ocurrir 5 eventos.
```

```
## [1] 2.872154 2.991811 6.525785 6.943782 8.448180 8.606774 9.067108 9.101540
## [9] 9.510745
```

Al final de este markdown verificaremos si los códigos son correctos con las pruebas señaladas en el PDF.

En este programa no consideramos W_0 en el vector de las W_n que regresa la función `RenovacionWn` por facilidad, memoria y por el hecho que este W_0 siempre vale 0.

2. Código para simular la trayectoria de un proceso Poisson compuesto $X_t = \sum_{i=1}^{N_t} Y_n$. El siguiente código simula un proceso de Poisson compuesto, nos apoyamos de la función anterior `RenovacionWn` para generar las W_n, es decir las T_n se distribuyen $\Gamma(1, \lambda) = exp(\lambda)$, consideremos que las Y_n tienen distribución $\Gamma(a, b)$.

```r
PoissonCompuesto <- function(t, lambda, a,b){
  #La funcion PoissonCompuesto recibe 4 parametros y regresa un dataframe con 2 columnas
  #t es el tiempo final hasta el que queremos observar el proceso
  #lambda es la intensidad de los tiempos de interarribo
  #a y b son los parametros de la distribucion gamma que tendran los saltos
  #el dataframe que se regresa contiene los tiempos de salto (Wn) y los tamaños de salto (Yn)
  Wn <- RenovacionWn(t, 1, lambda) #generamos nuestro tiempo de salto
  Yn <- rgamma(length(Wn), shape = a, rate=b)#generamos los tamaños de salto

  df_poission_compuesto <- data.frame(Wn, Yn)
  df_poission_compuesto
}
```

Veamos si lo que nos regresa el programa tiene sentido, más adelante lo comprobaremos matematicamente.

```r
set.seed(1)
PoissonCompuesto(16, 1/2, 1, 1/4)
```

```
##           Wn        Yn
## 1  0.3102827 1.2294933
## 2  4.0750859 0.3784763
## 3  7.6841109 0.6288061
## 4  9.3564662 1.2432215
## 5 11.8015535 1.8749276
## 6 14.1182640 0.2727894
```

3. Código para simular la trayectoria de un proceso de Cramer-Lundberg.

$$R_t = u + ct - \sum_{n=1}^{N(t)} Y_n$$

```r
CramerLundberg <- function(t, u, c, lambda, a,b){
  #La funcion CramerLundberg recibe 6 parametros y regresa un dataframe con 3 columnas
  #t es el tiempo final hasta el que queremos observar el proceso
  #u es el capital inicial
  #c es la prima
  #lambda es la intensidad de los tiempos de interarribo
  #a y b son los parametros de la distribucion gamma que tendran los saltos
  #Se regresa un data frame con las siguientes columnas
  #Wn tiempos de arribo (tiempos de salto)
  #Yn tamaño de salto
  #Rt el proceso en el tiempo t

  df_poissoncompuesto <- PoissonCompuesto(t, lambda, a,b)#apoyados de la funcion PoissionCompuesto
  #generamos los Wn y Yn


  df_cramerlundberg <-data.frame(Wn = df_poissoncompuesto$Wn, Yn = df_poissoncompuesto$Yn)
  df_cramerlundberg$Rt <- u + c*df_poissoncompuesto$Wn - cumsum(df_poissoncompuesto$Yn)


  return(df_cramerlundberg)
}
```

Simulemos el proceso:

```
CramerLundberg(10, 500, 10, 1/2, 1,1/5)
```

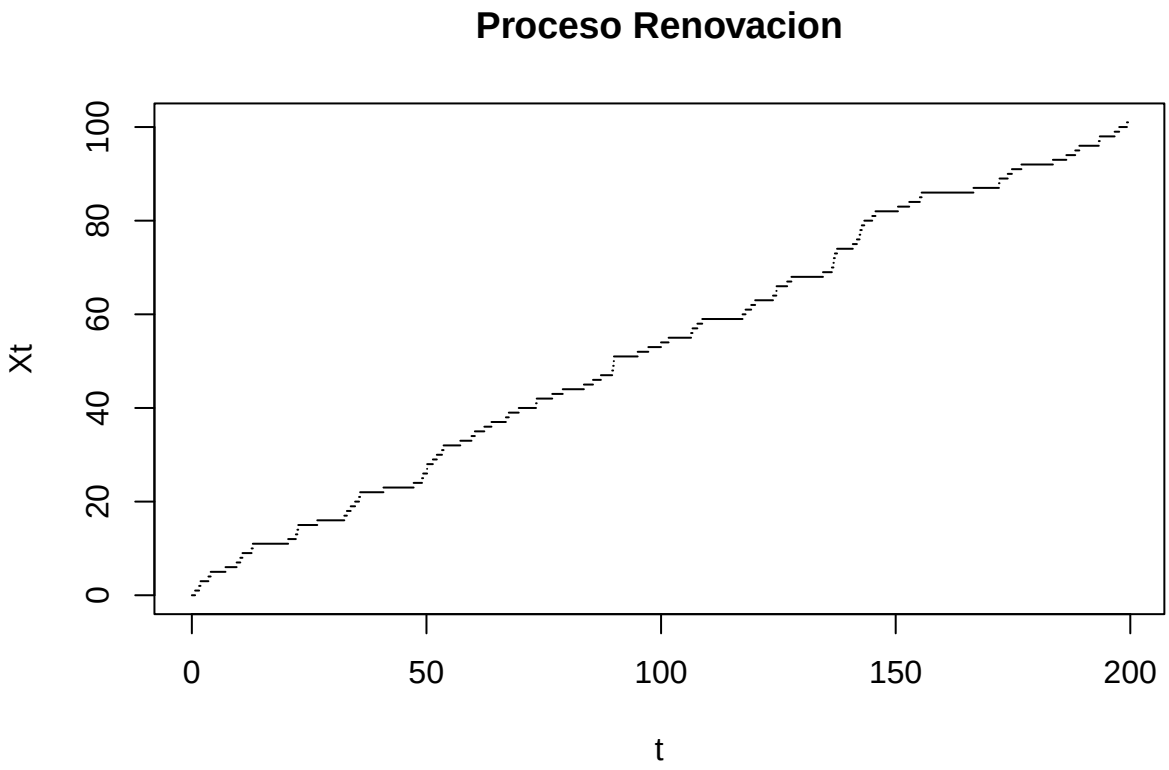
```
##         Wn        Yn       Rt
## 1 2.498584 6.0350130 518.9508
## 2 4.514847 7.8123933 531.3011
## 3 7.236737 0.6538808 557.8661
## 4 9.648713 1.0952686 580.8906
```

Verificaciones

i. Verificamos que 1 y 2 complen el teorema elemental de renovación y graficamos las trayectorias para cada uno.

```
#Primero grafiquemos un proceso de renovación, este es un
#proceso poisson con intensidad 1/2
renov<-RenovacionWn(200, 1, 1/2)
n <- length(renov)
xa <- c(0,renov[-n])
xb <- renov
y <- c(0:(n-1))

plot(xa, y, col = "white", main="Proceso Renovacion",xlab = "t", ylab = "Xt")
segments(x0=xa,y0=y,x1=xb, y1=y)
```



Ahora comprobamos el teorema fundamental de renovación, esperamos que

$$\lim_{t\to\infty} \frac{N(t)}{t} = \frac{1}{E(T_n)} = \frac{\beta}{\alpha}$$

. *El usuario debe ingresar los parámetros.*

```
#Simulamos el procesos de renovacion para comprobar el teorema elemental de renovacion

t<-1000 #mientras mas tiempo mejor es la aproximación
alpha <- 2
beta <- 1
Wn <- RenovacionWn(t, alpha, beta)
limite <- length(Wn)/t
limite
```

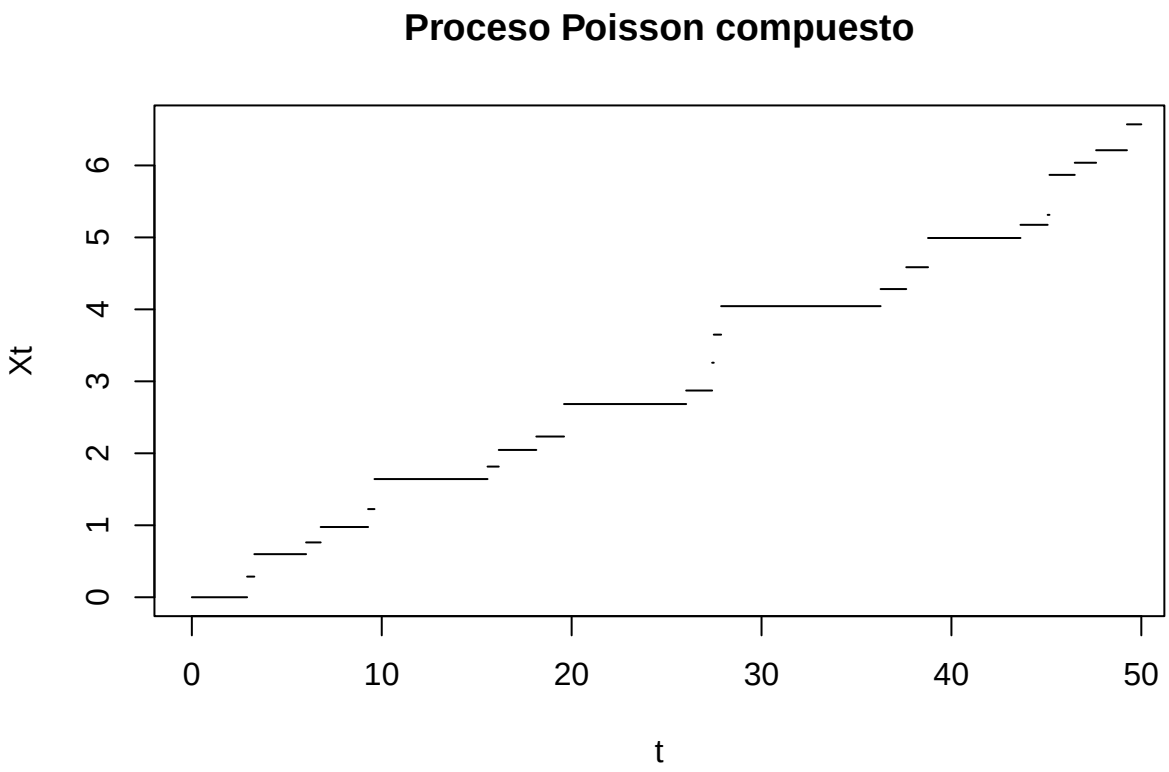
```
## [1] 0.508
```

```
#ahora grafiquemos el procesos poisson compuesto
#se pueden experimentar con parametros si asi lo desea

renov<-PoissonCompuesto(50, 1/2, 5, 16)

n <- nrow(renov)
xa <- c(0,renov$Wn)
xb <- c(renov$Wn,50)
y <- c(0,cumsum(renov$Yn))


plot(xa, y, col = "white", main="Proceso Poisson compuesto",xlab = "t", ylab = "Xt")
segments(x0=xa,y0=y,x1=xb, y1=y)
```



Ahora comprobamos el teorema fundamental de renovación con premio, esperamos que

$$\lim_{t\to\infty} \frac{\sum_{n=1}^{N(t)} Y_n}{t} = \frac{E(Y_n)}{E(T_n)} = \frac{\alpha\lambda}{\beta}$$

. *El usuario debe ingresar los parámetros.*

```
t<- 1000

lambda <- 5
alpha <- 1.3
beta <- 7.9
Xt<-PoissonCompuesto(t, lambda, alpha, beta)

limite2 <- sum(Xt$Yn)/t
limite2
```

```
## [1] 0.8065781
```

ii. Supongamos que las reclamaciones se hacen de forma $Yn \sim exp(\alpha)$ y comprobemos que la probabilidad de ruina se aproxima a

$$\psi(u) = \frac{\lambda}{\alpha c}\exp\{-(\alpha - \frac{\lambda}{c})\}$$

Tomemos los siguientes parámetros: $u = 10$, $\lambda = 8.25$, $\alpha = 1.2$, $c = 6.9$. Vemos que la probabilidad de ruina de debe de acercar a 0.9539843.

```
u = 10
lambda = 8.25
alpha = 1.2
c = 6.9
psi_u = (lambda/(alpha*c))*exp(-(alpha- lambda/c)*u)
psi_u
```

```
## [1] 0.9539843
```

Ahora calculemos la probabilidad de ruina.

```
set.seed(123)
n<-100
t<-4000
u <- 10
lambda = 8.25
alpha = 1.2
c = 6.9
cont <-0


for(i in 1:n){
  if(i%%10 == 0){print(paste(i,"%", sep=""))} #Porcentaje de carga
  df_cramer <- CramerLundberg(t, u, c,lambda ,1, alpha)
  for(j in df_cramer$Rt){
      if(j<0){
      cont <- cont + 1
      break
      }
}
}
```

```
## [1] "10%"
## [1] "20%"
## [1] "30%"
## [1] "40%"
## [1] "50%"
## [1] "60%"
## [1] "70%"
## [1] "80%"
## [1] "90%"
## [1] "100%"
```

```
#calculamos e imprimimos probabilidad de ruina
proba_ruina <- cont/n
proba_ruina
```

```
## [1] 0.95
```

Sustituimos el proceso Poisson compuesto que aparece en el modelo de Cramer-Lundber por un proceso de renovación general, donde $Y_n \sim \exp(\frac{1}{20})$ y $T_n \sim \Gamma(10, 2)$. Estimamos la probabilidad de ruina tomando $u = 1000$ y $c = 5$.

```r
u <- 1000 #capital inicial
c <- 5 #prima
n <- 1000
t<-10000
cont2 <- 0



#Obtenemos las veces que el proceso se va a la ruina
for(j in 1:n){
  Wn <- RenovacionWn(t, 10, 2) #generamos nuestro tiempo de salto gama(10,2)
  Yn <- rexp(length(Wn), rate=1/20)#generamos los tamaños de salto exp(1/20)
  Rt <- u + c*Wn - cumsum(Yn) #Generamos el proceso
  for(rt in Rt){
    if(rt < 0){
      cont2 <- cont2 + 1
      break
  }
}
}
```

```r
#calculamos e imprimimos probabilidad de ruina
proba_ruina2 <- cont2/n
proba_ruina2
```

```
## [1] 0
```

Obtenemos que la probabilidad de ruina es 0, esto tiene sentido; pues, la prima definida como c nos da ganancias de 5 cada unidad de tiempo. Por otro lado, usando el teorema de renovación con premio notamos que para tiempos grandes se van perdiendo una cantidad de

$$\frac{E(Y_n)}{E(T_n)} = \frac{20}{10/2} = 4$$

. Es decir, para tiempos grandes hay más ganancias que pérdidas, y dado que empezamos con un capital inicial grande (relativo a las pérdidas y ganancias), entonces es casi imposible que la aseguradora se vaya a la ruina. Se concluye que los cálculos númericos son tienen sentido.