Benemérita Universidad Autónoma de Puebla

Modelos de Simulación Actuarial

**Simulación del Modelo clásico de Cramér-Lundberg**

Brando Alberto Toribio García

13 de diciembre de 2020

# 1. Introducción

Dado que es un software especializado para la manipulación de datos y nos provee de paqueterías con funciones para la generación de valores aleatorios de las distribuciones deseadas, el código se escribió en R. Cabe destacar que a pesar de existir una paquetería llamada *ruin* y que esta nos provee de utilidades para el *Modelo clásico de Cramér-Lundberg* y las tan deseadas simulaciones de este para la primera parte, no se tomó en cuenta y se decidió crear el código "manualmente" para los procesos estocásticos involucrados, así como la consecuente modificación necesaria para la segunda versión de las simulaciones que se puede observar en el capítulo siguiente.

# 2. Simulaciones

El modelo matemático a considerar en las simulaciones para describir el balance de la compañía a un tiempo t es el *Modelo clásico de Cramér-Lundberg*, el cual está dado por

$$C_t = u + ct - \sum_{j=1}^{N_t} Y_j \tag{1}$$

donde $u \geq 0$, $c > 0$, $Y_1, Y_2, ...$, es una sucesión de v.a.i.i.d positivas con $E(Y_j) = \mu$ e independientes del proceso $\{N(t) : t \geq 0\}$ de parámetro $\lambda$.
Así, $u$ representa el capital inicial de la compañía aseguradora, $c$ el costo de la prima, es decir, $ct$ corresponde a la entrada por primas hasta el tiempo t, $Y_j$ el monto de la j-ésima reclamación y el proceso de Poisson $\{N(t) : t \geq 0\}$ el número de reclamaciones recibidas en el intervalo.
Por lo tanto $C_t$ es el balance de la compañía considerando capital inicial + ingresos - egresos.

Para realizar la simulación se empezó desarrollando el código para el proceso de Poisson que se utilizaría para modelar el número de reclamaciones $\{N(t) : t \geq 0\}$.
Por definición, tenemos

$$N_t = max\{n \geq 1 : T_1 + ... + T_n \leq t\}$$

donde $T_1, T_2, ...$ es una sucesión de v.a.i.i.d con distribución $exp(\lambda)$ y así, $\{N(t) : t \geq 0\}$ es el processo de Poisson de parámetro $\lambda$.

Por lo que nuestro código sería:

```
ProcesoPoissonHomogeno <- function(lambda_Nt,t){
  n <- 0
  Wn <- 0
  while(Wn <= t){
    Wn <- Wn + rexp(1,lambda_Nt)
    n <- n + 1
  }
  n
}
```

Por lo tanto, para el caso en el que $Y_1, Y_2, ...$ se distribuyen de manera $exp(\lambda_2)$, tenemos para la función $C_t$ del *Modelo clásico de Cramér-Lundberg*:

```
clcontinuo <- function(u,c,t,lambda_Nt,lambda_Yj){
  if (t == 0){
    Ct = u
    Ct
  }else{
    Xt <- 0
    Nt <- ProcesoPoissonHomogeno(lambda_Nt,t)
    for(j in 1:Nt){
      Xt <- Xt + rexp(1,lambda_Yj)
    }
    Ct <- u + c*t - Xt
    Ct
  }
}
```

Ahora, basta con construir una función que nos regrese la función $C_t$ evaluada en todos los enteros del intervalo $[0, t]$ un número n de simulaciones (No_simulaciones).

```
simular <- function(u,c,t,lambda_Nt,lambda_Yj,no_simulaciones){
  val2 <- NULL
  for(i in 1:no_simulaciones){
    val1 <- NULL
    for(k in 0:t){
      val1 <- c(val1,clcontinuo(u,c,k,lambda_Nt,lambda_Yj))
    }
    val2 <- c(val2,val1)
  }
  simulaciones <- data.frame(sim = rep(c(1:no_simulaciones), each = t+1),
  t = rep(c(0:t), no_simulaciones),
  val = val2)
  simulaciones
}
```

Para las gráficas que estaremos mostrando en las simulaciones haremos uso de la paquetería "ggplot2". Es decir, al inicio de nuestro código tendremos:

```
install.packages("ggplot2")
library(ggplot2)
```

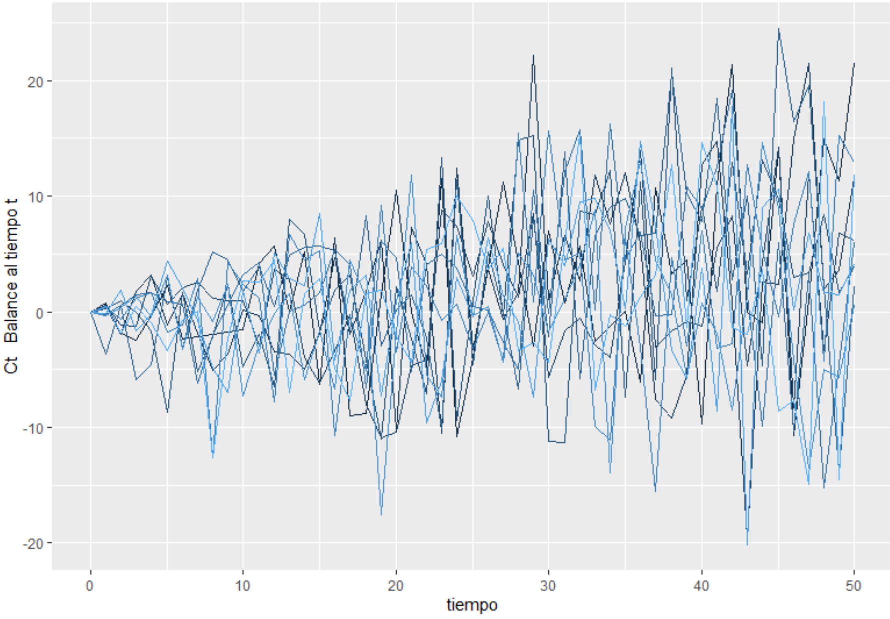
## 2.1. Simulación 1

Haremos 10 repeticiones del modelo para la primera simulación, donde tomaremos un capital inicial de cero, es decir, $u = 0$, un parámetro de $\lambda = 0{,}9$ para nuestro proceso de Poisson del número de reclamaciones, $c = 1$, $t = 50$ y $\mu = 1$. Notemos que dado $\mu = 1$ y como $1 = \mu = \mu_1 = E(Y) = \frac{1}{\lambda_2} = \frac{1}{1} = 1$, entonces $Y_1, Y_2, ...,$ se distribuyen de manera $exp(1)$.
Así, para guardar todo en un "data frame con factores en la primera columna", puesto que es el tipo de archivo que acepta "ggplot2", escribimos

```
grafica <- simular(0,1,50,0.9,1,10)
```

Finalmente, graficamos

```
ggplot(grafica, aes(x=t, y=val, group=sim)) +
geom_line(aes(color=sim)) + theme(legend.position="none") +
scale_x_continuous(name="tiempo") +
scale_y_continuous(name="Ct   Balance al tiempo t")
```




Simulación del *Modelo clásico de Cramér-Lundberg*
con $u = 0$, $\lambda = 0{,}9$, $c = 1$, $t = 50$ y reclamaciones
con distrubución $exp(1)$ y media $\mu = 1$.

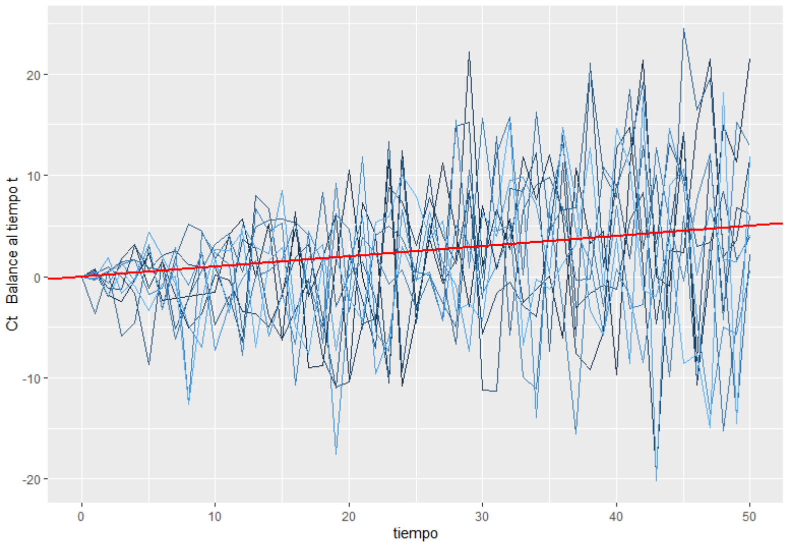
A simple vista podrían parecer inconsistentes nuestras simulaciones debido a los saltos de la misma, por lo que trazaremos la recta de la $E(C_t)$ para mostrar sentido en estas, dicha recta está dada por

$$E(C_t) = u + (c - \lambda\mu)t \tag{2}$$

y que para este caso particular es $E(C_t) = 0 + (1 - (0{,}9(\frac{1}{1})))t = 0{,}1t$. Por lo que sólo reescribimos nuestra graficación como:

```
ggplot(grafica, aes(x=t, y=val, group=sim)) +
geom_line(aes(color=sim)) + theme(legend.position="none") +
geom_abline(intercept = 0, slope = (1 - (0.9*(1/1))), color="red",size=1) +
scale_x_continuous(name="tiempo") +
scale_y_continuous(name="Ct   Balance al tiempo t")
```

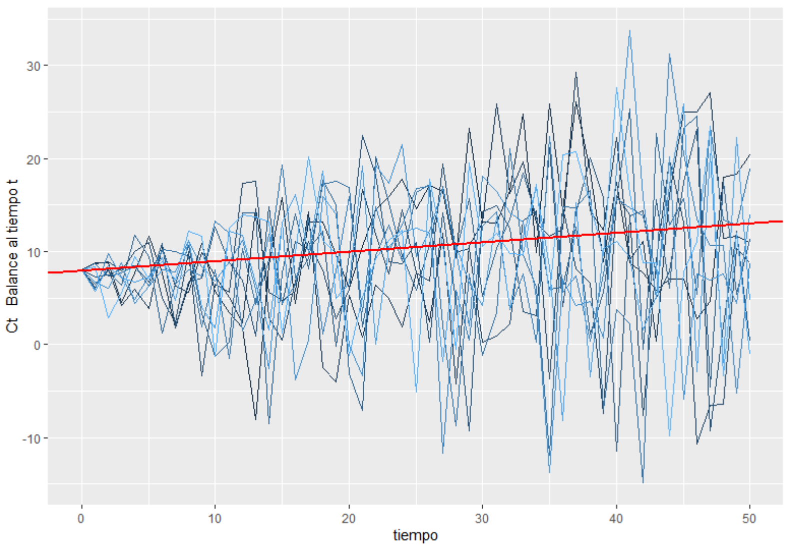
Y así, tenemos




Simulación del *Modelo clásico de Cramér-Lundberg* con $E(C_t) = 0{,}1t$,
$u = 0$, $\lambda = 0{,}9$, $c = 1$, $t = 50$ y reclamaciones
con distrubución $exp(1)$ y media $\mu = 1$.

## 2.2. Simulación 2

Si ahora decidiesemos que la compañía comenzara con un capital, digamos $u = 8$, pero con todos los demás valores iguales, entonces obtendríamos valores "desplazados" verticalmente hacia arriba, como se puede observar en la siguiente figura




Simulación del *Modelo clásico de Cramér-Lundberg* con $E(C_t) = 0{,}1t + 8$,
$u = 8$, $\lambda = 0{,}9$, $c = 1$, $t = 50$ y reclamaciones
con distrubución $exp(1)$ y media $\mu = 1$.

## 2.3. Simulación 3

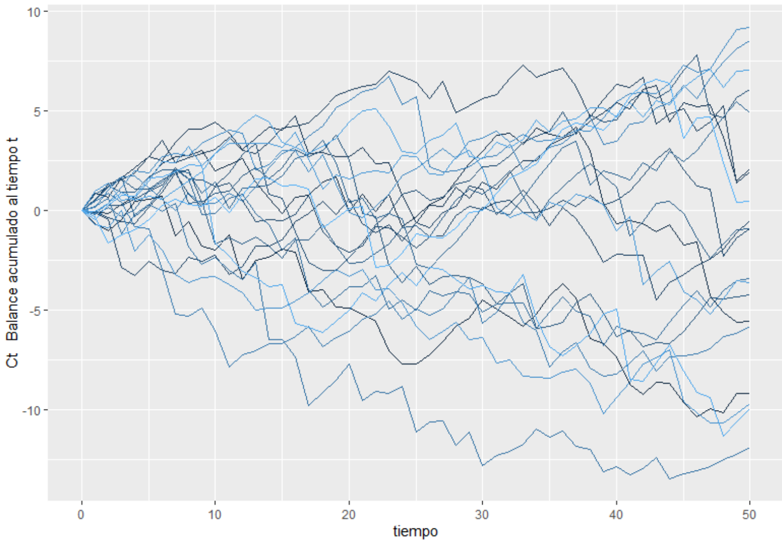
Ahora bien, estas simulaciones son generaciones del proceso $C_t$ donde obtenemos el balance para cada $s\epsilon[0, t], s\epsilon\mathbb{Z}$. Es decir, el proceso se reinicia en cada tiempo $s + 1$, por lo que ahora escribiremos un código que nos denote el monto de las reclamaciones y que este tome en cuenta lo que se tenía anteriormente y que así el balance sea una función acumulada de los ingresos y egresos anteriores como era el **objetivo** de estas simulaciones.
Empezamos escribiendo el código siguiente que nos denota el monto total de las reclamaciones

```
reclamaciones <- function(t,lambda_Nt,lambda_Yj){
  if (t == 0){
    Xt = 0
    Xt
  }else{
    Xt <- 0
    Nt <- ProcesoPoissonHomogeno(lambda_Nt,t)
    for(j in 1:Nt){
      Xt <- Xt + rexp(1,lambda_Yj)
    }
    Xt
  }
}
```

Y después reescribimos nuestro *Modelo clásico de Cramér-Lundberg* como

```
simular12 <- function(u,c,t,lambda_Nt,lambda_Yj,no_simulaciones){
  val2 <- NULL
  for(i in 1:no_simulaciones){
    val1 <- NULL
    val1 <- u
    for(k in 1:t){
      val1 <- c(val1,
      val1[k] + c - reclamaciones(1,lambda_Nt,lambda_Yj))
    }
    val2 <- c(val2,val1)
  }
  simulaciones <- data.frame(sim = rep(c(1:no_simulaciones), each = t+1),
  t = rep(c(0:t), no_simulaciones),
  val = val2)
  simulaciones
}
```

Así, para ilustrar un ejemplo del **objetivo** de estas simulaciones, se simuló el siguiente ejemplo:




Simulación del *Modelo clásico de Cramér-Lundberg* con $u = 0$, $\lambda = 1$, $c = 1$, $t = 50$ y reclamaciones con distrubución $exp(2)$ y media $\mu = 1/2$.

## 3. Probabilidad de ruina por Regla recursiva

Retomando la teoría vista en clase, tenemos la siguiente proposición

**Probabilidad de ruina con horizonte infinito**

Suponemos que la función de distribución $F(y)$ de una reclamación en el *Modelo clásico de Cramér-Lundberg* es continua. Entonces

1. $\frac{d}{du}\bar{\psi}(u) = \frac{\lambda}{c}(\bar{\psi}(u) - \int_0^u \bar{\psi}(u-y)dF(y))$
2. $\psi(0) = \frac{\lambda u}{c}$
3. $\psi(u) = \frac{\lambda}{c}(\int_u^\infty \bar{F}(y)dy + \int_0^u \psi(u-y)\bar{F}(y)dy)$

donde $\psi(u) = P(\tau < \infty \mid C_0 = u)$, es decir, $\psi(u)$ es la probabilidad de ruina con horizonte infinito dado un capital inicial de $u$.
Además, $\bar{\psi}(u) = 1 - \psi(u)$ es la probabilidad de no ruina y $\bar{F}(y) = 1 - F(y)$.

Luego, para nuestro caso de estudio tenemos que los montos de las reclamaciones tienen distribución exponencial, es decir $exp(\alpha)$, y los números de reclamos siguen un Proceso de Poisson con parámetro $\lambda$. Así, nosotros tenemos que la probabilidad de ruina para este caso está dado por

$$\psi(u) = \frac{\lambda}{\alpha c} e^{-(\alpha-\lambda/c)u} \tag{3}$$

Ahora creamos una función en R para facilitar la graficación y obtención de estos valores.

```
psi_ruina_exponencial <- function(u,lambda,alpha,c){
  psi_ruina_exponencial <- (lambda/(alpha*c))*exp(-u*(alpha-(lambda/c)))
  psi_ruina_exponencial
}
```

Así, para la **Simulación 1** de la Sección 2.1 donde $u = 0$, $\lambda = 0{,}9$, $c = 1$ y $\alpha = 1$ tendríamos que por (3), la ruina estará dada por

$$\psi(0) = \frac{0{,}9}{(1)(1)} e^{-0(1-0{,}9/1)} = 0{,}9$$

O bien,

```
> psi_ruina_exponencial(u = 0,lambda = 0.9,alpha = 1,c = 1)
[1] 0.9
```

Luego, para la **Simulación 2** donde $u = 8$, $\lambda = 0{,}9$, $c = 1$ y $\alpha = 1$, tenemos

```
> psi_ruina_exponencial(u = 8,lambda = 0.9,alpha = 1,c = 1)
[1] 0.4043961
```

Lo cual tiene sentido considerando que si comenzamos con un monto inicial mayor a cero, la probabilidad de llegar a la ruina tendría que ser menor a haber empezado con un monto de cero.
Finalmente, para la **Simulación 3** donde $u = 0$, $\lambda = 1$, $c = 1$ y $\alpha = 2$, la probabilidad de ruina será

```
> psi_ruina_exponencial(u = 0,lambda = 1,alpha = 2,c = 1)
[1] 0.5
```

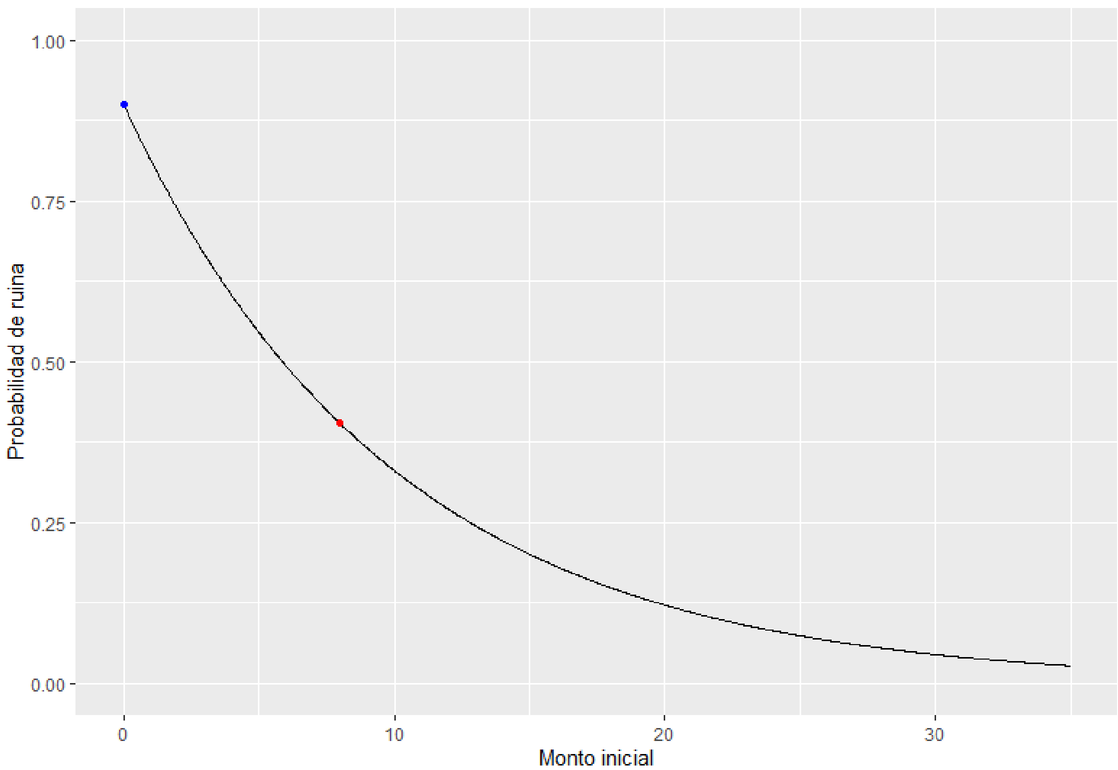
Para finalizar crearemos una función que nos grafique la probabilidad de ruina a diferente cantidad de montos iniciales, pero mismos parámetros.

```
library(ggplot2)

graficacion <- function(lambda,alpha,c,n){ #n: Monto maximo
  monto <- c(NA)
  monto[1] <- 0
  prob <- c(NA)
  prob <- psi_ruina_exponencial(0,lambda,alpha,c)
  for(i in seq(from=0.1, to=n, by=0.05)){
    monto <- c(monto,i)
    prob <- c(prob,psi_ruina_exponencial(i,lambda,alpha,c))
  }
  datos <- data.frame(x=monto, y=prob)
  print(qplot(x, y, data=datos, geom="line",
  xlab = "Monto inicial", ylab = "Probabilidad de ruina") +
  xlim(0, n) + ylim(0,1))
}
```

Así, para los parámetros de las **Simulaciones 1 y 2** donde $\lambda = 0{,}9$, $c = 1$ y $\alpha = 1$ tendríamos

```
graficacion(lambda = 0.9,alpha = 1,c = 1,n = 35) +
geom_point(aes(x=0, y=0.9), colour="blue") +
geom_point(aes(x=8, y=0.4043961), colour="red")
```

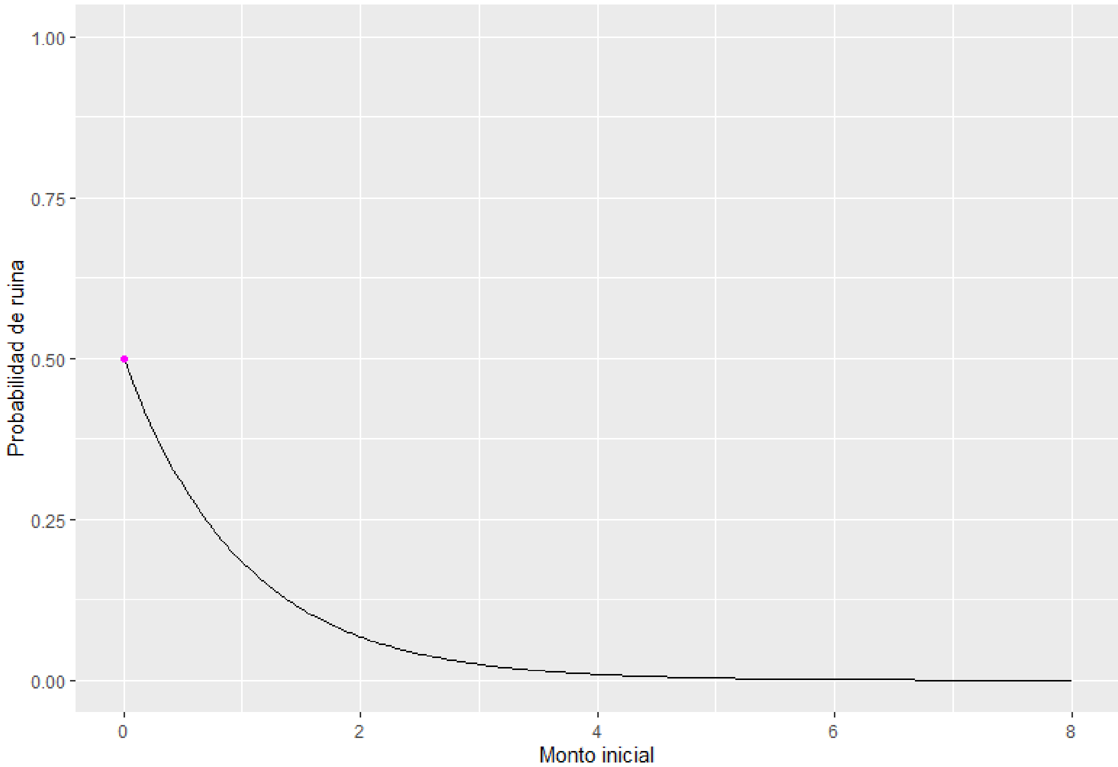



Probabilidad de ruina con horizonte infinito para el caso exponencial con $\lambda = 0{,}9$, $c = 1$ y $\alpha = 1$

Note que el Punto azul indica la probabilidad de ruina con $u = 0$ (Simulación 1) y el Punto rojo la probabilidad de ruina con $u = 8$ (Simulación 2).

Para los parámetros de la **Simulación 3** donde $\lambda = 1$, $c = 1$ y $\alpha = 2$ tenemos

```
graficacion(lambda = 1,alpha = 2,c = 1,n = 8) +
geom_point(aes(x=0, y=0.5), colour="magenta")
```




Probabilidad de ruina con horizonte infinito para el caso exponencial con $\lambda = 1$, $c = 1$ y $\alpha = 2$

Notemos ahora que el Punto magenta indica la probabilidad de ruina para el caso donde $u = 0$ en la Simulación 3.