Conmutados actuariales en R

Alarcón González Edgar Gerardo

11 de Junio del 2018

El objetivo de este documento es trabajar con elementos básicos de R de manera didáctica y aplicada. Por esta razón, no entraremos en detalle de cómo funciona la teoría ni las diferentes variaciones que tiene el modelo, simplemente nos remitiremos a ejemplificar su uso en un contexto real. En éste documento, se busca ejemplificar el uso de los ciclos y estructuras de control en R principalmetne, así como recalcar la ventaja que tiene éste lenguaje computacional al operar de manera vectorial.

Librerías

```
library(XLConnect) #Para leer archivos .xlsx
library(knitr)     #Para hacer tablas en LaTeX
```

Información inicial

Supongamos que contamos con la siguiente información inicial:

- La probabilidad de que una persona de edad x muera en $[x, x+1)$.
- Un Radix = 1,000,000
- Una tasa de interés del 5.5%

Las probabilidades que se nos otorgan las cargaremos desde una base de datos, dada por el siguiente código:

```
setwd("~/Actuaría/AMAT/R básico/Edgar/Bases de Datos/Conmutados")
Vida<-readWorksheetFromFile("Conmutados Actuariales.xlsx",sheet="Conmutados")
kable(head(Vida))
```

Edad	qx
12	0.000396
13	0.000427
14	0.000460
15	0.000495
16	0.000533
17	0.000575

Tabla de vida

Para comenzar con el cálculo de los valores conmutados, primero calcularemos una tabla de vida, ésta consiste en los valores q_x , p_x , l_x y d_x, dada la información inicial, nosotros contamos únicamente con las q_x y el primer valor de las l_x que corresponde al Radix. Primero realizaremos el cálculo para las p_x con la siguiente fórmula, seguido del código correspondiente:

$$p_x = 1 - q_x$$

```r
#Calculamos las Px
Vida$px<-1-Vida$qx
```

- Posteriormente, realizamos el cálculo de las l_x, estos valores se calculan como sigue:

$$l_x = \begin{cases} \text{Radix=1,000,000} & \text{si} \quad x = 12 \\ (p_{x-1})(l_{x-1}) & \text{si} \quad x > 12 \end{cases}$$

```r
#Establecemos un radix
radix<-1000000

#Calculamos lx
for(i in 1:length(Vida$Edad)){

  #El radix son todos los individuos vivos en la primera edad.
  if(i==1){
    lx<-c(radix)
    next #Ve al siguiente valor de "i" una vez hecho lo anterior.
  }

  #Realizamos el cálculo de los demás
  lx[i]<-Vida$px[i-1]*lx[i-1]

}

#Guardamos los valores en la base de datos y borramos las variables auxiliares:
Vida$lx<-lx ; rm(lx,radix)
```

- Realizamos el cálculo de las d_x, estos valores se calculan como sigue:

$$d_x = (l_x)(q_x)$$

```r
#Podemos calcular dx ya de la siguiente manera:
Vida$dx <- Vida$lx*Vida$qx
```

Una vez concluido lo anterior, nuestra tabla de vida está completa y se ve de la siguiente manera:

```r
kable(head(Vida))
```

Edad	qx	px	lx	dx
12	0.000396	0.999604	1000000.0	396.0000
13	0.000427	0.999573	999604.0	426.8309
14	0.000460	0.999540	999177.2	459.6215
15	0.000495	0.999505	998717.5	494.3652
16	0.000533	0.999467	998223.2	532.0530
17	0.000575	0.999425	997691.1	573.6724

(Lo mostrado en la tabla anterior es solamente un fragmento de la tabla completa con más edades.)

Cálculo de conmutados

Para ésta parte de los cálculos, necesitaremos hacer uso de la tasa de interés que habíamos mencionado en la información inicial, los valores conmutados actuariales que estamos por calcular son: D_x, N_x, S_x, C_x, M_x y R_x condicionados a la tasa de interés fijada anteriormente. Comenzamos utilizando la siguiente expresión:

$$D_x = (l_x)(1 + \text{tasa})^{-x}$$

```
#Establecemos una tasa de interés
Tasa<-0.055 #=5.50%

#Calculamos las Dx
Vida$Dx<-Vida$lx*(1+Tasa)^(-Vida$Edad)
```

- Realizamos el cálculo de las N_x, estos valores se calculan como sigue:

$$N_x = \sum_{i \geq x} D_i$$

```
#Calculamos las Nx
i<-1 ; Nx<-c()
attach(Vida)
while(i<=length(Edad)){

  Nx[i]<-sum(Dx[i:length(Edad)])
  i<-i+1

}
detach(Vida)

#Guardamos los valores en la base de datos:
Vida$Nx<-Nx ; rm(Nx)
```

- Realizamos el cálculo de las S_x, estos valores se calculan como sigue:

$$S_x = \sum_{i \geq x} N_i$$

```
#Calculamos las Sx
i<-1 ; Sx<-c()
attach(Vida)
repeat{

  Sx[i]<-sum(Nx[i:length(Edad)])
  i<-i+1

  if(i>length(Edad)){break}

}
detach(Vida)

#Guardamos los valores en la base de datos:
Vida$Sx<-Sx ; rm(Sx)
```

- Realizamos el cálculo de las C_x, estos valores se calculan como sigue:

$$C_x = (d_x)(1 + \text{tasa})^{-(x+1)}$$

```
#Calculamos las Cx
Vida$Cx<-Vida$dx*(1+Tasa)^-(Vida$Edad+1)
```

- Realizamos el cálculo de las M_x, estos valores se calculan como sigue:

$$M_x = \sum_{i \geq x} C_i$$

```
#Calculamos las Mx
i<-1 ; Mx<-c()
attach(Vida)
while(i<=length(Edad)){

  Mx[i]<-sum(Cx[i:length(Edad)])
  i<-i+1

}
detach(Vida)

#Guardamos los valores en la base de datos:
Vida$Mx<-Mx ; rm(Mx)
```

- Realizamos el cálculo de las R_x, estos valores se calculan como sigue:

$$R_x = \sum_{i \geq x} M_i$$

```
#Calculamos las Rx
i<-1 ; Rx<-c()
attach(Vida)
repeat{

  Rx[i]<-sum(Mx[i:length(Edad)])
  i<-i+1

  if(i>length(Edad)){break}

}
detach(Vida)

#Guardamos los valores en la base de datos:
Vida$Rx<-Rx ; rm(Rx,i,Tasa)
```

El lector debe notar las diferencias que hay entre escribir de manera vectorial una operación en R, contra el escribirla mediante un ciclo; es importante mencionar que los ciclos, en cualquier lenguaje de programación, son ineficientes, afortunadamente en el caso de R, podemos operar de manera vectorial, es decir, operando entrada a entrada un vector sin necesidad de un ciclo que haga esta tarea. Siempre que se puedan evitar los ciclos, el código será cada vez más eficiente.

Una vez terminado el proceso anterior, tenemos completa nuestra tabla de conmutados, podemos mostrar un fragmento de la tabla completa con la siguiente parte de código:

```
kable(head(Vida[,1:6])) ; kable(head(Vida[,c(1,7:11)]))
```

Edad	qx	px	lx	dx	Dx
12	0.000396	0.999604	1000000.0	396.0000	525981.5
13	0.000427	0.999573	999604.0	426.8309	498363.3
14	0.000460	0.999540	999177.2	459.6215	472180.5
15	0.000495	0.999505	998717.5	494.3652	447358.6
16	0.000533	0.999467	998223.2	532.0530	423826.7
17	0.000575	0.999425	997691.1	573.6724	401517.3

Edad	Nx	Sx	Cx	Mx	Rx
12	9558933	160033128	197.4300	27648.50	1215974
13	9032952	150474195	201.7072	27451.07	1188326
14	8534589	141441243	205.8797	27249.36	1160875
15	8062408	132906654	209.8981	27043.48	1133625
16	7615050	124844246	214.1229	26833.58	1106582
17	7191223	117229197	218.8365	26619.46	1079748

Ejercicios

Ya que tenemos calculados los valores conmutados, procederemos a realizar algunos ejercicios de cálculo de seguros de vida, pero primero, haremos uso de un pequeño truco que podemos explotar para facilitar la manera en que vamos a pensar y escribir el código, para esto, llamemos a nuestros renglones exactamente como las edades, de tal manera que cuando queramos hacer referencia a alguna edad en particular, podamos llamar a la misma mediante su valor en la posición de los renglones para nuestro `data.frame`; precaución, observe lo que pasa a continuación e intente explicar qué está sucediendo en cada uno de los ejemplos mostrados:

```
#Podemos aplicar el siguiente truco para facilitar llamadas a la base:
rownames(Vida)<-Vida$Edad

#Algunas especificaciones
Vida[12,1:4]
```

```
##    Edad       qx       px     lx
## 23   23 0.000897 0.999103 993523
```

```
Vida["12",1:4]
```

```
##    Edad       qx       px    lx
## 12   12 0.000396 0.999604 1e+06
```

```
Vida[as.character(12),1:4]
```

```
##    Edad       qx       px    lx
## 12   12 0.000396 0.999604 1e+06
```

```
Vida["12","qx"]
```

```
## [1] 0.000396
```

Una vez analizado lo anterior, comencemos a realizar los siguientes ejercicios:

- a) Calcular la probabilidad de que una persona de edad 12 sobreviva hasta edad 14

Solución:

En general tenemos la siguiente expresión:

$$ {}_tp_x = \prod_{i=0}^{t-1} p_{x+i} $$

La respuesta a la pregunta mediante código, es lo siguiente:

```
prod(Vida[1:2,"px"])
```

```
## [1] 0.9991772
```

- b) Calcular la probabilidad de que una persona de edad 30 sobreviva hasta edad 65

Solución:

```
prod(Vida[as.character(30:64),"px"])
```

```
## [1] 0.7840995
```

Mediante el principio de equivalencia:

- c) Calcular la tasa de interes anual "i" tal que el valor presente actuarial de $300 sean $100 para una persona de edad 25 en un periodo de 40 años.

Presentamos la siguiente ecuación de valor **al día de hoy**:

$$ 100 = 300({}_{40}p_{25})(1+i)^{-40} = 300({}_{40}p_{25})V^{-40} $$

Solución con código:

```
tPx <- prod(Vida[as.character(25:(25+39)),"px"])
abs(tPx - Vida["65","lx"]/Vida["25","lx"]) #Comprobación 40p25
```

```
## [1] 1.110223e-16
```

```
V_t <- solve(300*tPx,100)
i = 1/(V_t^(1/40)) - 1

#Comprobación
300*tPx*(1+i)^(-40)
```

```
## [1] 100
```

- d) Calcula la prima neta única de un seguro dotal puro para una persona de edad 28 temporal 42 años con una tasa de interés anual del 5.5% que tiene una suma asegurada de $10,000.

Presentamos la siguiente ecuación de valor **al día de hoy**:

$$ PNU = 10,000({}_{42}p_{28})(1+5.5\%)^{-42} = 10,000\left(\frac{D_{28+42}}{D_{28}}\right) $$

Solución con código:

```
V_t <- (1+0.055)^(-42)
tPx <- prod(Vida[as.character(28:(28+42-1)),"px"])
PNU<-V_t*tPx*10000 ; PNU
```

```
## [1] 733.751
```

```
#Lo mismo pero con los conmutados
Dotal_Puro<-Vida[as.character(28+42),"Dx"]/Vida["28","Dx"]
PNU<-10000*Dotal_Puro ; PNU
```

```
## [1] 733.751
```

- e) Haz el mismo cálculo pero para una persona de edad 25 temporal 1

Presentamos la siguiente ecuación de valor **al día de hoy**:

$$PNU = 10,000({}_1p_{25})(1+5.5\%)^{-1} = 10,000\left(\frac{D_{25+1}}{D_{25}}\right)$$

Solución con código:

```
V_t <- (1+0.055)^(-1)
tPx <- prod(Vida[as.character(25:(25+1-1)),"px"])
PNU<-V_t*tPx*10000 ; PNU
```

```
## [1] 9468.806
```

```
#Lo mismo pero con los conmutados
Dotal_Puro<-Vida[as.character(25+1),"Dx"]/Vida["25","Dx"]
PNU<-10000*Dotal_Puro ; PNU
```

```
## [1] 9468.806
```

- f) Calcula la prima neta única de un seguro que paga en caso de fallecimiento (al final del año de fallecimiento) para una persona de edad 70 temporal 2 con una tasa de interés anual del 5.5% que tiene una suma asegurada de $1,000,000 al final del año de fallecimiento.

Presentamos la siguiente ecuación de valor **al día de hoy**:

$$PNU = 1,000,000\sum_{t=0}^{2-1}(q_{70+t})({}_tp_{70})(1+5.5\%)^{-(t+1)} = 1,000,000\left(\frac{M_{70}-M_{70+2}}{D_{70}}\right)$$

Solución con código:

```
SA <- 1000000
V_t <- c((1+0.055)^-1,(1+0.055)^-2)
qx <- Vida[as.character(70:71),"qx"]
tPx <- c(1,Vida["70","px"])
#Cálculo
PNU <- SA * sum(V_t*qx*tPx) ; PNU
```

```
## [1] 54168.59
```

```
#Lo mismo pero con conmutados
Seguro_Temporal<-(Vida["70","Mx"]-Vida[as.character(70+2),"Mx"])/
                  Vida["70","Dx"]
PNU<-SA*Seguro_Temporal ; PNU
```

```
## [1] 54168.59
```