EST–24107: Tarea 1

Carlos Lezama, Marco Medina, Emiliano Ramírez y Santiago Villarreal

Jueves, 26 de agosto de 2021

Problema 1

Con el fin de comprobar la propiedad de falta de memoria de una distribución exponencial

$P(X > t + s \mid X > t) = P(X > s).$

$$F(x;\lambda) = \begin{cases} 1 - e^{-\lambda x} & x \geq 0, \\ 0 & x < 0. \end{cases}$$

definimos la siguiente función:

```
exponential_memoryless <- function(N, lambda) {
    t <- sample(1:10, N, replace = TRUE) # t
    s <- sample(1:10, N, replace = TRUE) # s

    # P(X > t + s | X > t)
    prob.1 <- (1 - pexp(t + s, lambda)) / (1 - pexp(t, lambda))

    # P(X > s)
    prob.2 <- 1 - pexp(s, lambda)

    return(tibble(prob.1, prob.2, almost.equal(prob.1, prob.2)))
}
```

que produce dos vectores aleatorios t y s de tamaño N para calcular $P(X > t + s \mid X > t)$ y $P(X > s)$. Finalmente, devuelve una matriz con las probabilidades calculadas y corrobora que se cumpla la igualdad en cada una de las N simulaciones con una columna lógica `TRUE` o `FALSE`.

Ahora bien, simulamos 100 y 1,000,000 de eventos, imprimiendo una matriz con probabilidades en las que no se cumple la igualdad.

```
lambda <- 1 / 3 # Parámetro
r.1 <- exponential_memoryless(100, lambda) # 100 simulaciones
r.2 <- exponential_memoryless(1000000, lambda) # 1 000 000 simulaciones

r.1[r.1[, 3] == FALSE, ]
```

```
## # A tibble: 0 x 3
## # ... with 3 variables: prob.1 <dbl>, prob.2 <dbl>,
## #   almost.equal(prob.1, prob.2) <lgl>
```

```
r.2[r.2[, 3] == FALSE, ]
```

```
## # A tibble: 0 x 3
## # ... with 3 variables: prob.1 <dbl>, prob.2 <dbl>,
## #   almost.equal(prob.1, prob.2) <lgl>
```

Nótese que la igualdad se cumple en todos los casos, corroborando que:

$$
\begin{aligned}
P(X > t + s \mid X > t) &= \frac{P\left(X > t + s, X > t\right)}{P(X > t)} \\
&= \frac{P(X > t + s)}{P(X > t)} \\
&= \frac{1 - F_X(t + s)}{1 - F_X(t)} \\
&= \frac{e^{-\lambda(t+s)}}{e^{-\lambda t}} \\
&= e^{-\lambda s} \\
&= P(X > s).
\end{aligned}
$$

Problema 4

Simulamos la posible llegada de un cliente $Y \sim \text{Bernoulli}(0.9)$ tal que dicho cliente llega $(X \mid Y = 1) = 5 + u_1$, donde $u_1 \sim P(\text{llegada})$. Finalmente, para determinar el tiempo de salida u_3, necesitamos $u_2 \sim P(\text{tiempo de consulta})$ tal que $u_3 = u_1 + u_2$.

Así pues, para 50 pacientes:

```
N <- 50 # número de pacientes
arrival <- duration <- numeric(N)

arrived <- rbinom(N, 1, 0.9) # qué pacientes llegaron
k <- length(which(arrived == 1)) # cuántos pacientes llegaron

arrival[which(arrived == 1)] <-
    5 + sample(-2:2,
               k,
               prob = c(0.1, 0.2, 0.4, 0.2, 0.1),
               replace = TRUE)
duration[which(arrived == 1)] <-
    sample(2:9,
           k,
           prob = c(0.1, 0.1, 0.1, 0.2, 0.2, 0.1, 0.1, 0.1),
           replace = TRUE)
```

```
arrival.time <- cumsum(arrival)

departure <- arrival.time + duration

# Clientes en espera cada 5 horas
queue.1 <- as.data.frame(table(cut(departure, breaks = seq(0, N * 5, by = 5))))
queue.2 <- as.data.frame(spline(queue.1$Var1, queue.1$Freq))
```