Simulación ITAM
3 de septiembre de 2021 2021-II

Tarea 1.

La fecha de entrega es el **miércoles 26 de agosto de 2021**. Subir la tarea vía Canvas **por equipo** *antes de la medianoche* (o se convierte en calabaza).

Lecturas

- Robert & Casella Capítulo 2 sección 2.1 y 2.2.
- Dagpunar Capítulo 2
- Good random number generators are (not so) easy to find
- Linear Congruential Generator in R

Problemas

1. Una propiedad importante de la distribución exponencial es su amnesia, o falta de memoria:
$$P(X > t + s|X > t) = P(X > s)$$
   i. Escriban en `R` una función con la que se pueda comprobar la propiedad de falta de memoria.
   ii. Usen la función que crearon en el inciso anterior para realizar la comparación de los dos términos de la igualdad, empleando primero una muestra de 100 observaciones y luego usando una muestra de 1,000,000 de observaciones. ¿Qué tan bien se cumple la igualdad en cada caso? Comparen su resultado con el resultado exacto, utilizando la densidad de la función exponencial.

Solución.

Podemos calcular las probabilidades a partir de una muestra dada y verificar si las probabilidades son iguales (o muy parecidas, dado que es una simulación numérica)

```r
set.seed(1)    # fijamos una semilla para reproducibilidad

memoryless <- function(n,t = 1,s = 1, lambda = 1){
  x <- rexp(n,rate = lambda)
  (ps <- length(x[x>s])/n)     # P(X>s)
  y <- x[x>t]        # Evento condicionante
  (ps_t <- length(y[y > s+t])/length(y))   # P(X>s+t|X >t)
  return(ps_t-ps)
```

```
}
memoryless(100)

[1] -0.08666667

memoryless(10000000)

[1] -0.0001762535
```

Para comparar los valores exactos usamos la distribución exponencial

```
s <- t <- 1
ps <- 1-pexp(s)
ps_t <- (1-pexp(s+t))/(1-pexp(s))
ps_t -ps

[1] 0
```

□

2. Lanzar una moneda honesta 500 veces y hacer una gráfica de:

   i r/n vs n, para $n = 1, 2, \ldots, 500$, donde n es el número de lanzamientos y r es el número de soles para esos n lanzamientos; y

   ii. $(2r - n)$ vs n, la diferencia entre el número de soles y águilas.

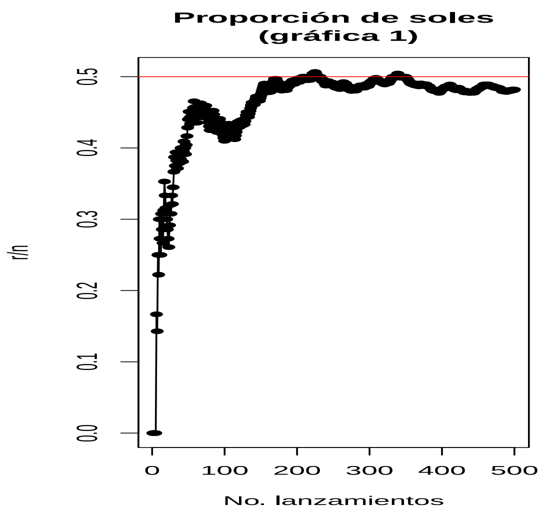
   Comentar sobre el comportamiento de r/n y $(2r - n)$

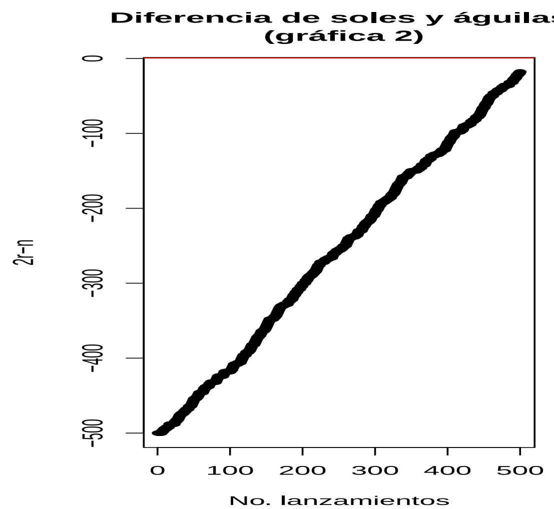
   Solución.

   Podemos simular el lanzamiento de una moneda generando variables aleatorias Bernoulli con parámetro $p = 0.5$.

```
x <- rbinom(n = 500, size = 1, prob = 0.5)
par(mfrow = c(1,2))
# Gráfica de r/n
plot(1:500, cumsum(x)/(1:500), type = "o", pch = 16,
     xlab = "No. lanzamientos", ylab = "r/n",
     main = "Proporción de soles\n (gráfica 1)")
abline(h = 0.5, col = "red")

# Gráfica de 2r-n
plot(1:500, (2*cumsum(x) - rep(500,500)), type = "o", pch = 16,
     xlab = "No. lanzamientos", ylab = "2r-n",
     main = "Diferencia de soles y águilas\n (gráfica 2)")
abline(h = 0.5, col = "red")
```





Podemos ver que conforme se aumenta el número de lanzamientos, el valor de r/n se aproxima al valor de $p = 0.5$. La diferencia entre águilas y soles se aproxima a 0 conforme aumenta el tamaño de la muestra.

□

3. Una canoa que contiene tres mujeres y tres hombres llega a una isla deshabitada. Discutan la información que requieren para modelar la sociedad de estos individuos y cómo el tamaño de la población crece con el tiempo. Por ejemplo, pueden hacer supuestos como los siguientes y hacer modificaciones para ver cómo cambiarían las proyecciones que hagan:

   - Todas las personas son adultos (digamos 20 años todos). La edad de las mujeres es importante para el tema de capacidad reproductiva.
   - Las parejas se determinan al inicio y no hay cambios de pareja a lo largo del tiempo
   - Cada pareja puede tener una bebé al año con probabilidad p, y éste sobrevive con probabilidad w.

   Con los supuestos que hagan, determinen el tiempo promedio en que se duplica la población.

   Solución.

   Se pueden hacer diferentes supuestos para este problema. Por ejemplo, podemos suponer que sólo un hombre es fértil, que las mujeres sólo pueden tener un determinado número de hijos en su periodo reproductivo, etc. O bien podemos hacer supuestos muy simples y considerar esencialmente árboles de expansión, por ejemplo.

   Con el supuesto inicial simple en el que cada hombre mantiene su relación con la misma mujer, y que se puede tener un bebé que vive con probabilidad p cada año y que es hombre o mujer con probabilidad w, y que la edad reproductiva comienza a los 18 años, etc., se podría modelar cómo va creciendo la población y calcular cosas como el tiempo promedio en el que se duplica la población. Un ejemplo podría ser como el siguiente:

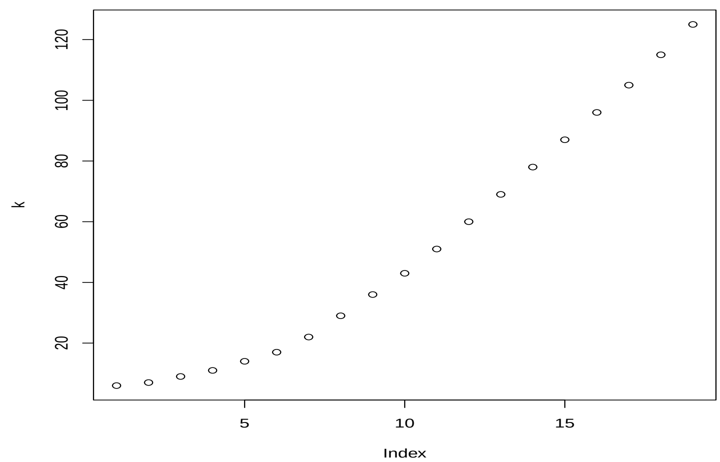
- 20 años todos, consideramos que cada año se tiene un bebé y se simularán 100 años.
- Cada persona vive sólo 100 años y tendrán una vida productiva hasta los 30 años.
- Los bebés se convertirán en adultos después de 20 años, cuando podrán tener vida reproducible.
- Se podrán unir parejas de diferentes padres (no hermanos).
- Los parámetros son:
  - probabilidad de tener un bebé en un año dado: p0 = 0.5
  - probabilidad de que el bebé sea niño: p1 = 0.5
  - probabilidad de que el bebé sobreviva: w = 0.4

```
# Ejercicio de simulación de población

poblacion <- function(parejas = 3, periodos = 100, p0 = 0.5, p1= 0.5, w= 0.4){
  nuevos <- list(NULL)
  pob <- 2 * parejas  # población inicial
  for(i in 1:periodos){
    nacimientos <- sum(rbinom(parejas,1,w)*rbinom(parejas,1,p0))
    niños <- rbinom(nacimientos,1,p1)
    nuevos[[i]] <- list(nacimientos = nacimientos,
                        niños = ifelse(nacimientos==0,0,sum(niños)),
                        niñas = ifelse(nacimientos==0,0,nacimientos - sum(niños)),
                        edad = i-1)
    pob[i+1] <- pob[i] + sum(unlist(lapply(nuevos,function(x)x$nacimientos)))

  }

  return(pob)
}
k <- poblacion(periodos = 18)
plot(k)
```



En este ejercicio no incorporé la formación de parejas después de 18 periodos, pero se podría seguir en esa dirección.

□

4. Considerar cómo podrían simular el siguiente modelo de una sala de cirugía que opera bajo citas:

   - Los pacientes se programan para llegar en cada 5 horas.
   - Independientemente de los otros pacientes, cada paciente falla a su cita con probabilidad 0.1
   - Independientemente de los otros pacientes, cada paciente tiene tiempos de llegada con la siguiente distribución:

Tiempo	2 hrs antes	1 hra antes	a tiempo	1 hra tarde	2 hrs tarde
probabilidad	1/10	1/5	2/5	1/5	1/10

   - Los tiempos de consulta tienen la siguiente distribución:

Tiempo en hrs	2	3	4	5	6	7	8	9
probabilidad	1/10	1/10	1/10	1/5	1/5	1/10	1/10	1/10

   - Los pacientes se atienden en el orden en el que llegan.

Solución.

a) Creamos la lista de llegadas. Se genera una posible llegada de un cliente I cada 5 horas como una variable aleatoria **Bernoulli** (0.9) para determinar si el cliente llega o no llega.

b) Sea $X|I = 1$, el tiempo en el que el cliente llega a su cita. Entonces $(X|I = 1) = 5 + u$, donde u tiene la distribución de probabilidad dada para las llegadas.

c) Para cada llegada, hay que generar la duración de la consulta S, con la distribución dada. Ya con la duración y el tiempo de llegada, se puede determinar el tiempo de salida.

Por ejemplo, podemos hacer un ejemplo sencillo: suponiendo que tenemos $n = 100$ citas.

```
n <- 100
X <- D <- numeric(n) #define un vector
(I <- rbinom(n,1,0.9)) #para determinar los clientes que llegaron
```

```
  [1] 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 0 1 0 1 1 0 1 1 1 1 1 1 0 1 1 1 0 1
 [38] 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 0 1 1 1 0 1 1 1 1 1 1 1
 [75] 0 1 0 1 0 1 0 1 1 1 1 0 1 1 0 0 1 1 1 1 1 1 1 1 1 1
```

```
k <- length(which(I == 1)) #cuenta qué pacientes llegaron
X[which(I == 1)] <- 5 + sample(-2:2,k,prob=c(0.1,0.2,0.4,0.2,0.1),replace=T)
# duraciones de atención
D[which(I == 1)] <- sample(2:9,k,prob=c(0.1,0.1,0.1,0.2,0.2,0.1,0.1,0.1),replace=T)
W <- cumsum(X) #tiempos de llegadas
(S <- W + D) # tiempos de salida
```

```
  [1]   0   9  15  22  25  32  35  43  46  48  59  64  64  71  75  69  84  86
 [19]  81  88  84  95 101  96 103 115 117 120 129 135 142 134 148 149 153 148
 [37] 160 162 169 166 164 173 179 183 187 187 193 194 198 202 208 215 223 228
 [55] 229 238 231 244 245 251 255 253 265 263 273 264 274 278 281 289 296 302
 [73] 303 310 304 311 309 323 314 325 320 333 331 339 345 339 349 353 348 348
 [91] 359 364 367 370 375 379 387 392 392 399
```

```
# obtenemos cuántos clientes en espera por cada 5 horas
table(cut(S,breaks=seq(0,500,by=5)))
```

```
    (0,5]    (5,10]   (10,15]   (15,20]   (20,25]   (25,30]   (30,35]   (35,40]
        0         1         1         0         2         0         2         0
  (40,45]   (45,50]   (50,55]   (55,60]   (60,65]   (65,70]   (70,75]   (75,80]
        1         2         0         1         2         1         2         0
  (80,85]   (85,90]   (90,95]  (95,100] (100,105] (105,110] (110,115] (115,120]
        3         2         1         1         2         0         1         2
(120,125] (125,130] (130,135] (135,140] (140,145] (145,150] (150,155] (155,160]
        0         1         2         0         1         3         1         1
(160,165] (165,170] (170,175] (175,180] (180,185] (185,190] (190,195] (195,200]
        2         2         1         1         1         2         2         1
(200,205] (205,210] (210,215] (215,220] (220,225] (225,230] (230,235] (235,240]
        1         1         1         0         1         2         1         1
(240,245] (245,250] (250,255] (255,260] (260,265] (265,270] (270,275] (275,280]
        2         0         3         0         3         0         2         1
(280,285] (285,290] (290,295] (295,300] (300,305] (305,310] (310,315] (315,320]
        1         1         0         1         3         2         2         1
(320,325] (325,330] (330,335] (335,340] (340,345] (345,350] (350,355] (355,360]
        2         0         2         2         1         3         1         1
(360,365] (365,370] (370,375] (375,380] (380,385] (385,390] (390,395] (395,400]
        1         2         1         1         0         1         2         1
(400,405] (405,410] (410,415] (415,420] (420,425] (425,430] (430,435] (435,440]
        0         0         0         0         0         0         0         0
(440,445] (445,450] (450,455] (455,460] (460,465] (465,470] (470,475] (475,480]
        0         0         0         0         0         0         0         0
(480,485] (485,490] (490,495] (495,500]
        0         0         0         0
```

$\square$

5. Probar que la parte fraccional de la suma de uniformes $[0,1]$ $U_1 + U_2 + \cdots + U_k$

es también uniforme en el intervalo $[0,1]$. (hint: lo pueden hacer por inducción matemática)

Solución.

Este ejercicio se puede probar por inducción. Para facilitar la notación, definamos $\{x\} = x - \lfloor x \rfloor$ como la parte fraccional de x. La densidad de $\{U_1 + U_2\}$ está dada por

$$f_{U_1+U_2}(x) = \begin{cases} x, & 0 \leq x \leq 1 \\ 2-x, & 1 < x \leq 2. \end{cases}$$

La distribución esta dada por la siguiente expresión, de la que ustedes pueden completar los detalles, considerando que $U_1 + U_2$ puede estar entre (0,1) y (1,2)

$$F(x) = \Pr[\{U_1 + U_2\} \leq x] = \int_{u=0}^{x} f_{U_1+U_2}(u)\, du + \int_{1}^{1+x} f_{U_1+U_2}(u)\, du = x.$$

□

6. El método del cuadrado medio de John von Neumann es el siguiente: comenzando con $Z_0 \in \{0, 1, \ldots, 99\}$, definir Z_n para $n \in \mathbb{N}$ a ser los dos dígitos de enmedio del número de 4 dígitos Z_{n-1}^2. Si Z_{n-1}^2 no tiene 4 dígitos, se le pegan a la izquierda con ceros. Por ejemplo, si $Z_0 = 64$, tenemos que $Z_0^2 = 4096$ y entonces $Z_1 = 09 = 9$. En el siguiente paso, encontramos que $Z_1^2 = 81 = 0081$, así que $Z_2 = 08 = 8$.

   - Escriban una función que calcule Z_n a partir de Z_{n-1}.
   - La salida del cuadrado medio tiene bucles. Por ejemplo, una vez que $Z_N = 0$, tendremos que $Z_n = 0$ para toda $n \geq N$. Escriban un programa que encuentre todos los ciclos del método del cuadrado medio y lístenlos.
   - Comenten sobre la calidad del método como generador de números aleatorios.
   - Hacer un diagrama como el mostrado en clase.

Solución.

La función que se solicita es, en el caso de `R`, la siguiente, que es una ligera modificación de la función que vimos en clase (la vista en clase, se terminaba cuando el número era muy corto, y no agregaba ceros a la izquierda). Esta función no da sólo el siguiente valor, sino que calcula toda la secuencia:

```
cm <- function(x){
  #a partir de la semilla x, se genera una sucesión de valores,
  #tomando los valores de enmedio de la serie
  u <- x/10^nchar(as.character(x))
  z <- x
  repeat{
    #verifica que el número tenga suficientes dígitos
    z <- z^2
    n <- nchar(z)
```

```
    if (n < 4) z <- paste(as.character(rep(4-n,0)),as.character(z),sep="")
    #partiendo de un tamaño de 7 dígitos, escogemos los dígitos centrales y vamos recorriendo
    z <- as.numeric(as.character(substr(z,floor(n/2),floor(n/2)+1)))
    u <- append(u,z/10000)
    #si el último número es 0 o ya se repiten termina
    if ((u[length(u)] == 0 ) | length(unique(u)) < length(u)) break
 }
 return(u)
}
cm(64) #ejemplo.
```

```
 [1] 0.6400 0.0009 0.0081 0.0056 0.0013 0.0016 0.0025 0.0062 0.0084 0.0005
[11] 0.0025
```

El siguiente inciso pide calcular las secuencias que se generan con cada número inicial. Podemos construir una lista con los ciclos de cada semilla.

```
Ciclo <- list(NULL)
for (i in 0:99) Ciclo[[i+1]] <- cm(i)
Ciclo[1:10] #ejemplos
```

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

[[2]]
[1] 1e-01 1e-04 1e-04

[[3]]
[1] 0.2000 0.0004 0.0016 0.0025 0.0062 0.0084 0.0005 0.0025

[[4]]
 [1] 0.3000 0.0009 0.0081 0.0056 0.0013 0.0016 0.0025 0.0062 0.0084 0.0005
[11] 0.0025

[[5]]
[1] 0.4000 0.0016 0.0025 0.0062 0.0084 0.0005 0.0025

[[6]]
[1] 0.5000 0.0025 0.0062 0.0084 0.0005 0.0025

[[7]]
[1] 0.6000 0.0036 0.0029 0.0084 0.0005 0.0025 0.0062 0.0084

[[8]]
[1] 0.7000 0.0049 0.0040 0.0060 0.0060

[[9]]
 [1] 0.8000 0.0064 0.0009 0.0081 0.0056 0.0013 0.0016 0.0025 0.0062 0.0084
[11] 0.0005 0.0025

[[10]]
 [1] 0.9000 0.0081 0.0056 0.0013 0.0016 0.0025 0.0062 0.0084 0.0005 0.0025
```

```
unlist(lapply(Ciclo,length))
```

```
  [1]  2  3  8 11  7  6  8  5 12 10  3 11 10  7  9  7  6 15 10  8  4 14  6  6  4
 [26]  6  7 12 14  6  4 16  9 13  8  7  7  8 14  6  3  7 16  6 13  9 12  5  5  4
 [51]  3  3  5  5 16  9  8  4  8  6  3 12  6 16 11  7  8  6  6 16  4  8 11 10  6
 [76]  6 15 14 13  4  4  9 12  8  6  7  7  9  7 14  3 15 13 12  9  9 15  4  3  5
```

Ya vimos que este generador no es bueno, ya que genera ciclos muy cortos, los ciclos más largos en este ejemplo son de longitud 16.

□

7. Si dos dados están cargados de tal manera que en un dado, el valor 1 aparecerá exactamente el doble de veces que los otros valores, y el otro dado está igualmente cargado hacia el 6, calculen la probabilidad p_s de que un total exactamente igual a s aparecerá en la suma de los dos dados, para $2 \leq s \leq 12$.

Solución.

Para cumplir con la condición, sea p la probabilidad de cualquiera de las caras del 2 al 6. Entonces $2p + 5p = 1$ y por lo tanto $p = 1/7$. Entonces en el primer dado el 1 aparecerá con probabilidad $2/7$ y el resto de los números con probabilidad $1/7$, y en el otro dado, la cara con el 6 es el que aparecerá con probabilidad $2/7$.

Para la suma, tenemos como ejemplo los siguientes:

$s = 2$: sólo se da con el par (1,1) con probabilidad $\frac{2}{49}$.

$s = 3$: Se puede dar con los pares (1,2) ($\frac{2}{49}$) o (2,1) ($\frac{1}{49}$). Entonces su probabilidad es $\frac{3}{49}$

$s = 4$: Se puede dar con (1,3) ($\frac{2}{49}$), (2,2) ($\frac{1}{49}$), (3,1) ($\frac{1}{49}$). En total, $\frac{4}{49}$

Y así sucesivamente, para finalmente obtener:

$s=2$	$s=3$	$s=4$	$s=5$	$s=6$	$s=7$	$s=8$	$s=9$	$s=10$	$s=11$	$s=12$
$\frac{2}{49}$	$\frac{3}{49}$	$\frac{4}{49}$	$\frac{5}{49}$	$\frac{6}{49}$	$\frac{9}{49}$	$\frac{6}{49}$	$\frac{5}{49}$	$\frac{4}{49}$	$\frac{3}{49}$	$\frac{2}{49}$

□