

# PROYECTO DE PROBABILIDAD

Valeria Del Pilar López Bosiga<sup>1</sup>, Alejandro Isaac Núñez Ayala<sup>1</sup>, Andrés Felipe Pórtela Guzmán<sup>1</sup> y  
Esneider Fabian Sierra Alba<sup>1</sup>

<sup>1</sup>Probabilidad 2015178-B – Departamento de Estadística

Universidad Nacional de Colombia sede Bogotá – vlopezbo@unal.edu.co, anuneza@unal.edu.co, aportelag@unal.edu.co,  
esierra@unal.edu.co

**Abstract—**

**Index Terms—**

## I. SELECCIÓN DE UN GENERADOR DE NÚMEROS (PSEUDO-)ALEATORIOS

En esta ocasión se usarán dos funciones que generan números aleatorios, principalmente serán números pseudo aleatorios, esto se debe a que el método de obtención de números aleatorios se basa en un proceso físico, y su fuente es principalmente de algún fenómeno físico, donde se espera que este sea aleatorio.

Estos fenómenos ocurren fuera de la computadora y es aquí el punto de inflexión puesto que la función `runif()` [1] y la propuesta particular (FP, *función particular*)<sup>1</sup>, toman datos exclusivamente dentro de la computadora o extraen información que se encuentra en la computadora. En el caso de la FP, uno de los datos que usa el código es la fecha actual del computador como uno de los parámetros, esta será la semilla inicialmente y por defecto.

En el caso de la función `runif()`, tenemos que esta se encuentra hecha en C, y su código es el siguiente<sup>2</sup>.

```
1 #include "nmath.h"
2
3 double runif(double a, double b)
4 {
5     if (!R_FINITE(a) || !R_FINITE(b) || b < a) ↵
6         ML_ERR_return_NAN;
7
8     if (a == b)
9         return a;
10    else {
11        double u;
12        do {u = unif_rand();} while (u <= 0 || u >= ↵
13            1);
14        return a + (b - a) * u;
15    }
```

Se puede observar que la función `runif()`, después de revisar si los parámetros de entrada son correctos, se llama

<sup>1</sup>Se trata de un código de generadores congruenciales lineales [2, p.426–430], esta función se vió en clase, la propuesta se encuentra en la siguiente página <https://bit.ly/39SAdlh> y en el documento llamado PartA.R

<sup>2</sup>El código, `runif.c`, fue extraído de la cuenta de SurajGupta en GitHub, donde se encuentran varias carpetas de R, <https://bit.ly/3bBUBhj>

una nueva función `unif_rand()`, la cual es realmente la función encargada de generar números aleatorios y es la que se observa a continuación<sup>3</sup>.

Por último dos puntos importantes por mencionar, es que se trata de una versión obsoleta, ya que no se encuentra en los parámetros de entrada el número de veces que se desea repetir u obtener números aleatorios. En segundo lugar, el código es coherente con la realidad ya que al ingresar un parametro como `runif(1,2,1)`, nos retorna un NaN y su respectivo warning, lo cual se refleja en el código expuesto a pesar de que si cumplan con el parametro de `R_FINITE()`.

```
1 static unsigned int I1=1234, I2=5678;
2
3 void set_seed(unsigned int i1, unsigned int i2)
4 {
5     I1 = i1; I2 = i2;
6 }
7
8 void get_seed(unsigned int *i1, unsigned int *i2)
9 {
10    *i1 = I1; *i2 = I2;
11 }
12
13 double unif_rand(void)
14 {
15     I1= 36969*(I1 & 0177777) + (I1>>16);
16     I2= 18000*(I2 & 0177777) + (I2>>16);
17     return ((I1 << 16)^(I2 & 0177777)) * ↵
18         2.328306437080797e-10; /* in [0,1) */
19 }
```

Como se observa se tratan de dos valores que serán las semillas, en esta ocasión tanto podemos modificarlas, como podemos dejarlas intactas, ya que están inicializadas desde la línea 1, los operadores `a<<b`, lo único que hacen es realizar la siguiente operación,  $a \cdot 2^b$ , en cambio `a>>b`, lo que produce es el caso contrario  $a/2^b$ . (Estos operadores se encuentran como desplazadores a izquierda o derecha)

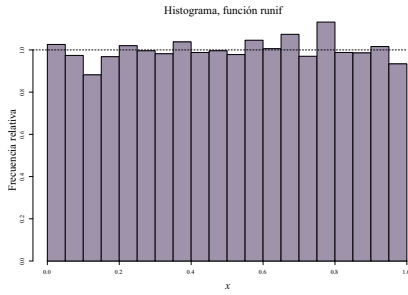
El operador `^`, realiza una operación de bajo nivel, directamente con el procesador, se conoce como el operador XOR, lo que hace es comparar dos cadenas de bits, bit a bit y si

<sup>3</sup>Nuevamente, este código `sunif.c`, fue extraído de las carpetas del usuario SurajGupta, <https://bit.ly/3u79aQ4>

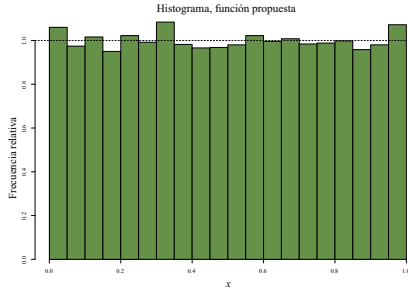
son iguales retorna un 0, si los dos bits de esas cadenas son diferentes, es decir 1 y 0, retorna un 1.

Todo esto se ha mencionado con la finalidad de entender por completo que sucede al llamar esta función en R, de manera que podemos concluir que se trata de un generador de números pseudo aleatorios, ya que se trata de dos semillas muy sencillas y que los datos que se toman son exclusivamente dentro de la computadora.

Sin más dilación, podemos observar en la Figura 1, podemos observar el comportamiento de ambas funciones y como estos valores están cercanos a una distribución uniforme; puesto que los histogramas están en frecuencias relativas, cabe destacar que el número de clases que tiene el histograma lo hace por defecto R y no se hizo uso de la regla de Sturges.



(a) Gráfica obtenida con la función `runif()`



(b) Gráfica obtenida con la función `RANDCN()`

Figura 1. Simulación de códigos

En la Figura 1a, se observa que la función que viene por defecto en la instalación de R, `runif()`, es una buena propuesta para obtener el comportamiento deseado de ser una distribución uniforme.

En el caso de la FP, que se llama como `RANDCN()`, el código lo puede encontrar adjunto al trabajo con el nombre de `PartA.R`, también cumple con ser una propuesta muy buena y si hacemos una comparación visual con respecto a la función `runif()`, se trata de una propuesta bastante buena y una posible opción que podamos usar en el futuro sin la necesidad de llamar funciones se encuentren por defecto, aunque esto no suponga un mejor rendimiento.

En conclusión, ambas propuestas son buenas para generar números pseudo-aleatorios entre 0 y 1, adicionalmente, se destaca que la opción de `runif(n, min = 0, max = 1)`, es en creces mejor, ya que los parámetros de entrada de la función nos permite intercambiar la cota superior e inferior en la cual deseamos interactuar y no tenemos la necesidad de

recurrir en la creación de un código que nos permite realizar eso, aunque como se ha expuesto en el documento, este código no tiene gran misterio.

## II. SIMULACIÓN DE VARIABLES ALEATORIAS DISCRETAS

Para la siguiente sección, se trabajarán con las siguientes funciones:

$$p_{X1}(x) = \begin{cases} k_1 & x = 6, 7, 8, \dots, 15 \\ 0 & \text{e.o.c} \end{cases} \quad (1)$$

$$p_{X2}(x) = \begin{cases} k_2 x & x = 1, 2, 3, \dots, 7 \\ 0 & \text{e.o.c} \end{cases} \quad (2)$$

$$X_3 \sim \text{Bin}(n = 10, p = 7/16)$$

De modo tal que para que ambas funciones de masa de probabilidad existan deben cumplir que su sumatoria, donde su probabilidad es diferente de cero, sea 1. de esta manera tenemos que:

$$\sum_{x=6}^{15} k_1 = 1$$

$$10k_1 = 1 \quad k_1 = \frac{1}{10}$$

De manera análoga se hayan los valores de  $k_2$

$$\sum_{x=1}^7 k_2 x = 1$$

$$k_2 + 2k_2 + 3k_2 + \dots + 7k_2 = 1$$

$$k_2 = \frac{1}{28}$$

Como tenemos las dos primeras funciones de masas de probabilidad, para el tercer caso, la función de masa de probabilidad tiene la forma de:

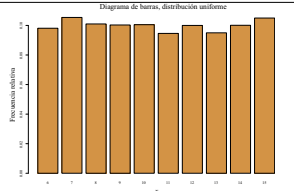
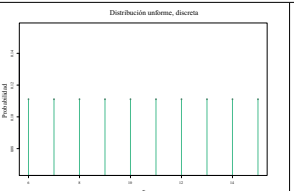
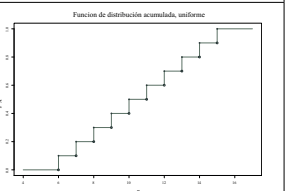
$$p_{X3}(x) = \begin{cases} \binom{10}{x} \left(\frac{7}{16}\right)^x \left(\frac{9}{16}\right)^{10-x} & x = 1, 2, \dots, 10 \\ 0 & \text{e.o.c} \end{cases} \quad (3)$$

Para fines ilustrativos y con la clase, se muestran los siguientes cuadros que tienen como fin, resolver por completo la parte B, del ejercicio. Cabe mencionar, que tanto los graficos, como la forma de obtener cada uno de valores dentro de la función se encuentran en el documento adjunto `PartB.R`.

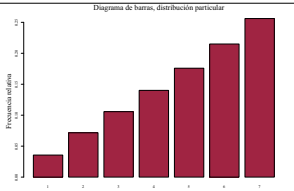
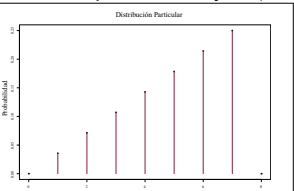
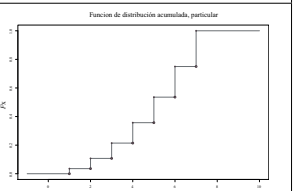
En el caso del Cuadro I, se observan las tres gráficas donde en un total de 10000 simulaciones se obtiene un diagrama de barras que presenta demasiada similitud con la función de masa de probabilidad (PMF), se añade adicionalmente a esta y a los siguientes Cuadros las gráficas de la función de densidad teórica.

De manera similar, se puede observar para el Cuadro II y III, una gran similitud entre el diagrama de barras y su PMF. Sin embargo, hay que tener en cuenta que la similitud entre

**Cuadro I**  
**PRIMERA DISTRIBUCIÓN ASIGNADA**

Función asignada	$p_{X1}(x) = \begin{cases} k_1 & x = 6, 7, 8, \dots, 15 \\ 0 & \text{e.o.c} \end{cases}$		
Función de masa de probabilidad	$p_{X1}(x) = \begin{cases} \frac{1}{10} & x = 6, 7, 8, \dots, 15 \\ 0 & \text{e.o.c} \end{cases}$		
Función de distribución	$F_{X1}(x) = \begin{cases} 0 & x \in (-\infty, 6) \\ 1/10 & x \in [6, 7) \\ 1/5 & x \in [7, 8) \\ 3/10 & x \in [8, 9) \\ 2/5 & x \in [9, 10) \\ 1/2 & x \in [10, 11) \\ 3/5 & x \in [11, 12) \\ 7/10 & x \in [12, 13) \\ 4/5 & x \in [13, 14) \\ 9/10 & x \in [14, 15) \\ 1 & x \in [15, \infty) \end{cases}$		
Resultados (10000 Simulaciones)			

**Cuadro II**  
**SEGUNDA DISTRIBUCIÓN ASIGNADA**

Función asignada	$p_{X2}(x) = \begin{cases} k_2 x & x = 1, 2, 3, \dots, 7 \\ 0 & \text{e.o.c} \end{cases}$		
Función de masa de probabilidad	$p_{X2}(x) = \begin{cases} x/28 & x = 1, 2, 3, \dots, 7 \\ 0 & \text{e.o.c} \end{cases}$		
Función de distribución	$F_{X2}(x) = \begin{cases} 0 & x \in (-\infty, 1) \\ 1/28 & x \in [1, 2) \\ 3/28 & x \in [2, 3) \\ 6/28 & x \in [3, 4) \\ 10/28 & x \in [4, 5) \\ 15/28 & x \in [5, 6) \\ 21/28 & x \in [6, 7) \\ 1 & x \in [7, \infty) \end{cases}$		
Resultados (10000 Simulaciones)			

el diagrama de barras y su distribución no sólo depende de como se haya programado el código en R, sino igualmente la función que genere números aleatorios.

$$f_X(x) = \begin{cases} k_1 x & 0 \leq x < 20 \\ k_2 & 20 \leq x \leq 30 \\ f_x(20^-) = f_x(20) & \\ 0 & \text{e.o.c} \end{cases} \quad (5)$$

### III. SIMULACIÓN DE VARIABLES ALEATORIAS CONTINUAS

Para la siguiente sección se consideran las siguientes funciones:

$$f_X(x) = \begin{cases} -kx & -10 \leq x < 0 \\ k & 0 \leq x \leq 10 \\ 0 & \text{e.o.c} \end{cases} \quad (6)$$

$$X \sim U(a = 7, b = 29/4)$$

(4) Para la segunda función se debe cumplir que:

Cuadro III  
TERCERA DISTRIBUCIÓN ASIGNADA

Función asignada	$X_3 \sim \text{Bin}(n = 10, p = 7/16)$		
Función de masa de probabilidad	$p_{X_3}(x) = \begin{cases} \binom{10}{x} \left(\frac{7}{16}\right)^x \left(\frac{9}{16}\right)^{10-x} & x = 1, 2, \dots, 10 \\ 0 & \text{e.o.c} \end{cases}$		
Función de distribución	$F_{X_3}(x) = \begin{cases} 0 & x \in (-\infty, 1) \\ \frac{13559717115}{549755813888} & x \in [1, 2) \\ \frac{122037454035}{318904458075} & x \in [2, 3) \\ \frac{1099511627776}{586862324685} & x \in [3, 4) \\ \frac{1099511627776}{83695633521} & x \in [4, 5) \\ \frac{1099511627776}{999054302211} & x \in [5, 6) \\ \frac{1099511627776}{1071097843851} & x \in [6, 7) \\ \frac{1099511627776}{136513817937} & x \in [7, 8) \\ \frac{137438953472}{547871184063} & x \in [8, 9) \\ \frac{549755813888}{1} & x \in [9, 10) \\ 1 & x \in [10, \infty) \end{cases} \approx \begin{cases} 0 & x \in (-\infty, 1) \\ 0.0247 & x \in [1, 2) \\ 0.1110 & x \in [2, 3) \\ 0.2900 & x \in [3, 4) \\ 0.5337 & x \in [4, 5) \\ 0.7612 & x \in [5, 6) \\ 0.9086 & x \in [6, 7) \\ 0.9742 & x \in [7, 8) \\ 0.9833 & x \in [8, 9) \\ 0.9966 & x \in [9, 10) \\ 1 & x \in [10, \infty) \end{cases}$		
Resultados (10000 Simulaciones)			

$$\begin{aligned}
 1 &= \int_{-\infty}^{\infty} f_X(x) dx \\
 &= \int_{-\infty}^0 0 dx + \int_0^{20} k_1 x dx + \int_{20}^{30} k_2 dx + \int_{30}^{\infty} 0 dx \\
 &= 200k_1 + 10k_2
 \end{aligned}$$

Además,

#### IV. SIMULACIÓN DE MOMENTOS DE UNA FUNCIÓN DE UNA VARIABLE ALEATORIA

##### IV-A. Ejercicio 1

- a. Empleando la función *rnorm* se generaron 10000 simulaciones de la variable aleatoria  $X \sim N(0.3, \sqrt{0.5})$  y se guardaron en un vector. Una vez hecho esto, se aplicó la función  $e^x$  a las realizaciones generadas, obteniendo las 10000 simulaciones de la variable aleatoria Y. A partir de eso se realizó un histograma para las simulaciones de Y y se superpuso la densidad de una distribución lognormal con parámetros  $\mu = 0.3$  y  $\sigma^2 = 0.5$ , obteniendo lo siguiente (véase, Figura 2):

Esto nos permite concluir que si X es una variable aleatoria con distribución normal, entonces  $Y = e^X$  tendrá una distribución lognormal, con los mismos parámetros de la variable X.

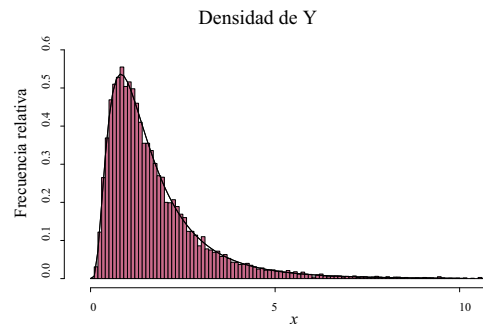


Figura 2. Histograma obtenida después de 10000 simulaciones

- b. Asumiendo que todos los momentos de las variables aleatorias existen, se calcularon los momentos  $E[\sqrt{Y}]$ ,  $E[Y]$  y  $E[Y^3]$ , por el método teórico, es decir usando la ley del estadístico inconsciente. Se tomó la siguiente fórmula para hallar cualquier momento de orden r de Y:

$$E[Y^r] = \exp\left(r\mu + \frac{r^2\sigma^2}{2}\right)$$

Reemplazando para cada uno de los valores pedidos, se

tiene que:

$$E[\sqrt{Y}] = \exp\left(\frac{1}{2}\mu + \frac{(1/2)^2 \sigma^2}{2}\right) \approx 1.237$$

$$E[Y] = \exp\left(\mu + \frac{\sigma^2}{2}\right) \approx 1.733$$

$$E[Y^3] = \exp\left(3\mu + \frac{3^2 \sigma^2}{2}\right) \approx 23.336$$

- c. Para calcular los mismos momentos pedidos anteriormente por medio de aproximaciones por medio de polinomios de Taylor nos presentan la siguiente fórmula:

$$E[Y] = E[g(X)] \approx g[E(X)] + \frac{g''[E(X)] Var(X)}{2}$$

Por lo tanto, si redefinimos la expresión, tenemos que:

$$E[Y^r] = E[g(X)^r] \approx g[E(X)^r] + \frac{g''[E(X)^r] Var(X)}{2}$$

Finalmente, si reemplazamos la formula para cada momento pedido, se tiene que:

$$E[\sqrt{Y}] \approx g[\sqrt{E(X)}] + \frac{g''[\sqrt{E(X)}] Var(X)}{2} \approx 2.161$$

$$E[Y] \approx g[E(X)] + \frac{g''[E(X)] Var(X)}{2} \approx 1.687$$

$$E[Y^3] \approx g\{[E(X)]^3\} + \frac{g''[E(X)^3] Var(X)}{2} \approx 1.284$$

- d. Para calcular los mismos momentos anteriores a través de aproximación numérica, se tiene en cuenta la siguiente formula:

$$E[Y] \approx \frac{1}{r} \sum_{i=1}^r g(x_i)$$

A partir de esto, con la función `rnorm()` se hicieron las correspondientes simulaciones de X en función del valor de r y así obtener los valores de los momentos de Y, resumidos en el siguiente Cuadro (véase, Cuadro IV).

Cuadro IV  
VALORES OBTNIDOS CON `RNORM()`

r	$E[\sqrt{Y}]$	$E[Y]$	$E[Y^3]$
$10^1$	1.187177	1.748383	23.40700
$10^2$	1.246916	1.730593	20.66907
$10^3$	1.222406	1.700345	28.33751
$10^4$	1.237060	1.736511	23.81004

- e. Por medio del Cuadro V, se compararan los resultados obtenidos en cada método descrito anteriormente.

Cuadro V  
COMPARACIÓN CON MÉTODOS

Valor teórico	Método 2	Err % <sub>2</sub>	Método 3 (r = 10000)	Err % <sub>3</sub>
1.236766	2.161638	42.79	1.237060	0.02
1.733253	1.687324	2.72	1.736511	0.19
23.33607	1.284210	1717.15	23.81004	1.99

A partir de esto, se puede ver que el método 2 no resulta tan efectivo a la hora de aproximar el valor de

los momentos de Y y esto se da ya que, los polinomios de Taylor al usar derivadas, buscar linealizar una función, sin embargo, es claro que el crecimiento de  $e^x$  es bastante acelerado, causando que la linealización no se ajuste al ritmo de dicha función, teniendo como consecuencia que la aproximación al valor teórico de los momentos no sea buena y fallé teniendo una gran distancia entre el valor real en comparación con el aproximado.

Por otro lado, la aproximación numérica por simulación tiende a ser efectiva a medida que el r es más grande. Esto se tiene, ya que la formula usada en este método es un promedio de las realizaciones de Y, por lo tanto a mas grande el valor de r, mas precisa se vuelve la aproximación al valor teórico.

#### IV-B. Ejercicio 2

- a. Se tiene que la siguiente función representa la demanda de empanadas en un negocio convencional que vende los domingos en la iclovía

$$p_{X2}(x) = \begin{cases} \frac{k\sqrt{x}}{\theta + 1 + x} & x = 1, 2, \dots, 100 \\ 0 & \text{e.o.c} \end{cases}$$

Para hallar el valor de k, se requiere realizar una suma en todos el soporte de  $D_X = \{1, 2, \dots, 100\}$ , siendo  $\theta = 7$ , el cuál tiene la siguiente forma:

$$\sum_{x=1}^{100} \frac{k\sqrt{x}}{8+x} = 1$$

$$k \frac{\sqrt{1}}{9} + k \frac{\sqrt{2}}{10} + \dots + k \frac{\sqrt{100}}{108} = 1$$

$$k \left( \frac{\sqrt{1}}{9} + \frac{\sqrt{2}}{10} + \dots + \frac{\sqrt{100}}{108} \right) = 1$$

$$k = \left( \frac{\sqrt{1}}{9} + \frac{\sqrt{2}}{10} + \dots + \frac{\sqrt{100}}{108} \right)^{-1}$$

$$k \approx 0.0787762$$

De este modo se tiene que la función de masa de probabilidad es:

$$p_{X2}(x) = \begin{cases} \frac{0.0787762\sqrt{x}}{8+x} & x = 1, 2, \dots, 100 \\ 0 & \text{e.o.c} \end{cases} \quad (7)$$

- b. Se realizó el siguiente código en R, con el fin de que pueda emular los valores de la variable aleatoria, se tiene que la función de masa de probabilidad y la de función de distribución acumulada, las gráficas respectivas<sup>4</sup> (véase, Figura 3).

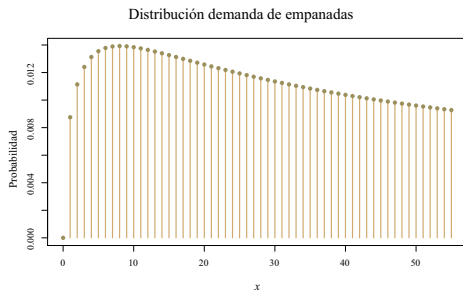
```
1 p_X2<-function(x) {
2   stopifnot(is.numeric(x))
3   if (x %in% c(0:100)) {
4     (0.0787762*sqrt(x))/(8+x)
5   }
}
```

<sup>4</sup>El código para exportar las gráficas de PMF y CDF se encuentra en el documento adjunto `PartD.R`

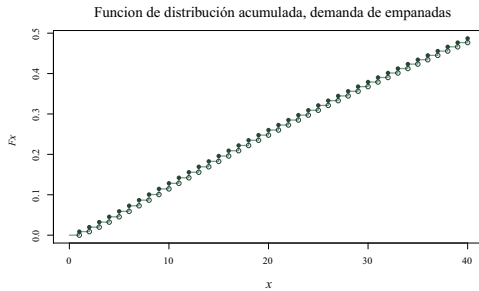
```

6   else{
7     0
8   }
9 }
10
11 F_X2<-function(x){
12   stopifnot(is.numeric(x))
13   a <- as.numeric(cut(x, c(-Inf,1:100,Inf), <-
14     right=FALSE))
15   n<-0
16   if (x>=1){
17     if (x>=100){
18       1
19     }
20     else{
21       for(i in 1:x){
22         n<-n+p_X2(i)
23       }
24       return(n)
25     }
26   }
27   else
28     0
29 }

```



(a) Función de masa de probabilidad, demanda de empanadas



(b) Función de distribución, demandas de empanadas

Figura 3. Gráficas de funciones de la demanda de empanadas

Una vez se tienen la función de masa de probabilidad y la función de distribución acumulada, podemos crear una función que nos retorne valores de la variable aleatoria, hacer el trabajo inverso. Esto es análogo al trabajo que se realizó en la parte B, con los números pseudo-aleatorios.

Con un programa de número pseudo-aleatorios podemos modelar la distribución y observar que se respeta su distribución después de 10000 simulaciones (véase, Figura 4), la forma es muy parecida a la función de masa de probabilidad en la Figura 3a, si se requiere una mayor resolución se debe aumentar el número de simulaciones, pero esto implicará que el computador, valga la redundancia, compute muchas más veces y el

tiempo de proceso se alargue, puesto que en esta ocasión la propuesta no es del todo recursiva.

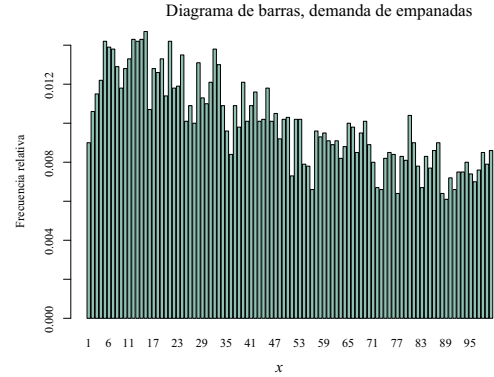


Figura 4. Histograma obtenida después de 10000 simulaciones

En el trabajo, la función que simulara los valores de la variable aleatoria se encuentran en el archivo adjunto PartD.R

- c. La ganancia de empanadas  $G(x, y)$  será dependiente de dos variables, la demanda el día domingo  $x$  y la cantidad de empanadas que se hacen el día anterior  $y$ . También esta función está dada por 3 parámetros, los cuales son: Valor de venta  $\eta(x, y)$ , el costo de producción  $\psi(x, y)$  y su reventa a una fundación  $\lambda(x, y)$ . Todo esto está denotado de la siguiente forma:

$$G(x, y) := \eta(x, y) + \lambda(x, y) - \psi(x, y)$$

Donde:

$$\begin{aligned} \eta : U \subseteq \mathbb{R}^2 &\longrightarrow V \subseteq \mathbb{R} \\ (x, y) &\longmapsto \eta(x, y) := 1500 \min\{x, y\} \end{aligned}$$

$$\begin{aligned} \lambda : U \subseteq \mathbb{R}^2 &\longrightarrow V \subseteq \mathbb{R} \\ (x, y) &\longmapsto \lambda(x, y) := \begin{cases} 600(y - x) & y > x \\ 0 & \text{e.o.c} \end{cases} \end{aligned}$$

$$\begin{aligned} \psi : U \subseteq \mathbb{R}^2 &\longrightarrow V \subseteq \mathbb{R} \\ (x, y) &\longmapsto \psi(x, y) := 900y \end{aligned}$$

El siguiente algoritmo, código en R, nos sirve para realizar el compute necesario en el Cuadro VI.

```

1 MinE<-function(x, y){
2   stopifnot(is.numeric(x))
3   stopifnot(is.numeric(y))
4   if (x>=y){return(y)}
5   else{return(x)}
6 }
7 Cost<-function(x, y){
8   stopifnot(is.numeric(x))
9   stopifnot(is.numeric(y))
10  if (y>x){return(y-x)}
11  else{return(0)}
12 }

```

```

13
14 G_X2<-function(x,y) {
15   stopifnot(is.numeric(x))
16   stopifnot(is.numeric(y))
17   1500*MinE(x,y)+600*Cost(x,y)-900*y
18 }

```

Cuadro VI  
GANANCIAS QUE RECIBIRÍA EL NEGOCIO

Ganancia	$x = 10$	$x = 50$	$x = 100$
$q = 10$	6000	6000	6000
$q = 60$	-9000	27000	36000
$q = 90$	-18000	18000	54000

- d. Se halla el valor esperado de la demanda de empanadas, denotado como  $E[X]$

$$\begin{aligned}
 E[X] &= \sum_{x=1}^{100} \frac{k x \sqrt{x}}{8+x} \\
 &= k \frac{\sqrt{1}}{9} + k \frac{2\sqrt{2}}{10} + \cdots + k \frac{100\sqrt{100}}{108} \\
 &= k \left( \frac{\sqrt{1}}{9} + \frac{2\sqrt{2}}{10} + \cdots + \frac{100\sqrt{100}}{108} \right) \\
 k &= 0.0787762 \\
 E[X] &\approx 44.8953
 \end{aligned}$$

Si el valor esperado es 44.8953, se realizarán 10000 simulaciones para ver como se comporta la ganancia debido a que la forma no es del todo similar a la teórica con 1000 simulaciones.

Se tiene que  $y = 44$  Empanadas a preparar, la siguiente figura (véase, Figura 5) muestra el comportamiento de la cantidad de dinero que se ganará:

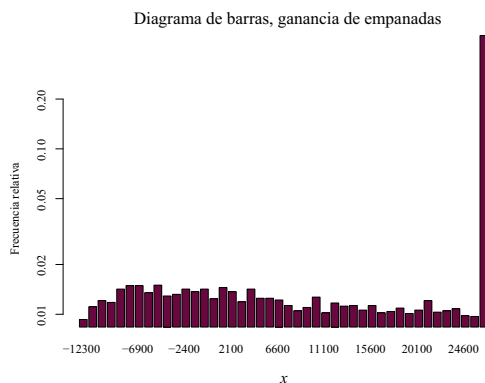


Figura 5. Histograma obtenida después de 10000 simulaciones

Como se observa en la Figura 5, cerca del 60% de los domingos que se venda, se ganará \$26400. Sin embargo, contamos con el evento posible de devolvemos a casa con una venta cercana de 12300, a pesar de que sea esta probabilidad menor del 0.01. También cabe destacar que el valor esperado de ganancia en este diagrama de barras (véase, Figura 5) es de \$15695.58, lo cual en buena parte

## V. APROXIMACIÓN DE INTEGRALES DEFINIDAS MEDIANTE SIMULACIÓN

### APÉNDICE A

#### DÓNDE SE ENCUENTRAN LOS ARCHIVOS ADJUNTOS

En el trabajo, se encontrará adjunto a él, un archivo `source.zip`, Dentro de él las carpetas que se encontrarán son: `Codes`, `Imagenes`, `Tablas` y las secciones específicas del documento.

Para compilar el documento debe seleccionar como archivo principal, el script llamado `00 main.tex`, allí se encontrarán el resto de códigos indexados. Para encontrar los códigos que se usaron en la elaboración del proyecto debe seguir la siguiente dirección `.../Codes/Codes/`, en ella se encontrarán los códigos del nombre con los cuales se hacen mención dentro del documento escrito y secciones de algunas partes para mostrarlas de manera parcial.

### REFERENCIAS

- [1] RDocumentation. (2020) Uniform: The uniform distribution. [Online]. Available: <https://bit.ly/3u6TzAf>
- [2] The R Core Team, *R: A Language and Environment for Statistical Computing*, The R Core Team.