Parte II Guias Practicas de R

Lilian Martínez

November 26, 2015

```r
#Práctica 10-Análisis de una variable bidimensional (categórica, continua)

#EJEMPLO 1
#Crea un vector de datos para cada proceso descrito en el problema.
A <- c(100,96,92,96,92); A
```

```
## [1] 100  96  92  96  92
```

```r
B <- c(76,80,75,84,82); B
```

```
## [1] 76 80 75 84 82
```

```r
C <- c(108,100,96,98,100); C
```

```
## [1] 108 100  96  98 100
```

```r
#Crea una hoja de datos teniendo como componentes (columnas) los tres vectores
Baterias <- data.frame(procesoA=A, procesoB=B, procesoC=C); Baterias
```

```
##   procesoA procesoB procesoC
## 1      100       76      108
## 2       96       80      100
## 3       92       75       96
## 4       96       84       98
## 5       92       82      100
```

```r
# Para editar los datos puede utilizar la función fix()
fix(Baterias)

#Guarda la hoja de datos en un archivo.
write.table(Baterias, file="Baterias.txt", append=FALSE, quote=TRUE, sep=" ",
na="NA", col.names=TRUE)

#Elimina todos objetos que existen en el espacio de trabajo (Workspace)
ls(); rm(list=ls(all=TRUE)); ls()
```

```
## [1] "A"         "B"         "Baterias" "C"
## character(0)
```

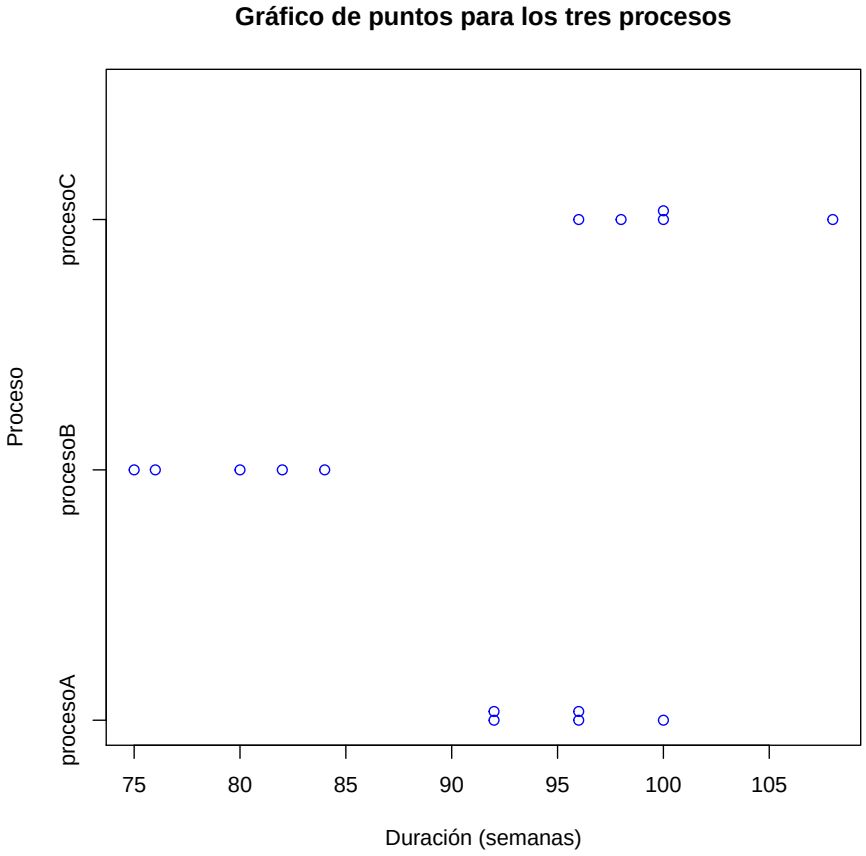
```r
#Recupera la hoja de datos,para probar si fue guardada.
Baterias <- read.table("Baterias.txt", header=TRUE); Baterias
```

```
##   procesoA procesoB procesoC
## 1      100       76      108
## 2       96       80      100
## 3       92       75       96
## 4       96       84       98
## 5       92       82      100
```

```r
#Conecta o adjunta la hoja de datos a la segunda ruta o lista de búsqueda.
attach(Baterias, pos=2)
search()
```

```
##  [1] ".GlobalEnv"        "Baterias"          "package:knitr"
##  [4] "package:stats"     "package:graphics"  "package:grDevices"
##  [7] "package:utils"     "package:datasets"  "package:methods"
## [10] "Autoloads"         "package:base"
```

```
#Dibuja un gráfico horizontal depuntos para los tres procesos.
stripchart(Baterias, main="Gráfico de puntos para los tres procesos",
method = "stack", vertical = FALSE, col="blue", pch=1, xlab="Duración (semanas)", ylab="Pr
```

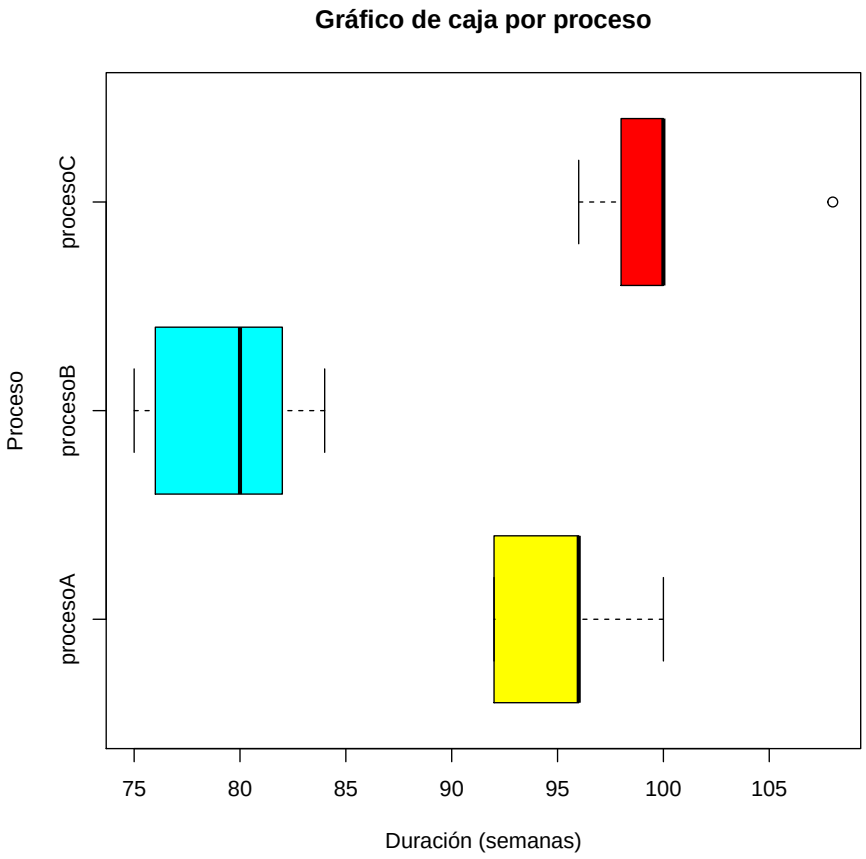



```
#Muestra un resumen estadístico para los tres procesos.
summary(Baterias)
```

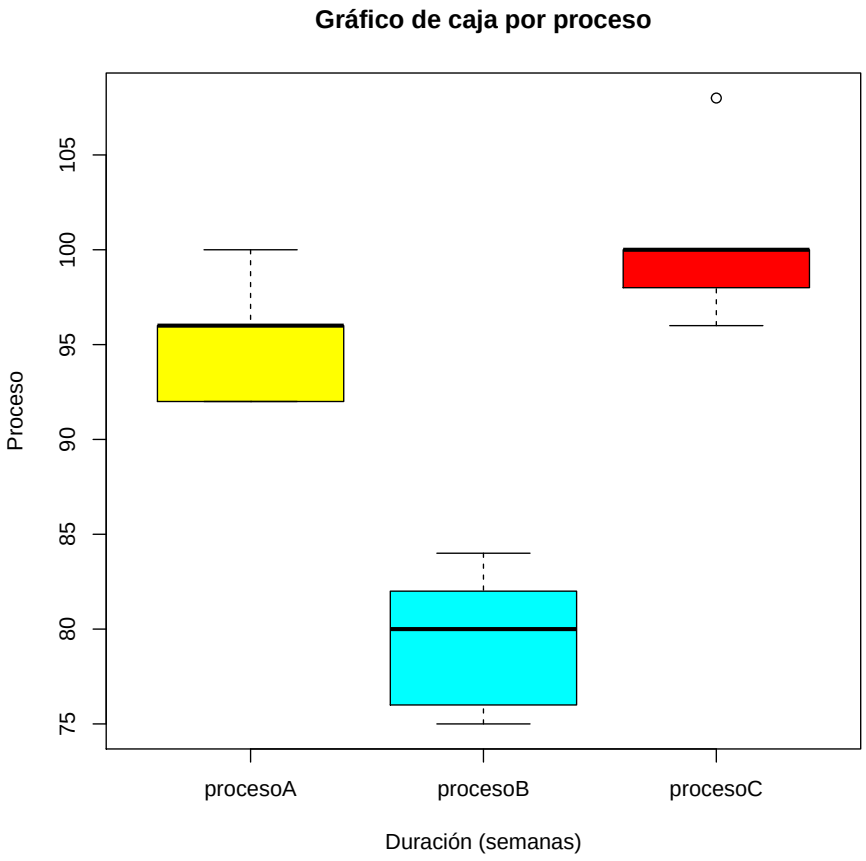
```
##     procesoA        procesoB       procesoC
##  Min.   : 92.0   Min.   :75.0   Min.   : 96.0
##  1st Qu.: 92.0   1st Qu.:76.0   1st Qu.: 98.0
##  Median : 96.0   Median :80.0   Median :100.0
##  Mean   : 95.2   Mean   :79.4   Mean   :100.4
##  3rd Qu.: 96.0   3rd Qu.:82.0   3rd Qu.:100.0
##  Max.   :100.0   Max.   :84.0   Max.   :108.0
```

```
#Dibuja un gráfico horizontal de cajas (box-plot) para los tres procesos.
```

```r
boxplot(Baterias, width=NULL, varwidth=TRUE,names, add= FALSE, horizontal = TRUE,
main="Gráfico de caja por proceso", border=par("fg"), col=c("yellow", "cyan",
"red"), xlab = "Duración (semanas)", ylab="Proceso")
```




```r
# Vertical
boxplot(Baterias, width=NULL, varwidth=TRUE,names, add= FALSE, horizontal = FALSE,
main="Gráfico de caja por proceso", border=par("fg"), col=c("yellow", "cyan",
"red"), xlab = "Duración (semanas)", ylab="Proceso")
```




```
#Presenta la matriz de covarianzas muestral.
options(digits=3)
S <- var(Baterias); S
```

```
##          procesoA procesoB procesoC
## procesoA     11.2     -1.6     12.4
## procesoB     -1.6     14.8     -4.7
## procesoC     12.4     -4.7     20.8
```

```
# Concatena los tres vectores dentro de un vector simple, junto con un vector factor indic
#la categoría o tratamiento (A, B, C) que origina cada observación.
Baterias <- stack(Baterias); Baterias
```

```
##    values      ind
## 1     100 procesoA
## 2      96 procesoA
## 3      92 procesoA
## 4      96 procesoA
## 5      92 procesoA
## 6      76 procesoB
```

```
## 7      80 procesoB
## 8      75 procesoB
## 9      84 procesoB
## 10     82 procesoB
## 11    108 procesoC
## 12    100 procesoC
## 13     96 procesoC
## 14     98 procesoC
## 15    100 procesoC
```

```
# Prueba de igualdad de medias por descomposición de la varianza en dos fuentes de variaci
aov.Baterias <- aov(values~ind, data=Baterias)

# Prueba de igualdad de medias en un diseño de una vía
oneway.test(values~ind, data=Baterias, var.equal = TRUE)
```

```
##
##  One-way analysis of means
##
## data:  values and ind
## F = 40, num df = 2, denom df = 10, p-value = 6e-06
```

```
#Deshace la concatenación del vector de valores y el vector indicador de categoría.
Baterias = unstack(Baterias);Baterias
```

```
##   procesoA procesoB procesoC
## 1      100       76      108
## 2       96       80      100
## 3       92       75       96
## 4       96       84       98
## 5       92       82      100
```

```
#Desconecta la hoja de datos de la segunda ruta o lista de búsqueda.
detach(Baterias, pos=2); search()
```

```
##  [1] ".GlobalEnv"        "package:knitr"     "package:stats"
##  [4] "package:graphics"  "package:grDevices" "package:utils"
##  [7] "package:datasets"  "package:methods"   "Autoloads"
## [10] "package:base"
```

```
#UNIDAD 2: Práctica 10-Análisis de una variable bidimensional (categórica, continua)
#EJEMPLO 2 Suponga que un estudiante hace una encuesta paraevaluar sí los
#estudiantes que fuman estudian menos que los que no fuman.

#ANÁLISIS ESTADÍSTICO DE LOS DATOS

#Crea dos vectores con los datos.
Fuma = c("Si","No","No","Si","No","Si","Si","Si","No","Si"); Fuma
```

```
##  [1] "Si" "No" "No" "Si" "No" "Si" "Si" "Si" "No" "Si"
```

```
Cantidad = c(1,2,2,3,3,1,2,1,3,2); Cantidad
```

```
##  [1] 1 2 2 3 3 1 2 1 3 2

#Crea una hoja de datos que tenga comocomponentes o columnas los dos vectores.
Estudia <- data.frame(Fuma=Fuma, Cantidad=Cantidad); Estudia

##    Fuma Cantidad
## 1    Si        1
## 2    No        2
## 3    No        2
## 4    Si        3
## 5    No        3
## 6    Si        1
## 7    Si        2
## 8    Si        1
## 9    No        3
## 10   Si        2

# Puedes editar los datos utilizando
fix(Estudia)

#Guarda la hoja de datos en un archivo.
write.table(Estudia, file="Estudia.txt", append=FALSE, quote=TRUE, sep=" ", na="NA",col.na


#Elimina los objetos almacenados enel área de trabajo (Workspace).
ls()

## [1] "aov.Baterias" "Baterias"     "Cantidad"     "Estudia"
## [5] "Fuma"         "S"

rm(list=ls(all=TRUE))
ls()

## character(0)

#Recupera desde el archivo la hoja de datos
Estudia <- read.table("Estudia.txt", header=TRUE)
Estudia

##    Fuma Cantidad
## 1    Si        1
## 2    No        2
## 3    No        2
## 4    Si        3
## 5    No        3
## 6    Si        1
## 7    Si        2
## 8    Si        1
## 9    No        3
## 10   Si        2
```

```r
#Conecta la hoja de datos a la segunda ruta o lista de búsqueda,
attach(Estudia, pos=2)
search()
```

```
##  [1] ".GlobalEnv"        "Estudia"           "package:knitr"
##  [4] "package:stats"     "package:graphics"  "package:grDevices"
##  [7] "package:utils"     "package:datasets"  "package:methods"
## [10] "Autoloads"         "package:base"
```

```r
#Crea una tabla de contigencia o de doble entrada.
tablaCont <- table(Estudia)
tablaCont
```

```
##     Cantidad
## Fuma 1 2 3
##   No 0 2 2
##   Si 3 2 1
```

```r
#Calcula las tablas de proporciones o de probabilidades.
options(digits=3)

# Proporciones basadas en el total de la muestra, la suma de filas y columnas suman 1
propTotal <- prop.table(tablaCont); propTotal
```

```
##     Cantidad
## Fuma   1   2   3
##   No 0.0 0.2 0.2
##   Si 0.3 0.2 0.1
```

```r
# Proporciones basadas en el total por fila, cada fila suma 1
propFila <- prop.table(tablaCont, 1)
propFila
```

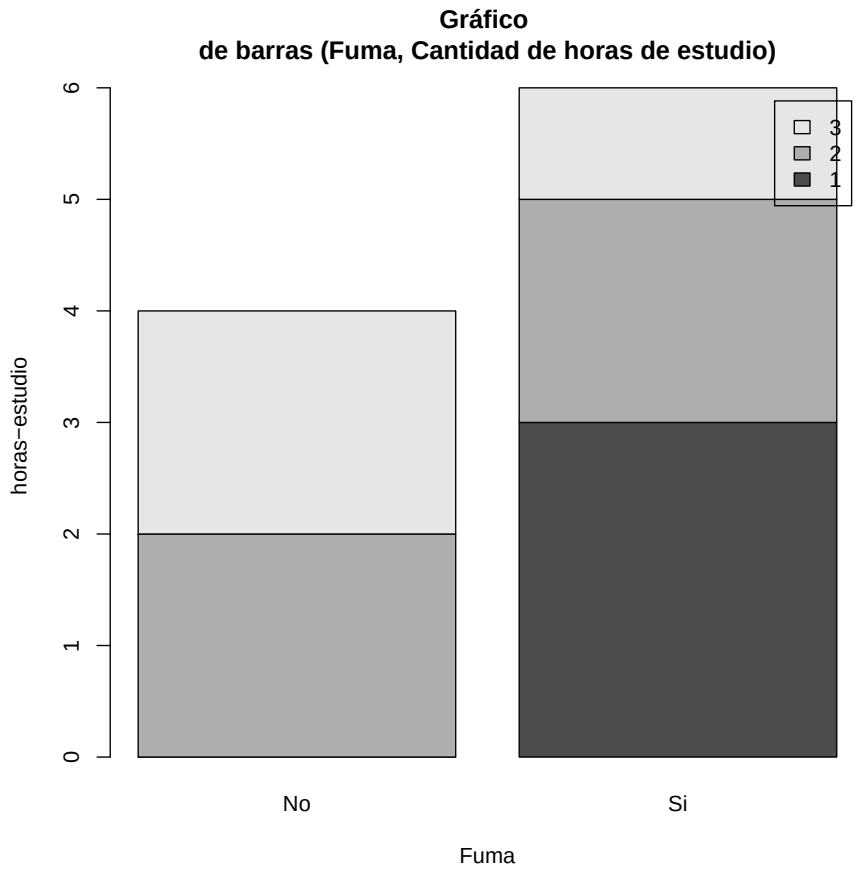
```
##     Cantidad
## Fuma     1     2     3
##   No 0.000 0.500 0.500
##   Si 0.500 0.333 0.167
```

```r
# Proporciones basadas en el total por columna, cada columna suma 1
propCol <- prop.table(tablaCont, 2)
propCol
```

```
##     Cantidad
## Fuma     1     2     3
##   No 0.000 0.500 0.667
##   Si 1.000 0.500 0.333
```

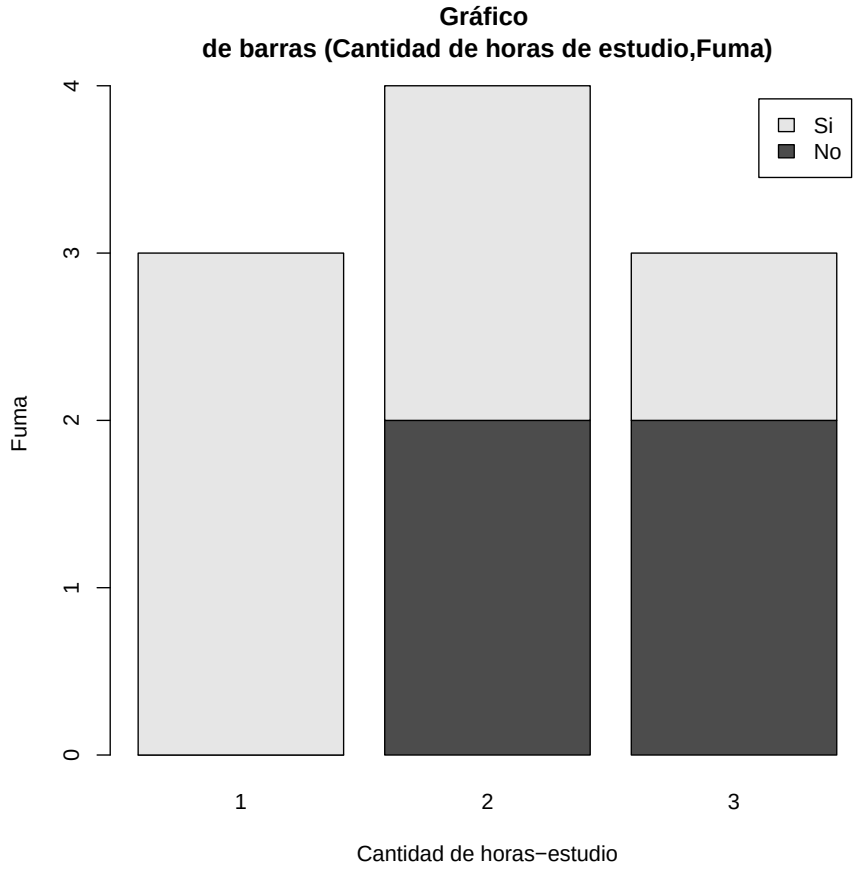
```r
#Construya los gráficos de barras de la variable bidimensional.
# Gráfico de barras apiladas con la frecuencia de Cantidad como altura
barplot(table(Estudia$Cantidad, Estudia$Fuma), beside = FALSE, horizontal=FALSE, main="Grá
de barras (Fuma, Cantidad de horas de estudio)", legend.text =T, xlab="Fuma", ylab="Cantid
horas-estudio")
```

```
## Warning in plot.window(xlim, ylim, log = log, ...):  "horizontal"
is not a graphical parameter
## Warning in axis(if (horiz) 2 else 1, at = at.l, labels = names.arg,
lty = axis.lty, :  "horizontal" is not a graphical parameter
## Warning in title(main = main, sub = sub, xlab = xlab, ylab = ylab,
...):  "horizontal" is not a graphical parameter
## Warning in axis(if (horiz) 1 else 2, cex.axis = cex.axis, ...):
"horizontal" is not a graphical parameter
```

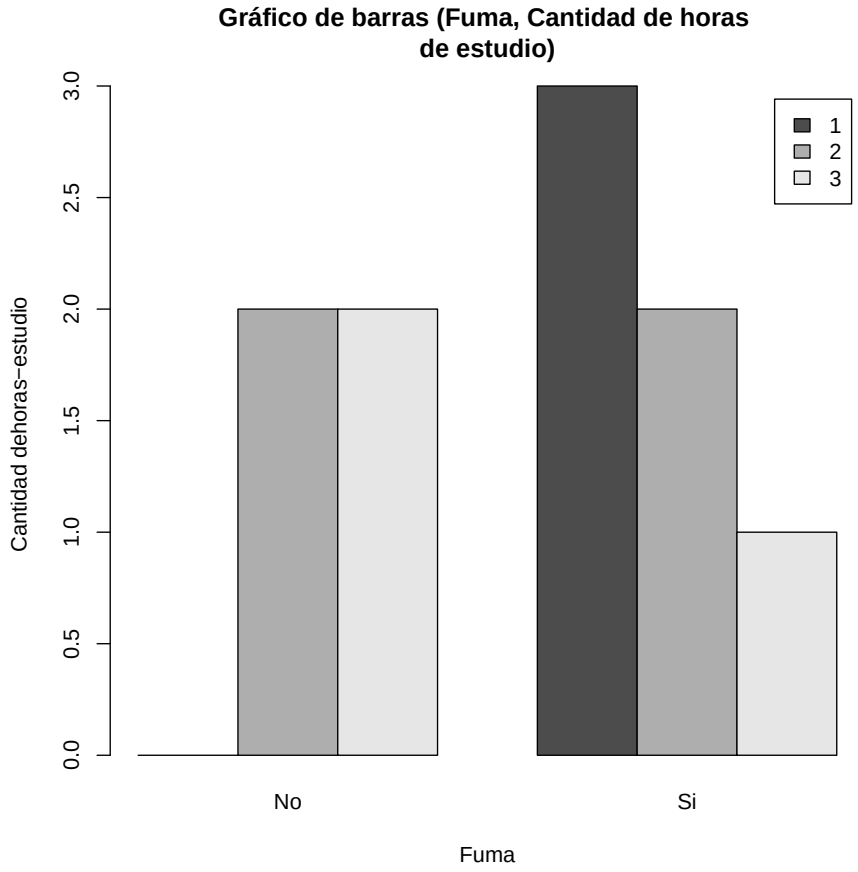



```
# Gráfico de barras apiladas con la frecuencia de Fuma como altura
barplot(table(Estudia$Fuma, Estudia$Cantidad), beside = FALSE, horizontal=FALSE,main="Gráf
de barras (Cantidad de horas de estudio,Fuma)", legend.text =T, xlab="Cantidad de horas-es
ylab="Fuma")

## Warning in plot.window(xlim, ylim, log = log, ...):  "horizontal"
is not a graphical parameter
## Warning in axis(if (horiz) 2 else 1, at = at.l, labels = names.arg,
lty = axis.lty, :  "horizontal" is not a graphical parameter
## Warning in title(main = main, sub = sub, xlab = xlab, ylab = ylab,
...):  "horizontal" is not a graphical parameter
```

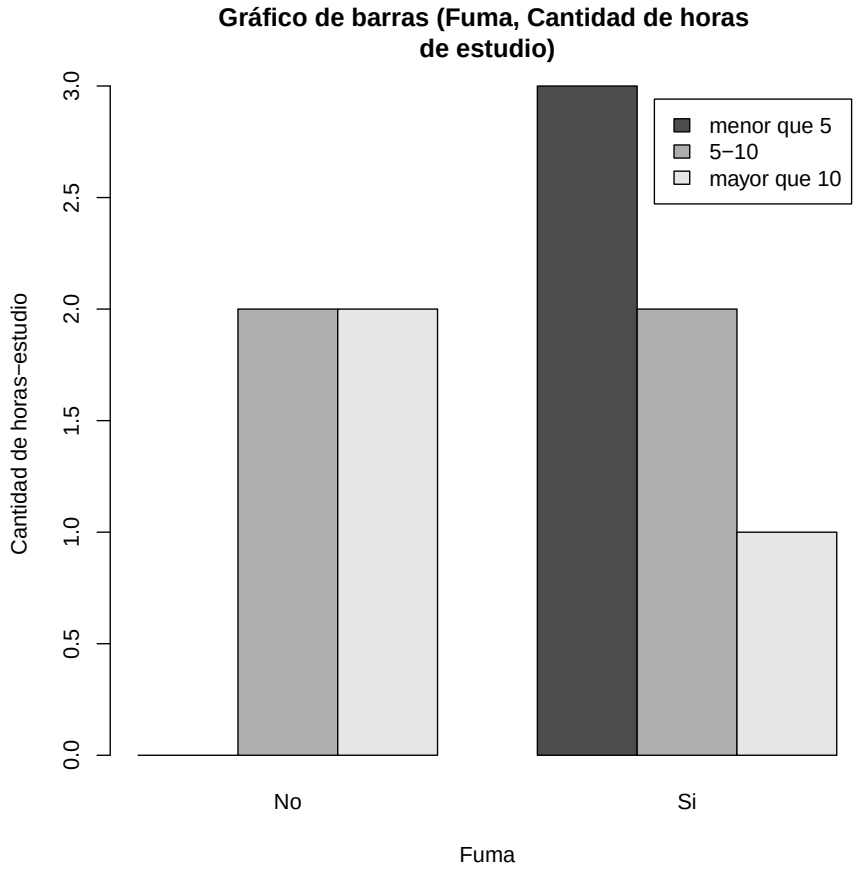
```
## Warning in axis(if (horiz) 1 else 2, cex.axis = cex.axis, ...):
"horizontal" is not a graphical parameter
```




```
# Gráfico de barras no apiladas y colocación de leyenda
# Crear un factor para los nombres en la leyenda
Fuma=factor(Estudia$Fuma); Fuma

##  [1] Si No No Si No Si Si Si No Si
## Levels: No Si

barplot(table(Estudia$Cantidad, Estudia$Fuma), main="Gráfico de barras (Fuma, Cantidad de
de estudio)", xlab="Fuma", ylab="Cantidad dehoras-estudio", beside=TRUE, legend.text=T)
```




```
barplot(table(Estudia$Cantidad, Estudia$Fuma), main="Gráfico de barras (Fuma, Cantidad de
de estudio)", xlab="Fuma", ylab="Cantidad de horas-estudio", beside=TRUE,
legend.text=c("menor que 5", "5-10", "mayor que 10"))
```




```
# Probabilidades esperadas para la prueba Chi-cuadrada
chisq.test(tablaCont) $expected
```

```
## Warning in chisq.test(tablaCont):  Chi-squared approximation may
be incorrect
```

```
##     Cantidad
## Fuma   1   2   3
##   No 1.2 1.6 1.2
##   Si 1.8 2.4 1.8
```

```
#UNIDAD 3: Práctica 13 - Espacios muestrales

#GENERACIÓN DE ESPACIOS MUESTRALES Y DE MUESTRAS ALEATORIAS.

#Simular 10 lanzamientos de una moneda
# vector del cual se tomará la muestra
moneda <- c("C", "+"); moneda
```

```
## [1] "C" "+"
```

```r
# tamaño de la muestra
n <- 10; n

## [1] 10

#generando la muestra aleatoria con reemplazamiento
lanzamientos <- sample(moneda, n, replace=TRUE); lanzamientos

##  [1] "+" "C" "C" "C" "+" "C" "C" "C" "C" "C"

#Elegir 6 números de una lotería de 54 números
# se define el espacio muestral del cual se tomará la muestra
espacio <- 1:54;espacio

##  [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23
## [24] 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46
## [47] 47 48 49 50 51 52 53 54

# se define el tamaño de la muestra
n <- 6; n

## [1] 6

#seleccionando la muestra sin reposición
muestra <- sample(espacio, n); muestra

## [1] 13 46 27  3 40 20

#Simular 4 lanzamientos de dos dados
espacio = as.vector(outer(1:6, 1:6, paste)); espacio

##  [1] "1 1" "2 1" "3 1" "4 1" "5 1" "6 1" "1 2" "2 2" "3 2" "4 2" "5 2"
## [12] "6 2" "1 3" "2 3" "3 3" "4 3" "5 3" "6 3" "1 4" "2 4" "3 4" "4 4"
## [23] "5 4" "6 4" "1 5" "2 5" "3 5" "4 5" "5 5" "6 5" "1 6" "2 6" "3 6"
## [34] "4 6" "5 6" "6 6"

# se define el tamaño de la muestra
n <- 4; n

## [1] 4

# finalmente se selecciona la muestra
muestra <- sample(espacio, n, replace=TRUE); muestra

## [1] "3 3" "3 3" "1 4" "4 5"

#Seleccionar cinco cartas de un naipe de 52 cartas
naipe = paste(rep(c("A", 2:10, "J", "Q", "K"), 4),
c("OROS","COPAS", "BASTOS","ESPADAS"));naipe
```

```
##  [1] "A OROS"     "2 COPAS"    "3 BASTOS"   "4 ESPADAS"  "5 OROS"
##  [6] "6 COPAS"    "7 BASTOS"   "8 ESPADAS"  "9 OROS"     "10 COPAS"
## [11] "J BASTOS"   "Q ESPADAS"  "K OROS"     "A COPAS"    "2 BASTOS"
## [16] "3 ESPADAS"  "4 OROS"     "5 COPAS"    "6 BASTOS"   "7 ESPADAS"
## [21] "8 OROS"     "9 COPAS"    "10 BASTOS"  "J ESPADAS"  "Q OROS"
## [26] "K COPAS"    "A BASTOS"   "2 ESPADAS"  "3 OROS"     "4 COPAS"
## [31] "5 BASTOS"   "6 ESPADAS"  "7 OROS"     "8 COPAS"    "9 BASTOS"
## [36] "10 ESPADAS" "J OROS"     "Q COPAS"    "K BASTOS"   "A ESPADAS"
## [41] "2 OROS"     "3 COPAS"    "4 BASTOS"   "5 ESPADAS"  "6 OROS"
## [46] "7 COPAS"    "8 BASTOS"   "9 ESPADAS"  "10 OROS"    "J COPAS"
## [51] "Q BASTOS"   "K ESPADAS"
```

```
# se define el tamaño de la muestra
n <- 5; n
```

```
## [1] 5
```

```
# se obtiene la muestra sin reemplazo (aunque no se especifique con replace=FALSE)
cartas <- sample(naipe, n) ; cartas
```

```
## [1] "A ESPADAS" "9 COPAS"   "A COPAS"   "8 BASTOS"  "3 ESPADAS"
```

```
#Generar una muestra aleatoria de tamaño 120,con los números del 1 al 6 en el que las prob
sample(1:6,120,replace=TRUE, c(0.5,0.25,0.15,0.04,0.03,0.03))
```

```
##   [1] 2 3 1 1 2 1 1 3 5 1 1 2 1 1 1 1 1 3 1 4 1 3 1 6 3 1 1 2 6 3 1 2 2 3 4
##  [36] 1 4 6 5 1 3 3 3 1 1 2 2 1 3 1 4 2 4 1 3 1 2 1 2 2 3 3 1 1 2 3 1 2 6 1
##  [71] 2 3 2 1 1 1 2 1 1 1 1 3 1 2 1 2 1 1 2 1 1 3 2 2 1 2 5 2 1 2 1 1 2 3 3
## [106] 4 3 1 2 1 1 1 3 1 1 2 1 2 6 1
```

```
#Escriba una función que reciba los números enteros entre 1 y 500 inclusive, la función re
#espacio formado por los números divisibles entre 7.Después de llamar a esta función se ex
#aleatoriamente 12 de estos números, con reemplazo.
```

```
# definiendo la función que generará el espacio formado
espacio <- function(num)
{
 numDiv7 <- numeric(0)
 ind <- 0
 for(i in 1:length(num))
   if ((num[i] %% 7)==0)
    {
     ind <- ind+1
     numDiv7[ind]=num[i]
    }
 return(numDiv7)
}
numeros <- 1:500
espacio
```

```
## function(num)
## {
##  numDiv7 <- numeric(0)
##  ind <- 0
##  for(i in 1:length(num))
##    if ((num[i] %% 7)==0)
##     {
##      ind <- ind+1
##      numDiv7[ind]=num[i]
##     }
##  return(numDiv7)
## }

# generando el espacio muestral
s <- espacio(numeros); s
```

```
##  [1]   7  14  21  28  35  42  49  56  63  70  77  84  91  98 105 112 119
## [18] 126 133 140 147 154 161 168 175 182 189 196 203 210 217 224 231 238
## [35] 245 252 259 266 273 280 287 294 301 308 315 322 329 336 343 350 357
## [52] 364 371 378 385 392 399 406 413 420 427 434 441 448 455 462 469 476
## [69] 483 490 497
```

```
# seleccionando la muestra
muestra <- sample(s, 12, replace=TRUE); muestra
```

```
##  [1] 105 133 350 161  49 196 315 273 182 224 469 483
```

```
#UNIDAD 3: Práctica 14 Distribuciones de probabilidad discreta
#CÁLCULO DE PROBABILIDADES.

#Ejemplo 1:
#Si un estudiante responde al azar a un examen de 8 preguntas de verdadero o falso.
#a) ¿Cuál es la probabilidad de que acierte 4?
dbinom(4,8,0.5)
```

```
## [1] 0.273
```

```
#b) ¿Cuál es la probabilidad de que acierte a lo sumo 2?
x <- 2; n=8; p=1/2
pbinom(x, size = n, prob = p, lower.tail=TRUE)
```

```
## [1] 0.145
```

```
#c) ¿Cuál es la probabilidad de que acierte 5 o más?
x <- 4; n=8; p=1/2

#primera forma
F <- 1 - pbinom(x, n, p, lower.tail=TRUE); F
```

```
## [1] 0.363
```

```r
#segunda forma
pbinom(4, size=8, prob=0.5, lower.tail=FALSE)
```

```
## [1] 0.363
```

```r
#Ejemplo 2:
#Una cierta área de Estados Unidos es afectada, en promedio, por 6 huracanes al año.
#Encuentre la probabilidad de que en un determinado año esta área sea afectada por:

#a) Menos de 4 huracanes.
x <- 3; mu <- 6
ppois(x, lambda = mu, lower.tail=TRUE)
```

```
## [1] 0.151
```

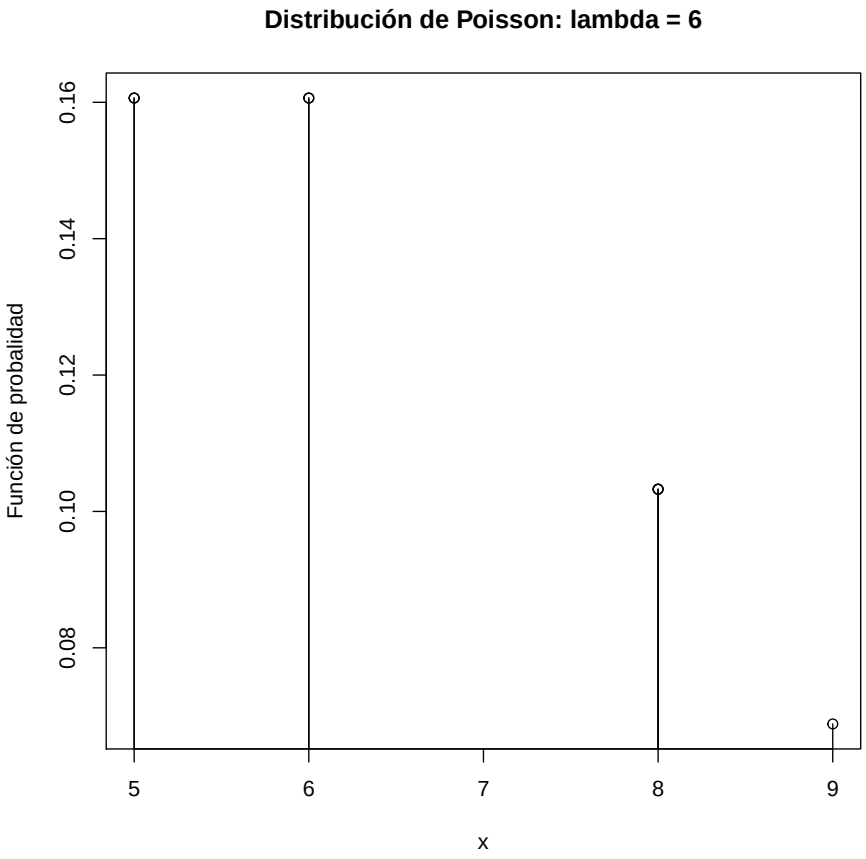
```r
#b) Entre 6 y 8 huracanes
#primera forma
sum(dpois(c(6,7,8),lambda = 6))
```

```
## [1] 0.402
```

```r
# segunda forma restar las probabilidades acumuladas
F8 <- ppois(8, lambda = 6, lower.tail=TRUE)
F5 <- ppois(5,lambda = 6, lower.tail=TRUE)
F8 - F5
```

```
## [1] 0.402
```

```r
#c) Represente gráficamente la función de probabilidad
#de la variable aleatoria X que mide el número de huracanes por año.
#n <- 30
#genera 30 valores de una distribución de Poisson con ??=6
x <- rpois(n, lambda=mu)

#calcula las probabilidades para cada valor generado
y <- dpois(x, lambda=mu)

#genera el gráfico de distribución
plot(x, y, xlab="x", ylab="Función de probalidad",
main="Distribución de Poisson: lambda = 6",type="h")

#une los puntos a las líneas
points(x, y, pch=21)
```




```
#Ejemplo 3:
#En un juego se disponen 15 globos llenos de agua, delos que 4 tienen premio. Los particip
#juego, con los ojos vendados, golpean los globos con un bate por orden hasta que cada uno

#a) ¿Cuál es la probabilidad de que elprimer participante consiga un premio?
# x define el número de globos con premio
x <- 0:2; m = 11; n <- 4; k=2

# se construye la distribución de frecuencias del número de premios
Tabla <- data.frame(Probabilidad=dhyper(x, m, n, k))
rownames(Tabla) <- c("Ningún premio","Solamente uno", "Dos premios")
Tabla

##               Probabilidad
## Ningún premio       0.0571
## Solamente uno       0.4190
## Dos premios         0.5238

#b) Si el primer participante ha conseguido sólo un premio, ¿cuál es la probabilidad de qu
#segundo participante consiga otro?
```

```r
x = 1; m= 10; n= 3; k= 2;
dhyper(x, m, n, k)
```

```
## [1] 0.385
```

```r
#Ejemplo 4:
#Un vendedor de alarmas de hogar tiene éxito enuna casa de cada diez que visita.
#Calcula:

#a) La probabilidad de que en un día determinado consiga vender la primera alarma en la se
#casa que visita.
# x define el número de intentos fallidos
x <- 0:5; p=0.1

# creando la tabla de distribución de frecuencias del número de intentos fallidos antes de
#obtener la primera venta.
Tabla <- data.frame(Probabilidad=dgeom(x, prob=p))

# nombrando las filas de la distribución de frecuencias
rownames(Tabla) <- c("Venta en el primer intento", "Venta en el segundo intento",
"Venta en el tercer intento", "Venta en el cuarto intento",
"Venta en el quinto intento", "Venta en el sexto intento")
Tabla
```

```
##                             Probabilidad
## Venta en el primer intento        0.1000
## Venta en el segundo intento       0.0900
## Venta en el tercer intento        0.0810
## Venta en el cuarto intento        0.0729
## Venta en el quinto intento        0.0656
## Venta en el sexto intento         0.0590
```

```r
#b) La probabilidad de que no venda ninguna después de siete viviendas visitadas.
x=0; n=7; p=0.1
dbinom(x, n, p, log = FALSE)
```

```
## [1] 0.478
```

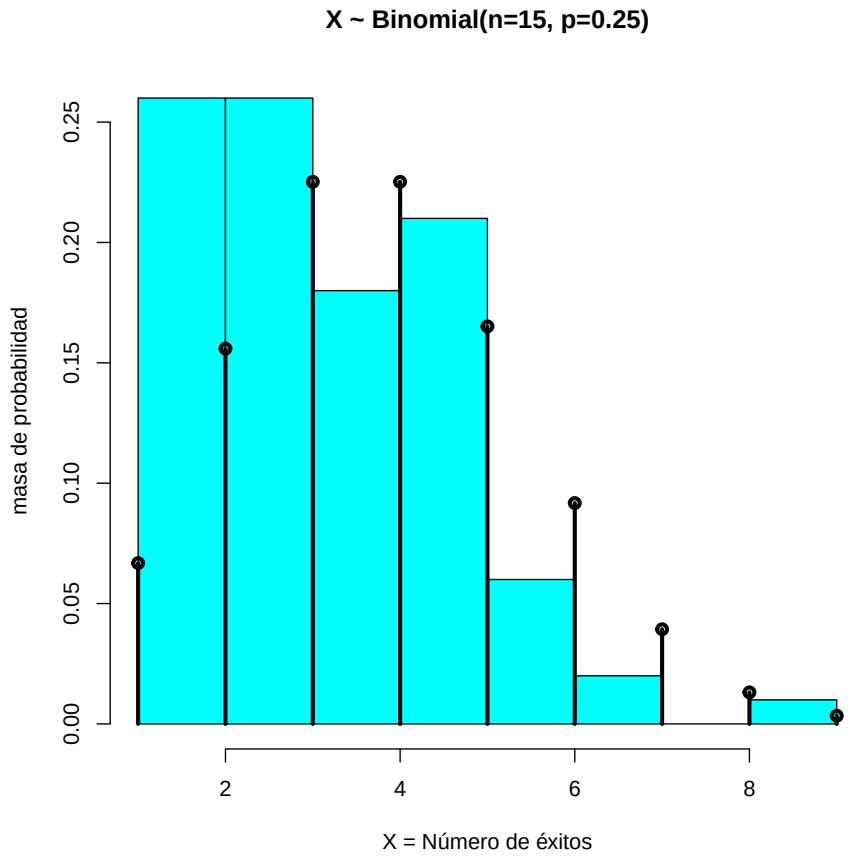
```r
#c) Si se plantea vender tres alarmas, ¿cuál es la probabilidad deque consiga su objetivo
#octava vivienda que visita?
y <- 0:5; r=3; p <- 0.1
Tabla <- data.frame(Probabilidad=dnbinom(y, size=r, prob=p))
rownames(Tabla) <- 0:5
Tabla
```

```
##   Probabilidad
## 0      0.00100
## 1      0.00270
## 2      0.00486
## 3      0.00729
## 4      0.00984
## 5      0.01240
```

```r
#GENERACIÓN DE MUESTRAS ALEATORIAS DE LAS DISTRIBUCIONES
#Ejemplo 1:
#Generar 100 números aleatorios de una distribución Binomial de parámetros n= 15 ensayos o
#y una probabilidad de éxito de 0.25.
# Definir los parámetros apropiados
n <- 15; p <- 0.25

# generar 100 números aleatorios binomiales
x = rbinom(100, n, p); x
```

```
##   [1] 6 3 5 2 5 5 5 7 5 1 6 5 1 1 3 3 4 2 2 4 4 3 5 3 2 6 4 4 3 3 3 5 5 3 3
##  [36] 5 3 2 6 5 4 5 2 4 2 2 3 2 3 2 5 2 4 2 3 5 4 3 5 7 4 4 5 5 2 3 2 3 1 3
##  [71] 2 4 2 4 9 3 2 2 3 2 6 4 5 5 3 4 3 2 1 1 4 4 3 4 3 3 5 5 3 6
```

```r
# Histograma para la muestra aleatoria de tamaño 100
hist(x, main="X ~ Binomial(n=15, p=0.25)", xlab="X = Número de éxitos",
ylab="masa de probabilidad", probability=TRUE, col="Cyan")

# Graficar la función de probabilidad teórica, use la función points(),
#no debe cerrar el gráfico obtenido con la instrucción anterior
xvals=0:n; points(xvals, dbinom(xvals, n, p), type="h", lwd=3)
points(xvals, dbinom(xvals, n, p), type="p", lwd=3)
```

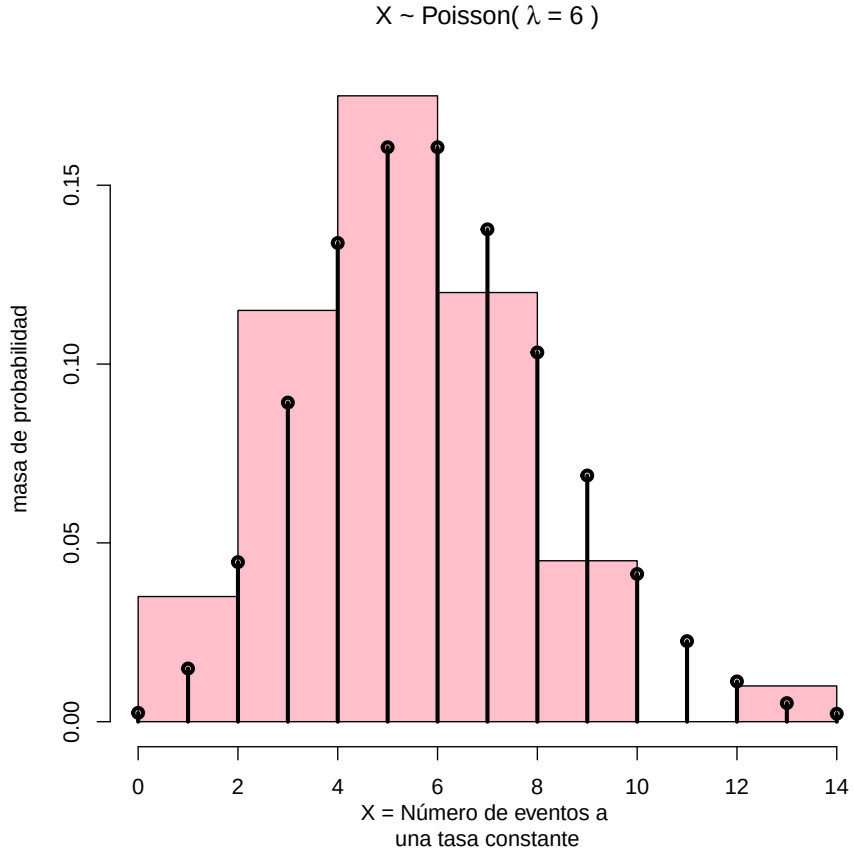



```
#Ejemplo 2:
#Generar 100 números aleatorios de una distribución Poisson con 200000
#ensayos o pruebas y una probabilidad de éxito de 3/100000

# Definir los parámetros apropiados
n <- 200000; p <- 3/100000; lambda=n*p

# generar 100 números aleatorios de la distribución
x = rpois(100, lambda); x
```

```
##   [1]  7  7  7  8  8  8  4  8  7  6  3 13  6  8  7  5  7  5  5  8  5  6  5
##  [24]  9  3  5  3  8  5  6  6  6  6  3  6  6 10  2  6  5  5  4  9  6  5  9
##  [47]  4  8  3  9  3  5  3  6  5  8  0  4  6  3  7  6  8  7  6  2  5  1  3
##  [70]  8  0  4  7 10  4  7  4  5  4  5  4  5 10  5 10  4  1  9  4 13  5  1
##  [93]  8  4  5  8  8  4  4  5
```

```
# Histograma para la muestra aleatoria de tamaño 100
hist(x, main=expression(paste("X ~ Poisson( ", lambda, " = 6 )")), xlab="X = Número de eve
una tasa constante", ylab="masa de probabilidad", probability=TRUE, col="pink")
```

```r
# Graficar la función de probabilidadteórica, use la función points()
xvals=0:n; points(xvals, dpois(xvals, lambda), type="h", lwd=3)
points(xvals, dpois(xvals, lambda), type="p", lwd=3)
```

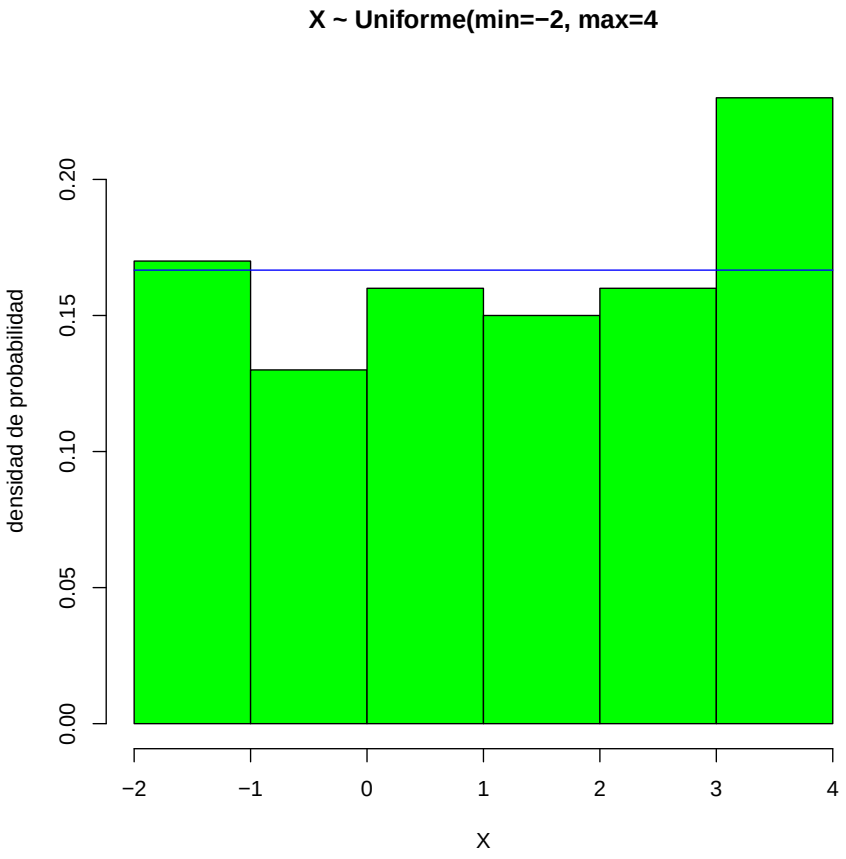



```r
#3. GENERACIÓN DE MUESTRAS ALEATORIAS DE LAS DISTRIBUCIONES
#Ejemplo 1:
#Generar 100 números aleatorios de una distribución Uniforme en [-2, 4]

#Definir los parámetros apropiados
min <- -2; max <- 4

#Generar 100 números aleatorios de la distribución
x = runif(100, min, max); x
```

```
##   [1] -1.0538  2.5133  3.8057  1.0215  2.7824  0.8395  3.8096 -0.0823
##   [9]  1.7396  1.7294  3.1855 -1.3191  1.7870  0.5221  0.1276 -1.0835
##  [17] -0.1442 -0.7418  2.6648  3.4860 -1.8605 -1.7565  2.2712  3.6213
##  [25] -1.4783  3.3601  2.2842  1.4992 -0.7821  2.3320  1.7817  0.5975
##  [33]  0.7804  0.5846  3.2540  3.8579  0.0375  0.5869 -1.4594  3.0806
##  [41]  3.8349  3.7505  2.8964 -1.2629  2.8829 -0.8356 -1.6744  0.0716
```

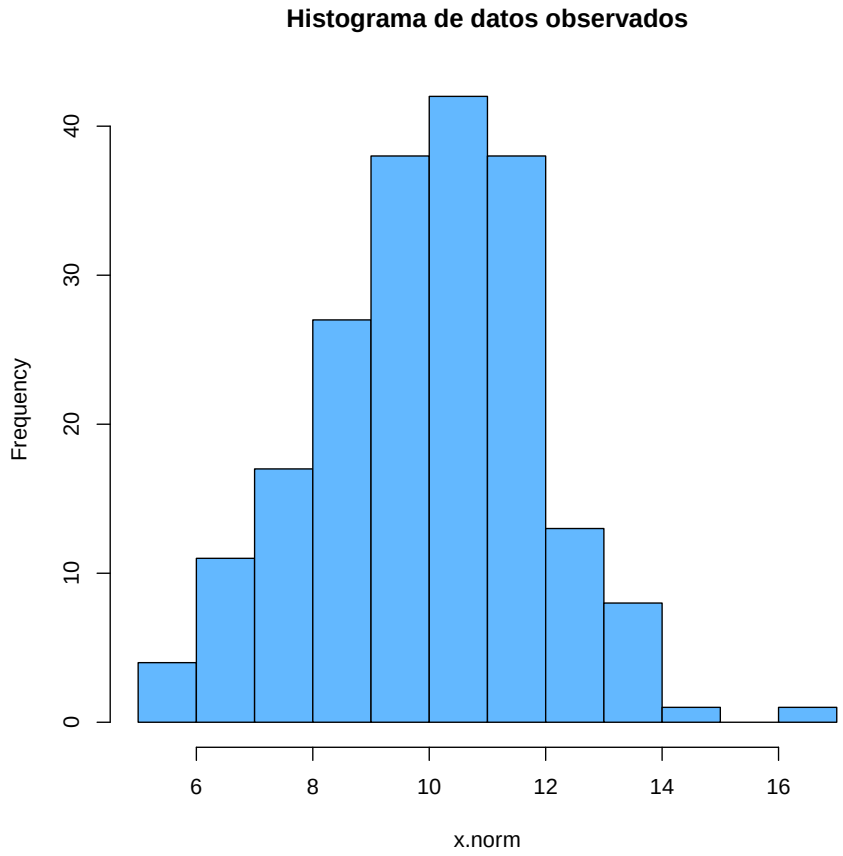
```
##  [49] -0.3991  3.2519  0.3540 -1.6604 -1.4238  3.0938  0.5681  3.1491
##  [57]  1.7629  0.8542 -1.7091  3.2531  0.0317 -0.4119  3.5731  3.8933
##  [65]  1.4422  2.4828  1.4095 -0.9630  3.8873  3.1218  1.3441  2.2815
##  [73]  2.0632 -0.3577 -1.8555 -1.9561  1.4785 -0.0694 -1.5619  1.8424
##  [81]  1.1848  1.9683  2.9156 -1.7200 -1.4292  0.2461  2.8784  2.5653
##  [89]  0.6444 -0.6981 -0.0917  1.1360  2.4387  3.5300  3.9581 -0.1509
##  [97]  3.0954  2.9737  3.0579  0.3238
```

```r
#Histograma para la nuestra aleatoria de tamaño 100
hist(x, main="X ~ Uniforme(min=-2, max=4", xlab="X", ylab="densidad de probabilidad",
     probability=TRUE, col="green")

#Graficar la función de densidad, use la función curve() para variable continua
curve(dunif(x, min, max), col="blue", add=TRUE)
```

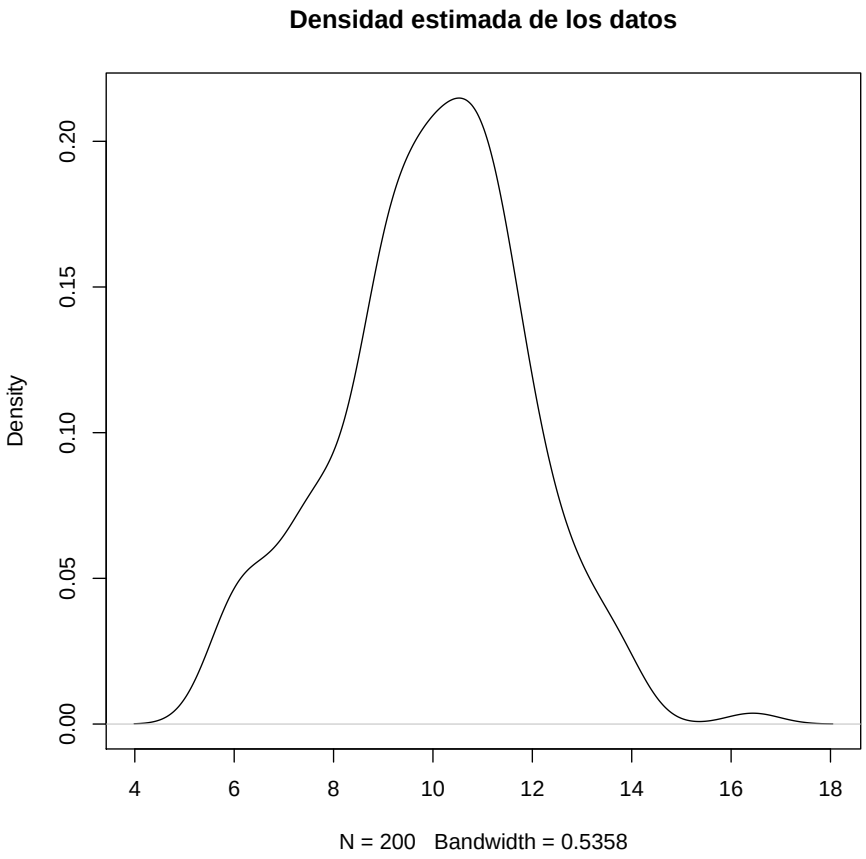



```r
#Ejemplo 2:
#Supongamos que tenemos una muestra de tamañon=200 perteneciente a una población normal
#N(10,2) con ??=10 y ??=2:

#genera los valores aleatorios de la distribución
```

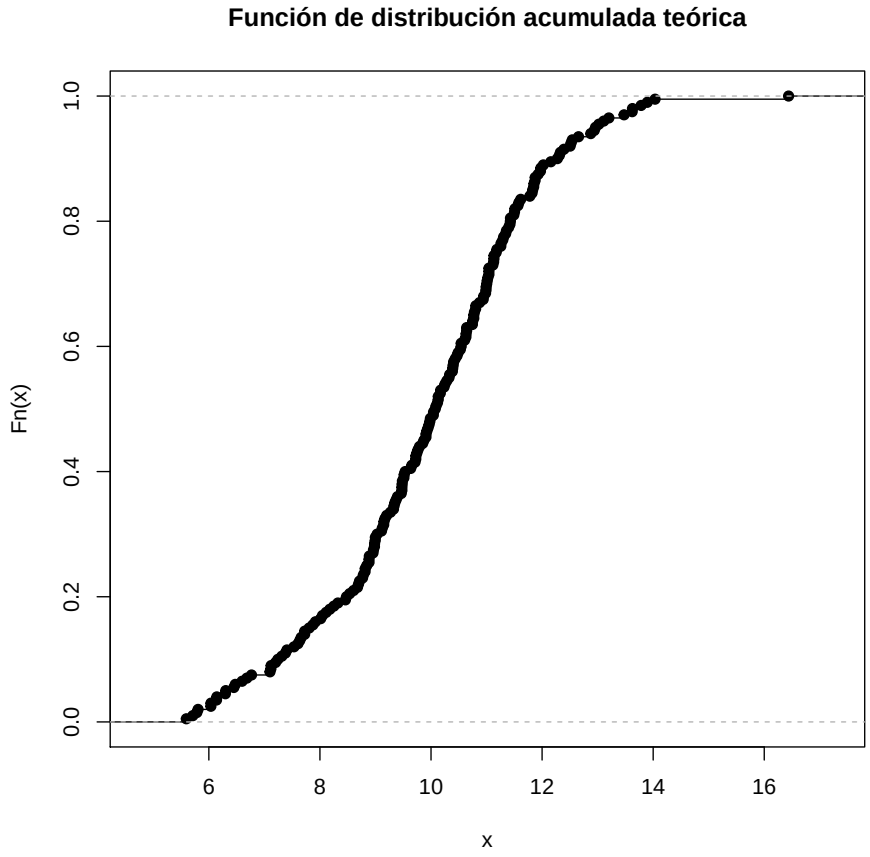
```r
x.norm <- rnorm(n=200,mean=10, sd=2)

# Podemos obtener un histograma usando la función hist()
hist(x.norm, breaks = "Sturges", freq = TRUE, probability = FALSE, include.lowest = TRUE,
right= TRUE, density = NULL, angle = 45, col = "steelblue1", border = NULL,
main = "Histograma de datos observados", axes = TRUE, plot = TRUE, labels = FALSE)
```




```r
# Podemos estimar la densidad de frecuencia usando la función density() y plot() para dibu
plot(density(x.norm), main="Densidad estimada de los datos")
```




```
# R permite calcular la función de distribución acumulada teórica con ecdf()
plot(ecdf(x.norm),main="Función de distribución acumulada teórica")
```

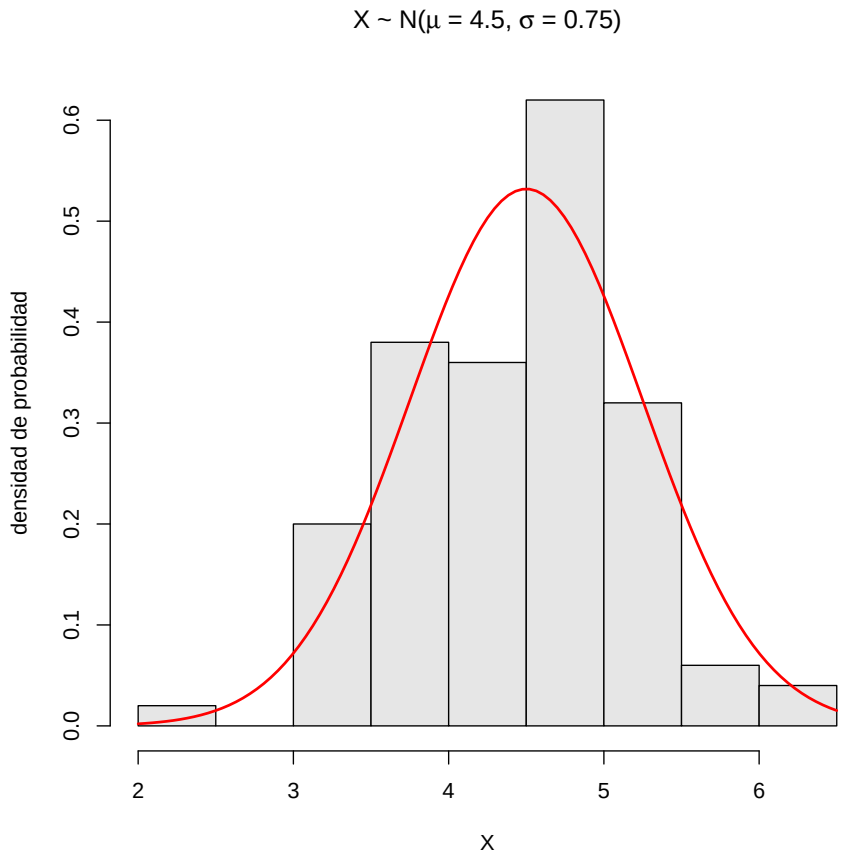



```
#Ejemplo 3:
#Generar 100 números aleatorios de una distribución Normal con media 4.5 y desviación está

# Definir los parámetros apropiados
media <- 4.5; desviacion <- 0.75

# generar 100 números aleatorios de la distribución
x = rnorm(100, media, desviacion); x
```

```
##   [1] 3.85 3.80 4.15 4.29 4.70 4.57 5.17 3.28 5.20 5.17 4.84 5.45 4.39 5.58
##  [15] 5.04 3.66 4.38 3.46 5.43 4.29 4.19 4.56 4.85 3.99 5.42 4.30 5.59 4.62
##  [29] 5.50 3.64 5.37 4.74 4.99 6.20 4.73 4.61 3.88 3.78 3.71 4.25 3.95 4.63
##  [43] 4.84 3.89 5.18 6.08 5.23 4.82 4.86 4.15 4.60 4.60 3.02 5.26 3.61 4.70
##  [57] 4.66 4.01 4.13 4.68 4.57 3.05 4.19 3.24 3.42 4.15 3.08 4.63 5.12 4.77
##  [71] 3.68 3.75 3.28 4.78 5.16 2.42 4.77 4.40 4.21 4.35 3.61 4.97 4.67 3.86
##  [85] 3.92 4.75 3.88 5.13 4.70 4.67 4.32 4.70 4.44 5.98 4.61 3.73 5.42 3.35
##  [99] 3.36 3.80
```

```
# Histograma para la nuestra aleatoria de tamaño 100
hist(x,main=expression(paste("X ~ N(", mu, " = 4.5, ", sigma, " = 0.75)")),
```

```r
xlab="X", ylab="densidad de probabilidad", probability=TRUE, col=gray(0.9))

# Graficar la función de densidad teórica, usando la función curve()
curve(dnorm(x, media, desviacion), col="red", lwd=2, add=TRUE)
```

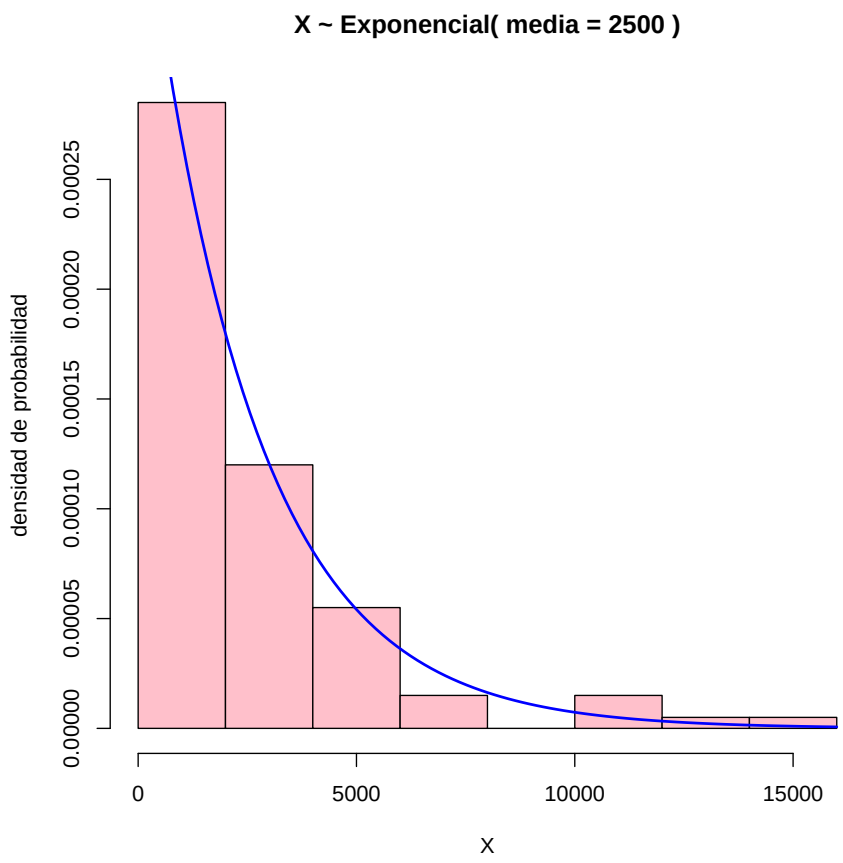



```r
#Ejemplo 4:
#Generar números aleatorios de una distribución exponencial. Por ejemplo, si la vida media
#bulbo de luz es 2500 horas, uno puede pensar que el tiempo de vida es aleatorio con una d
#exponencial que tiene media 2500. El único parámetro es la razón = 1/media.

# Definir el parámetro apropiado
media <- 2500; razon <- 1/media;n=100

# generar 100 números aleatorios de la distribución
x = rexp(n, razon); x
```

```
##   [1] 14679.8  3594.3    42.8  1233.9   937.6  2551.1  3363.8   488.5
##   [9]  2715.7  3651.8 11959.1  1433.5  4014.1  4514.0  3096.5   645.4
##  [17]   773.9  7084.5   477.8  1853.5  1369.6   841.9   945.2  1336.3
##  [25]  4497.0   796.0   278.2  1049.2  1138.9  2800.3   726.2  3134.2
```

```
##  [33]  4654.7  2761.7   102.5   764.8  3627.9  2427.5  2250.2   425.8
##  [41]  2573.0  1582.9    76.8  3716.5   165.6   478.7   850.8   752.3
##  [49]  2815.9  3267.7  1089.6  1323.9  1953.1   478.4   856.5 11723.3
##  [57]  2619.2  6198.8  2838.9  1782.7  2872.1   228.8    38.3  3470.8
##  [65]   331.7   117.7  3510.1   820.2  4108.1  4277.9   820.3   984.4
##  [73]   321.9  3179.5   781.0  6207.7   134.7  3121.4   117.6   244.2
##  [81]  1607.4   734.1  5565.6   527.2  1439.0  4581.1 11655.7  1359.2
##  [89]  5286.8   659.1    53.5  5059.4 12968.9  4770.1  1290.7  1012.2
##  [97]   585.7    68.9  1301.2  3784.5
```

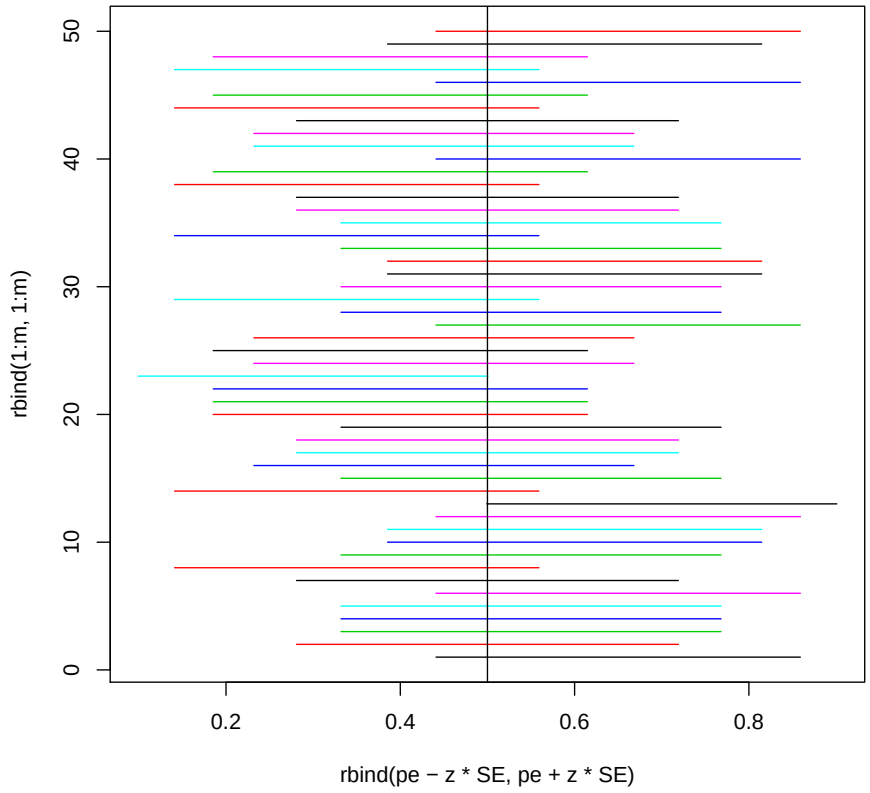
```
# Histograma para la nuestra aleatoria de tamaño 100
hist(x, main="X ~ Exponencial( media = 2500 )", xlab="X",
ylab="densidad de probabilidad", probability=TRUE, col="pink")

# Graficar la función de densidad, usando la función curve()
curve(dexp(x, razon), col="blue", lwd=2, add=TRUE)
```




```
#4. FUNCIONES DE DISTRIBUCIÓN Y SU INVERSA (LOS CUANTILES).
#Ejemplo 1:Para una Variable aleatoria X con distribución normal de media 1 y desviación
#estándar 1, ¿cuál es la probabilidad de que sea menor que 0.7?
```

```r
x <- 0.7
p <- pnorm(x, mean=1, sd=1, lower.tail = TRUE); p

## [1] 0.382

#Ejemplo 2:
#Para una variable aleatoria con distribución normal estándar, encontrar P[Z ??? 0.7 ] y P
z <- 0.7
p1 <- pnorm(z, mean=0, sd=1); p1

## [1] 0.758

p2 <- pnorm(z, mean=0, sd=1, lower.tail=FALSE); p2

## [1] 0.242

#Ejemplo 3:
#¿Qué valor de una variable aleatoria con distribución normal estándar, tiene 75%
#del área a la izquierda?.
p <- 0.75
z <- qnorm(p, mean=0, sd=1, lower.tail = TRUE); z

## [1] 0.674

#Ejemplo 4:
#¿Cuál es la probabilidad a la derecha de 18.55 para una Variable aleatoria X con
#distribución Chi-cuadrado de 12 grados de libertad?
x <- 18.55; gl <- 12
p <- pchisq(x, gl, lower.tail = FALSE); p

## [1] 0.1

#UNIDAD 4: Práctica 17 - Inferencia estadística, Estimación.

#SIMULACIÓN DEL CONCEPTO DE INTERVALO DE CONFIANZA PARA ESTIMAR UN PARÁMETRO.

#Ejemplo 1.
#Sea la variable aleatoria X = el número de caras obtenidas, al lanzar una moneda
#balanceada 20 veces. Simulamos 50 muestras para generar intervalos de 95% de
#confianza y así poder estimar la proporción verdadera de caras (p), y encontrar
#en cuántos de estos intervalos se encuentra el verdadero valor de la proporción

simulIntProp <- function(m=5, n=1, p, nivel.conf=0.95)
{
  X <- rbinom(m, n, p)
  # Matriz con 1000 valores aleatorios binomial(n,p), 50 muestras cada una de tamaño 20
  pe <<- X/n
  # Calcula la proporción estimada en cada una de las muestras.
  SE <<- sqrt(pe*(1-pe)/n)
  # Calcula la desviación estándarestimada en cada una de las muestras.
  alfa <- 1-nivel.conf
```

```r
  z <<- qnorm(1-alfa/2)
  Intervalo <<- cbind(pe - z*SE, pe + z*SE)
  # genera los extremos del intervalo de confianza
  nInter <<- 0
  # un contador para conocer en cuántos intervalos se encuentra la verdadera proporción.
  for(i in 1:m)
    if ((p >= Intervalo[i, 1]) && (p <= Intervalo[i, 2]))
      nInter <<- nInter + 1
  # función que cuenta cuántos intervalos contienen el verdadero valor del parámetro.
  return(nInter)
}
n=20; m= 50; p=0.5; nivel.conf=0.95
simulIntProp(m, n, p, nivel.conf)

## [1] 50

#Gráfico que muestra los intervalosde confianza de 95% que contienen y no contienen el ver
#valor del parámetro p.
matplot(rbind(pe - z*SE, pe + z*SE), rbind(1:m, 1:m), type="l", lty=1)
abline(v=p)
```



```r
#Ejercicio 1.
#Sea la variable aleatoria X = el número que se obtiene al lanzar un dado no cargado
#30 veces. Simular 56 muestras para generar intervalos de 95%de confianza para
#estimar el promedio (??), y encontrar cuántos de estos intervalos contiene el valor
#medio verdadero.

simulIntProp <- function(m=5, n=1, p, nivel.conf=0.95)
{
  X <- rbinom(m, n, p)
pe <<- X/n
SE <<- sqrt(pe*(1-pe)/n)
alfa <- 1-nivel.conf
z <<- qnorm(1-alfa/2)
Intervalo <<- cbind(pe - z*SE, pe + z*SE)
nInter <<- 0
for(i in 1:m)
if ((p >= Intervalo[i, 1]) && (p <= Intervalo[i, 2])) nInter <<- nInter + 1
return(nInter)
}
n=30; m= 56; p=0.5; nivel.conf=0.95
simulIntProp(m, n, p, nivel.conf)

## [1] 51
```