scrip1clase.R

arturo

2023-05-11

```
##INICIANDO EN EL LENGUAJE R##
##MC.Arturo Ramírez-Ordorica curso 2023
set.seed(150)
#Todo en R es un objeto.
#Tipos de estructuras de datos en R:
#Vector, lista, matriz, dataframe y factores
#class()#Permite verificar el tipo es dato que tenemos
#tipos de datos: character, numeric(floating point), integer, logical, complex
#ejemplos:
a=c("a","ww","r")
#principales funciones para revisar atributos de los objetos en R
class(a)#tipo de obejeto
```

```
## [1] "character"
```

```
typeof(a)#tipo de objeto (como está almacenado??)
```

```
## [1] "character"
```

```
length(a) #largo (vectoroes)
```

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

```
dim(a)#dimensiones (matrices)
```

```
## NULL
```

```
attributes(a)#metadatos asociados al objeto (nombres, p.ejemp) no todos los obejetos poseen atributos
```

```
## NULL
```

```
a<-c(a,"wz","z") #usando operador de asignación " <-"
print(a)#explicita la impresión del obejeto (invocamos objeto)
```

```
## [1] "a"  "ww" "r"  "wz" "z"
```

```
#¿Qué tipo de dato contienen los siguientes objetos?:
obj1="Lenguaje R"
obj2=1:10
obj3<-rep(FALSE,3)
obj4<-c(2L,3L,1000L)
obj5<-5.23443
obj6<-10:1

ls()#Qué objetos tenemos en nuestra sesión?
```

```
## [1] "a"    "obj1" "obj2" "obj3" "obj4" "obj5" "obj6"
```

```r
#Objetos NA (Not Avaible), NaN (Not a Number) e Inf
b<-c(3,NA,3)#Surge en espacios vacios en nuestras bases de datos. Huecos
0/0#Valores indefinidos
```

```
## [1] NaN
```

```r
1/0#¿Qué significa?
```

```
## [1] Inf
```

```r
-Inf
```

```
## [1] -Inf
```

```r
#Para identificarlos: is.na() y anyNA()
d<-seq(1:10)
print(d)
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10
```

```r
d[2]<-NA#Podemos usar el operador de asignación
d[5]<-NA
d[c(2,5)]<-NA
print(d)
```

```
##  [1]  1 NA  3  4 NA  6  7  8  9 10
```

```r
is.na(d)#¿Qué obtenemos?
```

```
##  [1] FALSE  TRUE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE
```

```r
anyNA(d)#¿Y aquí?
```

```
## [1] TRUE
```

```r
!is.na(d)
```

```
##  [1]  TRUE FALSE  TRUE  TRUE FALSE  TRUE  TRUE  TRUE  TRUE  TRUE
```

```r
##Vectores
#Un vector es una colección de objetos que tienen la misma categoría de dato
#(numerico, caracter, lógico, etc)
#c()#Función concatenar
un_vector<-c(1,2,3,6,11,43,1e10)
print(un_vector)
```

```
## [1] 1.0e+00 2.0e+00 3.0e+00 6.0e+00 1.1e+01 4.3e+01 1.0e+10
```

```r
un_vector[5]
```

```
## [1] 11
```

```r
un_vector[c(2,6)]
```

```
## [1]  2 43
```

```r
un_vector[-5]
```

```
## [1] 1.0e+00 2.0e+00 3.0e+00 6.0e+00 4.3e+01 1.0e+10
```

```r
#Podemos crear un vector con vector()
un_vector2<-vector("numeric",3)
?vector()
```

```r
#Crear un vector lógico de tamaño 10

#¿Cuál es su longitud?

#Funciones útiles para crear vectores:
seq(from=2,to=20,by=2)#Esta es una "función" de nombre seq() con argumentos
```

```
##  [1]  2  4  6  8 10 12 14 16 18 20
```

```r
#from=, to=, by=. Casi todas las funciones requieren de argumentos específicos
rep(2,4)#Para repetir un objeto n cantidad de veces
```

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

```r
#Ejercicio
#Crear un vector con los números del 1 al 100 de 5 en 5
#Crear un vector con 10 nombres
#unirlos
#¿Qué longitud tiene?

#OPERACIONES MATEMÁTICAS
#Podemos hacer operaciones con vectores...los operadores son fáciles de
#identificar por su sintaxis

#Ejercicio
#Crear un vector con la raíz cuadrada de los primeros 100 enteros
#Crear un vector con el cuadrado de los primeros 100 enteros
#arcotangente del vector anterior

##Nombres "names()"
vector_3<-vector("numeric",6)
print(vector_3)
```

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

```r
names(vector_3)
```

```
## NULL
```

```r
##Crea un vector de caracteres con los nombres
#"Jośe","Inés","Raúl","Arturo","Berenice","Maricela"
#¿Cómo llamamos a los casos de cada posición en el vector?
#Asigne calificaciones a cada caso

#mean()#media de calificaciones
#sd()#Desviación estandar

#Podemos hace operaciones sobre vectores: suma, resta, multiplicación
#Dar ejemplos

#MATRICES
#Pueden ser concebidos como conjuntos de vectores de la misma longitud
#ordenados en filas y/o columnas.
#Están compuestos por la misma clase de elementos (a semejanza de los vectores)
matrix(seq(1:20),nrow=5,ncol=4)
```

```
##      [,1] [,2] [,3] [,4]
```

```
## [1,]    1    6   11   16
## [2,]    2    7   12   17
## [3,]    3    8   13   18
## [4,]    4    9   14   19
## [5,]    5   10   15   20
```

```
#Llenar por filas??
matriz_1<-matrix(seq(1:20),nrow=5,ncol=4,byrow=TRUE)
##Posiciones [fila, columna]
matriz_1[3,2]
```

```
## [1] 10
```

```
##Podemos hace operaciones sobre matrices.
#Suma,multiplicación por un escalar
matriz_2<-matrix(seq(1:20),nrow=5,ncol=4,byrow=TRUE)
matriz_3<-matrix(seq(1:20),nrow=5,ncol=4)
matriz_2+matriz_3
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    2    8   14   20
## [2,]    7   13   19   25
## [3,]   12   18   24   30
## [4,]   17   23   29   35
## [5,]   22   28   34   40
```

```
matriz_2*matriz_3#multiplicación elemento a elemento
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    1   12   33   64
## [2,]   10   42   84  136
## [3,]   27   80  143  216
## [4,]   52  126  210  304
## [5,]   85  180  285  400
```

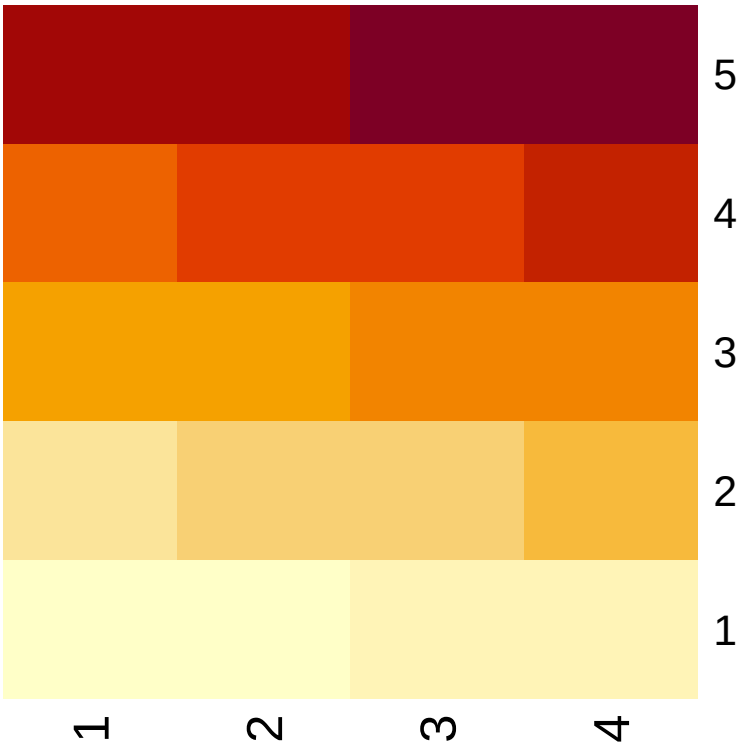
```
#dim() para ver las dimensiones en la matríz
dim(matriz_2)
```

```
## [1] 5 4
```

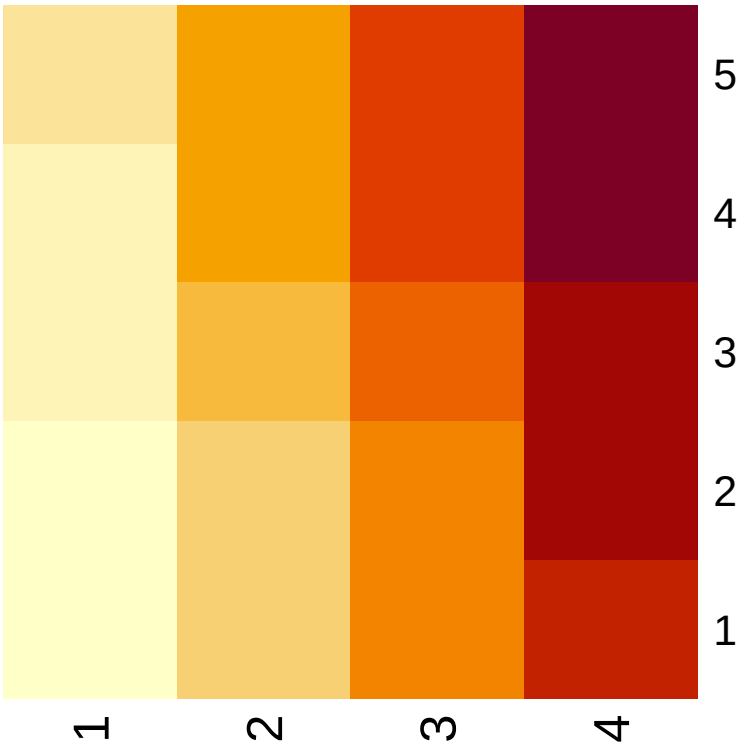
```
#Multiplicacion matricial
matriz_2 %*% t(matriz_3)#t() transpone la matriz y le das las dimensiones
```

```
##      [,1] [,2] [,3] [,4] [,5]
## [1,]  110  120  130  140  150
## [2,]  246  272  298  324  350
## [3,]  382  424  466  508  550
## [4,]  518  576  634  692  750
## [5,]  654  728  802  876  950
```

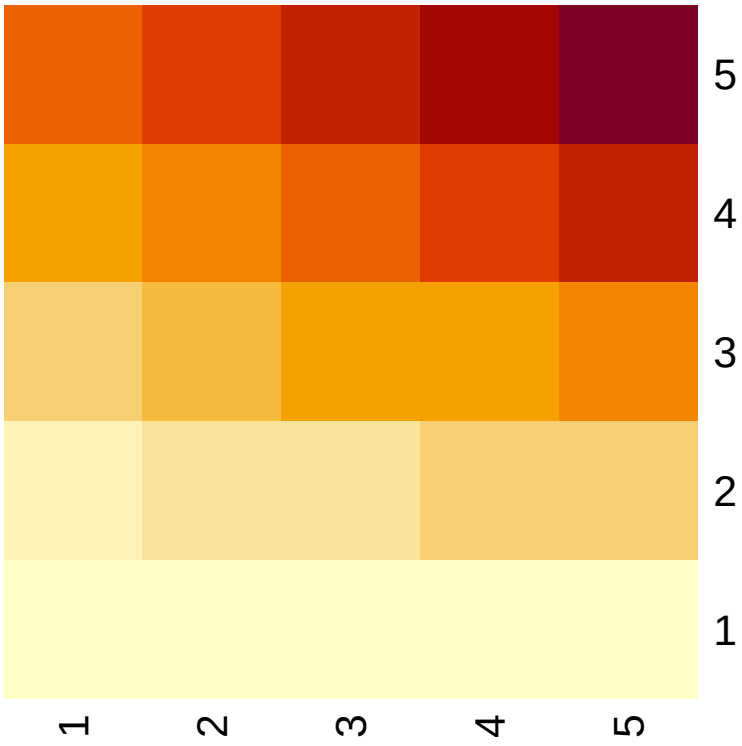
```
#adecuadas para multiplicarlas (5x4)%*%(4x5)=5x5

#Hagamos una pequeña gŕafica
heatmap(matriz_2,Rowv = NA,Colv = NA,scale="none")
```




```
heatmap(matriz_3,Rowv = NA,Colv = NA,scale = "none")
```




```
heatmap(matriz_2 %*% t(matriz_3),Rowv = NA,Colv = NA,scale = "none")
```




```
class(matriz_2)
```

```
## [1] "matrix" "array"
```

```
typeof(matriz_2)
```

```
## [1] "integer"
```

```
matriz_2[2,]
```

```
## [1] 5 6 7 8
```

```
#Para muchos casos, es conveniente saber los nombres de las columnas y
#renglones de las matrices
colnames(matriz_2)
```

```
## NULL
```

```
row.names(matriz_2)
```

```
## NULL
```

```
#Asignemos nombres a las columnas y las filas de la matriz

#Podemos crear matrices uniendo filas y columnas
#cbind()#unir columnas
#rbind()#unir filas

##LISTAS
#Es un objeto que contiene elementos de diferentes clases:
```

```r
lista_1<-list("a",TRUE,FALSE,vector_3,matriz_2)
length(lista_1)
```

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

```r
lista_1[[3]]
```

```
## [1] FALSE
```

```r
lista_1[[5]]
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    1    2    3    4
## [2,]    5    6    7    8
## [3,]    9   10   11   12
## [4,]   13   14   15   16
## [5,]   17   18   19   20
```

```r
lista_1[[5]][3,2]
```

```
## [1] 10
```

```r
length(lista_1)
```

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

```r
#Demosle nombres
names(lista_1)<-c("Caracter","Verdadero","Falso","Un vector","Una matriz")
#Llamemos elementos de la lista
lista_1$Verdadero
```

```
## [1] TRUE
```

```r
##DATAFRAME
#Son tablas de datos, y son un tipo especial de lista en forma de tabla.
#Los dataframe son el tipo más común de objeto que se usa en aplicaciones de
#análisis de datos

nombres=c("Joše","Inés","Raúl","Arturo","Berenice","Maricela")
calificaciones=c(9,8,10,9,8,10)
salon=c("A","C","E","C","D","C")
mi_tabla=cbind(nombres,calificaciones,salon)
print(mi_tabla)
```

```
##      nombres    calificaciones salon
## [1,] "Joše"     "9"            "A"
## [2,] "Inés"     "8"            "C"
## [3,] "Raúl"     "10"           "E"
## [4,] "Arturo"   "9"            "C"
## [5,] "Berenice" "8"            "D"
## [6,] "Maricela" "10"           "C"
```

```r
data(iris)
iris
```

```
##     Sepal.Length Sepal.Width Petal.Length Petal.Width    Species
## 1            5.1         3.5          1.4         0.2     setosa
## 2            4.9         3.0          1.4         0.2     setosa
## 3            4.7         3.2          1.3         0.2     setosa
```

```
## 4           4.6         3.1          1.5         0.2     setosa
## 5           5.0         3.6          1.4         0.2     setosa
## 6           5.4         3.9          1.7         0.4     setosa
## 7           4.6         3.4          1.4         0.3     setosa
## 8           5.0         3.4          1.5         0.2     setosa
## 9           4.4         2.9          1.4         0.2     setosa
## 10          4.9         3.1          1.5         0.1     setosa
## 11          5.4         3.7          1.5         0.2     setosa
## 12          4.8         3.4          1.6         0.2     setosa
## 13          4.8         3.0          1.4         0.1     setosa
## 14          4.3         3.0          1.1         0.1     setosa
## 15          5.8         4.0          1.2         0.2     setosa
## 16          5.7         4.4          1.5         0.4     setosa
## 17          5.4         3.9          1.3         0.4     setosa
## 18          5.1         3.5          1.4         0.3     setosa
## 19          5.7         3.8          1.7         0.3     setosa
## 20          5.1         3.8          1.5         0.3     setosa
## 21          5.4         3.4          1.7         0.2     setosa
## 22          5.1         3.7          1.5         0.4     setosa
## 23          4.6         3.6          1.0         0.2     setosa
## 24          5.1         3.3          1.7         0.5     setosa
## 25          4.8         3.4          1.9         0.2     setosa
## 26          5.0         3.0          1.6         0.2     setosa
## 27          5.0         3.4          1.6         0.4     setosa
## 28          5.2         3.5          1.5         0.2     setosa
## 29          5.2         3.4          1.4         0.2     setosa
## 30          4.7         3.2          1.6         0.2     setosa
## 31          4.8         3.1          1.6         0.2     setosa
## 32          5.4         3.4          1.5         0.4     setosa
## 33          5.2         4.1          1.5         0.1     setosa
## 34          5.5         4.2          1.4         0.2     setosa
## 35          4.9         3.1          1.5         0.2     setosa
## 36          5.0         3.2          1.2         0.2     setosa
## 37          5.5         3.5          1.3         0.2     setosa
## 38          4.9         3.6          1.4         0.1     setosa
## 39          4.4         3.0          1.3         0.2     setosa
## 40          5.1         3.4          1.5         0.2     setosa
## 41          5.0         3.5          1.3         0.3     setosa
## 42          4.5         2.3          1.3         0.3     setosa
## 43          4.4         3.2          1.3         0.2     setosa
## 44          5.0         3.5          1.6         0.6     setosa
## 45          5.1         3.8          1.9         0.4     setosa
## 46          4.8         3.0          1.4         0.3     setosa
## 47          5.1         3.8          1.6         0.2     setosa
## 48          4.6         3.2          1.4         0.2     setosa
## 49          5.3         3.7          1.5         0.2     setosa
## 50          5.0         3.3          1.4         0.2     setosa
## 51          7.0         3.2          4.7         1.4 versicolor
## 52          6.4         3.2          4.5         1.5 versicolor
## 53          6.9         3.1          4.9         1.5 versicolor
## 54          5.5         2.3          4.0         1.3 versicolor
## 55          6.5         2.8          4.6         1.5 versicolor
## 56          5.7         2.8          4.5         1.3 versicolor
## 57          6.3         3.3          4.7         1.6 versicolor
```

```
## 58          4.9         2.4          3.3         1.0 versicolor
## 59          6.6         2.9          4.6         1.3 versicolor
## 60          5.2         2.7          3.9         1.4 versicolor
## 61          5.0         2.0          3.5         1.0 versicolor
## 62          5.9         3.0          4.2         1.5 versicolor
## 63          6.0         2.2          4.0         1.0 versicolor
## 64          6.1         2.9          4.7         1.4 versicolor
## 65          5.6         2.9          3.6         1.3 versicolor
## 66          6.7         3.1          4.4         1.4 versicolor
## 67          5.6         3.0          4.5         1.5 versicolor
## 68          5.8         2.7          4.1         1.0 versicolor
## 69          6.2         2.2          4.5         1.5 versicolor
## 70          5.6         2.5          3.9         1.1 versicolor
## 71          5.9         3.2          4.8         1.8 versicolor
## 72          6.1         2.8          4.0         1.3 versicolor
## 73          6.3         2.5          4.9         1.5 versicolor
## 74          6.1         2.8          4.7         1.2 versicolor
## 75          6.4         2.9          4.3         1.3 versicolor
## 76          6.6         3.0          4.4         1.4 versicolor
## 77          6.8         2.8          4.8         1.4 versicolor
## 78          6.7         3.0          5.0         1.7 versicolor
## 79          6.0         2.9          4.5         1.5 versicolor
## 80          5.7         2.6          3.5         1.0 versicolor
## 81          5.5         2.4          3.8         1.1 versicolor
## 82          5.5         2.4          3.7         1.0 versicolor
## 83          5.8         2.7          3.9         1.2 versicolor
## 84          6.0         2.7          5.1         1.6 versicolor
## 85          5.4         3.0          4.5         1.5 versicolor
## 86          6.0         3.4          4.5         1.6 versicolor
## 87          6.7         3.1          4.7         1.5 versicolor
## 88          6.3         2.3          4.4         1.3 versicolor
## 89          5.6         3.0          4.1         1.3 versicolor
## 90          5.5         2.5          4.0         1.3 versicolor
## 91          5.5         2.6          4.4         1.2 versicolor
## 92          6.1         3.0          4.6         1.4 versicolor
## 93          5.8         2.6          4.0         1.2 versicolor
## 94          5.0         2.3          3.3         1.0 versicolor
## 95          5.6         2.7          4.2         1.3 versicolor
## 96          5.7         3.0          4.2         1.2 versicolor
## 97          5.7         2.9          4.2         1.3 versicolor
## 98          6.2         2.9          4.3         1.3 versicolor
## 99          5.1         2.5          3.0         1.1 versicolor
## 100         5.7         2.8          4.1         1.3 versicolor
## 101         6.3         3.3          6.0         2.5  virginica
## 102         5.8         2.7          5.1         1.9  virginica
## 103         7.1         3.0          5.9         2.1  virginica
## 104         6.3         2.9          5.6         1.8  virginica
## 105         6.5         3.0          5.8         2.2  virginica
## 106         7.6         3.0          6.6         2.1  virginica
## 107         4.9         2.5          4.5         1.7  virginica
## 108         7.3         2.9          6.3         1.8  virginica
## 109         6.7         2.5          5.8         1.8  virginica
## 110         7.2         3.6          6.1         2.5  virginica
## 111         6.5         3.2          5.1         2.0  virginica
```

```
## 112          6.4         2.7          5.3         1.9  virginica
## 113          6.8         3.0          5.5         2.1  virginica
## 114          5.7         2.5          5.0         2.0  virginica
## 115          5.8         2.8          5.1         2.4  virginica
## 116          6.4         3.2          5.3         2.3  virginica
## 117          6.5         3.0          5.5         1.8  virginica
## 118          7.7         3.8          6.7         2.2  virginica
## 119          7.7         2.6          6.9         2.3  virginica
## 120          6.0         2.2          5.0         1.5  virginica
## 121          6.9         3.2          5.7         2.3  virginica
## 122          5.6         2.8          4.9         2.0  virginica
## 123          7.7         2.8          6.7         2.0  virginica
## 124          6.3         2.7          4.9         1.8  virginica
## 125          6.7         3.3          5.7         2.1  virginica
## 126          7.2         3.2          6.0         1.8  virginica
## 127          6.2         2.8          4.8         1.8  virginica
## 128          6.1         3.0          4.9         1.8  virginica
## 129          6.4         2.8          5.6         2.1  virginica
## 130          7.2         3.0          5.8         1.6  virginica
## 131          7.4         2.8          6.1         1.9  virginica
## 132          7.9         3.8          6.4         2.0  virginica
## 133          6.4         2.8          5.6         2.2  virginica
## 134          6.3         2.8          5.1         1.5  virginica
## 135          6.1         2.6          5.6         1.4  virginica
## 136          7.7         3.0          6.1         2.3  virginica
## 137          6.3         3.4          5.6         2.4  virginica
## 138          6.4         3.1          5.5         1.8  virginica
## 139          6.0         3.0          4.8         1.8  virginica
## 140          6.9         3.1          5.4         2.1  virginica
## 141          6.7         3.1          5.6         2.4  virginica
## 142          6.9         3.1          5.1         2.3  virginica
## 143          5.8         2.7          5.1         1.9  virginica
## 144          6.8         3.2          5.9         2.3  virginica
## 145          6.7         3.3          5.7         2.5  virginica
## 146          6.7         3.0          5.2         2.3  virginica
## 147          6.3         2.5          5.0         1.9  virginica
## 148          6.5         3.0          5.2         2.0  virginica
## 149          6.2         3.4          5.4         2.3  virginica
## 150          5.9         3.0          5.1         1.8  virginica
```

```
#Nombres
#colnames()
#rownames()

#Para invocar columnas específicas de un data.frame, se utiliza los operadores
#[] y el $. No olvidar que un dataframe es un tipo de lista.
names(iris)
```

```
## [1] "Sepal.Length" "Sepal.Width"  "Petal.Length" "Petal.Width"  "Species"
```

```
iris$Petal.Length
```

```
##   [1] 1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 1.5 1.6 1.4 1.1 1.2 1.5 1.3 1.4
##  [19] 1.7 1.5 1.7 1.5 1.0 1.7 1.9 1.6 1.6 1.5 1.4 1.6 1.6 1.5 1.5 1.4 1.5 1.2
##  [37] 1.3 1.4 1.3 1.5 1.3 1.3 1.3 1.6 1.9 1.4 1.6 1.4 1.5 1.4 4.7 4.5 4.9 4.0
##  [55] 4.6 4.5 4.7 3.3 4.6 3.9 3.5 4.2 4.0 4.7 3.6 4.4 4.5 4.1 4.5 3.9 4.8 4.0
```

```
##  [73] 4.9 4.7 4.3 4.4 4.8 5.0 4.5 3.5 3.8 3.7 3.9 5.1 4.5 4.5 4.7 4.4 4.1 4.0
##  [91] 4.4 4.6 4.0 3.3 4.2 4.2 4.2 4.3 3.0 4.1 6.0 5.1 5.9 5.6 5.8 6.6 4.5 6.3
## [109] 5.8 6.1 5.1 5.3 5.5 5.0 5.1 5.3 5.5 6.7 6.9 5.0 5.7 4.9 6.7 4.9 5.7 6.0
## [127] 4.8 4.9 5.6 5.8 6.1 6.4 5.6 5.1 5.6 6.1 5.6 5.5 4.8 5.4 5.6 5.1 5.1 5.9
## [145] 5.7 5.2 5.0 5.2 5.4 5.1
```

```
#Ejercicio:Poner un nombre a cada fila del dataframe iris. Con formato
#"especie_número consecutivo. Usar paste()

##OPERADORES LÓGICOS
# !=, ==, <, >, <=, >=
#Usar operadores lógicos para encontrar el número de individuos de cada especie en el vector
#de nombres, y asignar un nombre a las filas con formato.
# "especie_número de individuo"

#R posee muchas funciones destinadas para la lectura de diferentes archivos
#Una de las más comunes es en archivos .csv o .txt

#Ejercicio: Crear un dataframe que simule una matriz de datos con
#50 observaciones y 100 señales m/z que tenga una distribución normal
#Introducir 1500 ceros aleatoriamente en el dataframe
datos<-rnorm(50*100,mean=0,sd=2)
muestra<-sample(datos,1500)#sample() sirve para hacer un muestreo de datos
posiciones<-match(muestra,datos)
datos[posiciones]<-0
head(datos)#Observe los ceros
```

```
## [1]  0.0000000 -0.1259925  0.0000000 -0.6283564 -0.5338925  0.3063189
```

```
matriz_mz<-matrix(datos,ncol=100,nrow=50,byrow=T)
dim(matriz_mz)
```

```
## [1]  50 100
```

```
mz<-seq(from=50,to=250,by=200/99)
mz<-round(mz,3)##función para redondear con 3 decimales
sufijo<-rep("m.z",length(mz))
etiqueta<-paste(mz,sufijo,sep="")
colnames(matriz_mz)<-etiqueta
obs<-1:50
obs_nomb<-paste(rep("obs",50),obs,sep = "")
rownames(matriz_mz)<-obs_nomb
mz_tabla<-data.frame(obs_nomb,matriz_mz)
dim(mz_tabla)
```

```
## [1]  50 101
```

```
########BASES DE ESTRUCTURAS DE CONTROL
##Ejercicio: imputar las celdas con ceros por la media de la columna correspondiente
head(mz_tabla[,-1])
```

```
##          X50m.z  X52.02m.z  X54.04m.z  X56.061m.z X58.081m.z X60.101m.z
## obs1  0.0000000 -0.1259925  0.0000000 -0.62835636 -0.5338925  0.3063189
## obs2 -0.4262052  2.2492999 -1.6109729 -0.92083979  1.1842919  1.9827198
## obs3 -4.0097574 -1.0257566 -1.7719826  0.38851388  1.1814521 -2.1385931
## obs4  0.0000000 -0.9667346 -1.3210172  1.75053837 -2.5718487 -2.0375397
## obs5  1.7970402  0.0000000  0.7064478  0.00000000 -0.8117584  0.0000000
```

```
## obs6  1.6642246  0.3695371 -0.8227750  0.07039719 -2.0436204 -2.8469576
##       X62.121m.z X64.141m.z X66.162m.z X68.182m.z  X70.202m.z X72.222m.z
## obs1  1.62777386  0.9952713   0.000000   0.000000 -0.04803872 -3.9183701
## obs2  0.00000000 -1.4287703  -2.650792   1.551627  2.94818191 -1.7943500
## obs3  2.63110279  1.3986544   1.344786   1.608230  0.10081603 -0.1234542
## obs4  0.60012178 -0.7002742   1.220649   1.115005  0.23195189  0.0000000
## obs5 -1.61020792 -0.3463557   1.270240   0.000000  0.00000000 -1.7058206
## obs6 -0.02326944 -0.5548660   5.427570   0.000000  0.00000000  2.1860106
##      X74.242m.z  X76.263m.z X78.283m.z X80.303m.z  X82.323m.z X84.343m.z
## obs1  0.0000000  0.00000000 -1.3400229  0.0000000  0.00000000   2.638293
## obs2  0.4355912  1.98582016  0.0000000  3.5810382  0.00000000   0.000000
## obs3 -0.7878479  0.01150795  0.0000000 -0.3075392  0.06641954   2.510765
## obs4  0.0000000 -3.74471021 -0.1710115  0.0000000 -2.66239316   0.000000
## obs5 -2.4754245 -0.82917707  0.0000000  0.0000000 -1.40494530  -1.036841
## obs6  2.0458952  0.00000000  0.2137679  4.0723677  0.40975968   0.000000
##      X86.364m.z X88.384m.z X90.404m.z X92.424m.z X94.444m.z  X96.465m.z
## obs1  0.0000000  0.0000000  0.0000000  2.6066167  0.0000000 -0.09412124
## obs2 -1.0105095  0.7975126  0.8615268  1.9966978  0.6402835  2.72557928
## obs3  0.0000000 -3.1689841  0.0000000 -0.3993511  3.4122103 -1.51171265
## obs4 -0.5437623  1.2586432  3.8556340  0.6952454  0.7377909  0.21812236
## obs5 -0.4650016  1.0036433  0.3496000  0.0000000 -1.8317490 -1.65115055
## obs6  0.0000000  0.0000000 -1.6388243 -0.5395251  0.0000000  0.00000000
##      X98.485m.z X100.505m.z X102.525m.z X104.545m.z X106.566m.z X108.586m.z
## obs1   4.936464   1.9458706   2.6026854    0.000000    3.305932 -0.62974529
## obs2   0.000000   1.0674235   3.3447234   -3.326909   -1.989398  2.89245020
## obs3   3.474618  -1.8710010  -0.1506432    1.336715   -1.145297 -0.78935287
## obs4   1.025022   0.0000000   2.5006001   -2.357621    3.065641 -2.27397713
## obs5   0.000000  -0.4926896   0.0000000    0.000000   -1.174237 -0.23591305
## obs6   0.000000   0.0000000   0.0000000    0.000000    1.222060  0.03859869
##      X110.606m.z X112.626m.z X114.646m.z X116.667m.z X118.687m.z X120.707m.z
## obs1   0.0000000  -4.9616547   0.6621267   2.4832033   0.0000000   0.4200961
## obs2  -0.4146838   1.3933288   3.1947586   0.0000000   0.2581256   2.3065666
## obs3  -2.8629205   3.0534229   1.4699825  -0.6085676  -1.8094082   0.1110803
## obs4   0.0000000   0.7650952   1.7290683  -0.2325731   0.0000000   2.9893068
## obs5   0.0000000   0.0000000   2.4108620   2.1752914  -0.5061462   3.4052008
## obs6  -0.8977424   0.0000000   1.0278843   0.0000000  -1.8269462   0.0000000
##      X122.727m.z X124.747m.z X126.768m.z X128.788m.z X130.808m.z X132.828m.z
## obs1   0.0000000  -3.5070609   0.0000000  -2.2591287  -1.6145784  -0.2222218
## obs2   0.2003260   0.0000000   5.3721502   0.0000000   0.0000000   0.0000000
## obs3  -0.8348974  -1.1708942  -0.4059908   0.6166674   0.6435734   1.0373043
## obs4  -4.1047855   0.0000000  -2.8644279   0.5157724   0.8711420   0.2524768
## obs5   0.0000000  -0.9898872   1.2775637   0.4083264   0.0000000   0.0000000
## obs6  -3.9214546   0.0000000  -1.1541469   2.4650556   3.8122354   0.0000000
##      X134.848m.z X136.869m.z X138.889m.z X140.909m.z  X142.929m.z X144.949m.z
## obs1    0.000000  0.00000000  -0.3179946    0.000000  0.000000000    0.000000
## obs2    2.190645 -0.01116674  -1.7339343    0.000000  1.013280333   -1.370315
## obs3    0.000000 -1.12957130  -1.6836150   -3.548956  0.007162986    0.000000
## obs4    1.449168  0.00000000   0.6652859    0.000000  0.000000000    0.000000
## obs5   -2.534034  0.00000000  -2.1242171    0.000000  0.000000000    0.000000
## obs6   -1.149559  0.83753932  -1.5124967   -2.301221 -1.845975342    0.000000
##      X146.97m.z X148.99m.z X151.01m.z X153.03m.z X155.051m.z X157.071m.z
## obs1  0.6228329  0.3718362   1.296503   0.000000   0.0000000  -0.1992737
## obs2  0.0000000  0.9476063   1.241180  -1.250780  -2.3648988  -0.6586787
## obs3  0.0000000  1.1681773   0.000000   2.657553  -0.3799528   0.0000000
```

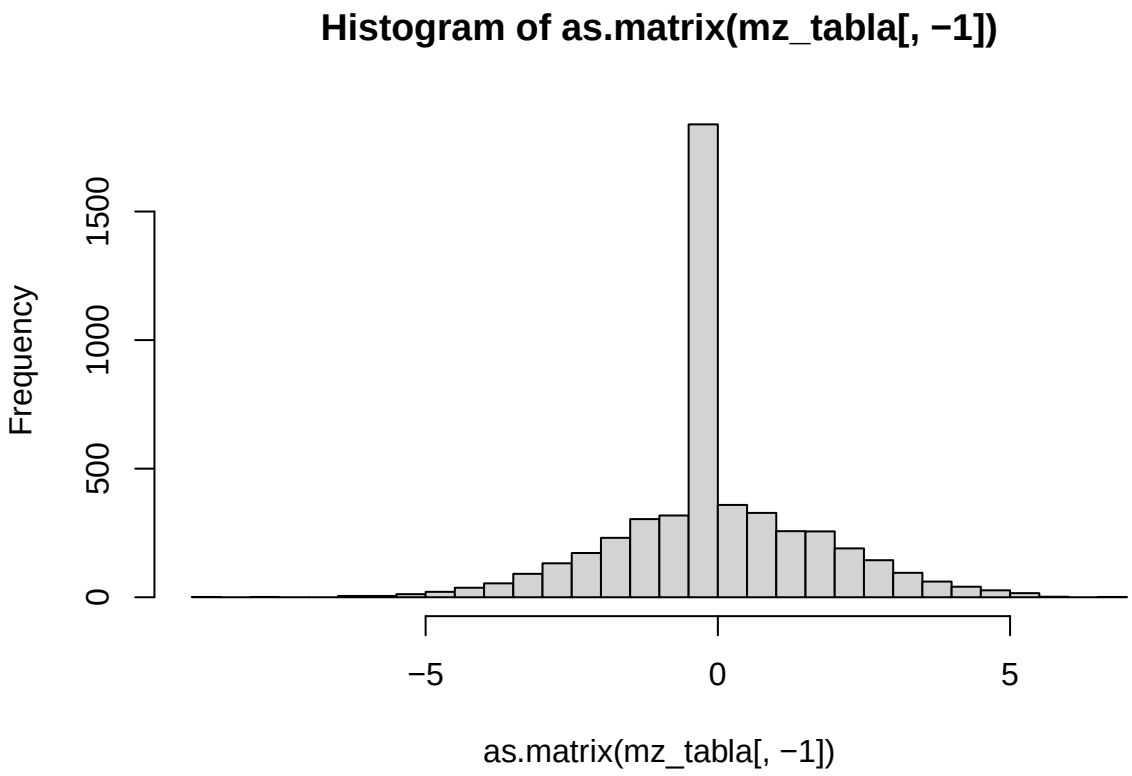
```
## obs4  0.0000000  0.7114281   1.769929  -1.597322  -2.0183154   4.2933619
## obs5  1.4031797 -3.1627742  -1.786523  -2.969469   0.0000000   2.3013091
## obs6  0.0000000  0.0000000   2.088286  -4.392641   0.0000000   0.0000000
##      X159.091m.z X161.111m.z X163.131m.z X165.152m.z X167.172m.z X169.192m.z
## obs1   0.0000000  -1.1281823   -2.335152 -0.02798184  -0.5895145   -1.516232
## obs2  -0.7514936   0.0000000    0.000000  4.10114191  -0.2728954   -1.685232
## obs3   0.0000000  -0.3957398    3.371374  2.09008241  -4.0856110   -1.196596
## obs4  -1.0025471   0.0000000    0.000000  0.00000000   0.0000000    1.680114
## obs5   4.8160783   0.0000000    2.041888  0.00000000   1.6075101   -2.082563
## obs6   0.5245136  -3.4376212    0.000000  0.00000000   0.0000000   -1.901551
##      X171.212m.z X173.232m.z X175.253m.z X177.273m.z X179.293m.z X181.313m.z
## obs1  -0.3915682  -2.2489513   1.7012266   0.1276565    4.095011   -2.611172
## obs2   0.8080812  -0.5692991   0.0000000  -0.2466469    0.000000   -1.190566
## obs3  -0.8693359  -0.9338455   3.7667588   0.0000000    1.447207   -2.504839
## obs4  -4.3177023   0.0000000   0.4291786   2.4209663    0.000000    2.974790
## obs5   0.0000000   0.8863137  -2.9728737   0.0000000   -3.484328    0.000000
## obs6  -0.6842793  -2.9939236   0.0000000  -2.7342106    0.000000    2.250891
##      X183.333m.z X185.354m.z X187.374m.z X189.394m.z X191.414m.z X193.434m.z
## obs1  -1.3340998   0.1134276    0.325511   0.7108966   0.0000000   0.0000000
## obs2   0.7339565  -1.4795206    0.000000   0.0000000   1.2188532  -0.8616321
## obs3  -3.9449611   0.0000000   -1.440518  -0.8332640  -0.1450090   0.0000000
## obs4   0.0000000  -2.4616069    0.000000   0.0000000   1.6178779   0.0000000
## obs5   3.0749854   2.0196282    1.915987   0.4220122  -0.2479638   2.8928955
## obs6  -0.5028535  -1.2395139    0.000000   0.2492900   0.0000000   0.2151277
##      X195.455m.z X197.475m.z X199.495m.z X201.515m.z X203.535m.z X205.556m.z
## obs1   0.0000000   0.0000000   0.8363056   0.6713203   2.5809570    2.329109
## obs2  -0.3355380   0.9421598  -0.8978864   0.0000000   0.0000000    0.000000
## obs3   4.7340746   4.8413077  -0.9709077   0.0000000   1.4113943    1.957221
## obs4  -0.3187038  -2.0573464   0.0000000   0.0000000   0.3028183    0.000000
## obs5  -2.1037409   0.0000000   3.9496848  -0.7192418  -1.7132518    0.000000
## obs6  -1.8061955  -2.2657964  -3.6522849   0.0000000   0.6746847   -1.347553
##      X207.576m.z X209.596m.z X211.616m.z X213.636m.z X215.657m.z X217.677m.z
## obs1   0.0000000   0.0000000   0.8389748   0.0000000   0.9389506  -0.3711466
## obs2   0.0000000  -0.9485082  -2.9783756  -0.9142364   0.0000000   0.0000000
## obs3  -0.1683552   1.2897262  -0.8719404  -1.0316291   0.7247943   0.0000000
## obs4  -2.0068917  -1.5252326  -1.0994904   0.4930276  -1.0988277  -0.3540121
## obs5   0.0000000   0.0000000  -2.8897492   0.0000000   0.0000000   0.0000000
## obs6   0.0000000  -0.1898367   0.0000000   0.0000000  -0.3718210   0.1057253
##      X219.697m.z X221.717m.z X223.737m.z X225.758m.z X227.778m.z X229.798m.z
## obs1   1.6067017   0.0000000   1.6003566  0.00000000   0.4699884  -4.0573440
## obs2  -1.3920361   2.0379413   2.7964048 -2.80330341   1.9609405  -2.6097233
## obs3  -0.3536187  -1.2627342   0.5567398  1.09484070  -6.2067145   0.0000000
## obs4   0.0000000   3.0520614  -1.9361418 -6.06389632   0.3846161   0.0000000
## obs5  -1.3563909  -0.4664681  -0.5601929  0.00000000  -2.0761255   0.0000000
## obs6   1.8019670   0.0000000  -2.2512236  0.03708497   0.3728533  -0.1252578
##      X231.818m.z X233.838m.z X235.859m.z X237.879m.z X239.899m.z X241.919m.z
## obs1   -2.628300    0.000000   3.1300794   0.0000000   4.0677407   0.0000000
## obs2    0.000000   -1.930223  -0.5428408  -0.6640057   0.0000000  -0.1210308
## obs3    3.267342    0.000000  -0.7460442   1.2751498   0.0000000  -0.8693399
## obs4    0.777673    0.000000  -3.1696938   0.0000000   2.3822645   0.0000000
## obs5    1.347656    0.000000  -2.5618093   0.7658205  -1.6896833   0.1973304
## obs6    1.710104    1.842641   0.1578963  -0.5359306   0.7961171  -0.4206866
##      X243.939m.z X245.96m.z X247.98m.z    X250m.z
## obs1   3.2666717 -3.0878662 -2.8188670 -0.9784915
```

```
## obs2   3.2397055  0.0000000 -1.0908588  0.1185388
## obs3   0.0000000 -0.4871793 -1.1857935  1.9426481
## obs4  -0.2421375 -2.4501181 -0.3274923  0.0000000
## obs5   0.0000000 -1.4793231  0.0000000  0.9929577
## obs6   2.6329761  0.0000000 -1.2921051  0.0000000
```

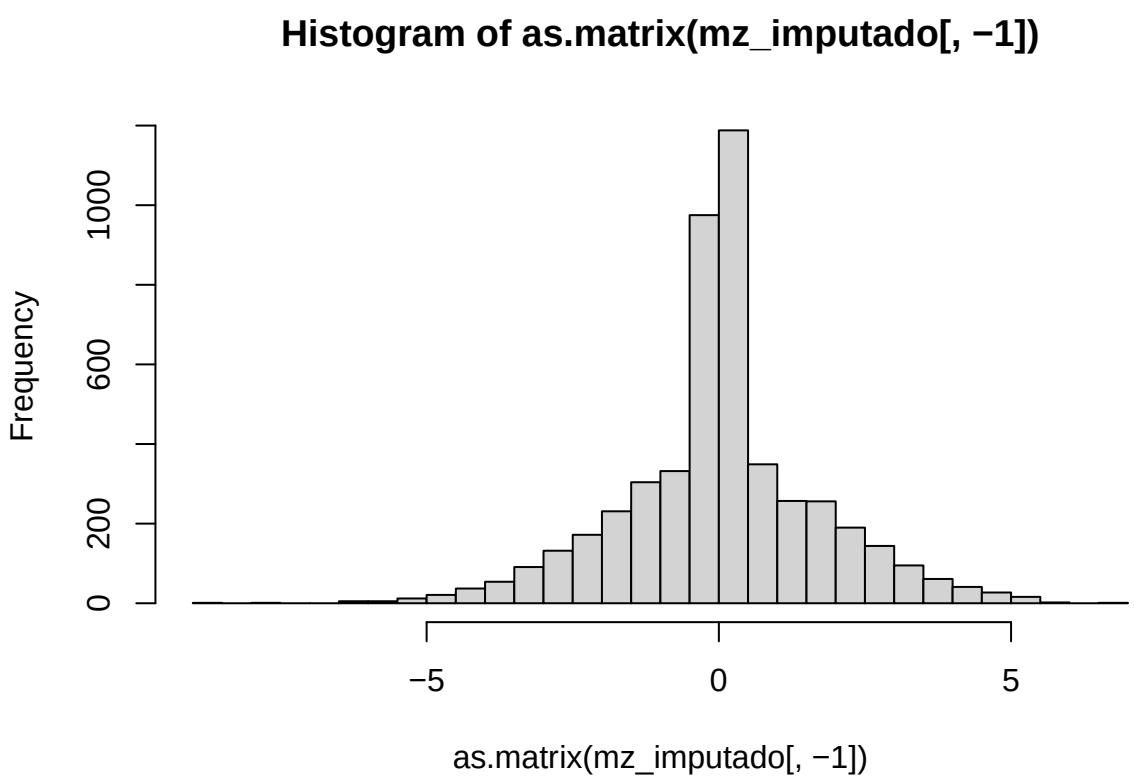
```
media_intensidad<-sapply(mz_tabla[,-1],mean,simplify = TRUE)#sapply entrega el resultado de una
#operación aplicada sobre cada columna de un dataframe. El promedio de las
#intensidades de cada m/z

##for...
#Permite repetir recursivamente una operación por un número dado de ciclos
mz_imputar<-mz_tabla[,-1]
#Vamos a repetir una operación de 1 al 100 (el número de columnas)
for (i in 1:length(media_intensidad)){
  mz_imputar[,i][mz_imputar[,i]==0]<-media_intensidad[i]
}

mz_imputar[,1][mz_imputar[,1]==media_intensidad[1]]
```

```
##  [1] 0.2360449 0.2360449 0.2360449 0.2360449 0.2360449 0.2360449 0.2360449
##  [8] 0.2360449 0.2360449 0.2360449 0.2360449 0.2360449 0.2360449 0.2360449
## [15] 0.2360449 0.2360449 0.2360449
```

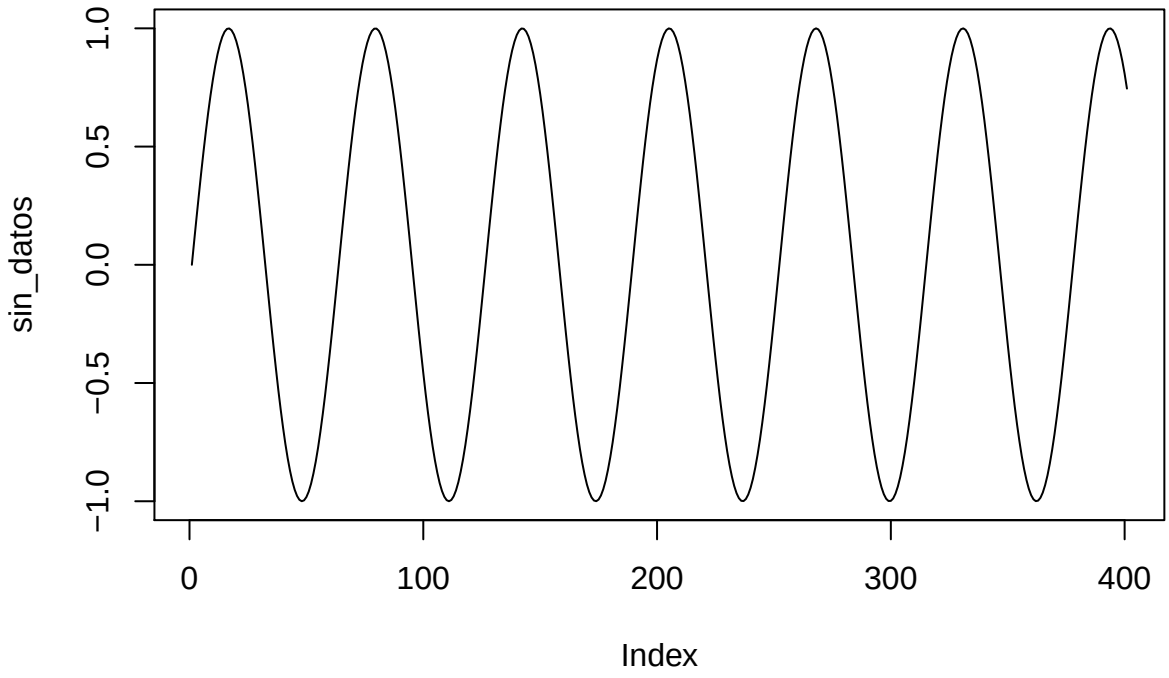
```
mz_imputado<-data.frame(obs_nomb,mz_imputar)##Dataframe imputado con la media

###HISTOGRAMAS EN R####
hist(as.matrix(mz_tabla[,-1]),breaks = 25)
```



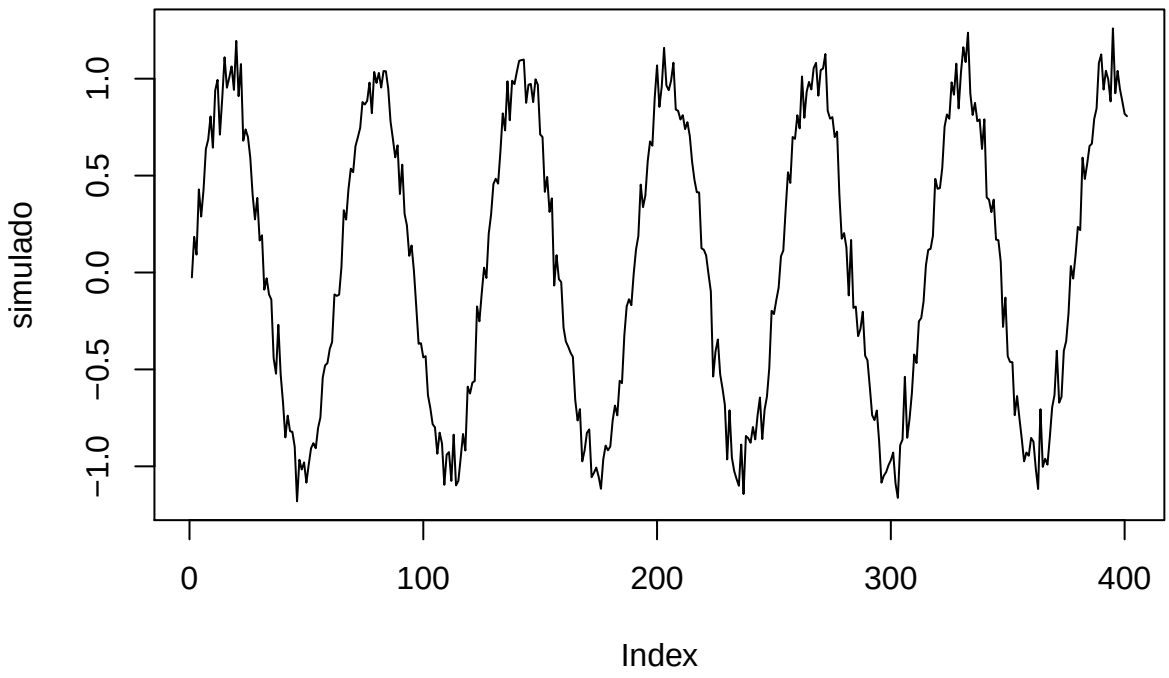

```
hist(as.matrix(mz_imputado[,-1]),breaks = 25)
```




```r
#as.matrix se conoce como función de coerción. Las funciones de coerción
#obligan al programa a interpretar un objeto como uno de una clase determinada
#funciones de coerción comunes: as.matrix(), as.character(), as.list(), as.data.frame()

#Ejercicio: crear un vector que simule datos periódicos con un error aleatorio
#y con regiones mínimas planas.
x<-seq(from=0, to=40,by=0.1)
sin_datos<-sin(x)
plot(sin_datos,type = "l")
```




```
error<-rnorm(length(sin_datos),mean=0,sd=0.1)
simulado=sin_datos+error
plot(simulado,type="l")
```




```
min(simulado)
```

```
## [1] -1.180082
```

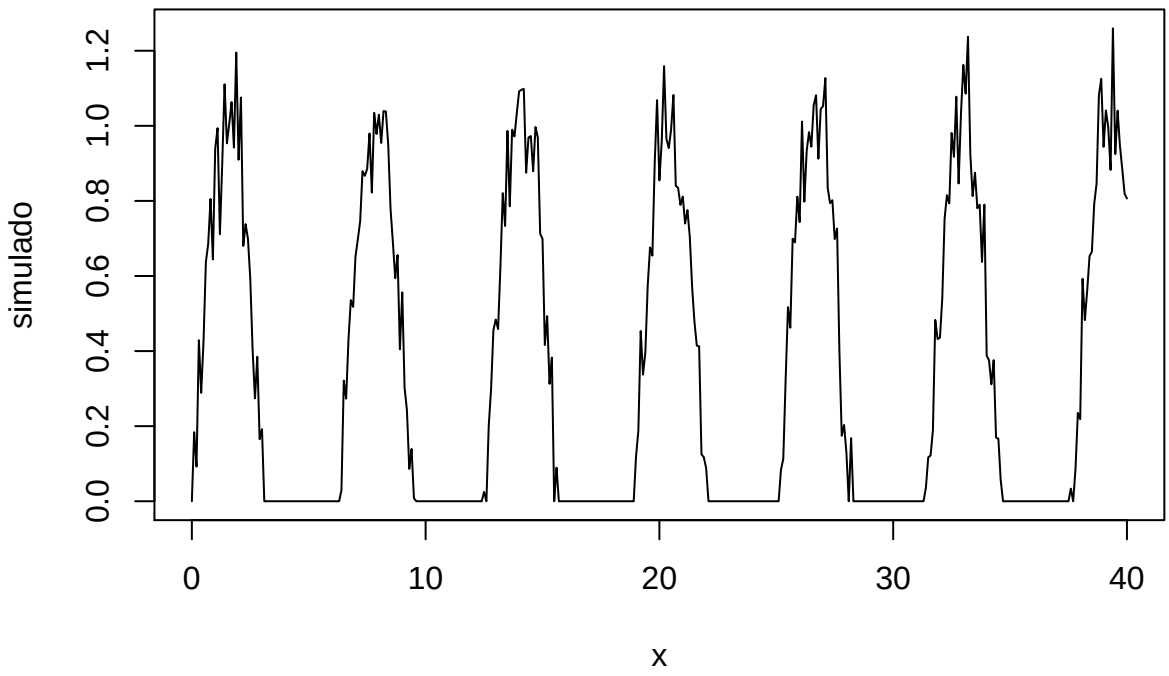
```
simulado[simulado<=0]<-0
plot(x,simulado,type="l")

##FIN DE SESIÓN####
##Librerias necesarias (instalar con calma)
library(MASS)
library(rgl)
library(plotrix)
```

```
##
## Attaching package: 'plotrix'
```

```
## The following object is masked from 'package:rgl':
##
##     mtext3d
```

```
library(Rtsne)
#library(umap)
library(uwot) #otra implementacion
```

```
## Loading required package: Matrix
```




```
library(ica)
library(vegan)
```

```
## Loading required package: permute
```

```
## Loading required package: lattice
```

```
## This is vegan 2.5-7
```

```
library(ggplot2)
library(factoextra)
```

```
## Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve3WBa
```

```
library(e1071)
library(vegan)
```