



Dia 2. Introducción al R básico

Acceso a datos en tablas.

Funciones básicas. Estructuras de control.

Visualización de datos geográficos en R.

Universidad de Guadalajara, CUCSH–CUCBA

Viacheslav Shalisko

16–24.07.2019

Datos fuente

La tabla de datos Datos_del_censo.csv – contiene fragmento de estudio del arbolado con $DAP \geq 5$ cm en los Centros Universitarios de la UdeG (datos del 2016)

Así se ve el código en el editor de RStudio

El resultado de ejecución con el compilador *Knit*

```
11
12 ▾ ### A. Cargar la tabla de datos
13 ▾ #### Estructura de datos (tabla `Datos_del_censo.csv`):
14 1. *Centro* - código del CU
15 2. *Especie* - nombre científico
16 3. *Codigo* - identificador único del árbol
17 4. *AB* - área basal del árbol (dm2)
18 5. *DTr* - diametro del tronco equivalente (cm)
19 6. *Alt* - estatura del árbol (m)
20 7. *DCop* - diametro promedio de la copa (m)
21 8. *ExcCop* - excentricidad de la copa
22
23 ▾ ```{r}
24 arbolado <- read.csv("Materiales/Datos_del_censo.csv")
25 dim(arbolado)
26 ▾ ```
27
28
29
30
31
32
33
```

fragmento del código R ("chunk")

A. Cargar la tabla de datos

Estructura de datos (tabla Datos_del_censo.csv):

1. Centro - código del CU
2. Especie - nombre científico
3. Codigo - identificador único del árbol
4. AB - área basal del árbol (dm²)
5. DTr - diametro del tronco equivalente (cm)
6. Alt - estatura del árbol (m)
7. DCop - diametro promedio de la copa (m)
8. ExcCop - excentricidad de la copa

```
arbolado <- read.csv("Materiales/Datos_del_censo.csv")
dim(arbolado)
```

```
## [1] 4785 8
```


Datos fuente

`head(arbolado)` ← Verificación de calidad de lectura de tabla

##	Centro	Especie	Codigo	AB	DTr	Alt	DCop	ExcCop
## 1	CUCOSTA	Pithecellobium dulce	C10-3-1	13.21	41.0	9.8	15.4	0.30
## 2	CUCOSTA	Acacia macracantha	C10-3-10	1.91	15.6	4.4	4.7	0.63
## 3	CUCOSTA	Pithecellobium lanceolatum	C10-3-100	0.95	11.0	6.4	9.5	0.91
## 4	CUCOSTA	Salix bonplandiana	C10-3-1000	0.83	10.3	7.7	4.9	0.20
## 5	CUCOSTA	Tabebuia rosea	C10-3-1001	0.28	6.0	2.4	2.0	0.63
## 6	CUCOSTA	Pithecellobium lanceolatum	C10-3-1002	0.51	8.1	4.1	4.3	0.71

`tail(arbolado)`

##	Centro	Especie	Codigo	AB	DTr	Alt	DCop	ExcCop
## 4780	CUCSUR	Araucaria heterophylla	C9-3-94	6.97	29.8	16.2	5.5	0.50
## 4781	CUCSUR	Ficus benjamina	C9-3-95	17.60	47.3	11.5	10.2	0.38
## 4782	CUCSUR	Simarouba glauca	C9-3-96	2.63	18.3	11.5	9.4	0.58
## 4783	CUCSUR	Jacaranda mimosifolia	C9-3-97	76.95	99.0	20.6	13.0	0.57
## 4784	CUCSUR	Jatropha cordata	C9-3-98	2.11	16.4	2.7	3.6	0.23
## 4785	CUCSUR	Jacaranda mimosifolia	C9-3-99	7.55	31.0	18.2	5.2	0.77

Datos fuente

Estructura del objeto *arbolado*

```
str(arbolado)
```

```
## 'data.frame':    4785 obs. of  8 variables:
## $ Centro : Factor w/ 4 levels "CUALTOS","CUCIENEGA",...: 3 3 3 3 3 3 3 3 3 3 ...
## $ Especie: Factor w/ 150 levels "Acacia farnesiana",...: 104 4 105 127 139 105 105 105 139 127 ...
## $Codigo  : Factor w/ 4780 levels "C10-3-1","C10-3-10",...: 1 2 3 4 5 6 7 8 9 10 ...
## $ AB      : num  13.21 1.91 0.95 0.83 0.28 ...
## $ DTr      : num  41 15.6 11 10.3 6 8.1 11.6 9 6 15.8 ...
## $ Alt      : num  9.8 4.4 6.4 7.7 2.4 4.1 4.4 4.5 4.3 11.6 ...
## $ DCop     : num  15.4 4.7 9.5 4.9 2 4.3 5.2 4.3 2.5 5.1 ...
## $ ExcCop   : num  0.3 0.63 0.91 0.2 0.63 0.71 0.84 0.84 0.75 0.42 ...
```

```
levels(arbolado$Centro)
```

```
## [1] "CUALTOS" "CUCIENEGA" "CUCOSTA" "CUCSUR"
```

Subconsultas

Más ejemplos de extracción desde un **dataframe**:

```
arbolado[arbolado$DTr > 10,]
```

– extraer registros para arbolado con DAP mayor que 10 cm

```
arbolado[arbolado$Centro == 'CUCSUR',]
```

– extraer registros para arbolado de CUCSUR

```
arbolado[arbolado[,1] == 'CUCSUR',]
```

– lo mismo

```
arbolado[arbolado$Centro == 'CUCSUR' & arbolado$DTr > 10,]
```

– extraer registros para arbolado de CUCSUR con DAP > 10 cm

```
arbolado[arbolado$Centro == 'CUCSUR' & arbolado$DTr > 10,  
                                                c("AB", "Alt")]
```

– extraer área basal y estatura para arbolado de CUCSUR con DAP > 10 cm

summary

```
summary(arbolado[,c(1,2,4:8)])
```

Seleccionamos solo columnas de interés

##	Centro	Especie	AB	
##	CUALTOS :1828	Quercus resinosa : 530	Min. : 0.200	
##	CUCIENEGA: 445	Pithecellobium lanceolatum: 501	1st Qu.: 0.950	
##	CUCOSTA :2143	Guazuma ulmifolia : 329	Median : 2.780	
##	CUCSUR : 369	Fraxinus uhdei : 278	Mean : 5.817	
##		Eysenhardtia polystachya : 245	3rd Qu.: 6.610	
##		Acacia macracantha : 176	Max. :456.340	
##		(Other) :2726		
##	DTr	Alt	DCop	ExcCop
##	Min. : 5.00	Min. : 1.500	Min. : 0.200	Min. :0.0000
##	1st Qu.: 11.00	1st Qu.: 4.200	1st Qu.: 3.800	1st Qu.:0.3700
##	Median : 18.80	Median : 6.200	Median : 5.500	Median :0.5100
##	Mean : 22.22	Mean : 7.121	Mean : 6.165	Mean :0.5026
##	3rd Qu.: 29.00	3rd Qu.: 9.100	3rd Qu.: 7.900	3rd Qu.:0.6500
##	Max. :241.00	Max. :26.600	Max. :64.500	Max. :0.9900
##		NA's :46		

Variable sin datos

mean, median, sd, var, quantile

Las funciones básicas de la estadística descriptiva incluyen:

1. Promedio y mediana - `mean()`, `median()`
2. Mínimo, máximo y rango - `min()`, `max()`, `range()`
3. Desviación estandar y varianza - `sd()`, `var()`
4. Cuantiles y rango intercuartilico - `quantile()`, `IQR()`

```
summary(arbolado$AB)
```

```
##      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    0.200   0.950   2.780   5.817   6.610  456.300
```

```
range(arbolado$AB)
```

```
## [1]    0.20 456.34
```

```
quantile(arbolado$AB)
```

```
##      0%      25%      50%      75%     100%
##    0.20    0.95    2.78    6.61  456.34
```

```
IQR(arbolado$AB)
```

```
## [1] 5.66
```

```
var(arbolado$AB)
```

```
## [1] 173.8403
```

```
sqrt(var(arbolado$AB))
```

```
## [1] 13.18485
```

```
sd(arbolado$AB)
```

```
## [1] 13.18485
```

quantile, boxplot.stats

Función *quantile* se puede emplear para subdividir la distribución de valores bajo reglas arbitrarias, determinar cuartiles, percentiles o cualquier otra secuencia de probabilidades.

```
probabilidades <- c(0.01,0.05,0.1,0.5,0.9,0.95,0.99)
quantile(arbolado$AB, probs = probabilidades)
```

##	1%	5%	10%	50%	90%	95%	99%
##	0.200	0.270	0.410	2.780	12.956	19.630	44.436

Función *boxplot.stats* puede ser útil para discriminar los valores atípicos. Este procedimiento depende del umbral *coef* definido en los parámetros, valor por defecto *coef*=1.5

```
boxplot.stats(arbolado$AB)$stats
```

```
## [1] 0.20 0.95 2.78 6.61 15.00
```

```
boxplot.stats(arbolado$AB, coef = 3)$stats
```

```
## [1] 0.20 0.95 2.78 6.61 23.52
```


aggregate

Estadísticas por grupos se puede calcular empleando función *aggregate*.

Nota que los grupos deben estar definidos como variable de tipo *factor*.

```
aggregate(arbolado$AB ~ arbolado$Centro, FUN = sum)
```

```
##   arbolado$Centro arbolado$AB
## 1         CUALTOS      8521.96
## 2       CUCIENEGA      3478.53
## 3       CUCOSTA     12647.99
## 4       CUCSUR       3184.82
```

```
aggregate(arbolado$AB ~ arbolado$Centro, FUN = length)
```

```
##   arbolado$Centro arbolado$AB
## 1         CUALTOS        1828
## 2       CUCIENEGA         445
## 3       CUCOSTA        2143
## 4       CUCSUR          369
```

aggregate

```
aggregate(arbolado$AB ~ arbolado$Centro, FUN = mean)
```

```
##   arbolado$Centro arbolado$AB
## 1      CUALTOS      4.661904
## 2    CUCIENEGA      7.816921
## 3     CUCOSTA      5.902002
## 4     CUCSUR      8.630949
```

```
aggregate(arbolado$AB ~ arbolado$Centro, FUN = range)
```

```
##   arbolado$Centro arbolado$AB.1 arbolado$AB.2
## 1      CUALTOS           0.20      114.36
## 2    CUCIENEGA           0.23       77.44
## 3     CUCOSTA           0.20     456.34
## 4     CUCSUR            0.21     419.84
```

apply, lapply, sapply, tapply, mapply

Estadística por columnas y/o grupos

El grupo de funciones apply permite realizar operaciones de forma cíclica sin necesidad de definir un ciclo de manera formal:

1. `lapply()` - aplicar una función a cada elemento de una lista, resultado es una lista de mismo largo que lista de entrada
2. `sapply()` - lo mismo que `lapply()`, pero con resultado simplificado (vector, matriz o array)
3. `apply()` - aplicar una función sobre dimensiones de un matriz de datos
4. `tapply()` - aplicar una función sobre grupos definidos en un vector, similar a `aggregate()`
5. `mapply()` - versión multivariante de `tapply()`

Compara

```
tapply(arbolado$AB, arbolado$Centro, FUN = mean)
```

```
##      CUALTOS CUCIENEGA   CUCOSTA   CUCSUR  
##  4.661904  7.816921  5.902002  8.630949
```

```
aggregate(arbolado$AB ~ arbolado$Centro, FUN = mean)
```

```
##      arbolado$Centro arbolado$AB  
## 1           CUALTOS    4.661904  
## 2          CUCIENEGA    7.816921  
## 3           CUCOSTA    5.902002  
## 4           CUCSUR     8.630949
```


lapply, sapply

Compara `lapply()` y `sapply()`

```
lapply(arbolado[,4:8],sd,na.rm = TRUE)
```

```
## $AB
## [1] 13.18485
##
## $DTr
## [1] 15.71243
##
## $Alt
## [1] 3.847538
##
## $DCop
## [1] 3.352365
##
## $ExcCop
## [1] 0.2009095
```

```
sapply(arbolado[,4:8],sd,na.rm = TRUE)
```

```
##      AB      DTr      Alt      DCop      ExcCop
## 13.1848514 15.7124261 3.8475377 3.3523652 0.2009095
```

Nota que se agrega el parámetro para omitir valores sin datos, el mismo parámetro se puede aplicar a funciones individuales.

```
sd(arbolado$Alt, na.rm = TRUE)
```

```
## [1] 3.847538
```

sapply

```
sapply(arbolado[,4:8],range,na.rm = TRUE)
```

```
##           AB DTr  Alt DCop ExcCop  
## [1,]    0.20   5   1.5   0.2   0.00  
## [2,]  456.34 241  26.6  64.5   0.99
```

```
sapply(arbolado[,4:8],function(x) length(x[!is.na(x)]))
```

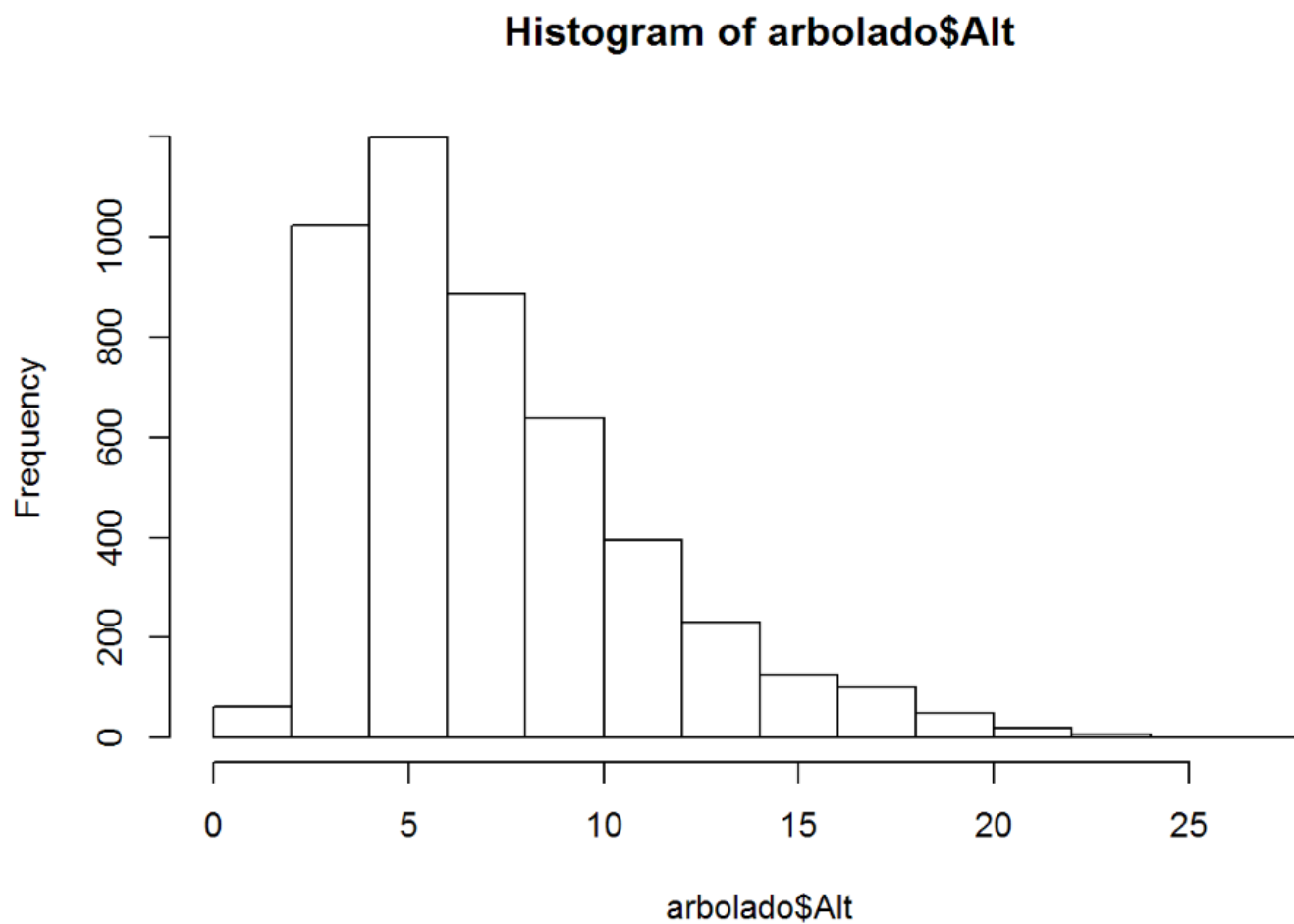
```
##      AB      DTr      Alt      DCop      ExcCop  
##  4785  4785  4739  4785  4785
```

función anónima

A veces se puede definir una *función anónima* en la misma estructura de *apply*. Es el método muy común de uso de las estructuras *apply*.

hist

```
hist(arbolado$Alt)
```



hist

```
hist(arbolado$Alt,  
     col = "lightgray",  
     xlab = "Clases de estatura del árbol (m)", ylab = "Frecuencia",  
     main = "Histograma de estaturas")  
````{r fig.width=8, fig.height=4, warning=FALSE}
```

Parámetros del “chunk”.

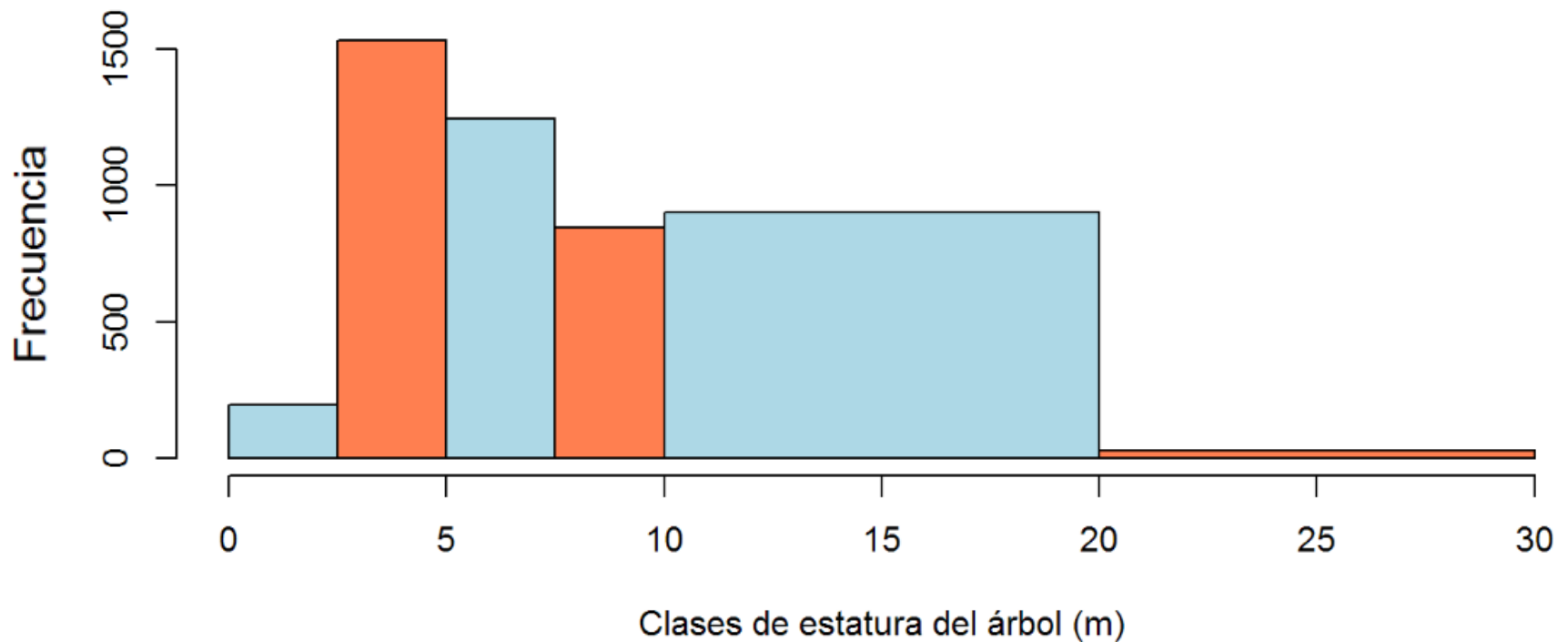
## Histograma de estaturas



# hist

```
hist(arbolado$Alt,
 col = c("lightblue","coral"),
 xlab = "Clases de estatura del árbol (m)", ylab = "Frecuencia",
 main = "Histograma de estaturas",
 breaks = c(0,2.5,5,7.5,10,20,30), prob = FALSE)
```

**Histograma de estaturas**



# hist

```
H0 <- hist(arbolado$Alt,
 col = c("lightblue", "coral"),
 xlab = "Clases de estatura del árbol (m)", ylab = "Frecuencia",
 main = "Histograma de estaturas",
 breaks = c(0, 2.5, 5, 7.5, 10, 20, 30), prob = FALSE)
```

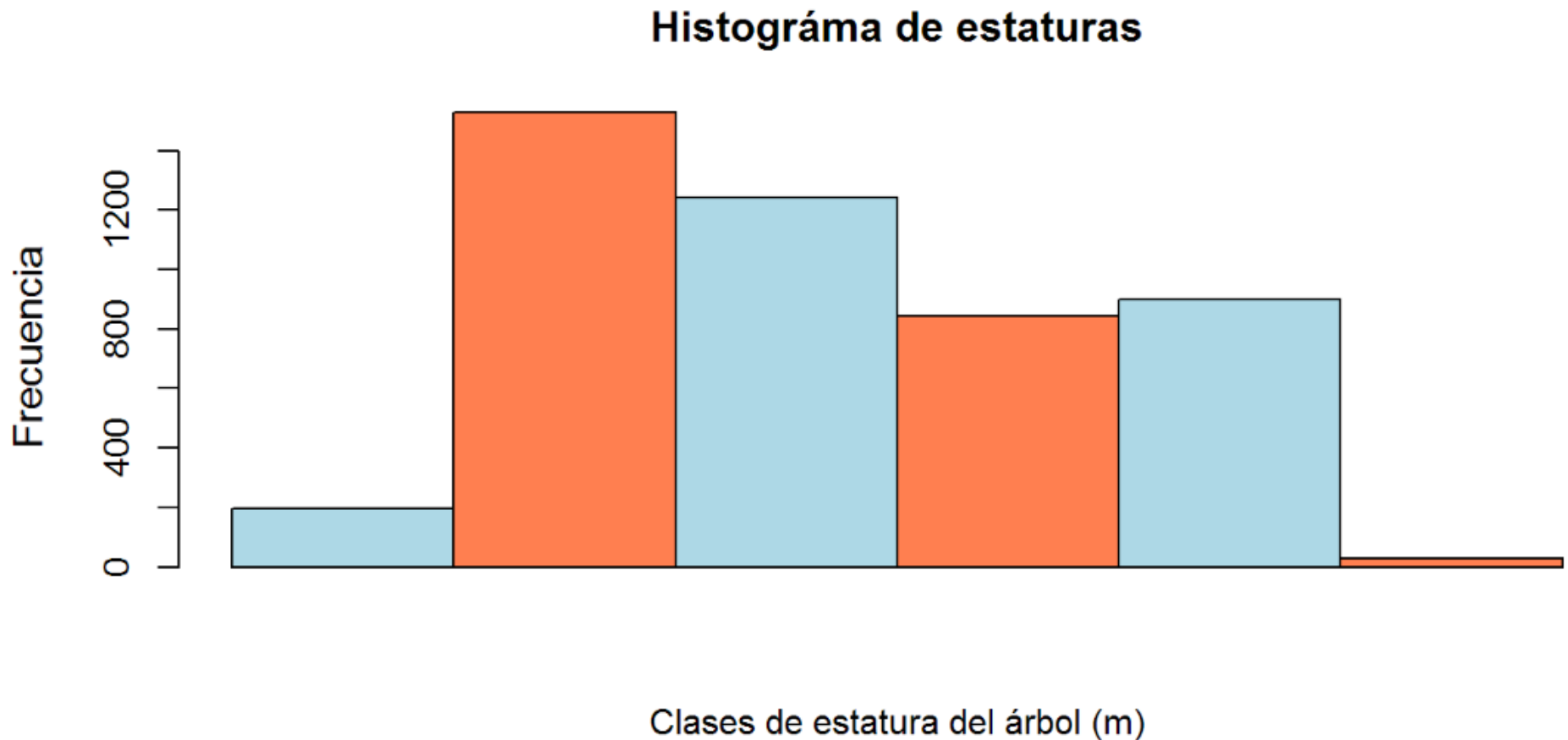
```
str(H0)
```

```
List of 6
$ breaks : num [1:7] 0 2.5 5 7.5 10 20 30
$ counts : int [1:6] 196 1529 1242 844 900 28
$ density : num [1:6] 0.0165 0.1291 0.1048 0.0712 0.019 ...
$ mids : num [1:6] 1.25 3.75 6.25 8.75 15 25
$ xname : chr "arbolado$Alt"
$ equidist: logi FALSE
- attr(*, "class")= chr "histogram"
```



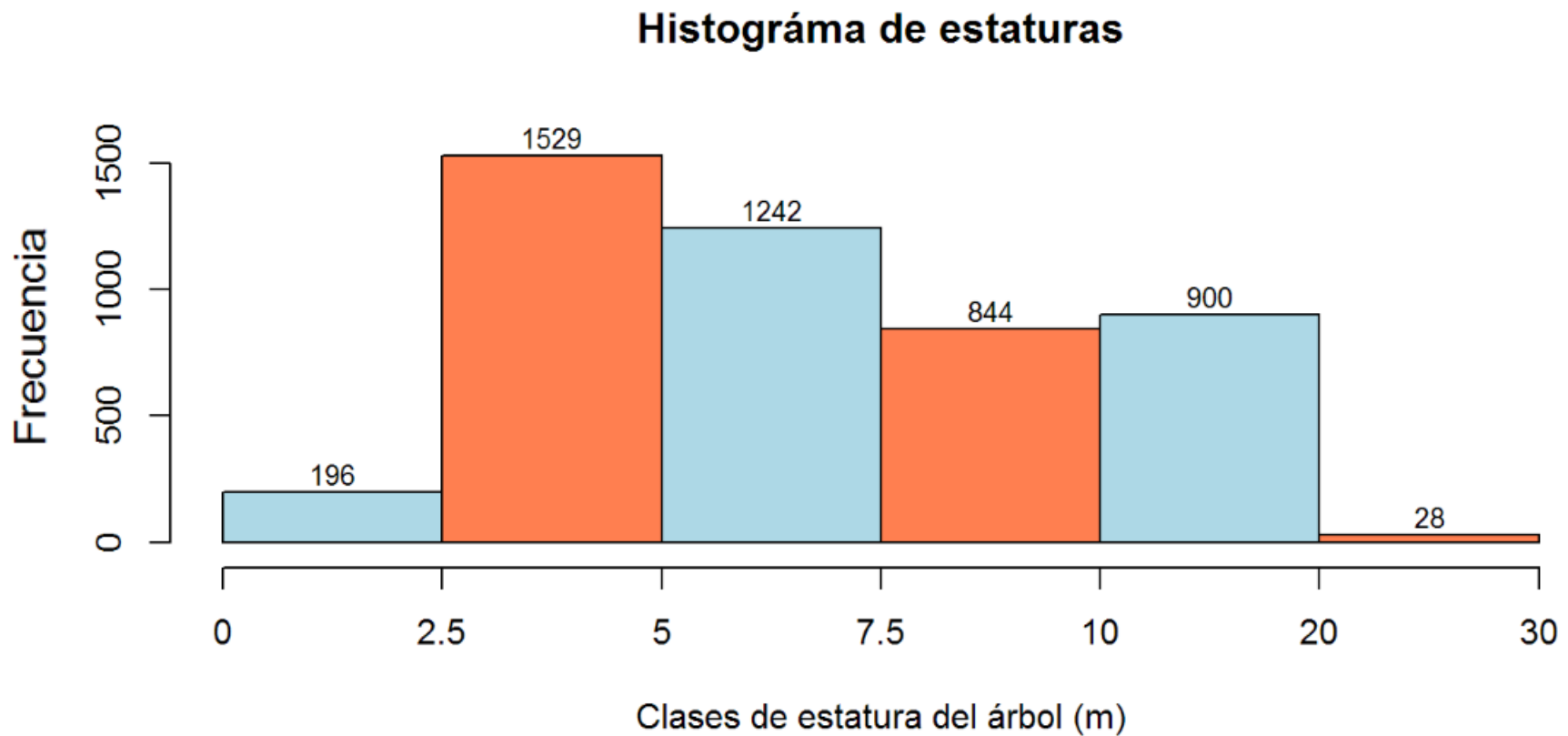
# barplot

```
barplot(H0$counts, space = 0,
 col = c("lightblue", "coral"),
 xlab = "Clases de estatura del árbol (m)", ylab = "Frecuencia",
 main = "Histograma de estaturas")
```



# barplot

```
B0 <- barplot(H0$counts, space = 0, ylim = c(-100,1700),
 col = c("lightblue","coral"),
 xlab = "Clases de estatura del árbol (m)", ylab = "Frecuencia",
 main = "Histograma de estaturas")
axis(at = seq(from = 0, to = 6, by = 1),
 labels = c(0,2.5,5,7.5,10,20,30), side = 1)
text(B0, H0$counts + 0.05 * max(H0$counts), labels=round(H0$counts), cex = 0.8)
```



# plot

La función *plot* ofrece gran flexibilidad en elaboración de las graficas bidimensionales.

Parámetros para colocar 2 gráficas en un cuadro

```
par(mfcol = c(1, 2))

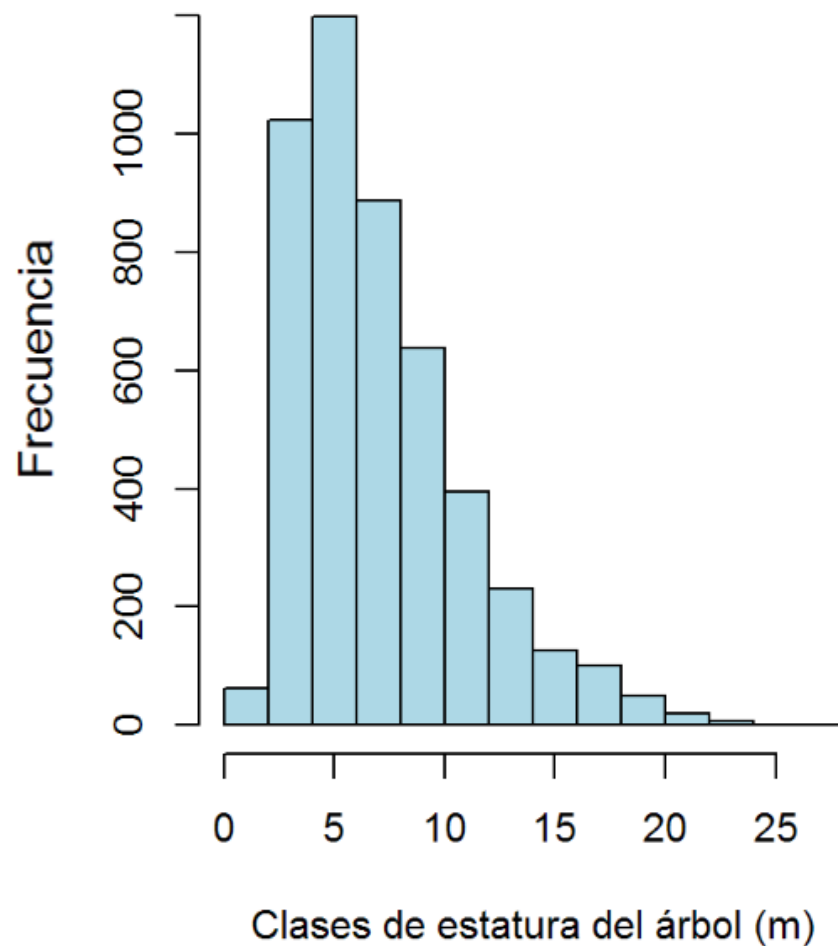
hist(arbolado$Alt,
 col = c("lightblue"),
 xlab = "Clases de estatura del árbol (m)", ylab = "Frecuencia",
 main = "Histograma de estaturas", prob = FALSE)

plot(density(arbolado$Alt, na.rm = TRUE),
 col = c("lightblue"), lty=1, lwd=2,
 xlab = "Estatura del árbol (m)", ylab = "Densidad",
 main = "Distribución de estaturas")
```

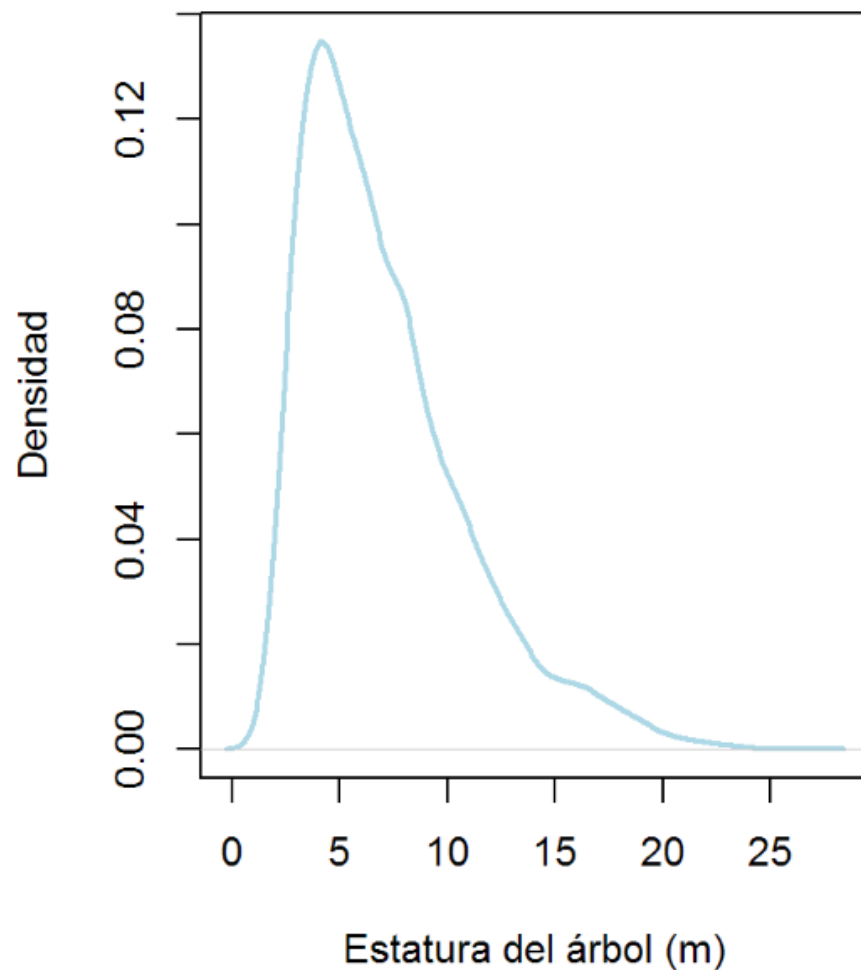


# plot vs. hist

**Histograma de estaturas**



**Distribución de estaturas**

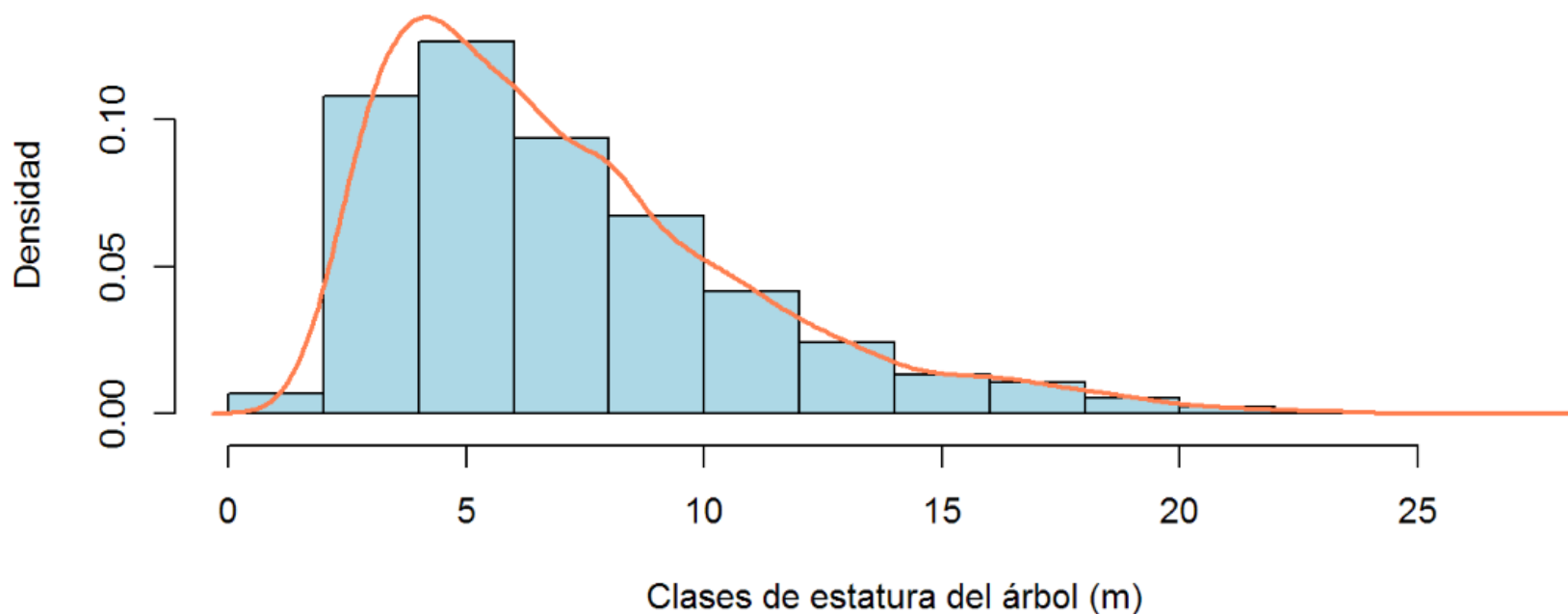


# hist

```
H1 <- hist(arbolado$Alt, ylim = c(-0.005,0.14),
 col = c("lightblue"),
 xlab = "Clases de estatura del árbol (m)", ylab = "Densidad",
 main = "Histograma de estaturas con gráfica de densidad de distribución", prob = TRUE)
```

```
densidad <- density(arbolado$Alt, na.rm = TRUE)
lines(densidad, col = c("coral"), lty=1, lwd=2)
```

## Histograma de estaturas con gráfica de densidad de distribución

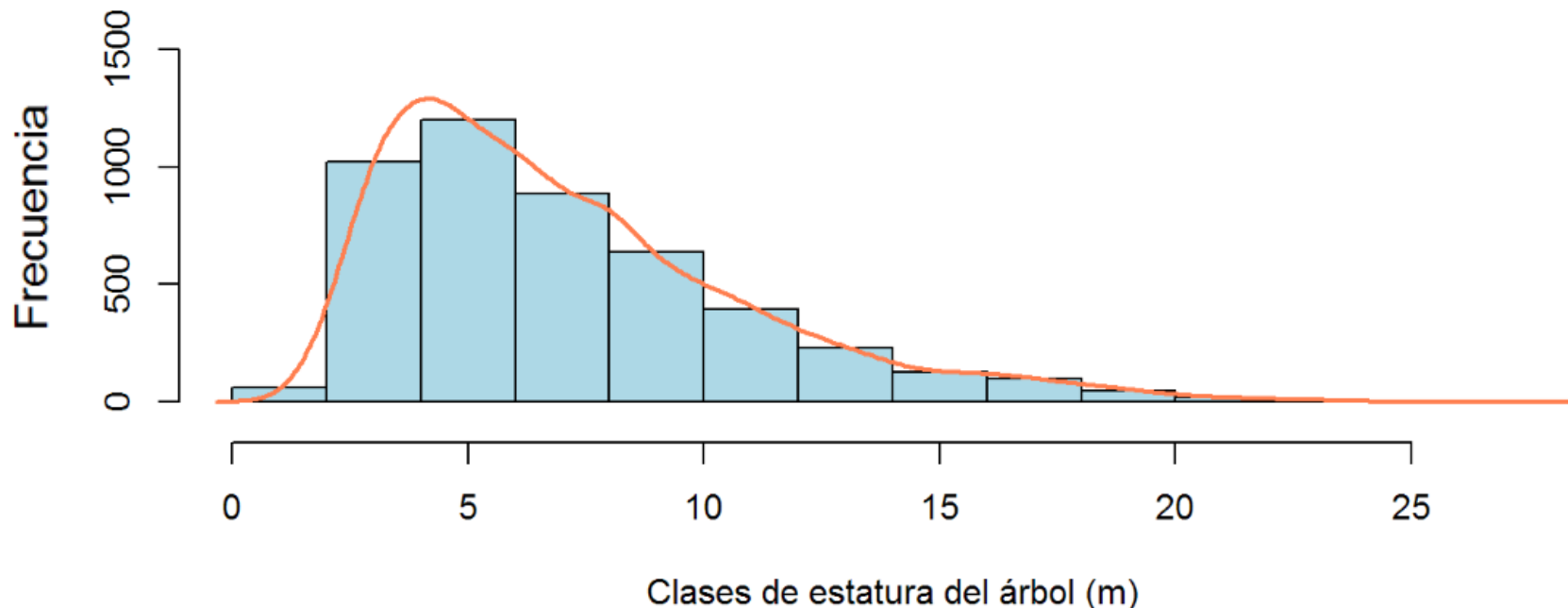


# hist

```
H2 <- hist(arbolado$Alt, ylim = c(-100,1700),
 col = c("lightblue"),
 xlab = "Clases de estatura del árbol (m)", ylab = "Frecuencia",
 main = "Histograma de estaturas con gráfica de densidad de distribución", prob = FALSE)

densidad <- density(arbolado$Alt, na.rm = TRUE)
densidad$y <- densidad$y*diff(H2$mids[1:2])*length(arbolado$Alt)
lines(densidad, col = c("coral"), lty=1, lwd=2)
```

## Histograma de estaturas con gráfica de densidad de distribución





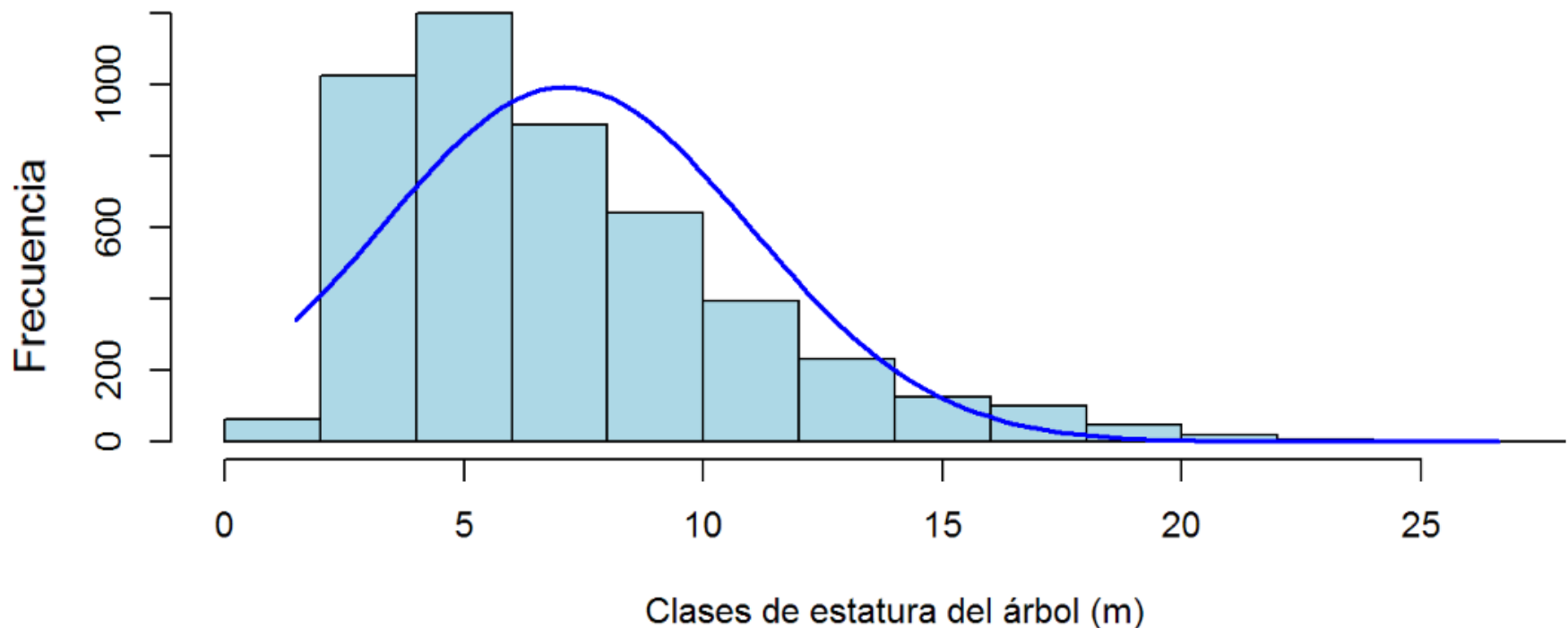
```

plot(H1, col = c("lightblue"),
 xlab = "Clases de estatura del árbol (m)", ylab = "Frecuencia",
 main = "Histograma de estaturas con grafica de distribución normal")

xfit<-seq(min(arbolado$Alt, na.rm = TRUE),max(arbolado$Alt, na.rm = TRUE),length=100)
yfit<-dnorm(xfit,mean=mean(arbolado$Alt, na.rm = TRUE),sd=sd(arbolado$Alt, na.rm = TRUE))
yfit <- yfit*diff(H1$mids[1:2])*length(arbolado$Alt)
lines(xfit, yfit, col="blue", lwd=2)

```

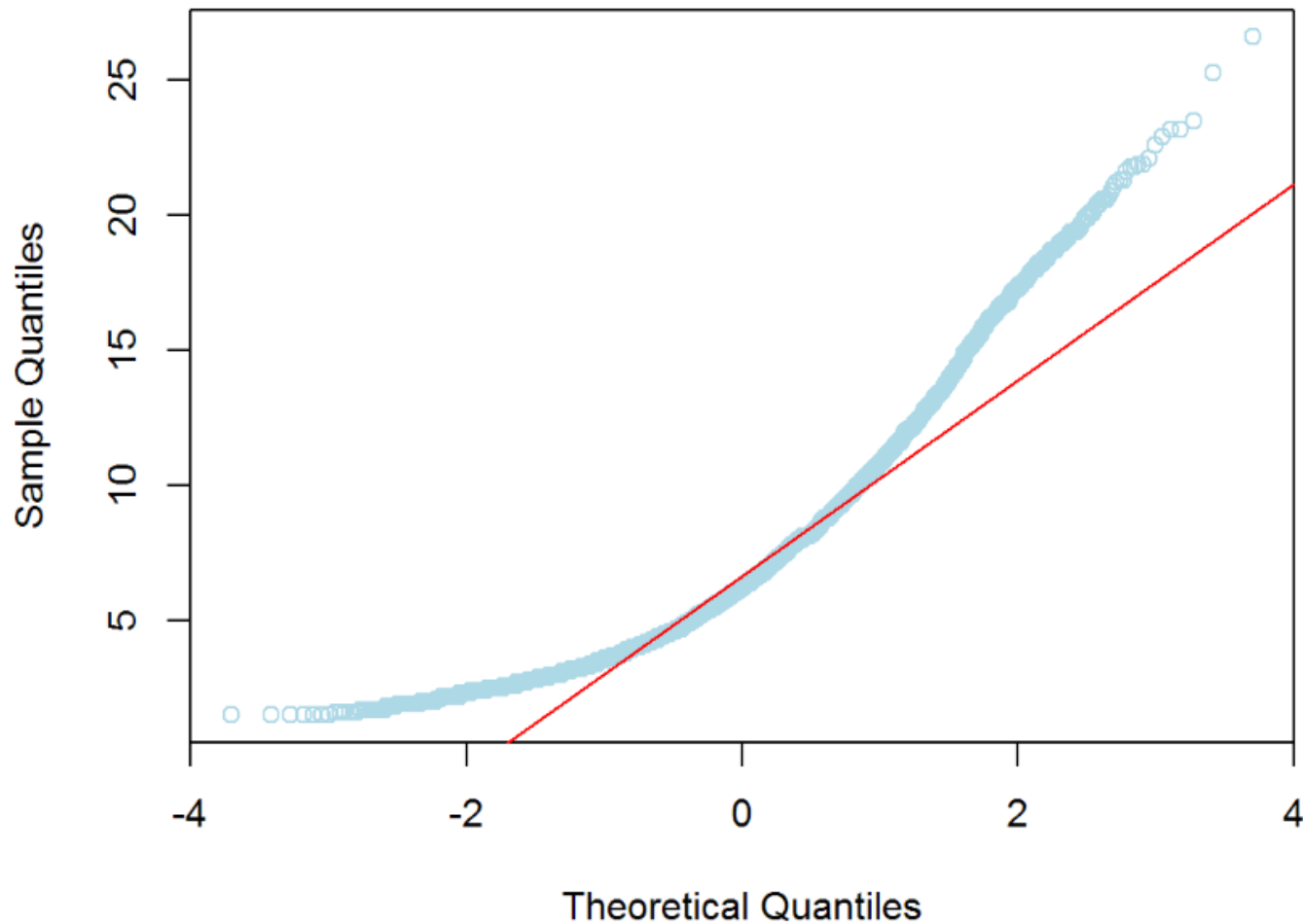
## Histograma de estaturas con grafica de distribución normal



# qqnorm

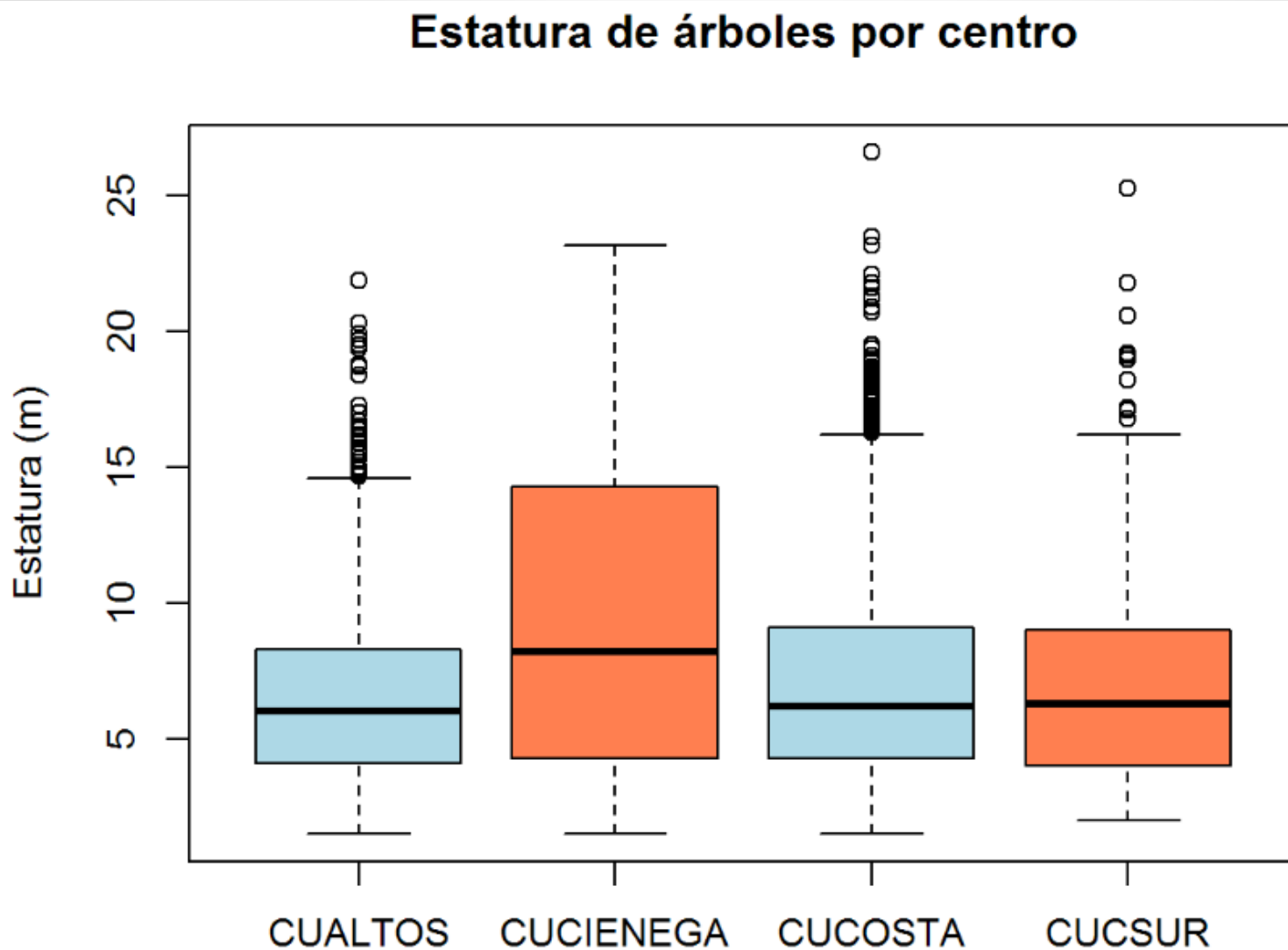
```
qqnorm(arbolado$Alt, col = "lightblue")
qqline(arbolado$Alt, col = "red")
```

Normal Q-Q Plot



# boxplot

```
boxplot(arbolado$Alt ~ arbolado$Centro,
 col = c("lightblue", "coral"),
 main = "Estatura de árboles por centro",
 ylab = "Estatura (m)")
```

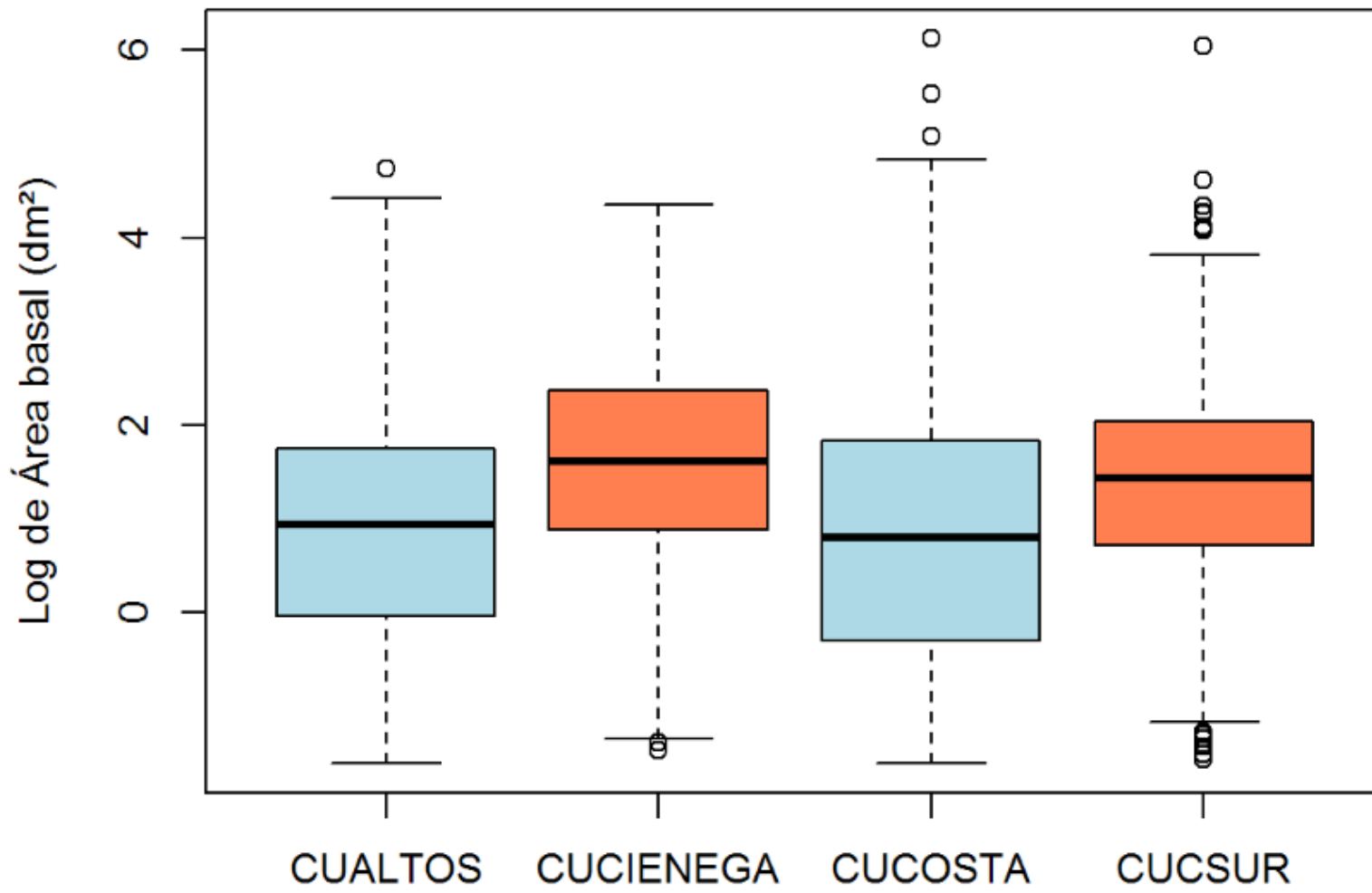




# boxplot

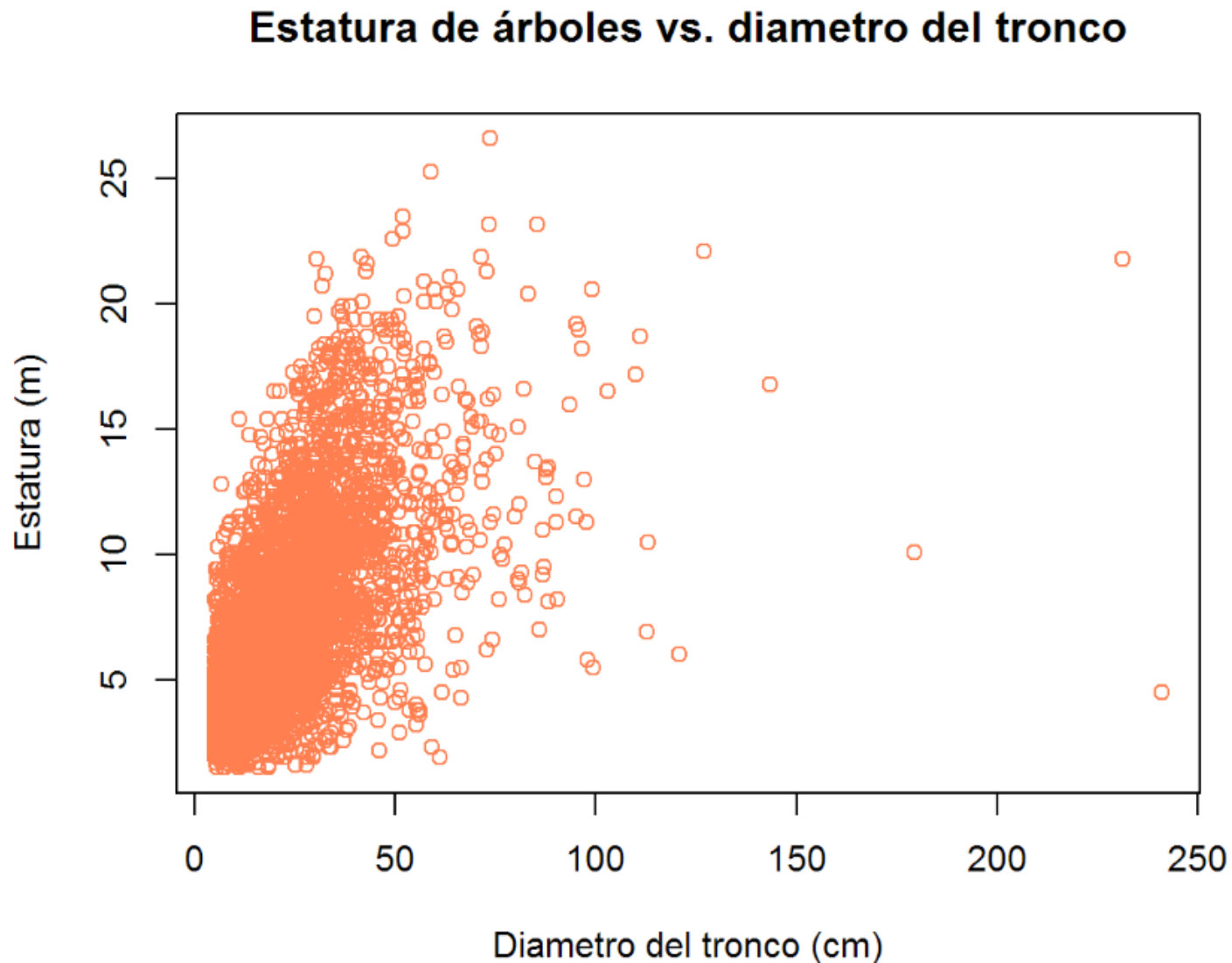
```
boxplot(log(arbolado$AB) ~ arbolado$Centro,
 col = c("lightblue", "coral"),
 main = "Área basal de árboles por centro",
 ylab = "Log de Área basal (dm²)")
```

## Área basal de árboles por centro



plot

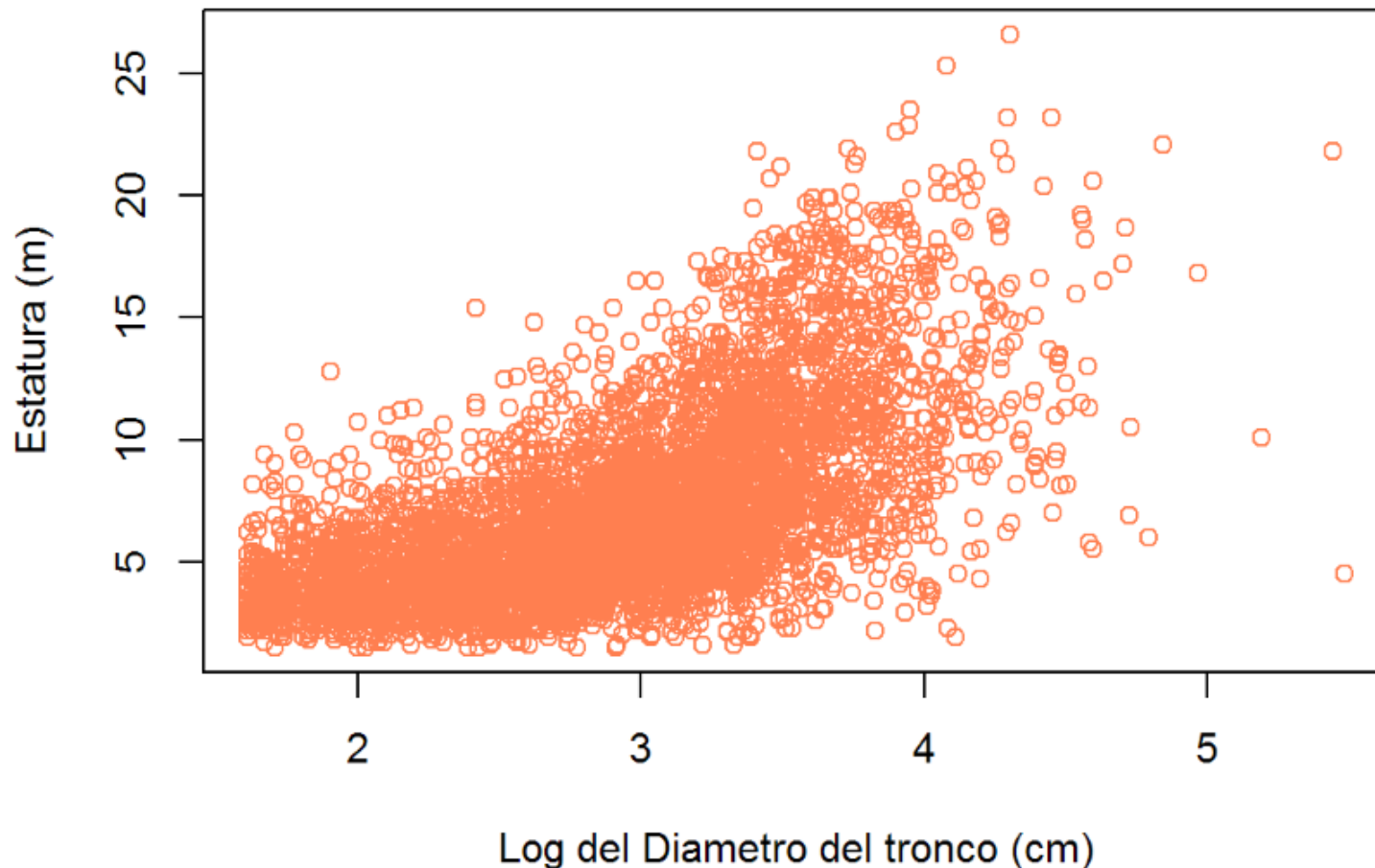
```
plot(arbolado$Alt ~ arbolado$DTr,
 col = "coral",
 main = "Estatura de árboles vs. diametro del tronco",
 ylab = "Estatura (m)", xlab = "Diametro del tronco (cm)")
```



```
plot(arbolado$Alt ~ log(arbolado$DTr),
 col = "coral",
 main = "Estatura de árboles vs. diametro del tronco",
 ylab = "Estatura (m)", xlab = "Log del Diametro del tronco (cm)")
```

← variable con transformación  
logarítmica

## Estatura de árboles vs. diametro del tronco





# plot

Vector de cuatro colores se asocia con los cuatro niveles del factor presentes en la variable categórica

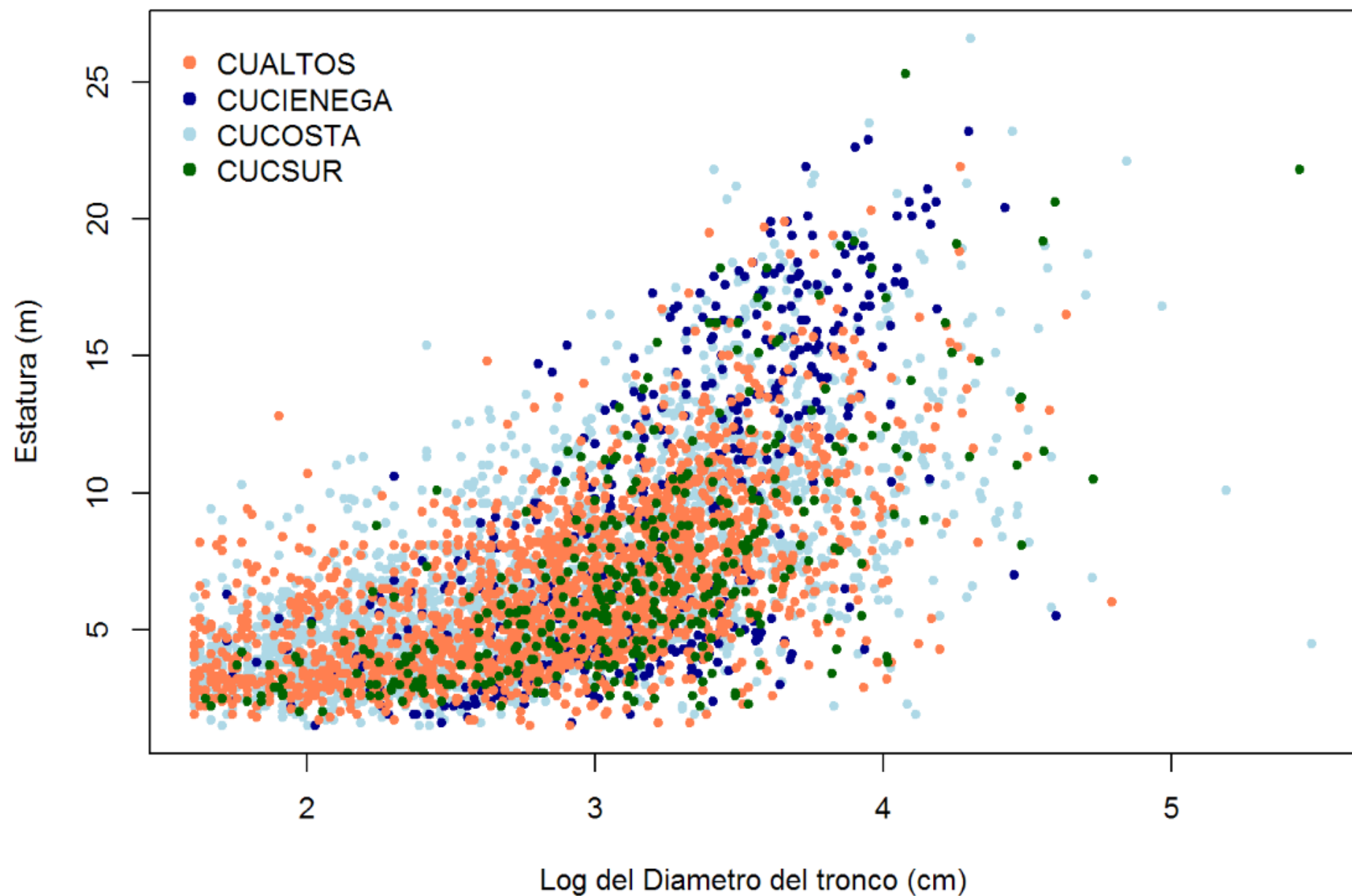
```
colors <- c("coral","darkblue","lightblue","darkgreen")
color.index <- as.numeric(arbolado$Centro)

plot(arbolado$Alt ~ log(arbolado$DTr),
 col = colors[color.index], pch = 20,
 main = "Estatura de árboles vs. diametro del tronco",
 ylab = "Estatura (m)", xlab = "Log del Diametro del tronco (cm)")

legend(1.5, 27, levels(arbolado$Centro),
 bty = "n", pch = c(19,19,19,19), col = colors)
```

Nombres de niveles de variable categórica se usan como parámetros de leyenda, y la secuencia de cuatro colores como parámetro col

## Estatura de árboles vs. diametro del tronco



# funciones

Sintaxis general de definición de las funciones

```
nombre_de_función <- function(variables_de_entrada) {
 cuerpo de función donde se usan las variables de entrada y se genera valor de salida
}
```

Existen la dos formas de definición de las funciones:

a) en una sola linea

b) con el bloque encerrado en los símbolos {}

Las funciones pueden ser anónimas o tener el nombre

```
trim <- function(x) gsub("^\\s+|\\s+$", "", x)
```

```
simpleCap <- function(x) {
 paste(toupper(substr(x, 1, 1)), substr(x, 2, nchar(x)), sep="")
}
```

# funciones

Aplicacion de las funciones para depurar nombres de las especies

```
lista_especies <- unique(as.character(arbolado$Especie))
length(lista_especies)
```

```
[1] 150
```

```
arbolado$Especie <- sapply(as.character(arbolado$Especie), simpleCap)
arbolado$Especie <- sapply(arbolado$Especie, trim)
lista_depurada_especies <- unique(arbolado$Especie)
arbolado$Especie <- as.factor(arbolado$Especie)

length(lista_depurada_especies)
```

```
[1] 144
```



# Estructuras de control

## Elementos de control de flujo de ejecución

1. `if`, `else`, `else if` - estructuras condicionales
2. `for` - ciclos (loops, bucles) con un número predeterminado de iteraciones
3. `while` - ciclos con un número indefinido de iteraciones
4. `repeat` - ciclos con un número infinito de iteraciones (se usan rara vez, principalmente en elementos de interface de usuario)
5. `next`, `return` - elementos que permiten interrumpir o saltar iteraciones en los ciclos

### Sintaxis general de las estructuras condicionales

```
if(condición lógica) {
 código para caso que la condición se cumple
} else {
 código para caso que la condición no se cumple
}
```

### Sintaxis general de los ciclos con numero predeterminado de elementos

```
for(iterador in rango_de_posibles_valores_del_iterador) {
 código del cuerpo de ciclo, donde se puede utilizar iterador como una variable
}
```

# Estructuras de control (ejemplo)

Ejemplo:

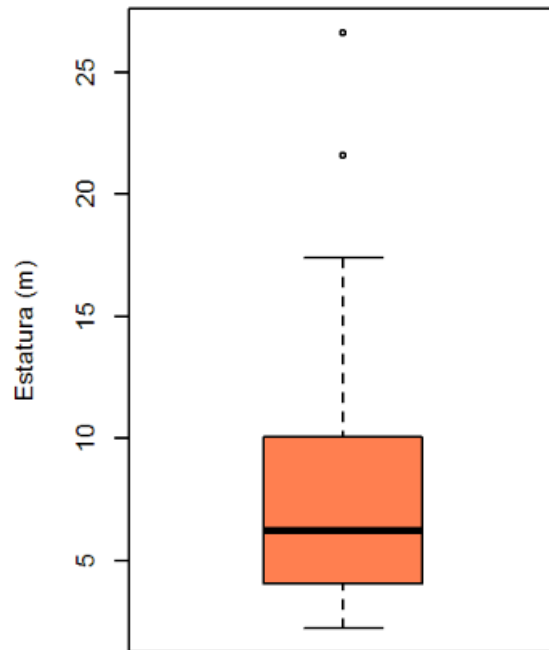
```
for(i in 1:length(lista_depurada_especies)){
 # seleccionar registros que corresponden a una especie determinada
 tabla_especie <- arbolado[arbolado$Especie == lista_depurada_especies[i],]
 # verificar que número de árboles de la especie es mayor que 50
 if(dim(tabla_especie)[1] > 50) {
 # imprimir nombre de especie y número de árboles
 cat(paste("<h4><i>", lista_depurada_especies[i], "</i></h4>"))
 cat(paste('<pre class="r"><code>', "n =", dim(tabla_especie)[1], "</code></pre>"))
 # especificar lienzo para tres graficas
 par(mfcol = c(1, 3))
 # dibujar tres graficas
 boxplot(tabla_especie$Alt,
 col = "coral", main = "Estatura de árboles", ylab = "Estatura (m)")
 boxplot(log(tabla_especie$DTr),
 col = "lightblue", main = "Diametro deo tronco", ylab = "Log del diametro de tronco (cm)")
 plot(tabla_especie$Alt ~ log(tabla_especie$DTr),
 col = "coral", main = "Estatura vs. diametro del tronco",
 ylab = "Estatura (m)", xlab = "Log del diametro del tronco (cm)")
 }
}
```

# Estructuras de control (ejemplo)

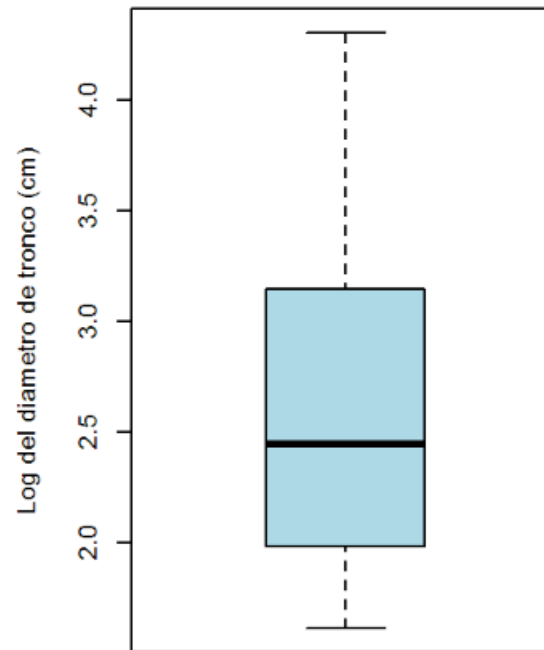
*Tabebuia rosea*

n = 159

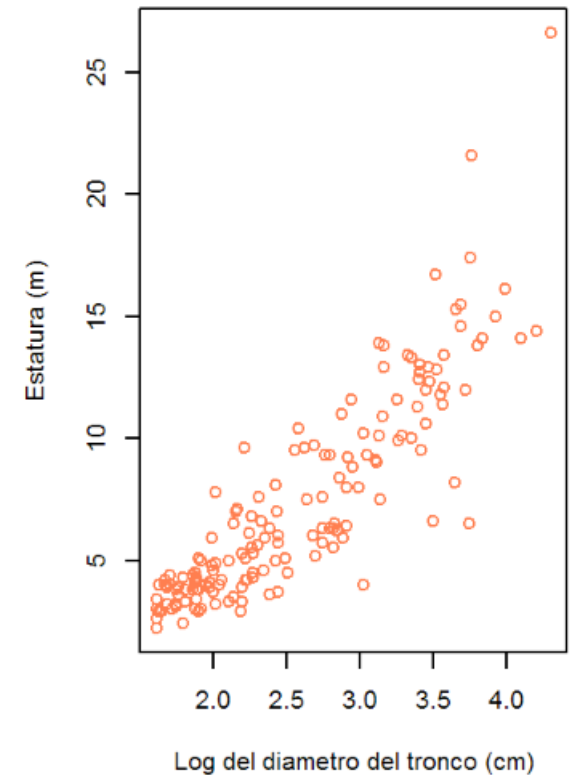
Estatura de árboles



Diametro deo tronco



Estatura vs. diametro del tronco



# Estructuras de control

Seguir con ponencia de R. Peng de John Hopkins University  
(disponible en Coursera, parte del curso “R programming”)



[https://www.youtube.com/watch?v=s\\_h9ruNwl\\_0](https://www.youtube.com/watch?v=s_h9ruNwl_0)

## Introduction to the R Language

### Control Structures

Roger Peng, Associate Professor  
Johns Hopkins Bloomberg School of Public Health



# PROFACAD: Producto 4 para el portafolio

## Producto para el portafolio: Producto 4. Ejercicio 3

Fecha de entrega del producto: 26 de julio 2019

Ejercicio 3: En equipos de 2 alumnos definir y aplicar una función que permita realizar la generación de los histogramas y diagramas de caja para los parámetros biométricos (estatura y DAP) de arbolado por especie en los datos del censo de arbolado en los Centros Universitarios de la Universidad de Guadalajara, solo para las especies con  $N \geq 50$ .

Producto para el portafolio: Reporte en el formato del documento R Markdown PDF con los fragmentos de código insertados que contiene los resultados de ejecución de la función definida en conformidad con las condiciones del ejercicio 3.

# Estructuras de datos en R

## Bibliografía complementaria

Mendoza Vega, J. B. (2018). *R para principiantes*. Libro electrónico recuperado de <https://bookdown.org/jboscomendoza/r-principiantes4/>

Peng, R. D. (2016). *R Programming for Data Science*. Leanpub. <https://leanpub.com/rprogramming>

Wickham. H. y Grolemond, G. (2017). *R for Data Science*. O'Reilly. <http://r4ds.had.co.nz/>