Tema 1. Estadística Descriptiva con R (3ª parte)

Estadística

Ángel Serrano Sánchez de León

Índice

- Matrices y data frames en R
- Frecuencias marginales
- Representaciones gráficas
- Covarianza y correlación
- Regresión lineal (ajustes)

Recordatorio de vectores

- Hasta ahora hemos trabajado en R con vectores:

```
> a <- c(3,4,7,0,1,2)
> a
[1] 3 4 7 0 1 2
```

- Accedemos a un elemento del vector utilizando el corchete, empezando en 1.

- Por defecto los vectores en R son siempre vectores fila. No existe el concepto de vector columna.

Matrices

- Para definir matrices, se utiliza la función `matrix`.
- Los datos se pasan por **columnas** (salvo que se utilice el parámetro `byrow=TRUE`).
- Hay que indicar el número de filas con `nrow`.
- En una matriz todos los elementos son del mismo tipo.

```
> b <- matrix(c(4,6,8,3,2,0,7,1), nrow=2)
  # Matriz con 2 filas
> b
     [,1] [,2] [,3] [,4]
[1,]    4    8    2    7
[2,]    6    3    0    1
```

Matrices: acceso a elementos

- El acceso a los elementos de la matriz se hace también con corchetes, indicando los índices separados por una coma.
- El primer índice es la fila, el segundo es la columna.

```
> b[1,1]
[1] 4
> b[2,3]
[1] 0
```

Matrices: acceso a elementos

- Para acceder a todos los elementos de una fila o de una columna, se deja el índice sin especificar.

```
> b
     [,1] [,2] [,3] [,4]
[1,]    4    8    2    7
[2,]    6    3    0    1
> b[,2] # Columna 2
[1] 8 3
> b[1,] # Fila 1
[1] 4 8 2 7
```

6

Matrices: dimensiones

- Dimensiones de la matriz:

```
> dim(b) # Devuelve vector con altura y
  anchura
[1] 2 4
> nrow(b) # Número de filas = altura
[1] 2
> ncol(b) # Número de columnas = anchura
[1] 4
```

Matrices: nombres de dimensiones

- Se pueden poner nombres a las dimensiones de la matriz.

```
> rownames(b)<-c("Fila1","Fila2")
> colnames(b)<-c("Col1","Col2","Col3","Col4")
> b
      Col1 Col2 Col3 Col4
Fila1    4    8    2    7
Fila2    6    3    0    1
```

- Otra manera de hacer lo mismo:

```
> dimnames(b) <- list(c("Fila1","Fila2"),
  c("Col1","Col2","Col3","Col4"))
```

- Acceso a un elemento con el nombre de las dimensiones:

```
> b["Fila2","Col4"]
[1] 1
```

8

Matrices: matriz transpuesta

- La matriz transpuesta es aquella en la que intercambiamos filas por columnas.

```
> t(b) # Las dimensiones son 4x2
     [,1] [,2]
[1,]    4    6
[2,]    8    3
[3,]    2    0
[4,]    7    1
```

9

Matrices: multiplicación

- La multiplicación de matrices según las **normas habituales del álgebra** se hace con %*%.

```
> b %*% t(b) # 2x2 = 2x4 * 4x2
        [Fila1] [Fila2]
[Fila1]     133      55
[Fila2]      55      46
> t(b) %*% b # 4x4 = 4x2 * 2x4
      [Col1] [Col2] [Col3] [Col4]
[Col1]     52     50      8     34
[Col2]     50     73     16     59
[Col3]      8     16      4     14
[Col4]     34     59     14     50
```

- La multiplicación **elemento a elemento** se hace con *.

```
> b * b
        [Col1] [Col2] [Col3] [Col4]
[Fila1]     16     64      4     49
[Fila2]     36      9      0      1
```

Matrices: sumas de elementos

- Para sumar todos los elementos de la matriz:

```
> sum(b)
[1] 31
```

- Sumar todas las filas (suma horizontal):

```
> rowSums(b)
Fila1 Fila2
   21    10
```

- Sumar todas las columnas (suma vertical):

```
> colSums(b)
Col1 Col2 Col3 Col4
  10   11    2    8
```

Data frames

- Los data frames son un tipo especial de matrices donde los datos no son homogéneos.

```
> examen <- data.frame(alumnos = c("Jorge", "Iñigo",
  "Guillermo"), notas = c(9,7.5,8))
> examen
    alumnos notas
1     Jorge   9.0
2     Iñigo   7.5
3 Guillermo   8.0
> str(examen)
'data.frame': 3 obs. of 2 variables:
 $ alumnos: Factor w/ 3 levels "Guillermo","Iñigo",..:
  3 2 1
 $ notas : num 9 7.5 8
```

`alumnos` y `notas` son los nombres de las componentes del data frame. Los valores de `alumnos` se crean como factores y los de `notas` como variables numéricas.

Data frames: acceso a elementos

- Acceso a elementos de un data frame mediante el símbolo $:

```
> examen$notas
[1] 9.0 7.5 8.0
> examen$alumnos
[1] Jorge Iñigo Guillermo
Levels: Guillermo Iñigo Jorge
> examen[2,1]
[1] Iñigo
Levels: Guillermo Iñigo Jorge
> examen[2,2]
[1] 7.5
```

Tablas de contingencia

- Supongamos que preguntamos a 10 personas si les gusta el helado del chocolate. Apuntamos sus respuestas junto a su sexo.

```
> sexo <- factor(c("H","M","M","H","H","M","M","M","H","M"))
> respuesta <- factor(c("S","S","N","S","S","S","S","S","N","S"))
> table(respuesta,sexo) # Calcula tabla de contingencia
          sexo
respuesta  H M
        N  1 1
        S  3 5
> addmargins(table(respuesta,sexo)) # Calcula frecuencias marginales
          sexo
respuesta  H M Sum
        N  1 1   2
        S  3 5   8
      Sum  4 6  10
```

Tablas de contingencia

- Sean 20 observaciones de alturas y pesos:

```
> alturas <- c(164,176,179,165,168,165,186,182,173,175,159,187,173,157,163,
  171,168,173,153,182)
> pesos <- c(64,77,82,62,71,72,85,68,72,75,81,88,72,71,74,69,81,67,65,73)
```

- Agrupemos ambas variables continuas en intervalos:

```
> alturas2 <- cut(alturas,breaks=seq(150,190,10), right=FALSE)
> pesos2 <- cut(pesos,breaks=seq(60,90,10), right=FALSE)
```

- Tabla de contingencia:

```
> t <- table(alturas2,pesos2)
> t
          pesos2
alturas2  [60,70) [70,80) [80,90)
[150,160)       1       1       1
[160,170)       2       3       1
[170,180)       2       4       1
[180,190)       1       1       2
```

Frecuencias marginales

- Cálculo de las frecuencias marginales:

```
> addmargins(t)
          pesos2
alturas2  [60,70) [70,80) [80,90) Sum
[150,160)       1       1       1   3
[160,170)       2       3       1   6
[170,180)       2       4       1   7
[180,190)       1       1       2   4
Sum             6       9       5  20
```

- Frecuencias marginales de las alturas (1ª dimensión):

```
> margin.table(t,1)
alturas2
[150,160) [160,170) [170,180) [180,190)
        3         6         7         4
```

- Frecuencias marginales de los pesos (2ª dimensión):

```
> margin.table(t,2)
pesos2
[60,70) [70,80) [80,90)
      6       9       5
```

Frecuencias marginales relativas

- Dividiendo por el total de datos conseguimos las frecuencias relativas:

```
> addmargins(t)*100/length(alturas)
          pesos2
alturas2  [60,70) [70,80) [80,90) Sum
[150,160)       5       5       5  15
[160,170)      10      15       5  30
[170,180)      10      20       5  35
[180,190)       5       5      10  20
Sum            30      45      25 100
```

Frecuencias condicionadas

- Fijémonos en el intervalo de peso [70,80). Las frecuencias absolutas condicionadas son:

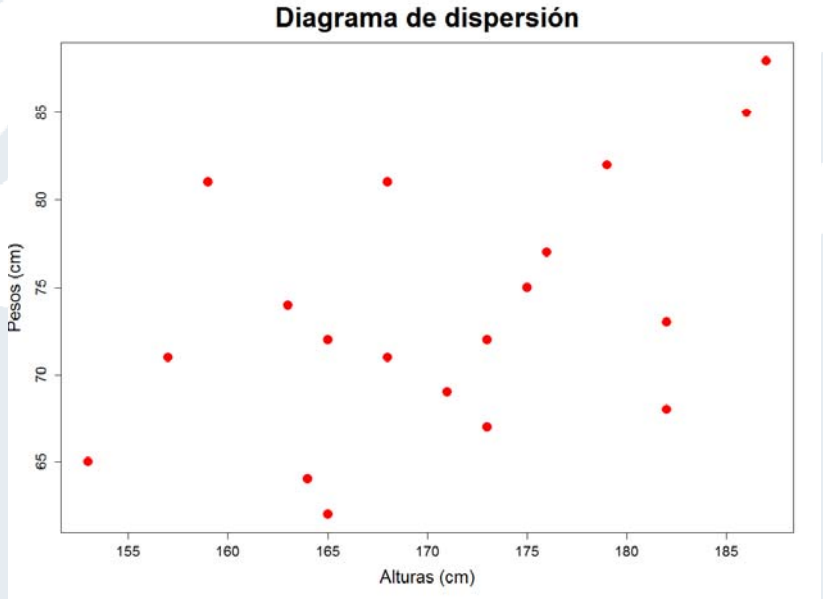
```
> t[,"[70,80)"]
[150,160) [160,170) [170,180) [180,190)
        1         3         4         1
> t[,2] # Es lo mismo pero más sencillo
[150,160) [160,170) [170,180) [180,190)
        1         3         4         1
```

- Frecuencias relativas condicionadas al intervalo de peso [70,80):

```
> t[,2]*100/sum(t[,2])
[150,160) [160,170) [170,180) [180,190)
 11.11111  33.33333  44.44444  11.11111
```

Diagrama de dispersión

```
> plot(alturas,pesos)
> plot(alturas,pesos,xlab="Alturas (cm)",ylab="Pesos
  (kg)",main="Diagrama de dispersión",col="red",
  pch=16,cex=1.5,cex.lab=1.5,cex.axis=1.2,cex.main=2)
```



Covarianza y correlación

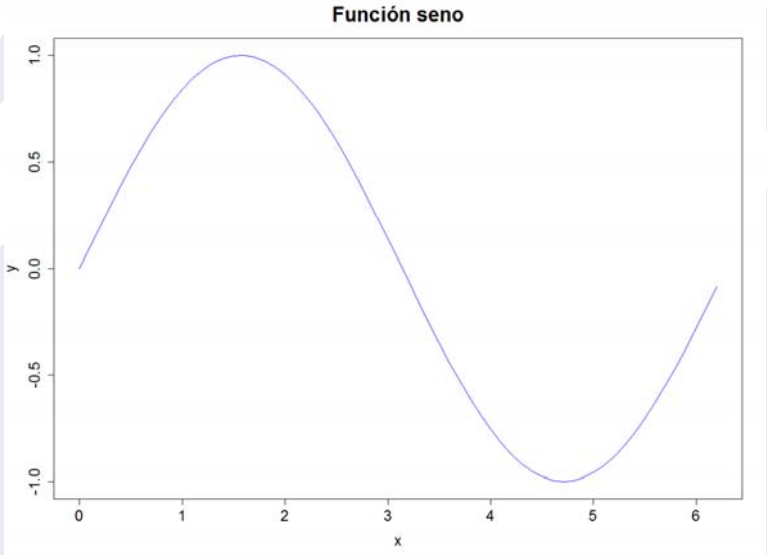
```
> var(alturas) # Varianza de las alturas
[1] 89.83947
> var(pesos) # Varianza de los pesos
[1] 49.94474
> cov(alturas,pesos) # Covarianza
[1] 33.39211

> cor(alturas,pesos) # Correlación lineal
[1] 0.4985002
```

- La correlación lineal es positiva: a mayor altura, mayor peso. Sin embargo, el coeficiente es de valor intermedio (la gráfica no es muy clara al respecto).

Diagrama de dispersión

```
> x <- seq(0,2*pi,0.1) # Secuencia de números entre 0 y 2π
  con un paso de 0.1
> y <- sin(x)
> plot(x,y,type="l",col="blue",main="Función seno",
  cex.lab=1.5,cex.axis=1.5,cex.main=2)
```

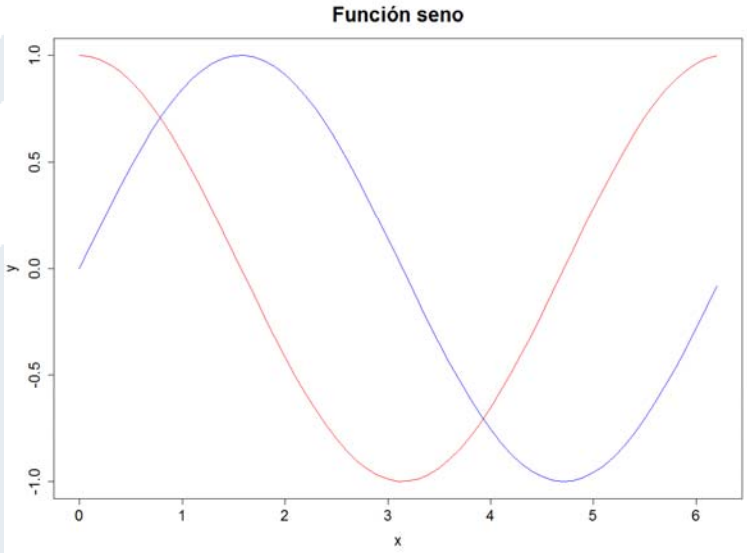


21

Diagrama de dispersión

- Para superponer una gráfica a otra creada previamente, se utiliza la función lines, con la misma notación.

```
> y2 <- cos(x)
> lines(x,y2,type="l",col="red")
```



22

Ajuste lineal (regresión)

- Ajuste de una recta de regresión a un conjunto de puntos (x,y) con la fórmula:

$$y = a + bx$$

- a = término independiente (ordenada en el origen)
- b = pendiente de la recta

$$a = \bar{y} - b\bar{x} \qquad b = \frac{s_{xy}^2}{s_x^2}$$

23

Ajuste lineal (regresión)

- Primero necesitamos dibujar el gráfico de dispersión `pesos` frente a `alturas`.

```
> plot(alturas,pesos,xlab="Alturas (cm)",ylab="Pesos
  (kg)",main="Diagrama de dispersión",col="red",
  pch=16,cex=1.5,cex.lab=1.5,cex.axis=1.2,cex.main=2)
```

- El ajuste lo realizamos con la función `lm` (=modelo lineal), siguiente la notación siguiente: `lm(variable eje Y ~ Variable eje X)`

```
> ajuste <- lm (pesos ~ alturas)
```

- Superponemos al gráfico la recta de regresión con `abline`.

```
> abline(ajuste,col="blue")
```

- Los coeficientes del ajuste son (*a*=término independiente y *b*=pendiente):

```
> ajuste$coefficients
(Intercept)   alturas
  9.9102200 0.3716863
```

24

Ajuste lineal (regresión)

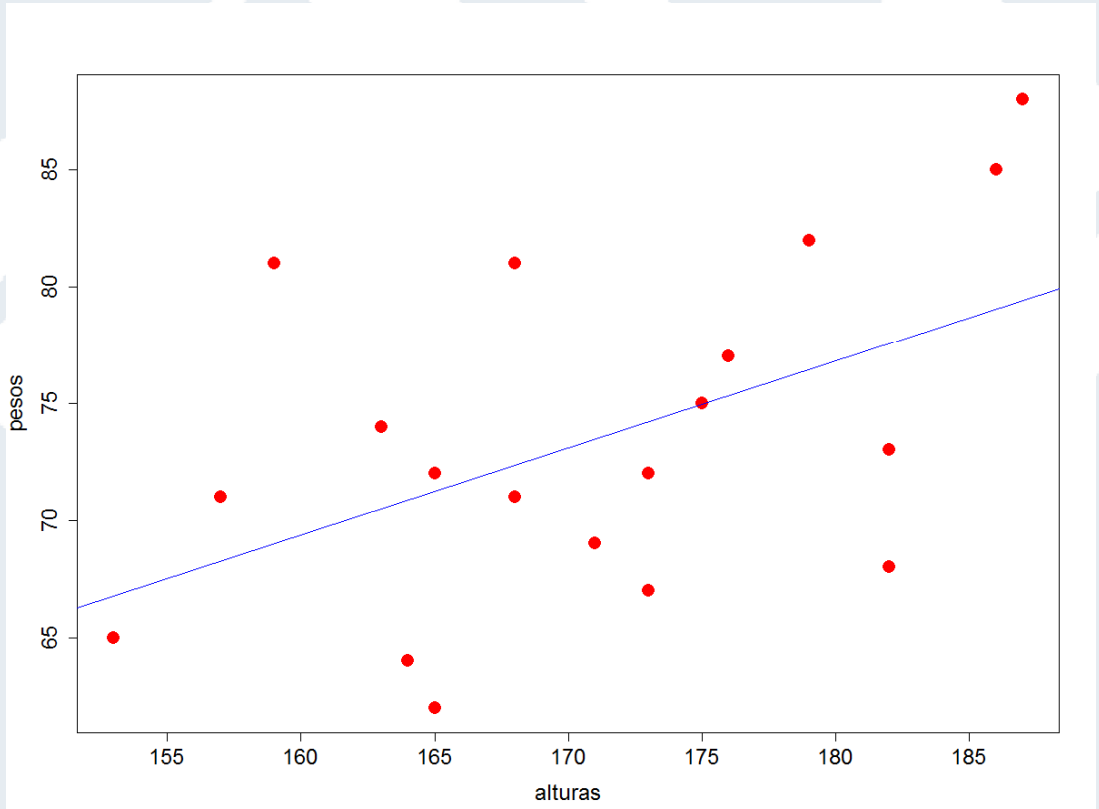
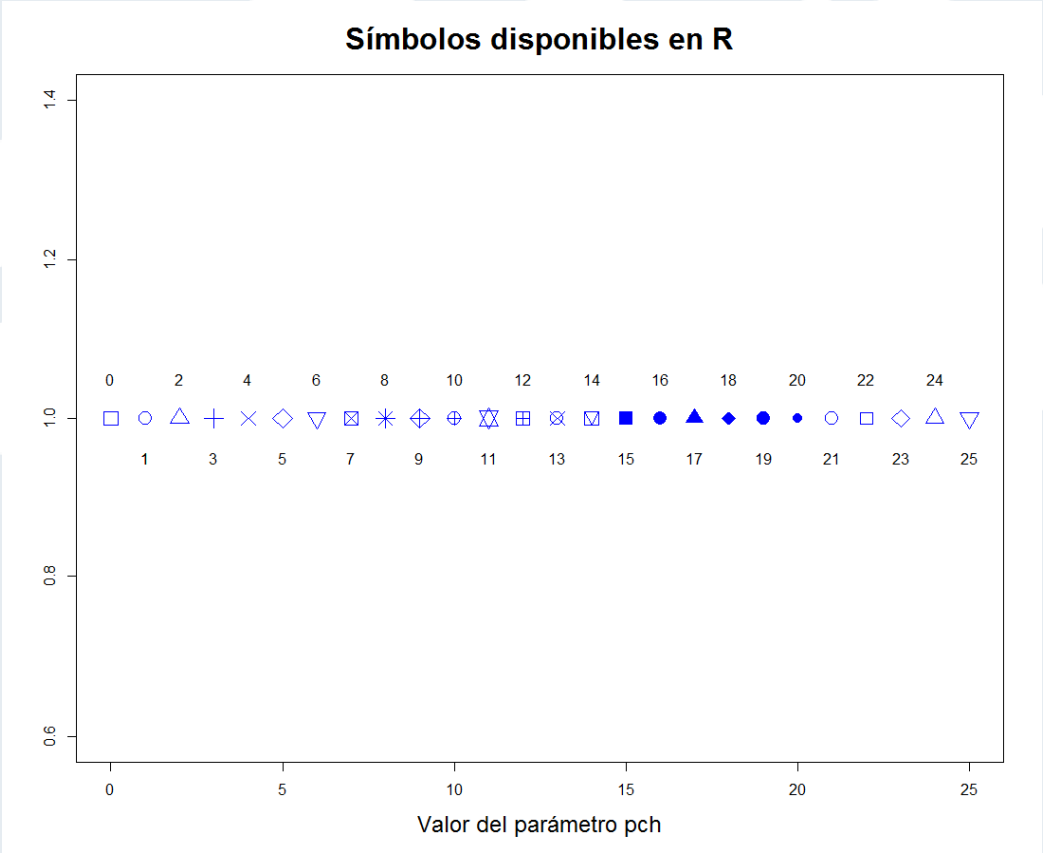


Diagrama de dispersión

- Hay 26 símbolos para los puntos de un diagrama de dispersión (parámetro pch de 0 a 25).

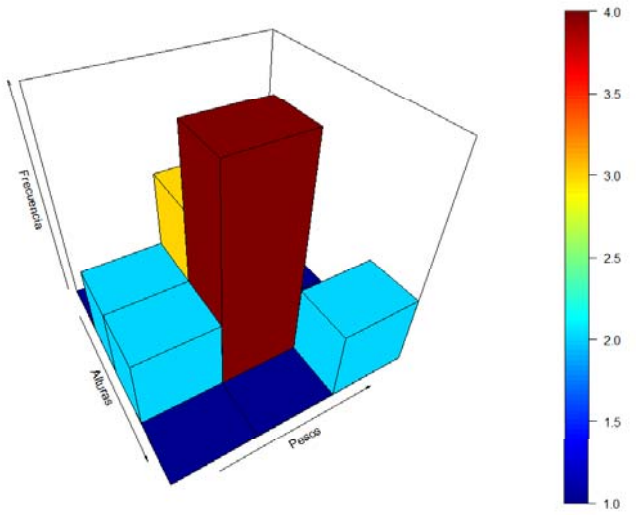
```
> plot(0:25,rep(1,26),pch=0:25,col="blue",cex=2,
  xlab="Valor del parámetro pch",ylab="", main="Símbolos
  disponibles en R",cex.main=2, cex.lab=1.5)
> text(seq(0,24,2),rep(1.05,13),seq(0,24,2)) # Numera de 0
  a 24, por encima del símbolo
> text(seq(1,25,2),rep(0.95,14),seq(1,25,2)) # Numera de 1
  a 25, por debajo del símbolo
```

Diagrama de dispersión



Histograma 3D

```
> require(plot3D) # Paquete plot3D
> hist3D(z=table(alturas2,pesos2),border="black",
  theta=60,xlab="Alturas",ylab="Pesos", zlab="Frecuencia")
```



Dataset de color de pelo y ojos

```
> HairEyeColor # Dataset color de pelo y ojos
, , Sex = Male
    Eye
Hair   Brown Blue Hazel Green
Black     32   11    10     3
Brown     53   50    25    15
Red       10   10     7     7
Blond      3   30     5     8
, , Sex = Female
    Eye
Hair   Brown Blue Hazel Green
Black     36    9     5     2
Brown     66   34    29    14
Red       16    7     7     7
Blond      4   64     5     8
```

Dataset de color de pelo y ojos

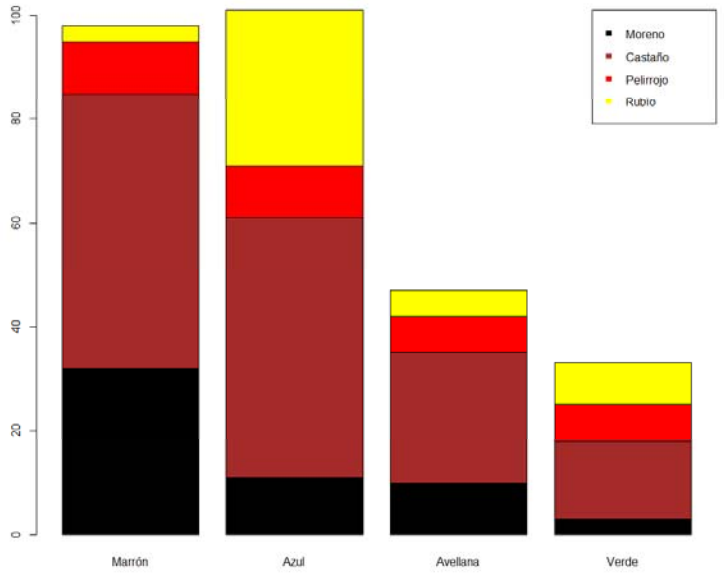
```
> addmargins(HairEyeColor)
, , Sex = Male
       Eye
Hair   Brown Blue Hazel Green Sum
Black     32   11    10     3  56
Brown     53   50    25    15 143
Red       10   10     7     7  34
Blond      3   30     5     8  46
Sum       98  101    47    33 279

, , Sex = Female
       Eye
Hair   Brown Blue Hazel Green Sum
Black     36    9     5     2  52
Brown     66   34    29    14 143
Red       16    7     7     7  37
Blond      4   64     5     8  81
Sum      122  114    46    31 313

, , Sex = Sum
       Eye
Hair   Brown Blue Hazel Green Sum
Black     68   20    15     5 108
Brown    119   84    54    29 286
Red       26   17    14    14  71
Blond      7   94    10    16 127
Sum      220  215    93    64 592
```

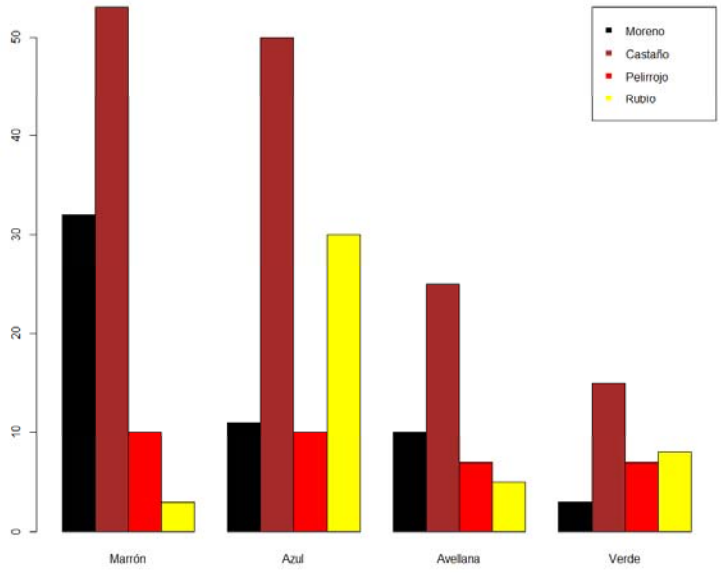
Dataset de color de pelo y ojos

```
> barplot(HairEyeColor[,,1],names=c("Marrón","Azul","Avellana",
  "Verde"),col=c("black","brown","red","yellow"))
> legend("topright",legend=c("Moreno","Castaño","Pelirrojo",
  "Rubio"),col=c("black","brown","red","yellow"),pch=15)
```



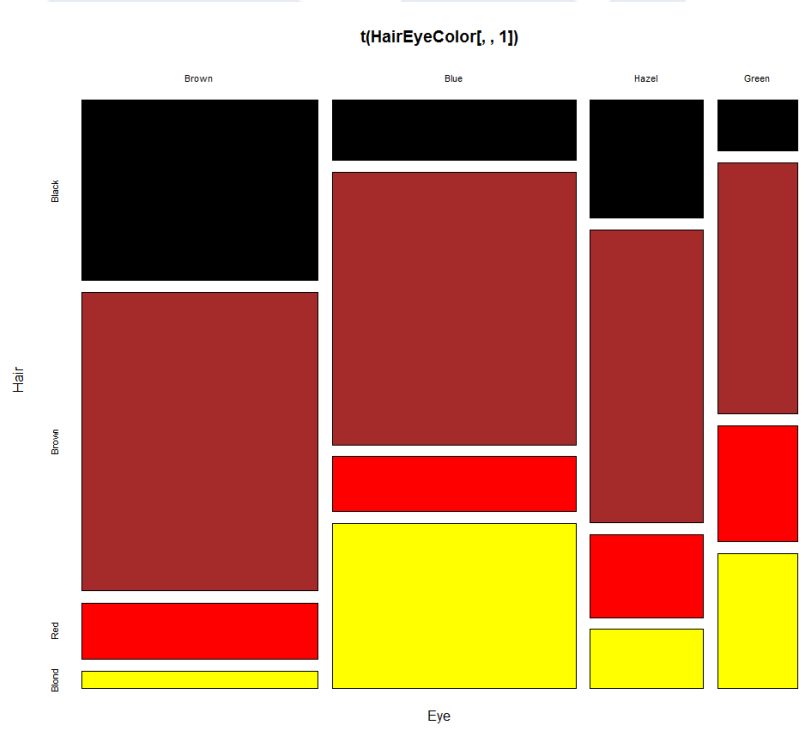
Dataset de color de pelo y ojos

```
> barplot(HairEyeColor[,,1],names=c("Marrón","Azul","Avellana",
  "Verde"),col=c("black","brown","red","yellow"),beside=TRUE)
> legend("topright",legend=c("Moreno","Castaño","Pelirrojo",
  "Rubio"),col=c("black","brown","red","yellow"),pch=15)
```



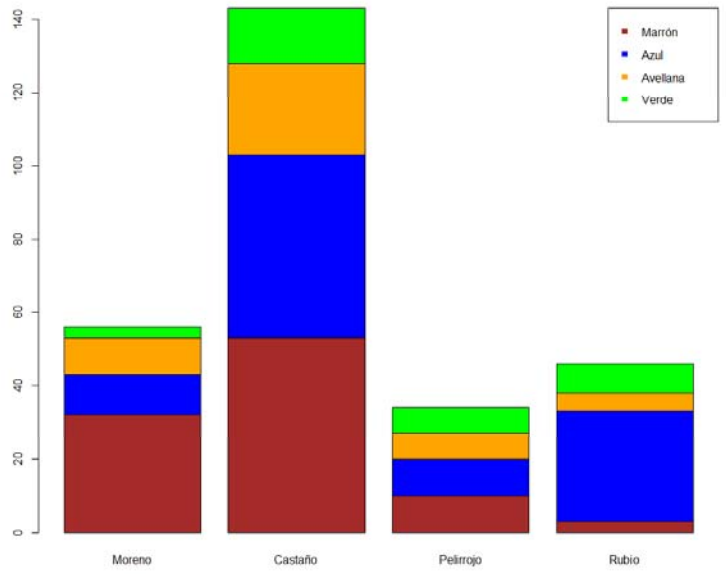
Dataset de color de pelo y ojos

```
> mosaicplot(t(HairEyeColor[,,1]),col=c("black", "brown","red", "yellow"))
```



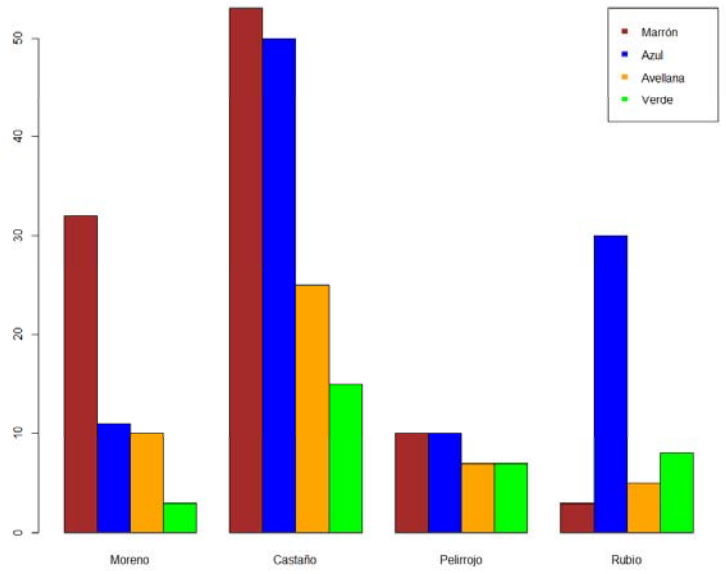
Dataset de color de pelo y ojos

```
> barplot(t(HairEyeColor[,,1]),names=c("Moreno","Castaño",
   "Pelirrojo","Rubio"),col=c("brown","blue","orange","green"))
> legend("topright",legend=c("Marrón","Azul","Avellana","Verde"),
   col=c("brown","blue","orange","green"),pch=15)
```



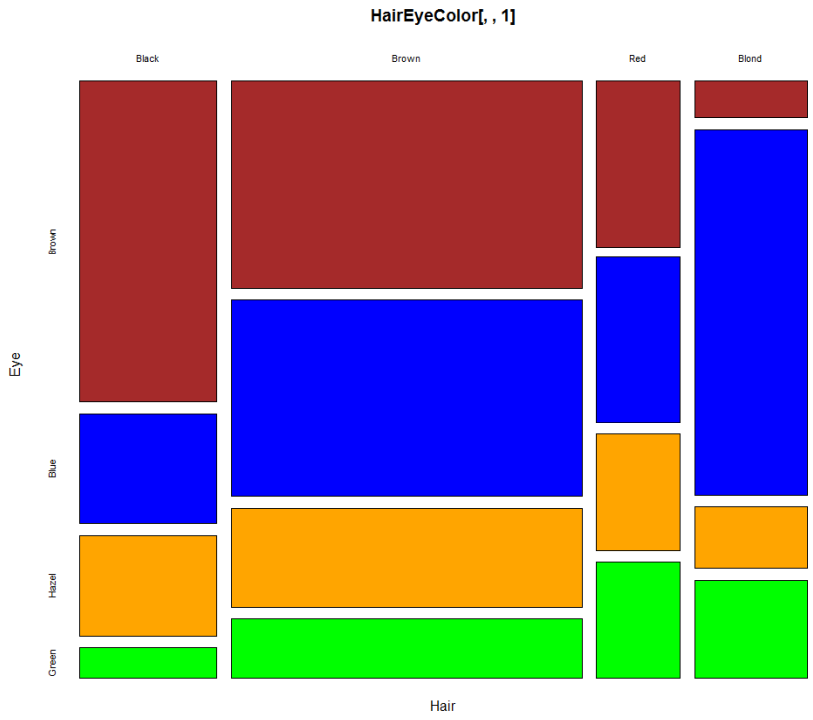
Dataset de color de pelo y ojos

```
> barplot(t(HairEyeColor[,,1]),names=c("Moreno","Castaño",
   "Pelirrojo","Rubio"),col=c("brown","blue","orange","green"), beside=TRUE)
> legend("topright",legend=c("Marrón","Azul","Avellana","Verde"),
   col=c("brown","blue","orange","green"),pch=15)
```



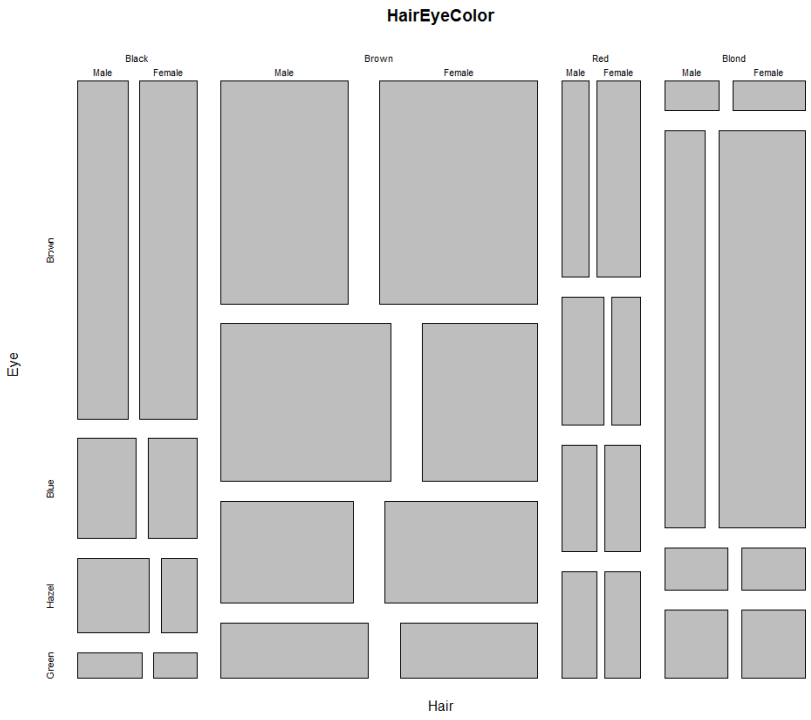
Dataset de color de pelo y ojos

```
> mosaicplot(HairEyeColor[,,1],col=c("brown","blue","orange", "green"))
```



Dataset de color de pelo y ojos

```
> mosaicplot(HairEyeColor)
```



Datasets en R

- R trae por defecto muchos datasets o conjuntos de datos con los que trabajar.
- Se encuentran en el paquete datasets.
- Para ver qué datasets tenemos disponibles:

```
> data()
Data sets in package ‘datasets’:
AirPassengers Monthly Airline Passenger Numbers 1949-1960
BJsales Sales Data with Leading Indicator BJsales.lead (BJsales) Sales Data
   with Leading Indicator
BOD Biochemical Oxygen Demand
CO2 Carbon Dioxide Uptake in Grass Plants
ChickWeight Weight versus age of chicks on different diets
DNase Elisa assay of DNase
[...]
```

Dataset iris

- El dataset Iris de Anderson está disponible en R como el data frame "iris". Veamos su estructura:

```
> str(iris)
'data.frame': 150 obs. of 5 variables:
 $ Sepal.Length: num 5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9
  ...
 $ Sepal.Width : num 3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1
  ...
 $ Petal.Length: num 1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4
  1.5 ...
 $ Petal.Width : num 0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2
  0.1 ...
 $ Species : Factor w/ 3 levels "setosa","versicolor",..:
  1 1 1 1 1 1 1 1 1 1 ...
```

Diagrama de dispersión

```
> plot(iris$Sepal.Length,iris$Sepal.Width,
  col="violet",pch=17)
```

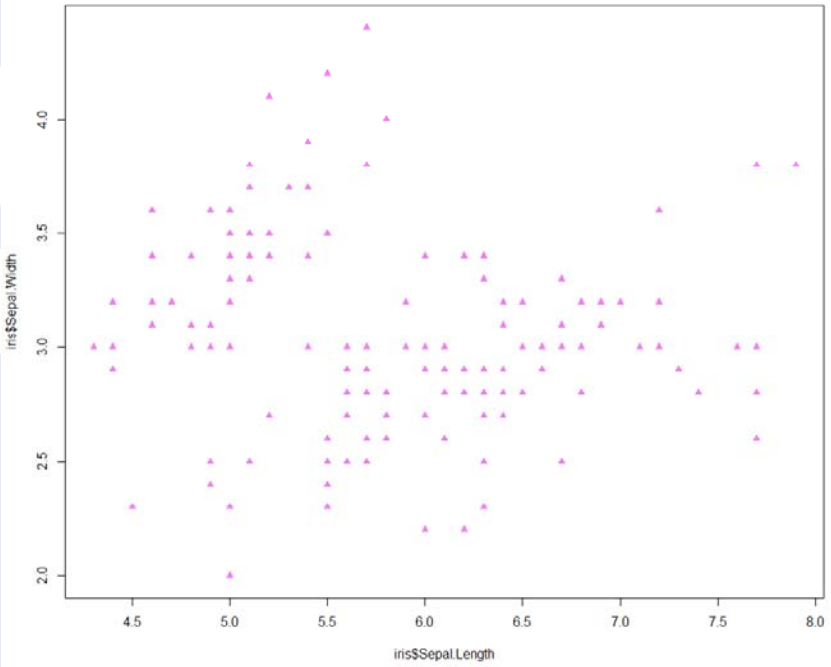


Diagrama de dispersión

```
> plot(iris$Sepal.Length,iris$Sepal.Width,
  col=iris$Species,pch=17)
> legend("topright",legend=unique(iris$Species),
  col=1:3,pch=17)
```

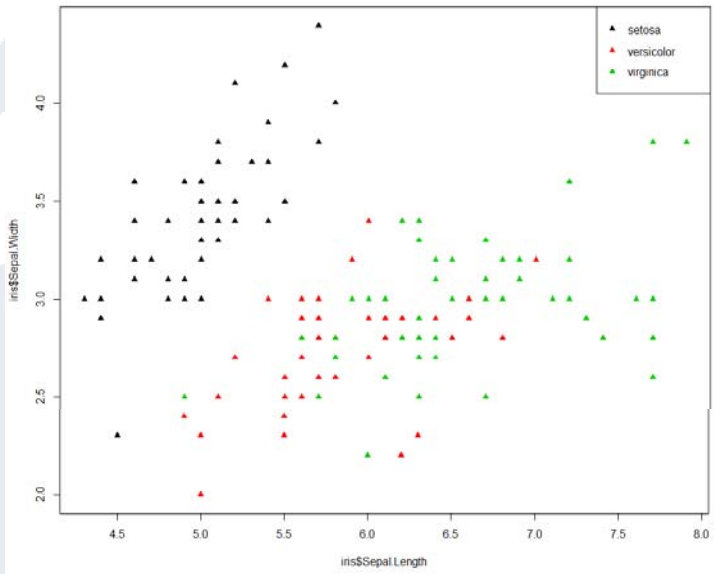
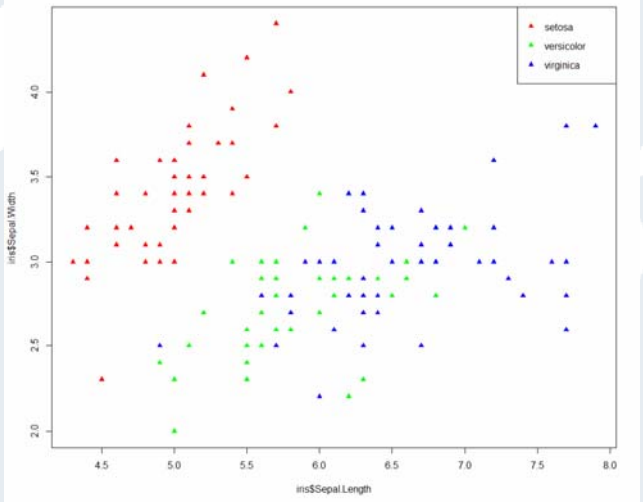


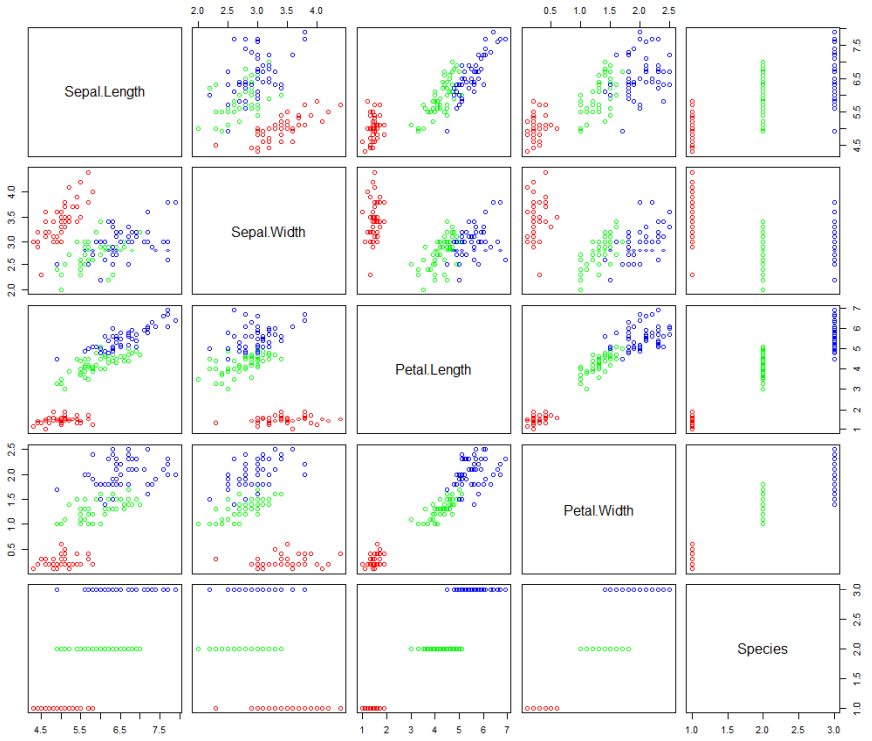
Diagrama de dispersión

```
> palette(c("red","green","blue"))  # Color 1 = rojo,
  color 2 = verde, color 3 = azul
> plot(iris$Sepal.Length,iris$Sepal.Width,
  col=iris$Species,pch=17)
> legend("topright",legend=unique(iris$Species),
  col=1:3,pch=17)
```



Matriz de diagramas de dispersión

```
> plot(iris,col=iris$Species)
```



Matriz de diagramas de dispersión

```
> palette("default") # Deshace la paleta de colores
  anterior al valor por defecto
> plot(iris,col=iris$Species)
```

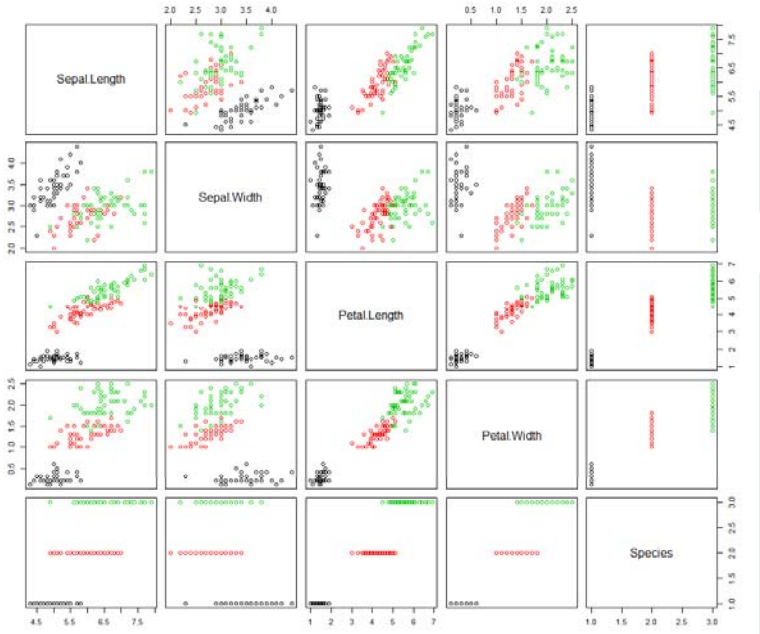
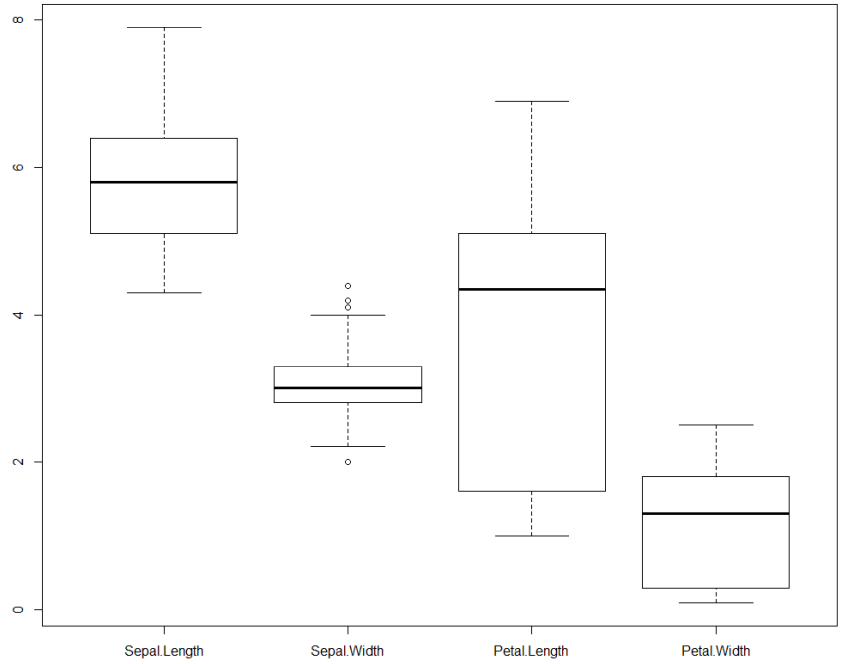


Diagrama de caja y bigotes

```
> boxplot(iris[1:4])
```



Matrices de covarianza y correlación

```
> cov(iris[1:4]) # Matriz de covarianza
             Sepal.Length Sepal.Width Petal.Length Petal.Width
Sepal.Length    0.6856935  -0.0424340    1.2743154   0.5162707
Sepal.Width    -0.0424340   0.1899794   -0.3296564  -0.1216394
Petal.Length    1.2743154  -0.3296564    3.1162779   1.2956094
Petal.Width     0.5162707  -0.1216394    1.2956094   0.5810063

> cor(iris[1:4]) # Matriz de correlación
             Sepal.Length Sepal.Width Petal.Length Petal.Width
Sepal.Length    1.0000000  -0.1175698    0.8717538   0.8179411
Sepal.Width    -0.1175698   1.0000000   -0.4284401  -0.3661259
Petal.Length    0.8717538  -0.4284401    1.0000000   0.9628654
Petal.Width     0.8179411  -0.3661259    0.9628654   1.0000000
```

Coordenadas paralelas

- En un diagrama de coordenadas paralelas para una variable multidimensional, representamos cada dimensión como una escala vertical paralela a las demás.
- A cada observación de la variable le corresponde una línea formada por segmentos conectados según los valores que toma cada componente.

```
> require("MASS") # Paquete MASS
> palette(c("red","green","blue")) # Recuperamos la paleta de
  colores rojo, verde, azul
> parcoord(iris[1:4],col=iris$Species)
> legend("top",legend=unique(iris$Species),col=1:3, lty=1)
```

47

Coordenadas paralelas

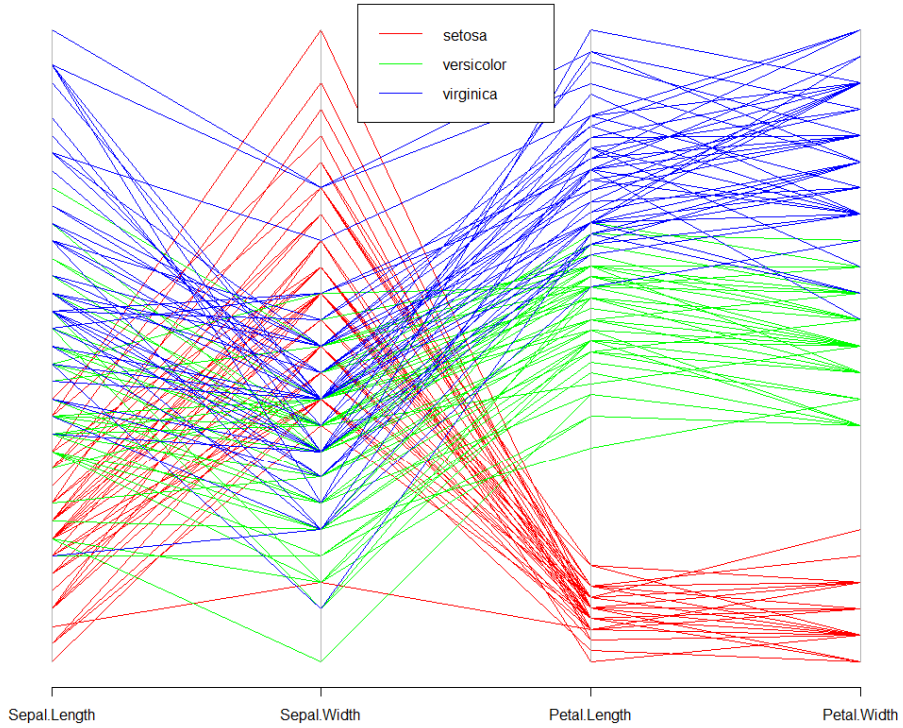


Diagrama radial (o de estrella)

- En un diagrama radial, las dimensiones se muestran como radios que parten de un centro y representan todas las observaciones a la vez.

```
> stars(iris[1:4],locations=(c(0,0)),
col.lines=iris$Species,key.loc=c(0,0))
```

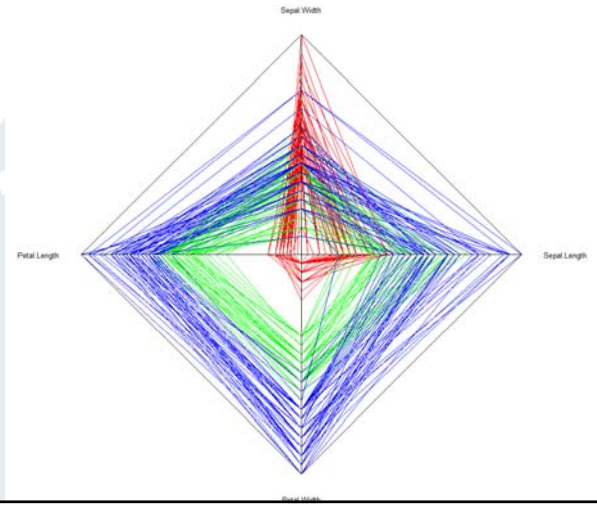


Diagrama radial (o de estrella)

- También podemos representar cada observación con una estrella independiente.

```
> stars(iris[1:4],col.lines=iris$Species)
```