

Gráficos con R

Welton Vieira dos Santos

16/2/2020

Representación gráfica con R

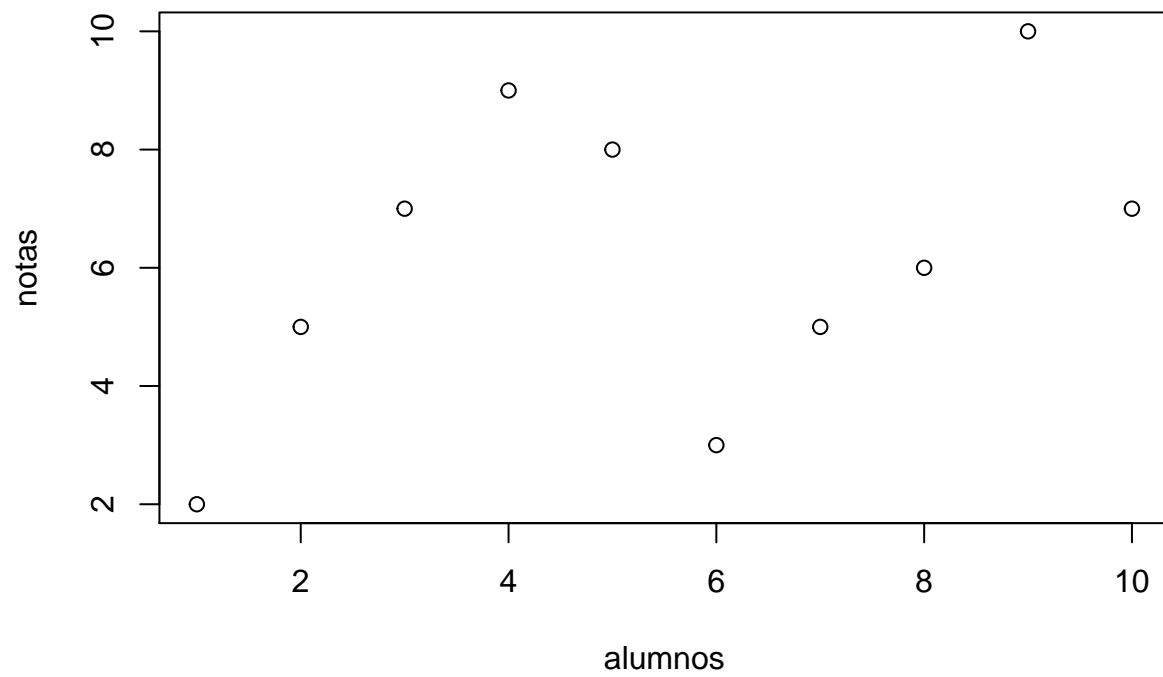
Gráficos con la función `plot`

Es una función que tiene la capacidad de plotar gráficas por puntos.

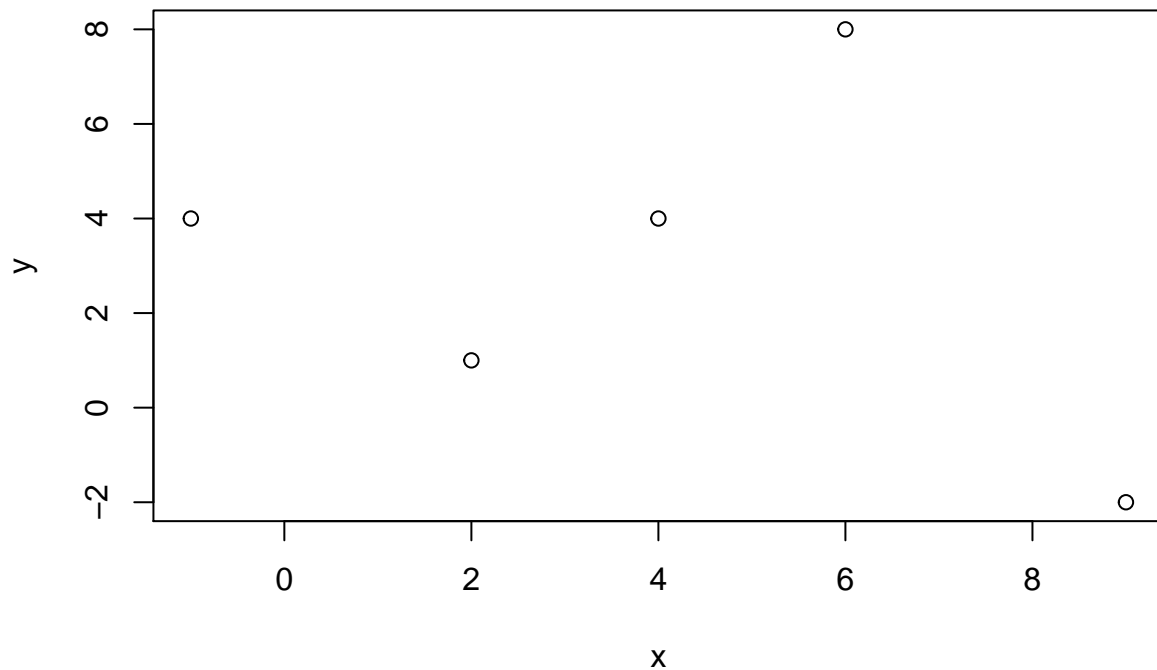
- `plot(x,y)`: para dibujar un gráfico básico de puntos siendo x,y vectores numéricos.
 - `plot(x)=plot(1:length(x),x)` donde `1:length(x)` representa el eje x y, el x representa aquí el eje y .
- `plot(x, función)`: para dibujar el gráfico de una función.

Ejemplo básico:

```
alumnos <- c(1:10)
notas <- c(2,5,7,9,8,3,5,6,10,7)
plot(alumnos, notas)
```



```
x <- c(2,6,4,9,-1)
y <- c(1,8,4,-2,4)
plot(x, y)
```



En las chunks de R puedes poner como opciones para administrar la visualización de los gráficos.

```
""{r primer_plot, fig.cap="Grafico basico explicando el uso del plot"}
```

```
x <- c(2,6,4,9,-1)
```

```
y <- c(1,8,4,-2,4)
```

```
plot(x, y)
```

```
""
```

```
x <- c(2,6,4,9,-1)
```

```
y <- c(1,8,4,-2,4)
```

```
plot(x, y)
```

Para centrar el gráfico en la hoja de impresión.

```
""{r fig.cap="Grafico basico explicando el uso del plot", fig.align='center'}
```

```
x <- c(2,6,4,9,-1)
```

```
y <- c(1,8,4,-2,4)
```

```
plot(x, y)
```

```
""
```

```
x <- c(2,6,4,9,-1)
```

```
y <- c(1,8,4,-2,4)
```

```
plot(x, y)
```

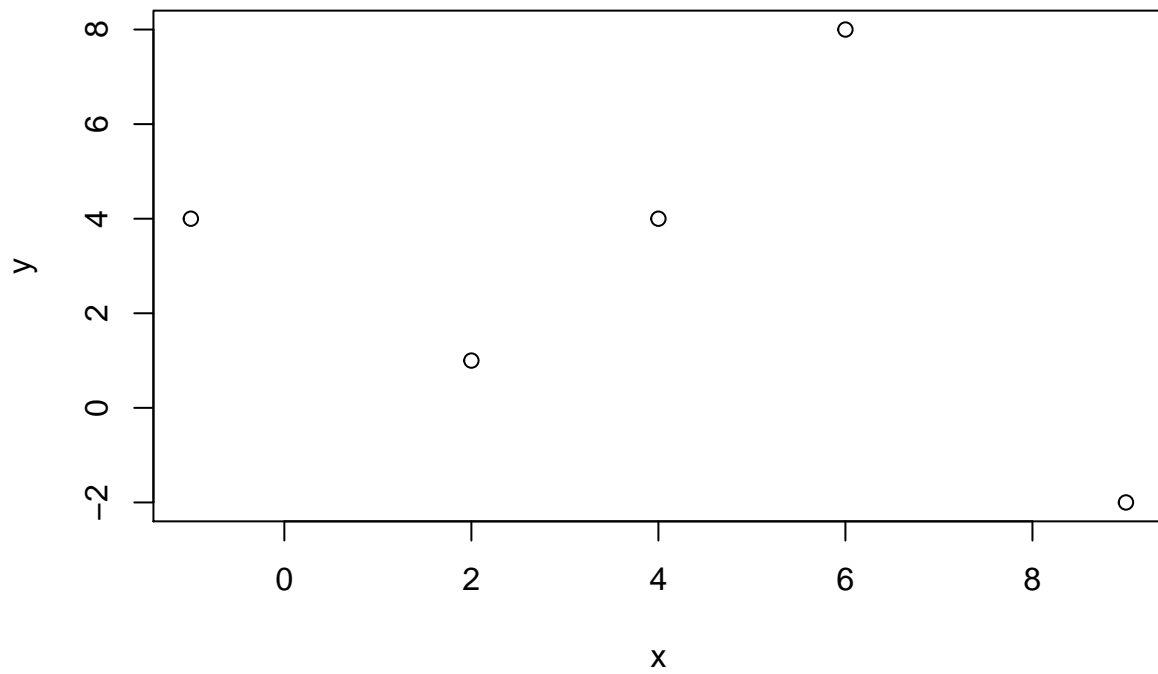


Figure 1: Gráfico básico explicando el uso del plot

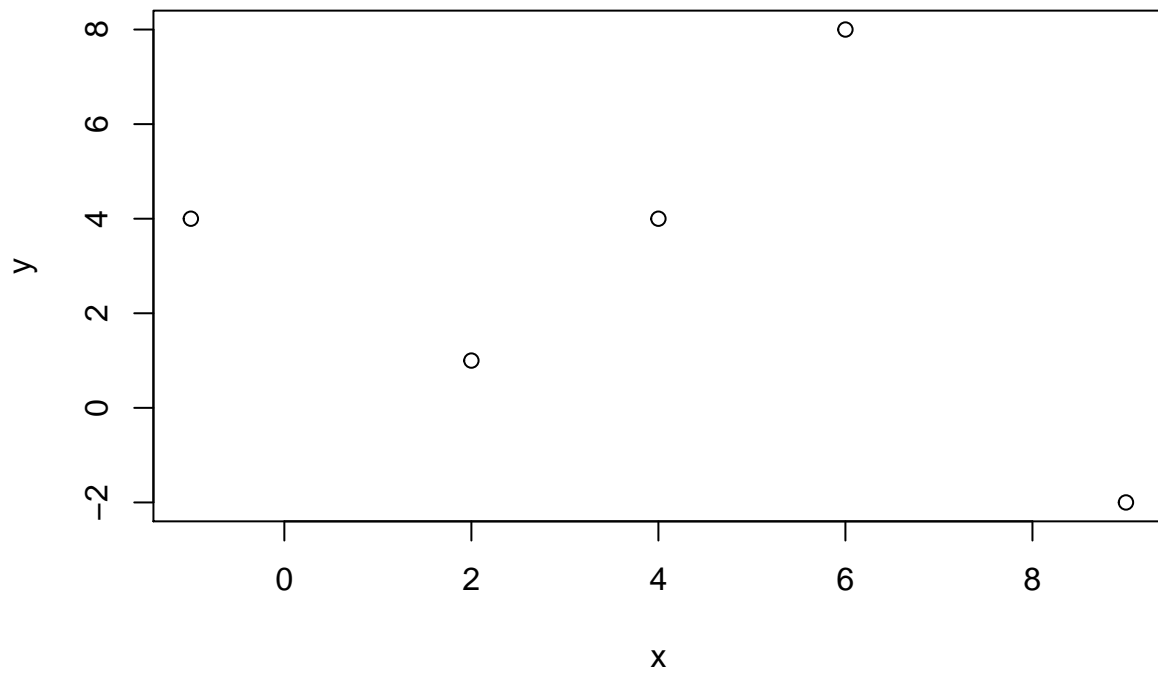
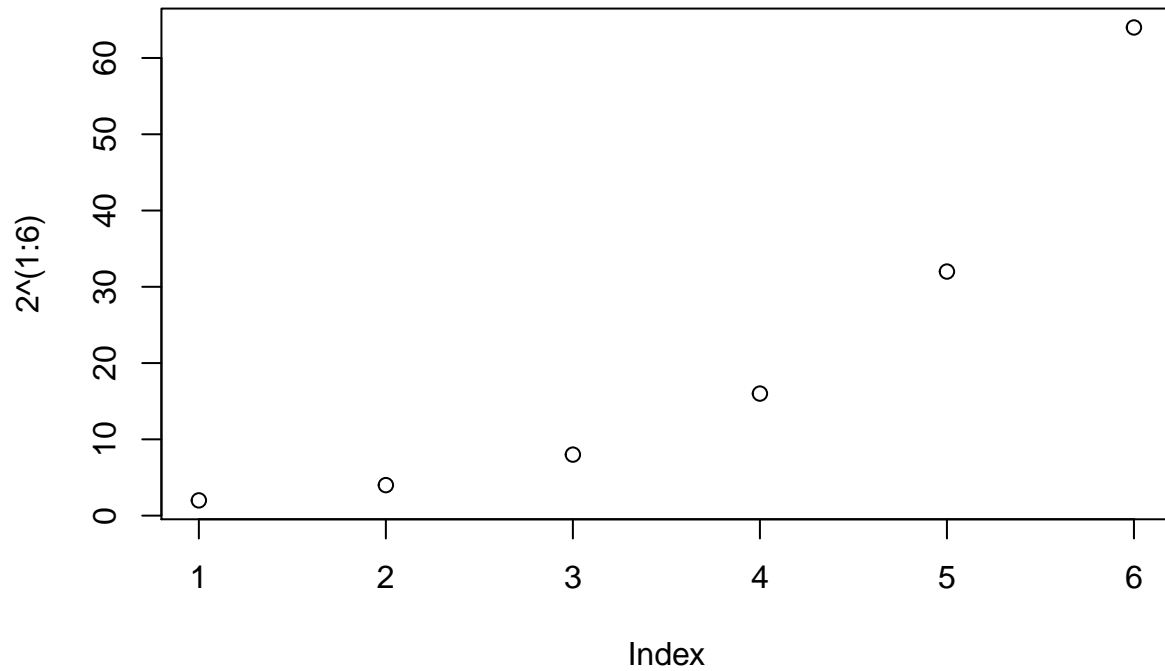


Figure 2: Gráfico básico explicando el uso del plot

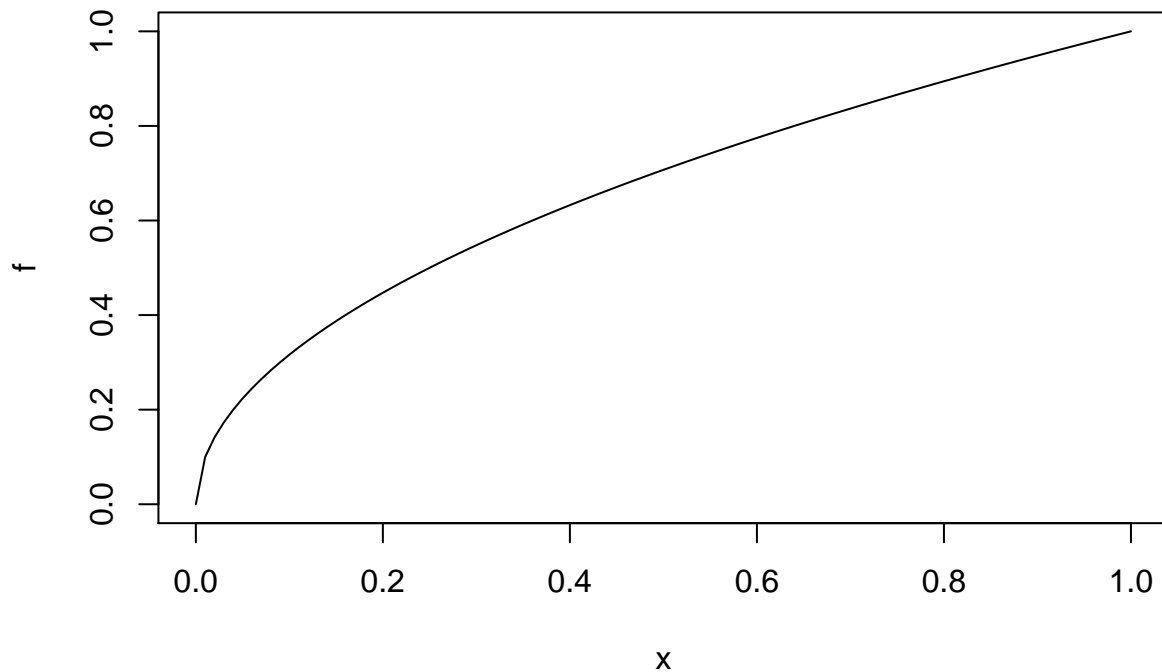
Si no incorporamos vector `y`, R nos va a tomar el parámetro `x` como si fuese el vector de datos `y`: `plot(1:n, x)`

```
plot(2^(1:6))
```



Si queremos representar una función $f(x)$:

```
f <- function(x){  
  sqrt(x)  
}  
plot(f)
```



Parámetros de función `plot()`

- **log**: para indicar que queremos el gráfico en escala logarítmica.
- **main("título")**: para poner título al gráfico. Si en vez de un texto queráis poner una expresión matemática, tenéis que utilizar la función **expression()**
- **xlab("etiqueta")**: para poner etiqueta al eje X
- **ylab("etiqueta")**: para poner etiqueta al eje Y
- **pch = n**: para elegir el símbolo de los puntos(pointer character). $n = 0, 1, \dots, 25$. El valor por defecto es **pch = 1**
- **cex**: para elegir el tamaño de los símbolos de los pointer character.
- **col = "color en inglés"**: para elegir el color de los símbolos. Gama de colores

*** Ejemplos:

```
# Ejemplo de plotar la sucesión de fibonacci
```

```
n = 1:20
```

```
fib = (1/sqrt(5))*((1+sqrt(5))/2)^n - (1/sqrt(5))*((1-sqrt(5))/2)^n
```

```
fib
```

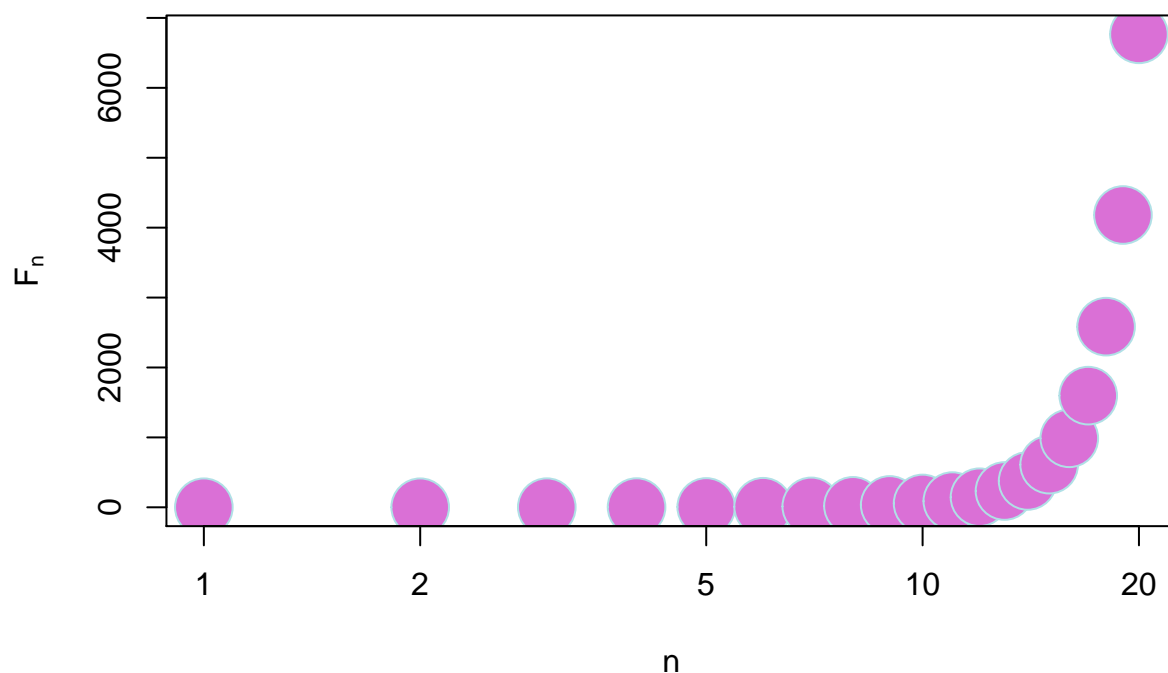
```
## [1] 1 1 2 3 5 8 13 21 34 55 89 144 233 377 610
```

```
## [16] 987 1597 2584 4181 6765
```

```
#Uso de etiquetas en el eje x y el eje y con una expresión matemática.
```

```
plot(fib, xlab = "n", ylab = expression(F[n]), main = "Sucesión de Fibonacci", pch = 21, cex = 4, col =
```

Sucesión de Fibonacci



Varios gráficos en la misma ventana

Se utiliza la función `par(mfrow = c(1,2))` para decir que quiere varios gráficos en la misma pantalla. En ese caso sería dos gráficos en la misma fila.

```
# Ejemplo de plotar la sucesión de fibonacci
```

```
n = 1:20
```

```
fib = (1/sqrt(5))*((1+sqrt(5))/2)^n - (1/sqrt(5))*((1-sqrt(5))/2)^n
```

```
fib
```

```
[1] 1 1 2 3 5 8 13 21 34 55 89 144 233 377 610
```

```
[16] 987 1597 2584 4181 6765
```

```
#Poner dos gráficos en la misma pantalla.
```

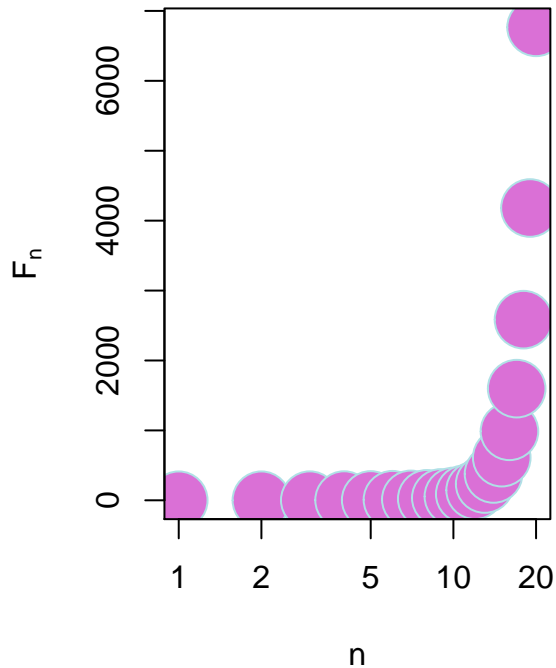
```
par(mfrow = c(1,2))
```

```
#Uso de etiquetas en el eje x y el eje y con una expresión matemática.
```

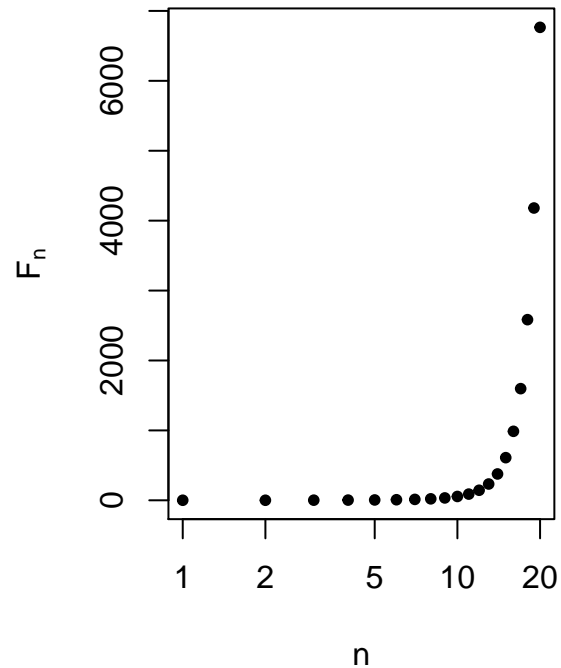
```
plot(fib, xlab = "n", ylab = expression(F[n]), main = "Sucesión de Fibonacci", pch = 21,
     cex = 4, col = "powderblue", bg = "orchid", log = "x")
```

```
plot(fib, xlab = "n", ylab = expression(F[n]), main = "Sucesión de Fibonacci", pch = 20,
     log = "x")
```


Sucesión de Fibonacci



Sucesión de Fibonacci



```
# Para resetear el gráfico para posterior uso.
par(mfrow = c(1,1))
```

- **type:** para elegir el tipo de gráfico que queremos:
 - **p:** puntos(valor por defecto)
 - **l:** líneas rectas que unen los puntos(dichos puntos no tienen símbolos)
 - **b:** líneas rectas que unen los puntos(dichos puntos tienen símbolos). Las líneas no traspasan los puntos.
 - **o:** como el caso anterior pero en este caso las líneas sí que traspasan los puntos.
 - **h:** histogramas de líneas
 - **s:** histogramas de escalones
 - **n:** para no dibujar los puntos

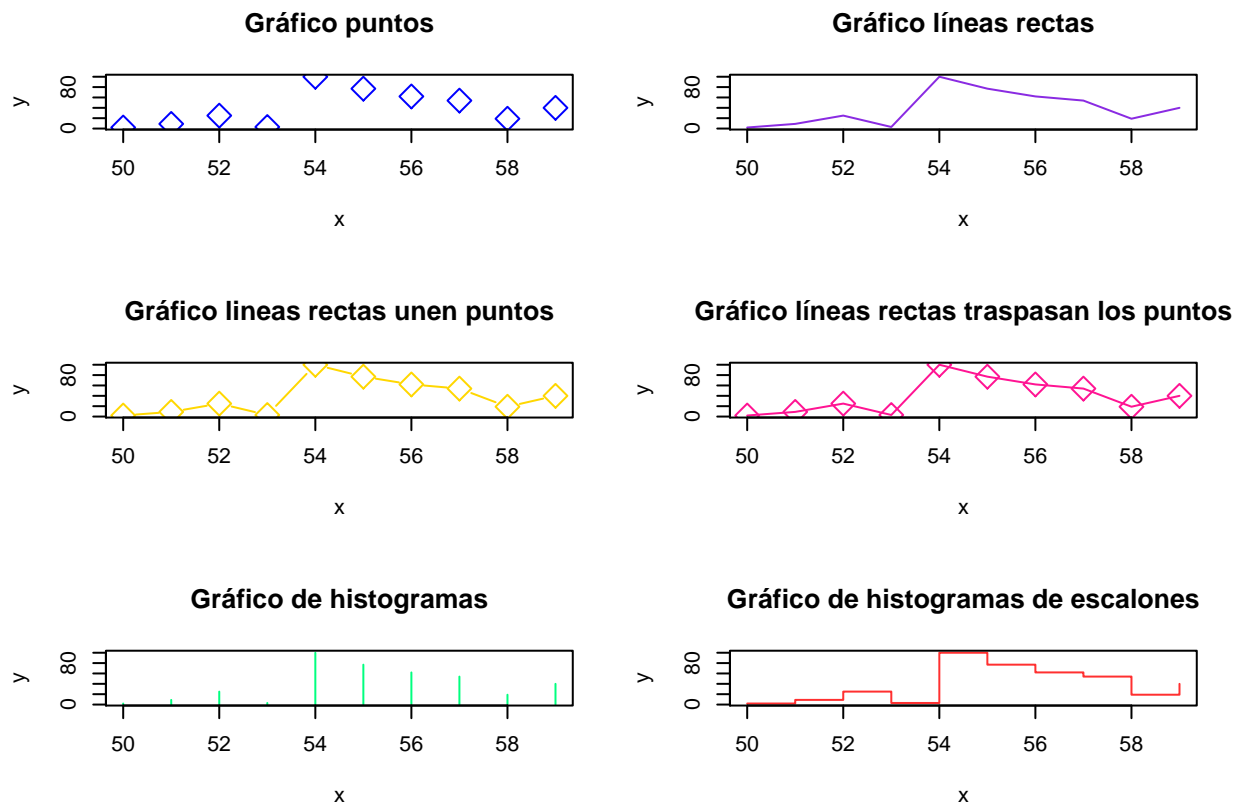
Ejemplos:

```
par(mfrow = c(3,2))
x <- c(50:59)
y <- c(2,9,25,3,100,77,62,54,19,40)
plot(x,y, pch = 23, cex = 2, col = "blue", type = "p",
     main = "Gráfico puntos")
plot(x,y, pch = 23, cex = 2, col = "blueviolet", type = "l",
     main = "Gráfico líneas rectas")
plot(x,y, pch = 23, cex = 2, col = "gold", type = "b",
```

```

main = "Gráfico líneas rectas unen puntos")
plot(x,y, pch = 23, cex = 2, col = "deeppink", type = "o",
     main = "Gráfico líneas rectas traspasan los puntos")
plot(x,y, pch = 23, cex = 2, col = "springgreen", type = "h",
     main = "Gráfico de histogramas")
plot(x,y, pch = 23, cex = 2, col = "firebrick1", type = "s",
     main = "Gráfico de histogramas de escalones")

```



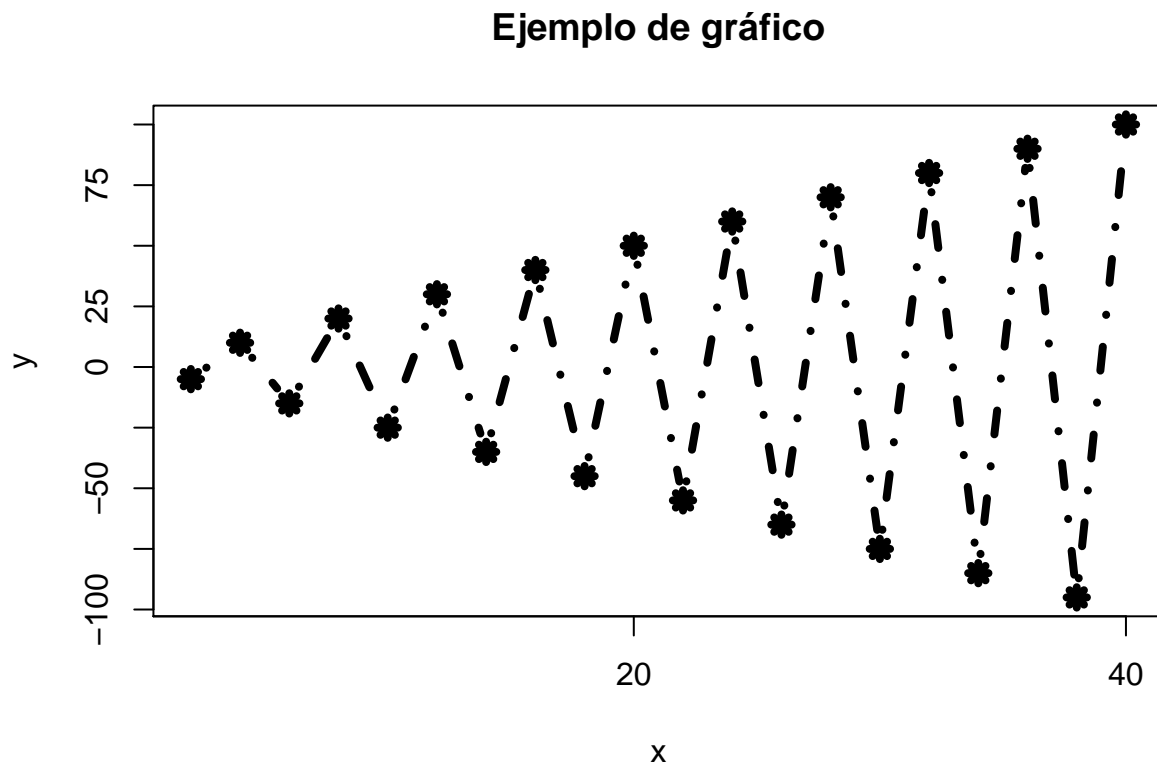
```

par(mfrow = c(1,1))

```

- **lty:** para especificar el tipo de línea
 - "solid" o 1: línea continua(valor por defecto).
 - "dashed" o 2: línea discontinua.
 - "dotted" o 3: línea de puntos.
 - "dotdashed" o 4: línea que alterna puntos y rayas.
- **lwd:** para especificar el grosor de las líneas
- **xlim:** para modificar el rango del eje X
- **ylim:** para modificar el rango del eje Y
- **xaxp:** para modificar posiciones de las marcas en el eje X
- **yaxp:** para modificar posiciones de las marcas en el eje Y

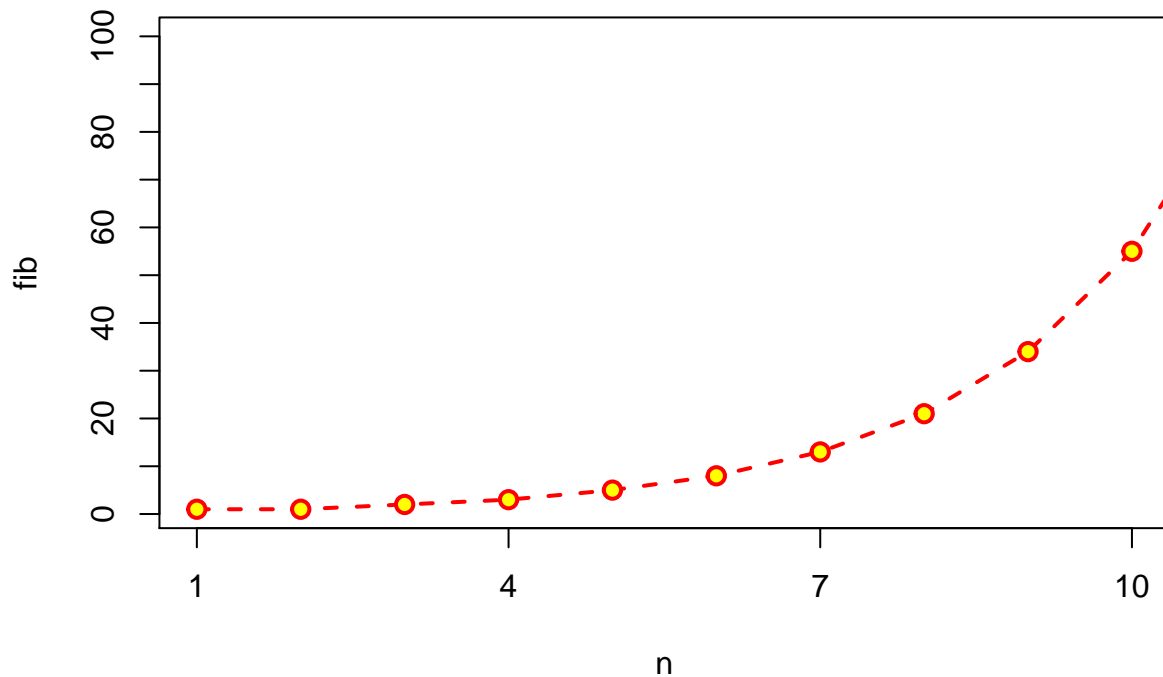
```
x <- (2*(1:20))
y <- (-1)^(1:20)*5*(1:20)
plot(x,y,main="Ejemplo de gráfico",pch=8,cex=1,type="b",lty=4,lwd=4,xaxp=c(0,40,2),
     yaxp=c(-100,100,8))
```



más ejemplos

```
plot(n, fib, pch=21, col="red", bg="yellow", cex=1.2, type="o", lty="dashed",
     lwd=2, xlim=c(1,10), ylim = c(1,100),
     yaxp=c(1,10,3),yaxp=c(0,100,10), main = "Fibonacci")
```

Fibonacci



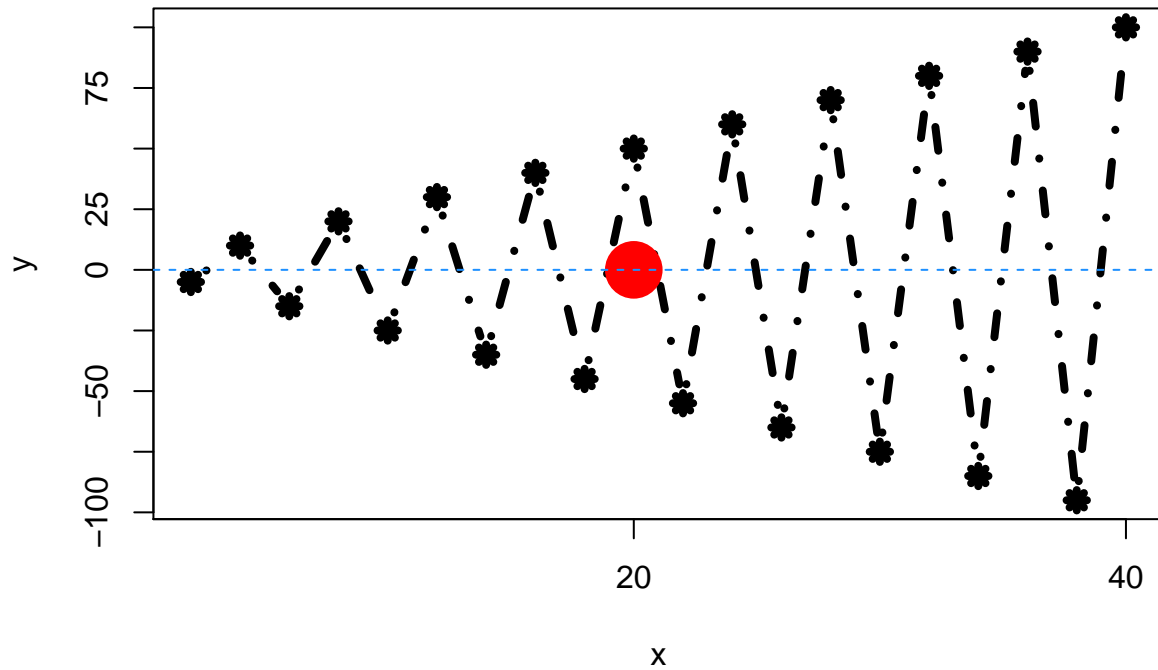
Añadir elementos al gráfico

- **points(x,y)**: añade un punto de coordenadas (x,y) a un gráfico ya existente.
- **abline**: para añadir una recta a un gráfico ya existente.
 - **abline(a,b)**: añade una recta $y = bx + a$.
 - **abline(v = x0)**: añade la recta vertical $x = x_0$. v puede estar asignado a un vector.
 - **abline(h = y0)**: añade la recta horizontal $y = y_0$. h puede estar asignado a un vector.

Ejemplo:

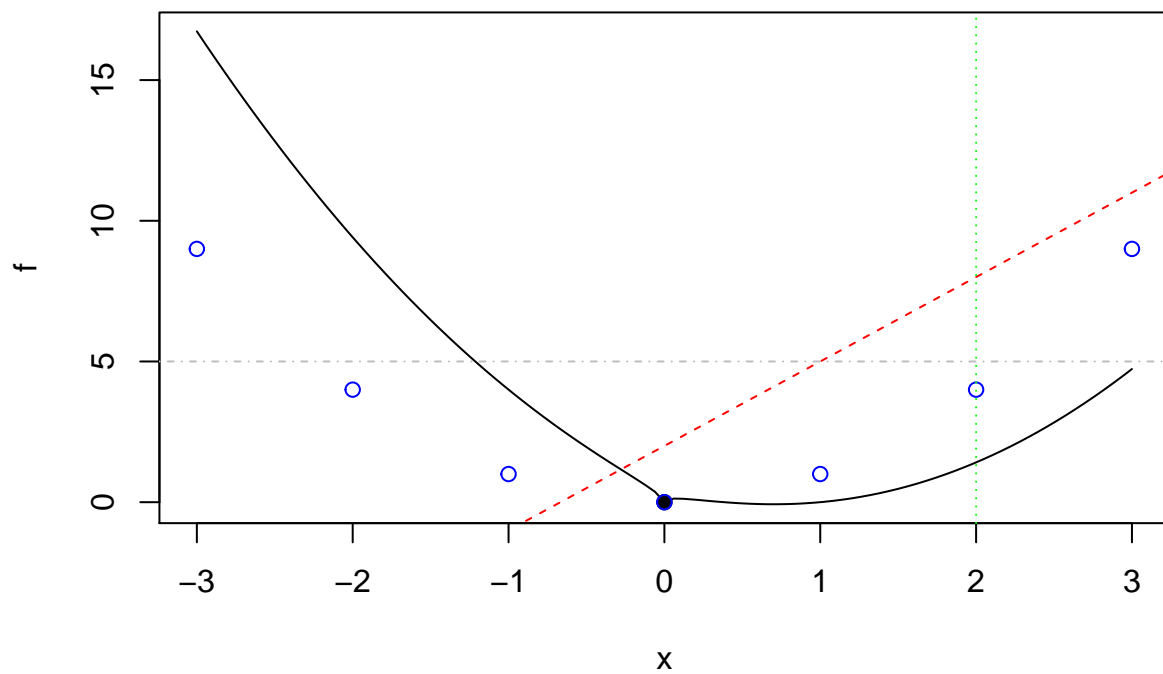
```
x <- (2*(1:20))
y <- (-1)^(1:20)*5*(1:20)
plot(x,y,main="Ejemplo de gráfico",pch=8,cex=1,type="b",lty=4,lwd=4,xaxp=c(0,40,2),
     yaxp=c(-100,100,8))
# Agregando un punto rojo en la coordenada x=20 e y=0
points(20, 0, col="red", cex=4, pch=16)
# Agregando una recta horizontal en x=0.
abline(h=0, lty=2, col="dodgerblue")
```

Ejemplo de gráfico

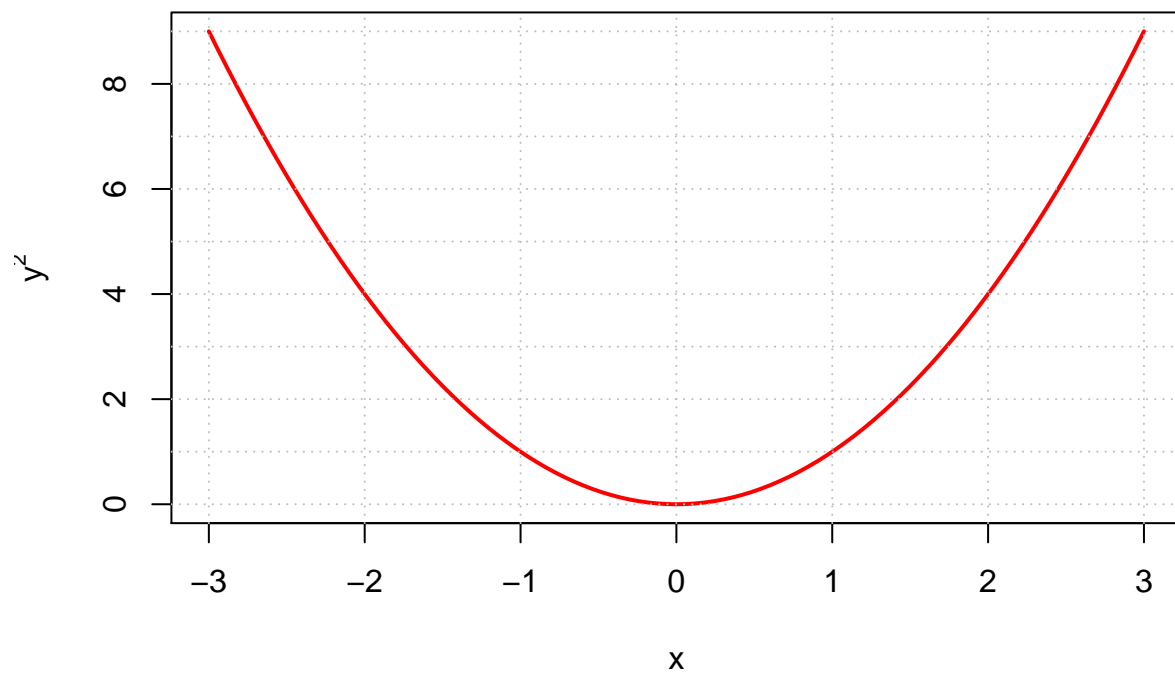


```
f <- function(x){  
  x^2-2*x + sqrt(abs(x))  
}
```

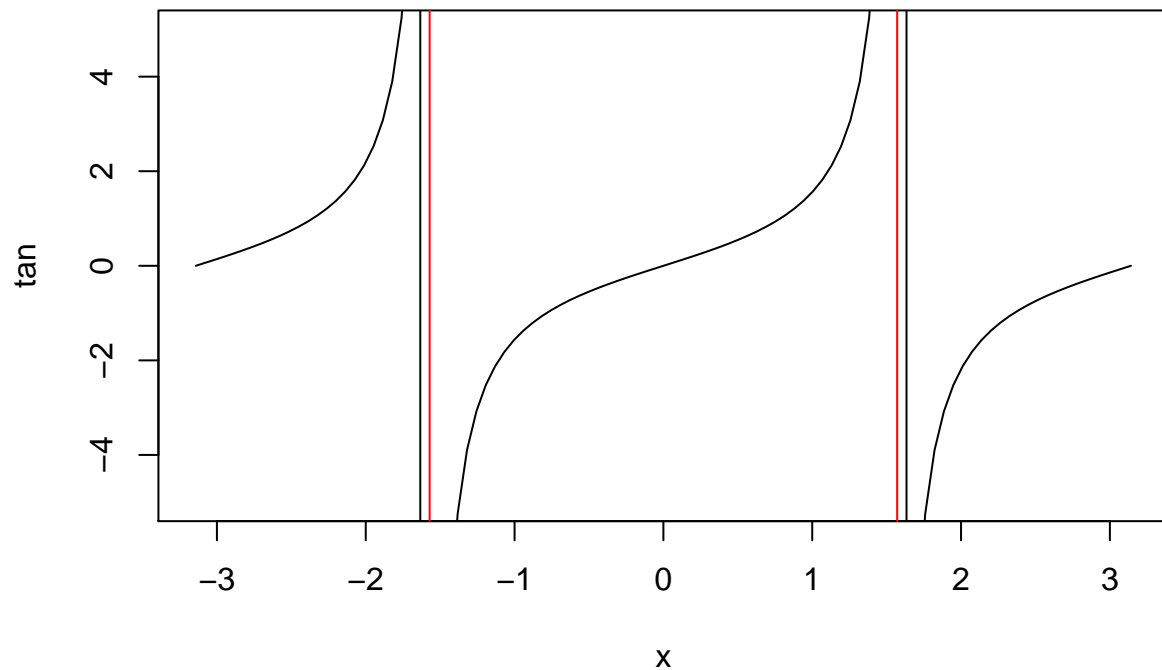
```
plot(f, xlim = c(-3,3))  
points(0,0, pch = 19)  
points(-3:3, (-3:3)^2, col = "blue")  
abline(2,3, lty = "dashed", col = "red")  
abline(v = 2, lty = "dotted", col = "green")  
abline(h = 5, lty = "dotdash", col = "gray")
```



```
f <- function(x){  
  x^2  
}  
  
plot(f, xlim = c(-3,3), col = "red", lwd = 2, ylab = expression(y^2), xlab = "x")  
abline(h = 0:9, v = -3:3, lty = "dotted", col = "grey")
```



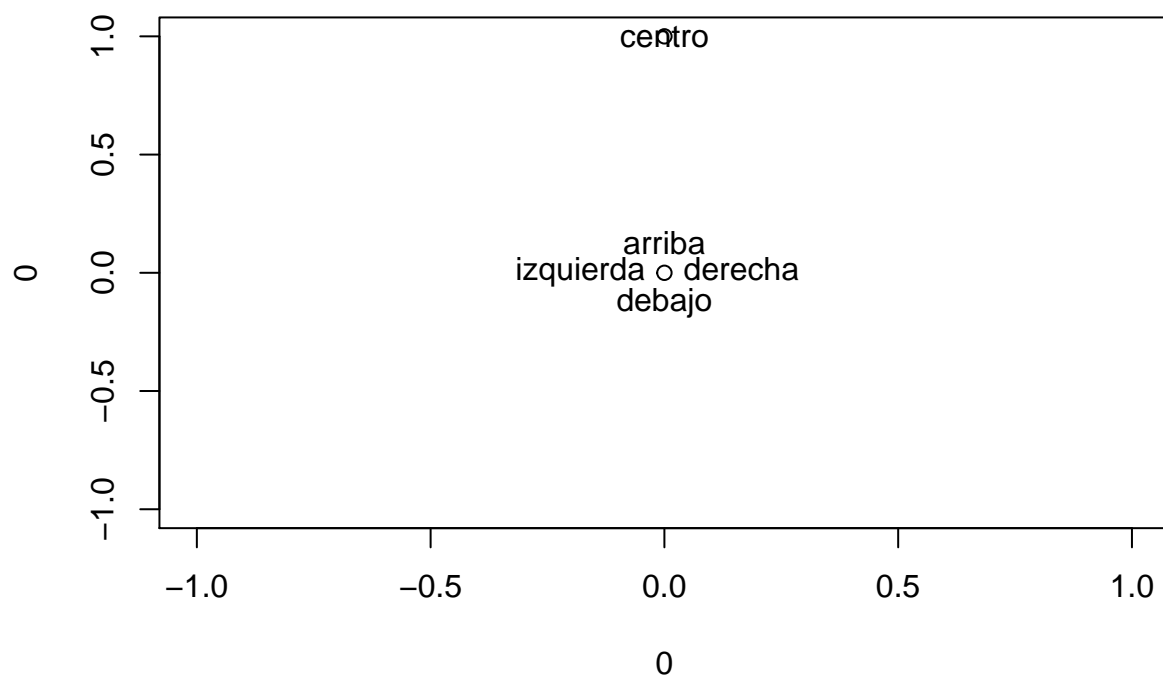
```
plot(tan, xlim = c(-pi,pi), ylim = c(-5,5))  
abline(v = c(-pi/2, pi/2), col = "red")
```



- **text(x,y,labels = "...")**: añade en el punto de coordenadas (x, y) el texto especificado como argumento de labels
 - **pos**: permite indicar la posición del texto alrededor de las coordenadas (x, y) . Admite los siguientes valores:
 - * 1: abajo
 - * 2: izquierda
 - * 3: arriba
 - * 4: derecha
 - * 5: sin especificar: el texto se sitúa centrado en el punto (x, y)

Ejemplo:

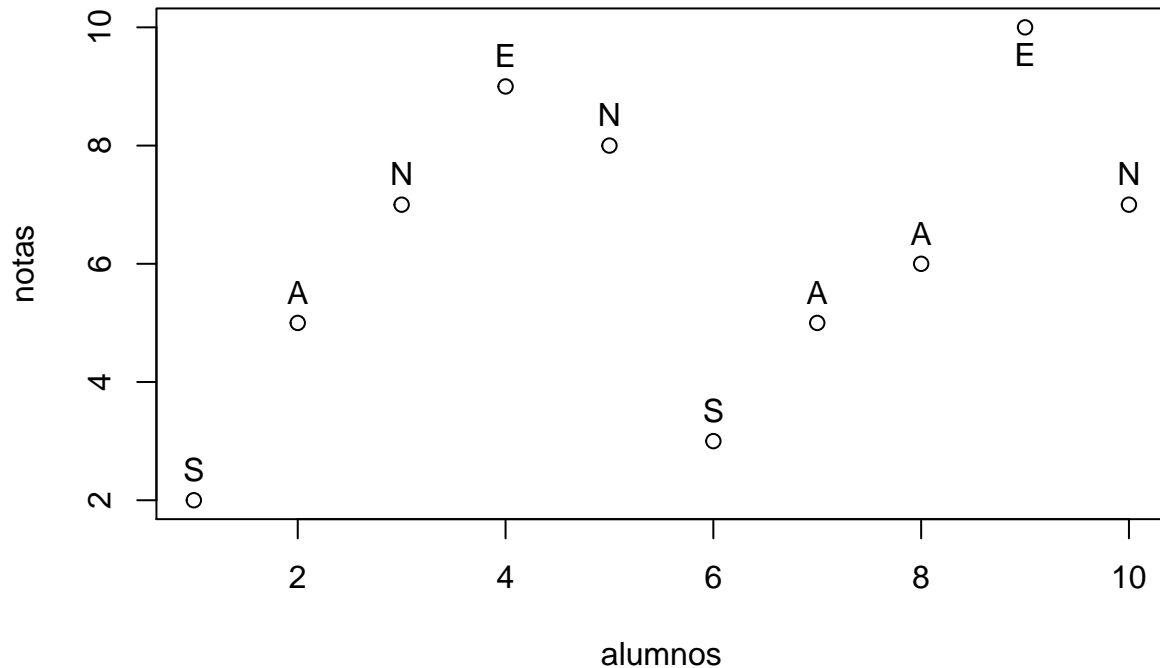
```
plot(0,0)
text(0,0, labels = "debajo", pos = 1)
text(0,0, labels = "izquierda", pos = 2)
text(0,0, labels = "arriba", pos = 3)
text(0,0, labels = "derecha", pos = 4)
points(0,1)
text(0,1, labels = "centro")
```

Ejemplo más complejo

```
alumnos <- c(1:10)
notas <- c(2,5,7,9,8,3,5,6,10,7)
plot(alumnos, notas, main = "Gráfico con texto")
text(alumnos, notas, labels = c("S", "A", "N", "E", "N", "S", "A", "A", "E", "N"),
      pos = c(rep(3, times = 8), 1, 3))
```

Gráfico con texto



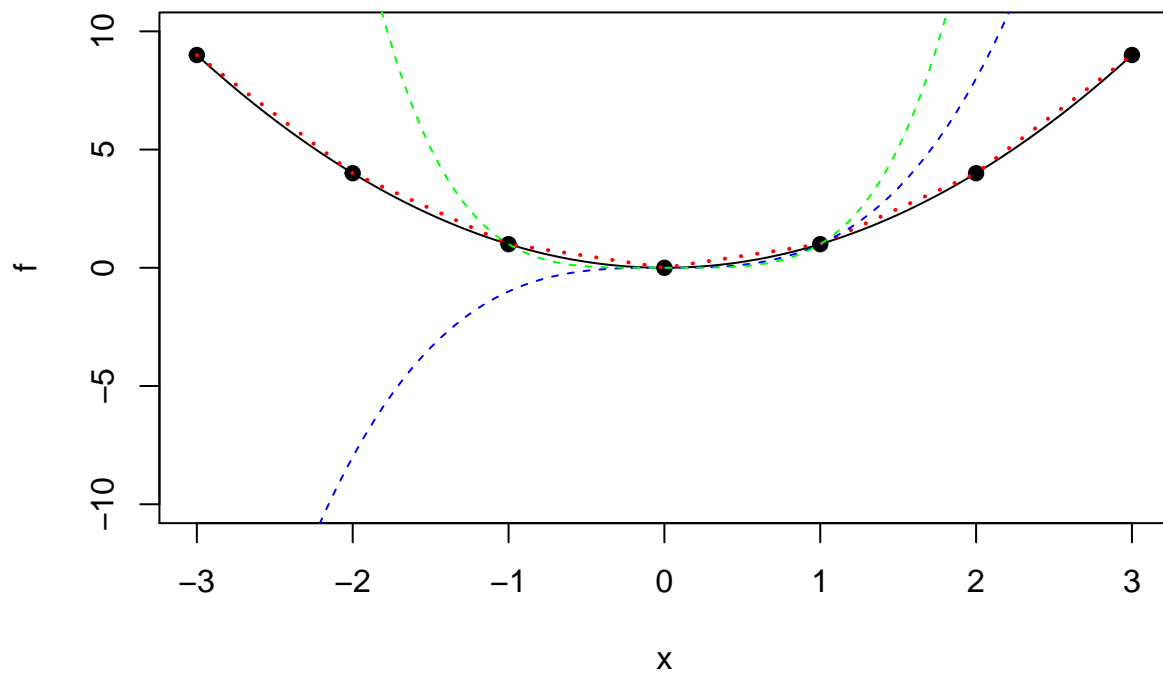
* **lines(x, y)**: añade a un gráfico existente una línea poligonal que une los puntos (x_i, y_i) sucesivos. x , y son vectores numéricos

- **curve(curva)**: permite añadir la gráfica de una curva a un gráfico existente
 - **add=TRUE**: si no, la curva no se añade.
 - La curva se puede especificar mediante una expresión algebraica con variable x , o mediante su nombre si la hemos definido antes

Un ejemplo de como plotar una línea. Plotar una función y despues poner una línea.

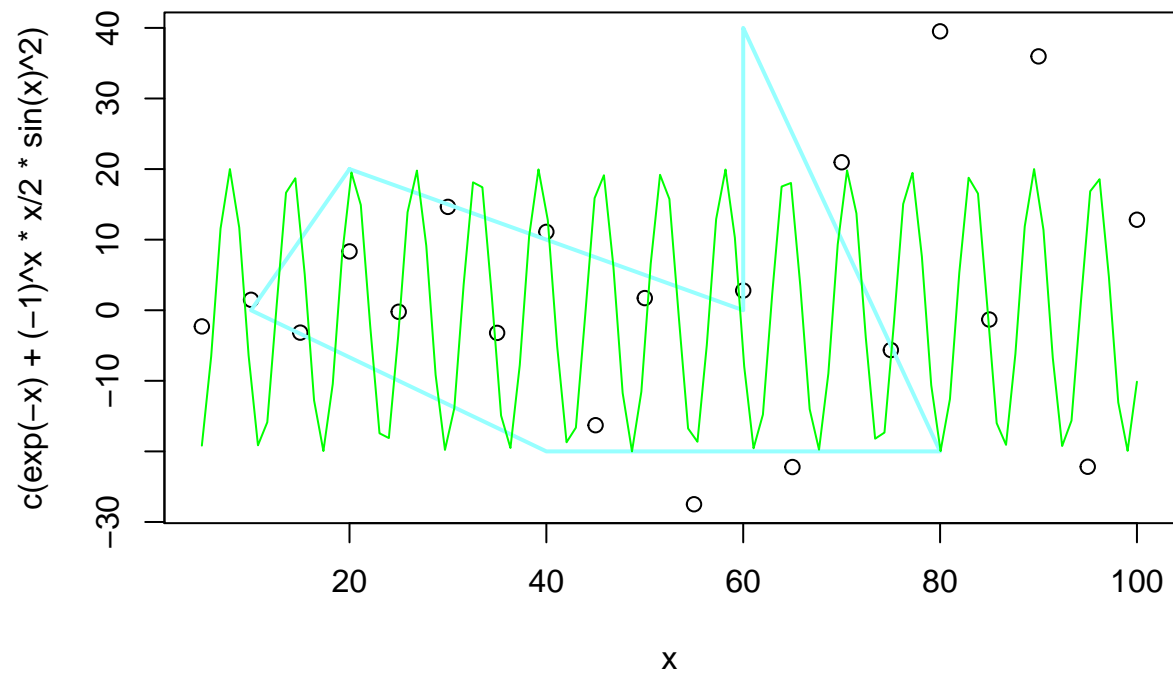
```
#crear una función, ese caso una parábola
f <- function(x){
  x^2
}
#Plotar la función dentro de un límite de -3 a 3.
plot(f, xlim = c(-3, 3), ylim = c(-10,10))
#Mostrar los puntos donde unirá las rectas
points(-3:3, f(-3:3), pch = 19)
#Plotar las rectas en los puntos creados.
lines(-3:3,f(-3:3),lwd = 2, lty = "dotted", col = "red")

#Añadir una curva
curve(x^3, lty = "dashed", col = "blue", add = TRUE)
curve(x^4, lty = "dashed", col = "green", add = TRUE)
```



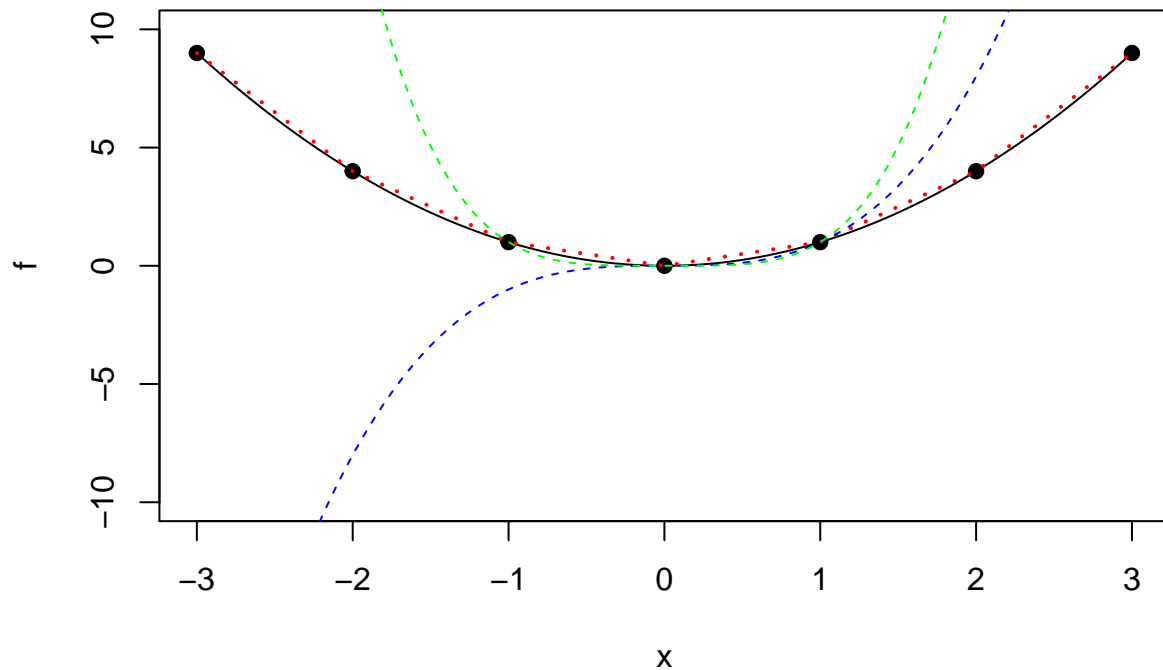
Otro ejemplo mas complejo

```
x = c(5*(1:20))
plot(x, c(exp(-x)+(-1)^x*x/2*sin(x)^2))
lines(c(20,10,40,80,60,60,20),c(20,0,-20,-20,40,0,20), lwd = 2, col = "darkslategray1")
curve(20*sin(x), add = TRUE, col = "green")
```



Leyendas en los gráficos

Muchas veces es difícil identificar de cual función pertenece a una gráfica, por ejemplo, la gráfica abajo lo tiene difícil de identificar a quien pertenece cada gráfica.

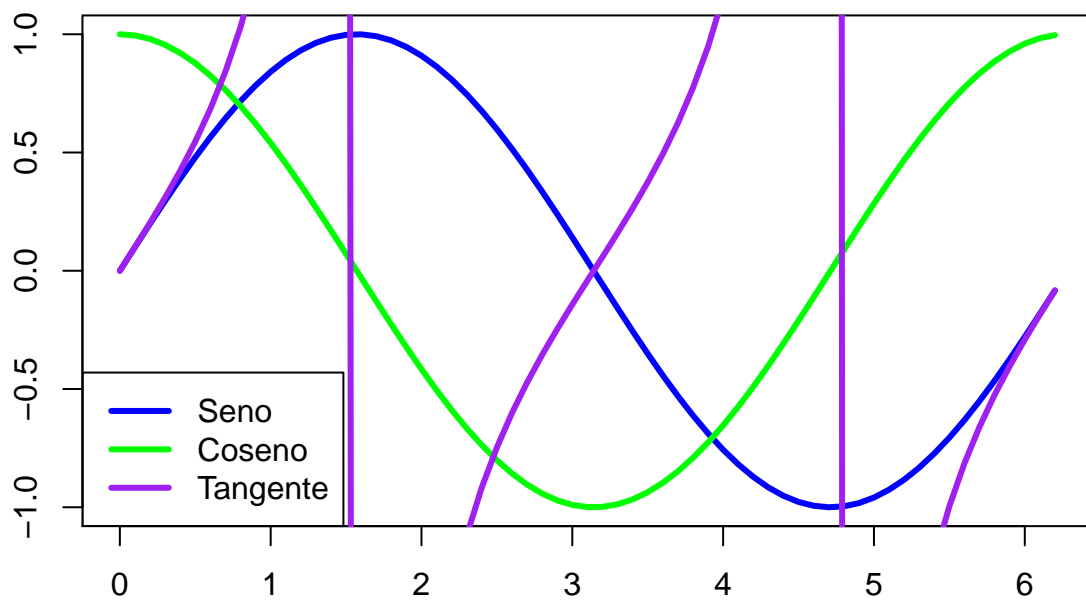


Para facilitar la identificación se hace uso de las leyendas en los gráficos.

- **legend(posición, legend = ...)**: para añadir una leyenda
 - La posición indica donde queremos situar la leyenda, Puede ser o bien las coordenadas de la esquina superior izquierda de nuestra leyenda, o bien una de las palabras siguientes:
 - * “bottom” / “bottomright” / “botomleft”
 - * “top” / “topright” / “topleft”
 - * “center” / “right” / “left”
 - **legend**: contiene el vector de nombres entre commillas con los que queremos identificar a las curvas en la leyenda.

Ejemplo:

```
x <- seq(0, 2*pi, 0.1)
plot(x, sin(x), type = "l", col = "blue", lwd = 3, xlab = "", ylab = "")
lines(x, cos(x), col = "green", lwd = 3)
lines(x, tan(x), col = "purple", lwd = 3)
legend("bottomleft", col = c("blue", "green", "purple"),
      legend = c("Seno", "Coseno", "Tangente"), lwd = 3, bty = "l")
```



otro ejemplo:

```
#crear una función, ese caso una parábola
f <- function(x){
  x^2
}
#Plotar la función dentro de un límite de -3 a 3.
plot(f, xlim = c(-3, 3), ylim = c(-10,10))
#Mostrar los puntos donde unirá las rectas
points(-3:3, f(-3:3), pch = 19)
#Plotar las rectas en los puntos creados.
lines(-3:3,f(-3:3),lwd = 2, lty = "dotted", col = "red")

#Añadir una curva
curve(x^3, lty = "dashed", col = "blue", add = TRUE)
curve(x^4, lty = "dashed", col = "green", add = TRUE)

#añadiendo la leyenda
legend(
  "bottomright",
  legend = c(expression(x^2), expression(x^3), expression(x^4)),
  lwd = 2,
  col = c("red", "blue", "green"),
  lty = c("dotted", "dashed", "dashed")
)
```

