Graficos

Martin Malo

2/7/2021

Graficos de la funcion `plot`

plot(x, y)

Con `fig.cap=`el chunk va a colocar una nota que le demos al grafico, y con `fig.align=`el chunk va a colocar el grafico segun le indiquemos

```
alumnos <- (1:10)
notas <- c(4,2,6,7,4,6,3,9,5,2)
plot(alumnos, notas)
```

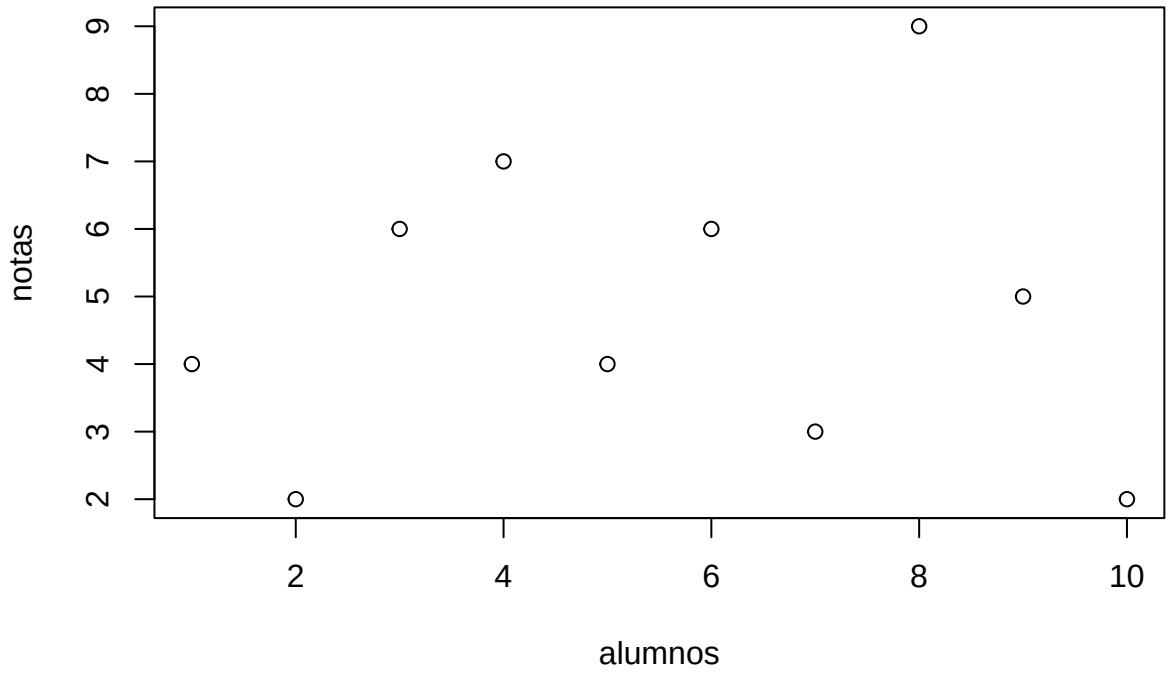
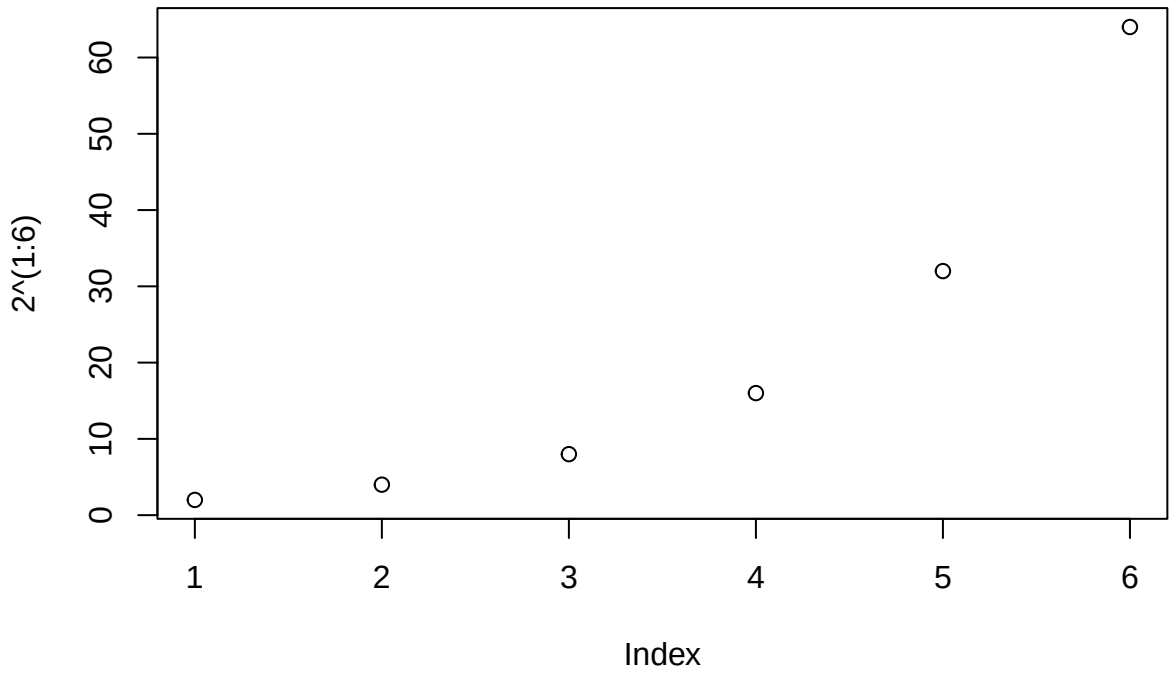


Figure 1: Grafico basico con datos de alumnos y notas

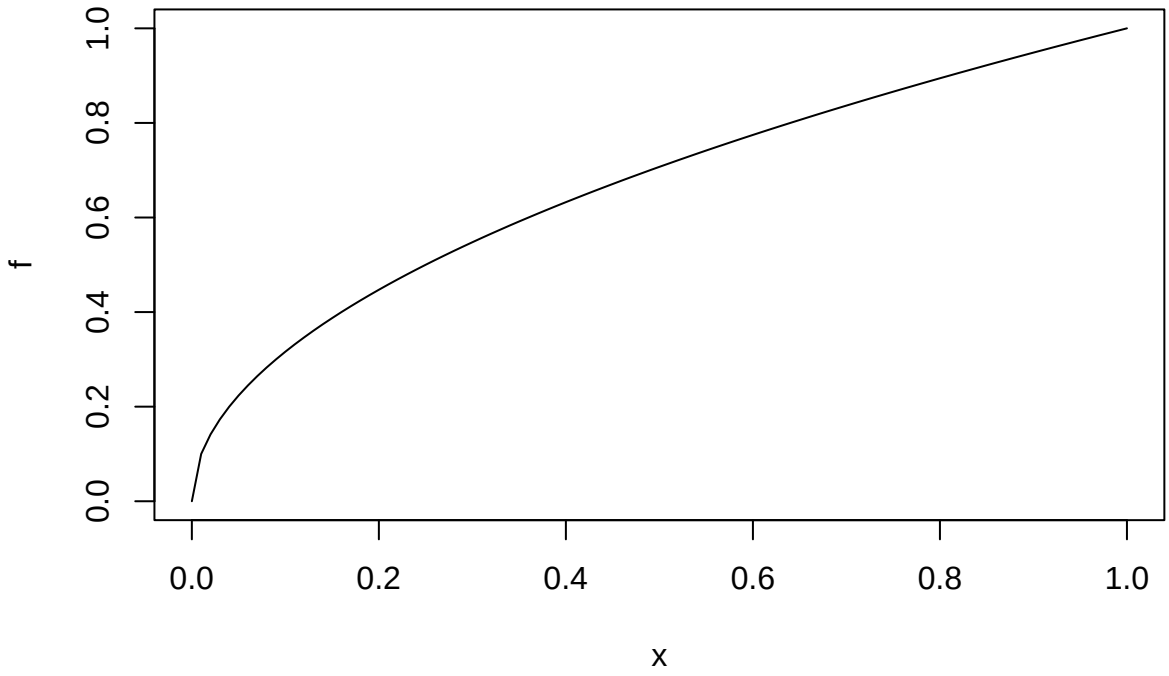
Si incorporamos vector `y`, `R`nos va tomar el parametro `x` como si fuese el vector de datos `y`. plot(1:n, x). Es decir, si solo damos un parametro a la funcion `plot`, `R`lo que hace es tomar ese parametro y ubicarlo en el eje de las `y` siempre.

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



Si queremos representar una funcion $f(x)$

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

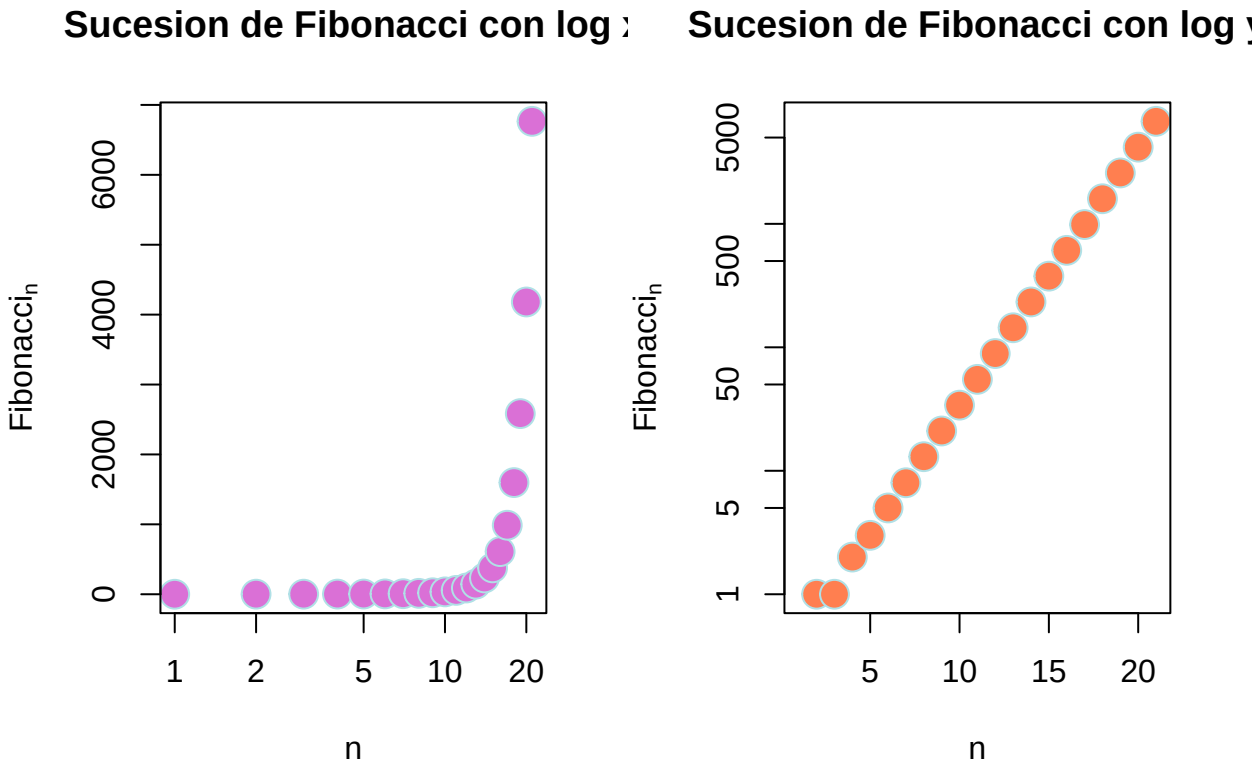


Parametros de la funcion `plot`

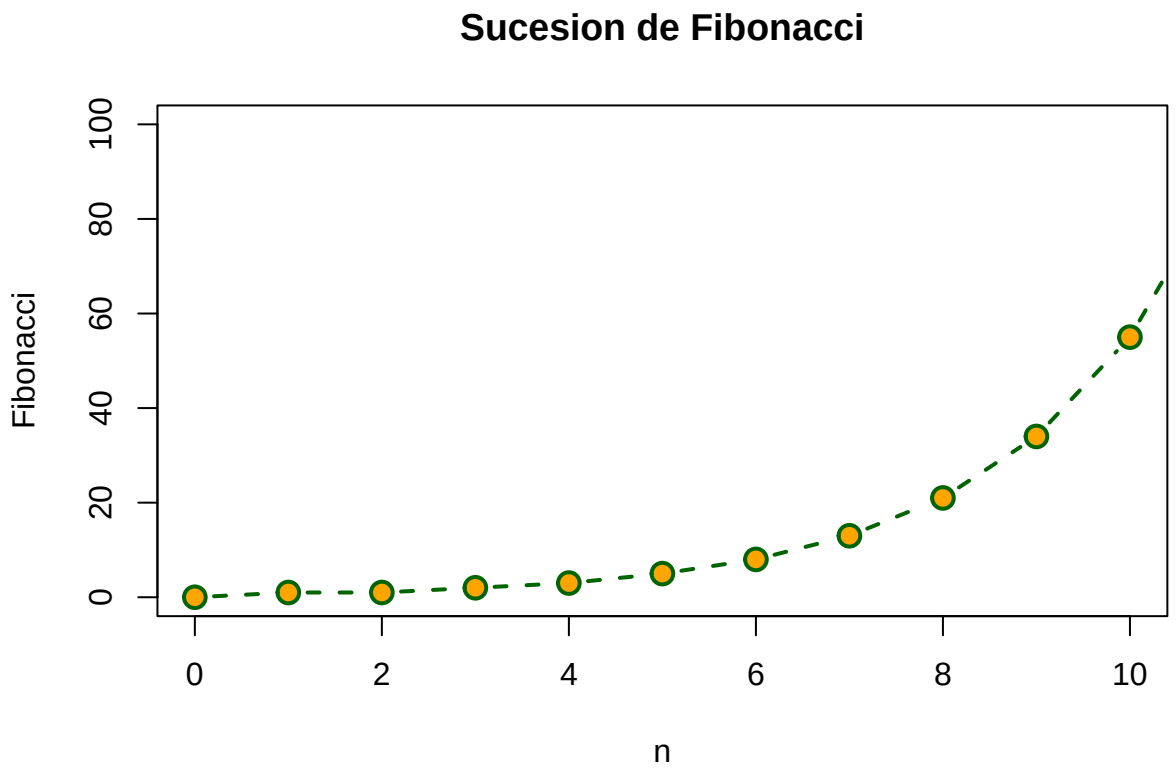
- Parametros
  - `log =` ajusta el rango del eje x, y o ambos de forma logaritmica
  - `xlab =` etiqueta para el eje x
  - `ylab =` etiqueta para el eje y
  - `expression()` etiquetas en formato similar a $LaTeX$
  - `main` ingresar titulo al grafico
  - `pch =` tipo de punto del grafico
  - `cex =` tamaño del punto del grafico. Por defecto esta = 1
  - `col =` color del punto del grafico y para ciertos puntos colorea bordes
  - `bg =` cuando `col =`colorea los bordes, este parametro colorea el punto
- **Si queremos ubicar dos graficos uno a lado de otros se debe utilizar, antes de la escritura del codigo, `par(mfrow = )`que crea una matriz de graficos y a partir del = indicar como colocarlos. Para el siguiente grafico queremos colocar 2 graficos en la misma fila entonces debemos indicar `c(1,2)` done 1 significa una fila y el 2 significa 2 columnas. Por ultimo, cuando hayamos cabado de escribir los codigos de los graficos debemos ingresar nuevamente `par(mfrow = )`.**

```
##  [1]    0    1    1    2    3    5    8   13   21   34   55   89  144  233  377
## [16]  610  987 1597 2584 4181 6765
```

```
## Warning in xy.coords(x, y, xlabel, ylabel, log): 1 y value <= 0 omitted from
## logarithmic plot
```



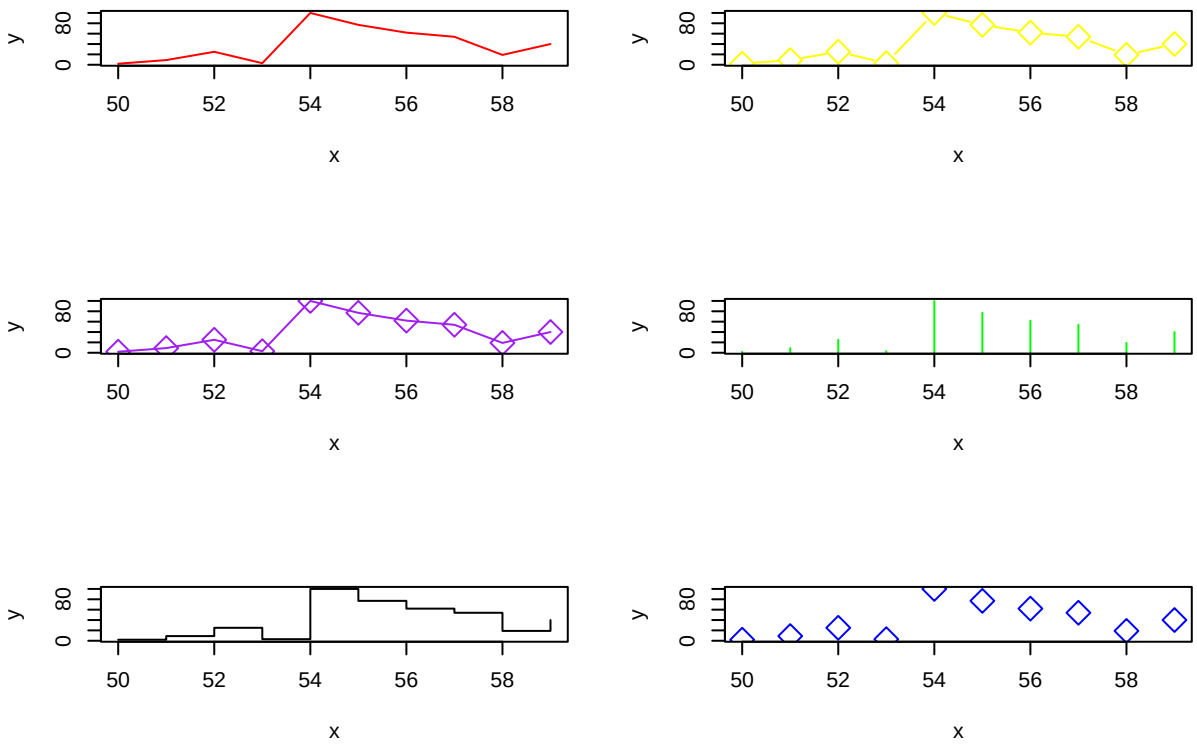

```
plot(n, fibonacci, type = "b", main = "Sucesion de Fibonacci", xlab = "n", ylab = "Fibonacci", pch = 21
     col = "darkgreen", bg = "orange", lty = 2, lwd = 2, xlim = c(0,10), ylim = c(0,100))
```



Más parametros

- `type =` para elegir el tipo de grafico que queremos
  - `p =` puntos (valor por defecto)
  - `l =` lineas rectas que unen puntos (dichos puntos no tienen simbolo)
  - `b =` (both) lineas rectas que unen puntos (dichas puntos tienen simbolo). Las lineas no traspasan los puntos
  - `o =` como el anterior pero ne este caso las lineas si traspasan los puntos
  - `h =` histograma de lineas
  - `s =` histograma de escalones
  - `n =` para no dibujar los puntos

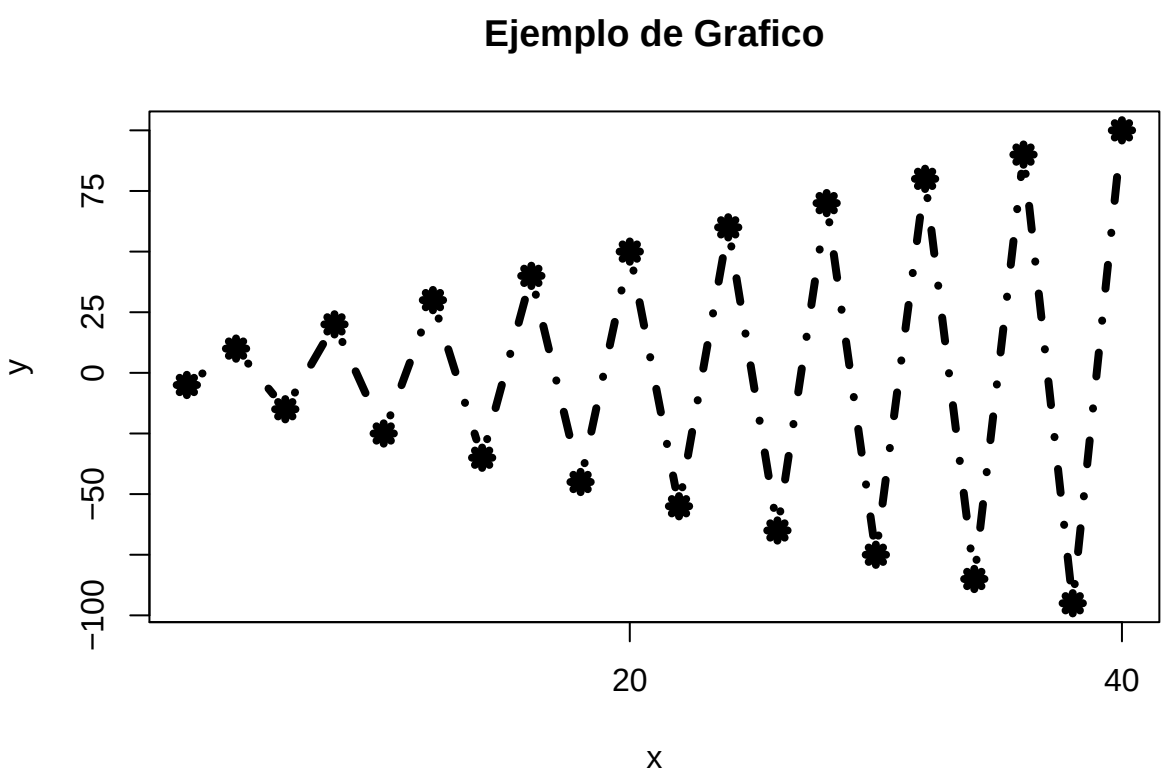
```
par(mfrow = c(3,2))
x <- 50:59
y <- c(2,9,25,3,100,77,62,54,19,40)
plot(x, y, pch = 23, cex = 2, col = "red", type = "l")
plot(x, y, pch = 23, cex = 2, col = "yellow", type = "b")
plot(x, y, pch = 23, cex = 2, col = "purple", type = "o")
plot(x, y, pch = 23, cex = 2, col = "green", type = "h")
plot(x, y, pch = 23, cex = 2, col = "black", type = "s")
plot(x, y, pch = 23, cex = 2, col = "blue", type = "p")
```




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

- `lty =` para especificar el tipo de linea
  - "solid" o 1: linea continua (por defecto)
  - "dashed" o 2: linea discontinua
  - "dotted" o 3: linea de puntos
  - "dotdashed" o 4: linea que alterna puntos y rayas
- `lwd =` para especificar el grosor de las lineas
- `xlim =` para modificar el rango del eje x
- `ylim =` para modificar el rando del eje y
- `xaxp =` para modificar posiciones de las marcas en el eje x
- `yaxp =` para modficiar 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 Grafico", pch = 8, cex = 1, type = "b", lty = 4,
     lwd = 4, xaxp = c(0,40,2), yaxp = c(-100,100,8))
```


Ejemplo de Grafico
y
75
25
0
−50
−100
20
40
x