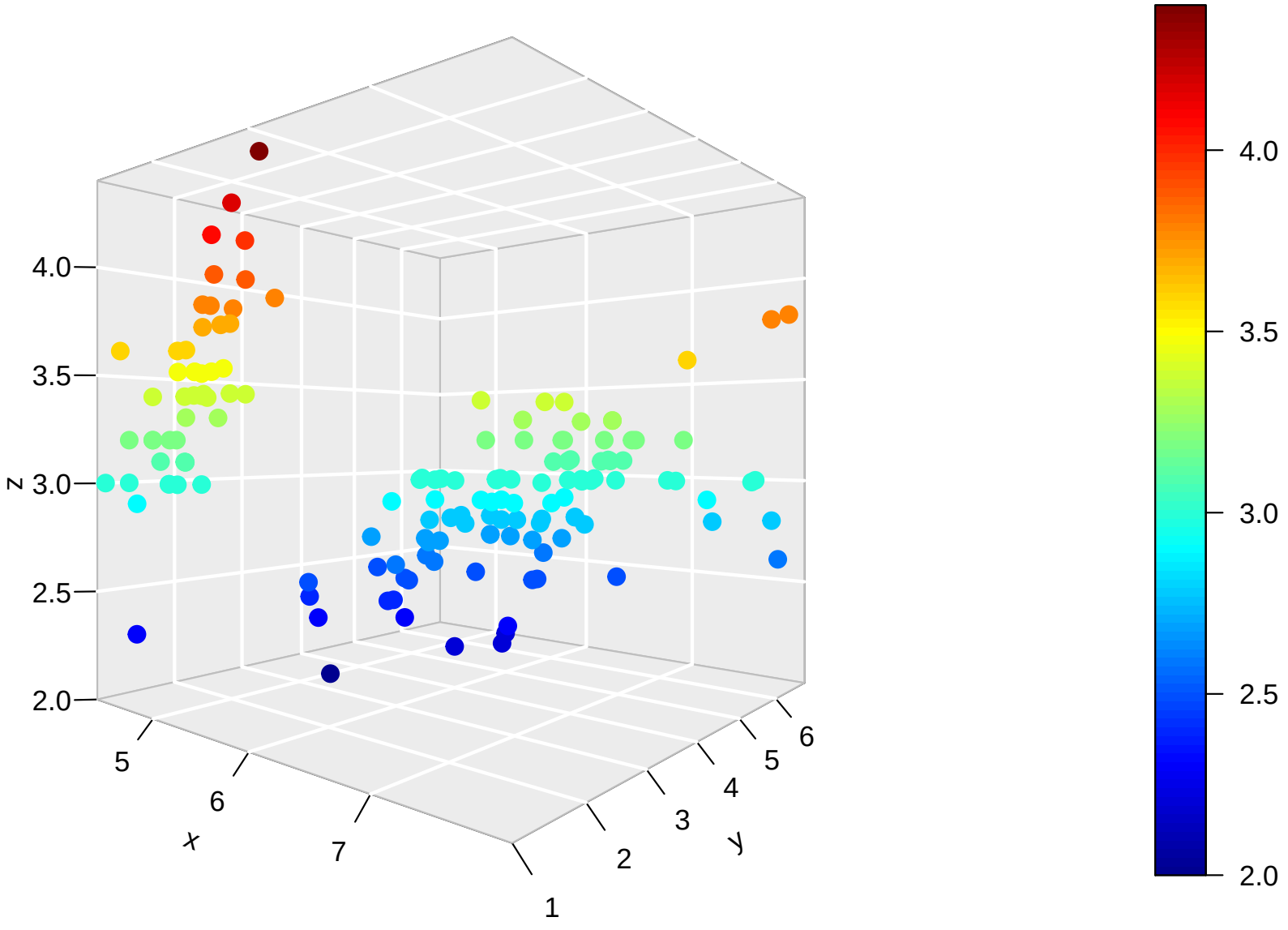
# R language and data analysis:plot basics

Qiang Shen

Oct.8,2016

# Visualization

- A picture is worth a thousand words



## Visualization

- "The adage 'A picture is worth a thousand words' refers to the notion that a complex idea can be conveyed with just a single still image, or that an image of a subject conveys its meaning/essence more effectively than a description does. It also aptly characterizes one of the main goals of visualization namely making it possible to absorb large amounts of data quickly." –Wikipedia

## R packages

- graphics: built-in packages
- lattice
- ggplot2: R's famous package for making beautiful graphics.
- ggvis: Interactive, web based graphics built with the grammar of graphics.
- rgl: Interactive 3D visualizations with R

## content

- create and save graphs

## content

- create and save graphs
- customize plot: symbols,lines,colour

## content

- create and save graphs
- customize plot: symbols,lines,colour
- annotate with text and titles

## content

- create and save graphs
- customize plot: symbols,lines,colour
- annotate with text and titles
- combined graphs

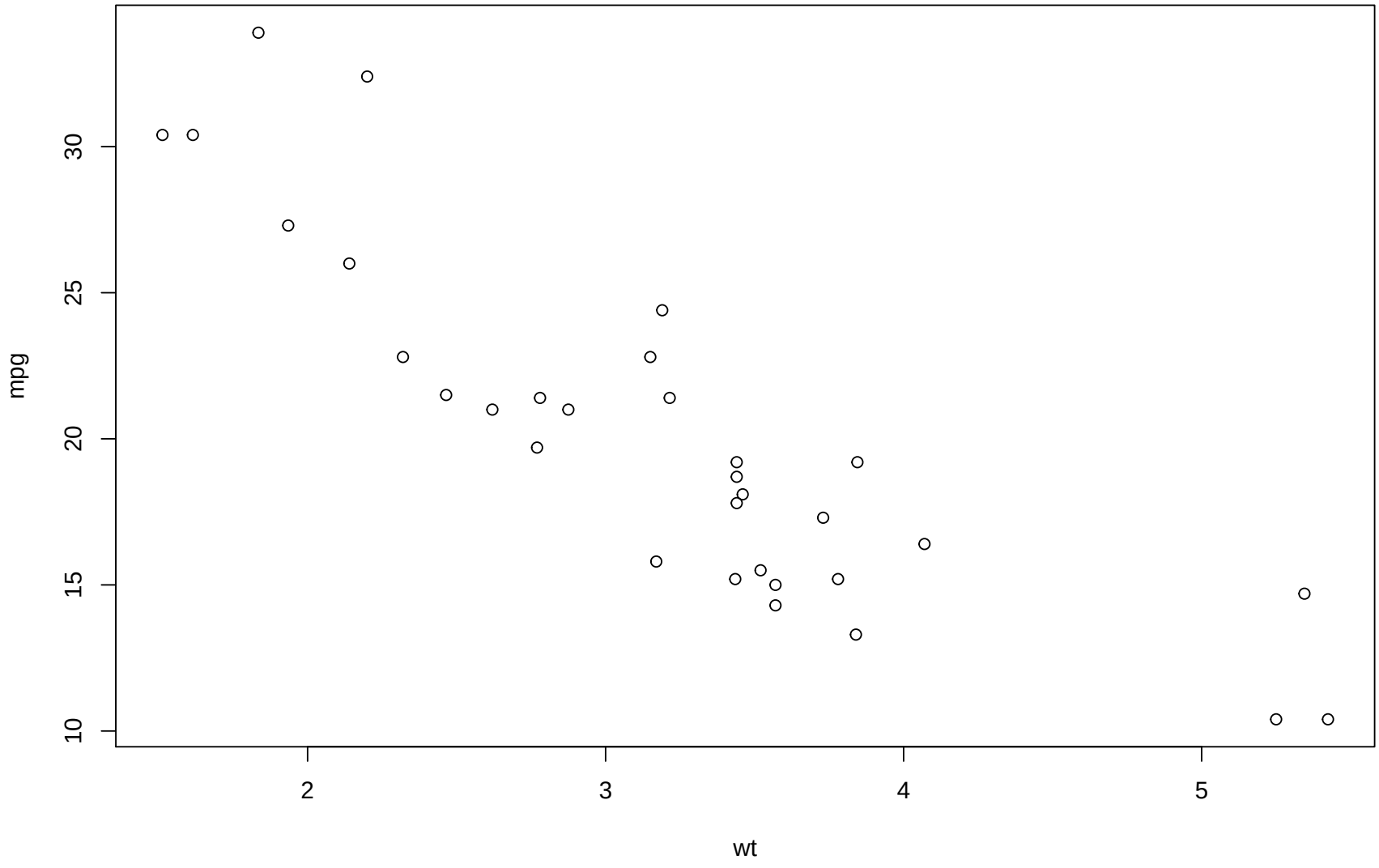
## demo

```
demo(graphics)
```

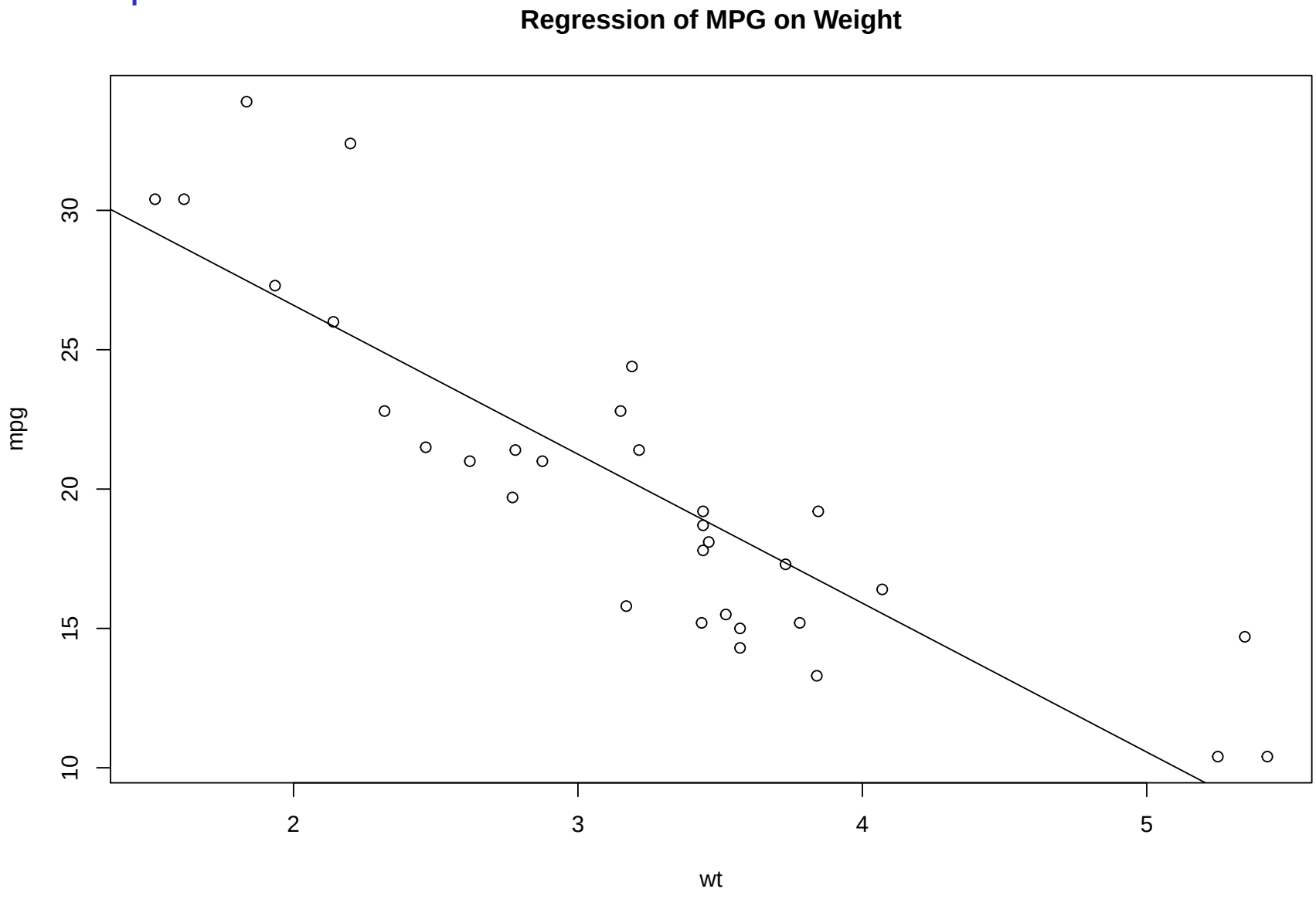
## content

- *create and save graphs*

# basic plot



# basic plot continued



## Export the figure

```
pdf("mtcars_demo.pdf")
with(mtcars,plot(wt,mpg))
abline(lm(mpg~wt,data=mtcars))
title("Regression of MPG on Weight")
dev.off()
```

## Export the figure

- vector (矢量图): pdf
- bitmap (位图): png,jpg,bmp,tiff

| Function | Output |
|---|---|
| `bmp("filename.bmp")` | BMP file |
| `jpeg("filename.jpg")` | JPEG file |
| `pdf("filename.pdf")` | PDF file |
| `png("filename.png")` | PNG file |
| `postscript("filename.ps")` | PostScript file |

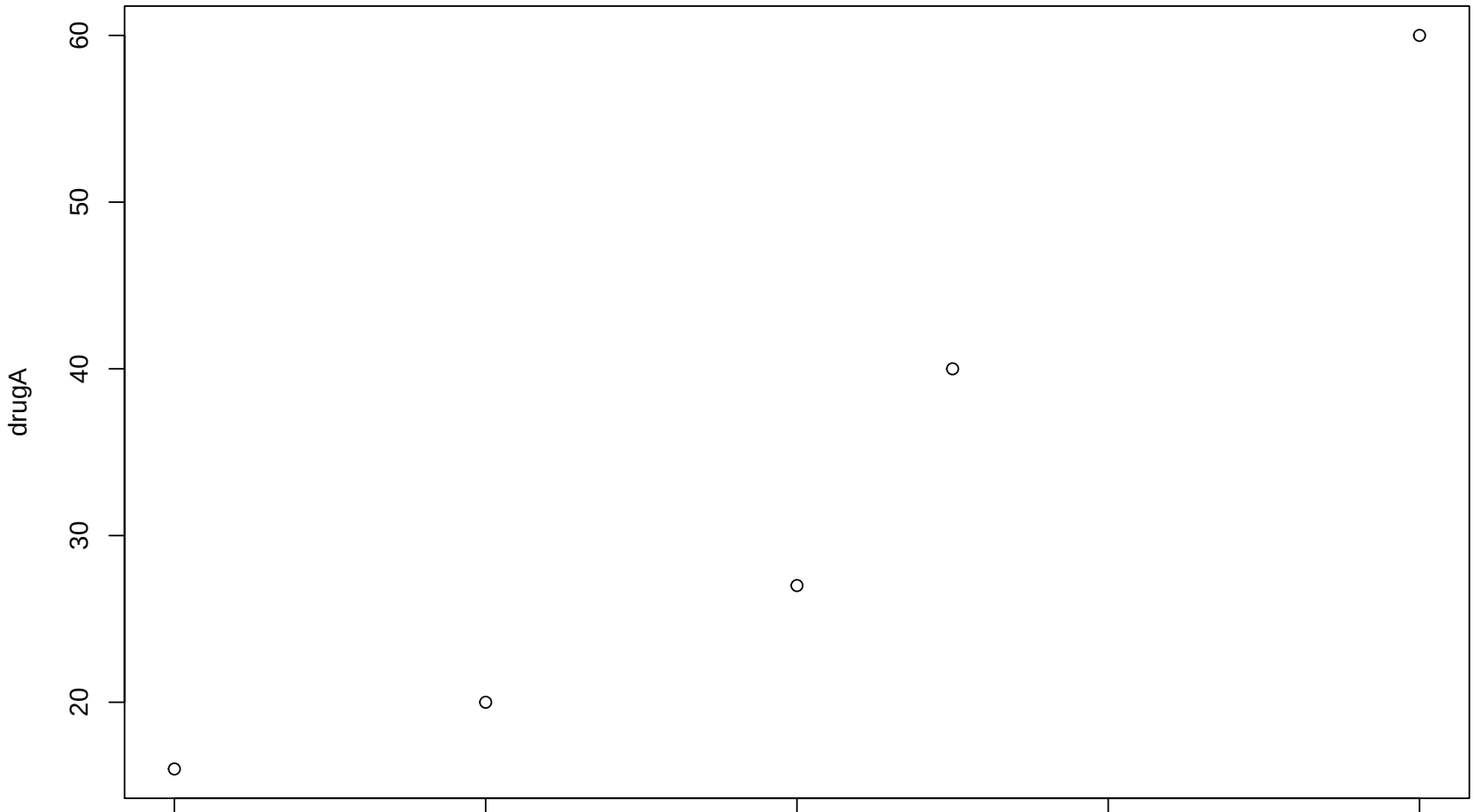
图 1:

## content

- create and save graphs
- *customize plot: symbols,lines,colour*

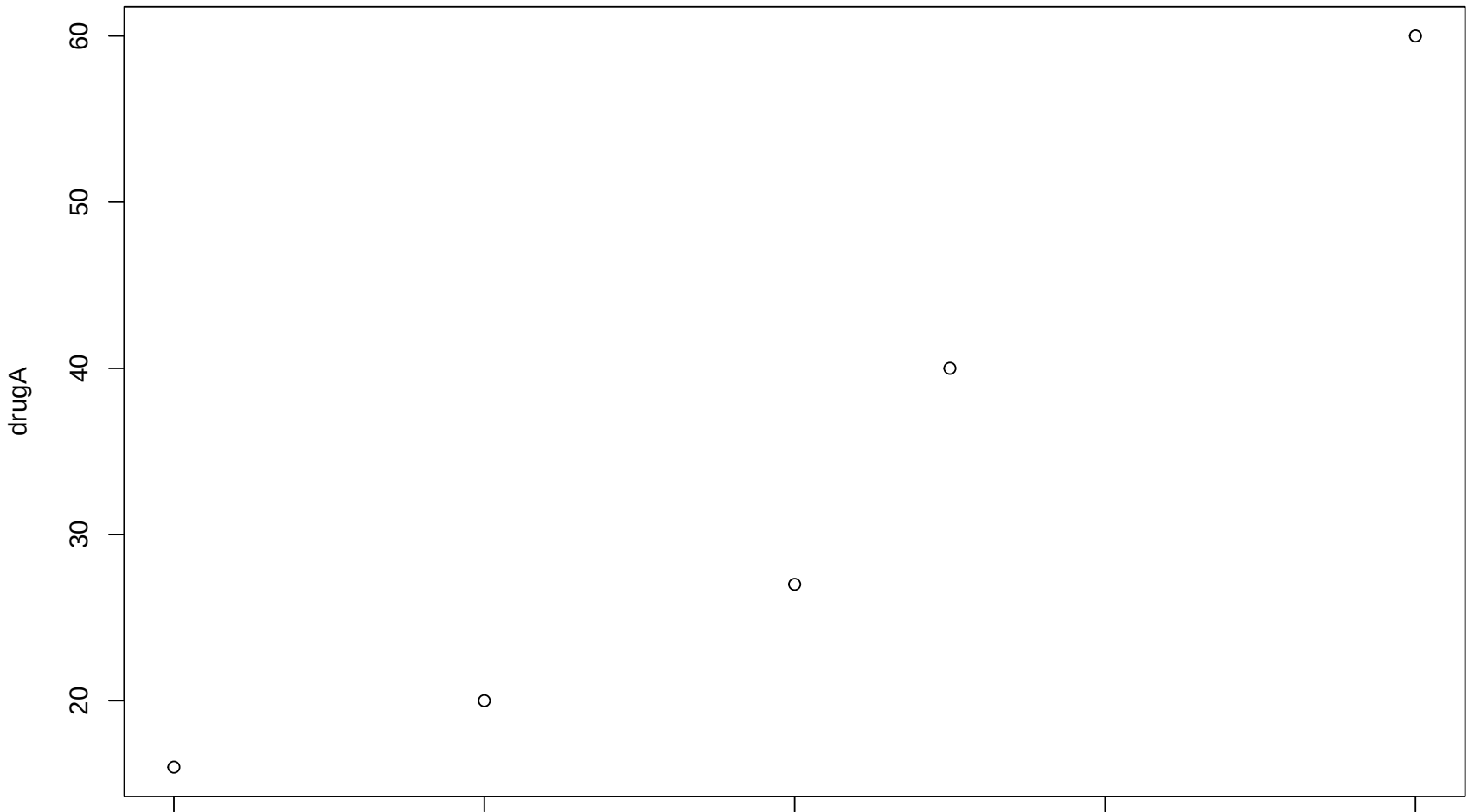
## plot: default type:**point**

```
dose <- c(20, 30, 40, 45, 60)
drugA <- c(16, 20, 27, 40, 60)
drugB <- c(15, 18, 25, 31, 40)
plot(dose, drugA) #default type:point (type='p')
```



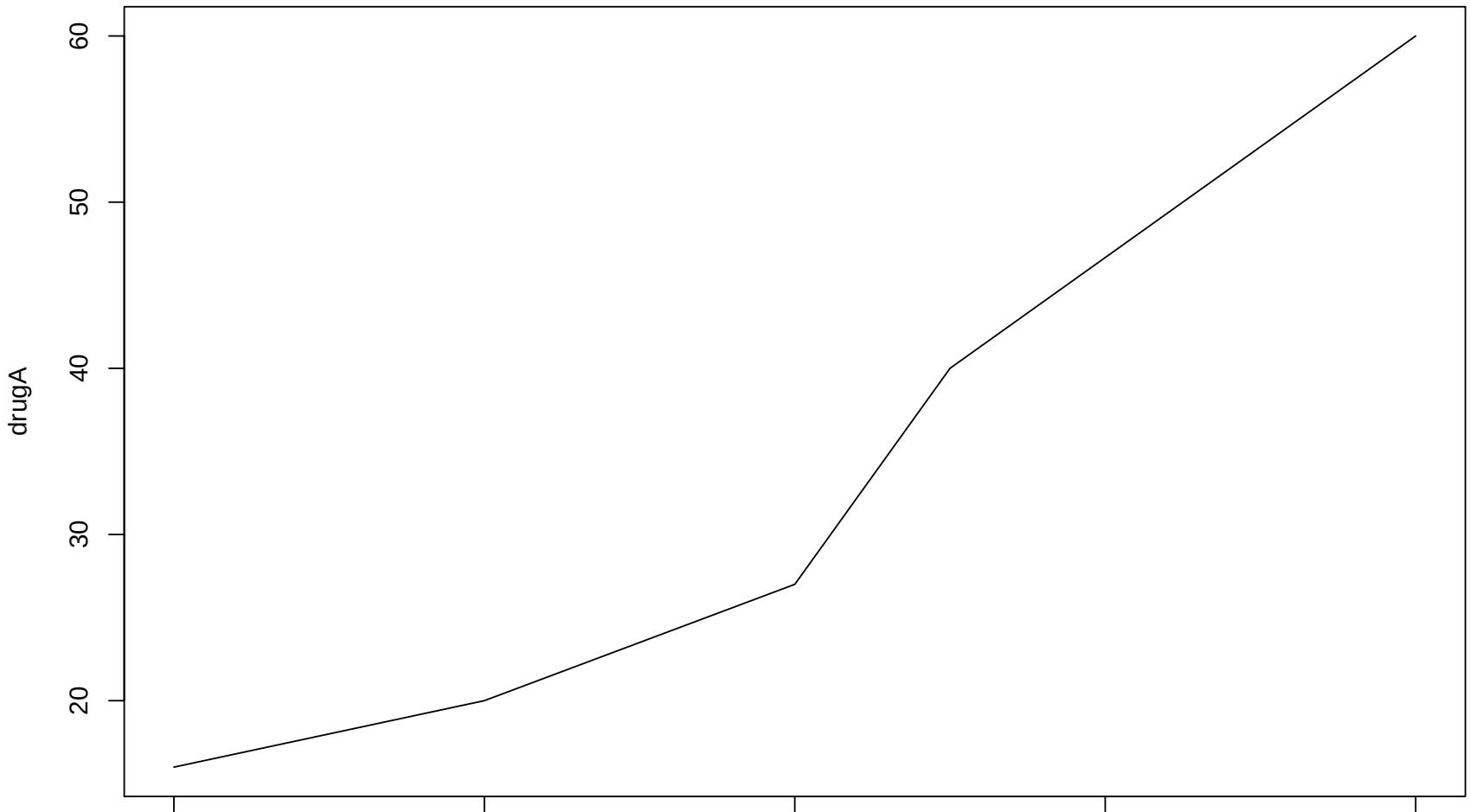
# plot:**point**

```
dose <- c(20, 30, 40, 45, 60)
drugA <- c(16, 20, 27, 40, 60)
drugB <- c(15, 18, 25, 31, 40)
plot(dose, drugA,type='p')
```



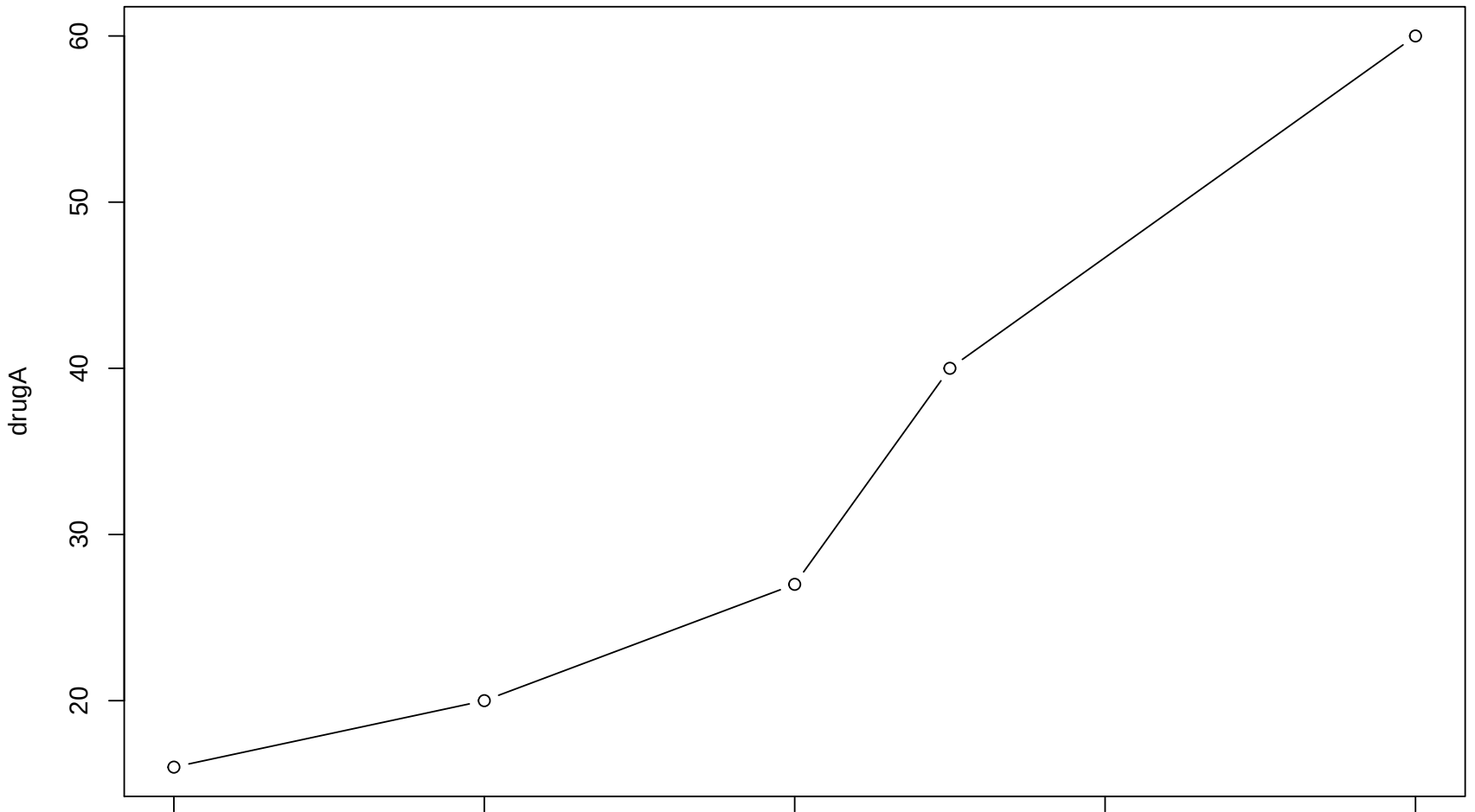
# plot:**line**

```
dose <- c(20, 30, 40, 45, 60)
drugA <- c(16, 20, 27, 40, 60)
drugB <- c(15, 18, 25, 31, 40)
plot(dose, drugA, type = "l")
```



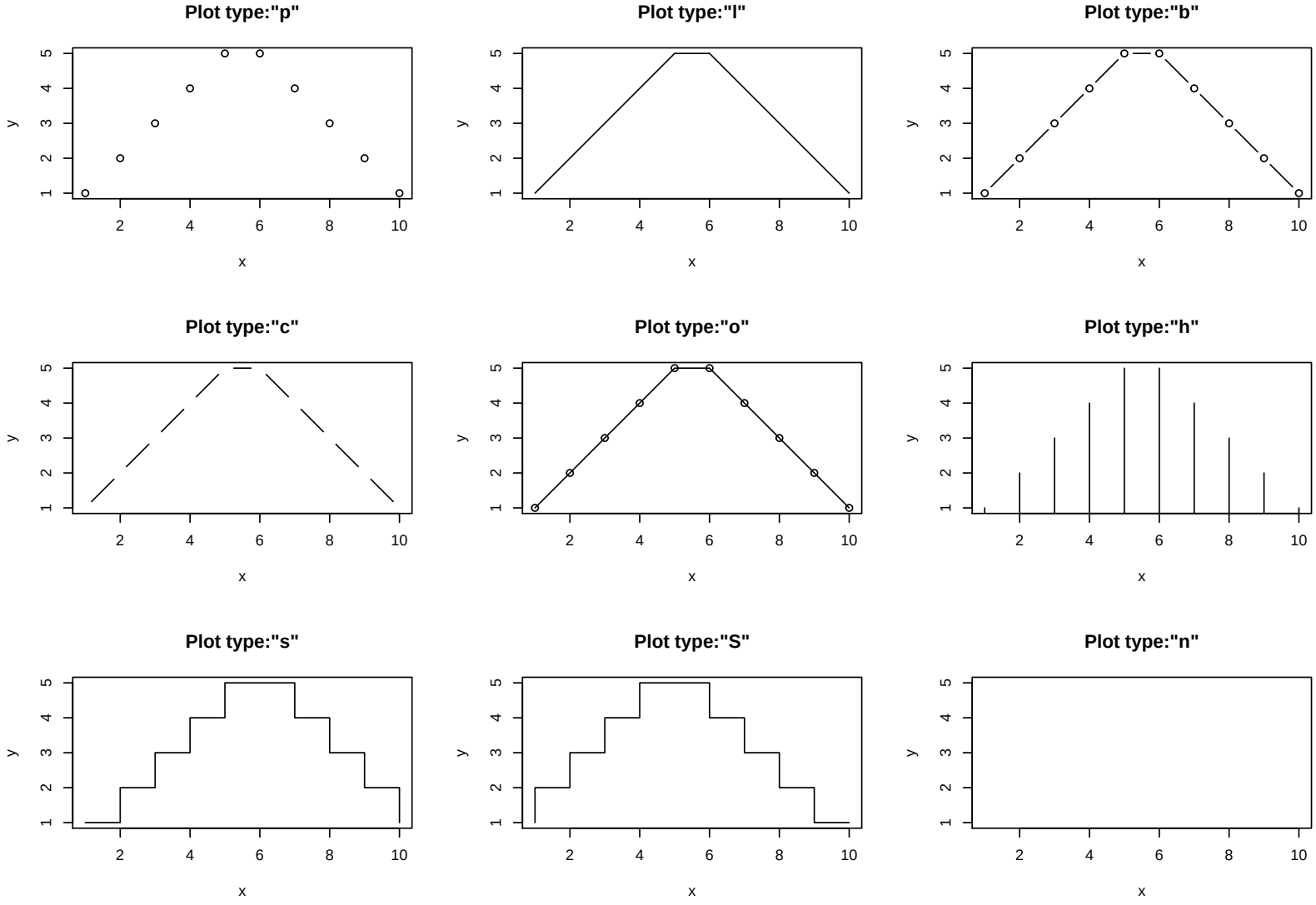
# plot: **point** and **line**

```
dose <- c(20, 30, 40, 45, 60)
drugA <- c(16, 20, 27, 40, 60)
drugB <- c(15, 18, 25, 31, 40)
plot(dose, drugA, type = "b")
```



# type parameter

- type parameter

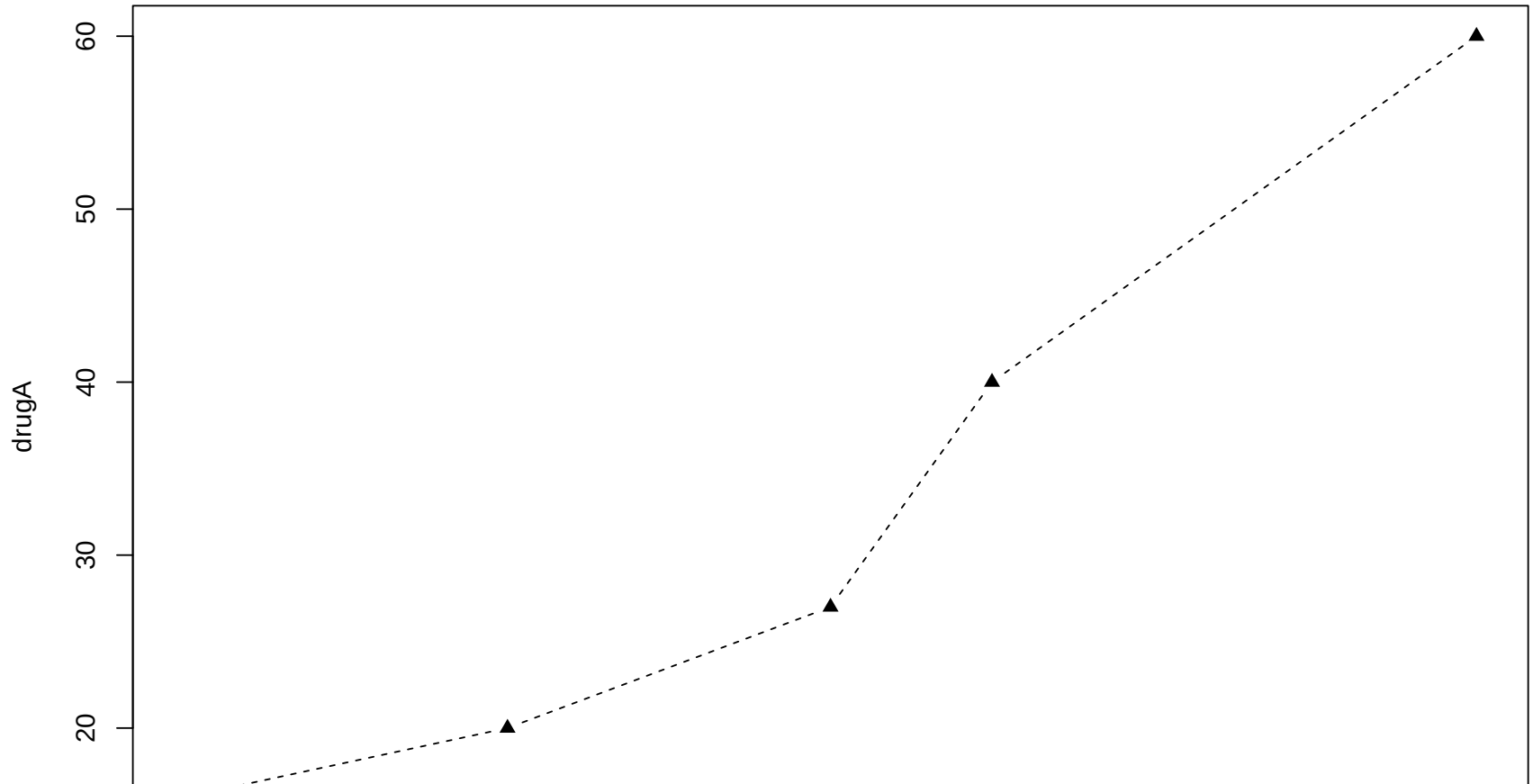



```
## [[1]]
```

## plot:line type and point type

- line type (`type='l'`): `lty`
- point type (`type='p'`): `pch`

```
opar <- par(no.readonly = TRUE)## get the default parameter
plot(dose, drugA, type = "b", lty = 2, pch = 17)
```



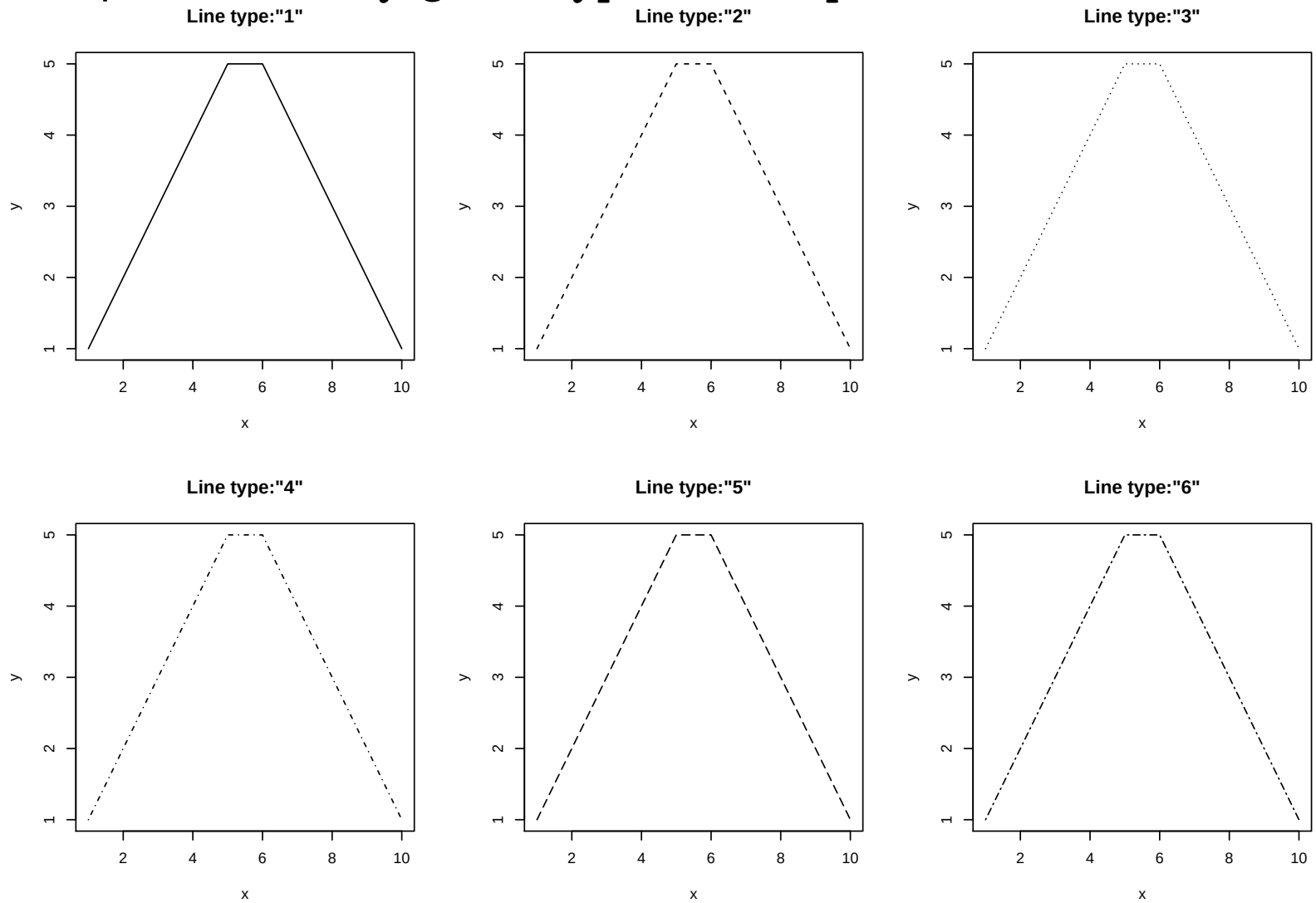
# Parameters for symbols and lines

| Parameter | Description |
|---|---|
| `pch` | Specifies the symbol to use when plotting points (see figure 3.4). |
| `cex` | Specifies the symbol size. `cex` is a number indicating the amount by which plotting symbols should be scaled relative to the default. 1 = default, 1.5 is 50% larger, 0.5 is 50% smaller, and so forth. |
| `lty` | Specifies the line type (see figure 3.5). |
| `lwd` | Specifies the line width. `lwd` is expressed relative to the default (1 = default). For example, `lwd=2` generates a line twice as wide as the default. |

图 2:

# line parameter

- line parameter `lty` given `type='l'` in `plot` function




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

# line parameter:line type

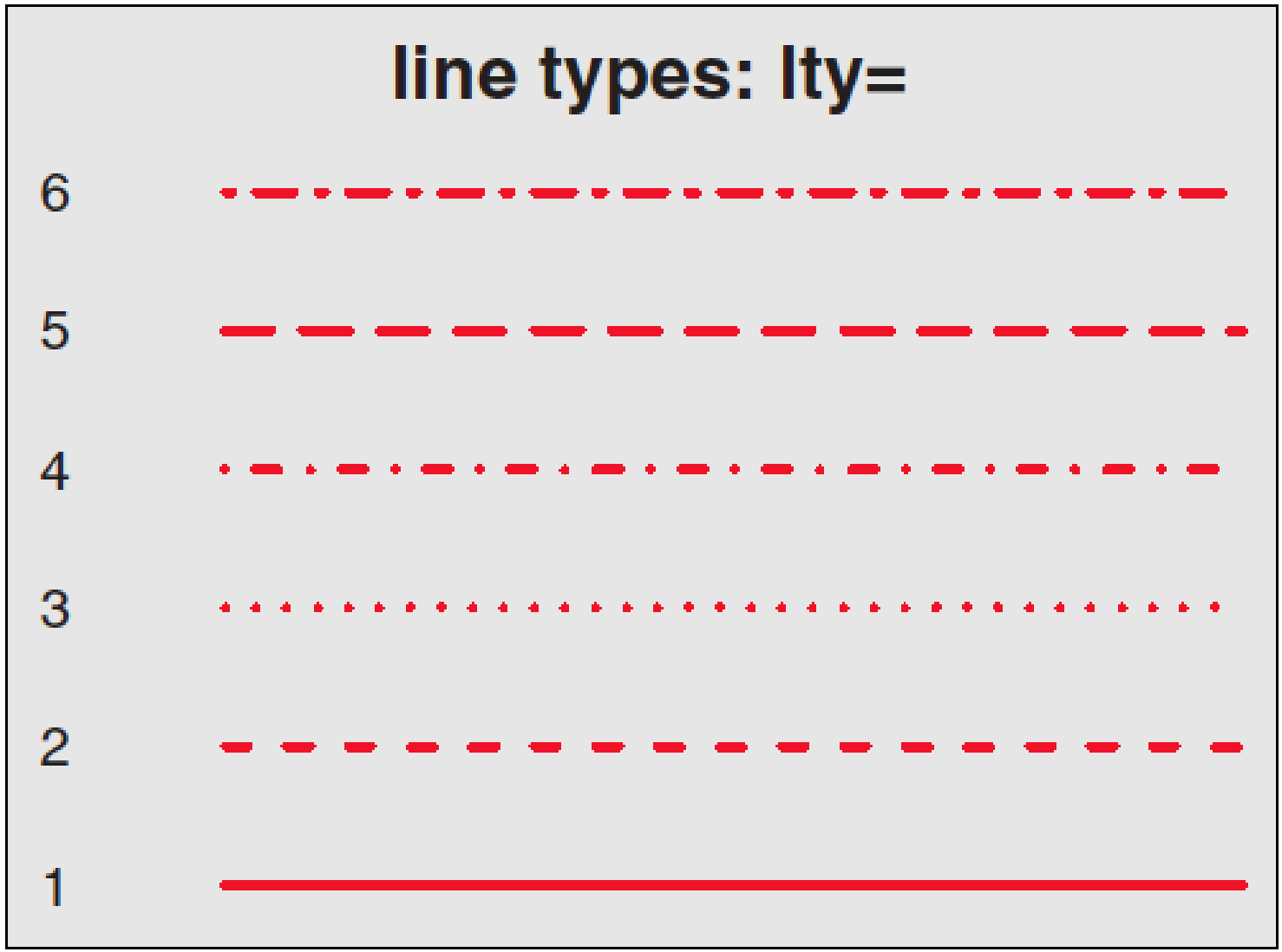
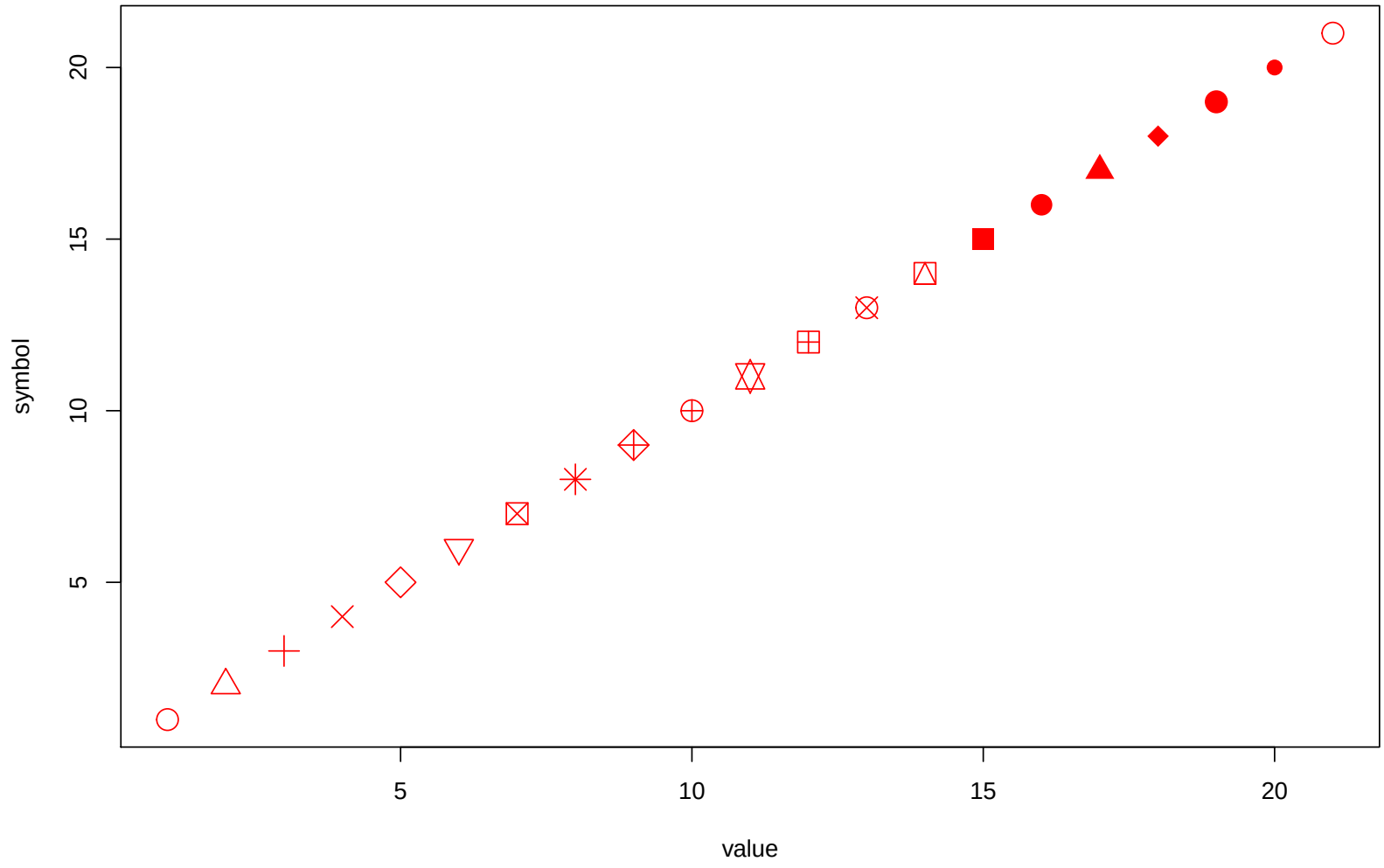



图 3:

# symbol parameter:point symbol

- ponit symbol parameter given `type='p'` in `plot` function



# symbol parameter:point symbol

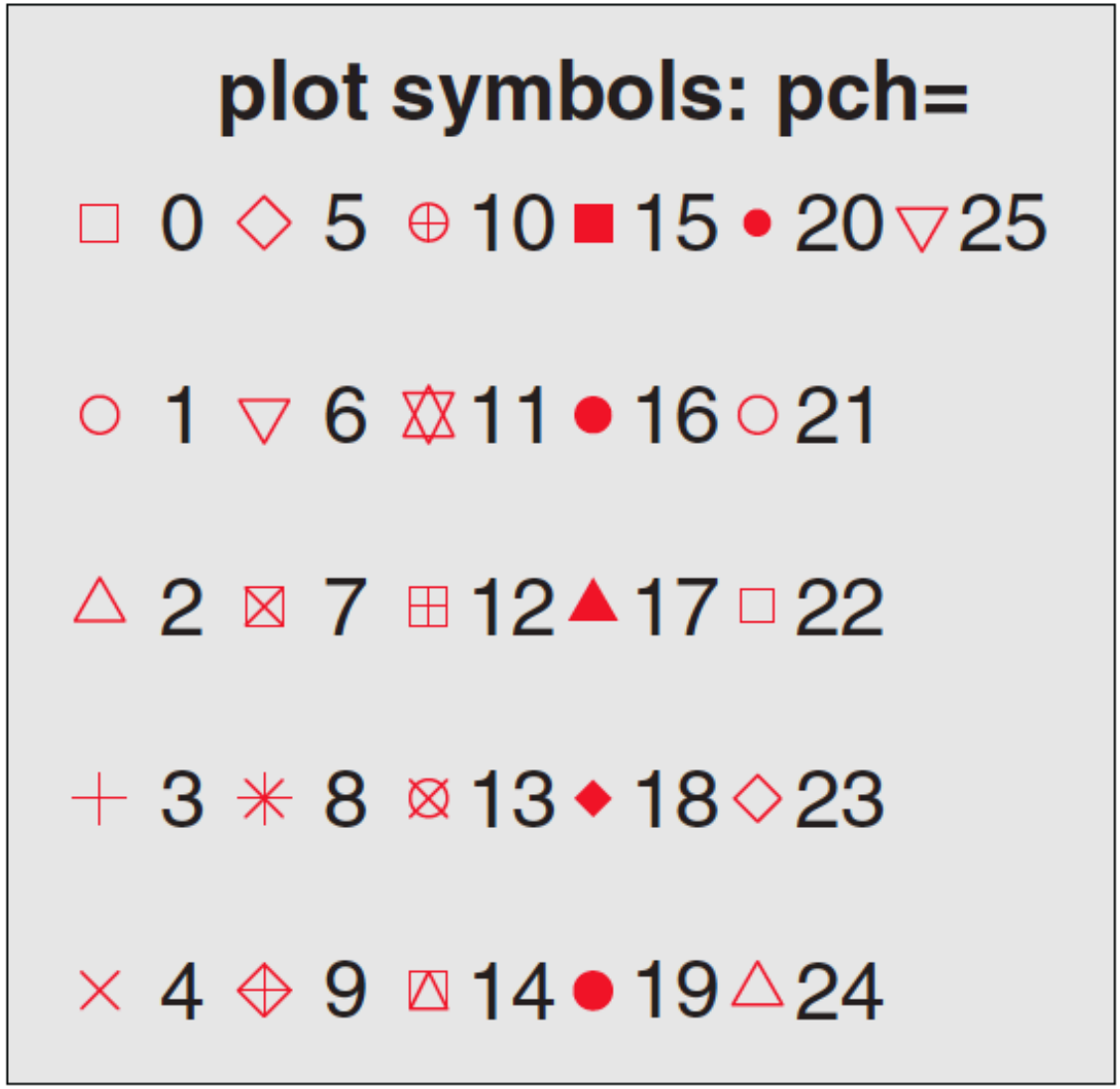
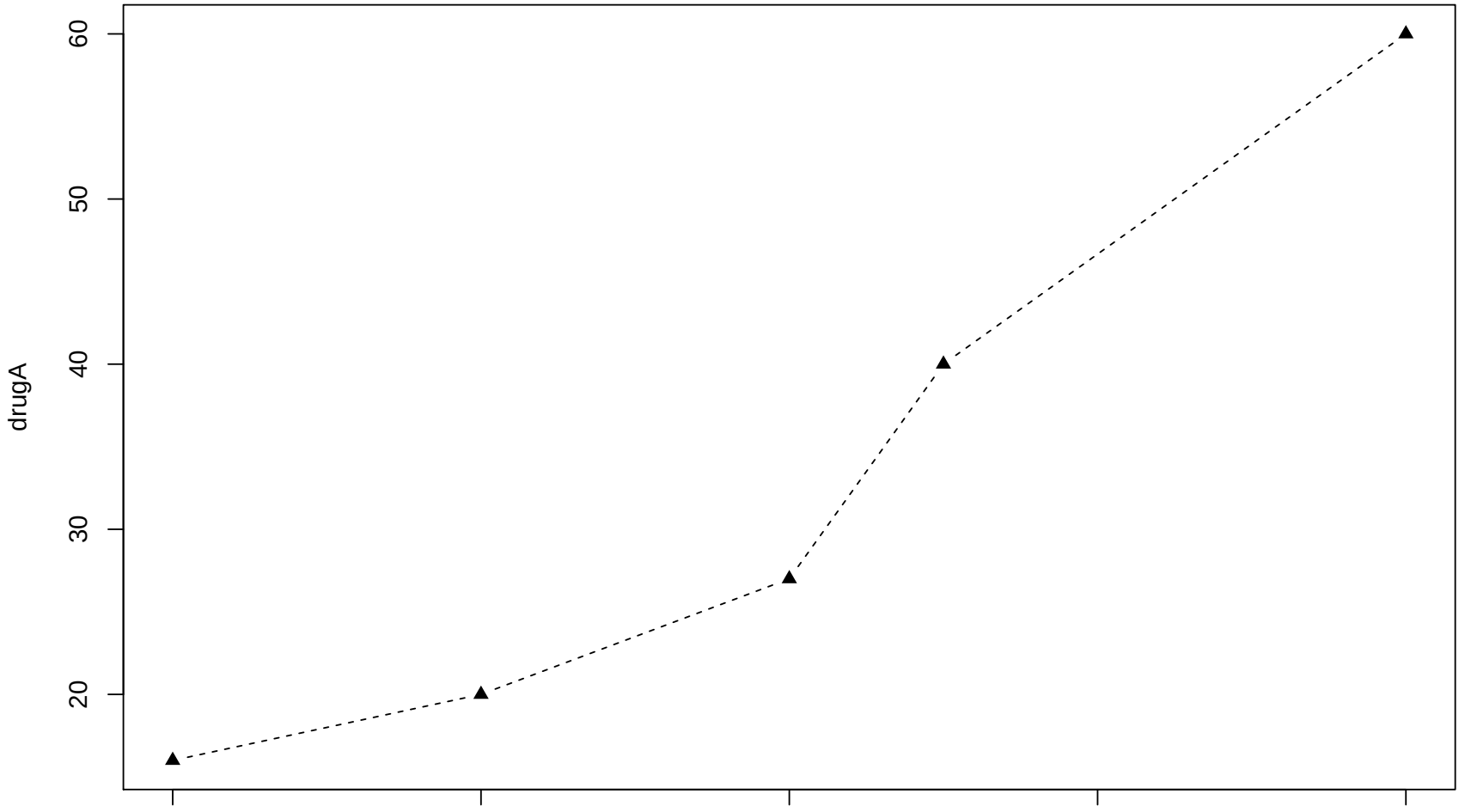



图 4:

## plot:line type and point type

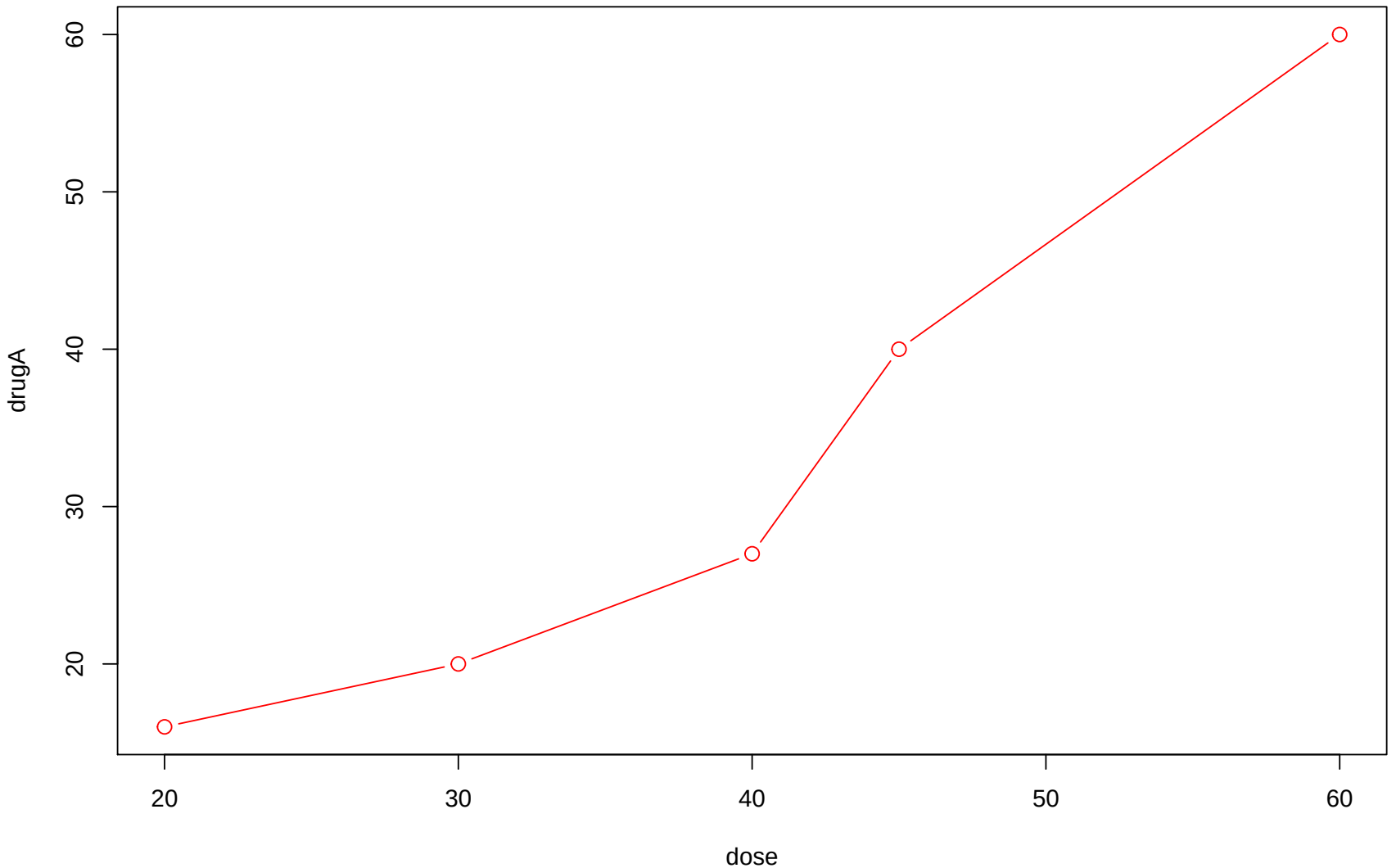
- line type (`type='l'`): `lty`
- point type (`type='p'`): `pch`

```
plot(dose, drugA, type = "b", lty = 2, pch = 17)
```



# plot:colour

```
opar <- par(no.readonly = TRUE)
plot(dose, drugA, type = "b", lty = 1,cex = 1.3,col='red')
```



# plot: colour

- `colors()`

```
length(colors())
```

```
## [1] 657
```

```
colors()[1:5]
```

```
## [1] "white"          "aliceblue"      "antiquewhite"  "antiquewh
## [5] "antiquewhite2"
```

# Palette:Aesthetics.

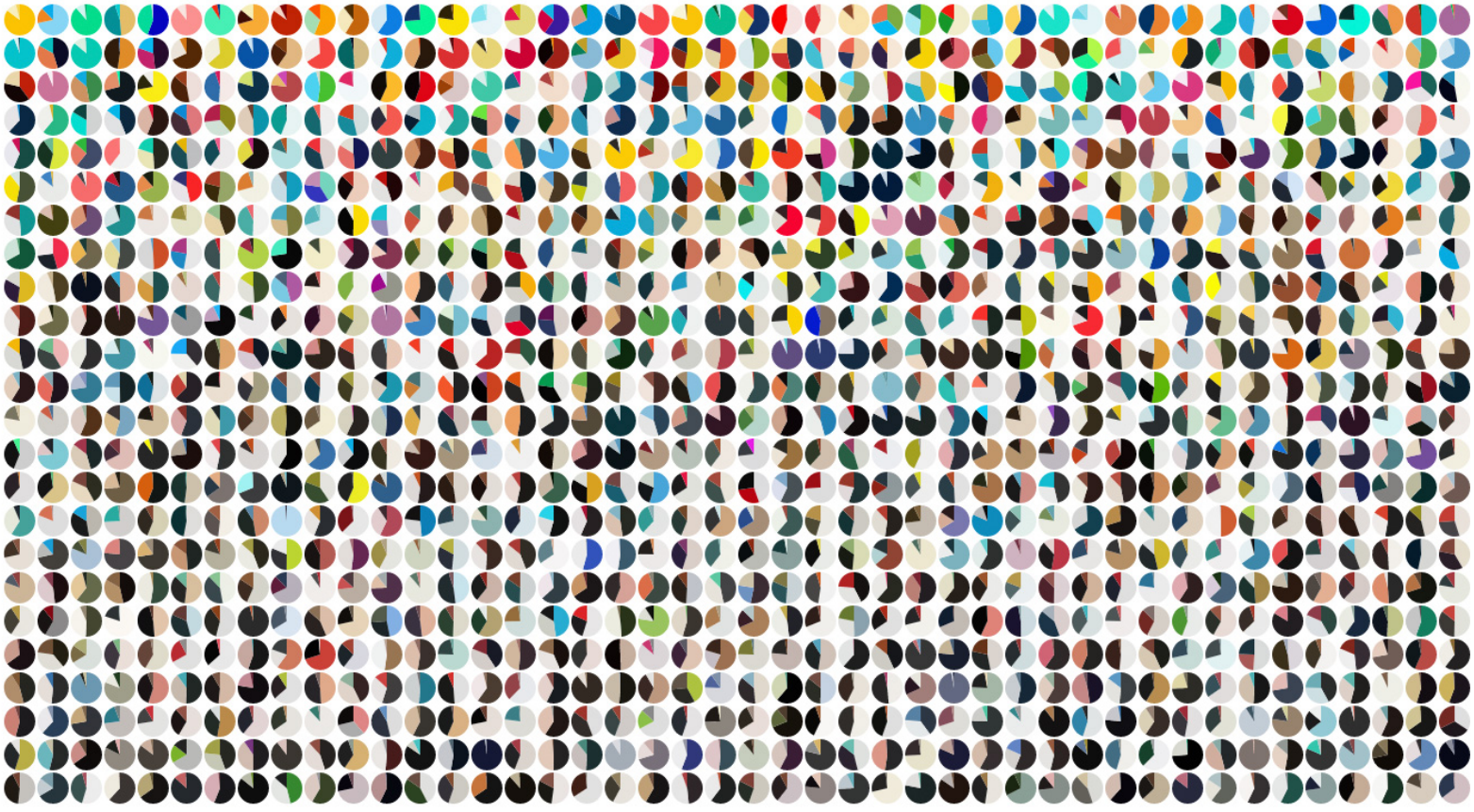


图 5:

# Palette:Aesthetics.

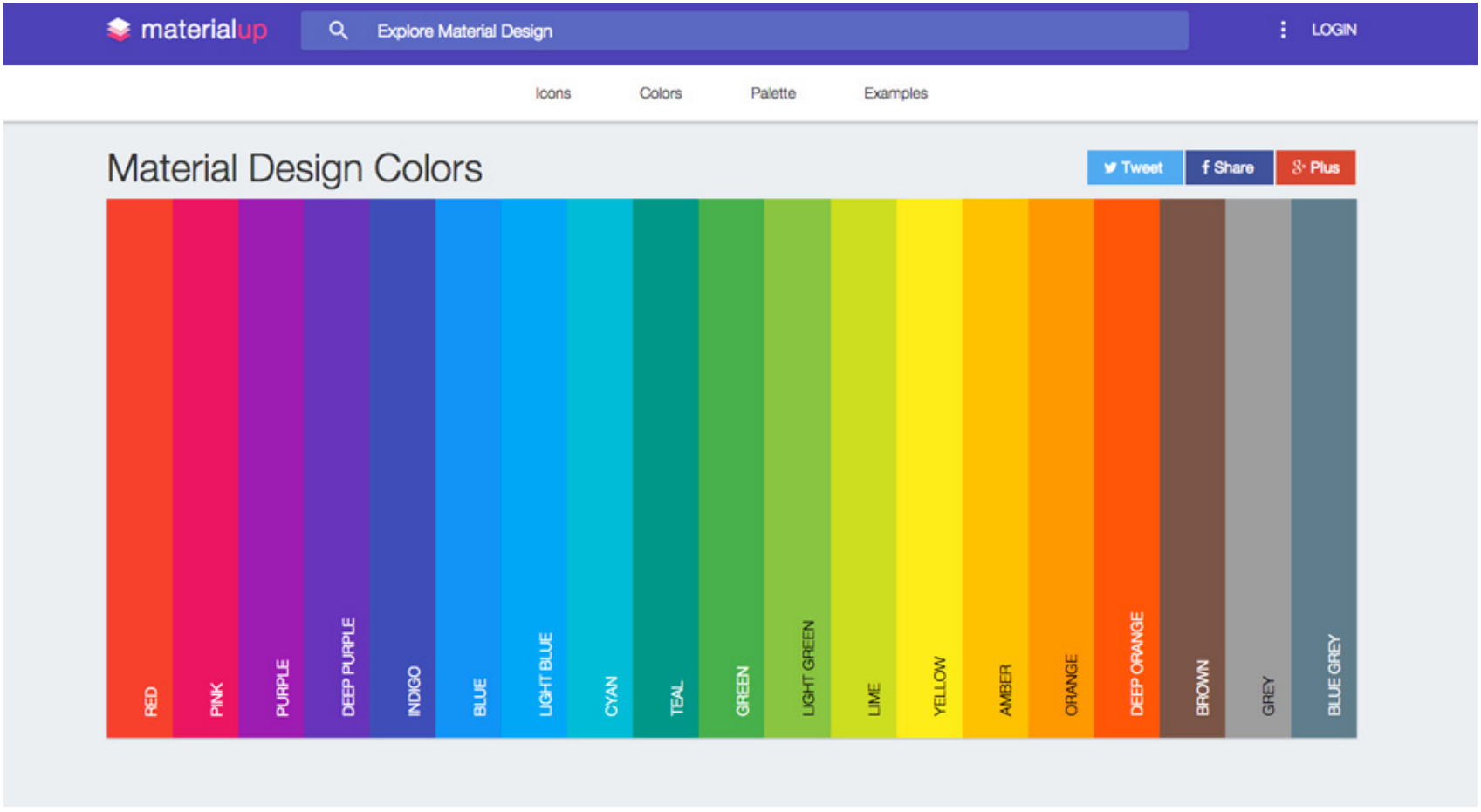



图 6:

# Palette:Aesthetics

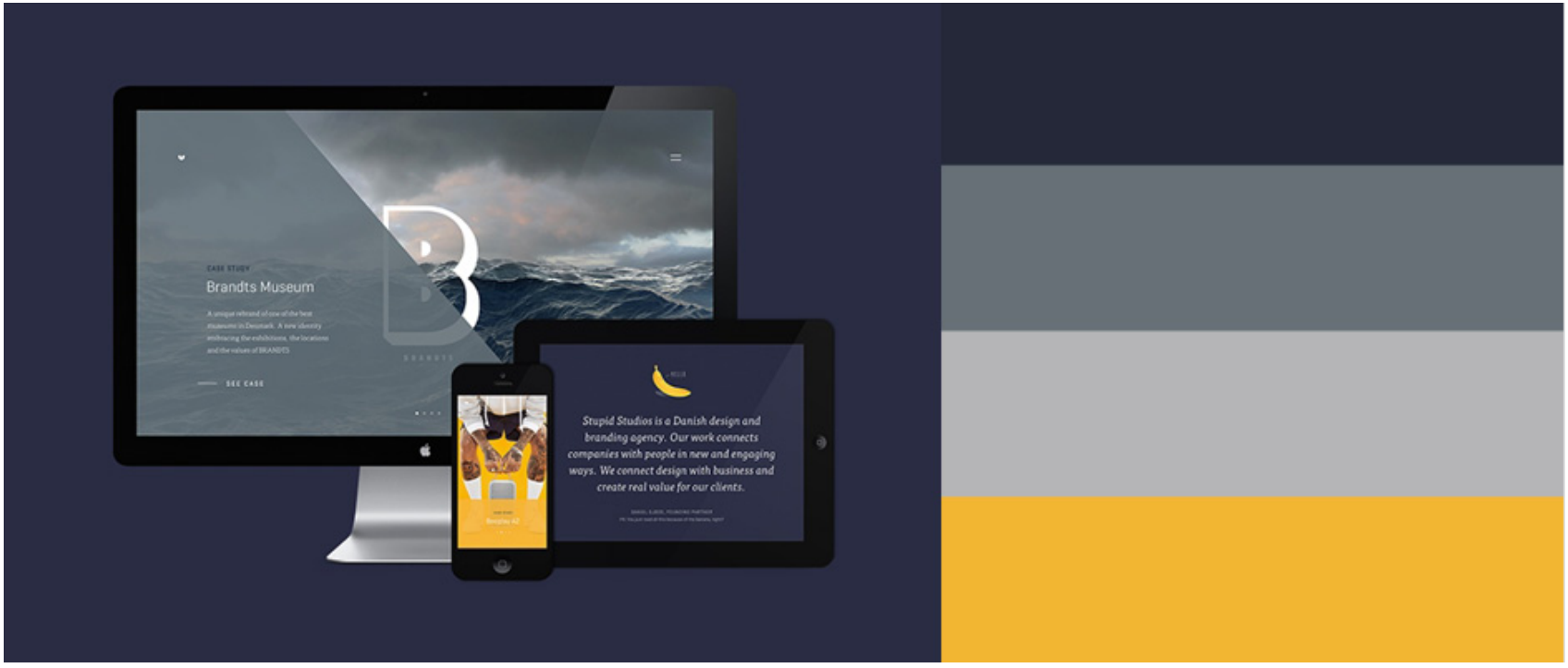
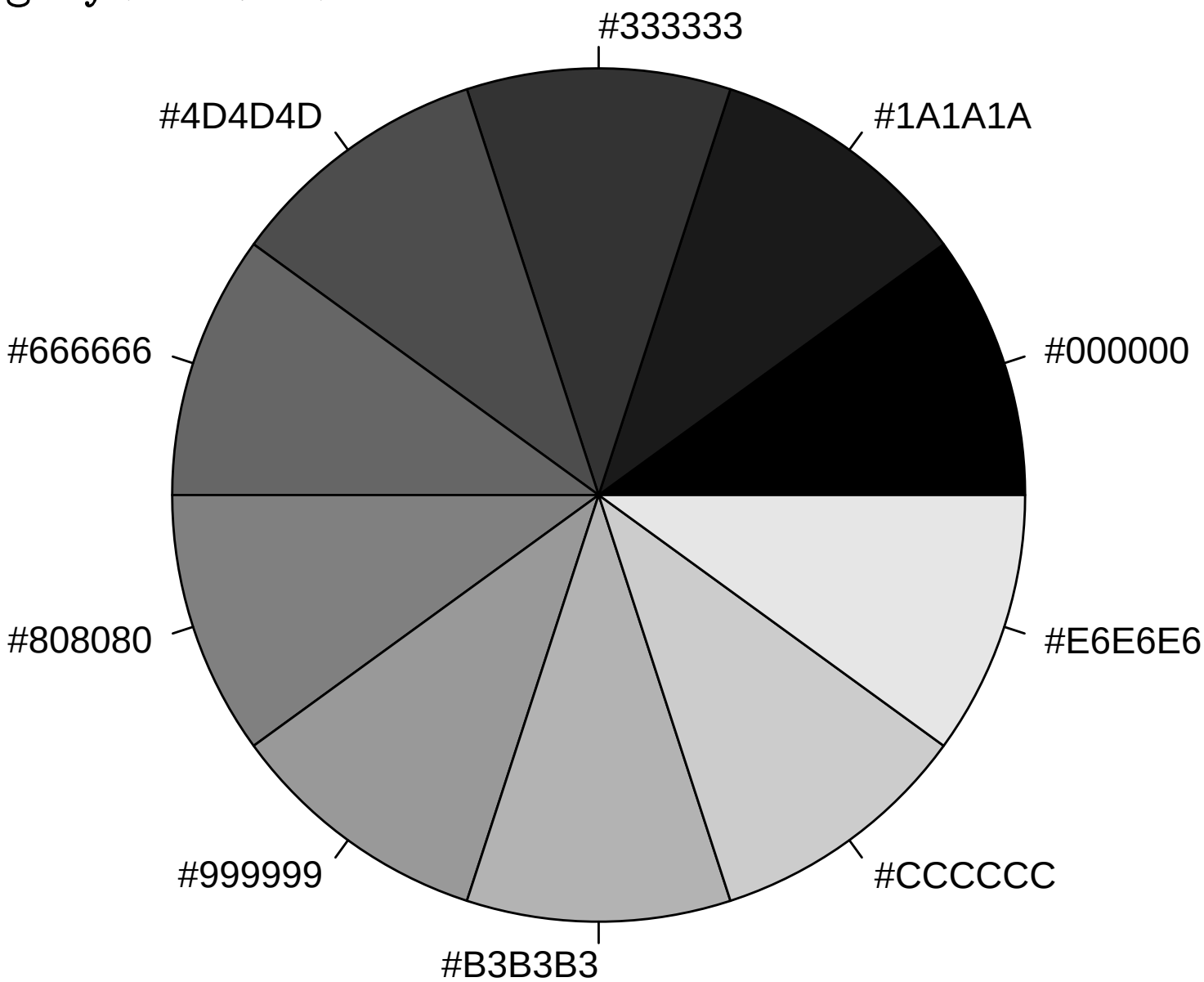



图 7:

- what can we do in R?

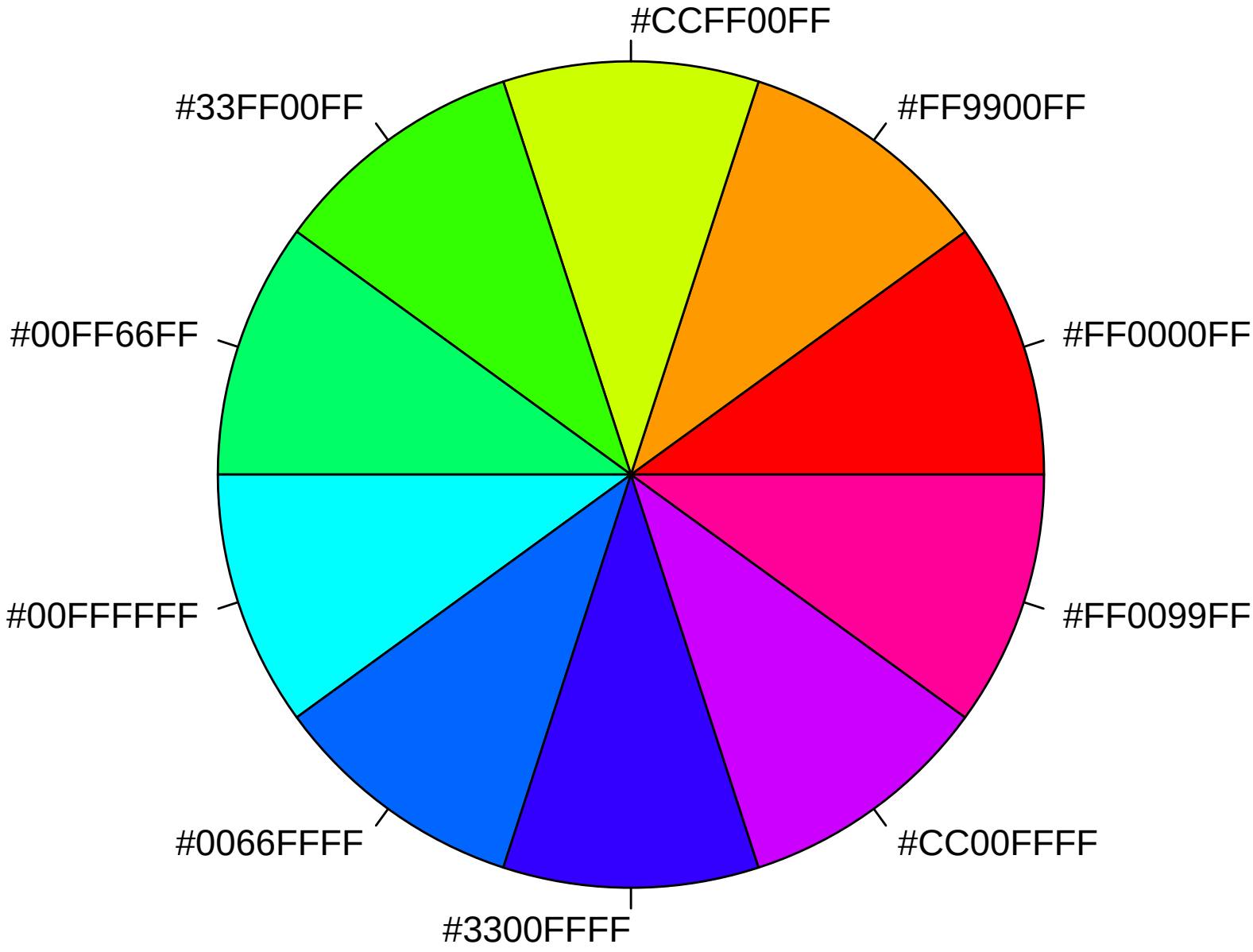
## grDevices:gray

- `gray(0:10/10)`



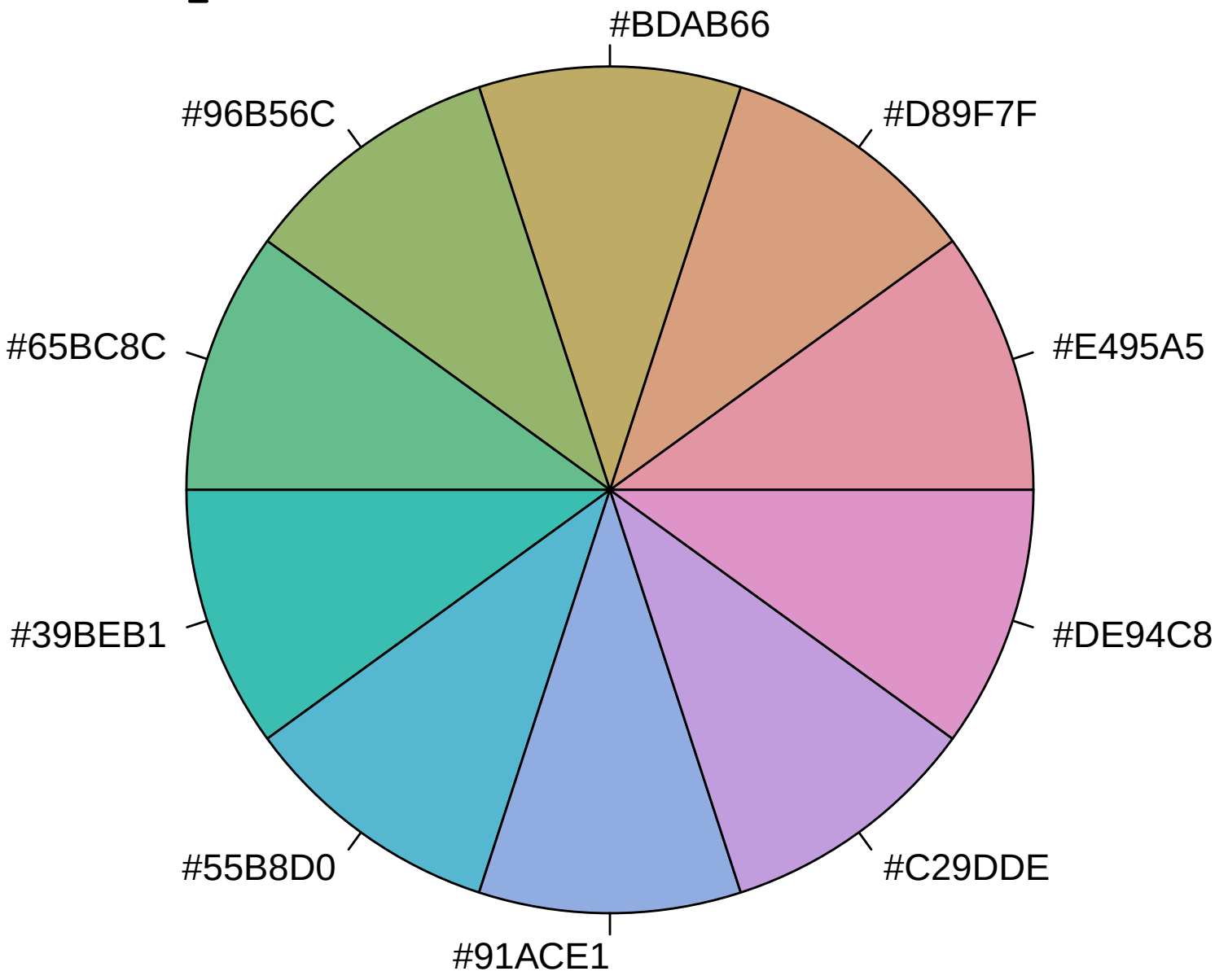
# grDevices: rainbow

- `rainbow(10)`

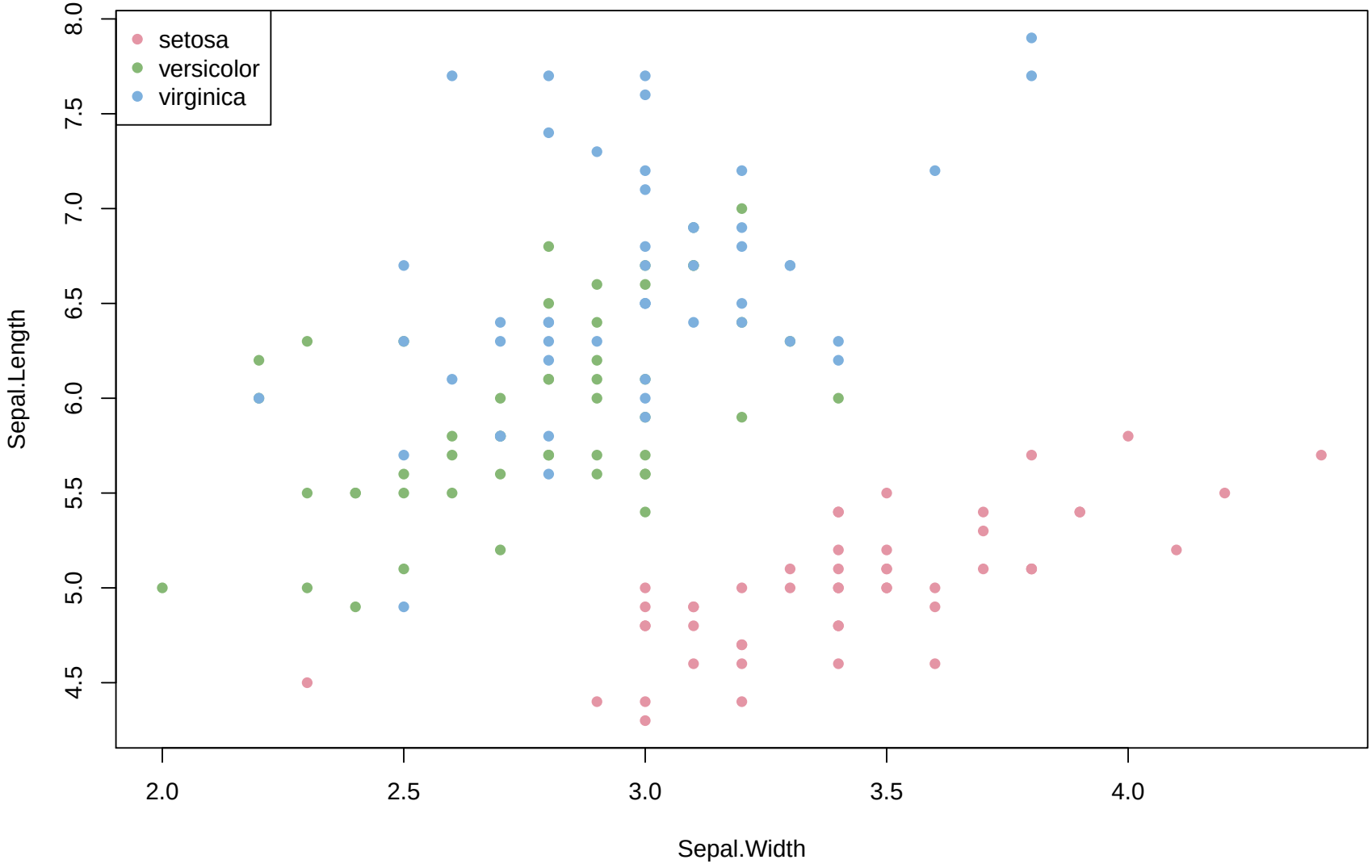


## colorspace:rainbow_hcl

- `rainbow_hcl(10)`



# colour example



## colour example

```
library(colorspace)
plot(Sepal.Length ~ Sepal.Width, col=
  rainbow_hcl(3)[c(Species)], data=iris, pch=16)
legend("topleft", pch=16, col=
  rainbow_hcl(3), legend=unique(iris$Species))
```

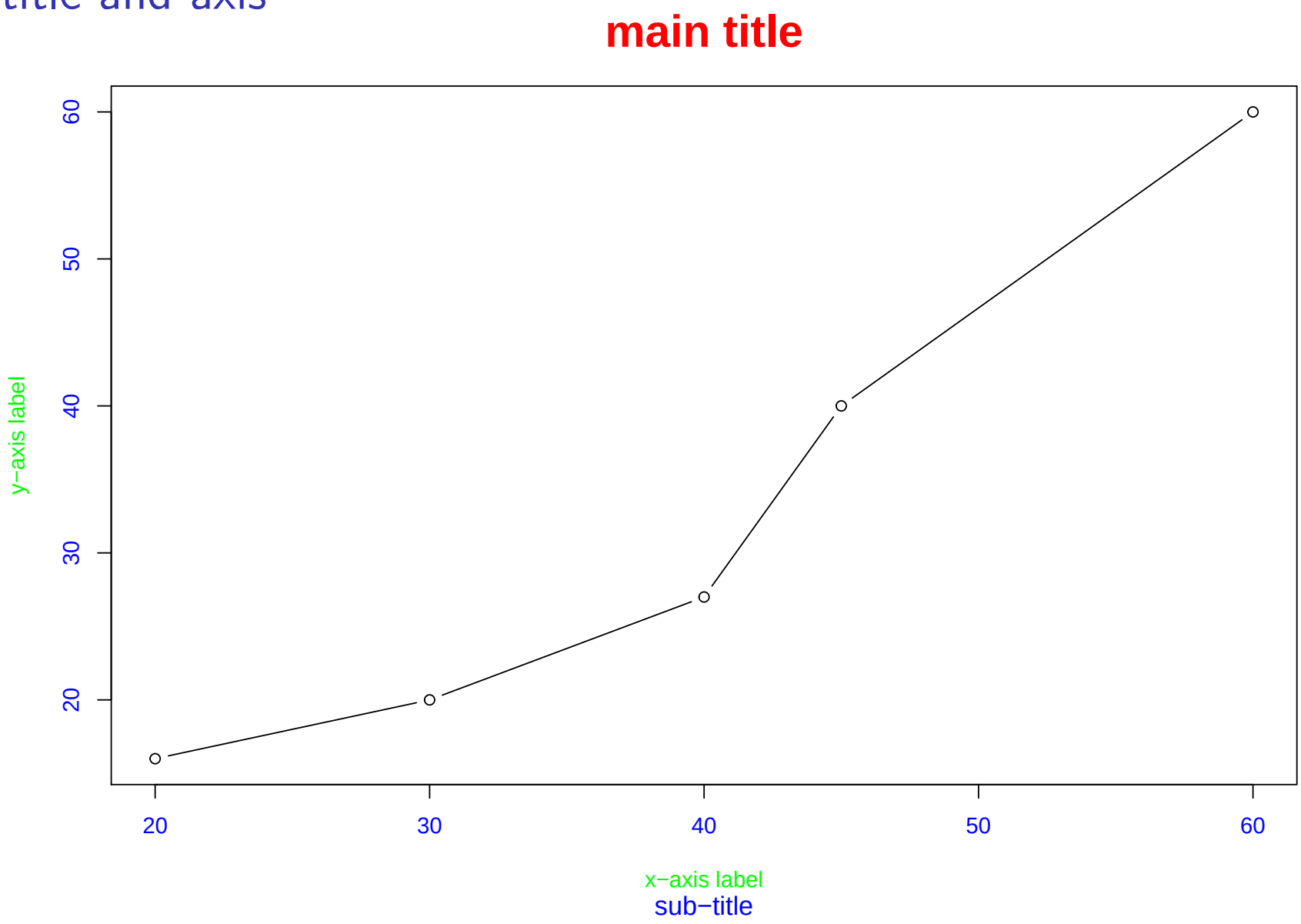
## content

- create and save graphs
- customize plot: symbols,lines,colour
- *annotate with text and titles*

## text and titles

- axis texts:axis
- axis labels:lab
- titles:main
- subtitles: sub

# title and axis



# Colour Parameter

- axis + title

| Parameter | Description |
|---|---|
| `col` | Default plotting color. Some functions (such as `lines` and `pie`) accept a vector of values that are recycled. For example, if `col=c("red", "blue")` and three lines are plotted, the first line will be red, the second blue, and the third red. |
| `col.axis` | Color for axis text. |
| `col.lab` | Color for axis labels. |
| `col.main` | Color for titles. |
| `col.sub` | Color for subtitles. |
| `fg` | Color for the plot's foreground. |
| `bg` | Color for the plot's background. |

# Size parameter

- axis + title

| Parameter | Description |
|---|---|
| `cex` | Number indicating the amount by which plotted text should be scaled relative to the default. 1 = default, 1.5 is 50% larger, 0.5 is 50% smaller, and so on. |
| `cex.axis` | Magnification of axis text relative to `cex`. |
| `cex.lab` | Magnification of axis labels relative to `cex`. |
| `cex.main` | Magnification of titles relative to `cex`. |
| `cex.sub` | Magnification of subtitles relative to `cex`. |

# Font parameter

- axis + title

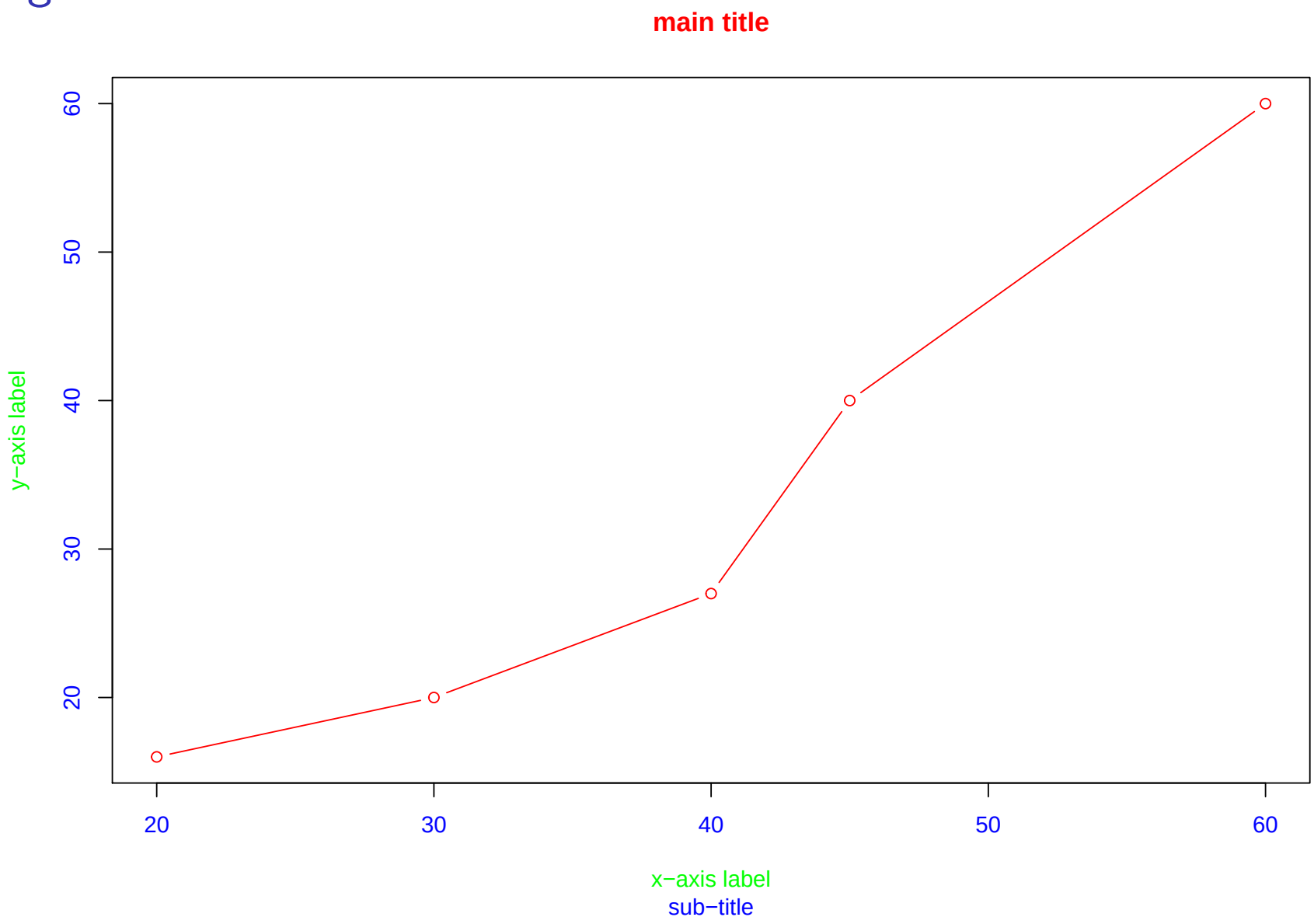
| Parameter | Description |
|---|---|
| `font` | Integer specifying the font to use for plotted text. 1 = plain, 2 = bold, 3 = italic, 4 = bold italic, and 5=symbol (in Adobe symbol encoding). |
| `font.axis` | Font for axis text. |
| `font.lab` | Font for axis labels. |
| `font.main` | Font for titles. |
| `font.sub` | Font for subtitles. |
| `ps` | Font point size (roughly 1/72 inch). The text size = `ps*cex`. |
| `family` | Font family for drawing text. Standard values are `serif`, `sans`, and `mono`. |

## parameters for colour,size and font

- common parameter

| | main | sub | lab | axis |
|---|---|---|---|---|
| col | col.main | col.sub | col.lab | axis.lab |
| cex | cex.main | cex.sub | cex.lab | cex.axis |
| font | font.main | font.sub | font.lab | font.axis |

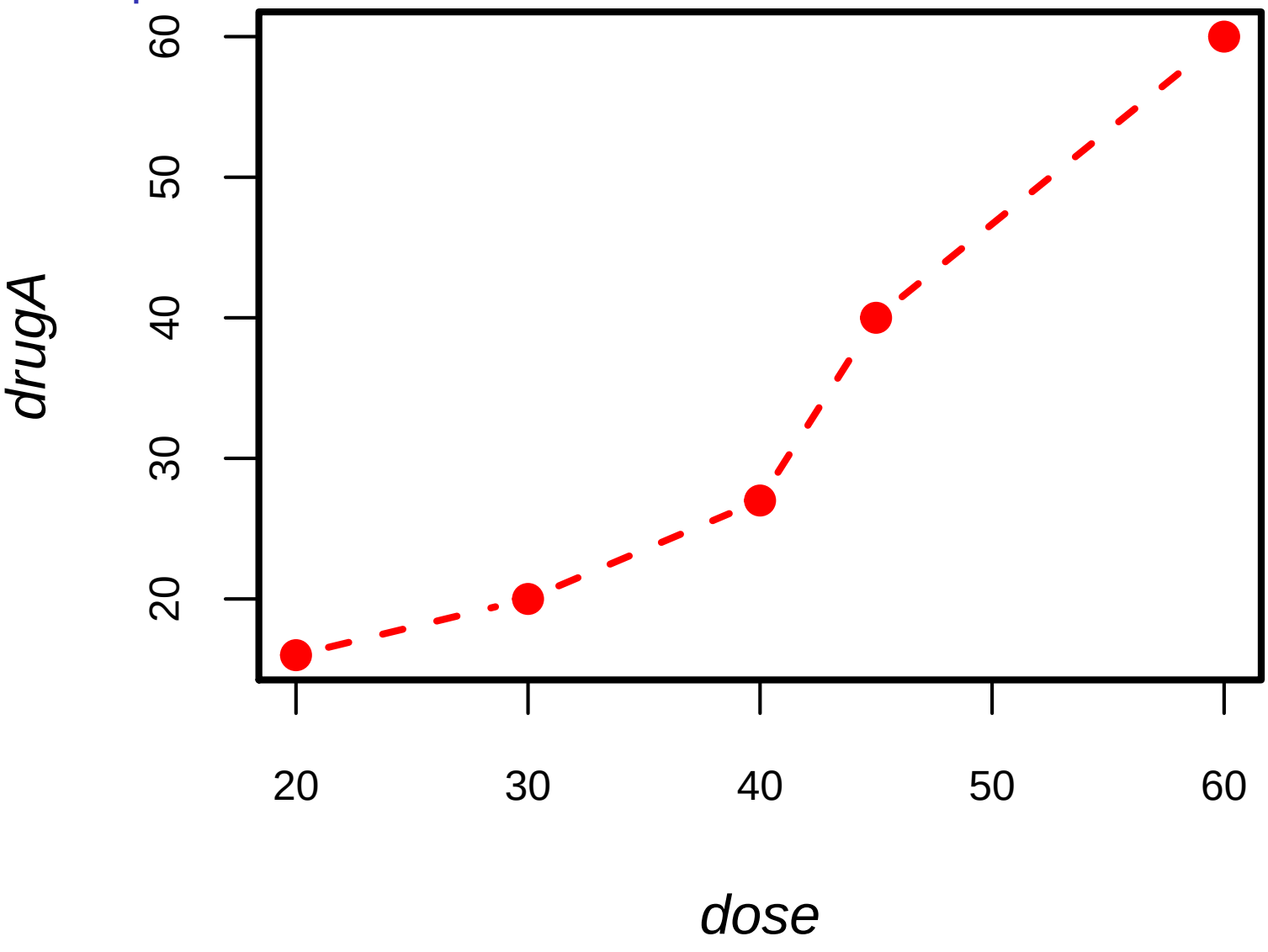
# figure demo


main title
y−axis label
20
30
40
50
60
20
30
40
50
60
x−axis label
sub−title

## figure demo

```
dose <- c(20, 30, 40, 45, 60)
drugA <- c(16, 20, 27, 40, 60)
par(ann=F) ##inhibit x y axis labels but not text.
plot(dose, drugA, type = "b",
     col = "red",col.axis='blue')
# plot(dose, drugA, type = "b",
#col = "red",col.axis='blue',xaxt='n',yaxt='n')
##inhibit axis tick and text
title(main='main title',col.main='red',cex.main=2,
sub='sub-title',col.sub='blue',
xlab='x-axis label',
ylab='y-axis label',col.lab='green',cex.lab=1)
```

## Example

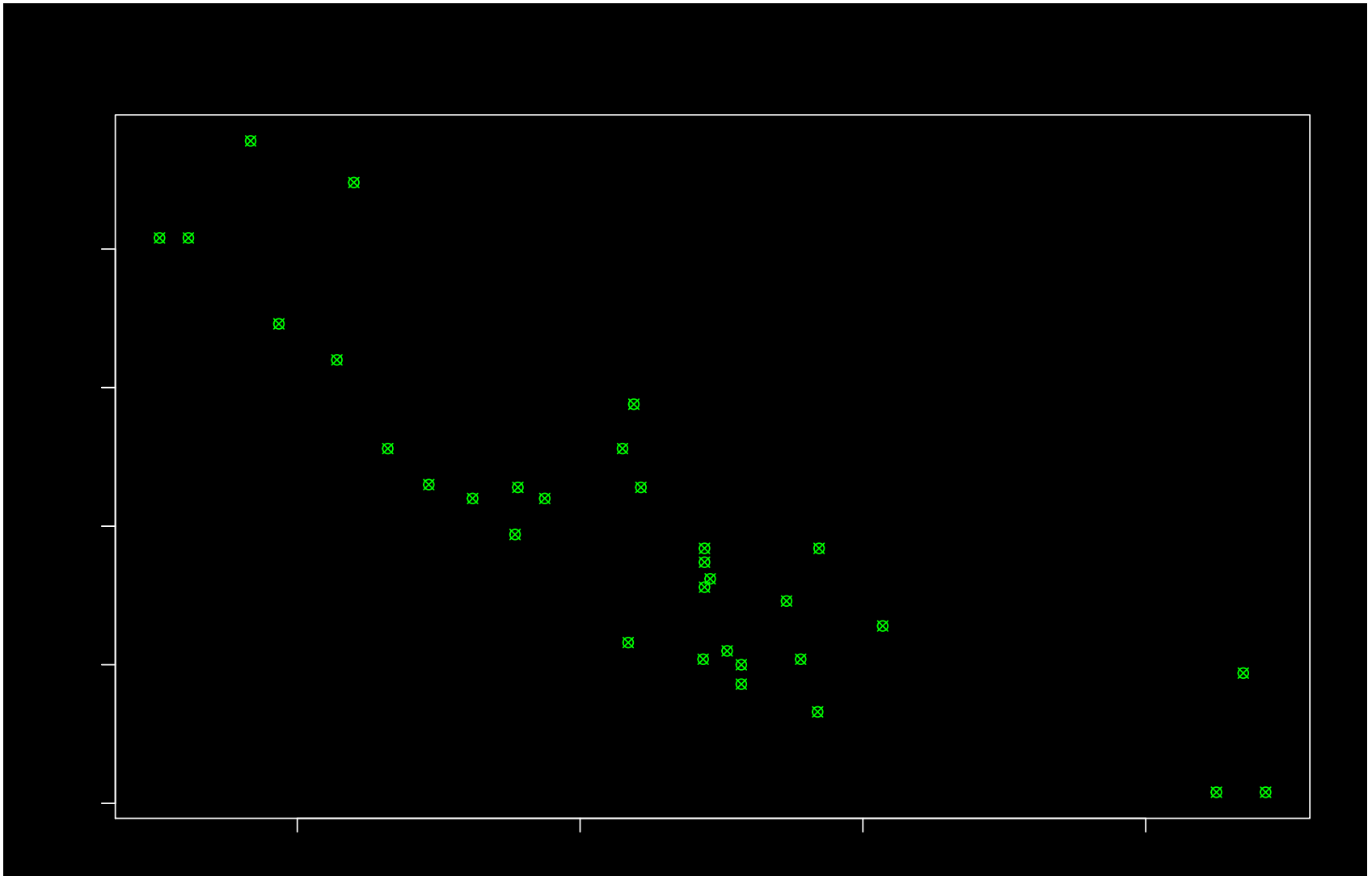


## Example

```
dose <- c(20, 30, 40, 45, 60)
drugA <- c(16, 20, 27, 40, 60)
par(font.lab=3, cex.lab=1,
    font.main=4, cex.main=2)
opar <- par(no.readonly = TRUE)
par(pin = c(3, 2))
##The current plot dimensions,
par(lwd = 2, cex = 1)
##magnified index of plotted text and symbols
par(cex.axis = 0.75, font.axis = 1)
plot(dose, drugA, type = "b", pch = 19,
     lty = 2, col = "red") ##pch plot symbols
```

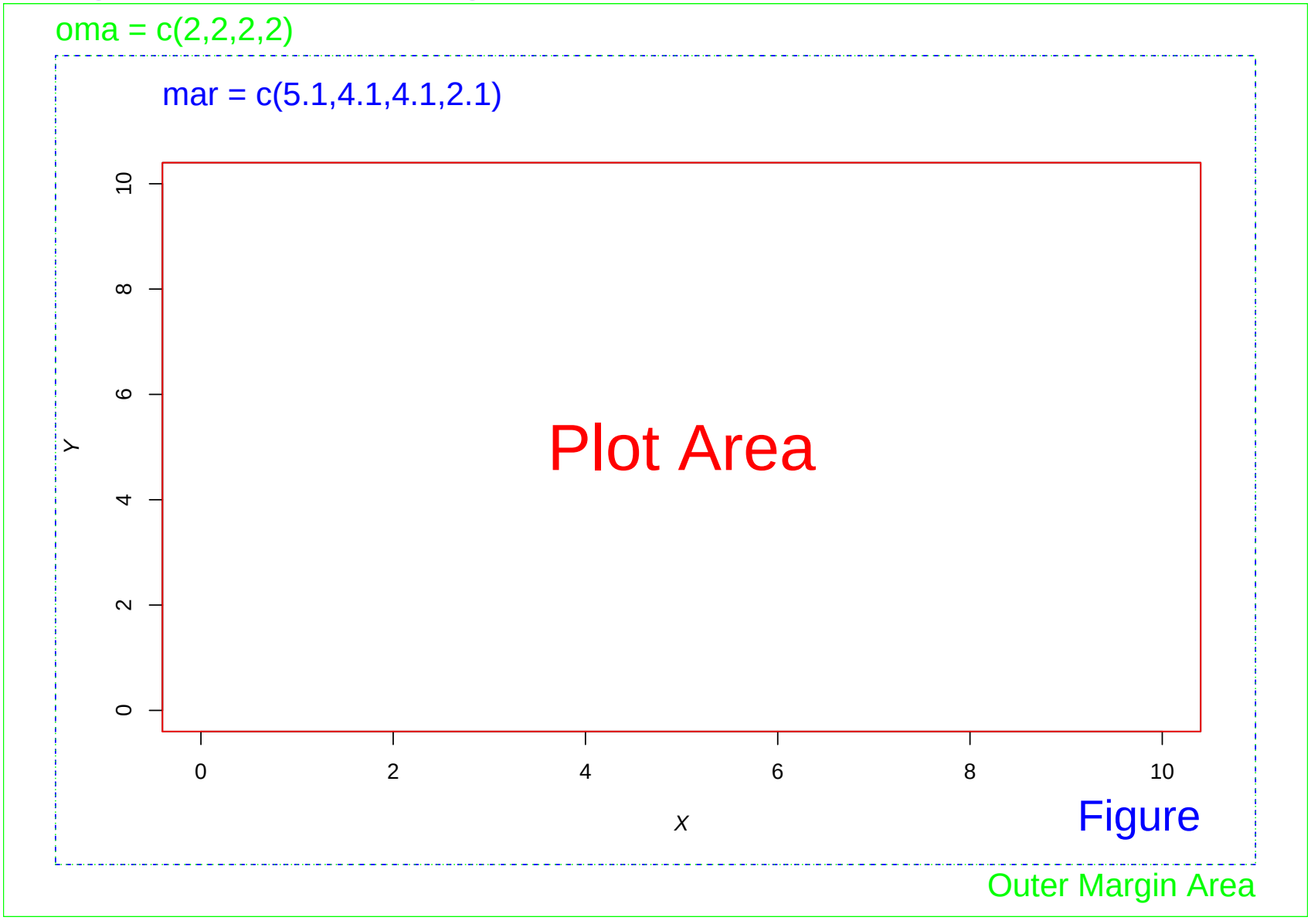
# Foreground and background

```
par(bg='black',fg='white')
with(mtcars,plot(wt,mpg,col='green',pch=13))
```



PRACTICE!

# Margin and outer margin



# Margin and outer margin

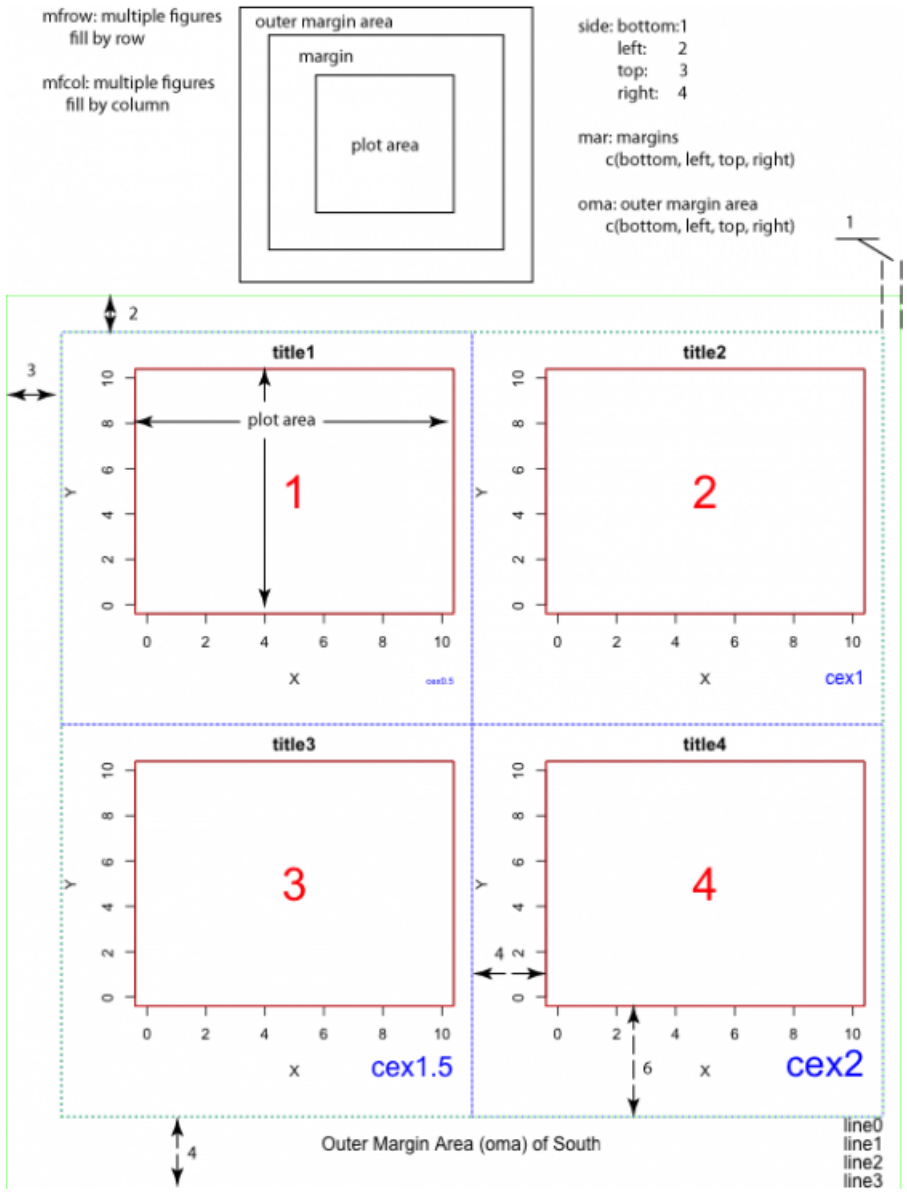



图 8:

```r
par(opar)
par('pin')
```

```
## [1] 8.76 5.16
```

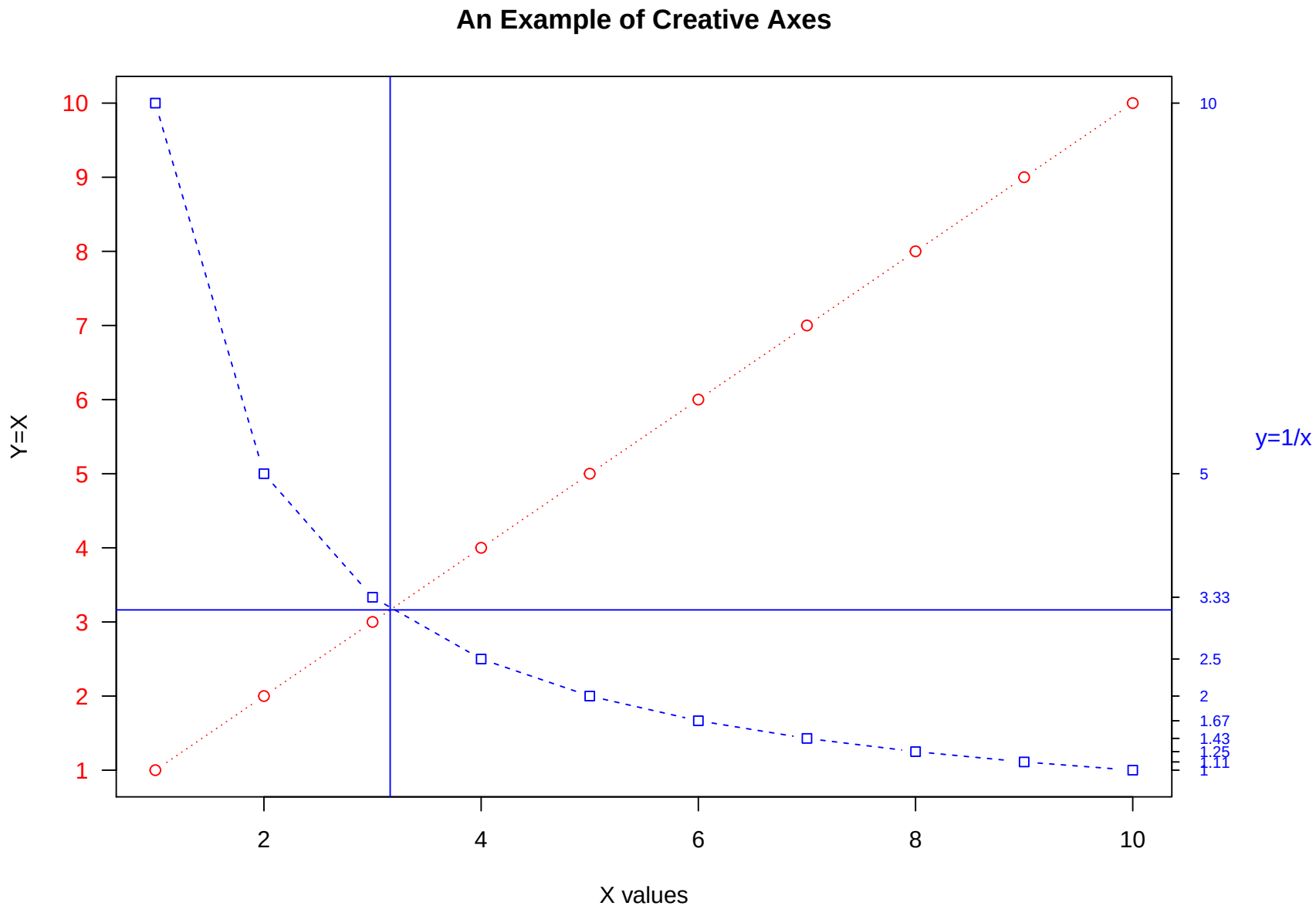
```r
par('mar')
```

```
## [1] 5.1 4.1 4.1 2.1
```

```r
par('oma')
```

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

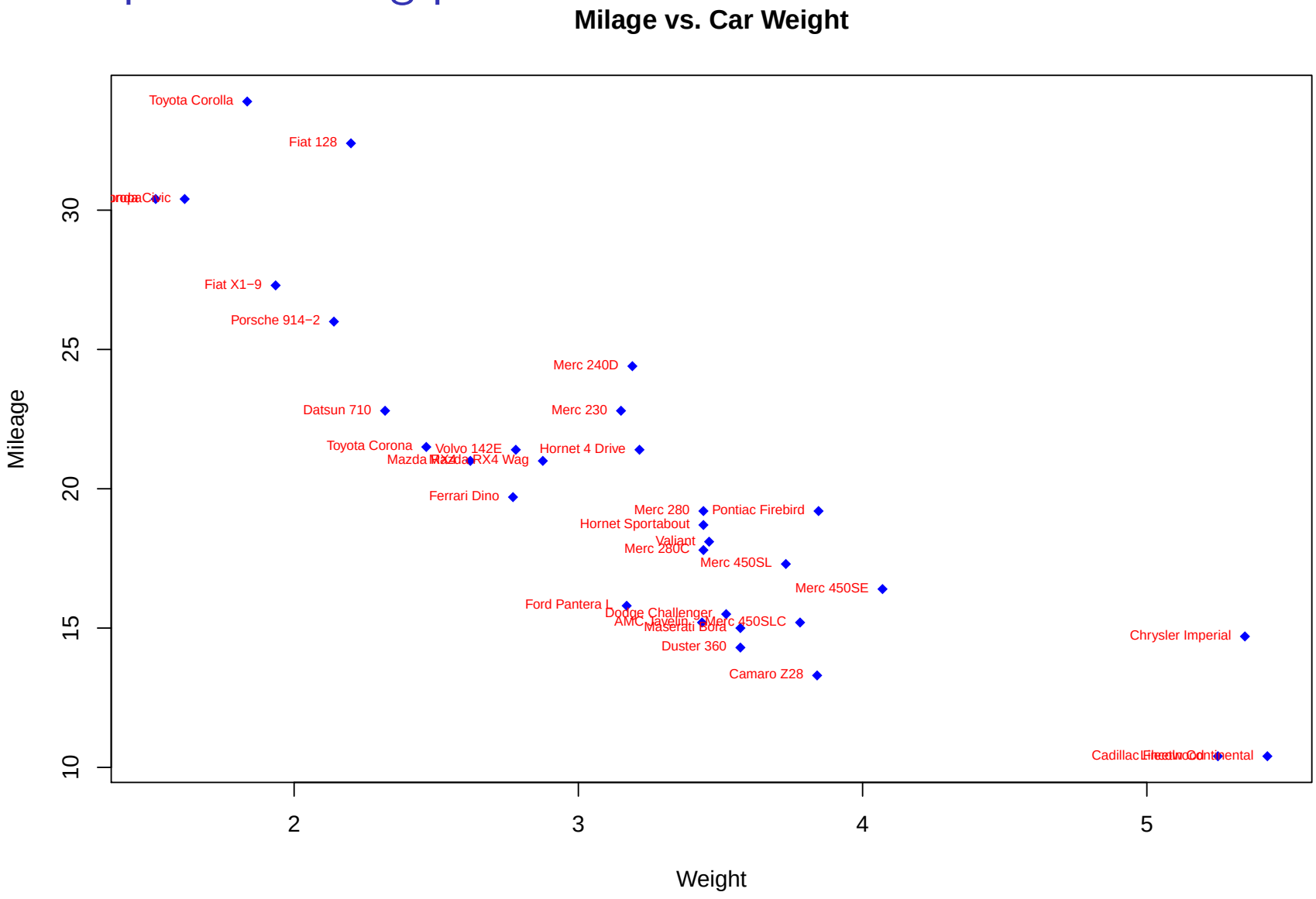
# An Example of two y Axes



# An Example of two y Axes

```
x <- c(1:10);y <- x;z <- 10/x
par(mar = c(5, 4, 4, 8) + 0.1)
plot(x, y, type = "b", lty = 3, pch = 21, col = "red",
     yaxt = "n",ann = FALSE)
lines(x, z, type = "b", pch = 22, col = "blue", lty = 2)
axis(side=2, at = x, labels = x, col.axis = "red",
     las = 1)
axis(side=4, at = z, labels = round(z, digits = 2),
     col.axis = "blue",las = 2, cex.axis = 0.7,tck = -0.01)
mtext("y=1/x", side = 4, line = 3, cex.lab = 1, las = 2,
col = "blue")
title("An Example of Creative Axes", xlab = "X values",
ylab = "Y=X")
abline(h=sqrt(10),lty=1,col='blue')
abline(v=sqrt(10),lty=1,col='blue')
```

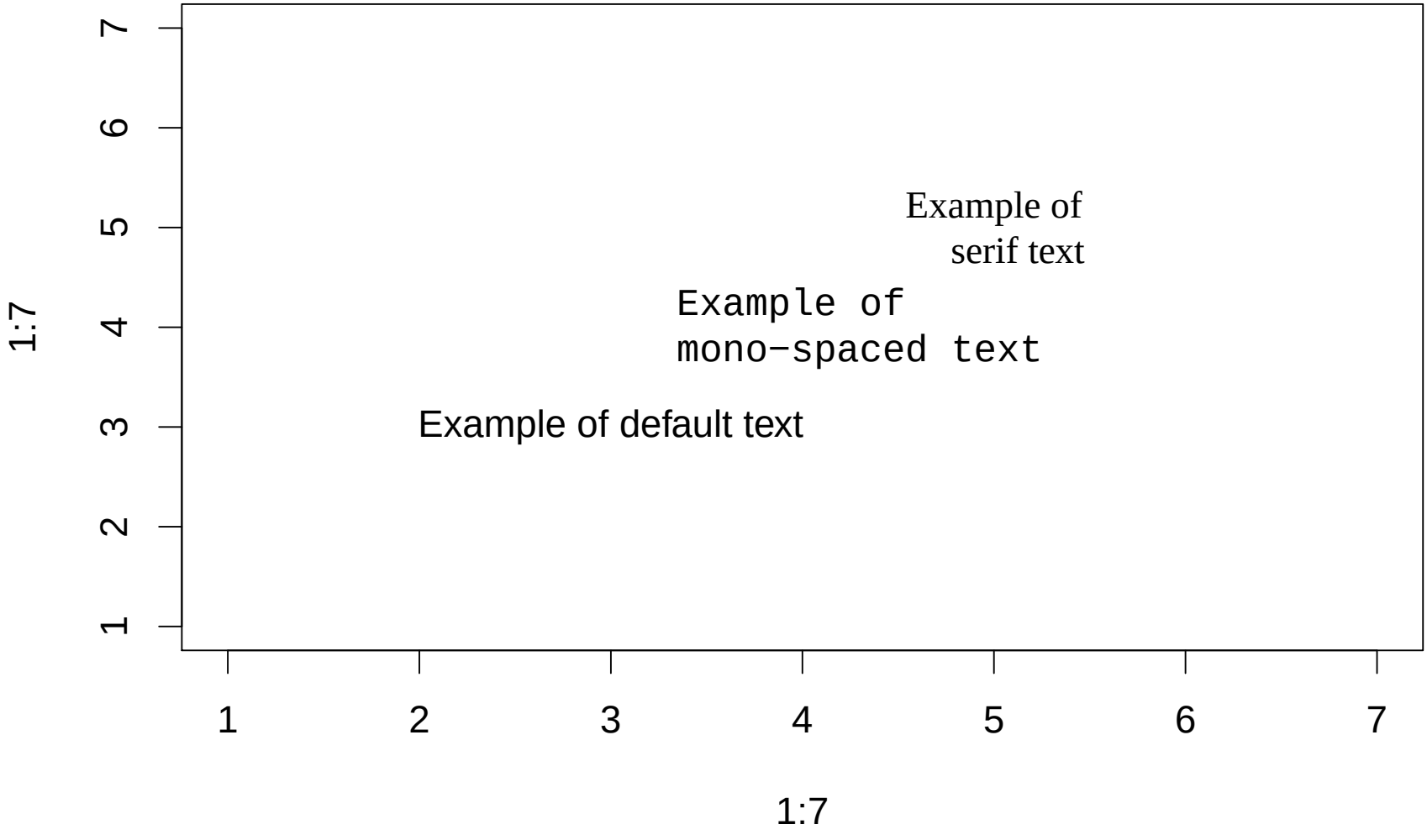
# Example of labeling points



## Example of labeling points

```
with(mtcars,plot(wt, mpg,main = "Milage vs. Car Weight",
                 xlab = "Weight",
ylab = "Mileage", pch = 18, col = "blue"))
text(mtcars$wt, mtcars$mpg, row.names(mtcars),
     cex = 0.6, pos = 2,
col = "red")
```

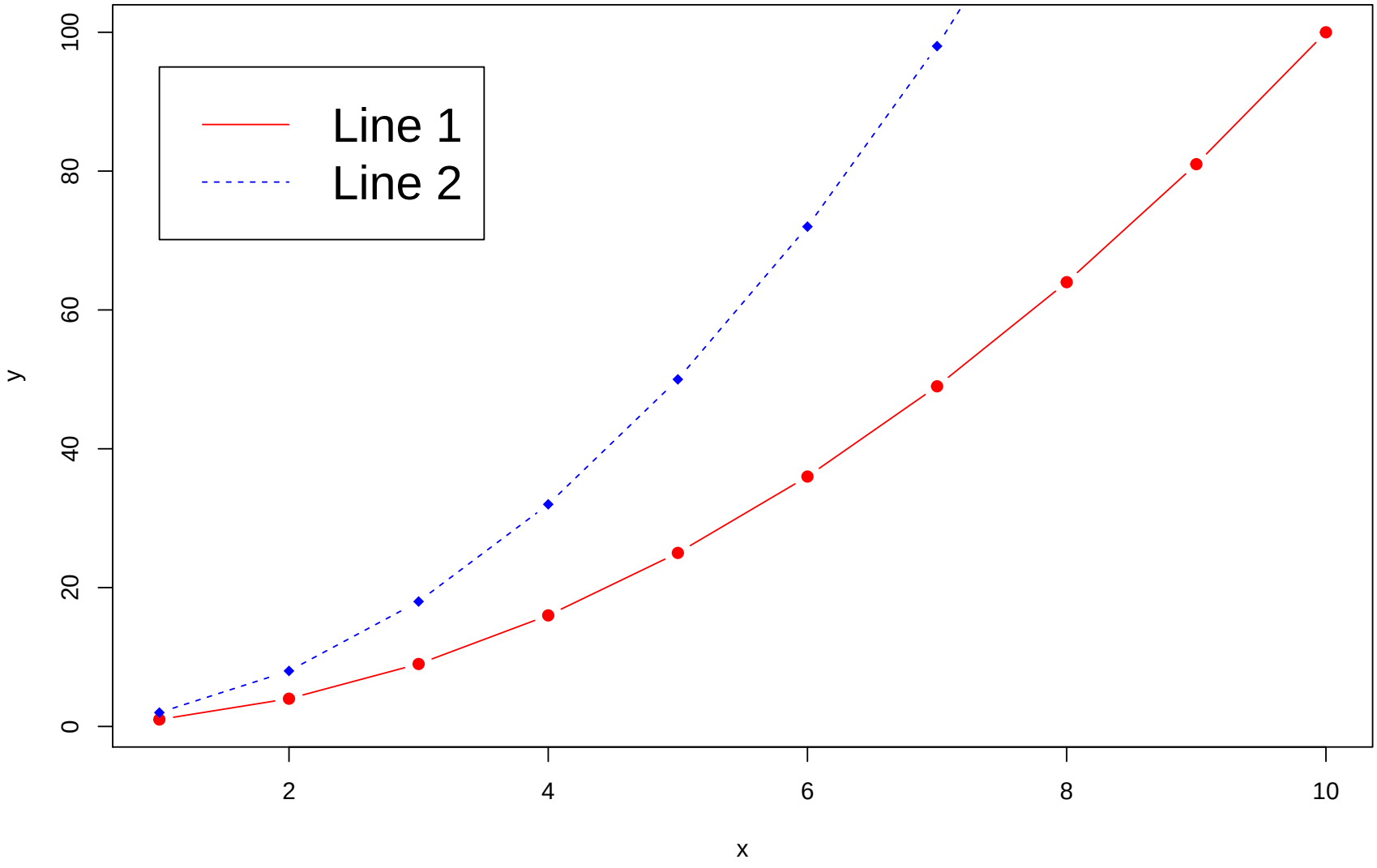
## View font families



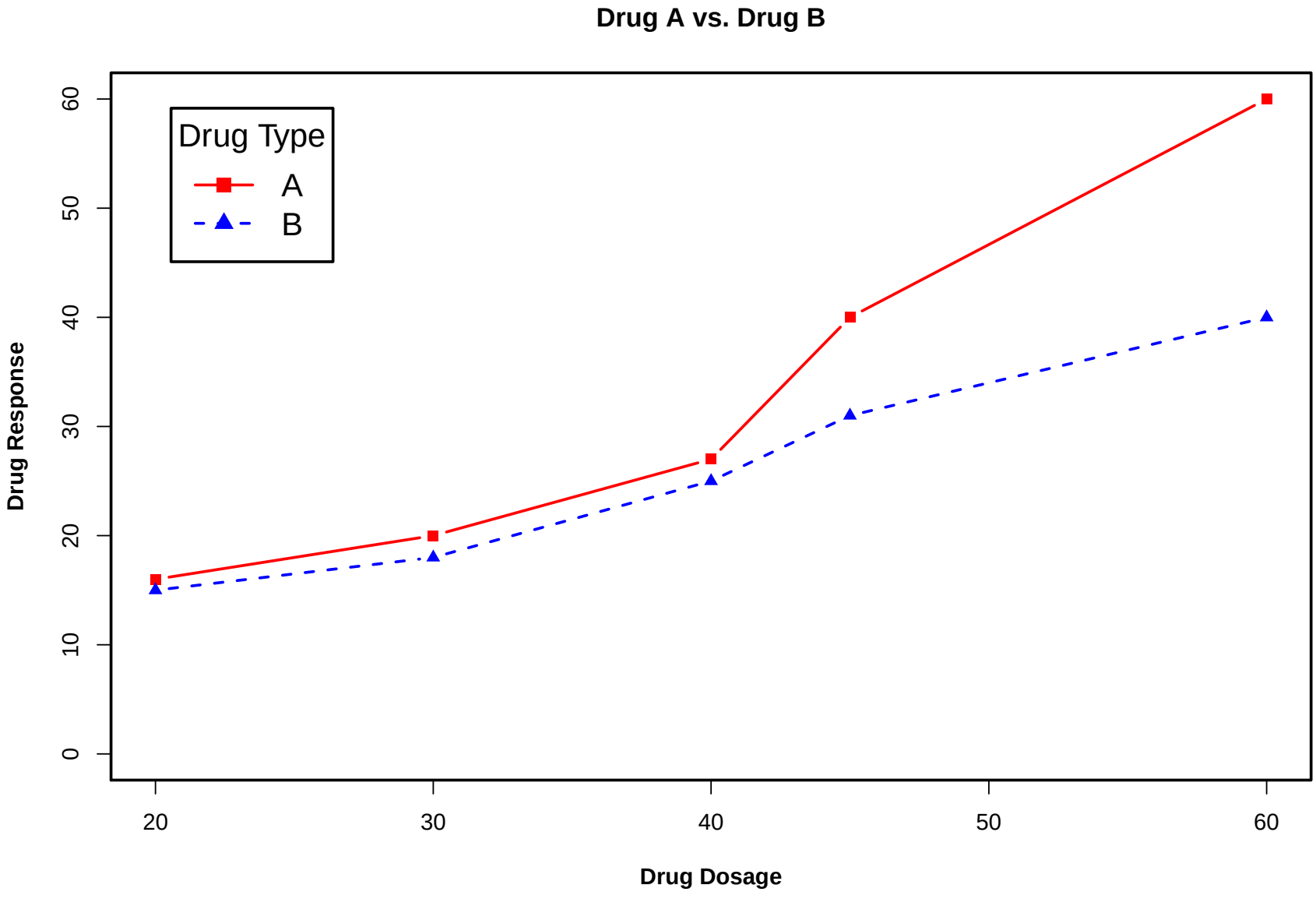
## View font families

```
opar <- par(no.readonly = TRUE)
par(cex = 1.5)
plot(1:7, 1:7, type = "n")
text(3, 3, "Example of default text")
text(4, 4, family = "mono", "Example of
     mono-spaced text")
text(5, 5, family = "serif", "Example of
     serif text")
par(opar)
```

# Legend



# Legend


Drug A vs. Drug B
Drug Type
A
B
Drug Response
0
10
20
30
40
50
60
20
30
40
50
60
Drug Dosage

# Legend

```
dose <- c(20, 30, 40, 45, 60)
drugA <- c(16, 20, 27, 40, 60)
drugB <- c(15, 18, 25, 31, 40)
opar <- par(no.readonly = TRUE)
par(lwd = 2, cex = 1, font.lab = 2)
plot(dose, drugA, type = "b", pch = 15, lty = 1,
     col = "red",
ylim = c(0, 60), main = "Drug A vs. Drug B",
xlab = "Drug Dosage",
ylab = "Drug Response")
lines(dose, drugB, type = "b", pch = 17, lty = 2,
col = "blue")
# abline(h = c(30), lwd = 3, lty = 2, col = "green")
legend("topleft", inset = 0.05, title = "Drug Type",
c("A", "B"), lty = c(1, 2), pch = c(15, 17),
col = c("red","blue"),cex=1)
par(opar)
```

# Legend position

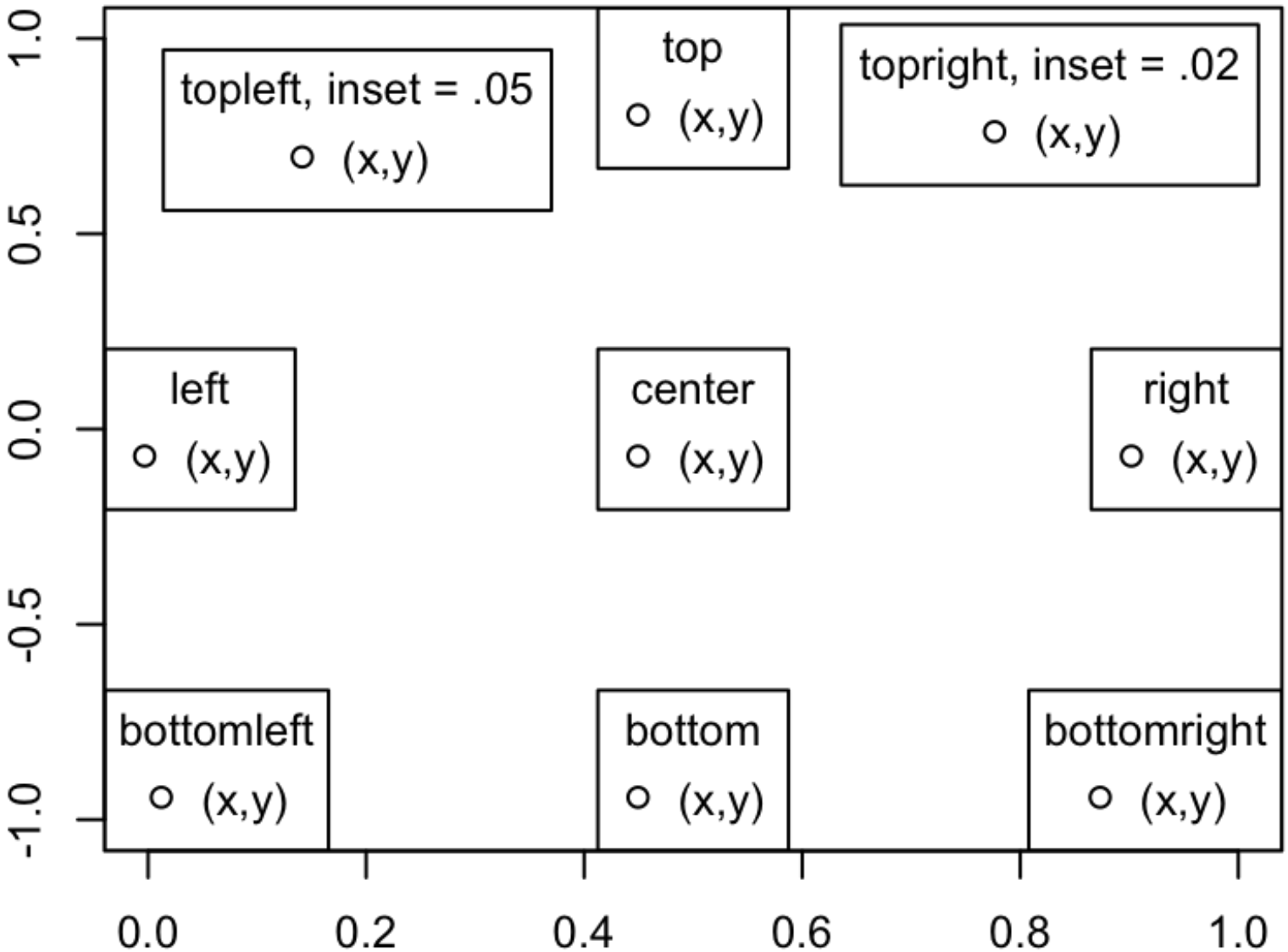


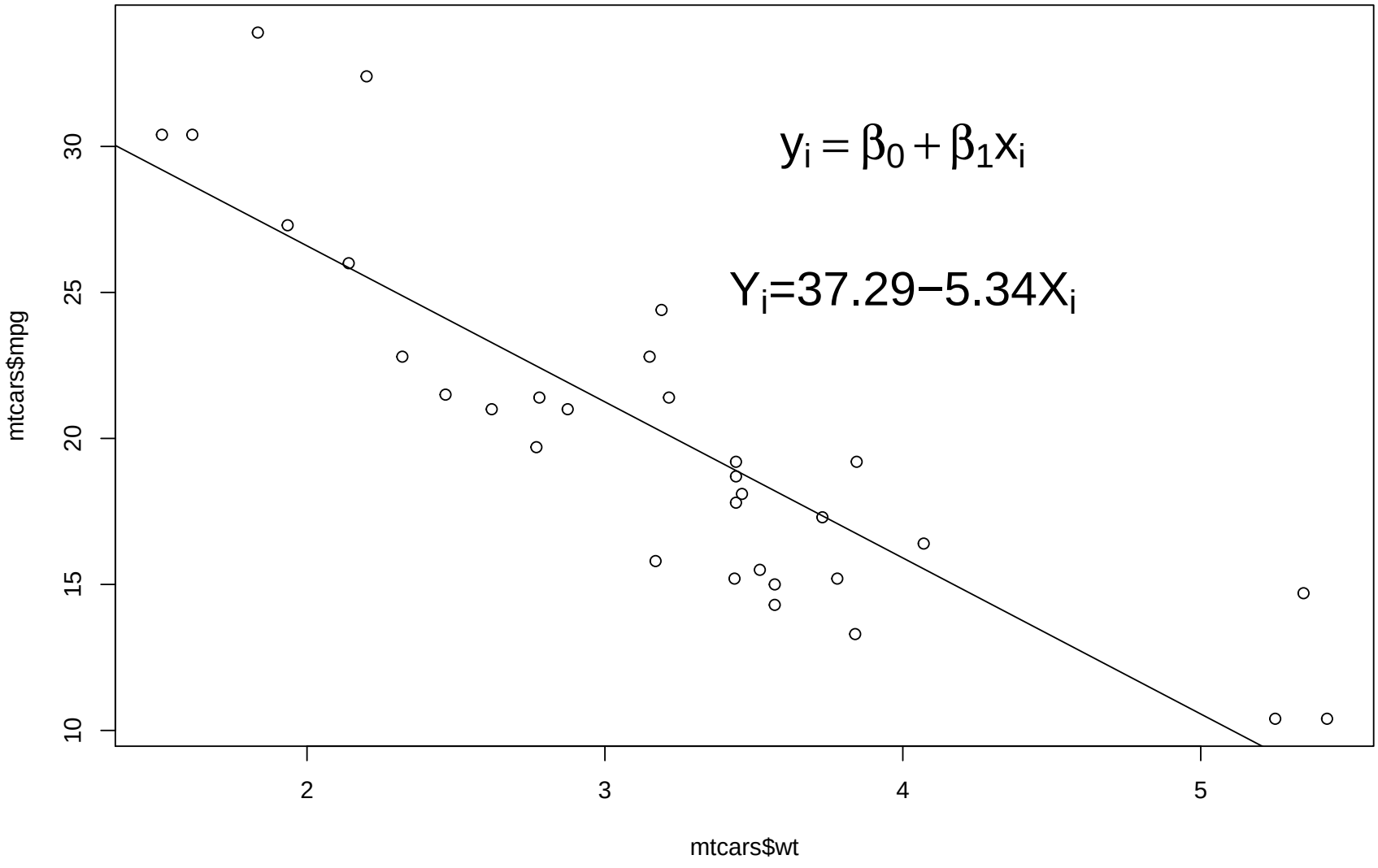

图 9:

# Equation notation

- `demo(plotmath)`

| Arithmetic Operators | | Lists | |
|---|---|---|---|
| x + y | $x + y$ | list(x, y, z) | $x, y, z$ |
| x - y | $x - y$ | **Relations** | |
| x * y | $xy$ | x == y | $x = y$ |
| x/y | $x/y$ | x != y | $x \neq y$ |
| x %+-% y | $x \pm y$ | x < y | $x < y$ |
| x%/%y | $x \div y$ | x <= y | $x \leq y$ |
| x %*% y | $x \times y$ | x > y | $x > y$ |
| x %.% y | $x \cdot y$ | x >= y | $x \geq y$ |
| -x | $-x$ | x %~~% y | $x \approx y$ |
| +x | $+x$ | x %=~% y | $x \cong y$ |
| **Sub/Superscripts** | | x %==% y | $x \equiv y$ |
| x[i] | $x_i$ | x %prop% y | $x \propto y$ |
| x^2 | $x^2$ | x %~% y | $x \sim y$ |
| **Juxtaposition** | | **Typeface** | |
| x * y | $xy$ | plain(x) | $x$ |
| paste(x, y, z) | $xyz$ | italic(x) | $x$ |
| **Radicals** | | bold(x) | $\mathbf{x}$ |
| sqrt(x) | $\sqrt{x}$ | bolditalic(x) | $\boldsymbol{x}$ |
| sqrt(x, y) | $\sqrt[y]{x}$ | underline(x) | $\underline{x}$ |

# Equation notation



## Equation notation

```
with(mtcars,plot(wt, mpg))
abline(lm(mpg~wt,data=mtcars))
result<-summary(lm(mpg~wt,data=mtcars))
text(4, 30, expression(y[i] == beta[0] + beta[1]*x[i]))

text(4,25,
substitute(
    paste(Y[i], "=", beta0, beta1,X[i]),
    list(beta0 = round(result$coe[1,1],2),
         beta1 = round(result$coe[2,1],2),2)))
```

## content

- create and save graphs
- customize plot: symbols,lines,colour
- annotate with text and titles
- *combined graphs*

# Margin and outer margin

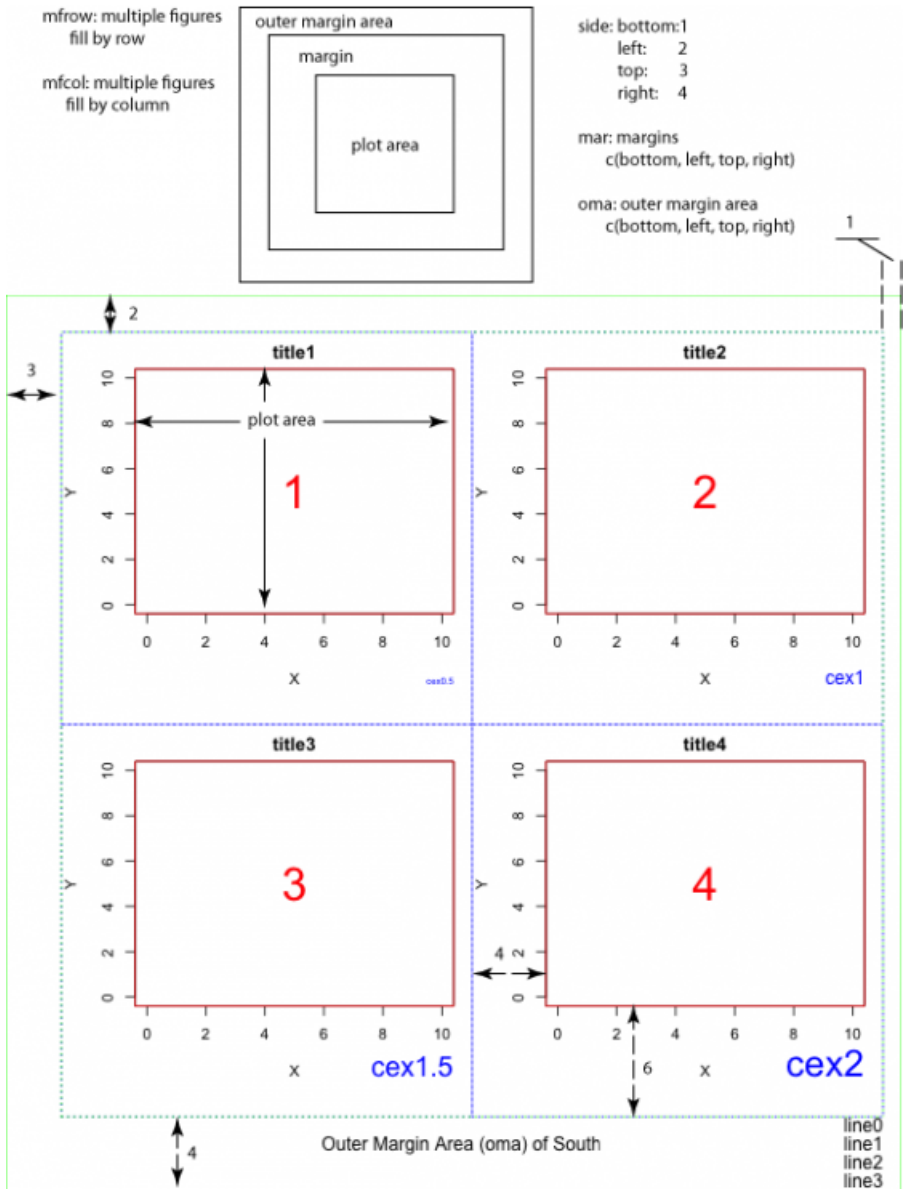
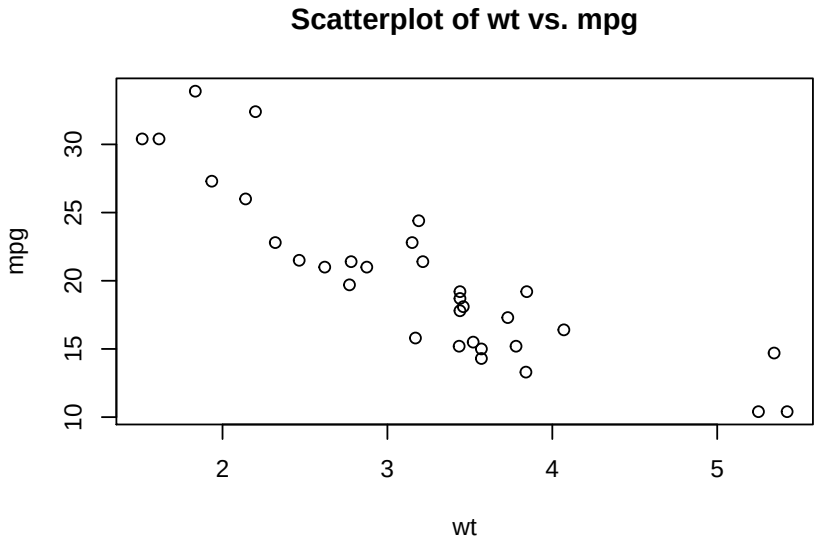
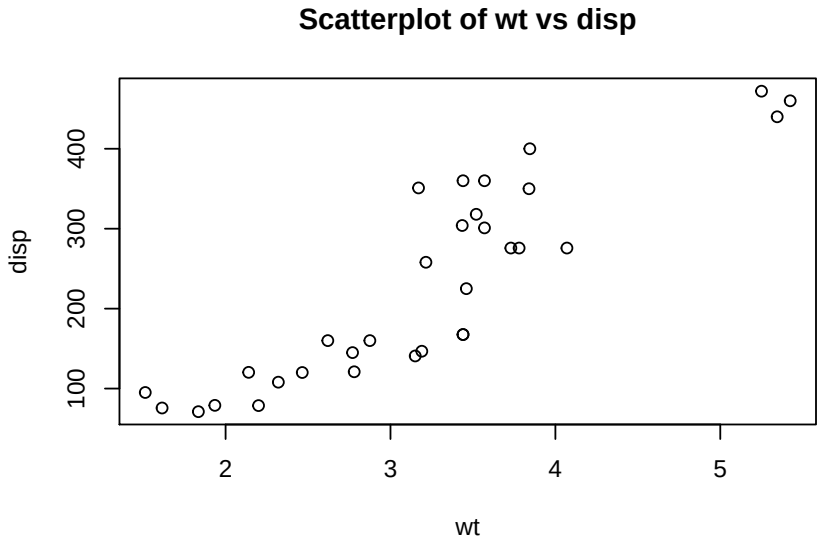
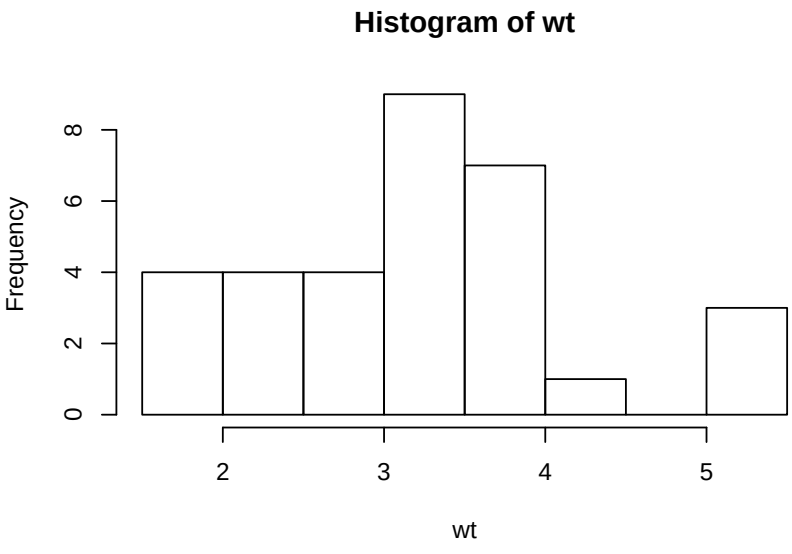


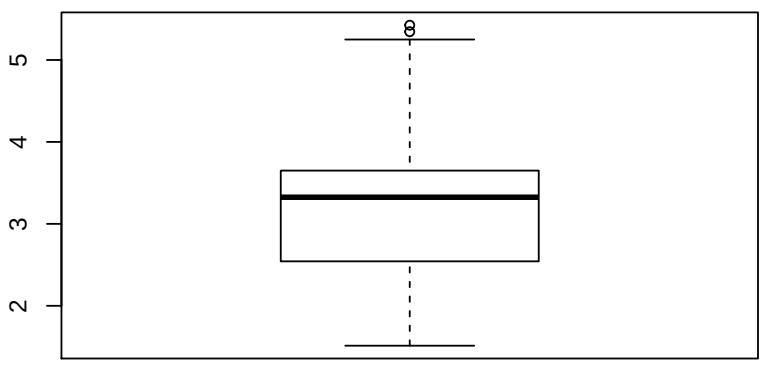

图 11:

# Combine graphs



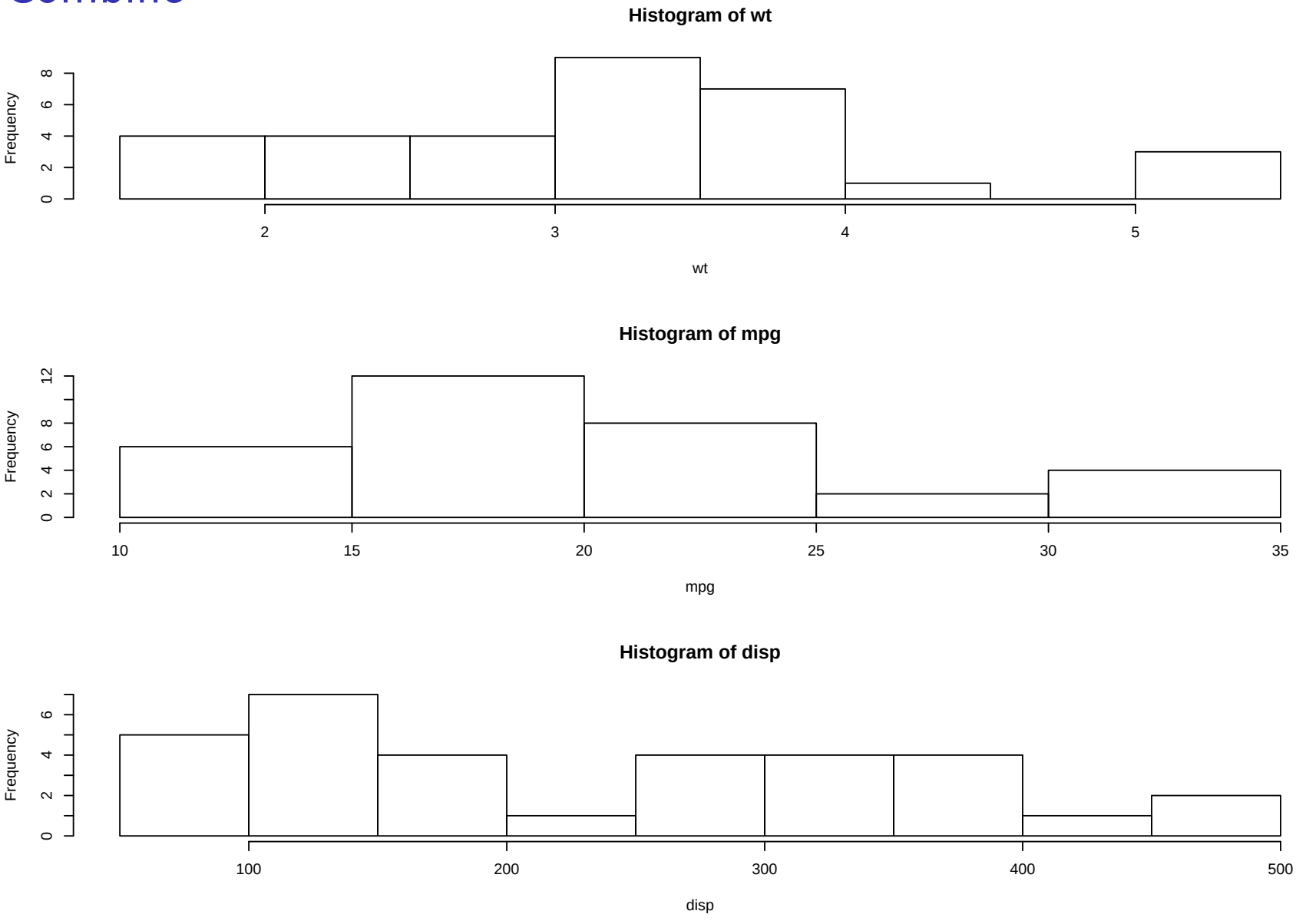



## Combine graphs

```
opar <- par(no.readonly = TRUE)
attach(mtcars)
par(mfrow = c(2,2))
plot(wt, mpg, main = "Scatterplot of wt vs. mpg")
plot(wt, disp,main = "Scatterplot of wt vs disp")
hist(wt, main = "Histogram of wt")
boxplot(wt, main = "Boxplot of wt")
detach(mtcars)
par(opar)
```

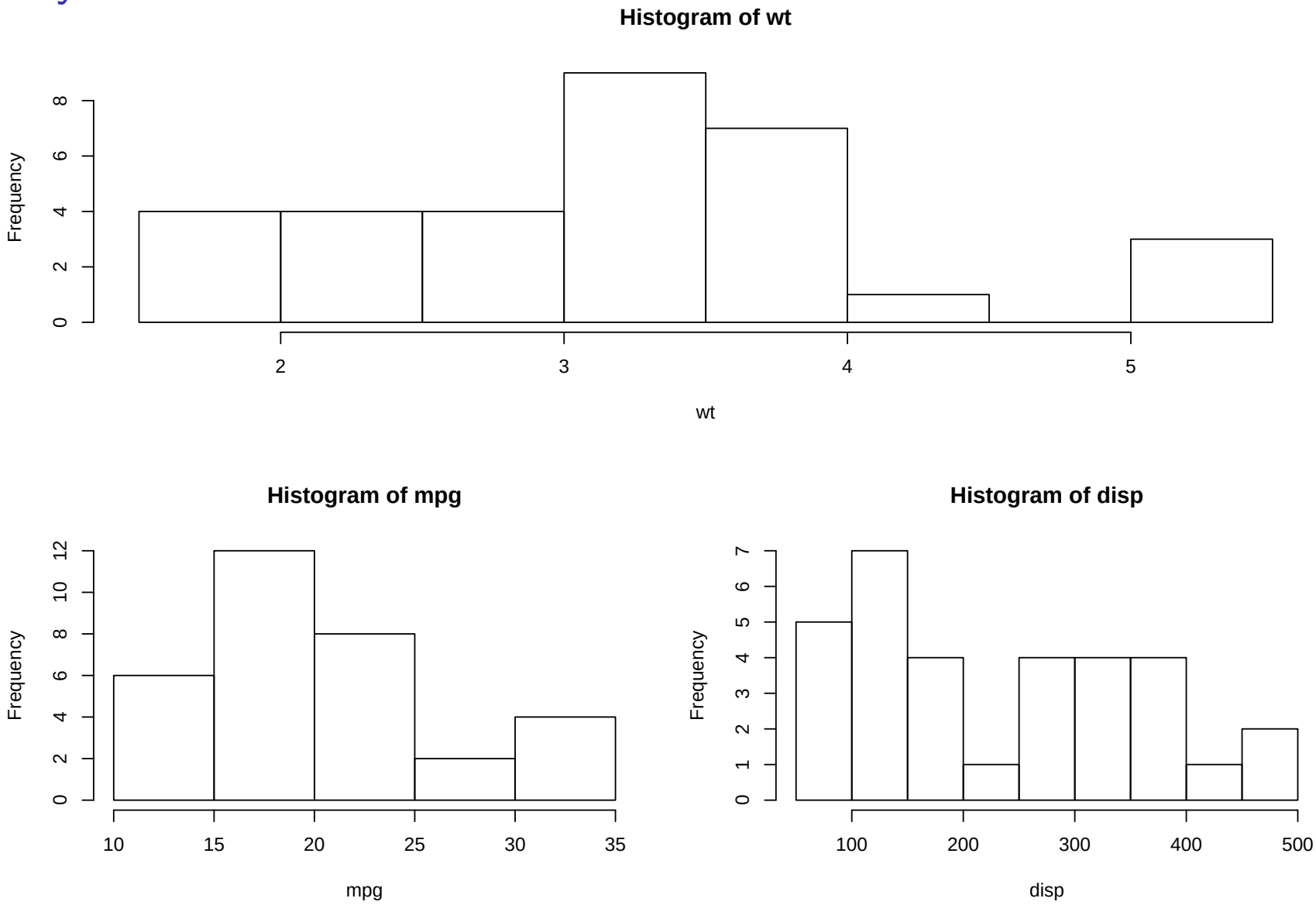
# Combine


Histogram of wt
Frequency
0
2
4
6
8
2
3
4
5
wt
Histogram of mpg
Frequency
0
2
4
6
8
12
10
15
20
25
30
35
mpg
Histogram of disp
Frequency
0
2
4
6
100
200
300
400
500
disp

## Combine

```
opar <- par(no.readonly = TRUE)
attach(mtcars)
par(mfrow = c(3, 1))
hist(wt)
hist(mpg)
hist(disp)
par(opar)
detach(mtcars)
```

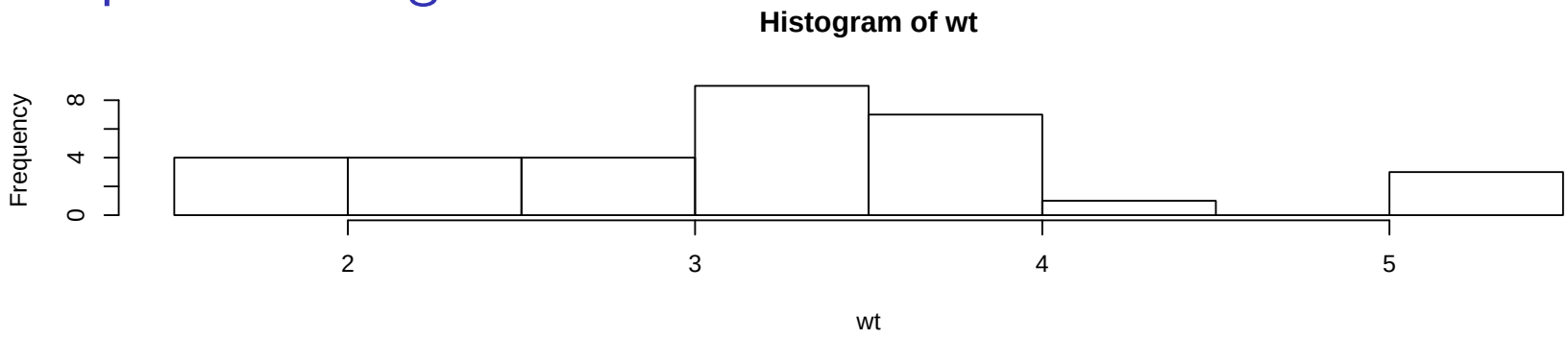
# Layout

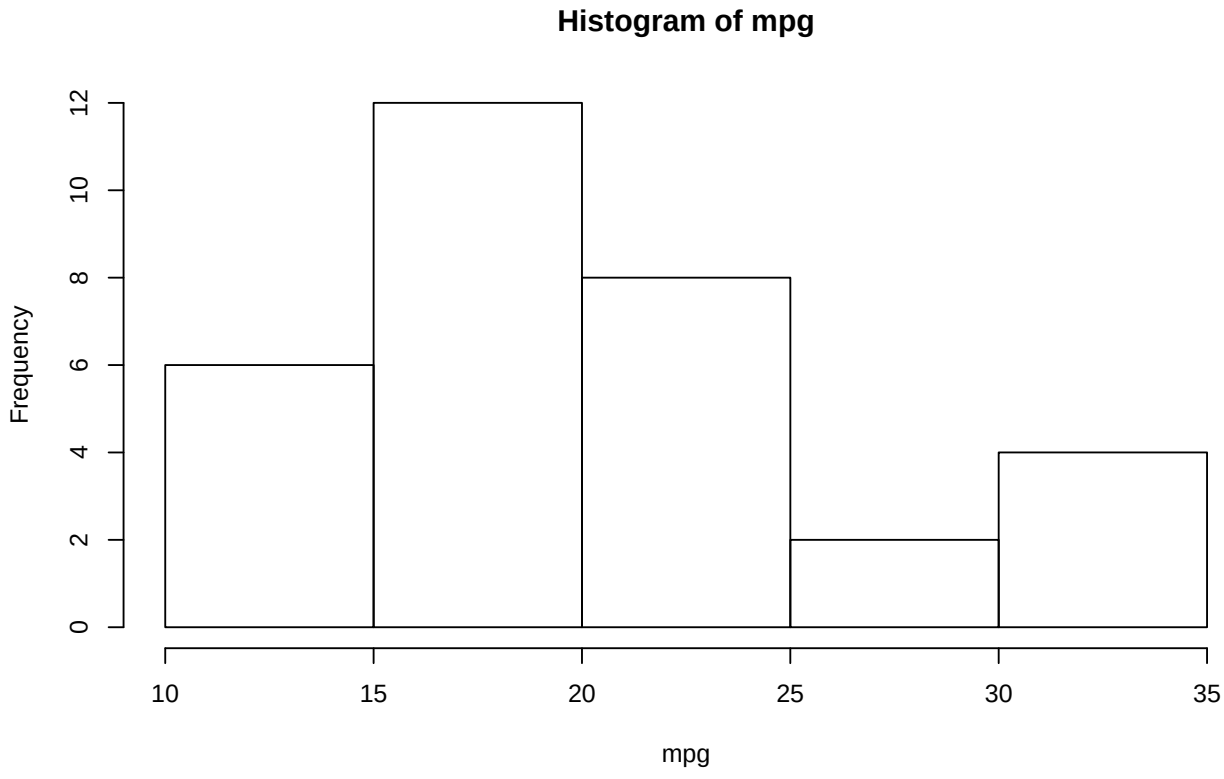

Histogram of wt
Frequency
0
2
4
6
8
2
3
4
5
wt
Histogram of mpg
Frequency
0
2
4
6
8
10
12
10
15
20
25
30
35
mpg
Histogram of disp
Frequency
0
1
2
3
4
5
6
7
100
200
300
400
500
disp

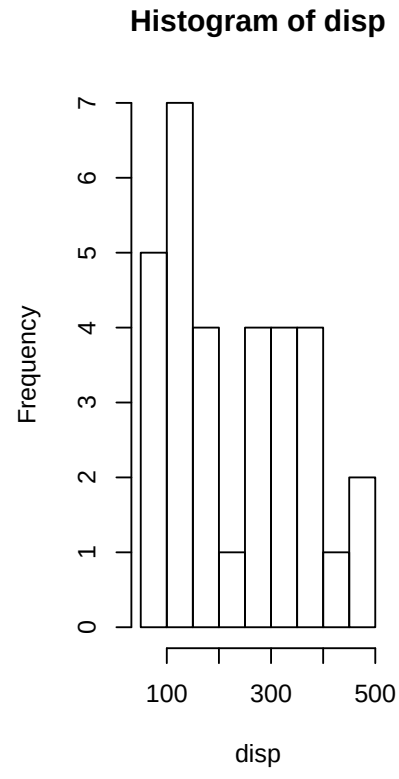
## Layout

```
attach(mtcars)
layout(matrix(c(1, 1, 2, 3), 2, 2, byrow = TRUE))
hist(wt)
hist(mpg)
hist(disp)
detach(mtcars)
```

# Proportion of figures



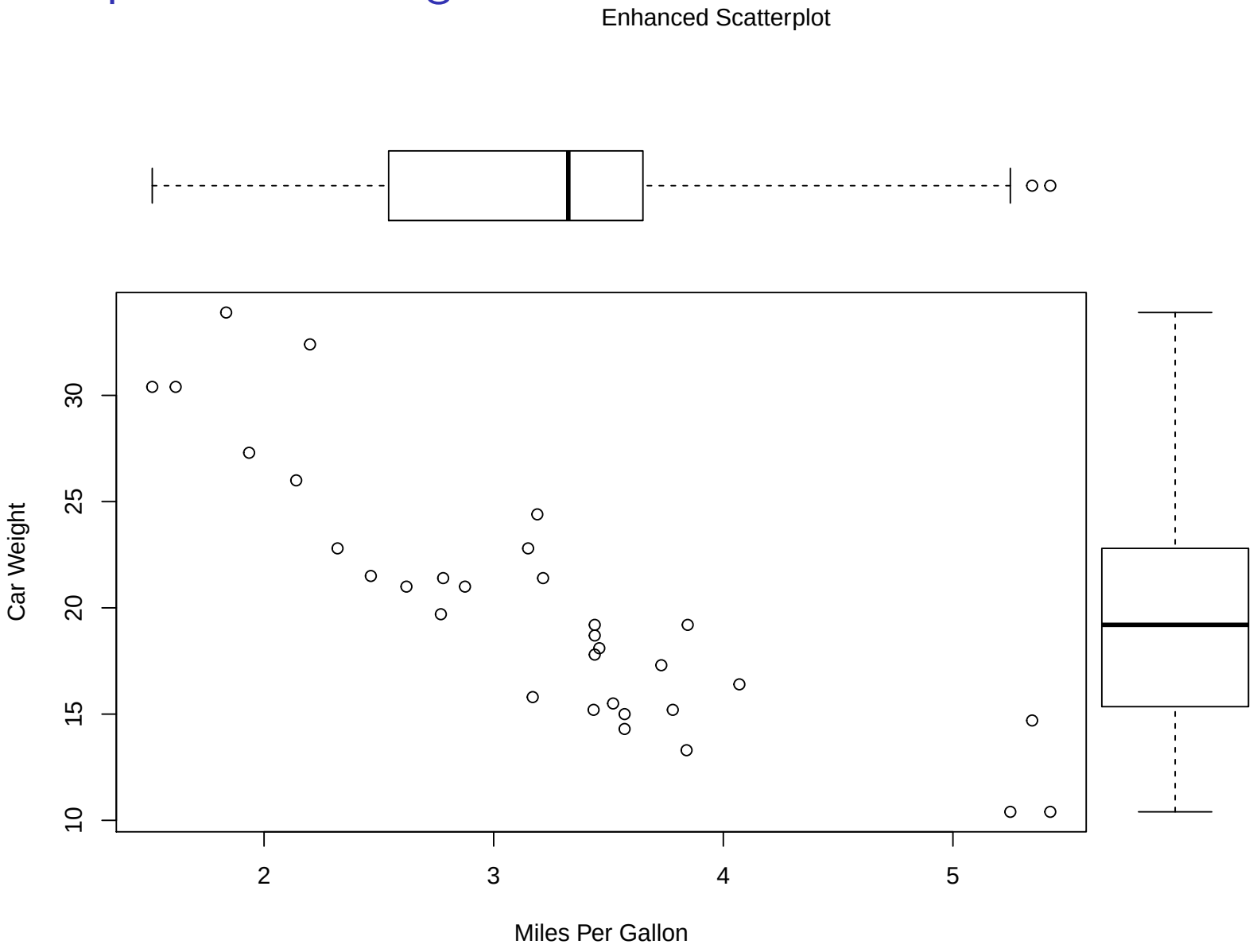



## Proportion of figures

```
par(mar=c(5.1,4.1,4.1,2))
par(oma=c(0,0,0,2))
attach(mtcars)
layout(matrix(c(1, 1, 2, 3), 2, 2, byrow = TRUE),
widths = c(3, 1), heights = c(1, 2))
#3:1 for heights 1:2 for widths
hist(wt)
hist(mpg)
hist(disp)
detach(mtcars)
```

# Fine placement of figures



## Fine placement of figures

```
opar <- par(no.readonly = TRUE)
par(fig = c(0, 0.8, 0, 0.8))
plot(mtcars$wt, mtcars$mpg, xlab = "Miles Per Gallon",
ylab = "Car Weight")
par(fig = c(0, 0.8, 0.55, 1), new = TRUE)
boxplot(mtcars$wt, horizontal = TRUE, axes = FALSE)
par(fig = c(0.6, 1, 0, 0.8), new = TRUE)
boxplot(mtcars$mpg, axes = FALSE)
mtext("Enhanced Scatterplot", side = 3, outer = TRUE,
line = -2)  ##side=3 top
par(opar)
```

## content

- create and save graphs
- customize plot: symbols,lines,colour
- annotate with text and titles
- combined graphs