Introduction to R programming for data science – day 5

Dr. Francesca Finotello
Medical University of Innsbruck, Austria

Plots

Plot

The plot function can be used to plot R objects. Its default usage produces a scatterplot of two variables *x* and *y*.

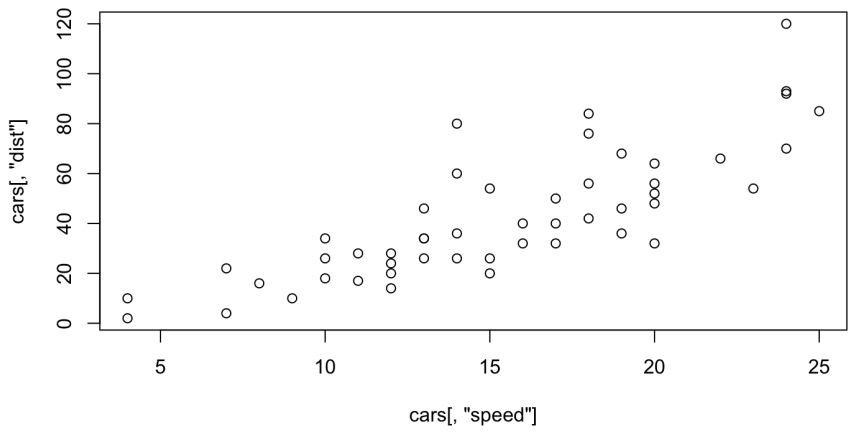
To show how plot works, we will use the dataset "cars" (already available as part of the "stats" package). It contains car speed and distance taken to stop recorded in the 1920s.

```
data(cars)
head(cars)
```

```
##   speed dist
## 1     4    2
## 2     4   10
## 3     7    4
## 4     7   22
## 5     8   16
## 6     9   10
```


Plot the *cars* dataset

```
plot(cars[,"speed"], cars[,"dist"])
```

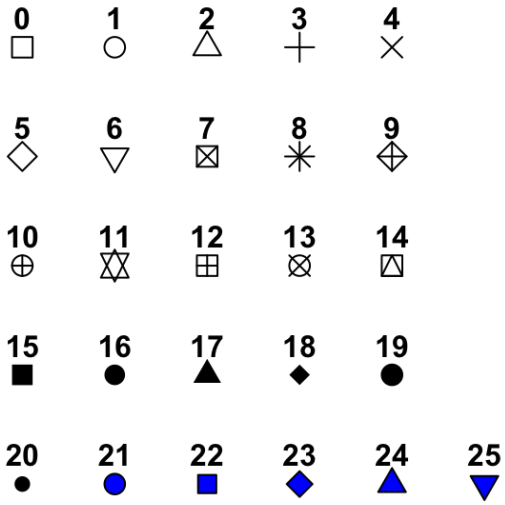


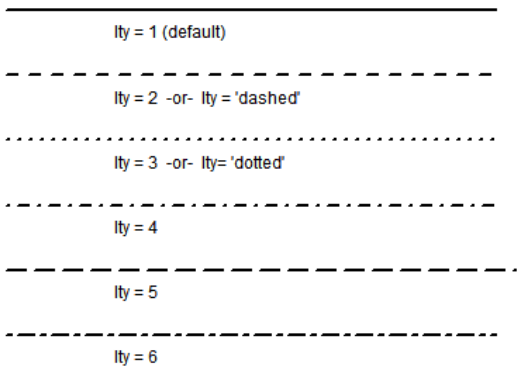
Plot arguments

The plot function has many arguments (see help(plot) and the web), among which:

- **type**: type of plot ("p" for points, "l" for lines, "b" for both…)
- **main**: title for the plot
- **xlab/ylab**: x/y-axis label
- **col**: colors for lines and points (single value or vector)
- **pch**: plotting characters or symbols (single value or vector)
- **lty**: line types (single value or vector)
- **lwd**: line width (default 1)
- **xlim/ylim**: numeric vectors giving the x/y-axis range
- **cex**: magnification (default 1; see also **cex.axis**, **cex.main**, **cex.lab**)

The *pch*, *lty*, and *col* arguments

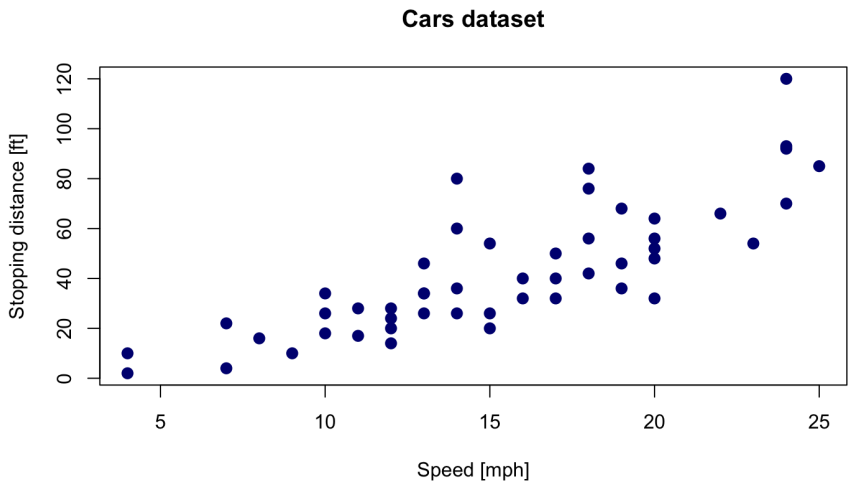




Colors can be specified as numbers, characters of color names or HEX codes, or using the rgb function (see next examples).

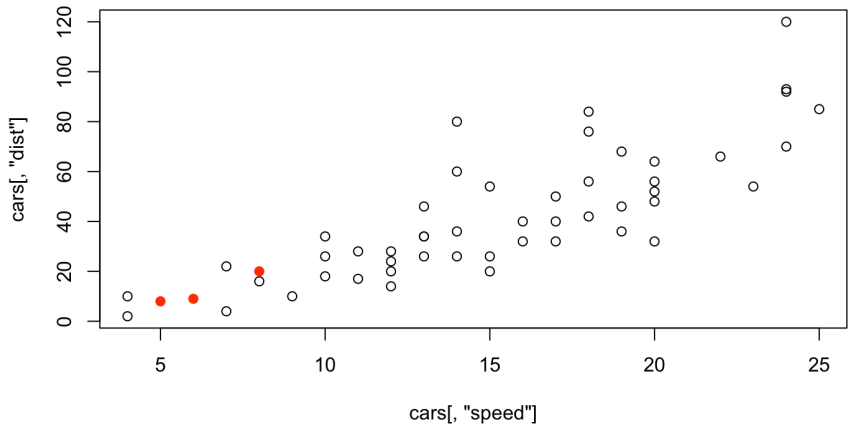
Re-plot the cars dataset

```
plot(cars[,"speed"], cars[,"dist"],
  xlab="Speed [mph]", ylab="Stopping distance [ft]",
  main="Cars dataset", pch=19, col="navyblue", cex=1.2)
```



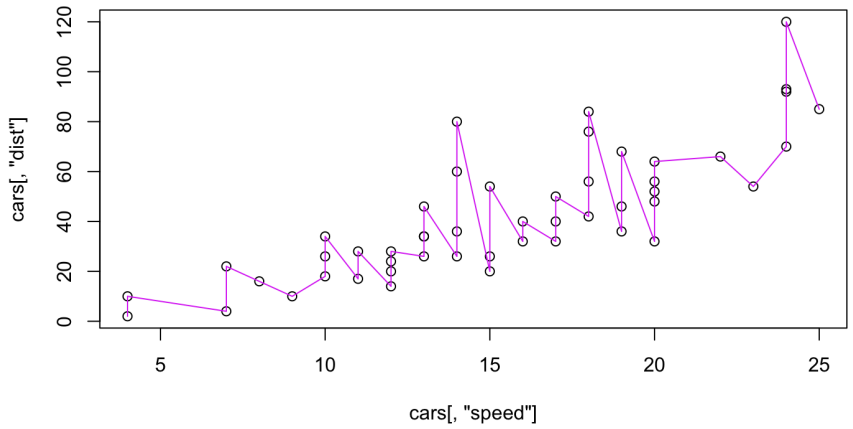
The *points* function

```
plot(cars[,"speed"], cars[,"dist"])
x <- c(5, 6, 8); y <- c(8, 9, 20)
points(x, y, col="orangered", pch=19) # Add points to a previous plot
```



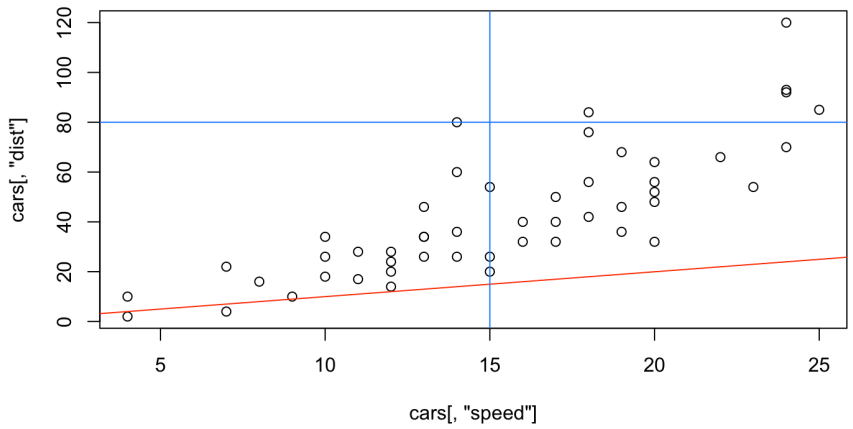
The *lines* function

```
plot(cars[,"speed"], cars[,"dist"])
lines(x=cars[,"speed"], y=cars[,"dist"],
  col="#dc42f4") # Add lines to a previous plot
```



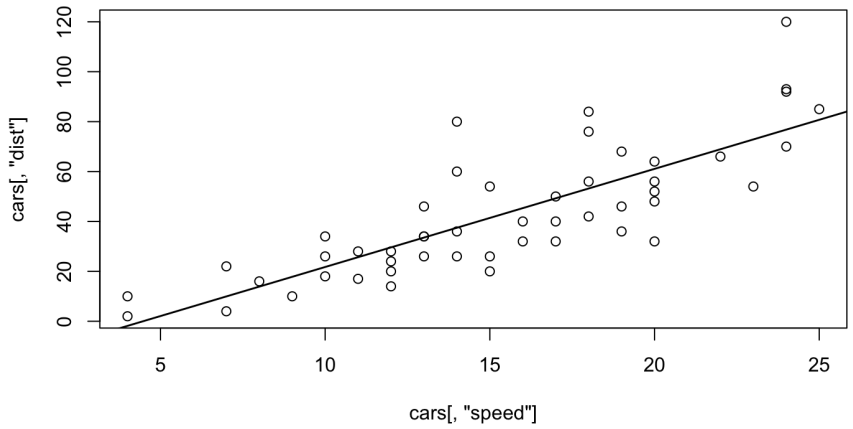
The *abline* function

```
plot(cars[,"speed"], cars[,"dist"])
abline(a=0, b=1, col="orangered") # Intercept and slope (here x=y)
abline(v=15, col="dodgerblue") # Vertical line
abline(h=80, col="dodgerblue") # Horizontal line
```



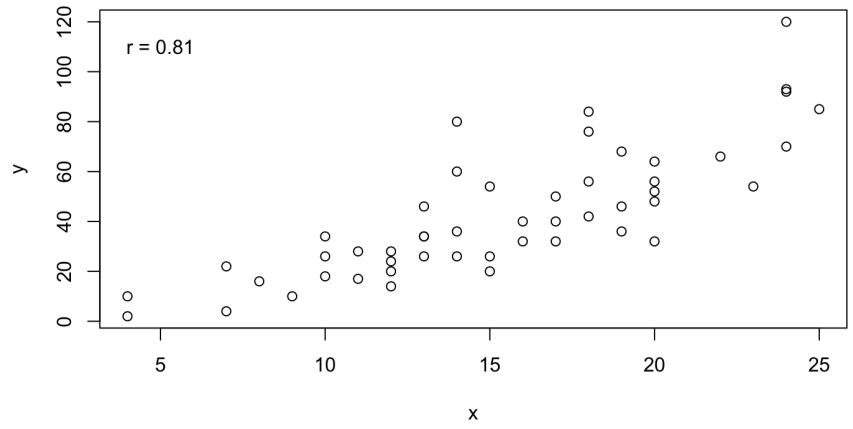
Adding a linear fit

```
plot(cars[,"speed"], cars[,"dist"])
lfit <- lm(dist~speed, data=cars) # Linear fit of dist (y) on speed (x)
abline(lfit, col=1, lwd=1.5)
```



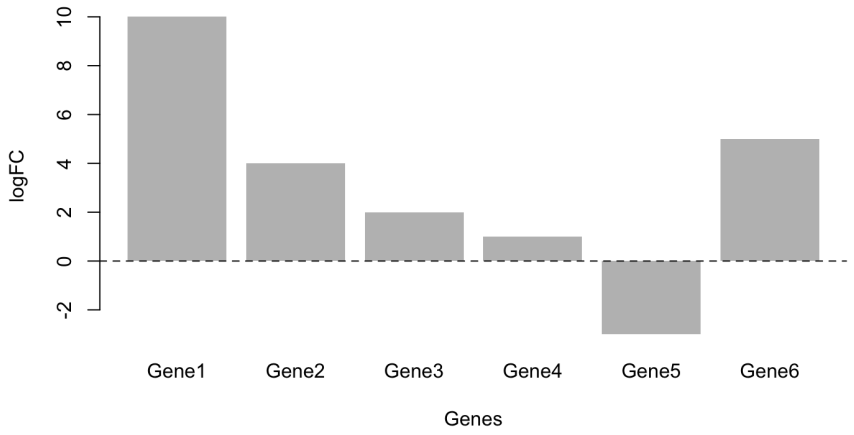
The *text* function

```
x <- cars[,"speed"]; y <- cars[,"dist"]; plot(x,y)
r <- round(cor(x,y), 2)
text(x=5, y=110, labels=paste("r =", r))
```



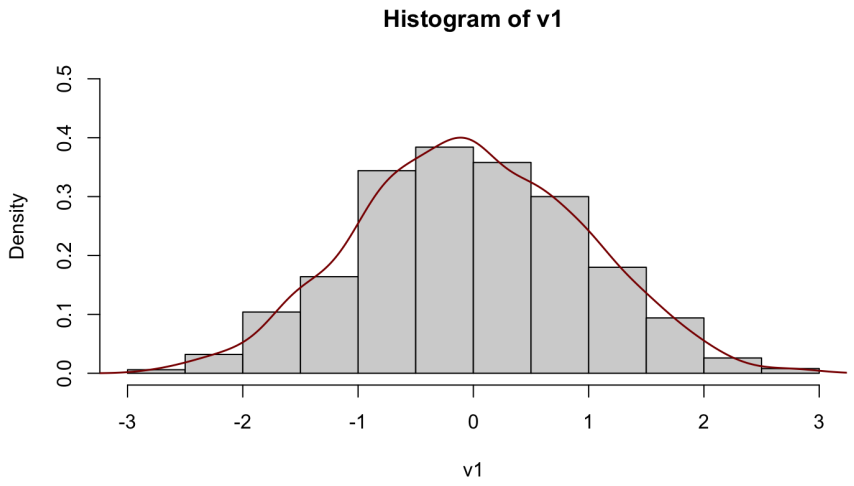
The *barplot* function

```
x <- c(10, 4, 2, 1, -3, 5)
names(x) <- c("Gene1", "Gene2", "Gene3", "Gene4", "Gene5", "Gene6")
barplot(x, xlab="Genes", ylab="logFC", border=NA)
abline(h=0, lty=2, col="grey4")
```



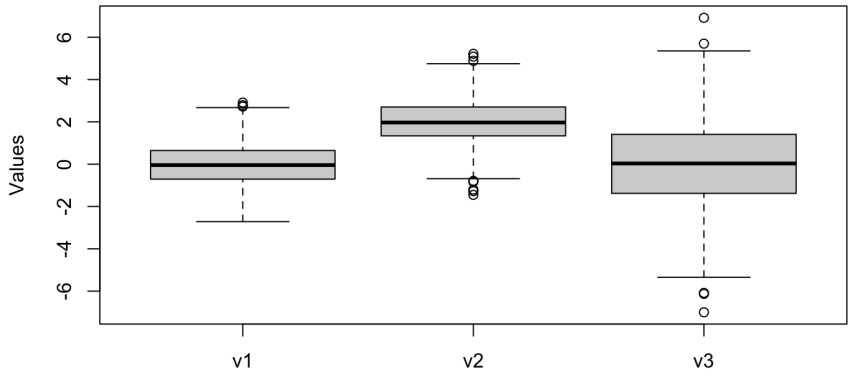
The *hist* and *density* functions

```
v1 <- rnorm(1000, mean=0, sd=1)
hist(v1, freq=FALSE, ylim=c(0,0.5))          # TRUE for frequencies, FALSE for densities
lines(density(v1), col="darkred", lwd=1.5)  # "density" does not plot
```

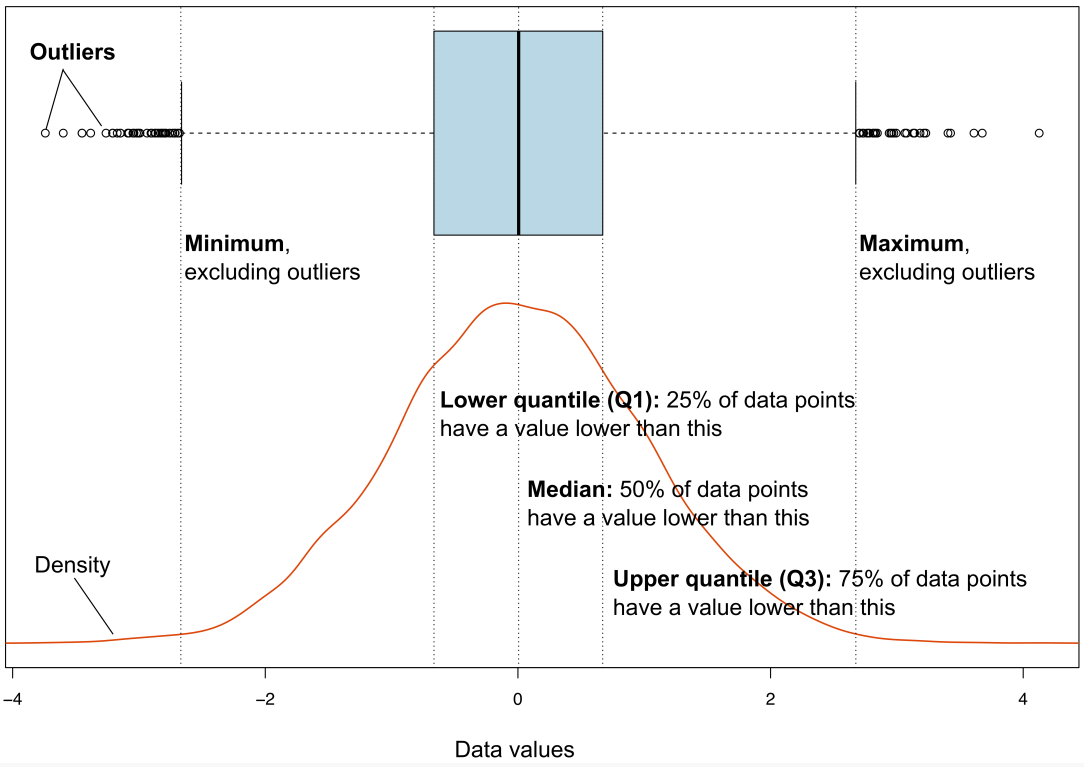


The *boxplot* function

```
v2 <- rnorm(1000, mean=2, sd=1)
v3 <- rnorm(1000, mean=0, sd=2)
boxplot(list(v1=v1, v2=v2, v3=v3), ylab="Values")
```



Boxplots



Save plots in a file

Functions like pdf and png can be used to save R plots into images and take as main input the figure name and path (see help for additional parameters)

```
pdf("Figures/boxplot.pdf")
boxplot(list(v1=v1, v2=v2, v3=v3), ylab="Values")
dev.off()
```

```
png("Figures/boxplot.png")
boxplot(list(v1=v1, v2=v2, v3=v3), ylab="Values")
dev.off()
```

dev.off must be used to close the device after plotting

dev.list can be used to know which devices are open

The *tidiverse* package collection

tidyverse is a collection of R packages designed for data science sharing common design philosophy, grammar, and data structures

You can install the complete **tidyverse** collection

```
install.packages("tidyverse")
```

Or only single packages, like ggplot2

```
install.packages("ggplot2")
```

The *ggplot2* package (1)

ggplot2 is a system for creating graphics, based on The Grammar of Graphics.

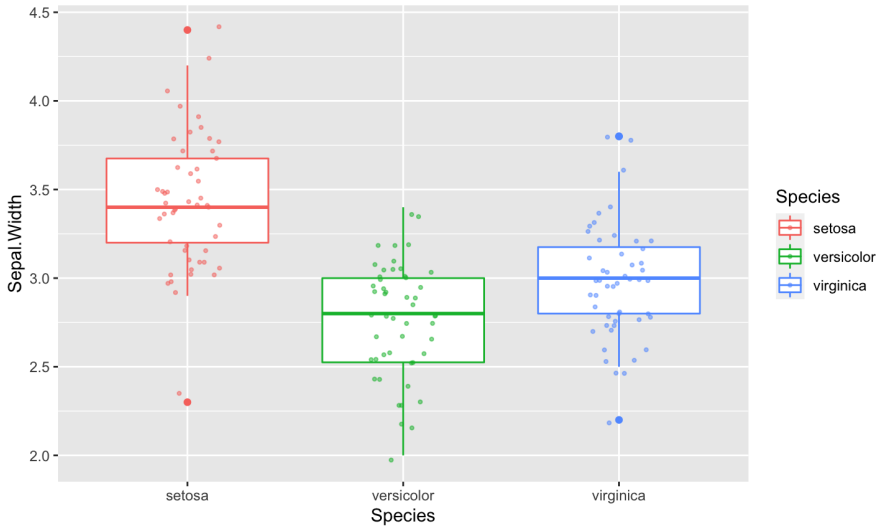
You specify:

- The data
- How to map variables to aesthetics
- Which graphical primitives to use

ggplot2 takes care of the details

The *ggplot2* package (2)

```
ggplot(iris, mapping = aes(x = Species, y = Sepal.Width, color = Species)) +
  geom_boxplot() +
  geom_jitter(width = 0.15, height = 0.05, alpha = 0.5, size = 0.75)
```



The *ggplot2* package (3)

```
ggplot(iris, mapping = aes(x = Species, y = Sepal.Width, color = Species)) +
  geom_violin() +
  geom_jitter(width = 0.15, height = 0.05, size = 0.8)
```