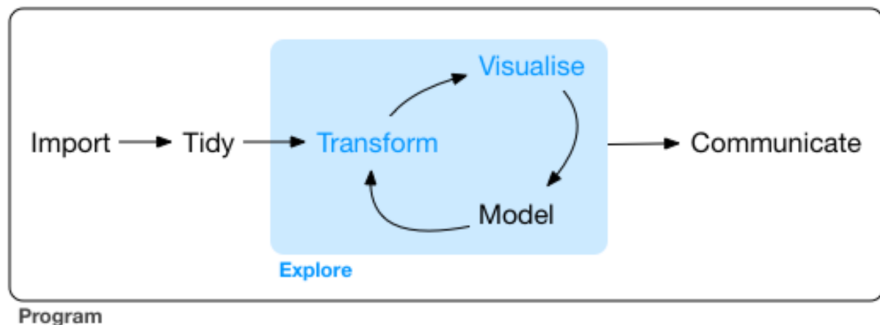


INTRODUCTION TO R

Lubor Homolka

September 6th, 2018

Data Science Flow



Wickham H, Grolemund G (2010). *R for Data Science*. O'Reilly.

It is often said that 80% of data analysis is spent on the process of cleaning and preparing the data (Dasu and Johnson 2003).

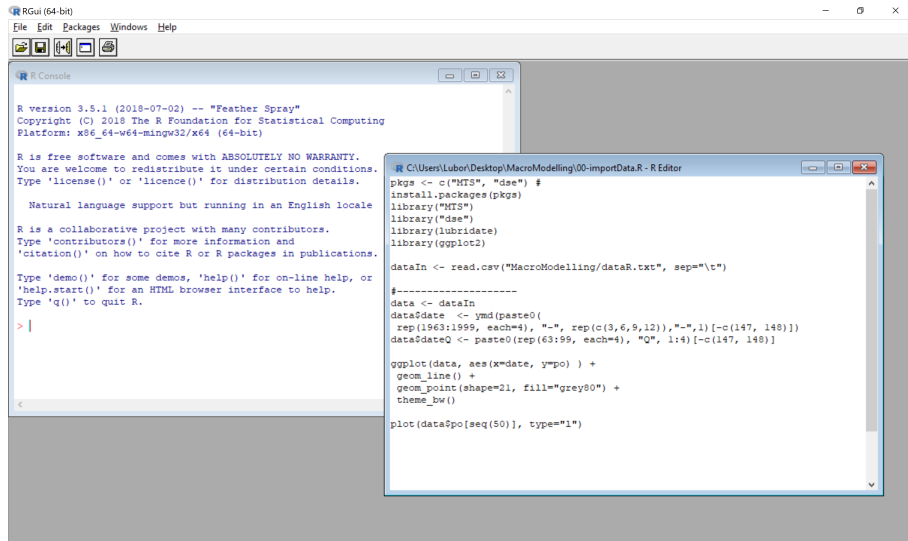
Dasu T, Johnson T (2003). *Exploratory Data Mining and Data Cleaning*.
John Wiley & Sons.

Obligatory intro to R and R language...

Before we start experimenting, we need to talk about:

- what is R
- basics of object-oriented programming

R GUI - here you spend the most time...



... until you need start doing reproducible research

The screenshot shows the RStudio interface with a script editor, environment pane, file explorer, and console.

```
C:/Users/Labor/Desktop/MyPaper/simpleVAR - RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help

# Report time
118 predObj <- forecast(FITARIMA, h=12)
119
120 predDF <- c(predObj$mean)
121
122 forecast <- data.frame(real=dataTest, fcst=predBF)
123
124 RMSE <- sd(forecast$real - forecast$fcst)
125 MAPE <- mean(abs(forecast$real - forecast$fcst)) / forecast$real * 100
126
127 perFresARIMA[, 1] <- countries[countryNum]
128 perFresARIMA[, 2] <- MAPE
129 perFresARIMA[, 3] <- RMSE
130 #cat("-----", print(MAPE), "****")
131
132 plotData <- data.frame(Reg=c(dataTrain.dataTest) )
133 plotData$time <- seq(nrow(plotData))
134
135 newt <- 85:96
136
137 plotTitle <- paste0( countries[countryNum], "\n",
138   "MAPE=", round(MAPE, 2), "%, ",
139   "RMSE=", round(RMSE, 2)
140 )
141
142 plot(plotDataReg ~ plotData$time, type="l", xaxt = "n", ylab="Car Registrations",
143   xlab="", main=plotTitle, ylim=c(0.1, 1+max(plotDataReg)))
144 axis(1, at=seq(0, 94, 12), labels=c(2011, 2017, 1))
145 lines(forecast$fcst-newt, col="red")
146
147 #
148 #
149
150 <--->
151 #orderedTab <- perFresVar[order(perFresVar$MAPE), ]
152 #rownames(orderedTab) <- C()
153 orderedTab <- merge(perFresARIMA, sindex, by="country") %>%
154   dplyr::arrange(MAPE)
155 kable(orderedTab, digits=2, "latex", booktabs = T, row.names = NA) %>%
156   kable_styling(latex_options = "striped")
157 #
158
159 \pagebreak
160
161 \section{Univariate Model} -- ETS)
162 <--->
163 <--->
164 # (Page 1)
```

Environment: empty

Files: History, Connections, Report Dataset, Global Environment

Files: Plots, Packages, Help, View, New Folder, Desktop, MyPaper, simpleVAR

Files: Name, Size, Modified

- History: 91 B, Jul 17, 2018, 8:34
- Data
- Figure
- Report - ksplos: 14.3 KB, Jun 13, 2018, 9:11
- Report-concordance: 205 B, Jul 17, 2018, 8:31
- Report: 33.7 KB, Jul 17, 2018, 8:31
- Report: 305.6 KB, Jul 17, 2018, 8:31
- Report: 10.1 KB, Jul 17, 2018, 8:31
- Report: 51.8 KB, Jul 17, 2018, 8:31
- Report: 32 KB, Jul 17, 2018, 8:31
- simpleVAR.Rproj: 217 B, Sep 4, 2018, 10:03

Console: Terminal

```
C:/Users/Labor/Desktop/MyPaper/simpleVAR
R version 3.5.1 (2018-07-02) -- "Feather Spray"
Copyright (C) 2018 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.
```

Object Oriented Programming

We have 4 students (Alice, Bob, Chelsea and Alice) who form a study group.

```
names <- c("Alice", "Bob", "Chelsea", "Alice")
```

names is an object.

Object Oriented Programming

We have 4 students (Alice, Bob, Chelsea and Alice) who form a study group.

```
names <- c("Alice", "Bob", "Chelsea", "Alice")
```

names is an object. We apply functions on objects:

```
unique(names)
```

```
## [1] "Alice" "Bob" "Chelsea"
```


Your turn - open R

- compute $1 + 1$

Your turn - open R

- compute $1 + 1$
- Click on File → New Script. Type $1 + 1$ into the script and press `ctrl+R`

Your turn - open R

- compute $1 + 1$
- Click on File → New Script. Type $1 + 1$ into the script and press ctrl+R
- store $1 + 1$ in object `simpleSum`

Your turn - open R

- compute $1 + 1$
- Click on File → New Script. Type $1 + 1$ into the script and press ctrl+R
- store $1 + 1$ in object `simpleSum`

```
simpleSum <- 1+1
```

- take $\sqrt{\text{simpleSum}}$, use `sqrt()` function

Your turn - open R

- compute $1 + 1$
- Click on File → New Script. Type $1 + 1$ into the script and press ctrl+R
- store $1 + 1$ in object `simpleSum`

```
simpleSum <- 1+1
```

- take $\sqrt{\text{simpleSum}}$, use `sqrt()` function

```
sqrt(simpleSum)  
## [1] 1.414214
```

Data Types

- numeric
- character, factor (important for variable coding, allows ordering)

```
a <- 1  
b <- "male"  
c <- "1"
```

```
class(a)
```

```
## [1] "numeric"
```

```
class(b)
```

```
## [1] "character"
```

```
class(c)
```

```
## [1] "character"
```

Other Important Types

What if we want to work with more values at the same time?

- `column = c()`
- `matrix = matrix()`
- `data frame = data.frame()`
- `list = list()`
- `tibble = tibble()`

Do you remember motivating example?

```
names <- c("Alice", "Bob", "Chelsea", "Alice")
```


Do you remember motivating example?

```
names <- c("Alice", "Bob", "Chelsea", "Alice")
```

Try following:

- 1 Create similar column called **scores**, which contains number of points from the test on a scale from 0–100 for each student.

Do you remember motivating example?

```
names <- c("Alice", "Bob", "Chelsea", "Alice")
```

Try following:

- 1 Create similar column called **scores**, which contains number of points from the test on a scale from 0–100 for each student.

```
scores <- c(60, 40, 55, 70)
```

- 2 Create `data.frame(names, scores)` and name it **classData**

Do you remember motivating example?

```
names <- c("Alice", "Bob", "Chelsea", "Alice")
```

Try following:

- 1 Create similar column called **scores**, which contains number of points from the test on a scale from 0–100 for each student.

```
scores <- c(60, 40, 55, 70)
```

- 2 Create `data.frame(names, scores)` and name it **classData**

```
classData <- data.frame(names, scores)
```

- 3 See what is inside of the `classData` object
- 4 Inspect `classData` by function `str()`

Do you remember motivating example?

```
names <- c("Alice", "Bob", "Chelsea", "Alice")
```

Try following:

- 1 Create similar column called **scores**, which contains number of points from the test on a scale from 0–100 for each student.

```
scores <- c(60, 40, 55, 70)
```

- 2 Create `data.frame(names, scores)` and name it **classData**

```
classData <- data.frame(names, scores)
```

- 3 See what is inside of the `classData` object
- 4 Inspect `classData` by function `str()`

```
str(classData)
```

```
## 'data.frame': 4 obs. of 2 variables:
```

```
## $ names : Factor w/ 3 levels "Alice","Bob",...: 1 2 3 1
```

```
## $ scores: num 60 40 55 70
```

5 Inspect `classData` by function `summary()`

5 Inspect `classData` by function `summary()`

```
summary(classData)
```

```
##          names          scores
## Alice   :2   Min.      :40.00
## Bob     :1   1st Qu.:51.25
## Chelsea:1   Median :57.50
##          Mean      :56.25
##          3rd Qu.:62.50
##          Max.      :70.00
```

5 Inspect `classData` by function `summary()`

```
summary(classData)

##          names          scores
## Alice   :2    Min.    :40.00
## Bob     :1    1st Qu.:51.25
## Chelsea:1    Median :57.50
##          Mean     :56.25
##          3rd Qu.:62.50
##          Max.     :70.00
```

Summary of the data is an object:

```
summaryTab <- summary(classData)
class(summaryTab) # ident. class(summary(classData))

## [1] "table"
```

Try it by yourself

- 1 generate 200 values from Normal distribution by function `rnorm(n=200)` and store values in `normData1`
- 2 compute standard deviation `sd()` and mean value `mean()`
- 3 create a histogram by `hist()`
- 4 generate another 200 values from Normal distribution and store them in object `normData2`
- 5 create `data.frame()` with 2 columns – `normData1` and `normData2` and call it `normalData`
- 6 create a scatter plot by `plot(normalData)`
- 7 compute a correlation of `normData1` and `normData2`, by function `cor()`. If you need help for function, type `?cor`

Import

Data Import

There are several ways how to read data into R:

- from your file (text file, Excel file)
- from online sources (Eurostat)
- other (web scraping through XML)

Data Import

There are several ways how to read data into R:

- from your file (text file, Excel file)
- from online sources (Eurostat)
- other (web scraping through XML)

Warning

Keeping data in MS Excel makes doing (untracable) changes too tempting!

Recommendation

Create one data file in .txt or .csv filetype. **DON'T** change it. Read it into R. Do all manipulations in R. All steps are recorded in the script.

- Store the file in `./myRproject/Data` folder

Recommendation

Create one data file in .txt or .csv filetype. **DON'T** change it. Read it into R. Do all manipulations in R. All steps are recorded in the script.

- Store the file in `./myRproject/Data` folder
- change R working directory

```
getwd() #to find current WD
```

```
## [1] "C:/Users/Lubor/Disk Google/Work/2018/R_workshop"
```

Recommendation

Create one data file in .txt or .csv filetype. **DON'T** change it. Read it into R. Do all manipulations in R. All steps are recorded in the script.

- Store the file in `./myRproject/Data` folder
- change R working directory

```
getwd() #to find current WD
```

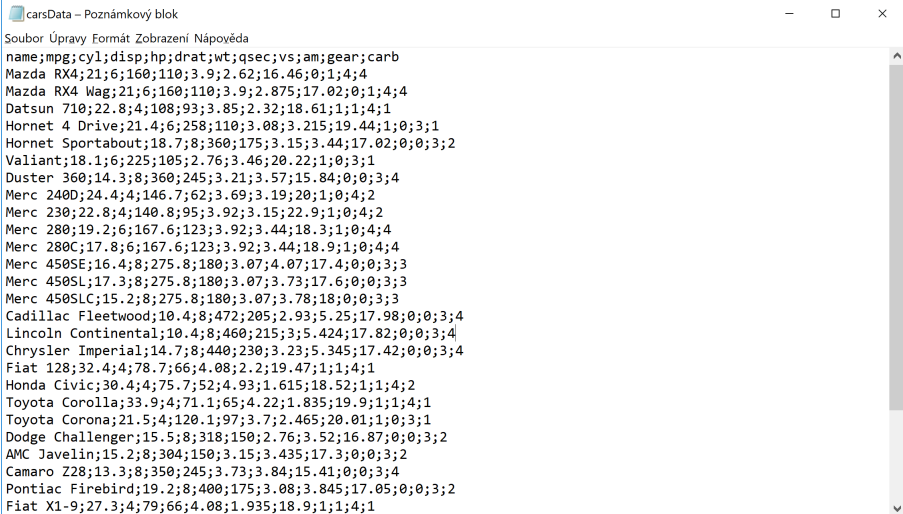
```
## [1] "C:/Users/Lubor/Disk Google/Work/2018/R_workshop"
```

Set the working directory one folder under data folder:

```
setwd("C:/Users/Lubor/Rworkshop")
```

if your data folder is in "C:/Users/Lubor/Rworkshop/Data"

Open carsData.txt



```
carsData - Poznámkový blok
Soubor Úpravy Formát Zobrazení Nápověda
name;mpg;cyl;disp;hp;drat;wt;qsec;vs;am;gear;carb
Mazda RX4;21;6;160;110;3.9;2.62;16.46;0;1;4;4
Mazda RX4 Wag;21;6;160;110;3.9;2.875;17.02;0;1;4;4
Datsun 710;22.8;4;108;93;3.85;2.32;18.61;1;1;4;1
Hornet 4 Drive;21.4;6;258;110;3.08;3.215;19.44;1;0;3;1
Hornet Sportabout;18.7;8;360;175;3.15;3.44;17.02;0;0;3;2
Valiant;18.1;6;225;105;2.76;3.46;20.22;1;0;3;1
Duster 360;14.3;8;360;245;3.21;3.57;15.84;0;0;3;4
Merc 240D;24.4;4;146.7;62;3.69;3.19;20;1;0;4;2
Merc 230;22.8;4;140.8;95;3.92;3.15;22.9;1;0;4;2
Merc 280;19.2;6;167.6;123;3.92;3.44;18.3;1;0;4;4
Merc 280C;17.8;6;167.6;123;3.92;3.44;18.9;1;0;4;4
Merc 450SE;16.4;8;275.8;180;3.07;4.07;17.4;0;0;3;3
Merc 450SL;17.3;8;275.8;180;3.07;3.73;17.6;0;0;3;3
Merc 450SLC;15.2;8;275.8;180;3.07;3.78;18;0;0;3;3
Cadillac Fleetwood;10.4;8;472;205;2.93;5.25;17.98;0;0;3;4
Lincoln Continental;10.4;8;460;215;3;5.424;17.82;0;0;3;4
Chrysler Imperial;14.7;8;440;230;3.23;5.345;17.42;0;0;3;4
Fiat 128;32.4;4;78.7;66;4.08;2.2;19.47;1;1;4;1
Honda Civic;30.4;4;75.7;52;4.93;1.615;18.52;1;1;4;2
Toyota Corolla;33.9;4;71.1;65;4.22;1.835;19.9;1;1;4;1
Toyota Corona;21.5;4;120.1;97;3.7;2.465;20.01;1;0;3;1
Dodge Challenger;15.5;8;318;150;2.76;3.52;16.87;0;0;3;2
AMC Javelin;15.2;8;304;150;3.15;3.435;17.3;0;0;3;2
Camaro Z28;13.3;8;350;245;3.73;3.84;15.41;0;0;3;4
Pontiac Firebird;19.2;8;400;175;3.08;3.845;17.05;0;0;3;2
Fiat X1-9;27.3;4;79;66;4.08;1.935;18.9;1;1;4;1
```

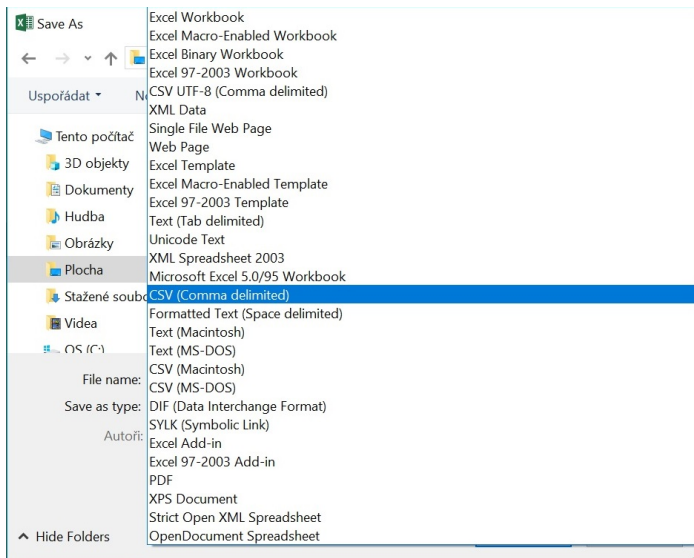
Open carsData.txt

How can I create similar file?

Use MS Excel

A6		Hornet Sportabout										
	A	B	C	D	E	F	G	H	I	J	K	L
1	name	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
2	Mazda RX4	21	6	160	110	3.9	2.62	16.46	0	1	4	4
3	Mazda RX4 Wag	21	6	160	110	3.9	2.875	17.02	0	1	4	4
4	Datsun 710	22.8	4	108	93	3.85	2.32	18.61	1	1	4	1
5	Hornet 4 Drive	21.4	6	258	110	3.08	3.215	19.44	1	0	3	1
6	Hornet Sportabout	18.7	8	360	175	3.15	3.44	17.02	0	0	3	2
7	Valiant	18.1	6	225	105	2.76	3.46	20.22	1	0	3	1
8	Duster 360	14.3	8	360	245	3.21	3.57	15.84	0	0	3	4
9	Merc 240D	24.4	4	146.7	62	3.69	3.19	20	1	0	4	2
10	Merc 230	22.8	4	140.8	95	3.92	3.15	22.9	1	0	4	2
11	Merc 280	19.2	6	167.6	123	3.92	3.44	18.3	1	0	4	4
12	Merc 280C	17.8	6	167.6	123	3.92	3.44	18.9	1	0	4	4
13	Merc 450SE	16.4	8	275.8	180	3.07	4.07	17.4	0	0	3	3
14	Merc 450SL	17.3	8	275.8	180	3.07	3.73	17.6	0	0	3	3
15	Merc 450SLC	15.2	8	275.8	180	3.07	3.78	18	0	0	3	3
16	Cadillac Fleetwood	10.4	8	472	205	2.93	5.25	17.98	0	0	3	4
17	Lincoln Continental	10.4	8	460	215	3	5.424	17.82	0	0	3	4
18	Chrysler Imperial	14.7	8	440	230	3.23	5.345	17.42	0	0	3	4
19	Fiat 128	32.4	4	78.7	66	4.08	2.2	19.47	1	1	4	1
20	Honda Civic	30.4	4	75.7	52	4.93	1.615	18.52	1	1	4	2
21	Toyota Corolla	33.9	4	71.1	65	4.22	1.835	19.9	1	1	4	1
22	Toyota Corona	21.5	4	120.1	97	3.7	2.465	20.01	1	0	3	1
23	Dodge Challenger	15.5	8	318	150	2.76	3.52	16.87	0	0	3	2

Save File as...



Let's do it!

```
setwd("C:/Users/Lubor/Rworkshop")
```

We will use function `read.table` and store data in the variable `dataIn`

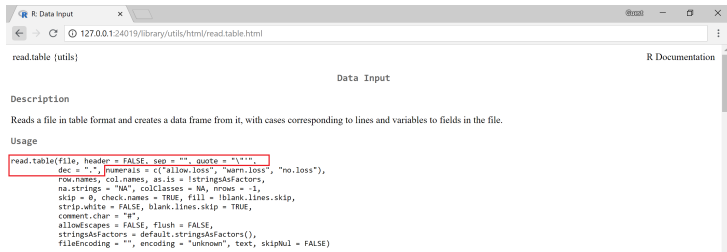
Let's do it!

```
setwd("C:/Users/Lubor/Rworkshop")
```

We will use function `read.table` and store data in the variable `dataIn`

Check the help file of the function:

`?read.table`



In our case:

```
dataIn <- read.table("./Data/carsData.txt",  
                      header = TRUE, sep=";")
```

In our case:

```
dataIn <- read.table("../Data/carsData.txt",  
                      header = TRUE, sep=";")
```

Inspect the first 6 rows by function `head()`

```
head(dataIn)
```

##		name	mpg	cyl	disp	hp	drat	wt	qsec
## 1		Mazda RX4	21.0	6	160	110	3.90	2.620	16.46
## 2		Mazda RX4 Wag	21.0	6	160	110	3.90	2.875	17.02
## 3		Datsun 710	22.8	4	108	93	3.85	2.320	18.61
## 4		Hornet 4 Drive	21.4	6	258	110	3.08	3.215	19.44
## 5		Hornet Sportabout	18.7	8	360	175	3.15	3.440	17.02
## 6		Valiant	18.1	6	225	105	2.76	3.460	20.22

Tidy Data

We talk about the tidy data when:

- 1 Each variable forms a column,
- 2 Each observation forms a row.
- 3 All data is stored in one table.

```
head(dataIn)
```

##		name	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear
## 1		Mazda RX4	21.0	6	160	110	3.90	2.620	16.46	0	1	
## 2		Mazda RX4 Wag	21.0	6	160	110	3.90	2.875	17.02	0	1	
## 3		Datsun 710	22.8	4	108	93	3.85	2.320	18.61	1	1	
## 4		Hornet 4 Drive	21.4	6	258	110	3.08	3.215	19.44	1	0	
## 5		Hornet Sportabout	18.7	8	360	175	3.15	3.440	17.02	0	0	
## 6		Valiant	18.1	6	225	105	2.76	3.460	20.22	1	0	

Transform

Welcome to the Tidyverse

Tidyverse is a collection of packages designed to work with tidy data.

```
library(tidyverse)
```

The most important libraries:

- dplyr – manipulation and transformation
- ggplot – advanced plotting
- tidyr – data reshaping tools

dplyr library

Core functions used in data manipulation:

- select
- filter
- mutate
- summarise
- group_by



But first, let's talk about pipes.

Pipes

Pipe `%>%` is a flow operator. It indicates how the data is passed from one function to another.

Consider following analysis:

```
rnorm(100) %>% abs() %>% sqrt() %>% hist()
```

Pipes

Pipe `%>%` is a flow operator. It indicates how the data is passed from one function to another.

Consider following analysis:

```
rnorm(100) %>% abs() %>% sqrt() %>% hist()
```

is equivalent to:

```
hist(sqrt(abs(rnorm(100))))
```

Pipes

Pipe `%>%` is a flow operator. It indicates how the data is passed from one function to another.

Consider following analysis:

```
rmnorm(100) %>% abs() %>% sqrt() %>% hist()
```

is equivalent to:

```
hist(sqrt(abs(rnorm(100))))
```

or even to:

```
normData <- rnorm(100)
absData  <- abs(normData)
sqrtData <- sqrt(absData)
hist(sqrtData)
```

Function: select

You want to do an analysis of **cyl** and **am** from the car dataset.

```
dataIn %>%  
  select(cyl, am) %>%  
  table()
```

```
##      am  
## cyl  0  1  
##   4  3  8  
##   6  4  3  
##   8 12  2
```

Function: `filter`

You want to do the same analysis as before, but only with $hp < 150$

Function: filter

You want to do the same analysis as before, but only with *hp* < 150

```
dataIn %>%  
  filter(hp < 150) %>%  
  select(cyl, am) %>%  
  table()
```

```
##      am  
## cyl 0 1  
##    4 3 8  
##    6 4 2
```


Function: filter

You want to do the same analysis as before, but only with $hp < 150$

```
dataIn %>%  
  filter(hp < 150) %>%  
  select(cyl, am) %>%  
  table()
```

```
##      am  
## cyl 0 1  
##    4 3 8  
##    6 4 2
```

What is this? $hp == 150$, $hp != 150$, $hp >= 150$

Function: `filter` - continued

- $A \ \& \ B$ means A **AND** B
- $A \ | \ B$ means A **OR** B

You want to do the same analysis as before, but only with
 $hp < 150$ **AND** $mpg > 14$

Function: filter - continued

- A & B means A **AND** B
- A | B means A **OR** B

You want to do the same analysis as before, but only with *hp* < 150 **AND** *mpg* > 14

```
dataIn %>%  
  filter(hp < 150 & mpg > 14) %>%  
  select(cyl, am) %>%  
  table()
```

```
##      am  
## cyl 0 1  
##    4 3 8  
##    6 4 2
```

Function: `mutate`.

Miles per gallon? We are in Europe! Let's convert it to units which make sense: 1 mile \sim 1.61 kilometer, 1 gallon \sim 3.79 liters. 1 mpg \sim 0.42 kmpl

```
carsData <- dataIn %>%  
  mutate(kmpl = 0.42*mpg)
```

Check whether it worked by displaying `head` of `carsData`. Show only name, mpg and kmpl columns:

Function: `mutate`.

Miles per gallon? We are in Europe! Let's convert it to units which make sense: 1 mile \sim 1.61 kilometer, 1 gallon \sim 3.79 liters. 1 mpg \sim 0.42 kmpl

```
carsData <- dataIn %>%  
  mutate(kmpl = 0.42*mpg)
```

Check whether it worked by displaying `head` of `carsData`. Show only name, mpg and kmpl columns:

Function: mutate.

Miles per gallon? We are in Europe! Let's convert it to units which make sense: 1 mile \sim 1.61 kilometer, 1 gallon \sim 3.79 liters. 1 mpg \sim 0.42 kmpl

```
carsData <- dataIn %>%  
  mutate(kmpl = 0.42*mpg)
```

Check whether it worked by displaying **head** of **carsData**. Show only name, mpg a and kmpl columns:

```
carsData %>% select(name, mpg, kmpl) %>% head()
```

```
##           name  mpg  kmpl  
## 1      Mazda RX4 21.0 8.820  
## 2    Mazda RX4 Wag 21.0 8.820  
## 3    Datsun 710 22.8 9.576  
## 4  Hornet 4 Drive 21.4 8.988  
## 5 Hornet Sportabout 18.7 7.854  
## 6     Valiant 18.1 7.602
```

Function: summarise

We want to know:

- 1 mean value of kmpl = **meanKMPL**
- 2 mean value of hp = **meanHP**

```
carsData %>%  
  summarise(  
    meanKMPL = mean(kmpl),  
    meanHP   = mean(hp)  
  )
```

```
##      meanKMPL    meanHP  
## 1  8.438062 146.6875
```

We have created a new data frame with 2 columns.

Function: `group_by`

Let's continue with the previous example. You want to do the analysis on subsets/groups by variable: `group_by(cyl)`

```
carsData %>%  
  group_by(cyl) %>%  
  summarise(  
    meanKMPL = mean(kmpl),  
    meanHP    = mean(hp)  
  )
```

```
## # A tibble: 3 x 3  
##   cyl meanKMPL meanHP  
##   <int>   <dbl>   <dbl>  
## 1     4    11.2    82.6  
## 2     6     8.29   122.  
## 3     8     6.34   209.
```


Visualise

Data Overview

```
str(dataIn)
```

```
## 'data.frame': 32 obs. of 12 variables:
## $ name: Factor w/ 32 levels "AMC Javelin",...: 18 19 5 13 14 31 ...
## $ mpg : num 21 21 22.8 21.4 18.7 18.1 14.3 24.4 22.8 19.2 ...
## $ cyl : int 6 6 4 6 8 6 8 4 4 6 ...
## $ disp: num 160 160 108 258 360 ...
## $ hp : int 110 110 93 110 175 105 245 62 95 123 ...
## $ drat: num 3.9 3.9 3.85 3.08 3.15 2.76 3.21 3.69 3.92 3.92 ..
## $ wt : num 2.62 2.88 2.32 3.21 3.44 ...
## $ qsec: num 16.5 17 18.6 19.4 17 ...
## $ vs : int 0 0 1 1 0 1 0 1 1 1 ...
## $ am : int 1 1 1 0 0 0 0 0 0 0 ...
## $ gear: int 4 4 4 3 3 3 3 4 4 4 ...
## $ carb: int 4 4 1 1 2 1 4 2 2 4 ...
```

Library ggplot

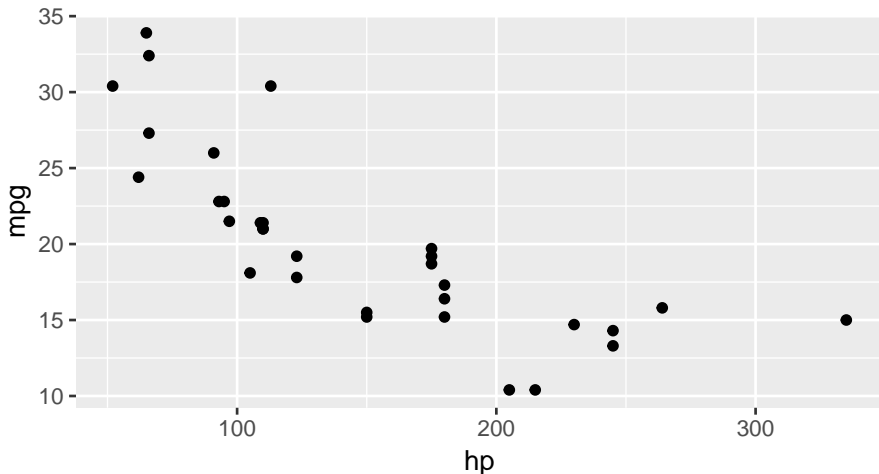
Library ggplots introduces plotting by layers.

- 1 Define underlying layer (data + variables)
- 2 Add **layers** (points, boxplots, . . .)

Every `ggplot()` plot contains **data reference** and **aesthetic mapping**.

```
ggplot(dataIn, aes(x=hp, y=mpg)) +  
  geom_point()
```

```
library(ggplot2)
ggplot(dataIn, aes(x=hp, y=mpg)) +
  geom_point()
```



Library ggplot

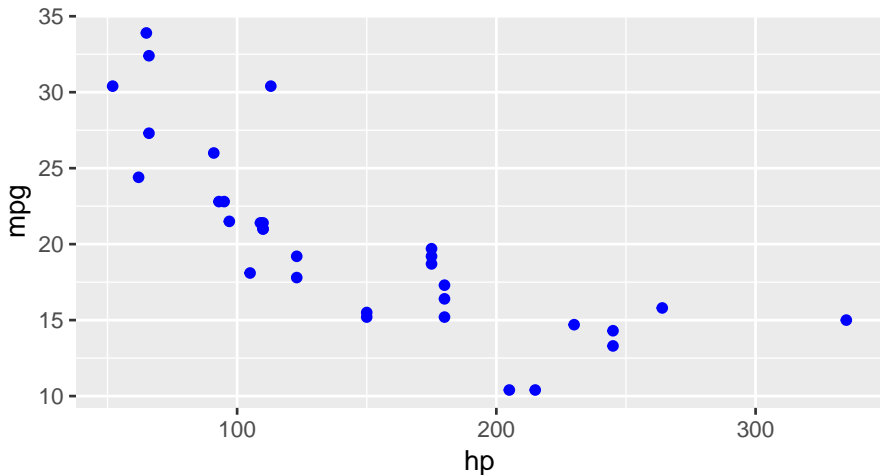
What makes ggplot powerful is an application of aesthetics:

- color
- fill
- size
- shape
- linetype

Take a look at following two plots.

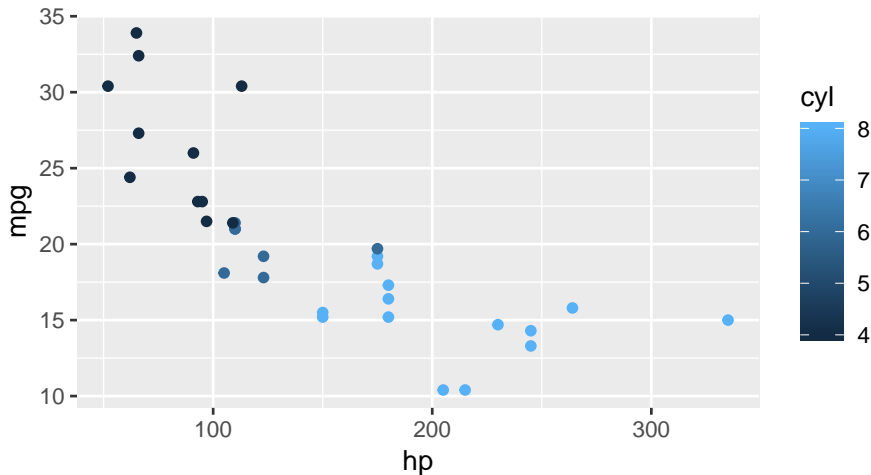
color as a Graphical Parameter

```
ggplot(dataIn, aes(x=hp, y=mpg)) +  
  geom_point(color="blue")
```

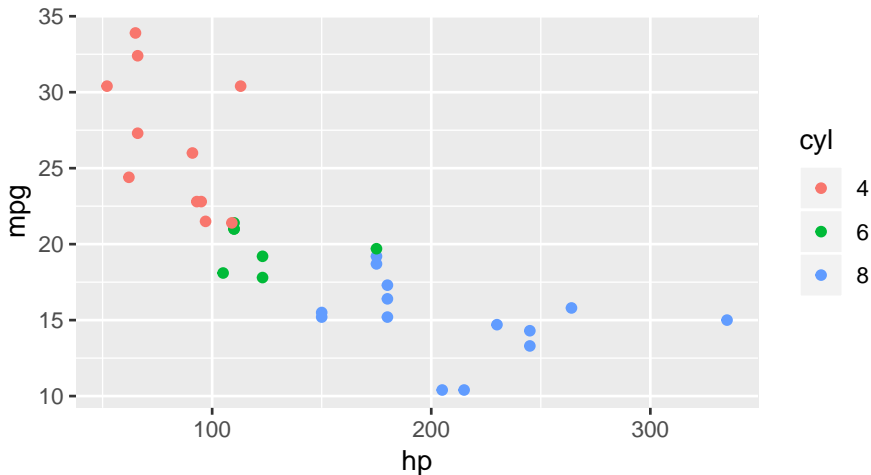


color as an aesthetics

```
ggplot(dataIn, aes(x=hp, y=mpg, color=cyl)) +  
  geom_point()
```



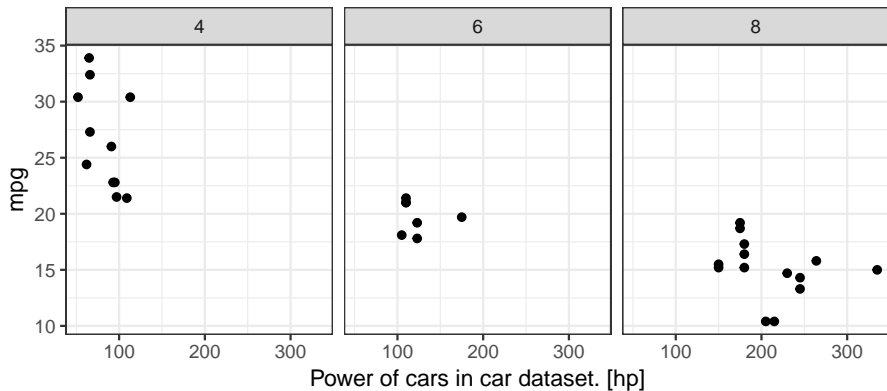
```
dataIn %>%  
  mutate(cyl = factor(cyl)) %>%  
  ggplot(., aes(x=hp, y=mpg, color=cyl)) +  
    geom_point()
```



We can adjust (almost) everything in ggplot through additional functions (just to mention few):

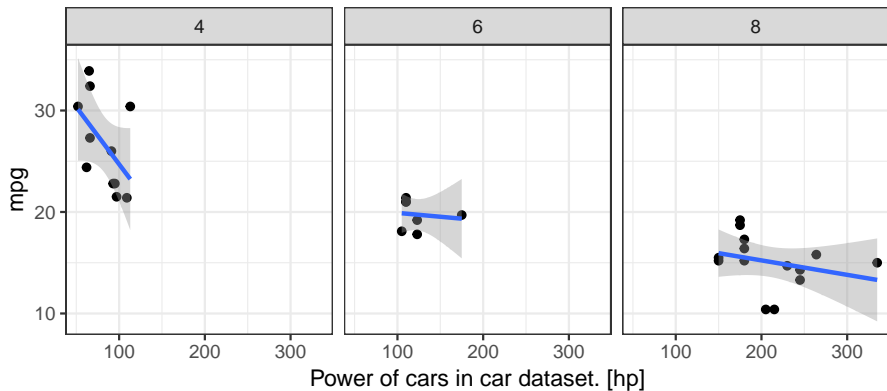
- `facet_grid()`
- `scale_y_continuous(name, limits, ...)`,
`scale_x_discrete()`
- `theme_bw()`

```
dataIn %>%
  mutate(cyl = as.factor(cyl)) %>%
  ggplot(., aes(x=hp, y=mpg)) +
  geom_point() +
  scale_x_continuous("Power of cars in car dataset. [hp]") +
  facet_grid(.~cyl) +
  theme_bw()
```



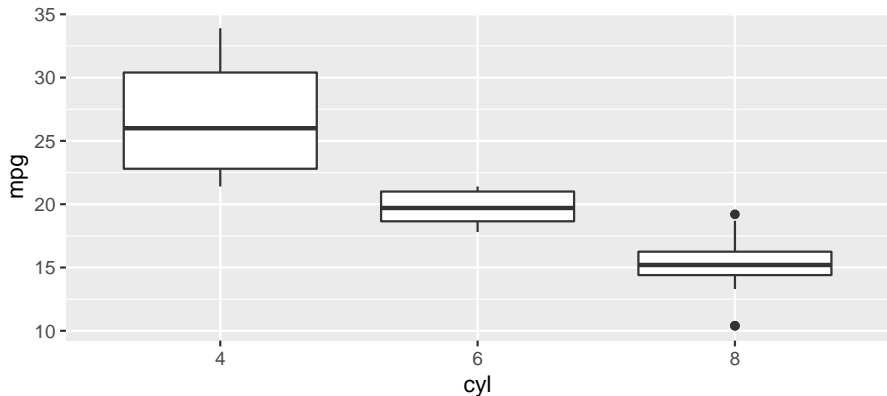
Why not to add regressions by `geom_smooth()`?

```
dataIn %>%  
  mutate(cyl = as.factor(cyl)) %>%  
  ggplot(., aes(x=hp, y=mpg)) + geom_point() +  
  scale_x_continuous("Power of cars in car dataset. [hp]") +  
  facet_grid(.~cyl) + theme_bw() + geom_smooth(method="lm")
```



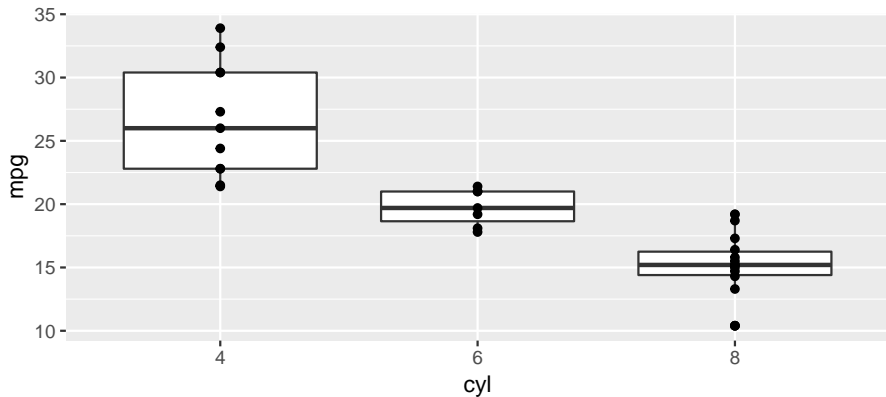
geom_boxplot

```
dataIn %>%  
  mutate(cyl = as.factor(cyl)) %>%  
  ggplot(aes(x=cyl, y=mpg)) +  
    geom_boxplot()
```



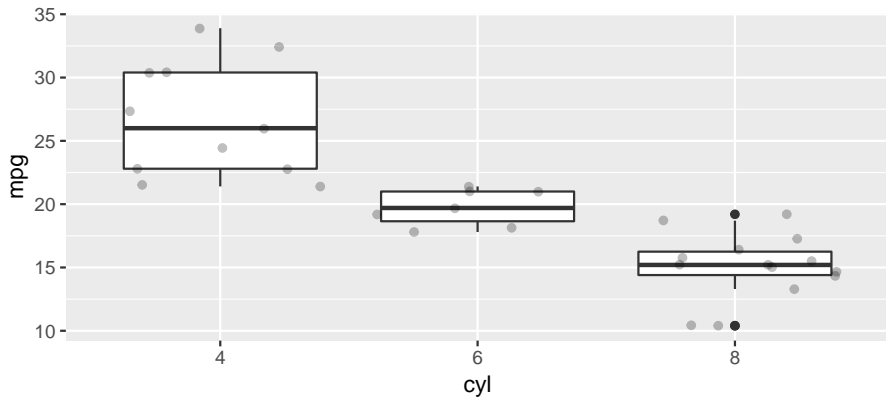
Let's add additional layer

```
dataIn %>%  
  mutate(cyl = as.factor(cyl)) %>%  
  ggplot(aes(x=cyl, y=mpg)) +  
    geom_boxplot() +  
    geom_point()
```



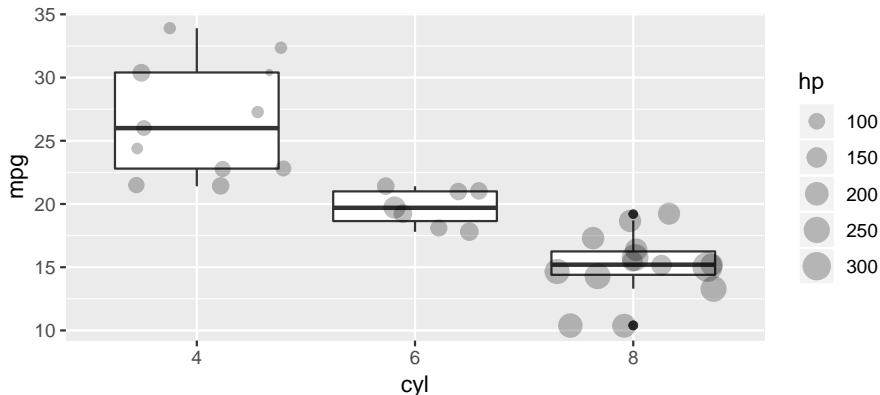
Add geom-specific aes

```
dataIn %>%  
  mutate(cyl = as.factor(cyl)) %>%  
  ggplot(aes(x=cyl, y=mpg)) +  
    geom_boxplot() +  
    geom_jitter(alpha=0.25)
```



Add geom-specific aes

```
dataIn %>%
  mutate(cyl = as.factor(cyl)) %>%
  ggplot(aes(x=cyl, y=mpg)) +
  geom_boxplot() +
  geom_jitter(aes(size=hp), alpha=0.25)
```



Processing ggplots

```
plot1 <- dataIn %>%  
  mutate(cyl = as.factor(cyl)) %>%  
  ggplot(aes(x=cyl, y=mpg) ) +  
    geom_boxplot() +  
    geom_point()  
  
plot1  
ggsave("./images/myPlot1.pdf", width=6, height = 4) # png
```

Try following:

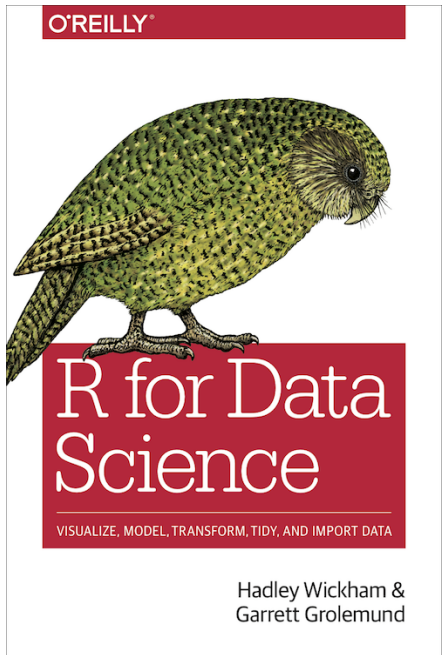
```
library(plotly)  
ggplotly(plot1)
```


Your Turn

There is a dataset in ggplot library called diamonds.

- 1 familiarise yourselves with the dataset: `?diamonds`
- 2 how many observations is in the dataset? Find this information in R!
- 3 Create a table from the main dataset which analyses price of diamonds and visualise it in ggplot.
- 4 Try to combine `filter`, `group_by` and **summarise**

- Book about R
- and good DS practices.
- Online version available [here](#).



Thank you!

homolka@utb.cz

