Data Science with R

Victor Coppin

2027-05-06

Table of contents

Preface

This is a Quarto book.

To learn more about Quarto books visit https://quarto.org/docs/books.

Part I

R Basics : Introduction to Data Science

1 The Tidyverse

The Tidyverse can be installed with a single line of code: install.packages("tidyverse")

This command installs the nine core packages of the Tidyverse: dplyr, forcats, ggplot2, lubridate, purrr, readr, stringr, tibble, and tidyr. These are considered the core of the Tidyverse because you'll use them in almost every analysis: - dplyr : manipulating data frames
- forcats : provides tools for dealing with categorical variables
- ggplot2 : producing statistical, or data, graphics
- lubridate : makes it easier to work with dates and times in R
- purr : working with functions and iteration in a functional programming style

#| label: load-tidyverse #| warning = FALSE #| message = FALSE

```r
library(tidyverse)
```

```
Warning: package 'tidyverse' was built under R version 4.4.3
```

```
Warning: package 'ggplot2' was built under R version 4.4.3
```

```
Warning: package 'tibble' was built under R version 4.4.3
```

```
Warning: package 'tidyr' was built under R version 4.4.3
```

```
Warning: package 'readr' was built under R version 4.4.3
```

```
Warning: package 'purrr' was built under R version 4.4.3
```

```
Warning: package 'dplyr' was built under R version 4.4.3
```

```
Warning: package 'forcats' was built under R version 4.4.3
```

```
Warning: package 'lubridate' was built under R version 4.4.3
```

```
-- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
v dplyr     1.1.4     v readr     2.1.5
v forcats   1.0.0     v stringr   1.5.1
v ggplot2   3.5.2     v tibble    3.3.0
v lubridate 1.9.4     v tidyr     1.3.1
v purrr     1.0.4
-- Conflicts ------------------------------------------ tidyverse_conflicts() --
x dplyr::filter() masks stats::filter()
x dplyr::lag()    masks stats::lag()
i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to becom
```

```r
library(dslabs)
```

```
Warning: package 'dslabs' was built under R version 4.4.3
```

```r
data(murders)
```

2 Manipulating Data frames with dplyr and purrr

2.1 Tidy Data

We say that a data table is in *tidy* format if each row represents one observation and columns represent the different variables available for each of these observations. The murders dataset is an example of a tidy data frame.

```
head(murders)
```

```
       state abb region population total
1    Alabama  AL  South    4779736   135
2     Alaska  AK   West     710231    19
3    Arizona  AZ   West    6392017   232
4   Arkansas  AR  South    2915918    93
5 California  CA   West   37253956  1257
6   Colorado  CO   West    5029196    65
```

Each row represents a state with each of the five columns providing a different variable related to these states: name, abbreviation, region, population, and total murders.

2.2 Manipulating Data Frames

> "The dplyr package from the tidyverse introduces functions that perform some of the most common operations when working with data frames and uses names for these functions that are relatively easy to remember. For instance, to change the data table by adding a new column, we use mutate. To filter the data table to a subset of rows, we use filter. Finally, to subset the data by selecting specific columns, we use select."

2.2.1 The `mutate` function

The `mutate` function is used to add new columns to a data frame or modify existing ones.

```
# Add a new column 'rate' to the murders data frame
murders  <- mutate(murders, rate = total / population * 100000)
```

Note: to compute the rate, we used `total` and `population` columns, which are not defined in the global environment. The `mutate` function allows us to use the names of the columns directly.

> "This is one of dplyr's main features. Functions in this package, such as mutate, know to look for variables in the data frame provided in the first argument.
> In the call to mutate above, `total` will have the values in `murders$total`. This approach makes the code much more readable and concise."

```
head(murders)
```

```
       state abb region population total     rate
1    Alabama  AL  South    4779736   135 2.824424
2     Alaska  AK   West     710231    19 2.675186
3    Arizona  AZ   West    6392017   232 3.629527
4   Arkansas  AR  South    2915918    93 3.189390
5 California  CA   West   37253956  1257 3.374138
6   Colorado  CO   West    5029196    65 1.292453
```

Note: the `mutate` function does not change the original data frame.

> "Although we have overwritten the original **murders** object, this does not change the object that is loaded with `data(murders)`.
> If we load the murders data again, the original will overwrite our mutated version."

2.2.2 Subsetting with `filter`

The `filter` function is used to subset rows based on logical conditions.

Filter the murders data frame to include only the entries for which the murder rate is lower than 0.71.

```
# Syntax : data, conditional statement.
filter(murders, rate <= 0.71)
```

```
         state abb        region population total      rate
1       Hawaii  HI          West    1360301     7 0.5145920
2         Iowa  IA North Central    3046355    21 0.6893484
3 New Hampshire  NH     Northeast    1316470     5 0.3798036
4  North Dakota  ND North Central     672591     4 0.5947151
5       Vermont  VT     Northeast     625741     2 0.3196211
```

2.2.3 Selecting columns with `select`

The `select()` function is used to extract specific columns from a data frame.

In the example below: - We create a new data frame containing only the columns state, region, and rate. - We then apply `filter()` to keep only the rows where the murder rate is less than or equal to 0.71.

```
state_region_rate_table <- select(murders, state, region, rate)
filter(state_region_rate_table, rate <= 0.71)
```

```
         state        region      rate
1       Hawaii          West 0.5145920
2         Iowa North Central 0.6893484
3 New Hampshire     Northeast 0.3798036
4  North Dakota North Central 0.5947151
5       Vermont     Northeast 0.3196211
```

2.2.4 Exercises

1. Load the dplyr package and the murders dataset.

```
library(dplyr)
library(dslabs)
data(murders)
```

2. Use the function mutate to add a column rank containing the rank, from highest to lowest murder rate. Make sure you redefine murders so we can keep using this variable.

```
murders <- mutate(murders, rate = total / population * 10^5)
murders <- mutate(murders, rank = rank(-rate))
murders %>% head()
```

```
       state abb region population total     rate rank
1    Alabama  AL  South    4779736   135 2.824424   23
2     Alaska  AK   West     710231    19 2.675186   27
3    Arizona  AZ   West    6392017   232 3.629527   10
4   Arkansas  AR  South    2915918    93 3.189390   17
5 California  CA   West   37253956  1257 3.374138   14
6   Colorado  CO   West    5029196    65 1.292453   38
```

```r
select(murders, state, population) %>% head()
```

```
       state population
1    Alabama    4779736
2     Alaska     710231
3    Arizona    6392017
4   Arkansas    2915918
5 California   37253956
6   Colorado    5029196
```

We can write population rather than murders$population. The function mutate knows we are grabbing columns from murders.

3. Use `select` to show the state names and abbreviations in murders. Do not redefine murders, just show the results.

```r
select(murders, state, abb)
```

```
                  state abb
1               Alabama  AL
2                Alaska  AK
3               Arizona  AZ
4              Arkansas  AR
5            California  CA
6              Colorado  CO
7           Connecticut  CT
8              Delaware  DE
9  District of Columbia  DC
10              Florida  FL
11              Georgia  GA
12               Hawaii  HI
13                Idaho  ID
14             Illinois  IL
```

```
15          Indiana  IN
16             Iowa  IA
17           Kansas  KS
18         Kentucky  KY
19        Louisiana  LA
20            Maine  ME
21         Maryland  MD
22    Massachusetts  MA
23         Michigan  MI
24        Minnesota  MN
25      Mississippi  MS
26         Missouri  MO
27          Montana  MT
28         Nebraska  NE
29           Nevada  NV
30    New Hampshire  NH
31       New Jersey  NJ
32       New Mexico  NM
33         New York  NY
34   North Carolina  NC
35     North Dakota  ND
36             Ohio  OH
37         Oklahoma  OK
38           Oregon  OR
39     Pennsylvania  PA
40     Rhode Island  RI
41   South Carolina  SC
42     South Dakota  SD
43        Tennessee  TN
44            Texas  TX
45             Utah  UT
46          Vermont  VT
47         Virginia  VA
48       Washington  WA
49    West Virginia  WV
50        Wisconsin  WI
51          Wyoming  WY
```

4. Use filter to show the top 5 states with the highest murder rates.

```r
filter(murders,  rank <= 5)
```

```
                 state abb        region population total       rate rank
```

```
1 District of Columbia  DC         South    601723    99 16.452753    1
2            Louisiana  LA         South   4533372   351  7.742581    2
3             Maryland  MD         South   5773552   293  5.074866    4
4             Missouri  MO North Central   5988927   321  5.359892    3
5       South Carolina  SC         South   4625364   207  4.475323    5
```

5. Create a new data frame called no_south that removes states from the South region. How many states are in this category? You can use the function nrow for this.

Note: We can remove rows using the != operator. For example, to remove Florida, we would do this:

```r
no_florida <- filter(murders, state != "Florida")
```

```r
# Create the new data frame without south region
no_south <- filter(murders, region != "South")
# Compute how many states are not in the south
select(no_south, state) %>% nrow()
```

```
[1] 34
```

There are 34 states which are not in the south

3 data-exploration

3.1 help

Shows the help page with a description of the dataset and its variables

```
?mpg  # for quick help lookup
help(mgp) # detailed help
```

3.2 Class

```
class(mpg)
```

```
[1] "tbl_df"     "tbl"        "data.frame"
```

3.3 str

```
str(mpg)
```

```
tibble [234 x 11] (S3: tbl_df/tbl/data.frame)
 $ manufacturer: chr [1:234] "audi" "audi" "audi" "audi" ...
 $ model       : chr [1:234] "a4" "a4" "a4" "a4" ...
 $ displ       : num [1:234] 1.8 1.8 2 2 2.8 2.8 3.1 1.8 1.8 2 ...
 $ year        : int [1:234] 1999 1999 2008 2008 1999 1999 2008 1999 1999 2008 ...
 $ cyl         : int [1:234] 4 4 4 4 6 6 6 4 4 4 ...
 $ trans       : chr [1:234] "auto(l5)" "manual(m5)" "manual(m6)" "auto(av)" ...
 $ drv         : chr [1:234] "f" "f" "f" "f" ...
 $ cty         : int [1:234] 18 21 20 21 16 18 18 18 16 20 ...
 $ hwy         : int [1:234] 29 29 31 30 26 26 27 26 25 28 ...
 $ fl          : chr [1:234] "p" "p" "p" "p" ...
 $ class       : chr [1:234] "compact" "compact" "compact" "compact" ...
```

3.4 glimpse

`glimpse()` (dplyr package) provides an overview of the data set/a transposed version of the data, showing the number of observations, variable names, data types, and a sample of the data stored in each variable

```r
glimpse(mpg)
```

```
Rows: 234
Columns: 11
$ manufacturer <chr> "audi", "audi", "audi", "audi", "audi", "audi", "audi", "~
$ model        <chr> "a4", "a4", "a4", "a4", "a4", "a4", "a4", "a4 quattro", "~
$ displ        <dbl> 1.8, 1.8, 2.0, 2.0, 2.8, 2.8, 3.1, 1.8, 1.8, 2.0, 2.0, 2.~
$ year         <int> 1999, 1999, 2008, 2008, 1999, 1999, 2008, 1999, 1999, 200~
$ cyl          <int> 4, 4, 4, 4, 6, 6, 6, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 8, 8, ~
$ trans        <chr> "auto(l5)", "manual(m5)", "manual(m6)", "auto(av)", "auto~
$ drv          <chr> "f", "f", "f", "f", "f", "f", "f", "4", "4", "4", "4", "4~
$ cty          <int> 18, 21, 20, 21, 16, 18, 18, 18, 16, 20, 19, 15, 17, 17, 1~
$ hwy          <int> 29, 29, 31, 30, 26, 26, 27, 26, 25, 28, 27, 25, 25, 25, 2~
$ fl           <chr> "p", "p", "p", "p", "p", "p", "p", "p", "p", "p", "p", "p~
$ class        <chr> "compact", "compact", "compact", "compact", "compact", "c~
```

3.5 summary

```r
summary(mpg)
```

```
 manufacturer          model               displ            year     
 Length:234         Length:234         Min.   :1.600   Min.   :1999  
 Class :character   Class :character   1st Qu.:2.400   1st Qu.:1999  
 Mode  :character   Mode  :character   Median :3.300   Median :2004  
                                       Mean   :3.472   Mean   :2004  
                                       3rd Qu.:4.600   3rd Qu.:2008  
                                       Max.   :7.000   Max.   :2008  
      cyl           trans               drv                 cty       
 Min.   :4.000   Length:234         Length:234         Min.   : 9.00  
 1st Qu.:4.000   Class :character   Class :character   1st Qu.:14.00  
 Median :6.000   Mode  :character   Mode  :character   Median :17.00  
 Mean   :5.889                                         Mean   :16.86  
 3rd Qu.:8.000                                         3rd Qu.:19.00  
```

```
 Max.   :8.000                                        Max.   :35.00  
      hwy             fl               class          
 Min.   :12.00   Length:234         Length:234        
 1st Qu.:18.00   Class :character   Class :character  
 Median :24.00   Mode  :character   Mode  :character  
 Mean   :23.44                                        
 3rd Qu.:27.00                                        
 Max.   :44.00                                        
```

Part II

ggplot2: Elegant Graphics for Data Analysis

4 Elegant Graphics for Data Analysis

The following content is provided from the book ggplot2: Elegant Graphics for Data Analysis written by Hadley Wickham, Danielle Navarro, and Thomas Lin Pedersen.ggplot2-book.org

ggplot2 is one of the core packages of the tidyverse library, designed for producing statistical (data) graphics. This package is based on the Grammar of Graphics (Wilkinson, 2005), which allows users to compose graphs by combining independent components.

ggplot2 is designed to work iteratively, layer by layer.

5 Grammar of Graphics

Created by Wilkinson in 2005, the Grammar of Graphics aims to "describe the fundamental features that underlie all statistical graphics."

> "In brief, the grammar tells us that a graphic maps the data to the aesthetic attributes of geometric objects. The plot may also include statistical transformations of the data and information about the plot's coordinate system. Facetting can be used to plot for different subsets of the data. The combination of these independent components is what makes up a graphic." (Wickham, Navarro, & Pedersen, 2023)

Resources - The built-in documentation
- cheatsheets

6 Key components

The data set : “Fuel economy data”

The `mpg` dataset is introduced in Wickham, Navarro, and Pedersen [1] for early plotting examples. It includes information about the fuel economy of popular car models in 1999 and 2008, collected by the US Environmental Protection Agency.

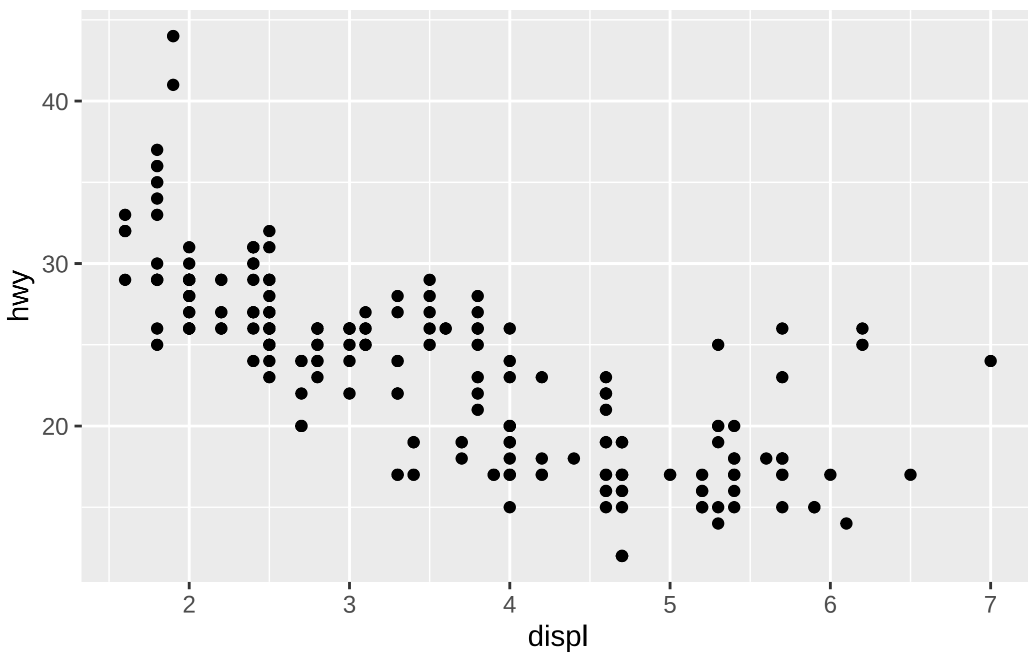
See Chapter 3 to have a look at the different ways to discover this dataset.

Three key components:

- data
- a set of aesthetic mappings between variables in the data and visual properties
- at least one layer which describes how to render each observation. Layers are usually created with a geom function.

For instance The code bellow call the data `mpg` and the aesthetic `aes()` that link :
- x to displ (engine displacement, in litres)
- y to hwy (highway miles per gallon)
Then a layer `geom_point()` is added on with `+` to create scatterplots.

```
ggplot(mpg, aes(x = displ, y = hwy)) +
    geom_point()
```



Tips >"Almost every plot maps a variable to x and y, so naming these aesthetics is tedious, so the first two unnamed arguments to aes() will be mapped to x and y. This means that the following code is identical to the example above:"[1]

```
ggplot(mpg, aes(displ,hwy)) +
    geom_point()
```

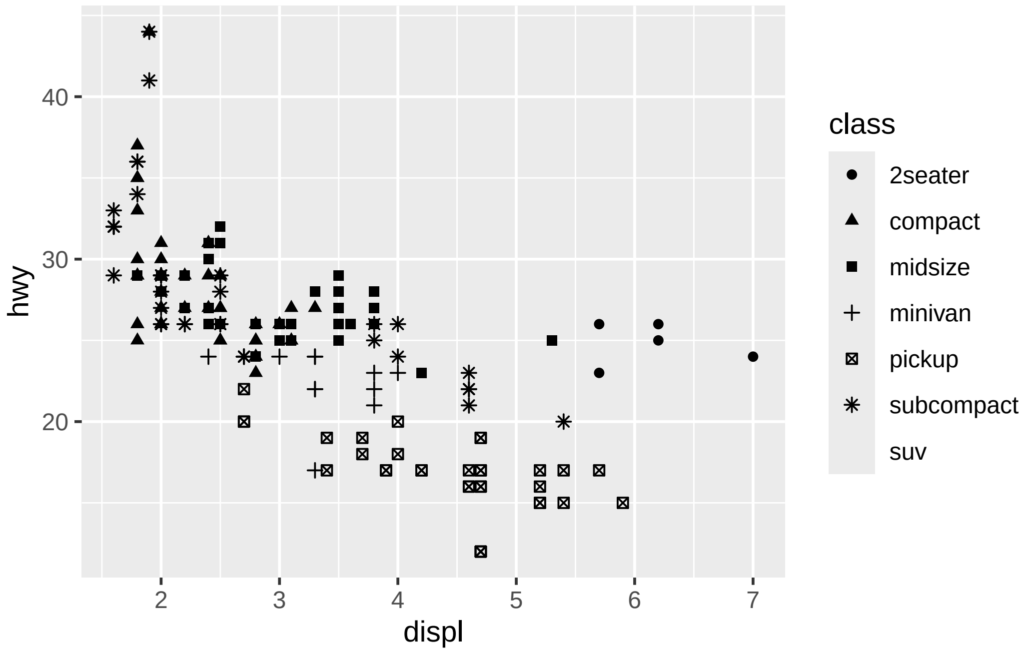
6.1 Aesthetic attributes : colour, size, shape

We can add options or parameters to the aes() functions : `aes(disply, hwy, colour = class)` : map the variable class for each (x,y) to a colour `aes(disply, hwy, size = cyl)` : geom_points size will be mapped to the cyl variable. `aes(disply, hwy, shape = drv)` : the shape aesthetic controls the symbols of points

```
ggplot(mpg, aes(displ,hwy, shape = class)) +
    geom_point()
```

```
Warning: The shape palette can deal with a maximum of 6 discrete values because more
than 6 becomes difficult to discriminate
i you have requested 7 values. Consider specifying shapes manually if you need
  that many of them.
```

```
Warning: Removed 62 rows containing missing values or values outside the scale range
(`geom_point()`).
```



To set an aesthetic to a fixed value, without scaling it, do so in the individual layer outside of aes(). Compare the following two plots:

Part III

Foundations of Statistical Analysis and Machine Learning

7

8 Mean Quadratic Error

The MQE is a measure of how close the estimator is to the true parameter value.

To compare estimator we can compute the mean quadratic Eroor, denoted by MQE :

$$\text{MQE}(\hat{\theta}_n) = \text{Var}\left(\hat{\theta}_n\right) + \left(b_\theta\left(\hat{\theta}_n\right)\right)^2$$

where $\beta_\theta(\hat{\theta}_n) = \mathbb{E}\left[\hat{\theta}_n\right] - \theta$ is the bias of the estimator $\hat{\theta}_n$.

We say that $\hat{\theta}_{n,1}$ is better than $\hat{\theta}_{n,2}$ if :

$$\forall n,\ \text{MQE}(\hat{\theta}_{n,1}) \leq \text{MQE}(\hat{\theta}_{n,2})$$

9 Example :

Let concider :
- $\hat{\theta}_{n,1} = \max(X_k)$ and $\hat{\theta}_{n,4} = \frac{n+1}{n} \cdot \hat{\theta}_{n,1}$

We have :

- $\text{MQE}(\hat{\theta}_{n,1}) = \dfrac{2\theta^2}{(n+1)(n+2)}$
- $\text{MQE}(\hat{\theta}_{n,4}) = \dfrac{\theta^2}{n(n+1)}$

$\forall n \geq 2,\ MQE(\hat{\theta}_{n,4}) < MQE(\hat{\theta}_{n,1})$

Thus, we can conclude that $\hat{\theta}_{n,4}$ is better than $\hat{\theta}_{n,1}$

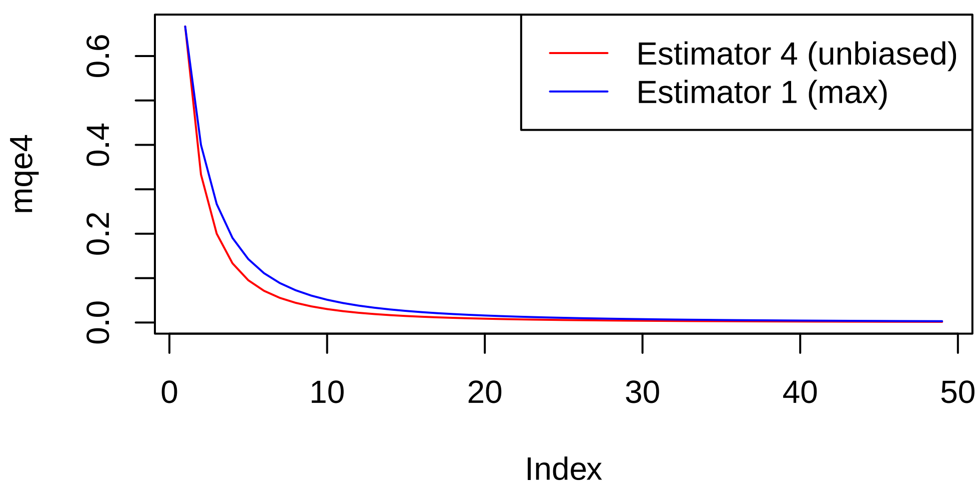
Remark: $\hat{\theta}_{n,4}$ is the best among the two estimators we have considered. Since $\hat{\theta}_{n,4}$ is unbiased, we know that for any unbiased estimator $\hat{\theta}_n$, we have:

$$\text{Cramer Rao-Bound } \leq \text{Var}\left(\hat{\theta}_n\right)$$

If $\text{Var}\left(\hat{\theta}_{n,4}\right)$ equals the Cramer-Rao bound, then the estimator cannot be improved; otherwise, improvement is possible.

10 Convergence Illustration in R

```
theta <- 2 # assumed true value of the parameter
# MQE or Variance of the estimators
mqe1 <- 2 * theta^2 / ((2:50 + 1) * (2:50 + 2))
mqe4 <- theta^2 / (2:50 * (2:50 + 1))
plot(mqe4, type = "l", col = "red")
lines(mqe1, col = "blue")
legend("topright",
       legend = c("Estimator 4 (unbiased)", "Estimator 1 (max)"),
       col = c("red", "blue"), lty = 1)
```



This plot shows that the unbiased estimator $\hat{\theta}n, 4$ consistently outperforms the maximum estimator $\hat{\theta}n, 1$ in terms of MQE, even for relatively small sample sizes (e.g., $n = 10$). However, as the sample size increases, the MQEs of both estimators get closer, meaning the performance gap narrows — although $\hat{\theta}_{n,4}$ remains superior for all n.

11 The Normal distribution

$X \sim \mathcal{N}(\mu, \sigma^2)$

PDF of a Normal distribution $\forall t \in \mathbb{R}, \quad f_X(t) = \frac{1}{\sqrt{2\pi}\sigma} e^{-(t-\mu)^2/2\sigma^2}$

11.1 rnorm()

The `rnorm` function create a vector of random numbers that follow a 'bell-shaped' distribution **Parameters**: - `n` the number of random value to generate - `mean` the center of the distribution (0 by default) - `sd` the spread of the distriburion (1 by default)

```
# normal  distribution with 50 random values, a mean of 0 and a standard deviation of 1
random_values <- rnorm(50, mean = 0, sd = 1)
head(random_values)
```

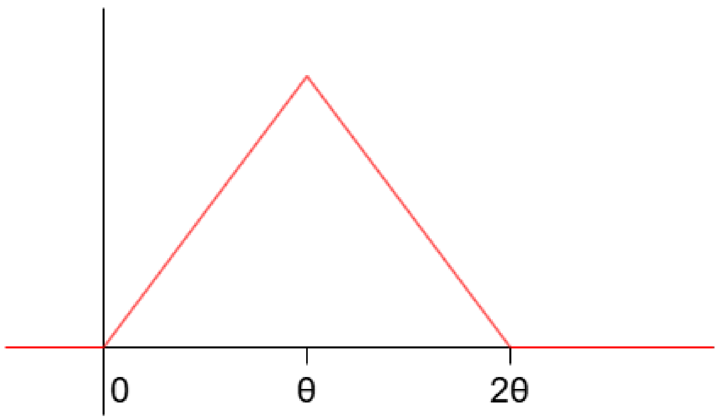
```
[1] -0.74820298 -2.08173922 -0.69746421  0.08150037 -1.32393847  0.60361818
```

Remark:

> "rnorm generates random deviates."
>
> In probability and statistics, a *random variate* (or simply *variate*) is a particular outcome or realization of a random variable.
> Other outcomes of the same random variable might yield different values — often referred to as random numbers [2].

12 Inverse Transform Sampling

From FSML2 exercice we get the following CFD from the graph below



$$F_X(t) = \begin{cases} 0 & \text{if } t < 0 \\ \frac{t^2}{2\theta^2} & \text{if } t \in [0, \theta] \\ -\frac{t^2}{2\theta^2} + \frac{2t}{2\theta} - 1 & \text{if } t \in (\theta, 2\theta) \\ 1 & \text{if } t \geq 2\theta \end{cases}$$

The computation of the inverse function, $F_X(t)^{-1}$ give us :

$$F_X^{-1} : [0, 1] \to [0, 2\theta]$$

$$F_X^{-1}(t) = \begin{cases} \sqrt{2\theta^2 \cdot t} & \text{if } t \in [0, \frac{1}{2}] \\ 2\theta - \sqrt{2\theta^2 \cdot (1 - t)} & \text{if } t \in (\frac{1}{2}, 1] \end{cases}$$

Which could be written as a sum with indicator functions as:

$$F_X^{-1}(t) = \sqrt{2\theta^2 \cdot t}\, 1_{t\in[0,\frac{1}{2}]} + 2\theta - \sqrt{2\theta^2 \cdot (1-t)}\, 1_{t\in]\frac{1}{2},1]}$$

Note: becareful to count just one time the value $tfrac12$

12.1 Inverse density function with R

Thanks to the last equation form, we can write $F_X^{-1}(t)$ in R easily :

The logical expressions like (t <= 1/2) and (t > 1/2) act as "indicator functions".
In R, TRUE is treated as 1 and FALSE as 0 in arithmetic operations. This means only the correct formula is applied for each value of t.
For example, if t = 0.3, (t <= 1/2) is TRUE (1), so the first formula is used. If t = 0.7, (t > 1/2) is TRUE (1), so the second formula is used.

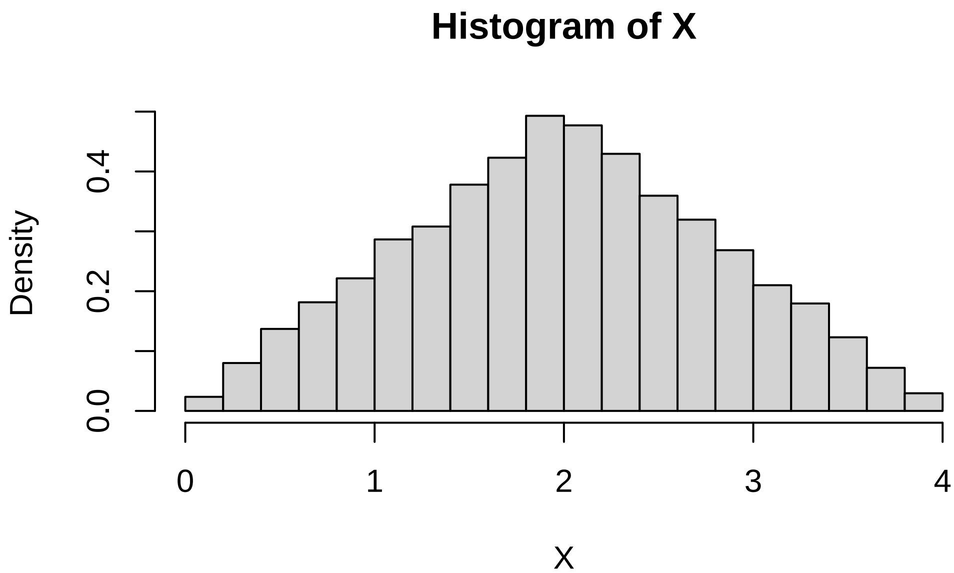
```
# Generate 10,000 random numbers uniformly distributed between 0 and 1
A <- runif(10000)

# Define the inverse transform function
invFX <- function(t, theta) {
# Logical expressions act as indicators (see explanation above)
  sqrt(2 * t * theta^2) * (0 <= t) * (t <= 1 / 2) +
    (2 * theta - sqrt(2 * theta^2 * (1 - t))) * (t > 1 / 2) * (t <= 1)
}

theta <- 2  # Set the parameter theta
X <- invFX(A, theta)   # Apply the inverse transform to the uniform random numbers
head(X)  # Display the first few values
```

```
[1] 1.9960660 1.8533371 3.3121470 0.8401302 3.2560975 2.3625927
```

Remark: In R, you do not need to use 'return()' if the value to return is the last line of the function. This is a common style in R, especially for simple functions.

12.2 Display the value of X

```r
hist(X,freq=FALSE)
```



12.3 Simulation of a density function thanks to uniform random variable

"We recognize the function f. To generate samples from a random variable X with an unknown density function, it is sufficient to know the inverse of its cumulative distribution function (i.e., $F_X^{-1}(t)$). By applying this inverse to samples from a uniform distribution, we can simulate values from X."

13 Construction of Estimators

There are two main approaches to constructing estimators in statistics:

- The Method of Moments
- Maximum Likelihood Estimation

The method of moments is a commonly used and straightforward technique in statistics. It is especially useful when you need to estimate a single parameter associated with a known distribution. The method involves equating sample moments (such as the mean or variance) to their theoretical counterparts and solving for the unknown parameters. Its main advantages are simplicity and broad applicability, making it an accessible introduction to parameter estimation. However, it can be less efficient than other methods and may not always use all the information available in the data.

Maximum likelihood estimation (MLE) is a more powerful and general approach. It involves finding the parameter values that maximize the likelihood function, i.e., the values that make the observed data most probable under the assumed model. While MLE often requires solving an optimization problem and can be more computationally intensive, it is widely used in practice due to its desirable statistical properties, such as efficiency and consistency, especially as sample size increases. MLE is particularly important in more complex models, including those with multiple parameters.

When we move to linear models, we will encounter the least squares method, which is closely related to maximum likelihood. In fact, for linear regression with normally distributed errors, the least squares estimator is also the maximum likelihood estimator. This connection highlights the central role of MLE in statistical modeling.

> **Professor's insight**
>
> "The method of moments is the most useful in practice because it is the one that almost everyone knows. Maximum likelihood, while more complex due to its optimization requirements, becomes especially valuable when we deal with models involving several parameters, such as linear models. In those cases, we will see that the least squares method and maximum likelihood are closely related and sometimes even equivalent."

14 Method of moments

> "The method of moments is fundamentally a consequence of the law of large numbers. While the law of large numbers is usually stated for the sample mean of random variables $X_1, \cdots, X_n$, it can also be applied to functions of these variables, such as their powers X_1^k. This means the law of large numbers can be generalized not just for the X_i themselves, but also for functions of X_i including their moments. This generalization forms the basis of the method of moments.

Let us consider $X_1, \cdots, X_n \quad i.i.d$ random variables whose density depends on an unknown parameter. Consider θ a function of this unknown parameter.

Remark: We specify "a function of this unknown parameter" because, in practice, we may not always be interested in estimating the parameter itself. Instead, we might be interested in estimating a function of it. For example, if $X \sim E(\lambda)$, we might be interested in estimating $\theta = 1/\lambda$

In the following methods, the k determines the moment we are using:

- $k = 1$: Use the first moment, *i.e.*, the mean $\mathbb{E}[X]$.
- $k = 2$: Use the second moment, $\mathbb{E}[X^2]$.
- $k = 3$: Use the third moment, $\mathbb{E}[X^3]$; the third **central** moment, $\mathbb{E}[(X - \mu)^3]$, is used to compute skewness. $\cdots$

14.1 Raw moments:

Let k be an integer ≥ 1 such that there exists a g function with :

$$\mathbb{E}\left[X_1^k\right] = g(\theta)$$

Note: the moment of order k is a function of the unknown parameter θ

> "In constructing estimators using the method of moments, we seek moments that depend on the unknown parameter we wish to estimate."

Then, an estimator $\hat{\theta}_n$ for θ is the solution of:

$$g(\hat{\theta}_n) = \frac{1}{n} \sum_{i=1}^{n} X_i^k$$

14.1.1 Example

Let us consider $X_1, \cdots, X_n$ *i.i.d* random variables $\sim \mathcal{U}([0;\theta])$

> "*we seek moments that depend on the unknown parameter*":

We know that since the random variables follow a uniform law defined on $[0;\theta]$, the expectation is $\dfrac{\theta}{2}$, so we know the first moment $k = 1$.

Therefore we could apply the method of moments :

$\mathbb{E}[X_1] = \dfrac{\theta}{2} = g(\theta)$

With $g(x) = \dfrac{x}{2}$ the function of the unknown parameter.

Here, we use the notation $\hat{\theta}_{n,1}$ to indicate that this estimator is based on the first moment $(k = 1)$.

$$g(\hat{\theta}_n) = \frac{1}{n}\sum_{i=1}^{n} X_i^k$$

$$g(\hat{\theta}_{n,1}) = \frac{\hat{\theta}_{n,1}}{2} = \frac{1}{n}\sum_{i=1}^{n} X_i = \overline{X}_n$$

$$\boxed{\hat{\theta}_{n,1} = 2 \cdot \overline{X}_n}$$

14.2 Centered Moment

Let k be an integer ≥ 2 such that there exists a function h satisfying:

$$\mathbb{E}\left[(X_1 - \mathbb{E}[X_1])^k\right] = h(\theta)$$

Then, an estimator $\hat{\theta}_n$ for θ is solution of :

$$h(\hat{\theta}_n) = \frac{1}{n}\sum_{i=1}^{n}(X_i - \overline{X}_n)^k$$

with

$$\overline{X}_n = \frac{1}{n}\sum_{i=1}^{n} X_i$$

Note: "The use of centered moments in method of moments is rare in practice because **centered moments depend on the expectation**, which is itself a function of the parameter and must also be estimated."

Warning: "These methods allow the construction of estimators for θ. **These estimators are not guaranteed to be unbiased**, but they are consistent under general conditions."

14.2.1 Example

Following the previous example, we could also use the centered moment of order 2, that is to say the second centered moment :

$\mathrm{Var}(X_1) = \dfrac{\theta^2}{12} = h(\theta)$

with $h(x) = \frac{x^2}{12}$

Another estimator $\hat{\theta}_{n,2}$ for θ is solution of :

$$h(\hat{\theta}_{n,2}) = \frac{1}{n}\sum_{i=1}^{n}(X_i - \overline{X}_n)^2$$

$$\frac{(\hat{\theta}_{n,2})^2}{12} = \frac{1}{n}\sum_{i=1}^{n}(X_i - \overline{X}_n)^2$$

We obtain :

$$\boxed{\hat{\theta}_{n,2} = \sqrt{\frac{12}{n}\sum_{i=1}^{n}(X_i - \overline{X}_n)^2}}$$

15 Maximum likelihood

Let us consider $X_1, \cdots, X_n$ $\;$ *i.i.d* random variables whose density depends on an unknown parameter θ .

In this context, the likelihood is defined by :

$$\mathcal{L}(x_1, \cdots, x_n;\, \theta) = \begin{cases} \displaystyle\prod_{k=1}^{n} \mathbb{P}(X_k = x_k) & \rightarrow \text{discrete case} \\ \displaystyle\prod_{k=1}^{n} f(x_k) & \rightarrow \text{continuous case} \end{cases}$$

An estimator $\hat{\theta}_n$ for θ is such that :

$$\mathcal{L}(x_1, \cdots, x_n; \hat{\theta}_n) = \max_{a \in \mathbb{R}} \mathcal{L}(x_1, \cdots, x_n; a)$$

16 Motivation for confidence interval

Let concider a Gaussian distribition with parameter 0 and 2

$$X \sim \mathcal{N}(0, 2)$$

Simulating the sample mean of a Gaussian distribution multiple times

```
M <- c() # initialize an empty vector to store the means

# Loop that will runs over three different sample sizes : 50, 500 and 5 000
for (k in c(50,500,5000))
{
    for (i in 1:50)       # we simulate the mean 50 times
    {
        A <- rnorm(k,0,sqrt(2)) # generate k observations from a normal distribution with mea
        m <- mean(A)
        M <- c(M,m)
    }
}
```

Part IV

Advanced Statistical Analysis and Machine Learning

17

Part V

Time Series Analysis

18

Part VI

Statistical Analysis of Massive and High Dimensional Data

19

20 Summary

In summary, this book has no content whatsoever.

References

[1] Hadley Wickham, Danielle Navarro, and Thomas Lin Pedersen. *ggplot2: Elegant Graphics for Data Analysis*. 2023. URL: https://ggplot2-book.org.

[2] Wikipedia contributors. *Random variate — Wikipedia, The Free Encyclopedia*. https://en.wikipedia.org/wiki/Random_variate. Accessed: 2025-07-08. 2025. URL: https://en.wikipedia.org/wiki/Random_variate.