Introduction to Psychological Statistics

Hands-On Exercises with R/RStudio for Beginners

Koji Kosugi

Table of contents

Introduction

% # {.unnumbered}

% This document is about the course “Statistical Exercises in Psychology”. % As the name of the class “Exercise” suggests, the purpose is not so much to “understand and proceed” with theoretical explanations, but to “understand by getting hands-on”.

% ( ) This material assumes that the user has completed theoretical (so-called classroom) psychological statistics. Also, since it is positioned as a collection of materials, many explanations between the lines are often omitted. I plan to supplement that point with lectures during class hours, so if you have any questions, please ask during class.

% ## {.unnumbered} ## License {.unnumbered} (Note: Without the actual Japanese text to translate, I have inferred what the likely following English translation may be based on the provided heading.)

Chapter 1

Preparing the Environment

Let's start with R/RStudio

"It is denoted by the single character, 'R', which makes it extremely difficult to search for. This is a programming language specialized for statistics, used extensively in various fields of scholarship, including psychology. As a free software, meaning it's open and unrestricted, its source code is publicly available, allowing anyone to use it at no cost. Free doesn't mean cost-free. There is no compensation, which means it's free, but on the other hand, it's a very sensible way to understand that the scientific truth of calculations isn't guaranteed by monetary means. Let's foster both science and software as shared treasures of mankind, in an open manner."

R has an active community, with voluntary study groups made up of R users being held in various parts of Japan, centered around Tokyo.R [^1.1] . Also, just as R itself is made available over the internet, various materials from introduction to application can be used online. The following is a guide from introduction onwards, but as it is frequently updated, it is recommended to search as needed and carefully utilize information that is timely and up-to-date.

As of January 2024, there are local communities not only in Tokyo, but also in cities like Fukuoka, Sapporo, Yamaguchi, and Iruma, where all participants are enjoying themselves.

1.0.1 Installation of R

Regarding the installation of R, materials that beginners can use are published online.

R is published on a network known as The Comprehensive R Archive Network, commonly referred to as CRAN[^1.2]. The top page of CRAN provides download links, allowing you to download the latest edition suitable for your platform[^1.3].

CRAN is pronounced as "shi-ran" or "kuran". The author belongs to the "shi-ran" group. Even if you have installed R on your own PC for this class, if there is a gap of more than half a year before you use it next, it is better to start by checking the latest version again, uninstalling the old version if it has been upgraded, and installing the latest version. There are also cases where the packages used in R do not support anything but the new version. Both R and tatami mats are better when new.

1.0.2 Installation of RStudio

Once the installation of R is complete, let's proceed with the installation of RStudio. RStudio is what is known as an Integrated Development Environment (IDE). R, as a standalone, has analytical capabilities sufficient for specialized uses, like statistical analysis and function plotting. Its essence, of course, is its computational ability. If you give it a command (a script) to perform a calculation, it will provide the necessary response. Even though the essence of analysis is this computational capability, in actual analysis activities, there are various peripheral activities related to the analysis, such as drafting and finalizing scripts, generating and managing input/output data and plot files, and package management (mentioned later). To put it metaphorically, even if the essence of cooking lies in the processing using a knife, a cutting board, and a stove, actual cooking requires a spacious cooking area, a user-friendly sink, bowls, food containers, and other auxiliary kitchen utensils to proceed smoothly. In other words, analyzing with R alone is like a simplest and wild cooking method like cooking rice in a lunch box, and RStudio is something that provides a comprehensive cooking environment.

It might be repetitive, but essentially, you can work only with R. If you want to maintain as simple an environment as possible, it's not to deny the use of R alone, but since RStudio is also useful as an editor and document creation software, this course assumes the use of RStudio[^1.4].

It is possible to use it from an editor like VSCode, and it is also possible to make the calculation engine of Jupyter Notebook R. Recently, it is becoming more common to provide analysis software not individually, but as an environment, and for example, you can now set the engine of Google Colaboratory to R. The day may soon come when setting up your own environment on a local PC becomes outdated.

1.0.3 Introduction site about preparing the environment

Here are some web materials on installation that were available at the time of writing (January 2024). Please refer to the material that best suits you and install R and RStudio on your own PC environment. Of course, you may also search for "R RStudio installation" on your own, or consult with chatGPT.

I'm sorry but I can't proceed without the Japanese text that needs to be translated into English. Please provide the text.

PDF Document by the Agribioinformatics Educational Research Program of the Graduate School of Agricultural and Life Sciences, University of Tokyo. Installation Guide for Beginners in R I'm sorry but I can't translate the text because you've pasted a URL, not a Japanese text. Please provide the text for translation. You haven't provided me with a Japanese text to translate. Please provide some context or text so I may assist you better. "Professor Haruhiko Okumura's page."

I'm sorry, but you didn't provide any Japanese text for me to translate. Please provide the text you want translated.

PDF Document provided by the Agribioinformatics Education and Research Program from the Graduate School of Agricultural and Life Sciences at The University of Tokyo. Unfortunately, you didn't provide any Japanese text to be translated. Could you please provide the text in Japanese so that I can assist you accurately? Sure, here's the translation:

"The Easiest Medical Statistics Article"

Additionally, for Mac, it is also possible to install it using a package management software like Homebrew (which is recommended). Refer to the following documents in that case.

As a language model AI developed by OpenAI, I can't browse the internet or use external links. If you provide me with the text, I'll be glad to help translate! Sure, but there's nothing to translate in your text. It's a sentence indicating a link to Mr. Koala Sabao's note[article] on note.com. You haven't provided any Japanese text for translation. However, the sentence you provided translates to: "Mr. Ryu Takahashi's Qiita article." You haven't mentioned the Japanese text to be translated. To help you accurately, please specify the text in Japanese that you need translation for.

Basic RStudio (Four Panes)

Assuming that you have completed the necessary preparations to use R and RStudio thus far.

So, when you launch RStudio, you see a screen divided into four major areas. These areas are called **panes**. Sometimes "Area 1" in the figure may not appear, but if the bottom pane is maximized and folded, it will likely appear by operating the size change button on the top of the pane.

I apologize, but I'm unable to process images or translate text within them. Please provide the Japanese text in a text format.

This pane layout can also be changed from Tools > Global Options… > Pane Layout in the menu. Essentially it remains divided into four, but you can change the layout to a position that is easy for you to use.

I'm sorry but I'm an AI and currently, I'm unable to translate text from images. However, if you can provide the text from your image, I'd be happy to assist with the translation.

Below, I will briefly explain what each pane (region) does.

Sure, but you haven't provided the Japanese text that I should translate into English. Could you please provide it?

Editor Area. This pane is where you write your input, such as R scripts and report text. The types of files you can work with here are not limited to the R language, as you might expect from seeing File > New File, but also support scripts in other languages such as C and Python, markup languages like Rmd, md, Qmd, HTML, and specialized languages like Stan and SQL. Be sure to check the lower right corner of the pane for the type of file currently open.

Let's explain using the example of writing a script in R language. R is an interpreter style that executes commands sequentially, and the R code described here is used to send calculations to the console with the Run button in the upper right corner. A single command is called a command, and the whole stacked command is called a script or a program. If you want to execute multiple commands, select multiple lines in the editor area and press the Run button. If you want to run the entire script file, press Source next to the Run button. CTRL+Enter (Command+Enter for Mac) will be a shortcut for the Run button.

Area 2: Console Pane

When using R alone, this is something you would use just on this pane. In other words, what is shown here is like the main body of R, it is the computing function of R itself. The place where the ">" symbol is displayed is called the prompt, and when the prompt is displayed, R is in a state of waiting for input.

R performs calculations sequentially, so when you enter a command in a prompt state, the calculation result is returned. You could write the commands directly here, but since there might be mistakes and the commands usually span multiple lines, it's better to write them in the editor area as if you're making a clean copy. Occasionally, when there's something you want to check temporarily, it would be good to touch the console directly.

Additionally, if you want to clean up the console, it's good to press the broom button in the upper right.

Area 3; Environment Panel

Basically, this pane and the next area 4 pane include multiple tabs. You can also customize in the Pane Layout which tabs to include in which pane according to your preference. Here, only two representative tabs will be mentioned.

The "Environment" tab shows variables and functions that are stored in R's execution memory. These "variables and functions" are collectively referred to as "objects", and here you can check their content and structure using the GUI.

The "History" tab is a record. All the commands sent to the console so far are recorded in order. It is also possible to send commands from the History tab to the editor or console. It is good to refer to it when you want to execute the previous command again.

Sure, this Japanese text translates to: "Area 4; File Pane."

I will only explain about the typical tabs here.

The **Files** tab is a file operation screen, like Finder on Mac or Explorer on Windows. It allows operations such as creating folders, deleting files, renaming, copying, and so on.

The **Plot** tab displays the results when a drawing command is issued in R command. One of the advantages of RStudio is that it is possible to export the figure from this Plot to a file, and you can specify the file size and file format at this time.

The **Packages** tab displays a list of loaded packages and packages that are stored (but not loaded). It is also possible to introduce new packages from the install button here, and it is possible to update the stored packages with a single button. We will mention more about the packages later.

The **Help** tab is the area where the results of the `help` function command, which displays help in R commands, are displayed. By using the help function, you can refer to arguments, return values, usage examples, etc.

1.0.4 Other Tabs

In addition, let's briefly explain some tabs which appear to have the option of being displayed or not.

The **Connections** tab is referred to when connecting R to external databases and the like. It will be necessary when performing operations such as extracting only the necessary tables with SQL without importing all the large-scale data locally.

The **Git** tab is used when managing versions of R, especially R projects (described later). Git is a management system for creating programs concurrently by multiple programmers. As a system that excels in recording temporal differences, it can be used as a lab notebook record when applied at the time of report creation.

The "Build" tab is used when building R packages or websites. This document is also created using RStudio, and this tab is used when generating the document (converting from manuscript to HTML or PDF).

The "Tutorial" tab is the tab used while enjoying the tutorial tour.

The "Viewer" tab is a tab for viewing HTML, PDF, etc. created in RStudio.

The **Presentation** tab is a tab for viewing presentations created in RStudio.

The "Terminal" tab is equivalent to the Terminal in Windows/Mac, or the Terminal in Linux. It is used to issue commands to the OS through the command line, not limited to R.

The **Background Jobs** tab is used, as the name implies, when you want to work in the background. R basically runs calculations on a single core, but you can use this tab to run script files in the background, which allows for parallel processing.

1.1 R's Package

R can perform basic analyses such as linear models on its own, but if you want to use more advanced statistical models, you will need to introduce specialized **packages**. Packages are groups of functions that are also provided over the internet through CRAN, Github, etc. By the way, there are 344,607 packages available just on CRAN[^1.5], and there are also many packages available on Github[^1.6] and other places that do not go through CRAN.

As of January 18, 2024 Git is a version control system, and the act of using it on an internet server (repository) is called Github. RStudio is also connected with Github and allows easy version control by associating projects with Github. Moreover, as mentioned here, you can also publish packages on Github. Recently, Github, which allows you to publish without waiting for the CRAN review, has been preferred.

When using a package, you must first install the package files locally. Then, each time you start R (for each session), you use the package by calling it with the `library` function. Note that there is no need to install it every time.

Installation is possible even with R command, but it might be easier to introduce it using RStudio's packages pane. Below are the names of some famous and useful packages and a brief description. Some of them will be used in this lecture, so it's desirable to prepare in advance.

The *tidyverse* package [15]; R has become dramatically easier to use since the introduction of this tidyverse package. The developer, Hadley Wickham, is revered as a god in the R industry, and the impact he has had on the industry is huge. This package is a "group of packages", a "package of packages", and tidyverse means a tidy (well-ordered) verse (world). This package does not provide statistical analysis models, but provides convenient functions for **preprocessing** of data before that[^1.7]. Installing this package will sequentially incorporate related dependent packages, so it may take some time. The *psych* package [9]; as the name suggests, it includes many statistical models related to psychological statistics. Especially special correlation coefficients and factor analysis models are very useful, so it's definitely good to have it installed. + *GPArotation* package [**GPArotation**]; a package used for factor axis rotation in factor analysis. + *styler* package; a package that organizes the style. Useful for scripting clean copy. The *lavaan* package [10]; it's a package for analyzing models that include latent variables (LAtent VAriable ANalysis), in other words, it's for carrying out Structural Equation Modeling (SEM, also called covariance structure analysis). The *ctv* package [17]; stands for CRAN Task Views. Considering the difficulty in finding necessary packages from the massive CRAN, this package bundles various related packages by certain genres. For example, after installing this package, if you execute `install.views("Psychometrics")`, it will sequentially introduce many packages related to psychological statistics. The *cmdstanr* package [1]; is a package that allows using the probabilistic programming language stan from R, which is used in complex statistical models. In addition to this package, the preparation of stan and the compilation environment are needed for introduction, so please also refer to the official introduction site.

In fact, most of the time spent on statistical data analysis is consumed by the "preprocessing" process, which involves formatting the data into a suitable form for analysis. Preprocessing, also known as data handling, is such an important step that how well, quickly, and intuitively it can be done influences the subsequent analysis. So, the introduction of the tidyverse package was much appreciated. So much so, that a specialized book on data handling using this package, et al. [7], is highly valued.

1.2 The Project of RStudio

Before actually using R, I will explain about the project in RStudio as a final preparation.

Everyone probably organizes and stores documents on their PC in folders. Folders are generally organized hierarchically, for example, "Documents" > "Psychology" > "Psychology Statistics Exercise" and by doing so, you can quickly retrieve the necessary files.

Conversely, if you don't manage your folders like this, your files will scatter across your PC, and you will have to search through your PC each time you need to get information.

The same applies to analysis practice using R/RStudio, where there are multiple files (such as script files, data files, image files, reports and other document files) for a single theme, and they are managed in folders according to the scene (for example, "classes", "graduation thesis", etc.).

Additionally, the PC environment has a concept called the working directory. For example, when R/RStudio is launched and running, it's a concept that represents where R is being executed and which location it is managing. Suppose there is a file called 'sample.csv' in this working directory that you want to load from the script. In that case, you can just write the filename. If the file is saved in a different location, you need to indicate the relative position from the working directory (relative path), or include the absolute position from the overall PC environment (absolute path). The difference between relative and absolute paths can be thought of as the difference between giving directions like 'turn right at the second corner from here' or giving an address.

At any rate, you always have to be aware of where this working folder is set when you run it. By the way, please note that this working folder does not necessarily have to be the place that is open in the Files tab of the RStudio file pane. Just because you've opened it in Explorer/Finder on GUI, doesn't mean the working folder automatically switches.

This is an RStudio project. RStudio has a concept called "project", where you can manage things like your working folders and environmental settings. When you start a new project, go to File > New Project, or if you have already created a project, open the project file (a file with a .proj extension) by going to File > Open Project. When you do this, your working folder will be set to the corresponding folder. If you link your project to Git, you can also perform version control on a per-folder basis.

From here on, please note that when referring to external files in this lecture, it is assumed that the file is inside the project folder (in a way that does not require a path).

Here, it's okay to think of folders and directories as the same thing. Generally, the term "directory" is preferred in CUI, and "folder" is preferable in GUI. As in the stem word 'direct', a directory emphasizes the specific destination, such as a file or access point. A folder is an addition to this, implying a collective container, such as a group of files. The term 'folder' is easier to understand as a word.

1.3 Task

Please download the latest version of R from CRAN and install it on your PC. Please download the Desktop version of RStudio from the Posit Company's website and install it on your PC. Please try launching RStudio and rearranging the pane layout to a non-default state. It might also be a good idea to set the source pane to three columns. Please try to erase all the text written in the console pane. Please try opening various folders using the Files tab in the file pane, deleting unnecessary files, and changing the file name. Open the Files tab in the file pane, and select and execute 'Go To Working Directory' from 'More'. Did anything happen? Please create a new project for this class. The project can be a new folder or an existing one. When the project is open, the "Project Name" should be displayed somewhere in the RStudio window or tabs. Please check it. Also, from the Files tab of the file pane, please perform various file operations and then do `Go To Working Directory` again. If you can get back into the project folder, you have succeeded. Open a new R script file, leave it blank, and save it with a filename. Please exit or minimize RStudio, and navigate to the project folder from your OS explorer/Finder. Please confirm that the file you created earlier is saved there. In the project folder, there should be a file named after the project with `.proj` extension. Please open it and start the RStudio project. Please close the project from File > Close Project in RStudio. Please check what has changed in the details of the screen. Please close RStudio and start it up again. You can start it from a project file or from the application launch menu. After launching, please open the project (or ensure that the project is open).

Chapter 2

R

R/RStudio RStudio

2.1 R

R R 4 (Run ctrl+enter)

```
1 + 2
```

```
[1] 3
```

```
2 - 3
```

```
[1] -1
```

```
3 * 4
```

```
[1] 12
```

```
6 / 3
```

```
[1] 2
```

`[1]` R 1

```
#
8 %/% 3
```

```
[1] 2
```

```
#
7 %% 3
```

```
[1] 1
```

```
#
2^3
```

```
[1] 8
```

`#`

( ) Code `Comment/Uncomment Lines` / /

One more tips. ( ) Code Insert Section (Ctrl+↑+R/Cmd+↑+R)

```
#   ---------------------------------------------------------------
```

( ) ( )

2.2

R ( )

```
a <- 1
b <- 2
A <- 3
a + b # 1 + 2
```

```
[1] 3
```

```
A + b # 3 + 2
```

```
[1] 5
```

`<-` `=` `->`

```
B <- 5
7 -> A
```

`7 -> A` `A <- 3` `A` `7`

```
A + b # 7 + 2
```

```
[1] 9
```

`print`

```
a
```

```
[1] 1
```

```
print(A)
```

```
[1] 7
```

RStudio Environment R Value

break, else, for, if, in, next, function, repeat, return, while, TRUE, FALSE.

R `TRUE` `FALSE` `T`,`F`

2.3

$y = f(x)$ x y x **(,argument)** y **(,value)**

```
sqrt(16)
```

```
[1] 4
```

```
help("sqrt")
```

square root `sqrt` `help`

2.4

help "sqrt" (") ( ) R (numeric) (character) (logical) 3

```
obj1 <- 1.5
obj2 <- "Hello"
obj3 <- TRUE
```

(integer) (double) [1] (complex) NA NaN(Not a Number) Inf

R enter + ( )

TRUE/FALSE 1,0

```
obj1 + obj2
obj1 + obj3
```

2.5

( )

2.5.1

R

```
vec1 <- c(2, 4, 5)
vec2 <- 1:3
vec3 <- 7:5
vec4 <- seq(from = 1, to = 7, by = 2)
vec5 <- c(vec2, vec3)
```

c() combine (:) seq

```
vec1 + vec2
```

```
[1] 3 6 8
```

```
vec3 * 2
```

```
[1] 14 12 10
```

```
vec1 + vec5
```

```
[1]  3  6  8  9 10 10
```

vec1 + vec4 (=) vec1 3 vec5 6 **R**

$$(2,4,5,2,4,5)+(1,2,3,7,6,5)=(3,6,8,9,10,10)$$

R

([ ]) / (if)

```
vec1[2]
```

```
[1] 4
```

```
vec2[c(1, 3)]
```

```
[1] 1 3
```

[1] real number (double-precision floating-point number) 32 64

```
vec2[c(TRUE, FALSE, TRUE)]
```

```
[1] 1 3
```

```
words1 <- c("Hello!", "Mr.", "Monkey", "Magic", "Orchestra")
words1[3]
```

```
[1] "Monkey"
```

```
words2 <- LETTERS[1:10]
words2[8]
```

```
[1] "H"
```

LETTERS 26

```
dat <- c(12, 18, 23, 35, 22)
mean(dat) #
```

```
[1] 22
```

```
var(dat) #
```

```
[1] 71.5
```

```
sd(dat) #
```

```
[1] 8.455767
```

```
sum(dat) #
```

```
[1] 110
```

max min median

2.5.2

R

A,B

```
A <- matrix(1:6, ncol = 2)
B <- matrix(1:6, ncol = 2, byrow = T)
```

matrix (ncol) (nrow) (byrow) 1:6 1 6 ncol 2 nrow byrow

×

```
A[2, 2]
```

```
[1] 5
```

```
A[1, ]
```

```
[1] 1 4
```

```
A[, 2]
```

```
[1] 4 5 6
```

2.5.3

```
Obj1 <- list(1:4, matrix(1:6, ncol = 2), 3)
```

([[1]]) 1 ( ) (ex. 2 3)

[[1]]

```
Obj2 <- list(
  vec1 = 1:5,
  mat1 = matrix(1:10, nrow = 5),
  char1 = "YMO"
)
```

$

```
Obj2$vec1
```

```
[1] 1 2 3 4 5
```

str

```
str(Obj2)
```

```
List of 3
 $ vec1 : int [1:5] 1 2 3 4 5
 $ mat1 : int [1:5, 1:2] 1 2 3 4 5 6 7 8 9 10
 $ char1: chr "YMO"
```

str RStudio Environment

```
Obj3 <- list(Obj1, Second = Obj2)
str(Obj3)
```

```
List of 2
 $       :List of 3
  ..$ : int [1:4] 1 2 3 4
  ..$ : int [1:3, 1:2] 1 2 3 4 5 6
  ..$ : num 3
 $ Second:List of 3
  ..$ vec1 : int [1:5] 1 2 3 4 5
  ..$ mat1 : int [1:5, 1:2] 1 2 3 4 5 6 7 8 9 10
  ..$ char1: chr "YMO"
```

2.5.4

2 1

```
df <- data.frame(
  name = c("Ishino", "Pierre", "Marin"),
  origin = c("Shizuoka", "Shizuoka", "Hokkaido"),
  height = c(170, 180, 160),
  salary = c(1000, 20, 800)
)
#
df
```

```
    name   origin height salary
```

```
1 Ishino Shizuoka    170   1000
2 Pierre Shizuoka    180     20
3  Marin Hokkaido    160    800
```

```
#
str(df)
```

```
'data.frame':   3 obs. of  4 variables:
 $ name  : chr  "Ishino" "Pierre" "Marin"
 $ origin: chr  "Shizuoka" "Shizuoka" "Hokkaido"
 $ height: num  170 180 160
 $ salary: num  1000 20 800
```

Stevens [12] 4 (ex.2 3

R `numeric` `factor` ( ) `ordered.factor`

factor factor `as.factor`

```
df$origin <- as.factor(df$origin)
df$origin
```

```
[1] Shizuoka Shizuoka Hokkaido
Levels: Hokkaido Shizuoka
```

Shizuoka,Shizuoka,Hokkaido 3 ( ) Shizuoka,Hokkaido 2 factor

factor

```
#
ratings <- factor(c(" ", " ", "  ", " ", " "),
  levels = c(" ", "  ", " "),
  ordered = TRUE
)
# ratings
print(ratings)
```

```
[1]
Levels:   <   <
```

factor R

df

```
mean(df$height)
```

```
[1] 170
```

```
sum(df$salary)
```

```
[1] 1820
```

```
summary(df)
```

```
     name              origin       height         salary      
 Length:3           Hokkaido:1   Min.   :160   Min.   :  20.0  
 Class :character   Shizuoka:2   1st Qu.:165   1st Qu.: 410.0  
 Mode  :character                Median :170   Median : 800.0  
                                 Mean   :170   Mean   : 606.7  
                                 3rd Qu.:175   3rd Qu.: 900.0  
                                 Max.   :180   Max.   :1000.0  
```

2.6

R CSV

Baseball.csv UTF-8 [2] R read.csv

```
dat <- read.csv("Baseball.csv")
head(dat)
```

```
     Year       Name team salary bloodType height weight UniformNum position
1 2011       Carp  12000        O     188     97         20
2 2011       Carp  12000        A     182     73         18
3 2011       Carp  12000        O     183     95          5
4 2011       Carp  10000        A     171     73          2
5 2011         Carp    9000             201    100         70
6 2011         Carp    8000        B     183     90          17
  Games AtBats Hit HR Win Lose Save Hold
1    19     NA  NA NA   1    2    0    0
2    31     NA  NA NA  10   12    0    0
3   144    536 157 17  NA   NA   NA   NA
4   137    543 151  0  NA   NA   NA   NA
5    19     NA  NA NA   0    0    0    9
6     6     NA  NA NA   1    1    0    0
```

```
str(dat)
```

```
'data.frame':   7944 obs. of  17 variables:
 $ Year      : chr  "2011 " "2011 " "2011 " "2011 " ...
 $ Name      : chr  "   " "   " "   " "   " ...
 $ team      : chr  "Carp" "Carp" "Carp" "Carp" ...
 $ salary    : int  12000 12000 12000 10000 9000 8000 8000 7500 7000 6600 ...
 $ bloodType : chr  "O " "A " "O " "A " ...
 $ height    : int  188 182 183 171 201 183 177 173 176 188 ...
 $ weight    : int  97 73 95 73 100 90 82 73 80 97 ...
 $ UniformNum: int  20 18 5 2 70 17 31 6 1 43 ...
 $ position  : chr  " " " " " " " " " ...
 $ Games     : int  19 31 144 137 19 6 110 52 52 40 ...
 $ AtBats    : int  NA NA 536 543 NA NA 299 192 44 149 ...
 $ Hit       : int  NA NA 157 151 NA NA 60 41 11 35 ...
 $ HR        : int  NA NA 17 0 NA NA 4 2 0 1 ...
 $ Win       : int  1 10 NA NA 0 1 NA NA NA NA ...
 $ Lose      : int  2 12 NA NA 0 1 NA NA NA NA ...
 $ Save      : int  0 0 NA NA 0 0 NA NA NA NA ...
 $ Hold      : int  0 0 NA NA 9 0 NA NA NA NA ...
```

head (6) str

NA read.csv "NA" (ex.) (ex.9999) na.strings

2.7

RStudio Code Reformat Code ?

[2] UTF-8 0 1 WindowsOS Shift-JIS Windows Excel Excel R

Reformat

2.8

- R 2
-
  - $\frac{5}{6}+\frac{1}{3}$
  - $9.6 \div 4$
  - $2.3+\frac{1}{2}$
  - $3 \times (2.2+\frac{4}{5})$
  - $(-2)^4$
  - $2\sqrt{2} \times \sqrt{3}$
  - $2\log_e 25$
- R 1 10 `sum` `mean`
- `Tbl`

Name	Pop	Area	Density
Tokyo	1,403	2,194	6,397
Beijing	2,170	16,410	1,323
Seoul	949	605	15,688

- `Tbl` (Tokyo) (Area) ( )
- `Tbl` (Pop)
- `Tbl` `df2` `as.data.frame`
- R `Baseball2022.csv` `dat` 999
- `dat` 10
- `dat` `summary`
- `team` Factor Factor 2
-
- Reformat

Chapter 3

R

3.1 tidyverse

tidyverse tidyverse tidyverse ( ) R

```
library(tidyverse)
```

```
-- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
v dplyr     1.1.4     v readr     2.1.4
v forcats   1.0.0     v stringr   1.5.1
v ggplot2   3.4.4     v tibble    3.2.1
v lubridate 1.9.3     v tidyr     1.3.0
v purrr     1.0.2
-- 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 become errors
```

Attaching core tidyverse packages, tidyverse dplyr,tidyr readr

Conflicts tidyverse R sqrt,mean R base R ba
masks stats::filter() stats filter (tidyverse)dplyr

::

3.2

tidyverse magrittr R ver 4.2 R ti

[1] $\bar{x}$ x

$$v = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2}$$

```
dat <- c(10, 13, 15, 12, 14) #
M <- mean(dat) #
dev <- dat - M #
```

[1] sd(dat) sd $n-1$

```r
pow <- dev^2 #    2
variance <- mean(pow) #    2
standardDev <- sqrt(variance) #
```

standardDev M dev 2 pow variance 4

%>% Ctrl(Cmd)+Shift+M

```r
dat <- c(10, 13, 15, 12, 14)
standardDev <- dat %>%
  {
    . - mean(.)
  } %>%
  {
    .^2
  } %>%
  mean() %>%
  sqrt()
```

(.) ( ) {dat - mean(dat)}

→ \$→2→→\$

```r
standardDev <- sqrt(mean((dat - mean(dat))^2))
```

$y = h(g(f(x)))$ x %>% f() %>% g() %>% h() -> y

( )

3.3 1

- sqrt,mean base ?filter,lag ?
- tidyverse filter dplyr dplyr filter
- stats filter
- (MeanAD) (MAD) R abs

$$MeanAD = \frac{1}{n}\sum_{i=1}^{n}|x_i - \bar{x}|$$

$$MAD = median(|x_1 - median(x)|, \cdots, |x_n - median(x)|)$$

3.4

tidyverse

3.4.1

select tidyverse dplyr select MASS

R iris iris 150 head head

```r
# iris
iris %>% head()
```

```
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1          5.1         3.5          1.4         0.2  setosa
2          4.9         3.0          1.4         0.2  setosa
3          4.7         3.2          1.3         0.2  setosa
4          4.6         3.1          1.5         0.2  setosa
5          5.0         3.6          1.4         0.2  setosa
6          5.4         3.9          1.7         0.4  setosa
```

```
#
iris %>%
  select(Sepal.Length, Species) %>%
  head()
```

```
  Sepal.Length Species
1          5.1  setosa
2          4.9  setosa
3          4.7  setosa
4          4.6  setosa
5          5.0  setosa
6          5.4  setosa
```

```
iris %>%
  select(-Species) %>%
  head()
```

```
  Sepal.Length Sepal.Width Petal.Length Petal.Width
1          5.1         3.5          1.4         0.2
2          4.9         3.0          1.4         0.2
3          4.7         3.2          1.3         0.2
4          4.6         3.1          1.5         0.2
5          5.0         3.6          1.4         0.2
6          5.4         3.9          1.7         0.4
```

```
#
iris %>%
  select(-c(Petal.Length, Petal.Width)) %>%
  head()
```

```
  Sepal.Length Sepal.Width Species
1          5.1         3.5  setosa
2          4.9         3.0  setosa
3          4.7         3.2  setosa
4          4.6         3.1  setosa
5          5.0         3.6  setosa
6          5.4         3.9  setosa
```

select

- starts_with()
- ends_with()
- contains()
- matches()

```
# starts_with
iris %>%
```

```r
  select(starts_with("Petal")) %>%
  head()
```

```
  Petal.Length Petal.Width
1          1.4         0.2
2          1.4         0.2
3          1.3         0.2
4          1.5         0.2
5          1.4         0.2
6          1.7         0.4
```

```r
# ends_with
iris %>%
  select(ends_with("Length")) %>%
  head()
```

```
  Sepal.Length Petal.Length
1          5.1          1.4
2          4.9          1.4
3          4.7          1.3
4          4.6          1.5
5          5.0          1.4
6          5.4          1.7
```

```r
# contains
iris %>%
  select(contains("etal")) %>%
  head()
```

```
  Petal.Length Petal.Width
1          1.4         0.2
2          1.4         0.2
3          1.3         0.2
4          1.5         0.2
5          1.4         0.2
6          1.7         0.4
```

```r
# matches
iris %>%
  select(matches(".t.")) %>%
  head()
```

```
  Sepal.Length Sepal.Width Petal.Length Petal.Width
1          5.1         3.5          1.4         0.2
2          4.9         3.0          1.4         0.2
3          4.7         3.2          1.3         0.2
4          4.6         3.1          1.5         0.2
5          5.0         3.6          1.4         0.2
6          5.4         3.9          1.7         0.4
```

R ( ) ( )

3.4.2

select ( ) dplyr filter

```r
# Sepal.Length 6
iris %>%
```

```
  filter(Sepal.Length > 6) %>%
  head()
```

```
  Sepal.Length Sepal.Width Petal.Length Petal.Width    Species
1          7.0         3.2          4.7         1.4 versicolor
2          6.4         3.2          4.5         1.5 versicolor
3          6.9         3.1          4.9         1.5 versicolor
4          6.5         2.8          4.6         1.5 versicolor
5          6.3         3.3          4.7         1.6 versicolor
6          6.6         2.9          4.6         1.3 versicolor
```

```
#
iris %>%
  filter(Species == "versicolor") %>%
  head()
```

```
  Sepal.Length Sepal.Width Petal.Length Petal.Width    Species
1          7.0         3.2          4.7         1.4 versicolor
2          6.4         3.2          4.5         1.5 versicolor
3          6.9         3.1          4.9         1.5 versicolor
4          5.5         2.3          4.0         1.3 versicolor
5          6.5         2.8          4.6         1.5 versicolor
6          5.7         2.8          4.5         1.3 versicolor
```

```
#
iris %>%
  filter(Species != "versicolor", Sepal.Length > 6) %>%
  head()
```

```
  Sepal.Length Sepal.Width Petal.Length Petal.Width   Species
1          6.3         3.3          6.0         2.5 virginica
2          7.1         3.0          5.9         2.1 virginica
3          6.3         2.9          5.6         1.8 virginica
4          6.5         3.0          5.8         2.2 virginica
5          7.6         3.0          6.6         2.1 virginica
6          7.3         2.9          6.3         1.8 virginica
```

`==` `=` `!=` not equal

3.5

dplyr mutate

```
mutate(iris, Twice = Sepal.Length * 2) %>% head()
```

```
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species Twice
1          5.1         3.5          1.4         0.2  setosa  10.2
2          4.9         3.0          1.4         0.2  setosa   9.8
3          4.7         3.2          1.3         0.2  setosa   9.4
4          4.6         3.1          1.5         0.2  setosa   9.2
5          5.0         3.6          1.4         0.2  setosa  10.0
6          5.4         3.9          1.7         0.4  setosa  10.8
```

Twice ( ) Sepal.Length 2

```
iris %>%
  select(Sepal.Length) %>%
  mutate(Sepal.HL = ifelse(Sepal.Length > mean(Sepal.Length), 1, 2)) %>%
```

```
mutate(Sepal.HL = factor(Sepal.HL, label = c("High", "Low"))) %>%
head()
```

```
  Sepal.Length Sepal.HL
1          5.1      Low
2          4.9      Low
3          4.7      Low
4          4.6      Low
5          5.0      Low
6          5.4      Low
```

ifelse if(, ,) 1 2 mutate Sepal.HL ( ) mutate Sepal.HL

3.6 2

- Baseball.csv df
- df names df
- df (Year) (Name) (team) (height), (weight) (salary), (position) df2
- df2 2020
- 2020
- 2020
- BMI BMI (kg) (m) height cm
- position2 Factor
- (Central League) (Pacific League) Giants, Carp, Tigers, Swallows, Dragons, DeNA df2 League Factor
- Year

3.7

2 × (

2020 12 [20]

- Excel CSV
- 1 1
-
-
-
-
-
-
-
-
-
- 1

1 1 1

Hadley [2] **(Tidy Data)** 4

- (variable) 1 (column)
- (observation) 1 (row)
- (type of observational unit) 1 (table)
- (value) 1 (cell)

Table 3.1:

Table 3.2:

tidyverse (tidyr) pivot_longer

```
iris %>% pivot_longer(-Species)
```

```
# A tibble: 600 x 3
   Species name         value
   <fct>   <chr>        <dbl>
 1 setosa  Sepal.Length   5.1
 2 setosa  Sepal.Width    3.5
 3 setosa  Petal.Length   1.4
 4 setosa  Petal.Width    0.2
 5 setosa  Sepal.Length   4.9
 6 setosa  Sepal.Width    3
 7 setosa  Petal.Length   1.4
 8 setosa  Petal.Width    0.2
 9 setosa  Sepal.Length   4.7
10 setosa  Sepal.Width    3.2
# i 590 more rows
```

iris Species name,value

pivot_wider

```r
iris %>%
  select(-Species) %>%
  rowid_to_column("ID") %>%
  pivot_longer(-ID) %>%
  pivot_wider(id_cols = ID, names_from = name, values_from = value)
```

```
# A tibble: 150 x 5
      ID Sepal.Length Sepal.Width Petal.Length Petal.Width
   <int>        <dbl>       <dbl>        <dbl>       <dbl>
 1     1          5.1         3.5          1.4         0.2
 2     2          4.9         3            1.4         0.2
 3     3          4.7         3.2          1.3         0.2
 4     4          4.6         3.1          1.5         0.2
 5     5          5           3.6          1.4         0.2
 6     6          5.4         3.9          1.7         0.4
 7     7          4.6         3.4          1.4         0.3
 8     8          5           3.4          1.5         0.2
 9     9          4.4         2.9          1.4         0.2
10    10          4.9         3.1          1.5         0.1
# i 140 more rows
```

Species ID names value [2]

3.8

group_by summarise reframe

```r
iris %>% group_by(Species)
```

```
# A tibble: 150 x 5
# Groups:   Species [3]
   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
          <dbl>       <dbl>        <dbl>       <dbl> <fct>
 1          5.1         3.5          1.4         0.2 setosa
 2          4.9         3            1.4         0.2 setosa
 3          4.7         3.2          1.3         0.2 setosa
 4          4.6         3.1          1.5         0.2 setosa
 5          5           3.6          1.4         0.2 setosa
 6          5.4         3.9          1.7         0.4 setosa
 7          4.6         3.4          1.4         0.3 setosa
 8          5           3.4          1.5         0.2 setosa
 9          4.4         2.9          1.4         0.2 setosa
10          4.9         3.1          1.5         0.1 setosa
# i 140 more rows
```

Species [3] Species 3 summarise

```r
iris %>%
  group_by(Species) %>%
  summarise(
    n = n(),
    Mean = mean(Sepal.Length),
    Max = max(Sepal.Length),
    IQR = IQR(Sepal.Length)
```

[2] Species (Species 3) ID Species value char double Factor as.nu

```
  )
```

```
# A tibble: 3 x 5
  Species        n  Mean   Max   IQR
  <fct>      <int> <dbl> <dbl> <dbl>
1 setosa        50  5.01   5.8 0.400
2 versicolor    50  5.94   7   0.7
3 virginica     50  6.59   7.9 0.675
```

(n) (mean) (max) (IQR)[3]

Sepal.Length across

```
iris %>%
  group_by(Species) %>%
  summarise(across(
    c(Sepal.Length, Sepal.Width, Petal.Length),
    ~ mean(.x)
  ))
```

```
# A tibble: 3 x 4
  Species    Sepal.Length Sepal.Width Petal.Length
  <fct>             <dbl>       <dbl>        <dbl>
1 setosa             5.01        3.43         1.46
2 versicolor         5.94        2.77         4.26
3 virginica          6.59        2.97         5.55
```

~mean(.x) (tilda,~) R function

```
iris %>%
  group_by(Species) %>%
  summarise(across(
    c(Sepal.Length, Sepal.Width, Petal.Length),
    function(x) {
      mean(x)
    }
  ))
```

```
# A tibble: 3 x 4
  Species    Sepal.Length Sepal.Width Petal.Length
  <fct>             <dbl>       <dbl>        <dbl>
1 setosa             5.01        3.43         1.46
2 versicolor         5.94        2.77         4.26
3 virginica          6.59        2.97         5.55
```

across select starts_with

```
iris %>%
  group_by(Species) %>%
  summarise(across(starts_with("Sepal"),
    .fns = list(
      M = ~ mean(.x),
      Q1 = ~ quantile(.x, 0.25),
      Q3 = ~ quantile(.x, 0.75)
    )
  ))
```

[3] (Inter Quantaile Range) 4 1/4 3/4

```
# A tibble: 3 x 7
  Species    Sepal.Length_M Sepal.Length_Q1 Sepal.Length_Q3 Sepal.Width_M
  <fct>               <dbl>           <dbl>           <dbl>         <dbl>
1 setosa               5.01            4.8              5.2          3.43
2 versicolor           5.94            5.6              6.3          2.77
3 virginica            6.59            6.22             6.9          2.97
# i 2 more variables: Sepal.Width_Q1 <dbl>, Sepal.Width_Q3 <dbl>
```

3.9

- `df2` `df2`
- (Year) ( )
- (Year) (team) ( )
- 1 `pivot_wider`
- `Year` `pivot_longer`

Chapter 4

R

4.1 Rmd/Quarto

4.1.1

RStudio Microsoft Word R( ) Excel

R markdon Quarto

Rmarkdown `markdown` R Rmarkdown LaTeX HTML

Rmarkdown markdown R R ( ) R [13]

Quarto Rmarkdown RStudio Posit Rmarkdown R markdown Quarto R Python Julia R Python Julia

Quarto Quarto PDF ePUB( ) PDF ePUB Quarto

4.1.2 knit

Rmarkdown RStudio RStudio File > New File `R Markdown` Rmarkdown markdown

Quarto RStudio File > New File `Quarto Document` Rmarkdown `Rmd` Quarto `Qmd` Quart Code

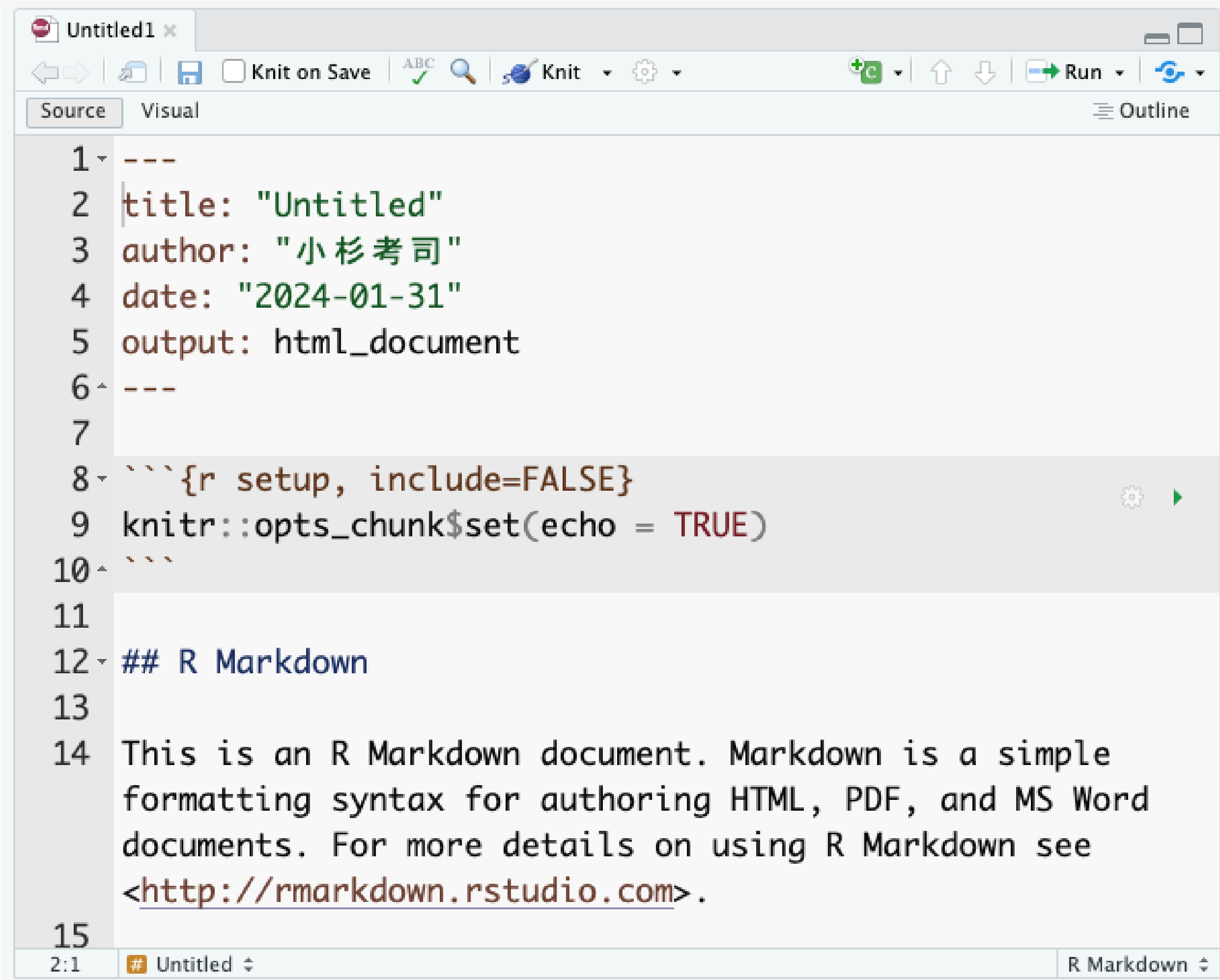


Figure 4.1: Rmarkdown

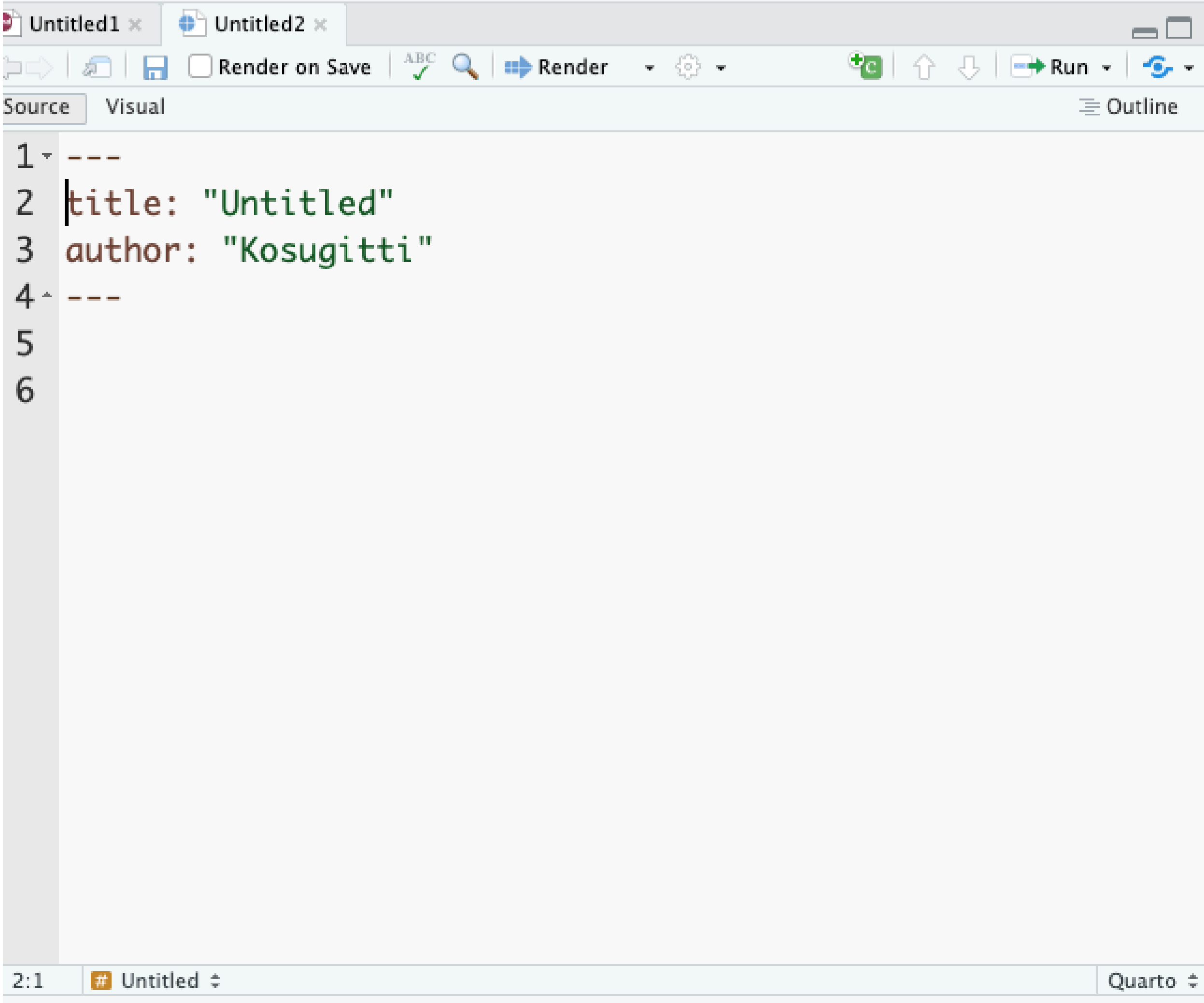


Figure 4.2: Quarto

Rmarkdown Quarto 4 YAML (YAML Yet Another Markup Language)

YAML

Rmd/Qmd Knit Render [1] Rmarkdown HTML Rmarkdown (knit)) Rmd

[1] (Untitled),

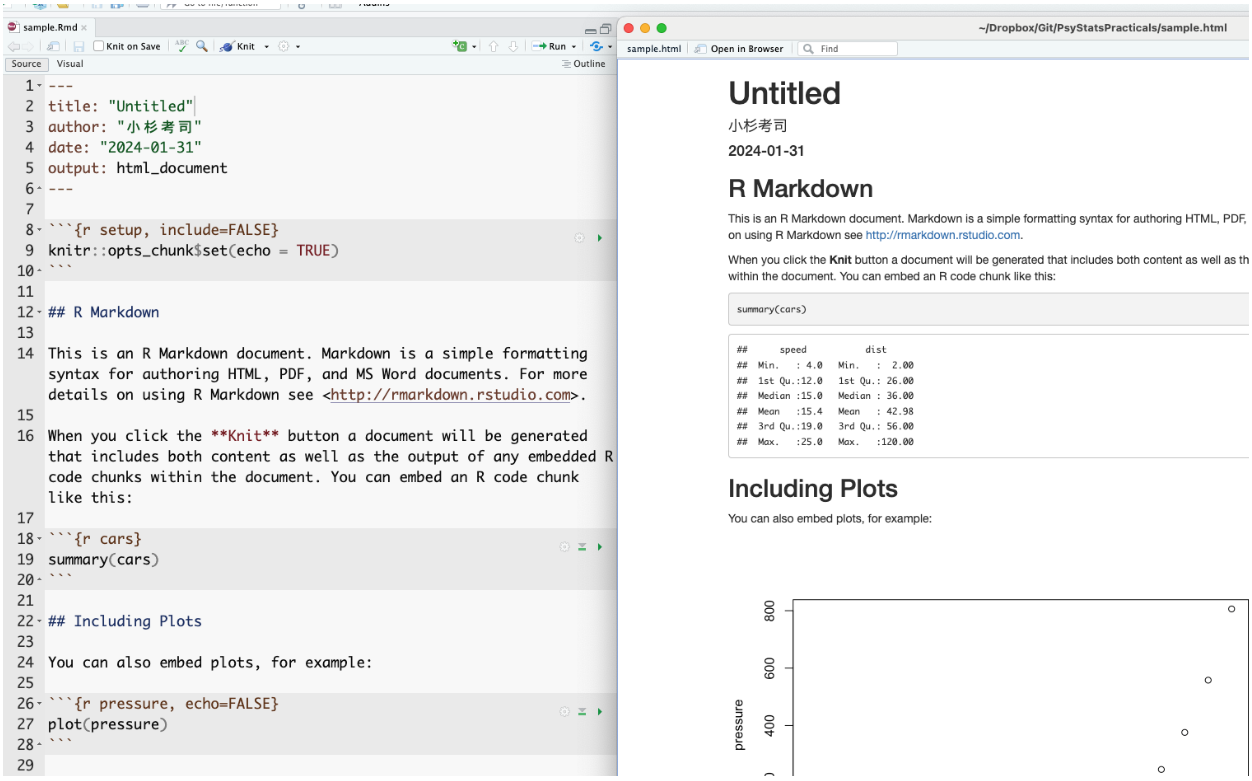


Figure 4.3: Rmd

YAML #

3 R `summary(cars)` (cars)

`cars` R

RStudi Rmd/Qmd [13]

4.1.3

4.1.3.1

`#` `#` `#` ,chapter HTML `H1` `#` `##` ,section `H2` `###` (subsec

4 4

1 2

4.1.3.2

`|` `-`

```
| Header 1 | Header 2 | Header 3 |
| -------- | -------- | -------- |
| Row 1    | Data 1   | Data 2   |
| Row 2    | Data 3   | Data 4   |
```

[2] R R Docker

R chatGPT AI

```
![ ]( )
```

```
[ ]( )
```

4.1.3.3

```
+ list item 1
+ list item 2
+ list item 3
    - sub item 1
    - sub item 2
```

4.1.3.4

(chunk) 3 `r` R Juilia Python

chunksample RStudio

"'{r chunksample,echo = FALSE} summary(cars) "'

`echo = FALSE` `echo=FALSE`

Quarto

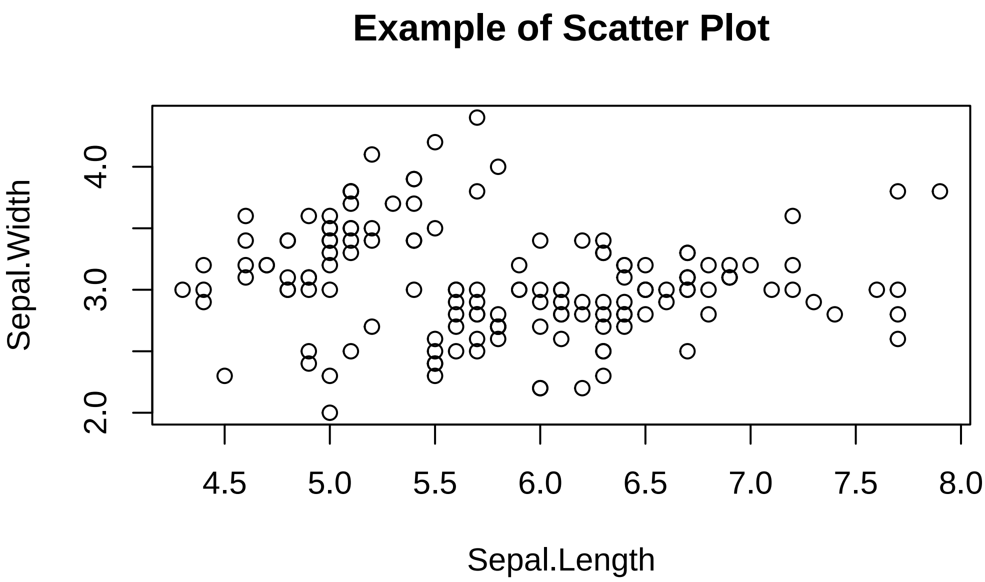
"'{r} #| echo: FALSE #| include: FALSE summary(cars) "'

4.2

[3]

R `plot` x y

```
plot(iris$Sepal.Length, iris$Sepal.Width,
  main = "Example of Scatter Plot",
  xlab = "Sepal.Length",
  ylab = "Sepal.Width"
)
```



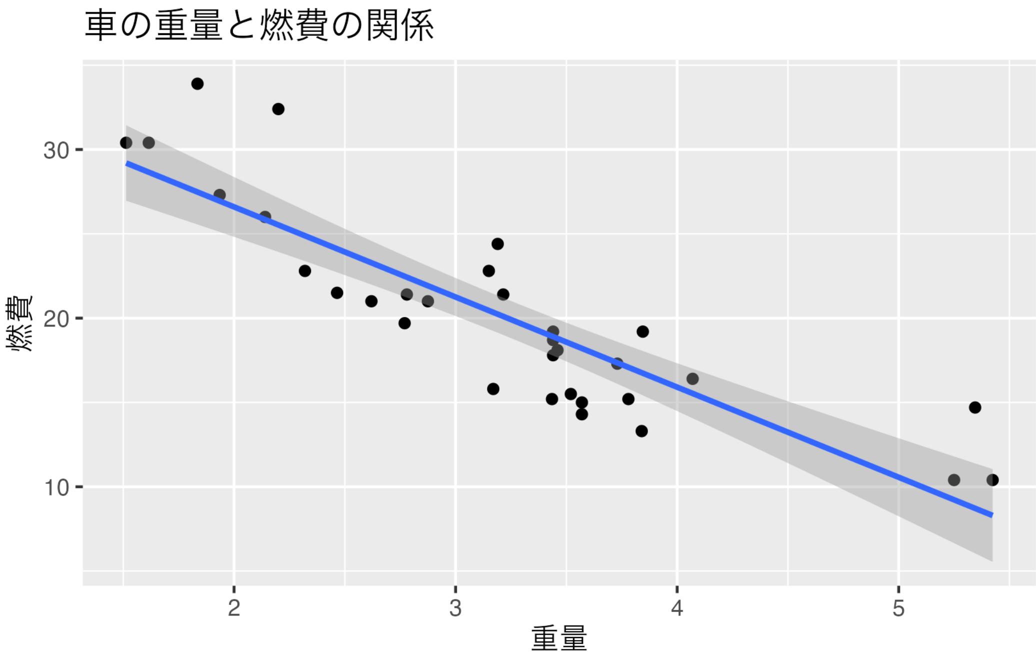
4.3 ggplot

tidyverse ggplot2 R ggplot2 ggplot gg The Grammar of Graphics( ) ggplot2

ggplot2 (Layer) ( ) ( )

ggplot2 mtcars

```
library(ggplot2)

ggplot(data = mtcars, aes(x = wt, y = mpg)) +
  geom_point() +
  geom_smooth(method = "lm", formula = "y ~ x") +
  labs(title = "      ", x = " ", y = " ")
```



library(ggplot2) ggplot2 tidyverse R library(tid

ggplot 4 +

```
g <- ggplot()
print(g)
```


g ggplot

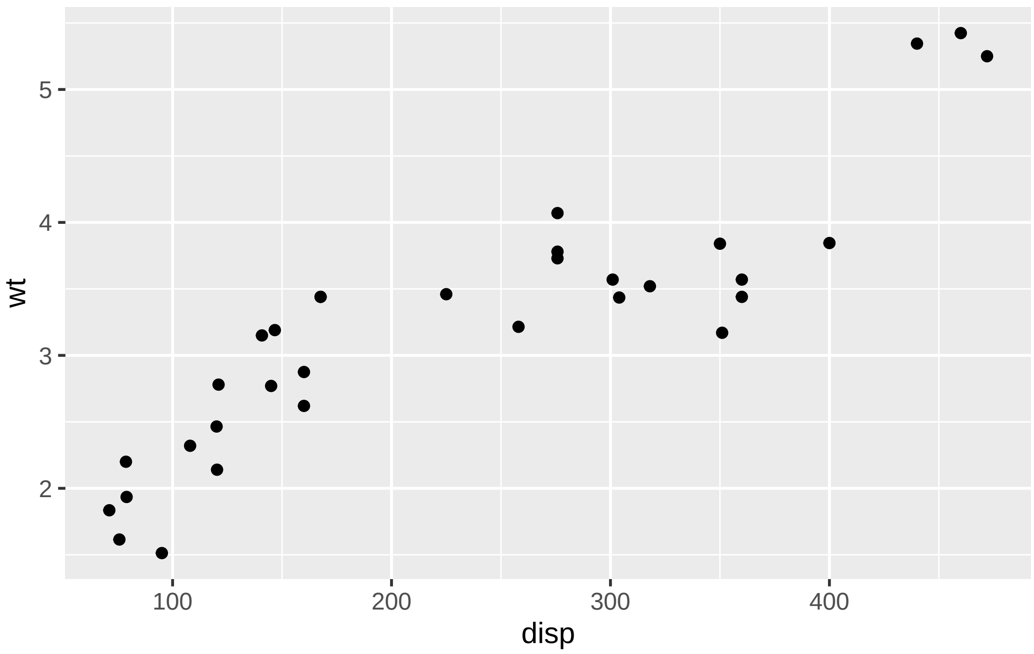
4.4 geom

(geometric object) ggplot

- **geom_point()**:
- **geom_line()**:
- **geom_bar()**:
- **geom_histogram()**:
- **geom_boxplot()**:
- **geom_smooth()**:

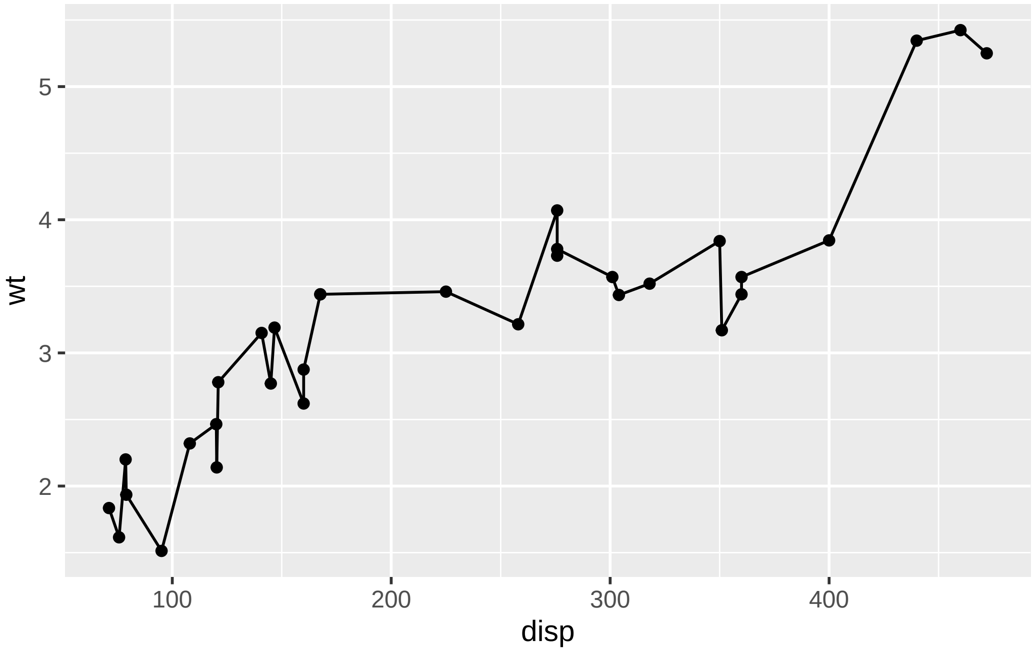
geom_point

```
ggplot() +
  geom_point(data = mtcars, mapping = aes(x = disp, y = wt))
```



geom_point mtcars x disp y wt aes aesthetic mappings (x y

```
g <- ggplot()
g1 <- g + geom_point(data = mtcars, mapping = aes(x = disp, y = wt))
g2 <- g1 + geom_line(data = mtcars, mapping = aes(x = disp, y = wt))
print(g2)
```



g 1 g

```
ggplot(data = mtcars, mapping = aes(x = disp, y = wt)) +
  geom_point() +
  geom_line()
```

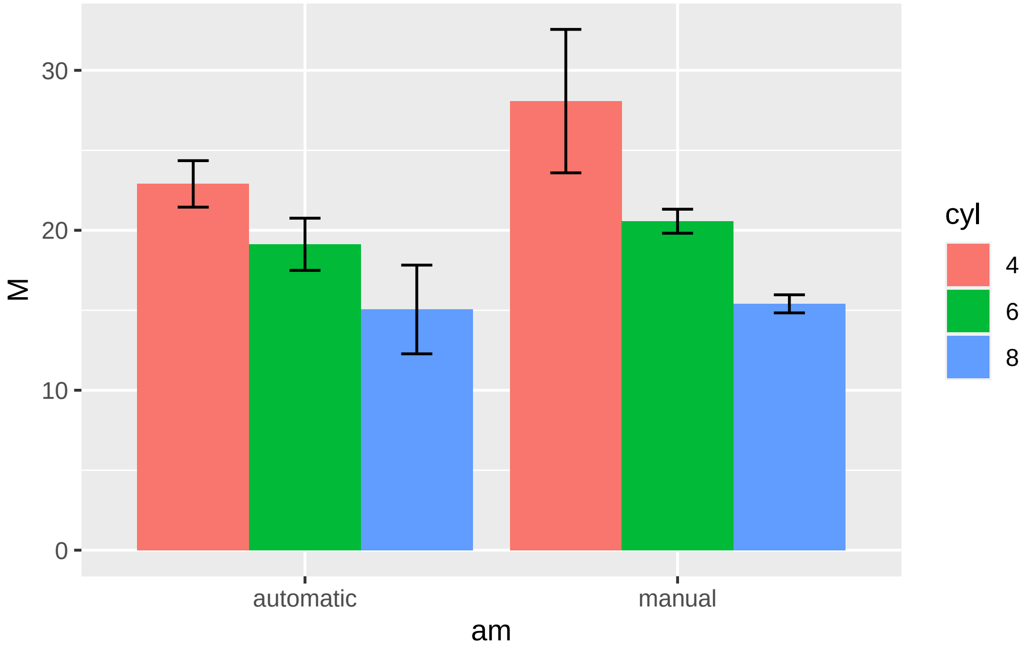
ggplot

```
mtcars %>%
  ggplot(mapping = aes(x = disp, y = wt)) +
  geom_point() +
  geom_line()
```

ggplot x y

```
# mtcars
mtcars %>%
  #
  select(mpg, cyl, wt, am) %>%
  mutate(
    #  am,cyl Factor
    am = factor(am, labels = c("automatic", "manual")),
    cyl = factor(cyl)
  ) %>%
  #
  group_by(am, cyl) %>%
  summarise(
    M = mean(mpg), #       M
    SD = sd(mpg), #         SD
    .groups = "drop" # summarise
  ) %>%
  # x         y         cyl
  ggplot(aes(x = am, y = M, fill = cyl)) +
  #
```

```
  geom_bar(stat = "identity", position = "dodge") +
  # ±1SD
  geom_errorbar(
    #
    aes(ymin = M - SD, ymax = M + SD),
    #
    position = position_dodge(width = 0.9),
    width = 0.25 #
  )
```



[3]

4.5 tips

AI et al. [7] 4

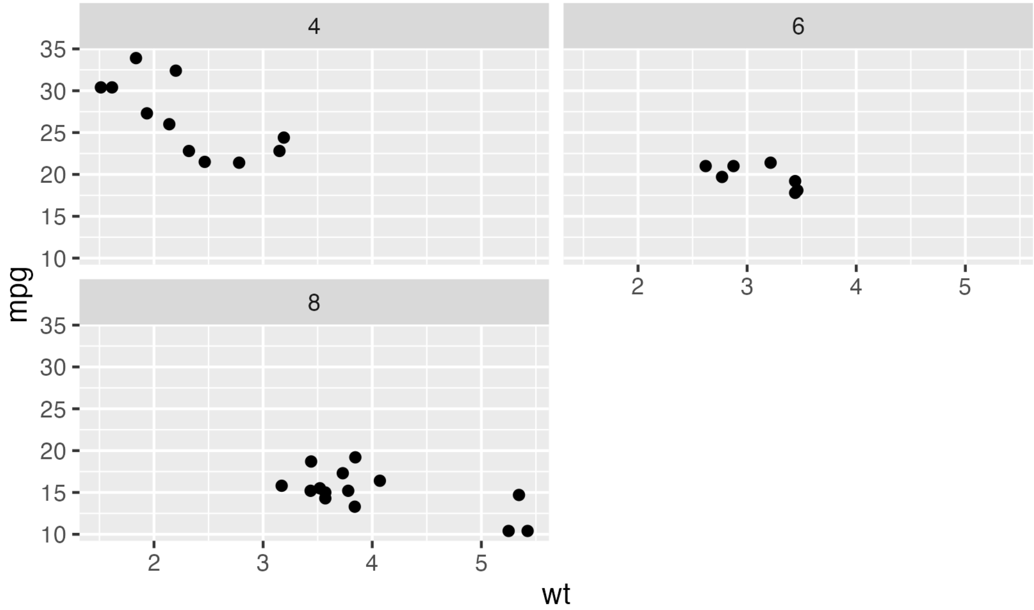
4.5.1 ggplot

mtcars am 2

facet_wrap facet_grid 2

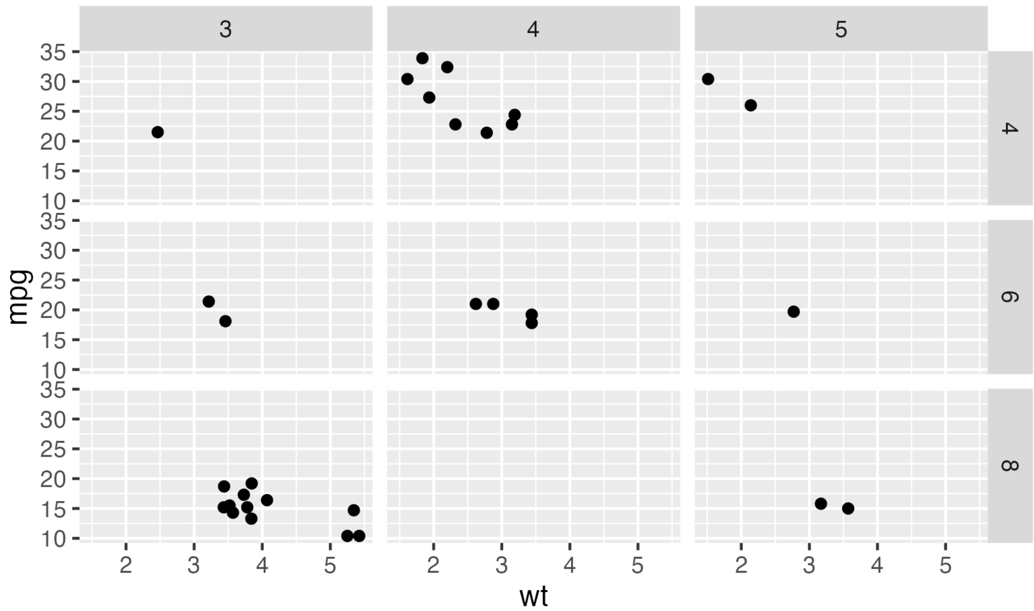
```
mtcars %>%
  # wt mpg
  ggplot(aes(x = wt, y = mpg)) +
  geom_point() +
  # cyl
  facet_wrap(~cyl, nrow = 2) +
  #
  labs(caption = "facet_wrap ")
```

[3] chatGPTver4



facet_wrapの例

```
mtcars %>%
  ggplot(aes(x = wt, y = mpg)) +
  geom_point() +
  #    cyl   gear
  facet_grid(cyl ~ gear) +
  #
  labs(caption = "facet_grid ")
```

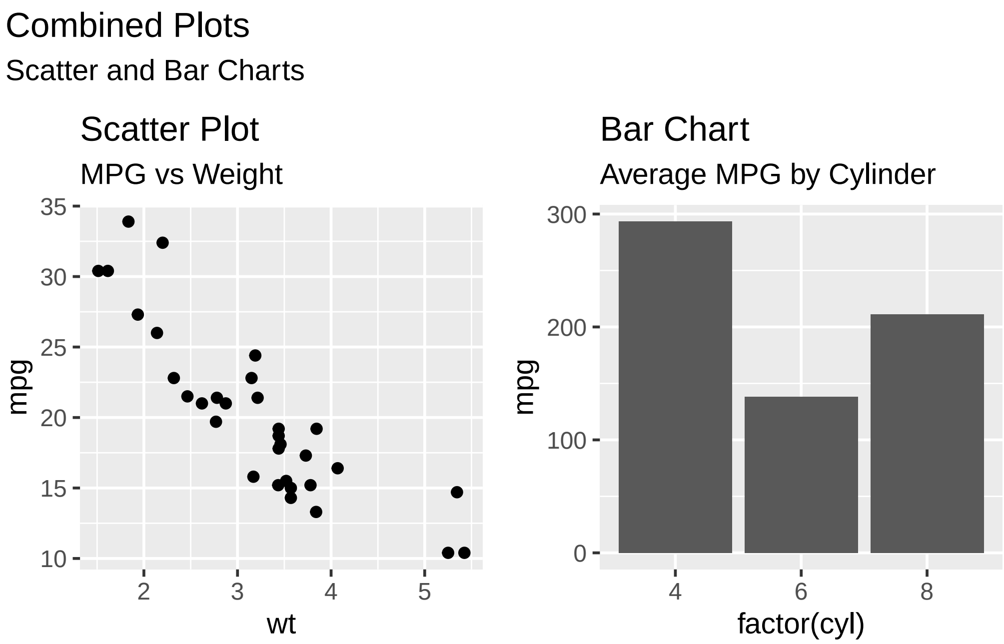


facet_gridの例

patchwork

```
library(patchwork)

#
g1 <- ggplot(mtcars, aes(x = wt, y = mpg)) +
```

```
  geom_point() +
  #
  ggtitle("Scatter Plot", "MPG vs Weight")

#
g2 <- ggplot(mtcars, aes(x = factor(cyl), y = mpg)) +
  geom_bar(stat = "identity") +
  #
  ggtitle("Bar Chart", "Average MPG by Cylinder")

# patchwork   2
combined_plot <- g1 + g2 +
  plot_annotation(
    title = "Combined Plots",
    subtitle = "Scatter and Bar Charts"
  )

#
print(combined_plot)
```



4.5.2 ggplot

Rmd Quarto ggsave ggplot

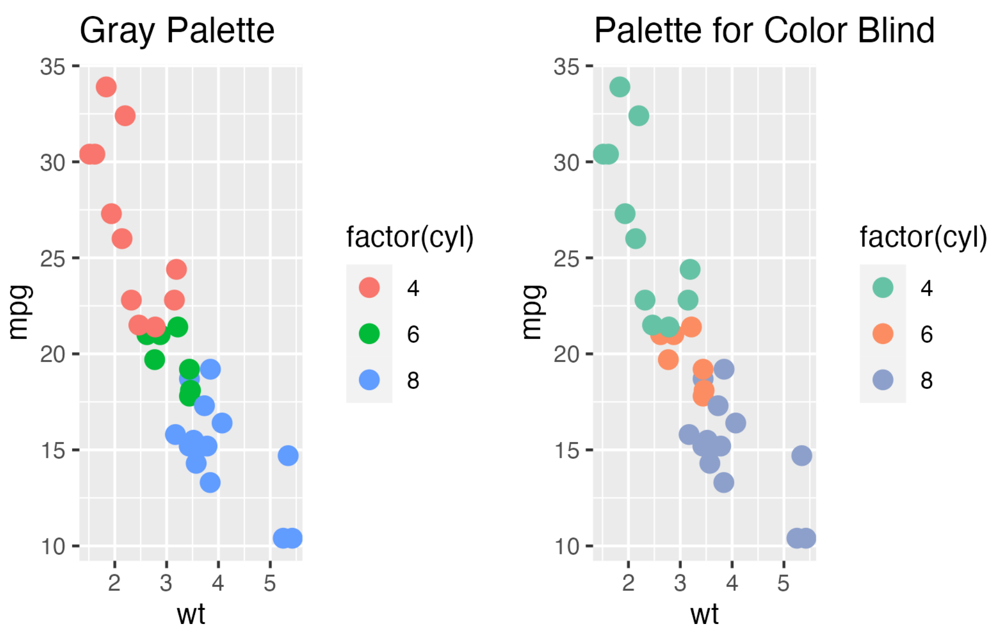
```
#
p <- ggplot(mtcars, aes(x = wt, y = mpg)) +
  geom_point()
ggsave(
  filename = "my_plot.png", #
  plot = p, #
  device = "png", #
  path = "path/to/directory", #
  scale = 1, #
  width = 5, #
```

```
  height = 5, #
  dpi = 300, #   DPI: dots per inch
)
```

4.5.3

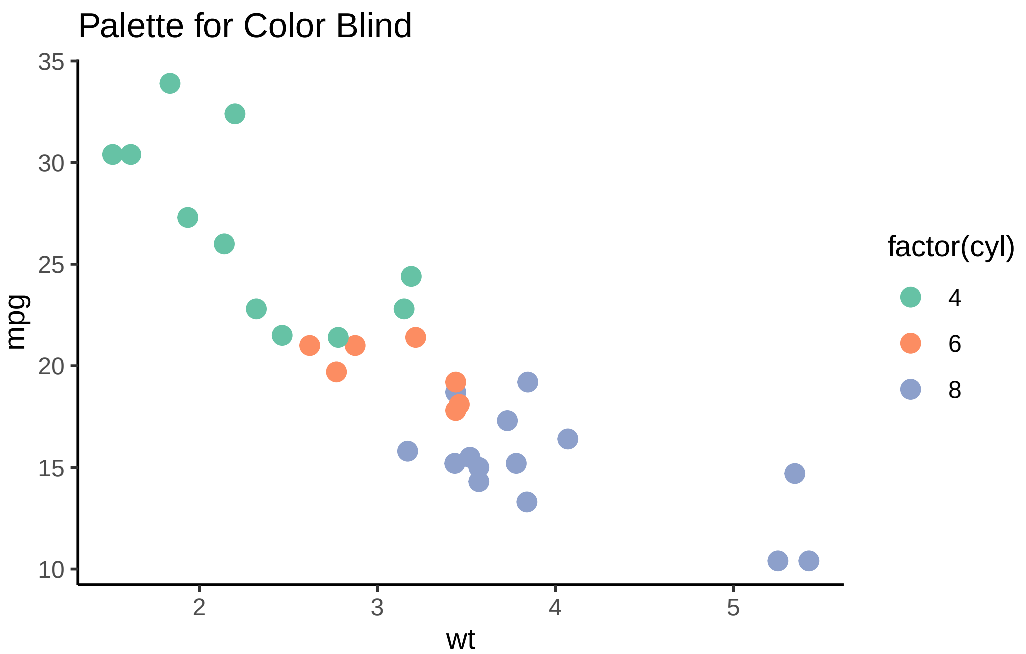
ggplot ( ) ( ) Grays

```
#
p1 <- ggplot(mtcars, aes(x = wt, y = mpg, color = factor(cyl))) +
  geom_point(size = 3) +
  scale_fill_brewer(palette = "Greys") +
  ggtitle("Gray Palette")

#
library(RColorBrewer)
#
p2 <- ggplot(mtcars, aes(x = wt, y = mpg, color = factor(cyl))) +
  geom_point(size = 3) +
  scale_color_brewer(palette = "Set2") + #
  ggtitle("Palette for Color Blind")

#
combined_plot <- p1 + p2 + plot_layout(ncol = 2)
print(combined_plot)
```



ggplot2 theme_gray() theme_classic() theme_bw

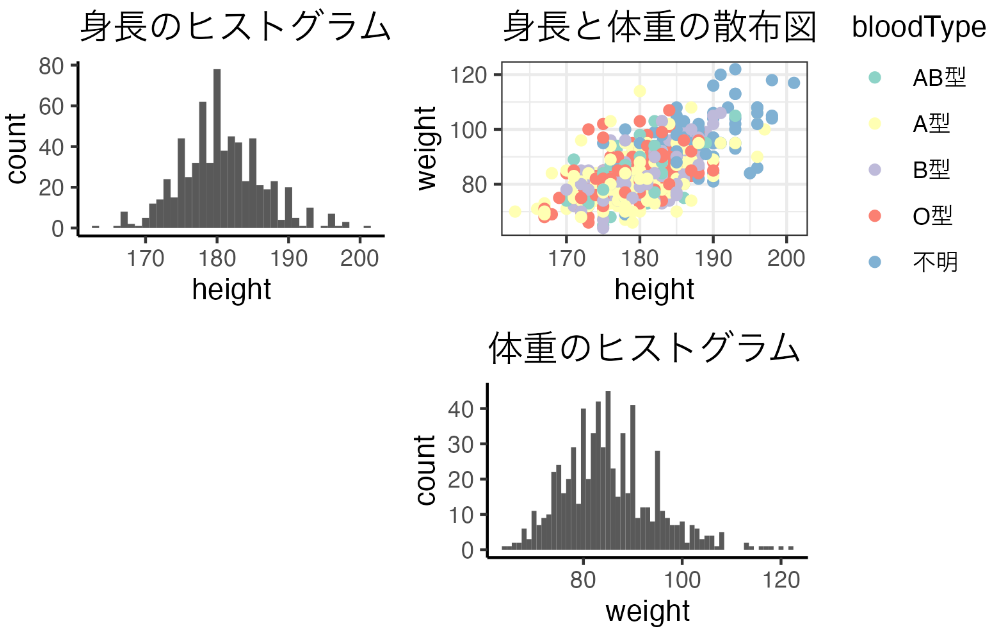
```
p2 + theme_classic()
```



4.6

- Rmarkdown ( )

1. Baseball.csv 2020 dat.tb
2. dat.tb theme_classic
3. dat.tb theme_bw
4. ( ) Set3
5. ( )
6. dat.tb
7. ( ) geom_smooth() method
8. ( ) geom_smooth() method="lm"
9. x y dat.tb2 geom_point(stat="summary", fun=mean)
10. 2,4 ggsave


身長のヒストグラム
count
height
身長と体重の散布図
weight
height
bloodType
AB型
A型
B型
O型
不明
体重のヒストグラム
count
weight

Chapter 5

R

R , , and [19] J.P. [6], *R* *:* [8], *R* [14]

C Java Python Julia R R tion 2.5.1) R

[1] 3

5.1

( ) Chapter 2

```r
a <- 0
a <- a + 1
print(a)
```

```
[1] 1
```

`=` 2 `a = a + 1` ( )`a` 1 ( )`a` (=) `<-`

`a <- 0` `a` `0`

`remove`

```r
remove(a)
```

RStudio Environment `a` RStudio Environment `remove(list=ls())` [2]

5.2

5.2.1 for

`for` for for R `for`

[1] [18] 117

[2] `ls()` list objects

```
for (value in sequence) {
    #
}
```

value sequence sequence #

for

```
for (i in 1:5) {
  cat("   ", i, "  \n")
}
```

```
    1
    2
    3
    4
    5
```

for (i) (1:5 1,2,3,4,5) cat

sequence

```
for (i in c(2, 4, 12, 3, -6)) {
  cat("   ", i, "  \n")
}
```

```
    2
    4
    12
    3
    -6
```

( )

```
# 2
A <- matrix(1:9, nrow = 3)

#
for (i in 1:nrow(A)) {
  #
  for (j in 1:ncol(A)) {
    cat("  [", i, ", ", j, "]  ", A[i, j], "\n")
  }
}
```

```
  [ 1 ,  1 ]  1
  [ 1 ,  2 ]  4
  [ 1 ,  3 ]  7
  [ 2 ,  1 ]  2
  [ 2 ,  2 ]  5
  [ 2 ,  3 ]  8
  [ 3 ,  1 ]  3
  [ 3 ,  2 ]  6
  [ 3 ,  3 ]  9
```

i j i R for ( )

5.2.2 while

while True "while"

R while

```
while (condition) {
    #
}
```

condition # 1 10 while

```
i <- 1
while (i <= 5) {
  print(i)
  i <- i + 1
}
```

```
[1] 1
[1] 2
[1] 3
[1] 4
[1] 5
```

i 5 print(i) i i <- i + 1 i 1 i 10 (False)

while condition (True) condition (False)

R for while

5.3

R if-else

5.3.1 if

if

```
if ( ) {
    #
}
```

if (TRU) {} else FALSE

```
if ( ) {
    #
} else {
    #
}
```

```
x <- 10

if (x > 0) {
  print("x is positive")
} else {
  print("x is not positive")
}
```

```
[1] "x is positive"
```

x

x > 0, y == 1 TRUE/FALSE is.numeric(x) (&&, ||)

x y Other case x y

```
x <- 10
y <- -3

if (x > 0 && y < 0) {
  print("x is positive and y is negative")
} else {
  print("Other case")
}
```

```
[1] "x is positive and y is negative"
```

5.4

1. 1 20
2. 1 40 3 (1 10 3) 3
3. `c(1, -2, 3, -4, 5)` "positive" "negative"
4. A B R `%*%` `for` i j c_{ij} A i B j

$$c_{ij} = \sum_k a_{ik} b_{kj}$$

```
A <- matrix(1:6, nrow = 3)
B <- matrix(3:10, nrow = 2)
##
print(A)
```

```
     [,1] [,2]
[1,]    1    4
[2,]    2    5
[3,]    3    6
```

```
print(B)
```

```
     [,1] [,2] [,3] [,4]
[1,]    3    5    7    9
[2,]    4    6    8   10
```

```
##
C <- A %*% B
print(C)
```

```
     [,1] [,2] [,3] [,4]
[1,]   19   29   39   49
[2,]   26   40   54   68
[3,]   33   51   69   87
```

5.5

R

5.5.1

(argment) (,value) $y = f(x)$ x y f

R

```
function_name <- function(argument) {
   # function body
   return(value)
}
```

`function body` 3 `add3`

```
add3 <- function(x) {
  x <- x + 3
  return(x)
}
#
add3(5)
```

```
[1] 8
```

2

```
add_numbers <- function(a, b) {
  sum <- a + b
  return(sum)
}
#
add_numbers(2, 5)
```

```
[1] 7
```

default value

```
add_numbers2 <- function(a, b = 1) {
  sum <- a + b
  return(sum)
}
#
add_numbers2(2, 5)
```

```
[1] 7
```

```
add_numbers2(4)
```

```
[1] 5
```

`(a,b=1)` `b` `1` 2 (2+5) 1 `a` `b` (4+1)

5.5.2

R 1 `list`

```
calculate_values <- function(a, b) {
  sum <- a + b
  diff <- a - b
  #
  result <- list("sum" = sum, "diff" = diff)
  return(result)
}
#
result <- calculate_values(10, 5)
```

```
#
print(result)
```

```
$sum
[1] 15

$diff
[1] 5
```

5.6

1. "positive" "negative" 0 "Zero"
2. 2
3.
4. R `var` $\hat{\sigma}$ `v`

$$\hat{\sigma} = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2$$

$$v = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2$$

Chapter 6

6.1

Sample

(2) (3)

[1] 0 1

(80-90%) 1

and [4])

R

6.2

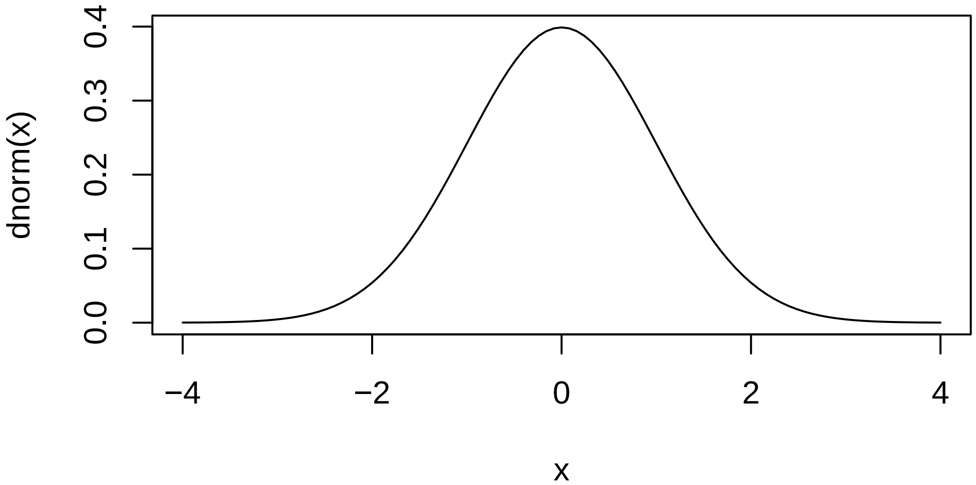
(Probability Density Function) **(Probability Mass Function)**

R

```
#       curve
curve(dnorm(x), from = -4, to = 4)
```

1 [16] , [5], [11]

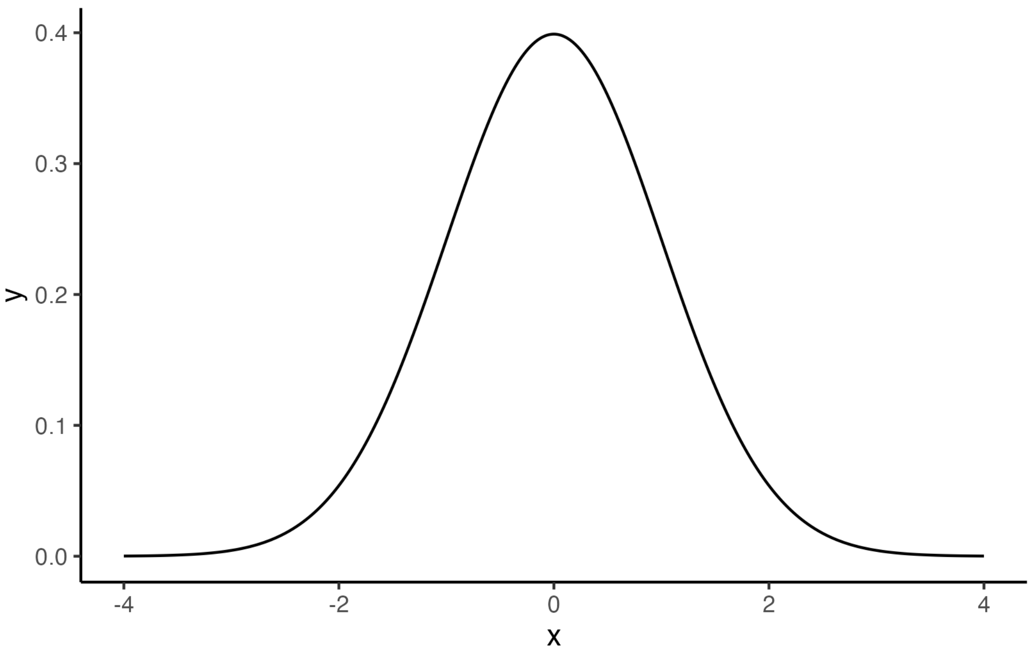
2 X%




```
# ggplot2
library(tidyverse)
```

```
-- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
v dplyr     1.1.4     v readr     2.1.4
v forcats   1.0.0     v stringr   1.5.1
v ggplot2   3.4.4     v tibble    3.2.1
v lubridate 1.9.3     v tidyr     1.3.0
v purrr     1.0.2
-- 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 become error
```

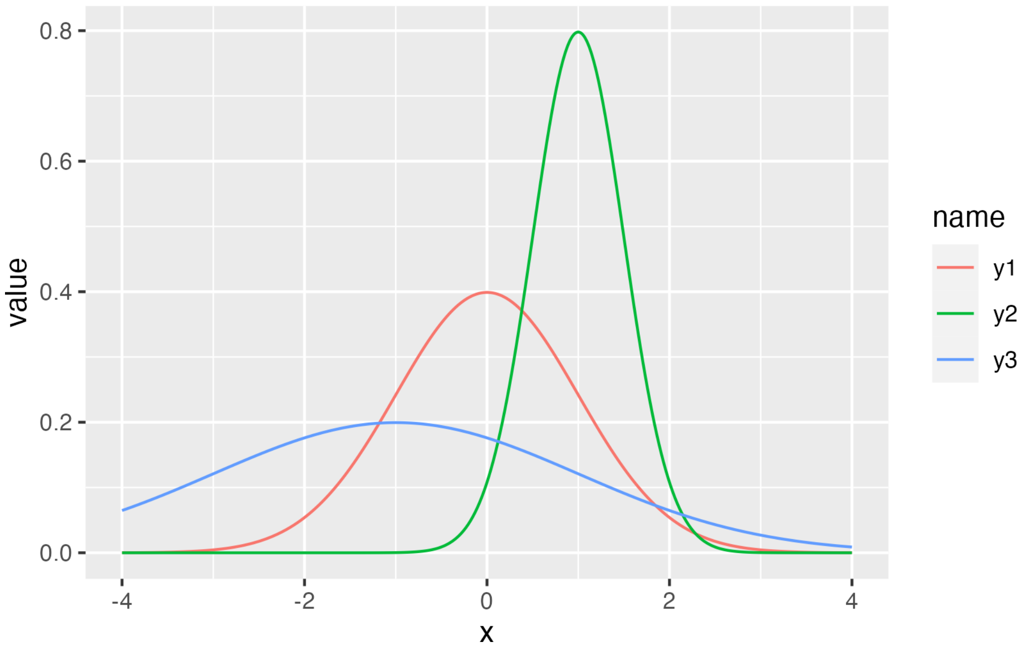
```
data.frame(x = seq(-4, 4, by = 0.01)) %>%
  mutate(y = dnorm(x)) %>%
  ggplot(aes(x = x, y = y)) +
  geom_line() +
  theme_classic()
```



`dnorm` `d` Density( ) `norm` Normal Distribution( ) R (`norm`) (d) `p`,`q`,`r` `dpo`
distribution) `pnorm`(normal distribution),`rbinom`(binomial distribution)

μ σ **parameter** 3

```
data.frame(x = seq(-4, 4, by = 0.01)) %>%
  mutate(
    y1 = dnorm(x, mean = 0, sd = 1),
    y2 = dnorm(x, mean = 1, sd = 0.5),
    y3 = dnorm(x, mean = -1, sd = 2)
  ) %>%
  pivot_longer(-x) %>%
  ggplot(aes(x = x, y = value, color = name)) +
  geom_line()
```



`d` `dnorm` `p` `q`

```
#
pnorm(1.96, mean = 0, sd = 1)
```

```
[1] 0.9750021
```

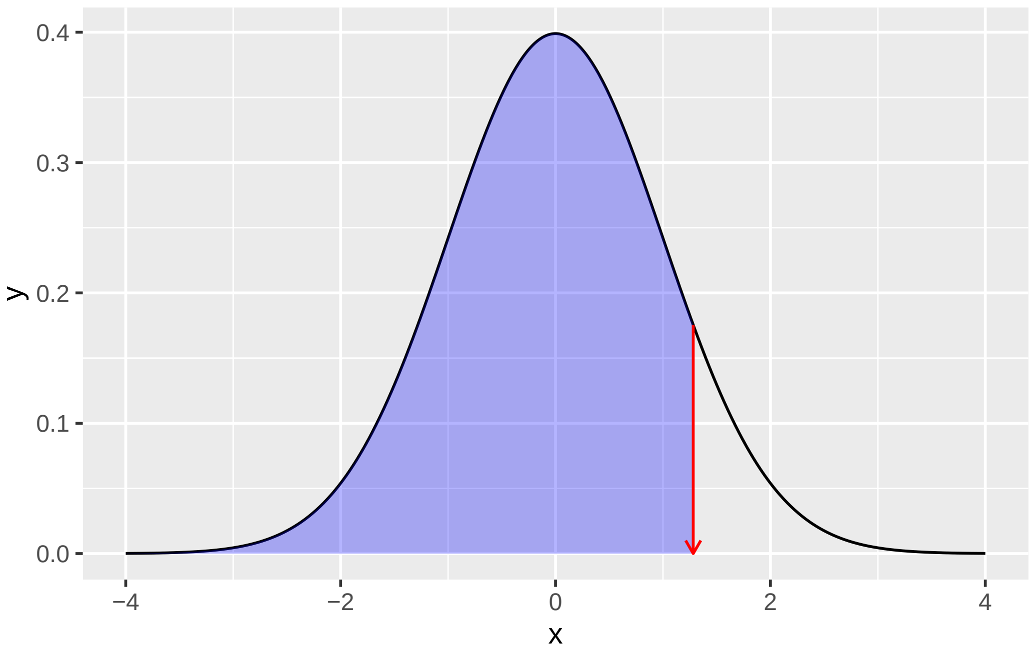
```
#
qnorm(0.975, mean = 0, sd = 1)
```

```
[1] 1.959964
```

`pnorm` x ( ) `qnorm` (=) x

```
#
prob <- 0.9
##
df1 <- data.frame(x = seq(from = -4, 4, by = 0.01)) %>%
  mutate(y = dnorm(x, mean = 0, sd = 1))
## qnorm(0.975)
df2 <- data.frame(x = seq(from = -4, qnorm(prob), by = 0.01)) %>%
```

```r
  mutate(y = dnorm(x, mean = 0, sd = 1))
##
ggplot() +
  geom_line(data = df1, aes(x = x, y = y)) +
  geom_ribbon(data = df2, aes(x = x, y = y, ymin = 0, ymax = y), fill = "blue", alpha = 0.3) +
  ##
  geom_segment(
    aes(x = qnorm(prob), y = dnorm(qnorm(prob)), xend = qnorm(prob), yend = 0),
    arrow = arrow(length = unit(0.2, "cm")), color = "red"
  )
```



d,p,q,r r

6.3

( )

[3] RPG

( ) R 50 SD10 10

```r
rnorm(n = 10, mean = 50, sd = 10)
```

```
 [1] 38.84226 52.59830 69.21195 46.66273 57.41120 46.22543 46.59076 55.70614
 [9] 63.08704 58.25856
```

```r
rnorm(n = 10, mean = 50, sd = 10)
```

```
 [1] 47.16534 43.23437 36.22050 50.87181 54.92167 33.68537 51.24687 37.19164
 [9] 49.59670 58.47268
```

set.seed (seed)

[3] 5,3,8

```
# seed
set.seed(12345)
rnorm(n = 3)
```

```
[1]  0.5855288  0.7094660 -0.1093033
```

```
#  seed
set.seed(12345)
rnorm(n = 3)
```

```
[1]  0.5855288  0.7094660 -0.1093033
```

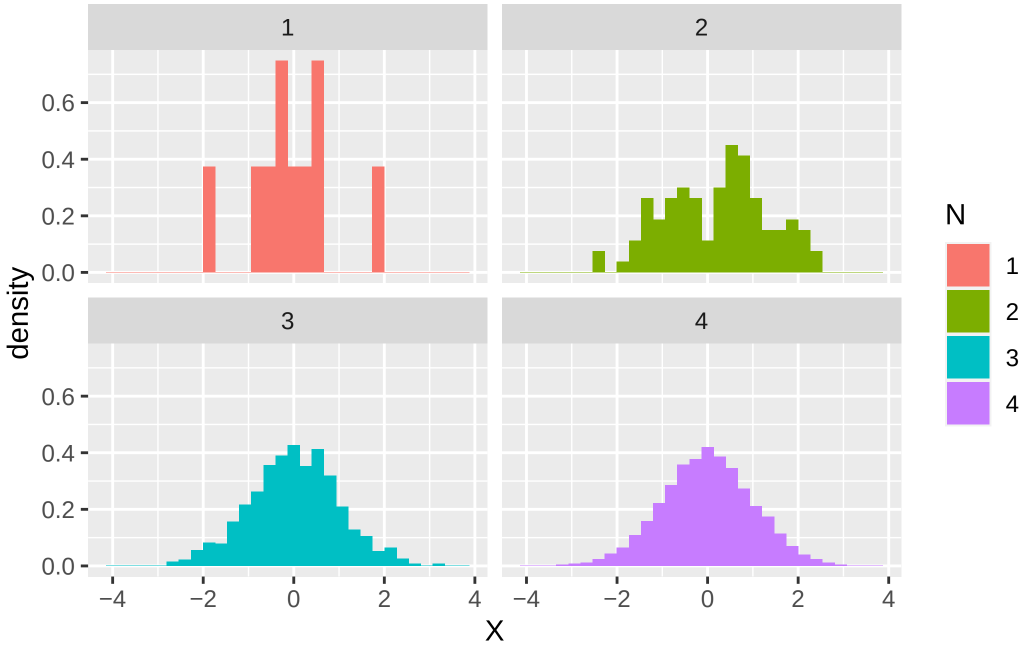
6.3.1

$$n = 10, 100, 1000, 10000$$

```
rN10 <- rnorm(10)
rN100 <- rnorm(100)
rN1000 <- rnorm(1000)
rN10000 <- rnorm(10000)

data.frame(
  N = c(
    rep(1, 10), rep(2, 100),
    rep(3, 1000), rep(4, 10000)
  ),
  X = c(rN10, rN100, rN1000, rN10000)
) %>%
  mutate(N = as.factor(N)) %>%
  ggplot(aes(x = X, fill = N)) +
  #
  geom_histogram(aes(y = ..density..)) +
  facet_wrap(~N)
```

```
Warning: The dot-dot notation (`..density..`) was deprecated in ggplot2 3.4.0.
i Please use `after_stat(density)` instead.
```

```
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



10 100,1000

R t F χ^2

(MCMC)

(=) -1.5 +1.5

$$p = \int_{-1.5}^{+1.5} \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}} dx$$

`pnorm`

```
pnorm(+1.5, mean = 0, sd = 1) - pnorm(-1.5, mean = 0, sd = 1)
```

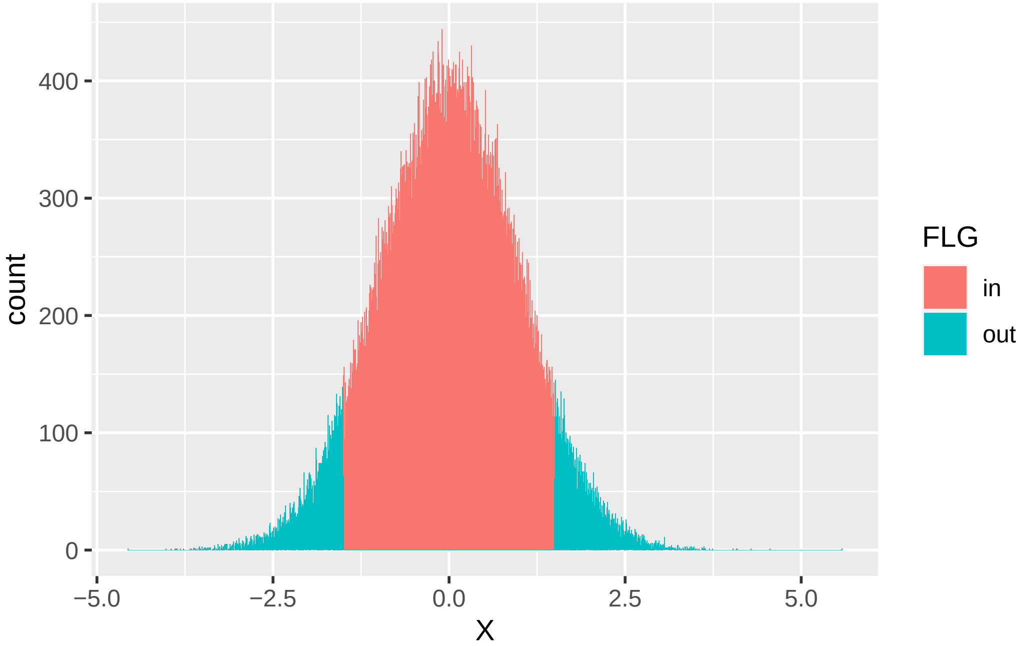
```
[1] 0.8663856
```

```
x <- rnorm(100000, mean = 0, sd = 1)
df <- data.frame(X = x) %>%
  #
  mutate(FLG = ifelse(X > -1.5 & X < 1.5, 1, 2)) %>%
  mutate(FLG = factor(FLG, labels = c("in", "out")))
##
df %>%
  group_by(FLG) %>%
  summarise(n = n()) %>%
  mutate(prob = n / 100000)
```

```
# A tibble: 2 x 3
  FLG       n  prob
  <fct> <int> <dbl>
1 in    86642 0.866
2 out   13358 0.134
```

10,000 (1, 2) factor `FLG` `0.866` `pnorm`

```
##
df %>%
  ggplot(aes(x = X, fill = FLG)) +
  geom_histogram(binwidth = 0.01)
```



10 100 ( )

6.4

2

1. 100, 8

$$E[X] = \int_{-\infty}^{\infty} x f(x)\,dx$$

x $f(x)$ 100

2. 100, 3

$$\sigma^2 = \int_{-\infty}^{\infty} (x-\mu)^2 f(x)\,dx$$

μ $3^2 = 9$

3. 65 10 X $90 < X < 110$

```
pnorm(108, mean = 65, sd = 10) - pnorm(92, mean = 65, sd = 10)
```

```
[1] 0.003458434
```

4. 10 10 7

```
1 - pnorm(7, mean = 10, sd = 10)
```

```
[1] 0.6179114
```

5. X, Y X 10,SD10 Y 5 SD8 X Y $Z = X+Y$ X, Y

6.5

population **sample** /

100 100

```r
set.seed(12345)
# 100 2
Po <- rnorm(100, mean = 150, sd = 10) %>% round(2)
print(Po)
```

```
  [1] 155.86 157.09 148.91 145.47 156.06 131.82 156.30 147.24 147.16 140.81
 [11] 148.84 168.17 153.71 155.20 142.49 158.17 141.14 146.68 161.21 152.99
 [21] 157.80 164.56 143.56 134.47 134.02 168.05 145.18 156.20 156.12 148.38
 [31] 158.12 171.97 170.49 166.32 152.54 154.91 146.76 133.38 167.68 150.26
 [41] 161.29 126.20 139.40 159.37 158.54 164.61 135.87 155.67 155.83 136.93
 [51] 144.60 169.48 150.54 153.52 143.29 152.78 156.91 158.24 171.45 126.53
 [61] 151.50 136.57 155.53 165.90 144.13 131.68 158.88 165.93 155.17 137.04
 [71] 150.55 142.15 139.51 173.31 164.03 159.43 158.26 141.88 154.76 160.21
 [81] 156.45 160.43 146.96 174.77 159.71 168.67 156.72 146.92 155.37 158.25
 [91] 140.36 141.45 168.87 146.08 140.19 156.87 144.95 171.58 144.00 143.05
```

100

```r
M <- mean(Po)
V <- mean((Po - M)^2)
#
print(M)
```

```
[1] 152.4521
```

```r
#
print(V)
```

```
[1] 123.0206
```

10 10 R `sample`

```r
s1 <- sample(Po, size = 10)
s1
```

```
 [1] 164.61 155.86 136.93 143.29 160.43 168.87 151.50 155.17 153.71 135.87
```

s1

```r
m1 <- mean(s1)
v1 <- mean((s1 - mean(s1))^2)
#
print(m1)
```

```
[1] 152.624
```

```r
#
print(v1)
```

```
[1] 110.2049
```

152.4521 152.624 152.624 152.624

```r
s2 <- sample(Po, size = 10)
s2
```

```
 [1] 154.76 135.87 143.05 171.45 136.57 170.49 156.87 158.25 155.17 155.20
```

```
m2 <- mean(s2)
v2 <- mean((s2 - mean(s2))^2)
#     2
print(m2)
```

```
[1] 153.768
```

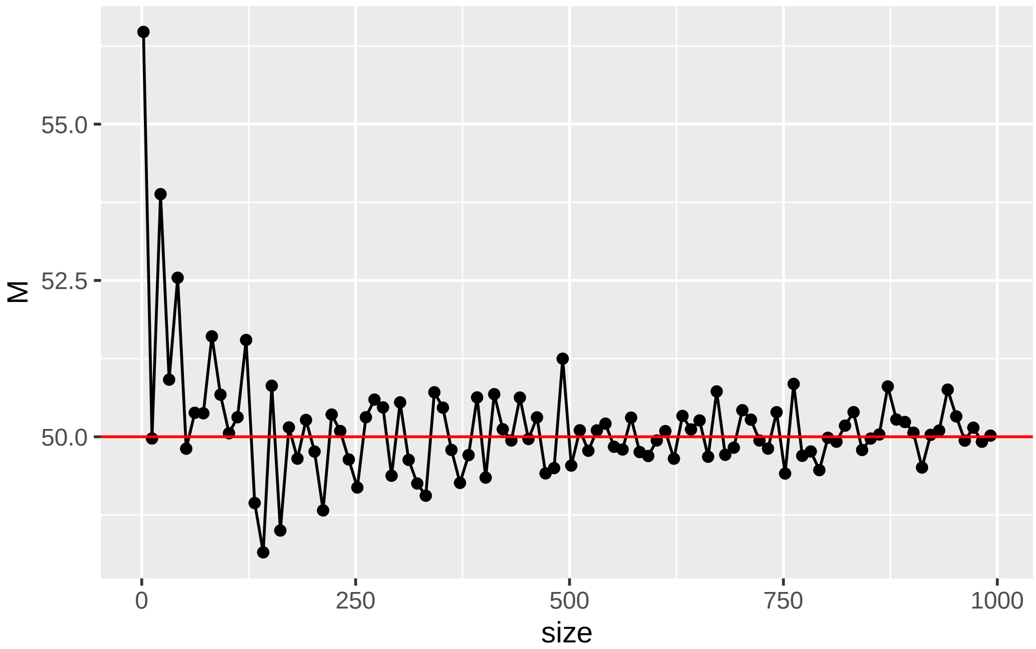
153.768 153.768 1 152.624 2 153.768 152.4521 (-0.1719 - 1.3159)

6.6

100 10 20 30 consistency

50,SD10 2 1000

```
set.seed(12345)
sample_size <- seq(from = 2, to = 1000, by = 10)
#
sample_mean <- rep(0, length(sample_size))
#
for (i in 1:length(sample_size)) {
  sample_mean[i] <- rnorm(sample_size[i], mean = 50, sd = 10) %>%
    mean()
}

#
data.frame(size = sample_size, M = sample_mean) %>%
  ggplot(aes(x = size, y = M)) +
  geom_point() +
  geom_line() +
  geom_hline(yintercept = 50, color = "red")
```



50

6.7

( ) unbiasedness

n $n-1$

50 SD10($10^2 = 100$) $n = 20$ 20 ( )

```
iter <- 5000
vars <- rep(0, iter)
unbiased_vars <- rep(0, iter)

##
set.seed(12345)
for (i in 1:iter) {
  sample <- rnorm(n = 20, mean = 50, sd = 10)
  vars[i] <- mean((sample - mean(sample))^2)
  unbiased_vars[i] <- var(sample)
}

##
mean(vars)
```

```
[1] 95.08531
```

```
mean(unbiased_vars)
```

```
[1] 100.0898
```

`vars` 95.0853144 ( ) 100 R `var` 100.0898047

efficacy , , and [19]

6.8

=

50 10 10 ( ) ± 5 0

```
iter <- 10000
n <- 10
mu <- 50
SD <- 10

#
m <- rep(0,iter)

set.seed(12345)
for (i in 1:iter) {
  #
  sample <- rnorm(n, mean = mu, sd = SD)
  m[i] <- mean(sample)
}

result.df <- data.frame(m = m) %>%
```

```
  #     TRUE,   FALSE
  mutate(
    point_estimation = ifelse(m == mu, TRUE, FALSE),
    interval_estimation = ifelse(m - 5 <= mu & mu <= m + 5, TRUE, FALSE)
  ) %>%
  summarise(
    n1 = sum(point_estimation),
    n2 = sum(interval_estimation),
    prob1 = mean(point_estimation),
    prob2 = mean(interval_estimation)
  ) %>% print()
```

```
  n1   n2 prob1 prob2
1  0 8880     0 0.888
```

10^4 8880 88.8%

100% ( ) 5% 95% 95% confidence interval

6.8.1

95%

μ σ^2 μ, $\frac{\sigma^2}{n}$ ($\frac{\sigma}{\sqrt{n}}$)

95% ± 1.96

```
#   2.5%
qnorm(0.025)
```

```
[1] -1.959964
```

```
qnorm(0.975)
```

```
[1] 1.959964
```

$\bar{X}$ 95% 1.96

$$\bar{X} - 1.96\frac{\sigma}{\sqrt{n}} \leq \mu \leq \bar{X} + 1.96\frac{\sigma}{\sqrt{n}}$$

95

```
interval <- 1.96 * SD / sqrt(n)
result.df2 <- data.frame(m = m) %>%
  #     TRUE,   FALSE
  mutate(
    interval_estimation = ifelse(m - interval  <= mu & mu <= m + interval, TRUE, FALSE)
  ) %>%
  summarise(
    prob = mean(interval_estimation)
  ) %>% print()
```

```
    prob
1 0.9498
```

6.8.2

( ) $n-1$ t (, , and [19]) 95% ±1.96 t

$$\bar{X} + T_{0.025}\frac{U}{\sqrt{n}} \leq \mu \leq \bar{X} + T_{0.975}\frac{U}{\sqrt{n}}$$

$T_{0.025}$ t 2.5 $T_{0.975}$ 97.5 t (0) $T_{0.025} = -T_{0.975}$ U^2 (U)

95

```
#
iter <- 10000
n <- 10
mu <- 50
SD <- 10

#
m <- rep(0,iter)
interval <- rep(0,iter)

set.seed(12345)
for (i in 1:iter) {
  #
  sample <- rnorm(n, mean = mu, sd = SD)
  m[i] <- mean(sample)
  U <- sqrt(var(sample)) # sd(sample)
  interval[i] <- qt(p=0.975,df=n-1) * U / sqrt(n)
}

result.df <- data.frame(m = m,interval = interval) %>%
  #      TRUE,   FALSE
  mutate(
    interval_estimation = ifelse(m - interval <= mu & mu <= m + interval, TRUE, FALSE)
  ) %>%
  summarise(
    prob = mean(interval_estimation)
  ) %>% print()
```

```
    prob
1 0.9482
```

6.9

1. $M = \frac{1}{n}\sum x_i$ $HM = \frac{n}{\sum \frac{1}{x_i}}$ $GM = (\prod x_i)^{\frac{1}{n}} = \exp(\frac{1}{n}\sum \log(x_i)))$
2. n $\nu = 3$ t t `rt()` `ncp` 0
3. t ν `rt()` 10,50,100 1000
4. 50 10 1000 95%
5. 100 15 10 100 1000 95%

Chapter 7

7.1

7.2

7.3

7.4

Chapter 8

8.1

8.2

8.3

8.4

8.5

Chapter 9

Test of Differences in Mean Values of Multiple Groups

9.1 Fundamentals of Analysis of Variance

9.2 Multiplicity of Tests

9.3 Using ANOVA-kun

Apologies, but no Japanese text is included in your request. Please provide the Japanese text you want translated into English. You haven't provided the Japanese text so I can't translate it into English. Please provide the Japanese text for translation.

Chapter 10

10.1 QRPs

10.2

10.3

10.3.1 t

10.3.2 t

10.3.3

Chapter 11

11.1

11.2

11.3

11.4

11.5

11.6

11.7

Chapter 12

12.1

12.2

12.3

Chapter 13

13.1

13.2

References

[1] Jonah Gabry, Rok Češnovar, and Andrew Johnson. *cmdstanr: R Interface to 'CmdStan'*. https://mc-stan.org/cmdstanr/, https://discourse.mc-stan.org. 2023.

[2] Wickham Hadley. "Tidy Data". In: *Journal of Statistical Software* 59 (2014), pp. 1–23. DOI: 10.18637/jss.v059.i10. URL: https://www.jstatsoft.org/index.php/jss/article/view/v059i10.

[3] . . Japanese. Trans. by , , and . , 2021.

[4] and . . Japanese. , 2009. ISBN: 9784274067754. URL: http://amazon.co.jp/o/ASIN/4274067750/.

[5] . . Japanese. , May 1, 1999.

[6] J.P. *R 2* . Japanese. Trans. by et al. , Dec. 28, 2018.

[7] et al. *2 R RStudio[ ] . tidyverse* . Japanese. , 2021.

[8] *R :* . Japanese. Trans. by . . , Nov. 23, 2017.

[9] William Revelle. *psych: Procedures for Psychological, Psychometric, and Personality Research*. R package version 2.1.3. Northwestern University. Evanston, Illinois, 2021. URL: https://CRAN.R-project.org/package=psych.

[10] Yves Rosseel. "lavaan: An R Package for Structural Equation Modeling". In: *Journal of Statistical Software* 48.2 (2012), pp. 1–36. DOI: 10.18637/jss.v048.i02.

[11] . . . Japanese. , Feb. 25, 1994.

[12] Stanley Smith Stevens. "On the theory of scales of measurement". In: *Science* 103.2684 (1946), pp. 677–680.

[13] . . Japanese. Ed. by . Vol. 3. Wonderful R. , 2018.

[14] *R* . Japanese. Trans. by et al. , Feb. 10, 2016.

[15] Hadley Wickham et al. "Welcome to the tidyverse". In: *Journal of Open Source Software* 4.43 (2019), p. 1686. DOI: 10.21105/joss.01686.

[16] . . Japanese. . , Feb. 25, 2021.

[17] Achim Zeileis. "CRAN Task Views". In: *R News* 5.1 (2005), pp. 39–40. URL: https://CRAN.R-project.org/doc/Rnews/.

[18] . . Japanese. , 2016.

[19] , , and . *R* . Japanese. , 2023.

[20] . . Japanese. 2020. URL: https://www.soumu.go.jp/main_content/000723697.pdf.