Kunio Takezawa

Learning Regression Analysis by Simulation

EXTRA MATERIALS extras.springer.com

Springer

Learning Regression Analysis by Simulation

Kunio Takezawa

Learning Regression Analysis by Simulation

Springer

Kunio Takezawa
National Agricultural and Food Research
Organization
Tsukuba, Ibaraki, Japan

Associate Professor (Cooperative Graduate
School System)
Graduate School of Life and Environmental
Sciences
University of Tsukuba, Ibaraki, Japan

This is the English translation from the Japanese language edition: Learning Regression Analysis by Simulation by Kunio Takezawa (author), Mingzhe Jin (editor)

Copyright © Kunio Takezawa 2012

All rights reserved

First published 2012 by KYORITSU SHUPPAN Co., Ltd., Tokyo, Japan

Additional material to this book can be downloaded from http://extras.springer.com

ISBN 978-4-431-54320-6 ISBN 978-4-431-54321-3 (eBook)
DOI 10.1007/978-4-431-54321-3
Springer Tokyo Heidelberg New York Dordrecht London

Library of Congress Control Number: 2013946308

For all who love truth

Preface

"In fact, I'm gonna show you what I mean with a little demonstration."
(True Romance, 1993)

There is wide agreement that people who do not specialize in statistics or mathematics do not like and have no need for a series of complicated mathematical formulae or a succession of abstract theorems as tools for learning statistics for practical purposes. Many points of view, however, are possible in determining which other options are appropriate. If abstract mathematical concepts are not important and the main purpose of practitioners is to learn methods for deriving concrete decisions from data generated in the real world, perhaps beginners should first become aware of the practical definitions of statistical methods and acquaint themselves with statistical software for PCs. They can do this by familiarizing themselves with examples of analyses of data obtained from natural and social phenomena, following the flow of analyses, and by analyzing similar data.

The underlying concept of this book makes it slightly different from other books in the field. The text does not treat data produced in real phenomena but, rather, describes attempts at various statistical methods including regression analysis using simulation data. This different approach is taken because populations generating data of known background unlimitedly should be used to appreciate numbers containing random errors as clear pictures cut from rich, tapestry-like images created by probability processes. Although people who are not attuned to, or do not have, a statistical sense tend to recognize a set of actual data as one reality, they are highly conscious of countless data lying behind the simulation data.

A series of simulation data generated under slightly different conditions shows how sensitively a statistical method responds to the varied nature of data. Additionally, if an event that occurs theoretically with a 5 % probability is observed with a similar probability on a PC, people will be convinced of the relationship between statistical theories and real world numbers. That is, simulation data are of great use for obtaining a clear understanding of the significance, purposes, features, and limitations of each statistical method.

When data produced in experiments and censuses are employed, it is not easy to determine whether the results obtained using a statistical method reflect the essential contents of the data in question, even if the analyst is well aware of the details and definitions of the phenomenon generating the data. If the analyst is not familiar with the data, the situation is all the worse, and results with different implications in more than one statistical method complicate the statistical inference.

On the other hand, when we utilize simulations using the free software R, it ensures the availability of diverse statistical methods ranging from the conventional to the latest ones. Since its inception, this software has continually been improved and its use has spread across the world. Therefore, with the advent of R and its subsequent development, the time is ripe for publishing an introductory book for learning statistical methods using simulations. This book is an attempt to introduce statistics along the lines laid out above. Even the minimum mathematical concepts for basic statistics can be understood through simulations using R. This approach will facilitate learning the roles of statistical methods using a specific series of numbers and acquiring the skills for analyzing actual data appropriately.

Acknowledgments

The author would like to sincerely thank the following distinguished researchers, who read and commented on the draft versions of the text.

Dr. Masashi Kiyota (Oceanic Ecosystem Group, Oceanic Resources Division, National Research Institute of Far Seas Fisheries, Fisheries Research Agency)

Dr. Takayuki Mitsunaga (Plant Protection Division & Agroinformatics Division & Statistical Advisory Group, Agricultural Research Center, National Agriculture and Food Research Organization)

Dr. Shigeyuki Mukawa (Ag Chem Discovery Laboratory, Life Science Research Institute, Kumiai Chemical Industry Co., Ltd.)

Dr. Mika Yasuda (Plant Protection Division, Agricultural Research Center, National Agriculture and Food Research Organization)

Dr. Hiroshi Shono (Field of Fisheries Biology and Oceanography, Faculty of Fisheries, Kagoshima University)

Dr. Daisuke Ochi (Ecologically Related Species Group, Tuna and Skipjack Resources Division, National Research Institute of Far Seas Fisheries, Fisheries Research Agency)

Dr. Keisuke Satoh (Tuna Fisheries Resources Group, Tuna and Skipjack Resources Division, National Research Institute of Far Seas Fisheries, Fisheries Research Agency)

Contents

Chapter 1
Linear Algebra

1.1 Starting Up and Executing R

Once the R statistical software has been downloaded and installed correctly, a workspace image file (Fig. 1.1) (with file name ".RData") will be located somewhere on your PC. R programs can be saved in this file. More than one workspace image file containing different R programs can be created on a PC by saving these image files under different names, e.g., "myprog.RData". Workspace image files can continue to be used, even if the version of R is updated.

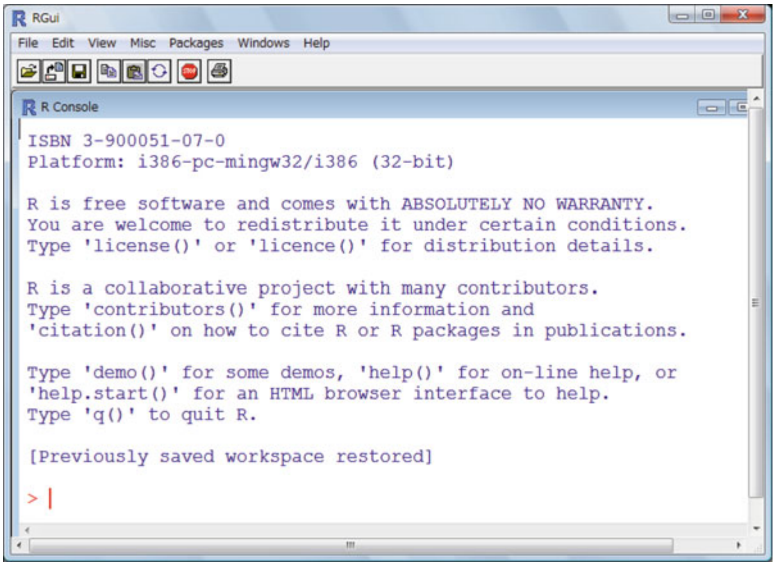
The R software package is executed by double-clicking on the workspace image file, displaying the screen as shown in Fig. 1.2. The window labeled "R Console" is called a console window. Various tasks can be carried out by inputting R commands into this console window. The last line displayed in this console window is ">", which indicates "I am ready. Please input R commands."

To create or edit an R program called `prog1`, input `fix(prog1)` in the console window (Fig. 1.3) and press the Enter key. An R editor is displayed on the screen (Fig. 1.4). If R has been installed with the default settings, notepad is started as the editor. Other editors can be used by changing the default settings.

"`function () {}`" appears in the R editor window. Now, a series of R commands can be written within the curly brackets "{ }" to create an R program. Arguments can be included within the parenthesis "`()`" for use when the R program is executed. Different constants or character strings can be passed as arguments when the R program is executed, thus allowing the same program to be used to compute different scenarios. If no arguments are set, the R commands given in the curly brackets "{ }" are executed as is. Such an R program always yields the same result.

Figure 1.5 shows an example of an R program. The value of `3` is assigned to `aa` in the first line. The symbol "`<-`" (representing an arrow and comprising a "smaller than" sign ("<") followed by a minus ("`-`")) indicates that what is on the right is assigned to what is on the left. Thus, `aa` is used to store various pieces of information. That is, `aa` is a variable. In the context of R, `aa` is referred to as

Fig. 1.1 Workspace image file, with file name ".RData"



```
ISBN 3-900051-07-0
Platform: i386-pc-mingw32/i386 (32-bit)

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

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

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

[Previously saved workspace restored]

> 
```

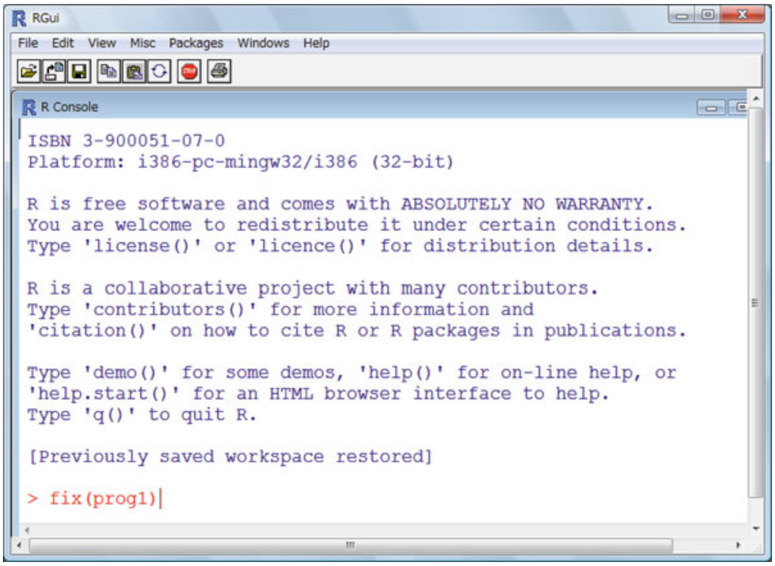
Fig. 1.2 Initial appearance of the screen when the R software is executed. A console window is displayed

an object. "Object" has a wider meaning than "variable". Even an R program is considered a kind of object. The term "object oriented" is an important concept in programing and system development. However, in the context of the R environment for creating and executing R programs, objects should merely be thought of as variables.

"=" can also be used to denote assignment, but this notation does not always work in the same way as "<-". Therefore, the use of "<-" is always recommended for assignment. The value of `-8` is assigned to `bb` in the second line. The result of the addition of `aa` and `bb` is assigned to `cc` in the third line. Finally, the content of `cc` is output to the console window in the fourth line. `function ()` at the top indicates that this R program does not make use of any arguments. Hence, this R program always yields the same result unless the commands in the body of this program are modified.

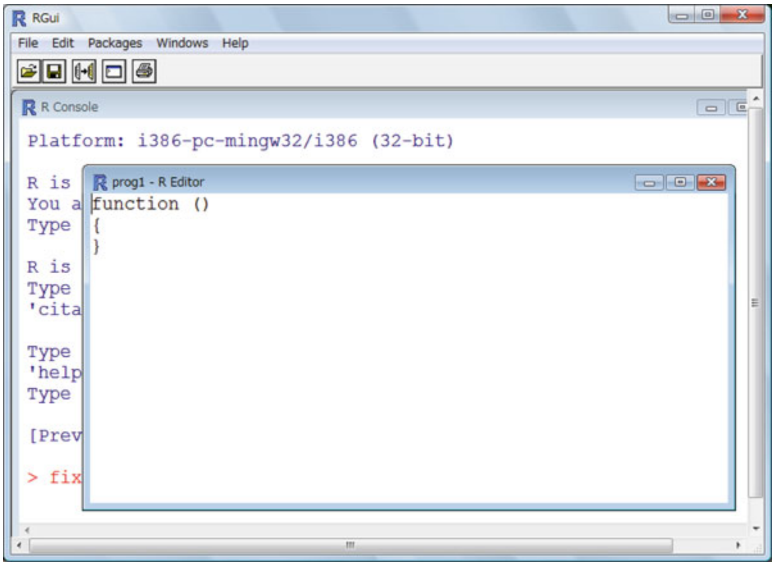
Click "×" in the upper right hand corner to quit the R editor (Fig. 1.6).

At this point a pop-up dialog window asking whether the edited R program should be saved appears on the screen (Fig. 1.7). If "Yes" is selected, the edited



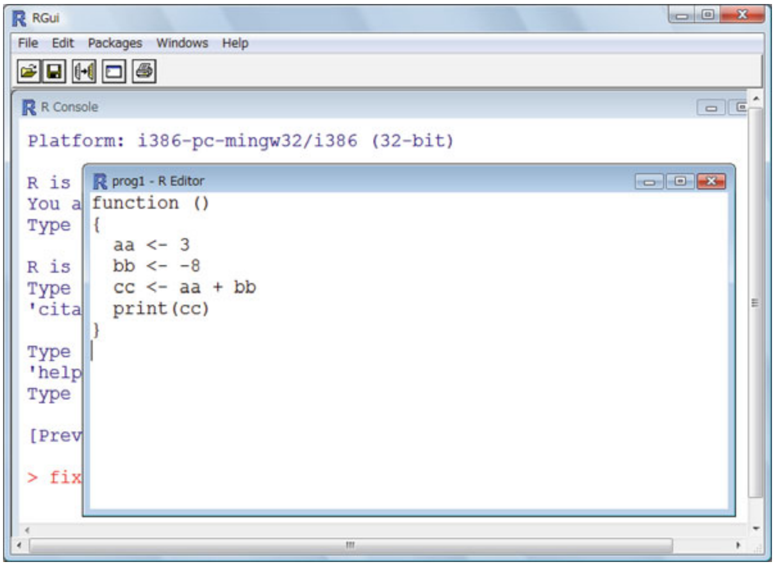
```
ISBN 3-900051-07-0
Platform: i386-pc-mingw32/i386 (32-bit)

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

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

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

[Previously saved workspace restored]

> fix(prog1)
```

Fig. 1.3 Construction of an R program



```
function ()
{
}
```

Fig. 1.4 The R editor

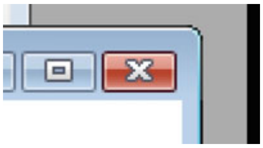


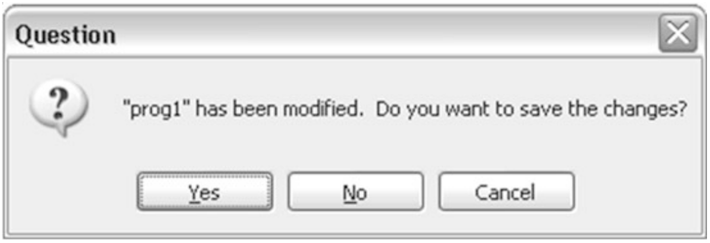
```
Platform: i386-pc-mingw32/i386 (32-bit)

R is
You a
Type

R is
Type
'cita

Type
'help
Type

[Prev

> fix
```

```
function ()
{
   aa <- 3
   bb <- -8
   cc <- aa + bb
   print(cc)
}
```

Fig. 1.5 Example R program that adds two numbers and outputs the result

Fig. 1.6 Quitting the R editor





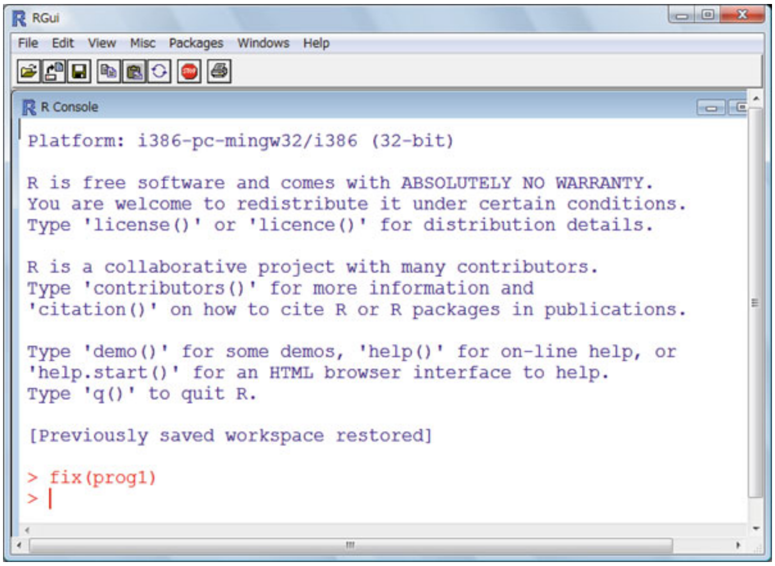
"prog1" has been modified. Do you want to save the changes?

Fig. 1.7 Pop-up dialog window asking whether the edited R program should be saved

R program is saved. Then, the R editor terminates leaving only the console window (Fig. 1.8).

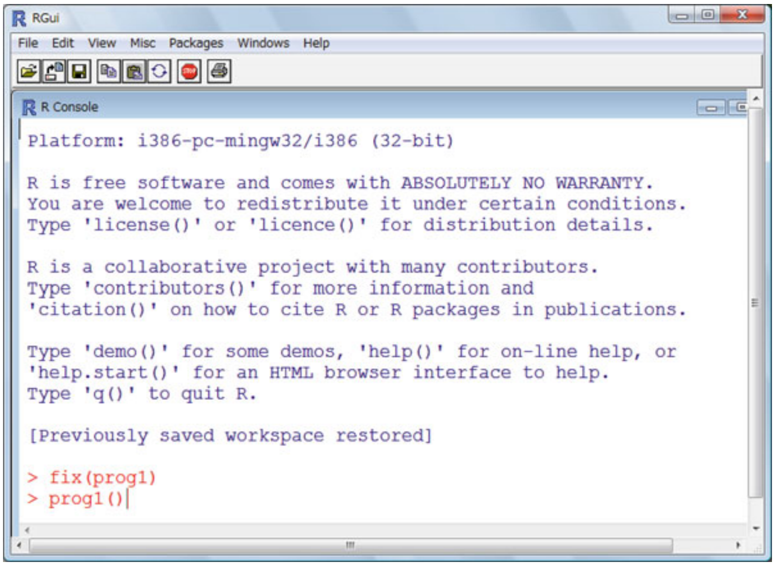
Input `prog1()` to execute `prog1` (Fig. 1.9). Next, press the Enter key. The result, giving the content of `cc`, is displayed as shown in Fig. 1.10. If `prog1()` needs to be modified, input `fix(prog1)` again.

It should be noted that if "Yes" is selected in response to the question whether the edited R program should be saved, the edited R program is in fact saved, but only in volatile memory. Hence, even if this step is done correctly, the newly created or



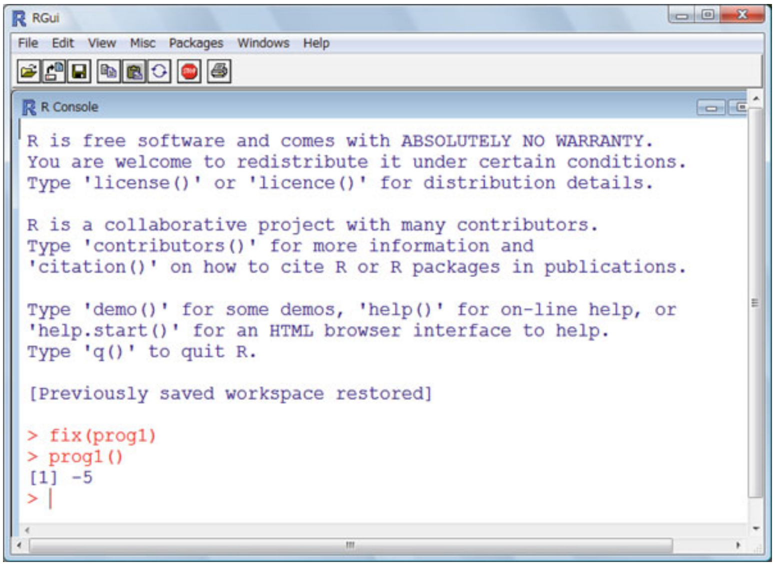
```
Platform: i386-pc-mingw32/i386 (32-bit)

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

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

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

[Previously saved workspace restored]

> fix(prog1)
> 
```

Fig. 1.8 Console window after quitting the R editor



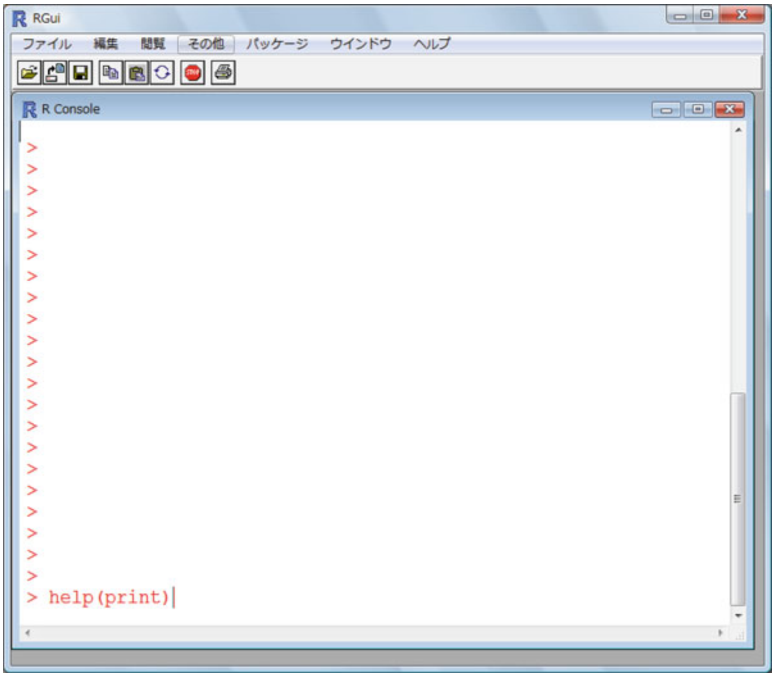
```
Platform: i386-pc-mingw32/i386 (32-bit)

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

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

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

[Previously saved workspace restored]

> fix(prog1)
> prog1()
```

Fig. 1.9 Executing `prog1()`



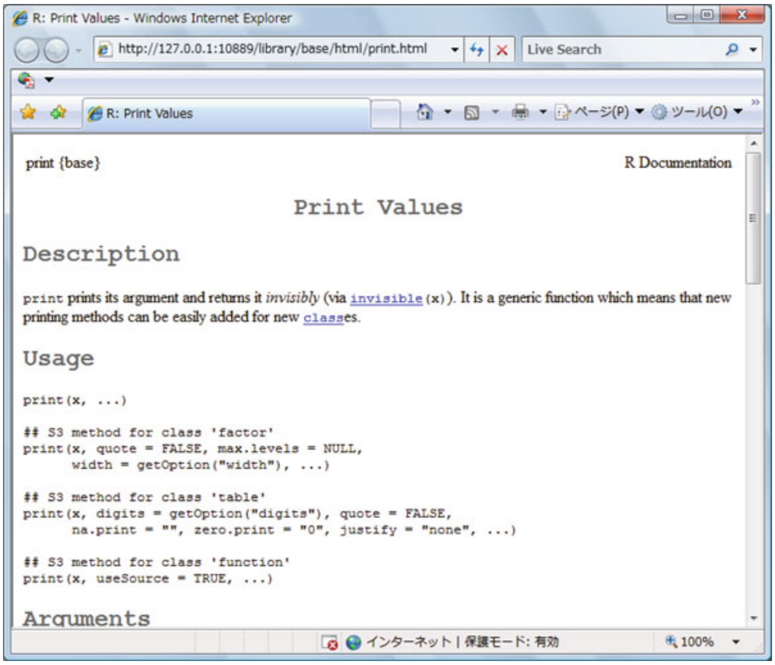
```
R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

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

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

[Previously saved workspace restored]

> fix(prog1)
> prog1()
[1] -5
>
```

Fig. 1.10 Result of executing `prog1()`



```
>
>
>
>
>
>
>
>
>
>
>
>
>
>
>
>
>
>
>
>
>
> help(print)
```

Fig. 1.11 Requesting help on the `print` command


R: Print Values - Windows Internet Explorer
http://127.0.0.1:10889/library/base/html/print.html
Live Search
R: Print Values
ページ(P)
ツール(O)
インターネット | 保護モード: 有効
100%

print {base} R Documentation

Print Values

Description

print prints its argument and returns it *invisibly* (via invisible(x)). It is a generic function which means that new printing methods can be easily added for new classes.

Usage

```
print(x, ...)

## S3 method for class 'factor'
print(x, quote = FALSE, max.levels = NULL,
      width = getOption("width"), ...)

## S3 method for class 'table'
print(x, digits = getOption("digits"), quote = FALSE,
      na.print = "", zero.print = "0", justify = "none", ...)

## S3 method for class 'function'
print(x, useSource = TRUE, ...)
```

Arguments

Fig. 1.12 Explanation of the `print()` command

updated R program could be lost by an abnormal termination of the R software. You should therefore, quit R and then restart it to save the R programs you have created or updated in the workspace image file (that is, on a hard disk, CD, or DVD); this prevents them from being lost by an abnormal termination of the R software. It is advisable to save the workspace before executing an R program that could cause the R software to terminate abnormally. You can also save the workspace image by selecting "Save Workspace" from the File menu. This avoids having to exit R and restart it.

Executing `help()` in the console window displays information on the R command given as the argument in the parentheses (Fig. 1.11). For example, if you need information on `print`, execute `help(print)`. This operation displays an explanation of the `print` command (Fig. 1.12).

1.2 Vectors

The subsequent sections in this chapter describe basic concepts in linear algebra and their numerical handling using R. These concepts and techniques form the basis of understanding statistical calculation and the implementation thereof in R.

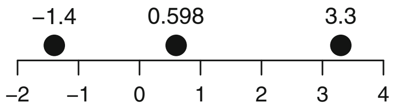


Fig. 1.13 A number line

Numbers such as -1.4, 0.598, and 3.3 are called scalars. They are represented as points on a number line (Fig. 1.13). Plain lower case letters (small letters) such as a, b, x, and y are used to denote variables storing scalar values. This notation is henceforth adopted in this book.

On the other hand, a combination of more than one number is often treated as a single entity. For example, if the account balance of country A is minus 20,000 dollars, and that of country B is 50,000 dollars, it is natural to represent this as $(-20000, 50000)$. In this notational convention, -20000 and 50000 are called elements (or components) of a vector. Notation such as $(-20000, 50000)$ is called the element form or component form). The combination of two numbers is referred to as a 2-dimensional vector, while the combination of three numbers is referred to as a 3-dimensional vector.

If a variable is required to represent $(-20000, 50000)$, a lower case letter in bold face such as $\mathbf{a}$, $\mathbf{b}$, $\mathbf{x}$, or $\mathbf{y}$ is commonly used. In high school mathematics, plain lower case letters with arrows on top such as $\overrightarrow{a}$, $\overrightarrow{b}$, $\overrightarrow{x}$, or $\overrightarrow{y}$ are usually used to denote vectors. In undergraduate mathematics, on the other hand, letters in bold type such as $\mathbf{a}$, $\mathbf{b}$, $\mathbf{x}$, or $\mathbf{y}$ are commonly used for this purpose. This notational difference arises because a vector is considered to be a concept representing movement from one point to another in high school mathematics, whereas it is understood as an abstract concept based on axioms in university undergraduate or higher level mathematics. Nevertheless, in pure mathematics at an undergraduate or higher level, vectors usually do not need to be discriminated from scalars, and hence both scalars and vectors are denoted by a, b, x, and y. Furthermore, if plain type and bold type are used to distinguish scalars and vectors, respectively, text using this notation may be hard to read and proofread. Therefore, letters such as $\underline{a}$, $\underline{b}$, $\underline{x}$, and $\underline{y}$ are sometimes used to depict vectors [1]. In R, scalars are treated as vectors with one element, and hence scalars and vectors are denoted similarly.

A two-dimensional vector is represented as an arrow on a flat plain. Figure 1.14 (left) illustrates a vector, the element form of which is $(4, 4.5)$, while Fig. 1.14 (right) shows a vector, the element form of which is $(-4, -4.5)$. The length of a vector is expressed as $|\mathbf{a}|$. If the element form of $\mathbf{a}$ is (a_1, a_2), it is defined as

$$|\mathbf{a}| = \sqrt{a_1^2 + a_2^2}. \tag{1.1}$$

If $\mathbf{a}$ is a q-dimensional vector with elements $(a_1, a_2, \ldots, a_q)$, $|\mathbf{a}|$ is written as

$$|\mathbf{a}| = \sqrt{\sum_{i=1}^{q} a_i^2}. \tag{1.2}$$

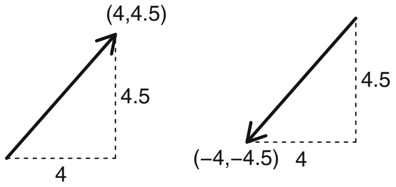


Fig. 1.14 Examples of vectors: (*left*) vector $(4, 4.5)$, and (*right*) vector $(-4, -4.5)$

R Program [1 - 1]

The element form is used when a vector is constructed in R. For example, to construct vector `aa` with two elements (−1.5, −4), the appropriate R program is:

```
vec1()
  function ()
  {
    aa <- c(-1.5, -4)
  }
```

Here, `c` denotes the start of the combination of elements, and thus, in this case, `c` must be a lower case letter. Since `c` is always used in this way, `c` cannot be used as the name of a variable. For the same reason, names such as `sin`, `cos`, `log`, and `if` have specific meanings, and hence problems are likely to occur if such names are used to denote variables.

The content of `aa` is displayed by the R program:

```
vec2()
  function ()
  {
    aa <- c(-1.5, -4)
    print(aa)
  }
```

Execution of `vec2()` gives the following results. The `()` in the command `vec2()` is used to accommodate arguments when executing the program. Even if no arguments are given, `()` is required.

```
  -1.5 -4.0
```

R Program [1 - 1] End

Both addition and subtraction of vectors is defined. For example, assume two vectors: **a** with elements $(-3, 5)$, and **b** with elements $(1, -9)$. Then, $\mathbf{a} + \mathbf{b}$ gives $(-2, -4)$. That is, the corresponding elements in the vectors are added. As another example, $\mathbf{a} - \mathbf{b}$ gives $(-4, 14)$. Since corresponding elements are needed for the addition or subtraction operations, addition and subtraction between $(-3, 5)$ (a vector consisting of two elements) and $(1, -9, 8)$ (a vector consisting of three elements) cannot be realized. Furthermore, addition or subtraction between $(-3, 5)$ (a vector) and 2.5 (a scalar) cannot be done.

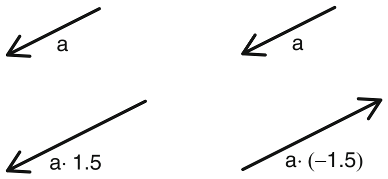


Fig. 1.15 Multiplication between a vector and a scalar: ($\mathbf{a} \cdot 1.5$) (*left*) and ($\mathbf{a} \cdot (-1.5)$) (*right*)

Multiplication between a vector and a scalar, however, can be implemented. Let $\mathbf{a}$ be a 2-dimensional vector (e.g., $\mathbf{a} = (a_1, a_2)$), and q be a scalar. Then,

$$q\mathbf{a} = \mathbf{a}q = (qa_1, qa_2). \tag{1.3}$$

If the scalar is positive, the direction of the vector remains the same (Fig. 1.15 (left)). If the scalar is negative, on the other hand, the direction of the resulting vector changes to the opposite direction (Fig. 1.15 (right)). Definitions for a vector with more than 2 dimensions are similar to those in Eq. (1.3).

The value representing the relationship between two vectors is referred to as the inner product. Suppose the element form of $\mathbf{a}$ is (a_1, a_2), and that of $\mathbf{b}$ is (b_1, b_2). Then the inner product of $\mathbf{a}$ and $\mathbf{b}$ is written as $\mathbf{a} \cdot \mathbf{b}$. This is defined as

$$\mathbf{a} \cdot \mathbf{b} = a_1 b_1 + a_2 b_2. \tag{1.4}$$

If $\mathbf{a}$ and $\mathbf{b}$ are q-dimensional vectors with elements $(a_1, a_2, \ldots, a_q)$ and $(b_1, b_2, \ldots, b_q)$, respectively, $\mathbf{a} \cdot \mathbf{b}$ is written as

$$\mathbf{a} \cdot \mathbf{b} = \sum_{i=1}^{q} a_i b_i. \tag{1.5}$$

Another definition of $\mathbf{a} \cdot \mathbf{b}$ is given below using θ, which is the angle between $\mathbf{a}$ and $\mathbf{b}$ (Fig. 1.16).

$$\mathbf{a} \cdot \mathbf{b} = |\mathbf{a}|\ |\mathbf{b}| \cos(\theta). \tag{1.6}$$

Equations (1.4) and (1.6) formulate

$$\cos(\theta) = \frac{a_1 b_1 + a_2 b_2}{|\mathbf{a}|\ |\mathbf{b}|}. \tag{1.7}$$

This relationship yields the angle between two vectors.

Equation (1.6) is valid if $\mathbf{a}$ and $\mathbf{b}$ are vectors with more than two dimensions. Hence, Eq. (1.7) is a similar equation if $\mathbf{a}$ and $\mathbf{b}$ have more than two dimensions.

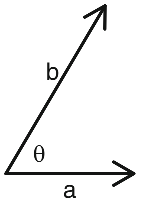


Fig. 1.16 Definition of θ

R Program [1 - 2]

Addition and subtraction of vectors and displaying the results can be implemented in R as follows:

```
vec3()
  function ()
  {
    aa <- c(8, 7)
    bb <- c(-5, 3)
    cc <- aa + bb
    dd <- aa - bb
    print(cc)
    print(dd)
  }
```

The result is:

```
   3 10
  13  4
```

As mentioned above, addition or subtraction between a scalar and a vector cannot be implemented. However, the R program given below can be executed.

```
vec4()
  function ()
  {
    aa <- c(8, 7)
    bb <- -5
    cc <- aa + bb
    dd <- aa - bb
    print(cc)
    print(dd)
  }
```

The result is:

```
   3  2
  13 12
```

In other words, although `bb` denotes the scalar `-5`, it can be transformed into `c(-5, -5)` when addition or subtraction between `aa` and `bb` is carried out. This is because `aa` is a 2-dimensional vector.

Addition or subtraction between a 2-dimensional vector and a 3-dimensional vector cannot be executed either. The following program shows what happens in R.

```
vec5()
  function ()
  {
    aa <- c(8, 7)
    bb <- c(-5, 6, 10)
    cc <- aa + bb
    dd <- aa - bb
    print(cc)
    print(dd)
  }
```

This R program yields:

```
   3 13 18
  13  1 -2
  Warning messages:
  1: In aa + bb :
    longer object length is not a multiple of shorter
     object length
  2: In aa - bb :
    longer object length is not a multiple of shorter
     object length
```

`aa` is changed to `c(8, 7, 8)` when addition or subtraction between `aa` and `bb` is carried out. However, the following warning message regarding the number of elements of `aa` and `bb` appears: "Longer object length is not a multiple of shorter object length". Executing `vec4()`, on the contrary, does not result in such a warning message because the number of elements of `aa` is 2, and that of `bb` is 1. In each case, if the lengths of the vectors in an addition or subtraction operation differ, their lengths are equalized by repeating certain elements in the shorter vector. Hence, it is safer to confirm that the lengths of the vectors are the same before addition or subtraction is carried out.

R Program [1-2] End

R Program [1-3]

Multiplication between a vector and a scalar is carried out as follows:

```
vec11()
  function ()
  {
    aa <- c(8, 7)
    qq <- -1.5
    cc <- aa * qq
    dd <- qq * aa
    print(cc)
    print(dd)
  }
```

This program yields:

```
  -12.0 -10.5
  -12.0 -10.5
```

R Program [1 - 3] End

R Program [1 - 4]

The inner product between two vectors is calculated in the following way:

```
vec21()
  function ()
  {
    aa <- c(1, -3)
    bb <- c(-2, 9)
    inn <- aa %*% bb
    print(inn)
  }
```

`%*%` calculates the inner product between two vectors. The result is:

```
  -29
```

The inner product between vectors with different dimensions cannot be calculated. For example, calculation of the inner product between a 2-dimensional vector and a 3-dimensional vector is attempted in the following R program:

```
vec22()
  function ()
  {
    aa <- c(1, -3)
    bb <- c(-2, 9, 1)
    inn <- aa %*% bb
    print(inn)
  }
```

The result is:

```
  Error in aa %*% bb : non-conformable arguments
```

R Program [1 - 4] End

1.3 Matrices

Examples of matrices are given below:

$$\mathbf{A} = \begin{pmatrix} -3.4 \\ 1.5 \end{pmatrix}, \qquad \mathbf{B} = \begin{pmatrix} 2 & 9.9 \end{pmatrix}, \tag{1.8}$$

$$\mathbf{C} = \begin{pmatrix} -1 & 4 \\ 1.5 & 4.2 \end{pmatrix}, \qquad \mathbf{X} = \begin{pmatrix} 7 & -2.7 \\ -0.5 & -9.4 \\ 5 & 2.7 \end{pmatrix}, \tag{1.9}$$

$$\mathbf{Y} = \begin{pmatrix} 7 & -2.7 & 2 \\ -0.5 & -9.4 & 1.1 \end{pmatrix}, \qquad \mathbf{Z} = \begin{pmatrix} 0.2 & -3.7 & 1.3 \\ 5 & 1.4 & -2.8 \\ 9 & 7.2 & 4.1 \end{pmatrix}. \tag{1.10}$$

A variable representing a matrix is usually denoted by a capital letter in bold face, such as **A**, **B**, **C**, **X**, **Y**, and **Z**. This notation is henceforth adopted in this book. In the above example, **A** is a matrix with 2 rows and 1 column. Simply put, it is a 2×1 matrix; this is read as "2-by-1 matrix". In other words, the size of this matrix is 2 $\times$ 1, or the dimension of this matrix is 2 $\times$ 1. Similarly, **B** is a 1 $\times$ 2 matrix, **C** is a 2 $\times$ 2 matrix, **X** is a 3 $\times$ 2 matrix, **Y** is a 2 $\times$ 3 matrix, and **Z** is a 3 $\times$ 3 matrix. That is, the number of elements aligned vertically gives the number of rows, while the number of elements aligned horizontally specifies the number of columns. If the number of rows in a matrix is the same as the number of columns, such as **C**, the matrix is called a square matrix.

A is a matrix consisting of only a single column; such a matrix is called a column vector. Likewise, **B** is a matrix consisting of only a single row; such a matrix is called a row vector. The image of a column in a newspaper serves as a visual clue to remember the definition of a column vector, because both of these are vertically long. On the other hand, the image of a rowing boat serves to remember the definition of a row vector because both of these are horizontally long.

The term "vector" on its own, is sometimes used to refer to both these kinds of vectors. In older literature, the term "vector" was used to refer to a row vector, which is not its current meaning; "vector" is referred to a column vecor in newer literature. Hence, the element form is something like **A** in Eq. (1.8). It should be noted that descriptions of vectors such as $(-3.4, 1.5)$ or $(-3.4 \;\; 1.5)$ (no commas between numbers) are ecological sound owing to the fact that they use less paper. Hence, expressions such as $\mathbf{A} = (-3.4 \;\; 1.5)^t$, $\mathbf{A} = (-3.4 \;\; 1.5)^T$, or $\mathbf{A} = (-3.4 \;\; 1.5)'$ are often used. In these examples, the symbols "^{t}", "^{T}", and "$'$" indicate a transposed matrix. Both "^{t}" and "^{T}" refer to the initial letter of "transpose". Some examples are given below:

$$\mathbf{A} = \begin{pmatrix} -3.4 \\ 1.5 \end{pmatrix} = \begin{pmatrix} -3.4 & 1.5 \end{pmatrix}^t, \qquad \mathbf{A}^t = \begin{pmatrix} -3.4 \\ 1.5 \end{pmatrix}^t = \begin{pmatrix} -3.4 & 1.5 \end{pmatrix}. \tag{1.11}$$

Because $\mathbf{A}^t$ representing the transposed matrix of **A** could be mistaken for the t-th power of **A**, the notation "${}^t\mathbf{A}$" is occasionally adopted, although not in this book.

If the elements of a matrix are variables, the notation given below is commonly used.

$$\mathbf{C} = \begin{pmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \end{pmatrix}, \qquad \mathbf{X} = \begin{pmatrix} x_{11} & x_{12} \\ x_{21} & x_{22} \\ x_{31} & x_{32} \end{pmatrix}. \tag{1.12}$$

In other words, the elements of a matrix are represented as plain lower case letters with subscripts. The subscripts of an element in a matrix or vector are called suffixes. The first suffix is the index of the row to which the element belongs, while the second is the index of the column to which the element belongs. The order of the suffixes is "which row" followed by "which column"; the word "rococo style", which sounds like an abbreviated form of "row and column style", can be used to remember this. Using this notation with reference to variable c, expression of c_{11} as an element of $\mathbf{C}$ could cause confusion. To avoid this problem, c_{11} is sometimes written as $[\mathbf{C}]_{11}$.

R Program [1 - 5]

In R, a vector, a column vector, and a row vector are distinguished.

```
vec31()
  function ()
  {
  # (1)
    xx <- c(1, -3)
    print("xx")
    print(xx)
  # (2)
    xxc <- matrix(xx, ncol = 1)
    print("xxc")
    print(xxc)
  # (3)
    xxr <- matrix(xx, nrow = 1)
    print("xxr")
    print(xxr)
  }
```

R treats # and everything that follows until the end of the line, as a comment. A comment does not affect the behavior of the R program. However, when # is surrounded by quotation marks ("), it is regarded as a letter.

(1) `xx` is defined as a vector with elements $(1, -3)$. Next, `xx` is output as:

```
[1] "xx"
[1]  1 -3
```

Since "`[1]`" appears to the left of the elements only, it indicates that this is a vector, which can be regarded as a series of numbers, and not a column vector or a row vector.

(2) Transformation of `xx` into a column vector gives `xxc`. This is the result of executing `matrix(xx, ncol = 1)`. In this R command, `ncol=1` specifies that a matrix with `1` column is constructed. The column vector in `xxc` is output as:

```
        [,1]
[1,]      1
[2,]     -3
```

`[,1]` means that the first column is displayed below. `[1,]` means that the element displayed immediately to the right is that in the first row. Likewise, `[2,]` means that the element displayed to the right is that in the second row.

(3) Transformation of `xx` into a row vector yields `xxr`. `nrow = 1` specifies that a matrix with `1` row is constructed. The row vector of `xxr` is output as:

```
         [,1]  [,2]
    [1,]    1    -3
```

`[,1]` indicates that the element in the first column is given below. `[,2]` indicates that the element in the second column is given below. `[1,]` means that the elements displayed immediately to the right are those in the first row.

R Program [1 - 5] End

R Program [1 - 6]

In R, applying `matrix()` to a vector results in diverse matrices. Elements, column vectors, or row vectors can be extracted from the resultant matrices.

```
vec36()
  function ()
  {
  # (1)
    yy <- c(-1, 9, 8, -4)
    print("yy")
    print(yy)
  # (2)
    yym <- matrix(yy, ncol = 2)
    print("yym")
    print(yym)
  # (3)
    yym21 <- yym[2, 1]
    print("yym21")
    print(yym21)
  # (4)
    yym2r <- yym[2, , drop = F]
    print("yym2r")
    print(yym2r)
  # (5)
    yym2r_drop <- yym[2, ]
    print("yym2r_drop")
    print(yym2r_drop)
  # (6)
    yym1c <- yym[,1, drop = F]
    print("yym1c")
    print(yym1c)
  # (7)
    yym1c_drop <- yym[ ,1]
    print("yym1c_drop")
    print(yym1c_drop)
  }
```

(1) yy denotes a vector consisting of 4 elements. The vector is displayed as follows:

```
[1] "yy"
[1] -1  9  8 -4
```

(2) Applying `matrix()` to `yy` gives a 2×2 matrix, called `yym`. In `matrix()`, `ncol = 2` specifies that the matrix comprises two column vectors. If `matrix()` is used to transform a vector into a matrix, the elements of the vector are aligned as vertical column vectors in the construction of the matrix. Using `matrix(yy,`
`ncol = 2, byrow = T)`, however, the elements of the vector are aligned as horizontal row vectors in constructing the matrix. Matrix `yym` is displayed below:

```
[1] "yym"
      [,1] [,2]
[1,]    -1    8
[2,]     9   -4
```

(3) The (2, 1) element of `yym` (y_{21}, if the elements of `yym` are referred to as $\{y_{ij}\}$) is extracted and stored in `yym21`, which is displayed below:

```
[1] "yym21"
[1] 9
```

(4) The second row of `yym` is extracted as a row vector and stored in `yym2r`. If "`, drop = F`" is omitted, the extracted vector is not a row vector, but merely a vector. `F` can be written as `FALSE`, since both `F` and `FALSE` mean negative in an R program. This means that a column vector is not transformed into merely a vector. On the other hand, `T` or `TRUE` means the opposite of `FALSE`, in other words, positive.
`yym2r` is displayed below:

```
[1] "yym2r"
      [,1] [,2]
[1,]     9   -4
```

(5) The second row of `yym` is extracted as a row vector without "`, drop = F`". The result is stored in `yym2r_drop`. On outputting `yym2r_drop`, we see that it is not a matrix, but a vector:

```
[1] "yym2r_drop"
[1]  9 -4
```

(6) The first column in `yym` is extracted as a column vector and stored in `yym1c`. If "`, drop = F`" is omitted, the extracted vector is not a column vector, but only a vector. `yym1c` is displayed below:

```
[1] "yym1c"
     [,1]
[1,]   -1
[2,]    9
```

(7) When the first column of `yym` is extracted as a column vector, "`, drop = F`" is not specified. The result is stored in `yym1c_drop`. On outputting `yym1c_drop`, we see that it is not a matrix, but a vector:

```
[1] "yym1c_drop"
[1] -1  9
```

R Program [1 - 6] End

1.4 Addition of Two Matrices

Addition of two matrices is defined as

$$\begin{pmatrix} a_{11} & a_{12} \end{pmatrix} + \begin{pmatrix} b_{11} & b_{12} \end{pmatrix} = \begin{pmatrix} a_{11} + b_{11} & a_{12} + b_{12} \end{pmatrix}, \tag{1.13}$$

$$\begin{pmatrix} a_{11} \\ a_{21} \end{pmatrix} + \begin{pmatrix} b_{11} \\ b_{21} \end{pmatrix} = \begin{pmatrix} a_{11} + b_{11} \\ a_{21} + b_{21} \end{pmatrix}, \tag{1.14}$$

$$\begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} + \begin{pmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{pmatrix} = \begin{pmatrix} a_{11} + b_{11} & a_{12} + b_{12} \\ a_{21} + b_{21} & a_{22} + b_{22} \end{pmatrix}. \tag{1.15}$$

Subtraction between two matrices is defined in a similar way.

R Program [1 - 7]

Addition and subtraction between two matrices is carried out using Eq. (1.13) (page 18) and Eq. (1.14) (page 18).

```
vec38()
  function ()
  {
  # (1)
    pp <- c(2, -9)
    ppr <- matrix(pp, ncol = 2)
    print("ppr")
    print(ppr)
  # (2)
    qq <- c(-5, -4)
    qqr <- matrix(qq, ncol = 2)
    print("qqr")
    print(qqr)
  # (3)
    rrr <- ppr + qqr
    print("rrr")
    print(rrr)
  # (4)
    ppc <- t(ppr)
    print("ppc")
    print(ppc)
  # (5)
    qqc <- t(qqr)
    print("qqc")
    print(qqc)
  # (6)
    rrc <- ppc - qqc
    print("rrc")
    print(rrc)
  }
```

(1) Vector `pp` is transformed into a row vector, `ppr`. The resulting `ppr` is output as:

```
[1] "ppr"
     [,1] [,2]
[1,]    2   -9
```

(2) Vector `qq` is transformed into a row vector, `qqr`. The resulting `qqr` is output as:

```
[1] "qqr"
     [,1] [,2]
[1,]   -5   -4
```

(3) The result of `ppr + qqr` is stored in `rrr`, which is output:

```
[1] "rrr"
     [,1] [,2]
[1,]   -3  -13
```

(4) The transposed matrix of `ppr` is obtained and the result is stored in `ppc`. `t()` creates a transposed matrix. `ppc` is output:

```
[1] "ppc"
     [,1]
[1,]    2
[2,]   -9
```

(5) The transposed matrix of `qqr` is obtained and the result is stored in `qqc`, which is output:

```
[1] "qqc"
     [,1]
[1,]   -5
```

```
     [2,]    -4
```

(6) The result of `ppc - qqc` is stored in `rrc` and output as:

```
     [1] "rrc"
          [,1]
     [1,]     7
     [2,]    -5
```

R Program [1-7] End

R Program [1-8]

Addition of two matrices is carried out using Eq. (1.15) (page 18).

```
vec39()
  function ()
  {
  # (1)
    pp <- c(-3, -1)
    ppc <- matrix(pp, ncol = 1)
    print("ppc")
    print(ppc)
  # (2)
    qq <- c(-5, -4)
    qqc <- matrix(qq, ncol = 1)
    print("qqc")
    print(qqc)
  # (3)
    pqmat <- cbind(ppc, qqc)
    print("pqmat")
    print(pqmat)
  # (4)
    rr <- c(7, -8)
    rrc <- matrix(rr, ncol = 1)
    print("rrc")
    print(rrc)
  # (5)
    ss <- c(-2, 9)
    ssc <- matrix(ss, ncol = 1)
    print("ssc")
    print(ssc)
  # (6)
    rsmat <- cbind(rrc, ssc)
    print("rsmat")
    print(rsmat)
  # (7)
    sum1 <- pqmat + rsmat
    print("sum1")
    print(sum1)
  }
```

(1) Vector pp is transformed into a column vector, ppc, which is output:

```
[1] "ppc"
     [,1]
[1,]   -3
[2,]   -1
```

(2) Vector qq is transformed into a column vector, qqc, which is output:

```
[1] "qqc"
     [,1]
[1,]   -5
[2,]   -4
```

(3) cbind() combines ppc and qqc to create a 2×2 matrix. The result is stored in pqmat and output:

```
[1] "pqmat"
     [,1] [,2]
[1,]   -3   -5
[2,]   -1   -4
```

The initial letter “c” in cbind() means “by column”. Hence, cbind() combines two matrices horizontally by placing column vectors together. For example, combining a 2×2 matrix and a 2×2 matrix using cbind() results in a 2×4 matrix. Hence, the column numbers in the two matrices become the serial number (i.e., [,1] [,2] [,3] [,4]).

(4) Vector rr is transformed into a column vector rrc, which is output:

```
[1] "rrc"
     [,1]
[1,]    7
[2,]   -8
```

(5) Vector ss is transformed into a column vector ssc, which is output:

```
[1] "ssc"
     [,1]
[1,]   -2
[2,]    9
```

(6) Combining rrr and ssr using cbind() yields a 2×2 matrix. The resultant matrix is stored in rsmat and output as:

```
[1] "rsmat"
     [,1] [,2]
[1,]    7   -2
[2,]   -8    9
```

(7) The result of `pqmat + rsmat` is stored in `sum1` and output as:

```
[1] "sum1"
     [,1] [,2]
[1,]    4   -7
[2,]   -9    5
```

R Program [1 - 8] End

1.5 Multiplying Two Matrices

Multiplication of two matrices is defined as

$$\begin{pmatrix} a_{11} & a_{12} \end{pmatrix} \begin{pmatrix} b_{11} \\ b_{21} \end{pmatrix} = a_{11}b_{11} + a_{12}b_{21}, \tag{1.16}$$

$$\begin{pmatrix} a_{11} \\ a_{21} \end{pmatrix} \begin{pmatrix} b_{11} & b_{12} \end{pmatrix} = \begin{pmatrix} a_{11}b_{11} & a_{11}b_{12} \\ a_{21}b_{11} & a_{21}b_{12} \end{pmatrix}, \tag{1.17}$$

$$\begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \begin{pmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{pmatrix} = \begin{pmatrix} a_{11}b_{11} + a_{12}b_{21} & a_{11}b_{12} + a_{12}b_{22} \\ a_{21}b_{11} + a_{22}b_{21} & a_{21}b_{12} + a_{22}b_{22} \end{pmatrix}. \tag{1.18}$$

Let us make this more general. Suppose that **A** is a $l \times m$ matrix with elements $a_{ij} (1 \le i \le l, 1 \le j \le m)$. Moreover, suppose that **B** is an $m \times n$ matrix with elements $b_{ij} (1 \le i \le m, 1 \le j \le n)$. If **AB** is stored in **C**, with the elements of **C** defined as $c_{ij} (1 \le i \le l, 1 \le j \le n)$, we have

$$c_{ij} = \sum_{k=1}^{m} a_{ik}b_{kj} \qquad (1 \le i \le l, 1 \le j \le n). \tag{1.19}$$

An easy way to remember this equation is to note that in the suffixes of c_{ij} on the left hand side and those of $a_{ik}b_{kj}$ on the right hand side, k is placed next to another k in $a_{ik}b_{kj}$, whereas c_{ij} does not refer to k, because $\sum_{k=1}^{m}$ represents the summation with respect to k. Moreover, if multiplication of two matrices **A** and **B** is possible, the following transposition equation holds.

$$(\mathbf{AB})^t = \mathbf{B}^t\mathbf{A}^t. \tag{1.20}$$

This equation can be verified as follows:

$$[(\mathbf{AB})^t]_{ij} = [\mathbf{AB}]_{ji}$$

$$
\begin{aligned}
&= \sum_{k=1}^{m} [\mathbf{A}]_{jk} [\mathbf{B}]_{ki} \\
&= \sum_{k=1}^{m} [\mathbf{B}]_{ki} [\mathbf{A}]_{jk} \\
&= \sum_{k=1}^{m} [\mathbf{B}^t]_{ik} [\mathbf{A}^t]_{kj} \\
&= [\mathbf{B}^t \mathbf{A}^t]_{ij} . \qquad (1.21)
\end{aligned}
$$

$\boxed{\text{R Program [1 - 9]}}$

By using Eqs. (1.16) (page 22) and (1.17) (page 22), we can perform multiplication of matrices.

```
vec41()
  function ()
  {
  # (1)
    pp <- c(2, -9)
    ppr <- matrix(pp, ncol = 2)
    print("ppr")
    print(ppr)
  # (2)
    qq <- c(-5, -4)
    qqc <- matrix(qq, ncol = 1)
    print("qqc")
    print(qqc)
  # (3)
    rr <- ppr %*% qqc
    print("rr")
    print(rr)
  # (4)
    pp <- c(2, -9)
    ppc <- matrix(pp, ncol = 1)
    print("ppc")
    print(ppc)
  # (5)
    qq <- c(-5, -4)
    qqr <- matrix(qq, ncol = 2)
    print("qqr")
    print(qqr)
  # (6)
    rrm <- ppc %*% qqr
    print("rrm")
    print(rrm)
  }
```

(1) Vector `pp` is transformed into a row vector, `ppr`, which is output as:

```
[1] "ppr"
     [,1] [,2]
[1,]    2   -9
```

(2) Vector `qq` is transformed into a column vector, `qqc`, which is output as:

```
[1] "qqc"
     [,1]
[1,]   -5
[2,]   -4
```

(3) The result of `ppr %*% qqc` is stored in `rr`, which is output. `%*%` can be used to multiply two matrices as well as two vectors (i.e., inner product).

```
[1] "rr"
     [,1]
[1,]   26
```

(4) Vector `pp` is transformed into a column vector, `ppc`, which is output as:

```
[1] "ppc"
     [,1]
[1,]    2
[2,]   -9
```

(5) Vector `qq` is transformed into a row vector `qqr`, which is output as:

```
[1] "qqr"
     [,1] [,2]
[1,]   -5   -4
```

(6) `%*%` multiplies two matrices. The result of `ppc %*% qqr` is stored in `rrm`, which is output as:

```
[1] "rrm"
     [,1] [,2]
[1,]  -10   -8
[2,]   45   36
```

R Program [1 - 9] End

R Program [1 - 10]

Using Eq. (1.18) (page 22), multiplication of matrices can be performed.

```
vec46()
  function ()
  {
  # (1)
    aa <- c(1, -3, 4, 2)
    aam <- matrix(aa, ncol = 2)
    print("aam")
    print(aam)
  # (2)
    bb <- c(-3, 1, 6, -4)
    bbm <- matrix(bb, ncol = 2)
    print("bbm")
    print(bbm)
  # (3)
    ccm <- aam %*% bbm
    print("ccm")
    print(ccm)
  # (4)
    ddm <- bbm %*% aam
    print("ddm")
    print(ddm)
  }
```

(1) Matrix `aam` is constructed and output as follows:

```
[1] "aam"
     [,1] [,2]
[1,]    1    4
[2,]   -3    2
```

(2) Matrix `bbm` is composed and output as follows:

```
[1] "bbm"
     [,1] [,2]
[1,]   -3    6
[2,]    1   -4
```

(3) The result of `aam %*% bbm` is stored in `ccm`, and output:

```
[1] "ccm"
     [,1] [,2]
[1,]    1  -10
[2,]   11  -26
```

(4) The result of `bbm %*% aam` is stored in `ddm` and output. Comparing `ddm` with the result of (3), `ccm`, shows the difference between `bbm %*% aam` and `aam %*% bbm`.

```
[1] "ddm"
     [,1] [,2]
[1,]  -21    0
[2,]   13   -4
```

R Program [1 - 10] End

R Program [1 - 11]

An example where multiplication between two matrices cannot be carried out is shown below.

```
vec51()
  function ()
  {
  # (1)
    pp <- c(2, -9)
    ppc <- matrix(pp, ncol = 1)
    print("ppc")
    print(ppc)
  # (2)
    qq <- c(-5, -4)
    qqc <- matrix(qq, ncol = 1)
    print("qqc")
    print(qqc)
  # (3)
    rr <- ppc %*% qqc
    print("rr")
    print(rr)
  }
```

(1) Vector `pp` is transformed into a column vector, `ppc`, which is output as follows:

```
[1] "ppc"
     [,1]
[1,]    2
[2,]   -9
```

(2) Vector `qq` is transformed into a column vector, `qqc`, which is output as follows:

```
[1] "qqc"
     [,1]
[1,]   -5
[2,]   -4
```

(3) The result of `ppc %*% qqc` is stored in `rr` and output. Since a column vector cannot be multiplied by another column vector, the result of `vec51()` is:

```
Error in ppc %*% qqc : non-conformable arguments
```

R Program [1 - 11] End

R Program [1 - 12]

Next, we verify Eq. (1.20) (page 22).

```
vec56()
  function ()
  {
  # (1)
    aa <- c(2, -9, 1, 3, -3, 6)
    aam <- matrix(aa, ncol = 2)
    bb <- c(3, 4, -5, -9, 8, 4)
    bbm <- matrix(bb, ncol = 3)
  # (2)
    ccm <- t(aam %*% bbm)
    print("ccm")
    print(ccm)
    ddm <- t(bbm) %*% t(aam)
    print("ddm")
    print(ddm)
  }
```

(1) `aam` and `bbm` are given.
(2) Both sides of Eq. (1.20) are calculated and the result is displayed.

`vec56()` yields:

```
[1] "ccm"
     [,1] [,2] [,3]
[1,]   18  -39   27
[2,]  -37   72  -59
[3,]   28  -84   32
[1] "ddm"
     [,1] [,2] [,3]
[1,]   18  -39   27
[2,]  -37   72  -59
[3,]   28  -84   32
```

R Program [1 - 12] End

1.6 Identity and Inverse Matrices

Any number (scalar) multiplied by 1 is equal to the original number. In the context of matrices, the identity matrix plays much the same role as unity (1). An identity matrix is usually denoted by **I**, following the initial letter of "identity". It can also be denoted by **E**, following the initial letter of "elementary". Examples of identity matrices are:

$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \quad \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}. \tag{1.22}$$

In other words, an identity matrix is a square matrix, where all its diagonal elements are 1, and all the remaining elements are 0. If the element of a square matrix (**X**) is described as x_{ij}, the values of i and j are the same for the diagonal elements. A 2×2 identity matrix is sometimes written as $\mathbf{I}_2$, while a 3×3 identity matrix is sometimes written as $\mathbf{I}_3$.

If the result of multiplying two scalar numbers (a and b) is 1, b is called the inverse number (reciprocal number) of a. For example, 0.2 is the inverse of 5. In the context of matrices, if the following relationship between two square matrices (**A** and **B**) holds, **B** is called the inverse matrix of **A**.

$$\mathbf{AB} = \mathbf{BA} = \mathbf{I} \tag{1.23}$$

This relationship can also be written as

$$\mathbf{A}^{-1} = \mathbf{B}. \tag{1.24}$$

It should be noted that some matrices have inverse matrices, while others do not. A matrix that has an inverse matrix is called a regular matrix.

For example, let **A** and **B** be defined as

$$\mathbf{A} = \begin{pmatrix} -2 & 1 \\ -8 & 5 \end{pmatrix}, \quad \mathbf{B} = \begin{pmatrix} -2.5 & 0.5 \\ -4 & 1 \end{pmatrix}. \tag{1.25}$$

Then, we have

$$\begin{pmatrix} -2 & 1 \\ -8 & 5 \end{pmatrix} \begin{pmatrix} -2.5 & 0.5 \\ -4 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \tag{1.26}$$

$$\begin{pmatrix} -2.5 & 0.5 \\ -4 & 1 \end{pmatrix} \begin{pmatrix} -2 & 1 \\ -8 & 5 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}. \tag{1.27}$$

Since Eq. (1.23) holds, **B** is the inverse matrix of **A**, and vice versa.

Equation (1.24) is represented in element form as

$$\begin{pmatrix} -2 & 1 \\ -8 & 5 \end{pmatrix}^{-1} = \begin{pmatrix} -2.5 & 0.5 \\ -4 & 1 \end{pmatrix}. \tag{1.28}$$

A 2×2 matrix (e.g., Eq. (1.28)), which consists of two rows, or alternatively stated, which consists of two columns, is represented as

$$\mathbf{A} = \begin{pmatrix} [\mathbf{A}]_{11} & [\mathbf{A}]_{12} \\ [\mathbf{A}]_{21} & [\mathbf{A}]_{22} \end{pmatrix}. \tag{1.29}$$

Then, $\mathbf{A}^{-1}$ (the inverse matrix of **A**) is written as

$$\mathbf{A}^{-1} = \frac{1}{[\mathbf{A}]_{11}[\mathbf{A}]_{22} - [\mathbf{A}]_{12}[\mathbf{A}]_{21}} \begin{pmatrix} [\mathbf{A}]_{22} & -[\mathbf{A}]_{12} \\ -[\mathbf{A}]_{21} & [\mathbf{A}]_{11} \end{pmatrix}. \tag{1.30}$$

The following calculation proves Eq. (1.30).

$$\begin{aligned}
\mathbf{AA}^{-1} &= \begin{pmatrix} [\mathbf{A}]_{11} & [\mathbf{A}]_{12} \\ [\mathbf{A}]_{21} & [\mathbf{A}]_{22} \end{pmatrix} \frac{1}{[\mathbf{A}]_{11}[\mathbf{A}]_{22} - [\mathbf{A}]_{12}[\mathbf{A}]_{21}} \begin{pmatrix} [\mathbf{A}]_{22} & -[\mathbf{A}]_{12} \\ -[\mathbf{A}]_{21} & [\mathbf{A}]_{11} \end{pmatrix} \\
&= \frac{1}{[\mathbf{A}]_{11}[\mathbf{A}]_{22} - [\mathbf{A}]_{12}[\mathbf{A}]_{21}} \begin{pmatrix} [\mathbf{A}]_{11} & [\mathbf{A}]_{12} \\ [\mathbf{A}]_{21} & [\mathbf{A}]_{22} \end{pmatrix} \begin{pmatrix} [\mathbf{A}]_{22} & -[\mathbf{A}]_{12} \\ -[\mathbf{A}]_{21} & [\mathbf{A}]_{11} \end{pmatrix} \\
&= \frac{1}{[\mathbf{A}]_{11}[\mathbf{A}]_{22} - [\mathbf{A}]_{12}[\mathbf{A}]_{21}} \begin{pmatrix} [\mathbf{A}]_{11}[\mathbf{A}]_{22} - [\mathbf{A}]_{12}[\mathbf{A}]_{21} & -[\mathbf{A}]_{11}[\mathbf{A}]_{12} + [\mathbf{A}]_{12}[\mathbf{A}]_{11} \\ [\mathbf{A}]_{21}[\mathbf{A}]_{22} - [\mathbf{A}]_{22}[\mathbf{A}]_{21} & -[\mathbf{A}]_{21}[\mathbf{A}]_{12} + [\mathbf{A}]_{22}[\mathbf{A}]_{11} \end{pmatrix} \\
&= \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}.
\end{aligned} \tag{1.31}$$

The following equation is also satisfied.

$$\mathbf{A}^{-1}\mathbf{A} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \tag{1.32}$$

Furthermore, if the size of square matrix **A** is the same as that of **B** (that is, both the number of rows and the number of columns in **A** are the same as those in **B**), and both these matrices have inverse matrices, the following equation for the multiplication of matrices and inverse matrices is satisfied.

$$(\mathbf{AB})^{-1} = \mathbf{B}^{-1}\mathbf{A}^{-1} \tag{1.33}$$

This equation can be proved as follows.

First, the following equation is defined.

$$(\mathbf{AB})^{-1} = \mathbf{P}. \tag{1.34}$$

It follows that if we take the inverse of both sides in the above equation, we obtain

$$\mathbf{AB} = \mathbf{P}^{-1}. \tag{1.35}$$

Multiplying both sides from the left by $\mathbf{A}^{-1}$ yields

$$\mathbf{B} = \mathbf{A}^{-1}\mathbf{P}^{-1}. \tag{1.36}$$

Then, by multiplying both sides from the right by **P**, we have

$$\mathbf{BP} = \mathbf{A}^{-1}. \tag{1.37}$$

Finally, multiplying both sides from the left by $\mathbf{B}^{-1}$ gives

$$\mathbf{P} = \mathbf{B}^{-1}\mathbf{A}^{-1}. \tag{1.38}$$

Thus, Eq. (1.33) holds.

Moreover, the following equation is satisfied:

$$(\mathbf{A}^t)^{-1} = (\mathbf{A}^{-1})^t. \tag{1.39}$$

This equation can be proved as follows.

First, we set

$$(\mathbf{A}^{-1})^t \mathbf{A}^t = \mathbf{Q}. \tag{1.40}$$

We transpose the matrix on the left hand side. Then, by applying Eq. (1.20) (page 22), we have

$$\mathbf{A}(\mathbf{A}^{-1}) = \mathbf{I}. \tag{1.41}$$

Therefore, the following equation holds:

$$\mathbf{Q}^t = \mathbf{I}. \tag{1.42}$$

By transposing the matrices on both sides, we have

$$\mathbf{Q} = \mathbf{I}. \tag{1.43}$$

Substituting this equation into Eq. (1.40) yields

$$(\mathbf{A}^{-1})^t \mathbf{A}^t = \mathbf{I}. \tag{1.44}$$

This equation indicates that $(\mathbf{A}^{-1})^t$ is the inverse matrix of $\mathbf{A}^t$. Thus, we can derive

$$(\mathbf{A}^{-1})^t = (\mathbf{A}^t)^{-1}. \tag{1.45}$$

This concludes the proof of Eq. (1.39).

R Program [1 - 13]

An inverse matrix is constructed as follows.

```
vec61()
  function ()
  {
  # (1)
    aa <- matrix(c(-2, -8 ,1, 5), ncol = 2)
    print("aa")
    print(aa)
  # (2)
    bb <- solve(aa)
    print("bb")
    print(bb)
  # (3)
    cc <- aa %*% bb
    print("cc")
    print(cc)
  # (4)
    dd <- bb %*% aa
    print("dd")
    print(dd)
  }
```

(1) Matrix `aa` is created and output as:

```
[1] "aa"
     [,1] [,2]
[1,]   -2    1
[2,]   -8    5
```

(2) `solve()` gives the inverse matrix of `aa`, and the result, stored in `bb`, is output:

```
[1] "bb"
     [,1] [,2]
[1,] -2.5  0.5
[2,] -4.0  1.0
```

(3) `aa %*% bb` is calculated, and the result, stored in `cc`, is output:

```
[1] "cc"
     [,1] [,2]
[1,]    1    0
[2,]    0    1
```

(4) `bb %*% aa` is derived, and the result, stored in `dd`, is output:

```
[1] "dd"
     [,1] [,2]
[1,]    1    0
[2,]    0    1
```

R Program [1 - 13] End

R Program [1 - 14]

Equation (1.33) (page 29) can be verified as shown below.

```
vec66()
  function ()
  {
  # (1)
    aa <- c(2, -9, 1, 3)
    aam <- matrix(aa, ncol = 2)
    bb <- c(3, 4, -5, -9)
    bbm <- matrix(bb, ncol = 2)
  # (2)
    ccm <- solve(aam %*% bbm)
    print("ccm")
    print(ccm)
    ddm <- solve(bbm) %*% solve(aam)
    print("ddm")
    print(ddm)
  }
```

(1) Matrices `aam` and `bbm` are given.
(2) Both sides of Eq. (1.33) are calculated and the result is displayed.

`vec66()` gives:

```
[1] "ccm"
           [,1]       [,2]
[1,] -0.1714286 -0.1809524
[2,] -0.1428571 -0.0952381
[1] "ddm"
           [,1]       [,2]
[1,] -0.1714286 -0.1809524
[2,] -0.1428571 -0.0952381
```

R Program [1 - 14] End

1.7 Simultaneous Equations

Let us consider the following simultaneous equations:

$$\begin{cases} -2x_1 + x_2 = 4 \\ -8x_1 + 5x_2 = -11. \end{cases} \tag{1.46}$$

By transforming Eq. (1.46) into matrix form, we have

$$\begin{pmatrix} -2 & 1 \\ -8 & 5 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 4 \\ -11 \end{pmatrix}. \tag{1.47}$$

Both sides of Eq. (1.47) are multiplied from the left by $\begin{pmatrix} -2 & 1 \\ -8 & 5 \end{pmatrix}^{-1}$. Then, we have

$$\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} -2 & 1 \\ -8 & 5 \end{pmatrix}^{-1} \begin{pmatrix} 4 \\ -11 \end{pmatrix}. \tag{1.48}$$

Using Eq. (1.28), we can derive

$$\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} -2.5 & 0.5 \\ -4 & 1 \end{pmatrix} \begin{pmatrix} 4 \\ -11 \end{pmatrix} = \begin{pmatrix} -15.5 \\ -27 \end{pmatrix}. \tag{1.49}$$

Thus, we have the solution of Eq. (1.46).

R Program [1 - 15]

Next, we explain how to solve simultaneous equations.

```
vec71()
  function ()
  {
  # (1)
    aa <- matrix(c(-2, -8 ,1, 5), ncol = 2)
    print("aa")
    print(aa)
  # (2)
    ff <- matrix(c(4, -11), ncol = 1)
    print("ff")
    print(ff)
  # (3)
    aain <- solve(aa)
    print("aain")
    print(aain)
  # (4)
    xx <- aain %*% ff
    print("xx")
    print(xx)
  # (5)
    dd <- aa %*% xx
    print("dd")
    print(dd)
  # (6)
    xx2 <- solve(aa, ff)
    print("xx2")
    print(xx2)
  }
```

(1) Matrix `aa` (the matrix on the left hand side of Eq. (1.47) (page 32)) is given and output as follows:

```
[1] "aa"
     [,1] [,2]
[1,]   -2    1
[2,]   -8    5
```

(2) Column vector `ff` (the matrix located on the right hand side of Eq. (1.47)) is created and output as follows:
```
[1] "ff"
     [,1]
[1,]    4
[2,]  -11
```
(3) `solve()` derives the inverse matrix of `aa`. The result is stored in `aain` and is output:
```
[1] "aain"
     [,1] [,2]
[1,] -2.5  0.5
[2,] -4.0  1.0
```
(4) Using Eq. (1.48) (page 32), the simultaneous equations are solved. The result is stored in `xx` and output:
```
[1] "xx"
      [,1]
[1,] -15.5
[2,] -27.0
```
(5) `xx` is confirmed to be the correct solution.
```
[1] "dd"
     [,1]
[1,]    4
[2,]  -11
```
(6) `solve(aa, ff)` also yields the solution of the simultaneous equations. The result is stored in `xx2` and output. The R command, `solve(aa, ff)`, gives the solution of simultaneous equations in an easier way than following steps (3) and (4).
```
[1] "xx2"
      [,1]
[1,] -15.5
[2,] -27.0
```

R Program [1 - 15] End

1.8 Diagonalization of a Symmetric Matrix

A symmetric matrix is a specific type of square matrix, such as

$$\begin{pmatrix} 5 & -2 \\ -2 & 0 \end{pmatrix}, \quad \begin{pmatrix} 1 & -3 & 9 \\ -3 & 2 & 4.5 \\ 9 & 4.5 & -4 \end{pmatrix}. \tag{1.50}$$

In other words, if the elements of the square matrix **H** are referred to as h_{ij} $(1 \le i \le n, 1 \le j \le n)$, a matrix that satisfies $h_{ij} = h_{ji}$ is called a symmetric matrix.

Assume that the 2×2 symmetric matrix **H** satisfies

$$\mathbf{U}^{-1}\mathbf{H}\mathbf{U} = \boldsymbol{\Lambda}, \tag{1.51}$$

where **U** is a 2×2 matrix and $\boldsymbol{\Lambda}$ is a 2×2 diagonal matrix. **U** and $\boldsymbol{\Lambda}$ are written as

$$\mathbf{U} = \begin{pmatrix} u_{11} & u_{12} \\ u_{21} & u_{22} \end{pmatrix}, \qquad \boldsymbol{\Lambda} = \begin{pmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{pmatrix}. \tag{1.52}$$

Derivation of $\boldsymbol{\Lambda}$ satisfying Eq. (1.51) is termed diagonalization by **U**.

From Eqs. (1.51) and (1.52) we obtain

$$\mathbf{H}\mathbf{U} = \begin{pmatrix} h_{11}u_{11} + h_{12}u_{21} & h_{11}u_{12} + h_{12}u_{22} \\ h_{21}u_{11} + h_{22}u_{21} & h_{21}u_{12} + h_{22}u_{22} \end{pmatrix}, \tag{1.53}$$

$$\mathbf{U}\boldsymbol{\Lambda} = \begin{pmatrix} u_{11} & u_{12} \\ u_{21} & u_{22} \end{pmatrix} \begin{pmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{pmatrix} = \begin{pmatrix} \lambda_1 u_{11} & \lambda_2 u_{12} \\ \lambda_1 u_{21} & \lambda_2 u_{22} \end{pmatrix}. \tag{1.54}$$

Hence, we have

$$\begin{pmatrix} h_{11}u_{11} + h_{12}u_{21} & h_{11}u_{12} + h_{12}u_{22} \\ h_{21}u_{11} + h_{22}u_{21} & h_{21}u_{12} + h_{22}u_{22} \end{pmatrix} = \begin{pmatrix} \lambda_1 u_{11} & \lambda_2 u_{12} \\ \lambda_1 u_{21} & \lambda_2 u_{22} \end{pmatrix}. \tag{1.55}$$

This can be transformed into

$$\begin{pmatrix} h_{11} - \lambda_1 & h_{12} \\ h_{21} & h_{22} - \lambda_1 \end{pmatrix} \begin{pmatrix} u_{11} \\ u_{21} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \qquad \begin{pmatrix} h_{11} - \lambda_2 & h_{12} \\ h_{21} & h_{22} - \lambda_2 \end{pmatrix} \begin{pmatrix} u_{12} \\ u_{22} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}. \tag{1.56}$$

If either u_{11} or u_{21} is not zero and either u_{12} or u_{22} is not zero, the following equations are satisfied:

$$\frac{h_{11} - \lambda_1}{h_{12}} = \frac{h_{21}}{h_{22} - \lambda_1}, \qquad \frac{h_{11} - \lambda_2}{h_{12}} = \frac{h_{21}}{h_{22} - \lambda_2}. \tag{1.57}$$

These equations can be transformed into

$$(h_{11} - \lambda_1)(h_{22} - \lambda_1) - h_{12}h_{21} = 0, \qquad (h_{11} - \lambda_2)(h_{22} - \lambda_2) - h_{12}h_{21} = 0. \tag{1.58}$$

Equation (1.58) is transformed into a more general form in the context of matrices:

$$\begin{vmatrix} h_{11} - \lambda_1 & h_{12} \\ h_{21} & h_{22} - \lambda_1 \end{vmatrix} = 0, \qquad \begin{vmatrix} h_{11} - \lambda_2 & h_{12} \\ h_{21} & h_{22} - \lambda_2 \end{vmatrix} = 0. \tag{1.59}$$

These are sometimes written as

$$|\mathbf{H} - \lambda_1 \mathbf{I}| = 0, \qquad |\mathbf{H} - \lambda_2 \mathbf{I}| = 0, \tag{1.60}$$

where $\mathbf{I}$ is an identity matrix and $|\mathbf{H} - \lambda_1 \mathbf{I}|$ is called the determinant of $(\mathbf{H} - \lambda_1 \mathbf{I})$. Since Eq. (1.60) consists of two equations with the same form, λ_1 and λ_2 are the solutions of the equation $|\mathbf{H} - \lambda \mathbf{I}| = 0$. Thus, λ_1 and λ_2 are the solutions of the quadratic equation:

$$(h_{11} - \lambda)(h_{22} - \lambda) - h_{12}h_{21} = 0. \tag{1.61}$$

Since these are the solutions of the quadratic equation, $\lambda_1 = \lambda_2$ may hold. Equation (1.61) is transformed into

$$\lambda^2 + (h_{11} + h_{22})\lambda + h_{11}h_{22} - h_{12}h_{21} = 0. \tag{1.62}$$

The discriminant of this quadratic equation is

$$\begin{aligned} D &= (h_{11} + h_{22})^2 - 4(h_{11}h_{22} - h_{12}h_{21}) \\ &= (h_{11} - h_{22})^2 + 4h_{12}h_{21}. \end{aligned} \tag{1.63}$$

Since $\mathbf{H}$ is a symmetric matrix, $h_{12} = h_{21}$ is satisfied, and hence, $h_{12}h_{21} \geq 0$ holds. Therefore, we obtain $D \geq 0$. Thus, both λ_1 and λ_2 are real numbers.

Not only can the determinant of a 2×2 square matrix be defined, but also the determinants of larger square matrices such as a 3×3 square matrix. Equations (1.59) and (1.60) are called the eigenequations (characteristic equations) of $\mathbf{H}$. λ_1 and λ_2 given by an eigenequation are called eigenvalues (or characteristic values) of $\mathbf{H}$. Once λ_1 and λ_2 have been obtained, substitution into Eq. (1.56) leads to $\begin{pmatrix} u_{11} \\ u_{21} \end{pmatrix}$ and $\begin{pmatrix} u_{12} \\ u_{22} \end{pmatrix}$. These vectors are termed eigenvectors (or characteristic vectors). If we have eigenvectors, diagonalization of a symmetric matrix is realized using Eq. (1.51) (page 35).

The eigenvectors given by an eigenequation for a symmetric matrix formulate $\mathbf{U}$ (Eq. (1.51) (page 35)), which satisfies

$$\mathbf{U}^{-1} = \mathbf{U}^t. \tag{1.64}$$

In other words, when an inverse matrix of $\mathbf{U}$ is required, the transposed matrix of $\mathbf{U}$ can be used. If $\mathbf{U}$ has this characteristic, it is called an orthogonal matrix. The word "orthogonal" originates from the following characteristic of $\mathbf{U}$: if the column vectors constituting $\mathbf{U}$ are called $\mathbf{u}_1$ and $\mathbf{u}_2$ ($\mathbf{u}_1 = (u_{11} \ \ u_{21})^t$, $\mathbf{u}_2 = (u_{12} \ \ u_{22})^t$), $\mathbf{u}_1$ and $\mathbf{u}_2$ are orthogonal. That is, the inner product of $\mathbf{u}_1$ and $\mathbf{u}_2$ is zero.

Diagonalization of a symmetric matrix is applicable in various calculations. For example, let both sides of Eq. (1.51) (page 35) be raised to the power m (m is a positive integer). The result is

$$(\mathbf{U}^{-1}\mathbf{H}\mathbf{U})^m = \boldsymbol{\Lambda}^m. \tag{1.65}$$

Since $\mathbf{U}^{-1}\mathbf{U} = \mathbf{I}$ holds, we have

$$(\mathbf{U}^{-1}\mathbf{H}\mathbf{U})^m = \mathbf{U}^{-1}\mathbf{H}^m\mathbf{U} = \boldsymbol{\Lambda}^m. \tag{1.66}$$

Furthermore, we can obtain

$$\mathbf{H}^m = \mathbf{U}\boldsymbol{\Lambda}^m\mathbf{U}^{-1}. \tag{1.67}$$

Since $\boldsymbol{\Lambda}$ is a diagonal matrix, $\boldsymbol{\Lambda}^m$ is written as

$$\boldsymbol{\Lambda}^m = \begin{pmatrix} \lambda_1^m & 0 \\ 0 & \lambda_2^m \end{pmatrix}. \tag{1.68}$$

The same is true for a larger symmetric matrix than $\boldsymbol{\Lambda}$. Hence, use of Eq. (1.67) yields $\mathbf{H}^m$ by a simple calculation.

R Program [1 - 16]

Next, we explain how a symmetric matrix is diagonalized.

```
vec81()
  function ()
  {
  # (1)
    hh <- matrix(c(-2, -8 ,-8, 3), ncol = 2)
    print("hh")
    print(hh)
  # (2)
    eigen1 <- eigen(hh)
  # (3)
    lam <- eigen1$values
    print("lam")
    print(lam)
  # (4)
    uu <-  eigen1$vectors
    print("uu")
    print(uu)
  # (5)
    det1 <- det(hh-lam[1] * diag(2))
    print("det1")
    print(det1)
  # (6)
    det2 <- det(hh-lam[2] * diag(2))
    print("det2")
    print(det2)
  }
```

(1) Matrix hh is given and displayed.

```
[1] "hh"
     [,1] [,2]
[1,]   -2   -8
[2,]   -8    3
```

(2) eigen() solves the eigenequation of hh. The resultant eigenvalues and eigenvectors are obtained and stored in eigen1.

(3) eigen1 is an object in the form of a list. An object in this form usually contains several list components. One of the list components of eigen1 is values. eigen1$values indicates that a component called values is extracted from the list components stored in eigen1. The list component values consists of eigenvalues. Then, eigen1$values is saved as lam and is displayed as follows:

```
[1] "lam"
[1]  8.881527 -7.881527
```

(4) Eigenvectors stored in eigen1 are extracted. The result, which is stored in uu, is displayed as follows:

```
[1] "uu"
           [,1]       [,2]
[1,] -0.5923365 -0.8056907
[2,]  0.8056907 -0.5923365
```

(5) The eigenequation (the equation on the left in Eq. (1.60) (page 36)) is confirmed. det() yields a determinant (Eq. (1.60)), lam[1] is the first eigenvalue, and diag(2) is a 2×2 identity matrix.

```
[1] "det1"
[1] 9.664738e-15
```

This value is very close to zero.

(6) The eigenequation (the equation on the left in Eq. (1.60) (page 36)) is confirmed. lam[2] is the second eigenvalue.

```
[1] "det2"
[1] -2.043504e-15
```

This value is very close to zero.

R Program [1 - 16] End

R Program [1 - 17]

Here, we calculate the 5-th power of a symmetric matrix.

```
vec91()
  function ()
  {
  # (1)
    aa <- matrix(c(-2, -8 ,-8, 3), ncol = 2)
    print("aa")
    print(aa)
  # (2)
    aa5 <- aa %*% aa %*% aa %*% aa %*% aa
    print("aa5")
    print(aa5)
  # (3)
    eigen1 <- eigen(aa)
  # (4)
    lam <- eigen1$values
    uu <-  eigen1$vectors
  # (5)
    diag5 <- diag(c(lam[1]^5, lam[2]^5))
    print("diag5")
    print(diag5)
  # (6)
    aa5eigen <- uu %*% diag5 %*% t(uu)
    print("aa5eigen")
    print(aa5eigen)
  }
```

(1) Matrix aa is given and calculated.

```
[1] "aa"
     [,1] [,2]
[1,]   -2   -8
[2,]   -8    3
```

(2) The 5-th power of aa is stored in aa5 and displayed as follows:

```
[1] "aa5"
       [,1]   [,2]
[1,]   -352 -40888
[2,] -40888  25203
```

(3) eigen() solves the eigenequation of aa. The eigenvalues and eigenvectors are stored in eigen1.

(4) The eigenvalues stored in eigen1 are extracted into lam. The eigenvectors stored in eigen1 are extracted into uu.

(5) The diagonal matrix diag5 is constructed. The first and second diagonal elements are, respectively, lam[1]^5 and lam[2]^5 (^5 means "to the power of 5"). Matrix diag5 is displayed as follows:

```
[1] "diag5"
          [,1]      [,2]
[1,] 55263.49      0.00
[2,]     0.00 -30412.49
```

(6) Using Eq. (1.67) (page 37), we calculate `aa` to the power of 5. This is stored in `aa5eigen` and displayed as follows:

```
[1] "aa5eigen"
        [,1]   [,2]
[1,]    -352 -40888
[2,] -40888  25203
```

R Program [1 - 17] End

1.9 Quadratic Forms

The following equation is a specific form of a function with two variables (x_1 and x_2). This form is called the quadratic form. Similar definitions are used for functions with more than two variables.

$$y = \mathbf{x}^t \mathbf{A} \mathbf{x}, \tag{1.69}$$

where **x** and **A** are given as

$$\mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}, \quad \mathbf{A} = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix}. \tag{1.70}$$

A is a diagonal matrix. That is, $a_{21} = a_{12}$ holds.

Equation (1.69) is written as

$$\begin{aligned} y &= a_{11}x_1^2 + (a_{12} + a_{21})x_1x_2 + a_{22}x_2^2 \\ &= a_{11}x_1^2 + 2a_{12}x_1x_2 + a_{22}x_2^2, \end{aligned} \tag{1.71}$$

where if **A** is a diagonal matrix, it can be transformed into the form of Eq. (1.51) (page 35) by solving the eigenequation of **A**. Thus, using Eq. (1.64) (page 36), we have

$$\mathbf{A} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^{-1}. \tag{1.72}$$

By substituting this equation into Eq. (1.69), Eq. (1.64) (page 36) yields

$$\begin{aligned} y &= \mathbf{x}^t \mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^{-1}\mathbf{x} \\ &= (\mathbf{U}^t\mathbf{x})^t \boldsymbol{\Lambda} (\mathbf{U}^t\mathbf{x}). \end{aligned} \tag{1.73}$$

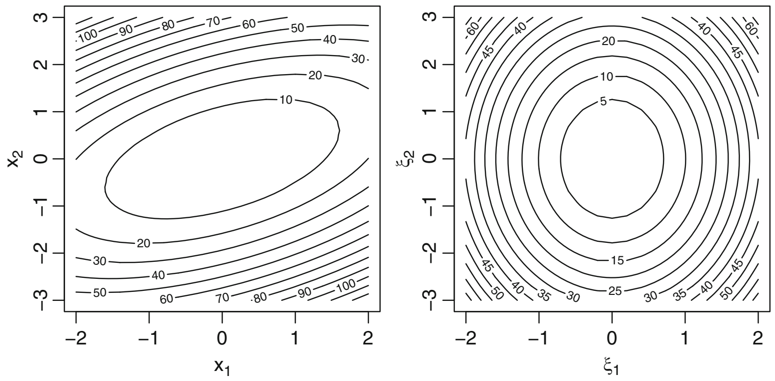


Fig. 1.17 Graphs of a function in quadratic form, where both λ_1 and λ_2 (Eq. (1.76) (page 41)) are positive: *left* contour with axes x_1 and x_2, and *right* contour with axes ξ_1 and ξ_2

When the second equation above is derived from the first equation, $\mathbf{x}^t\mathbf{U}$ is transformed into $(\mathbf{U}^t\mathbf{x})^t$ (Eq. (1.20) (page 22)).

The following substitution is carried out in Eq. (1.73).

$$\boldsymbol{\xi} = \mathbf{U}^t\mathbf{x}. \tag{1.74}$$

Then, we have

$$y = \boldsymbol{\xi}^t \boldsymbol{\Lambda} \boldsymbol{\xi}. \tag{1.75}$$

Using $\boldsymbol{\xi} = (\xi_1, \xi_2)^t$, the following equation is obtained:

$$y = \lambda_1\xi_1^2 + \lambda_2\xi_2^2. \tag{1.76}$$

Since this equation does not include the term $\xi_1\xi_2$, it can be dealt with easily. The form of Eq. (1.75) is called the diagonal form.

If both λ_1 and λ_2 are positive in Eq. (1.76) (page 41), y is positive unless both ξ_1 and ξ_2 are zero. In this instance, Eq. (1.69) (page 40) is called a positive definite function and $\mathbf{A}$ is called a positive definite matrix. If $\mathbf{A}$ is a positive definite matrix, the value of Eq. (1.76) takes the minimum value when $\xi_1 = \xi_2 = 0$. If both λ_1 and λ_2 are positive or zero, $\mathbf{A}$ is called a positive semidefinite matrix. Furthermore, if both λ_1 and λ_2 in Eq. (1.75) are negative, y is negative unless both ξ_1 and ξ_2 are zero. In this instance, Eq. (1.69) is called a negative definite function and $\mathbf{A}$ is called a negative definite matrix.

Figure 1.17 depicts the graphs for a function in quadratic form. Both λ_1 and λ_2 (Eq. (1.76) (page 41)) are positive. Figure 1.17 (left) shows the contour with axes x_1 and x_2, while Fig. 1.17 (right) shows the contour with axes ξ_1 and ξ_2. The latter contour shows how the contour lines form vertical ovals because Eq. (1.76)

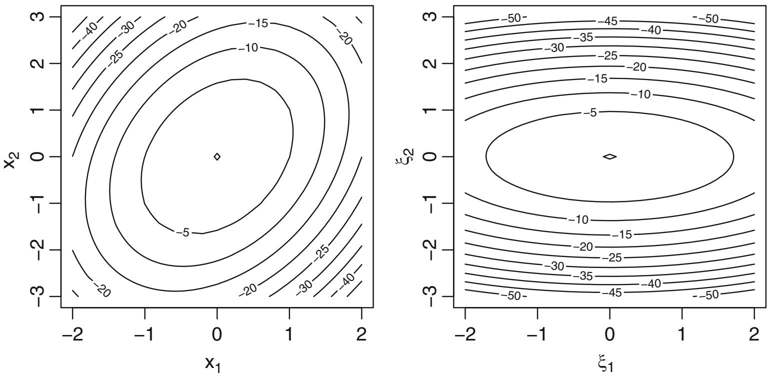


Fig. 1.18 Graphs of a function in quadratic form, where both λ_1 and λ_2 (Eq. (1.76) (page 41)) are negative: *left* contour with axes x_1 and x_2, and *right* contour with axes ξ_1 and ξ_2

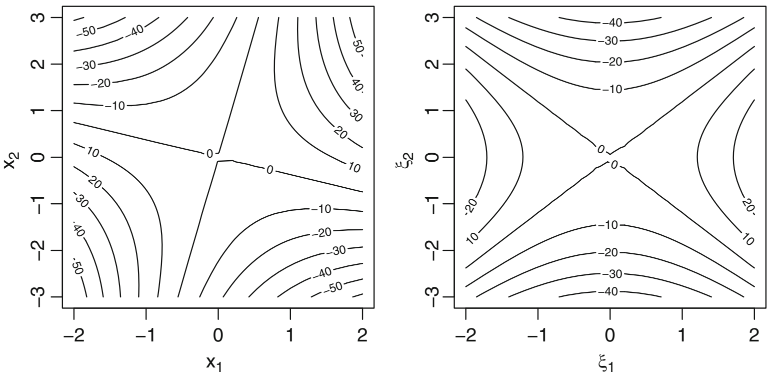


Fig. 1.19 Graphs of a function in quadratic form with λ_1 (Eq. (1.76) (page 41)) positive and λ_2 (Eq. (1.76)) negative: *left* contour with axes x_1 and x_2, and *right* contour with axes ξ_1 and ξ_2

includes terms ξ_1^2 and ξ_2^2 only. Moreover, the center point of the elliptically-shaped lines (the original point) depicts the minimum value of Eq. (1.76) (page 41). On the other hand, Fig. 1.18 illustrates the graphs for a function in quadratic form where both λ_1 and λ_2 (Eq. (1.76) (page 41)) are negative. Although Fig. 1.18 (right) also shows that the contour lines form vertical ovals, the center point of the elliptically-shaped lines (the original point) depicts the maximal value of Eq. (1.76) (page 41). In contrast, Fig. 1.19 depicts the graphs when λ_1(Eq. (1.76) (page 41)) is positive and λ_2 is negative. The behavior of these contour lines is considerably different from those of the previous two examples. The original point does not reflect either the minimum or the maximum point of the function given by Eq. (1.76). This point is called a saddle point. If a vertical straight line crossing the center point is drawn in Fig. 1.19 (left), the value on the straight line varies as $(-40, -30, -20, -10, 0, -10, -20, -30, -40)$ from the top to the bottom. Hence, zero is the maximum value. However, if a horizontal straight line crossing the center point is drawn in the same figure, the value on the straight line varies as (20, 10, 0, 10, 20). Here, zero is the minimum value. That is, the center point is considered both the maximum and minimum point depending on the direction.

A necessary and sufficient condition for **A** to be a positive semidefinite matrix is that **A** is written as (page 144 in [2]):

$$\mathbf{A} = \mathbf{F}^t\mathbf{F}. \tag{1.77}$$

This can be proved as follows.

If there exists **F** satisfying Eq. (1.77), the equation below holds for an arbitrary **x**.

$$\mathbf{x}^t\mathbf{A}\mathbf{x} = \mathbf{x}^t\mathbf{F}^t\mathbf{F}\mathbf{x} = |\mathbf{F}\mathbf{x}|^2, \tag{1.78}$$

where $|\cdot|$ indicates the length of a vector (Eq. (1.1) (page 8)). This value is always positive or zero. Hence, **A** is a positive semidefinite matrix.

On the contrary, if **A** is a positive semidefinite matrix, diagonalization is realized as given below (Eq. (1.51) (page 35)).

$$\mathbf{A} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^{-1}, \tag{1.79}$$

where all the diagonal elements of $\boldsymbol{\Lambda}$ are positive or zero. Hence, we obtain

$$\boldsymbol{\Lambda} = \begin{pmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{pmatrix} = \begin{pmatrix} \sqrt{\lambda_1} & 0 \\ 0 & \sqrt{\lambda_2} \end{pmatrix} \begin{pmatrix} \sqrt{\lambda_1} & 0 \\ 0 & \sqrt{\lambda_2} \end{pmatrix}. \tag{1.80}$$

Therefore, we set

$$\tilde{\boldsymbol{\Lambda}} = \begin{pmatrix} \sqrt{\lambda_1} & 0 \\ 0 & \sqrt{\lambda_2} \end{pmatrix}. \tag{1.81}$$

Thus, Eq. (1.78) (page 43) is transformed into

$$\mathbf{A} = \mathbf{U}\tilde{\boldsymbol{\Lambda}}\tilde{\boldsymbol{\Lambda}}\mathbf{U}^{-1} = (\tilde{\boldsymbol{\Lambda}}\mathbf{U}^{-1})^t(\tilde{\boldsymbol{\Lambda}}\mathbf{U}^{-1}). \tag{1.82}$$

In other words, we have Eq. (1.77) (page 43).

From the above, a necessary and sufficient condition for **A** to be a positive semidefinite matrix is that **A** is written as Eq. (1.77) (page 43). This proof is easily generalized to any **A** with an arbitrary size.

Even a complex function can be approximated by a quadratic form (Eq. (1.69) (page 40)) at points depicting maximal values (to be precise, local maximal values), minimal values (or more precisely, local minimal values), or saddle points. Hence, in order to understand the behavior of a function near such points, matrices representing quadratic forms are classified into categories, such as positive definite or negative definite matrices. Thus, the points can be discerned from the three different types described above. Therefore, the classification of a matrix used in a quadratic form plays an important role in ascertaining the behavior of the function.

R Program [1 - 18]

The behavior of a positive definite function and its diagonal form is illustrated below.

```
vec101()
  function ()
  {
  # (1)
    par(mfrow = c(1,2), mai = c(2, 1, 1.5, 0.1),
     omi = c(0, 0, 0, 0))
  # (2)
    aa <- matrix(c(5, -3 , -3, 8), ncol = 2)
    print("aa")
    print(aa)
  # (3)
    n1 <- 21
    xx1 <- seq(from = -2, to = 2, length = n1)
    n2 <- 61
    xx2 <- seq(from = -3, to = 3, length = n2)
  # (4)
    yym <- matrix(rep(0, length = n1 * n2), ncol = n2)
    for(ii in 1:n1){
      for(jj in 1:n2){
      xx12 <- c(xx1[ii], xx2[jj])
      yym[ii, jj] <- t(xx12) %*% aa %*% xx12
      }
    }
  # (5)
    contour(xx1, xx2, yym, xlab = expression(x[1]),
     ylab = expression(x[2]), cex.axis = 0.95,
     cex.lab = 0.9)
  # (6)
    eigen1 <- eigen(aa)
  # (7)
    lam <- eigen1$values
    print("lam")
    print(lam)
    uu <-  eigen1$vectors
    print("uu")
    print(uu)
  # (8)
    lamm <- diag(lam)
  # (9)
    yyml <- matrix(rep(0, length = n1 * n2), ncol = n2)
    for(ii in 1:n1){
      for(jj in 1:n2){
      xx12 <- c(xx1[ii], xx2[jj])
      yyml[ii, jj] <- t(xx12) %*% lamm %*% xx12
      }
    }
  # (10)
    contour(xx1, xx2, yyml, xlab = expression(paste(xi,
     scriptstyle(1))), ylab = expression(paste(xi,
     scriptstyle(2))), cex.axis = 0.95, cex.lab =0.9)
  }
```

(1) `par()` sets the graphics area. `mfrow = c(1,2)` specifies that two graphs are aligned horizontally. `mai = c(2, 1, 1.5, 0.1)` assigns the sizes of the margins around each graph starting at the lower margin, and then setting the left, upper, and right margins, in order. `omi` = assigns the size of the outer margin surrounding the two graphs.

(2) Matrix `aa` is given and displayed. Since `aa` is a symmetric matrix, it is an example of $\mathbf{A}$ (Eq. (1.69) (page 40)).

(3) `seq()` gives a series of equally-spaced numbers. The result is stored in `xx1`. `seq(from = -2, to = 2, length = n1)` indicates an equally-spaced sequence from -2 to 2, with the length of the sequence specified by `n1`. Thus, `xx1` is set to $-2.0, -1.8, -1.6, \ldots, 2.0$. Another `seq()` yields another equally-spaced sequence. The result is stored in `xx2`. `seq(from = -3, to = 3, length =2)` indicates an equally-spaced sequence from -3 to 3, with the length of the sequence specified by `n2`. Thus, `xx2` is set to $-3.0, -2.9, -2.8, \ldots, 3.0$

(4) The values of a function in quadratic form (y in Eq. (1.69), Eq. (1.71), and Eq. (1.75)) are calculated at the grid points formed by `xx1` and `xx2`. The resultant values are stored in `yym`. `rep(0, length = n1 * n2)` yields a series of zeroes with length `n1 * n2`.`matrix(, ncol = n2)` constructs a matrix with `n2` columns.

(5) `contour()` depicts the contour of the values in `yym`. `cex.axis = 0.95` specifies the size of the letters along the axes. `cex.lab = 0.9` specifies the size of the letters in the label.

(6) `eigen()` solves the eigenequation of `aa`. The resultant eigenvalues and eigenvectors are stored in `eigen1`.

(7) The eigenvalues saved in `eigen1` are extracted and stored in `lam`, which is displayed. `lam` is $\boldsymbol{\Lambda}$ (Eq. (1.72) (page 40)). The eigenvectors saved in `eigen1` are extracted and stored in `uu`, which is displayed. `uu` is $\mathbf{U}$ (Eq. (1.72)).

(8) A diagonal matrix in which the diagonal elements are `lam` is created and stored in `lamm`.
(9) The values of the function in quadratic form given by `lamm` are calculated at grid points formed by `xx1` and `xx2`. The resultant values are stored in `yyml`. The values in `yyml` represent the values of the grid points formed by the rotated axes.
(10) `contour()` illustrates the contour lines of the values in `yyml`.

`vec101()` results in:

```
[1] "aa"
     [,1] [,2]
[1,]    5   -3
[2,]   -3    8
[1] "lam"
[1] 9.854102 3.145898
[1] "uu"
           [,1]       [,2]
[1,] -0.5257311 -0.8506508
[2,]  0.8506508 -0.5257311
```

Executing `vec101()` also outputs Fig. 1.17 (page 41).

The code demarcated by (2) in `vec101()` is replaced by the following code and the resultant R program is called `vec111()`.

```
# (2)
  aa <- matrix(c(-5, 1 , 1, -2), ncol=2)
  print("aa")
  print(aa)
```

The result of executing `vec111()` is:

```
[1] "lam"
[1] -1.697224 -5.302776
[1] "uu"
           [,1]       [,2]
[1,] -0.2897841 -0.9570920
[2,] -0.9570920  0.2897841
```

Executing `vec111()` also outputs Fig. 1.18 (page 42).

The code demarcated by (2) in `vec101()` is replaced by the following code and the resultant R program is called `vec121()`.

```
# (2)
  aa <- matrix(c(4, 5 , 5, -2), ncol=2)
  print("aa")
  print(aa)
```

The result of executing `vec121()` is:

```
[1] "lam"
[1]  6.830952 -4.830952
[1] "uu"
```

```
             [,1]       [,2]
[1,] -0.8701999  0.4926988
[2,] -0.4926988 -0.8701999
```

Executing `vec121()` outputs Fig. 1.19 (page 42).

R Program [1 - 18] End

References

1. Burnham KP, Anderson DR (2002) Model selection and multi-model inference: a practical information-theoretic approach, 2nd edn. Springer, New York
2. Takezawa K (2006) Introduction to nonparametric regression. Wiley, New York

Chapter 2
Distributions and Tests

2.1 Sampling and Random Variables

A “population” denotes a set in which elements satisfy specific conditions, such as all people in a country, or all small cars in a country. In statistical terms, extracting part of the population is called “sampling from the population.” Instead of sampling, we sometimes say we are conducting a trial. Hence, a sample is a subset of the population. Measurements, surveys, or observations over the sample produce data. Data consist of numbers, categories, images, and text. Data can also be called realizations, observations (observed values), or measurements. The term “data set” (or dataset) is used when the plurality of the data should be emphasized. In practical data analysis, however, the distinction between the sample and the data (or dataset) is not usually important.

The number of extracted samples is represented as the “sample size”. When 100 people have their weight measured, the sample size is 100. On the other hand, the “number of samples” indicates the number of batches of samples. Hence, if the weight measurement of 100 people is conducted once, the number of samples is 1. Hence, a statistical test using the result of one survey with one condition is called a one-sample test. However, if the result of the survey is classified into two groups and a comparison between the two groups is conducted, it becomes a two-sample test. Hence, the number of samples depends on the standpoint of the sampling.

Values representing the characteristics of a population are termed parameters. For example, when a population is all of the people in a country, the percentage of people in a specific age range is a parameter. In a narrow sense, however, parameters indicate the values used in an equation for specifying the probability density function, which describes the appearance of a specific population. If the population is all of the people in a country, a sample (in this example, some people) is extracted by sampling. Data are produced by checking the ages of the people in the sample. In particular, sampling in social surveys is called a sample survey (sampling survey). Values obtained by performing calculations with the data are called statistics. Statistics depict the appearance of a sample. Hence, statistics are synonymous with estimated parameters. Since the results of sampling are affected by chance, statistics are also affected by chance. Therefore, while parameters are specific values associated with a population, most statistics change with every sampling. For example, 100 people are sampled from all of the people in a country, and the average (a kind of statistic) of their ages is calculated. If this procedure is repeated several times, the resulting values are slightly different from one another. Hence, collected statistics constitute a distribution. A particular statistic that gives a result for estimating a parameter is termed an estimator. When the parameters are defined to be all of the values representing the characteristics of a population, a statistic is synonymous with an estimator.

On the other hand, an estimate is the value of an estimator in a context that emphasizes that the value is given by the result of one sampling. That is, an estimate is a value yielded by a specific sample. Hence, it is a fixed value that does not accompany a distribution. Since an estimator is a set of estimates obtained by several samplings, it is represented as a distribution. In a practical situation, however, we do not usually need to distinguish between one sampling and several samplings. Therefore, the terms of an estimate and an estimator are used interchangeably in most situations.

In a usual sampling, we cannot extract all elements (or individuals) from a population. For example, it is difficult to conduct a survey of all of the people in a country. To overcome this situation, part of the population is collected and extracted at random. If random sampling is conducted, the results change with each sampling. Estimates derived from samples given by one or more samplings depict the appearance of a population. Data analysis (or data mining) consists of: (a) calculating estimates using samples; (b) estimating the appearance of a population using the estimates; (c) making decisions, predictions, controls, and graphics using the results of (b).

When we assume that the result of the first sampling is called x_1, that of the second sampling is called x_2, that of the third sampling is called x_3, and so forth, these results are collectively called X. That is, when we write "This sampling results in X," we mean that x_1 may come out, x_2 may come out, x_3 may come out, and so forth. This is not necessarily discursive knowledge. For example, let us assume that $x_1, x_2, x_3, \ldots$ are estimates. Let us also suppose that we obtain the value 2 as the estimate from 70 out of 100 samplings, and the value 1 in the other 30 samplings. We know that values other than 1 and 2 do not appear. In such a situation, although X does not represent a specific value, it describes the probability of the value that X provides. Such an X is called a random variable (stochastic variable). If sampling is conducted several times, the estimates given by respective samplings are distributed. Since the distributions are considered to depict probabilities, the estimators yielded by the distributions of the estimates are random variables.

A random variable is usually represented as a capital letter (upper case letter), such as X or Y. On the other hand, the values resulting from each sampling, which are realizations, are nonrandom variables (nonstochastic variables). They are usually represented as a lower case letter (small letter). Hence, if we need to emphasize that a variable is a random variable, it is usually represented as a capital letter. If, on the other hand, we wish to emphasize that a variable is not a random variable but a number (nonrandom variable) yielded by data, it is usually represented as a lower case letter.

On the premise that sampling is carried out several times, the values resulting from a sample are random variables. On the contrary, the values given by a specific sample are nonrandom variables. Hence, this distinction is sometimes not important. However, the difference between values (estimates) given by a specific sampling and estimators reflecting the distributions in a population plays a significant role in equations of generalized linear regression or mixed models.

2.2 Probability Distribution

When a random variable (X) gives a value (i.e., a realization) that is one of $\{x_1, x_2, x_3, \ldots, x_m\}$, X is said to have a discrete type probability distribution, or simply a discrete distribution. The probability that a random variable (X) yields x_k is written as $P(X = x_k)$. $P(x_k)$ is another expression; it is used when the random variable does not need to be named X as far as the probabilities of taking the values of respective $\{x_k\}$ are known. When $\{x_k\}$ includes all the possible values given by X, we have

$$\sum_{i=1}^{m} P(X = x_k) = 1. \tag{2.1}$$

Assume that $P(a \leq X \leq b)$ stands for the possibility that the value yielded by a random variable (X) lies between a and b ($a < b$). When $P(a \leq X \leq b)$ is represented as the following equation, X has a continuous type probability distribution, or simply a continuous distribution.

$$P(a \leq X \leq b) = \int_a^b den(x)dx, \tag{2.2}$$

where $den(x)$ is a function that takes a positive value or 0. It is called a probability density function. Because the value provided by X should take a value between $-\infty$ and ∞, we have

$$P(-\infty \leq X \leq \infty) = \int_{-\infty}^{\infty} den(x)dx = 1. \tag{2.3}$$

The function $F(a)$ defined below is termed a cumulative probability distribution function or a distribution function.

$$F(a) = \int_{-\infty}^{a} den(x)dx. \tag{2.4}$$

Using a distribution function, Eq. (2.2) is written as

$$F(b) - F(a) = \int_a^b den(x)dx. \tag{2.5}$$

When X has a continuous type probability distribution, the expectation (expected value) of X is represented as $E[X]$. Its definition is:

$$E[X] = \int_{-\infty}^{\infty} xden(x)dx. \tag{2.6}$$

The expectation indicates the average of data obtained from a population with a specific probability density function. In most settings, the average of the realizations ($\{x_1, x_2, x_3, \ldots x_m\}$) of X takes a value close to $E[X]$ when m is large.

2.3 Normal Distribution and the Central Limit Theorem

A diverse range of probability density functions have continuous distributions ($den(x)$ in Eq. (2.2), page 51). A representative example is the normal distribution. The probability density function of the normal distribution ($den(x)$) is written as

$$den(x) = \left(\frac{1}{\sqrt{2\pi\sigma^2}}\right) \exp\left(-\frac{1}{2\sigma^2}(x-\mu)^2\right), \tag{2.7}$$

where μ is the mean and σ^2 is the variance. The positive square root of the variance is termed the standard deviation (σ). When μ is -2 and σ^2 is 9, the appearance of $den(x)$ is illustrated in Fig. 2.1. Since Eq. (2.2) (page 51) holds for a normal distribution, the equation below is satisfied (Fig. 2.1).

$$P(a \leq X \leq b) = \int_a^b \left(\frac{1}{\sqrt{2\pi\sigma^2}}\right) \exp\left(-\frac{1}{2\sigma^2}(x-\mu)^2\right) dx. \tag{2.8}$$

Since Eq. (2.3) also holds, we have

$$\int_{-\infty}^{\infty} \left(\frac{1}{\sqrt{2\pi\sigma^2}}\right) \exp\left(-\frac{1}{2\sigma^2}(x-\mu)^2\right) dx = 1. \tag{2.9}$$

Regardless of the values of μ and σ^2, a normal distribution gives the relationships below (Fig. 2.2).

$$\int_{\mu}^{\mu+\sigma} \left(\frac{1}{\sqrt{2\pi\sigma^2}}\right) \exp\left(-\frac{1}{2\sigma^2}(x-\mu)^2\right) dx$$

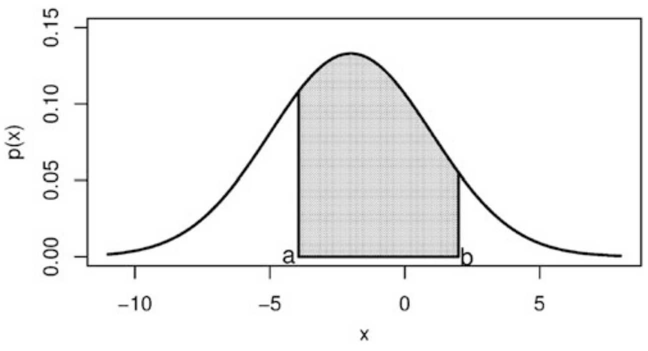


Fig. 2.1 Probability density function of a normal distribution ($den(x)$). Equation (2.8) gives the area colored gray

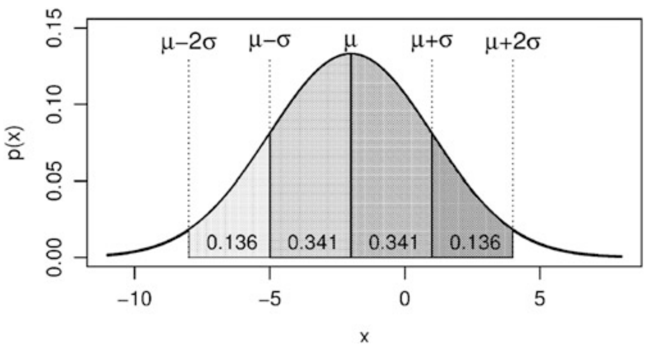


Fig. 2.2 Values of the integral of the probability density function of a normal distribution. The integral ranges are defined by μ and σ

$$= \int_{\mu-\sigma}^{\mu} \left(\frac{1}{\sqrt{2\pi\sigma^2}}\right) \exp\left(-\frac{1}{2\sigma^2}(x-\mu)^2\right) dx \approx 0.3413447, \tag{2.10}$$

$$\begin{aligned} &\int_{\mu+\sigma}^{\mu+2\sigma} \left(\frac{1}{\sqrt{2\pi\sigma^2}}\right) \exp\left(-\frac{1}{2\sigma^2}(x-\mu)^2\right) dx \\ &= \int_{\mu-2\sigma}^{\mu-\sigma} \left(\frac{1}{\sqrt{2\pi\sigma^2}}\right) \exp\left(-\frac{1}{2\sigma^2}(x-\mu)^2\right) dx \approx 0.1359051. \end{aligned} \tag{2.11}$$

The central limit theorem states that the average of realizations yielded by a random variable that obeys a probability density function with a finite variance is distributed like a normal distribution. Figure 2.3 (top) shows a histogram of $50,000$ data. It indicates that these data obey a probability density function far from a normal distribution. These data are then divided into $1,000$ groups, each with 50 elements. The average of the elements in each respective group is calculated, and $1,000$ data

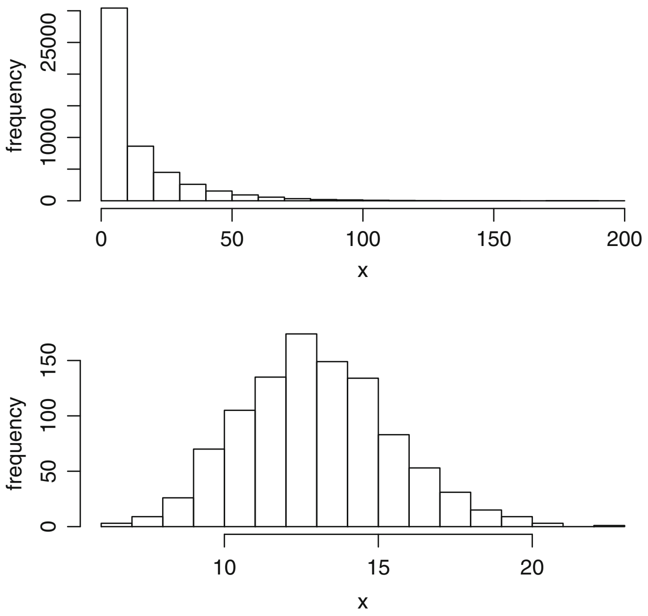


Fig. 2.3 Histogram of 50, 000 data that obey a probability density function far from a normal distribution (*top*). Histogram of the respective averages of sets of 50 data (*bottom*)

are obtained. The histogram of the resultant data is illustrated in Fig. 2.3 (bottom). The appearance is pretty close to that of a normal distribution. That is, even if data obey a probability density function far from a normal distribution, a new set of data yielded by taking the average of respective subsets of the data obey a probability density function close to a normal distribution. Using this characteristic, even when the obtained data obey a probability density function far from a normal distribution, a new data set given by averaging respective groups of data can be treated by the theories of the normal distribution. However, the number of data is reduced by averaging, and some information in the data is thus lost. Hence, this strategy should be avoided as far as possible—it is preferable to retain the original data. On the other hand, when a statistical theory is developed, use of the central limit theorem sometimes leads to simple results because of the availability of theories for the normal distribution (e.g., page 47 in [2]).

R Program [2 - 1]

The value of $\int_a^b den(x)dx$ in a normal distribution indicates the proportion of all realizations of the normal distribution between a and b.

```
norm10()
   function (){
   # (1)
```

```
  set.seed(162)
  nd <- 10000
  xx <- rnorm(n = nd, mean = -2, sd = 3)
# (2)
  aa <- -4
  bb <- -2
  xxab <- xx[xx >= aa & xx <= bb]
# (3)
  pp1 <- length(xxab) / nd
  print("pp1")
  print(pp1)
# (4)
  pp2 <-  pnorm(q=-2, mean = -2, sd = 3) -
   pnorm(q = -4, mean = -2, sd = 3)
  print("pp2")
  print(pp2)
}
```

(1) `set.seed()` sets an initial value for pseudo-random numbers. The number of data (`nd`) is given. `rnorm()` produces the `nd` realizations of the normal distribution with mean (−2) and standard deviation 3, and they are stored in `xx`. This procedure is equivalent to sampling from a population that obeys a normal distribution with mean (−2) and standard deviation 3.
(2) `xx[xx>=aa & xx<=bb]` extracts values between −4 and −2 from `xx`. The results are stored in `xxab`. Using `xx[xx>=aa & xx<=bb]`, values that satisfy the conditions described in `[]` are selected from the elements of the vector `xx`. "&" in `xx>=aa & xx<=bb` means "and". On the other hand, "or" is represented by "|."
(3) The number of elements of `xxab` (a part of `xx`) is divided by the number of data (`nd`). The result is stored in `pp1`. Then, `pp1` is the proportion of `xxab` to `xx`. `length()` yields the number of elements of a vector.
(4) The area from −4 through −2 in a normal distribution is calculated. `pnorm(q = -2, mean = -2, sd = 3)` provides $\int_{-\infty}^{-2}\left(\frac{1}{\sqrt{2\pi\sigma^2}}\right)^n \exp\left(-\frac{1}{2\sigma^2}(x-\mu)^2\right)dx$ when $\mu = -2$ and $\sigma^2 = 3^2$.
`pnorm(q = -4, mean = -2, sd = 3)` derives $\int_{-\infty}^{-4}\left(\frac{1}{\sqrt{2\pi\sigma^2}}\right)^n \exp\left(-\frac{1}{2\sigma^2}(x-\mu)^2\right)dx$ when $\mu = -2$ and $\sigma^2 = 3^2$. Hence, `pp2` is the value of $\int_{-4}^{-2}\left(\frac{1}{\sqrt{2\pi\sigma^2}}\right)^n \exp\left(-\frac{1}{2\sigma^2}(x-\mu)^2\right)dx$ (Eq. (2.5), page 52).

The results of `norm10()` are as follows.

```
[1] "pp1"
[1] 0.246
```

```
  [1] "pp2"
  [1] 0.2475075
```

Whereas pp1 is the value given by the simulation, pp2 is the theoretical value calculated by the probability density function. These two values are close.

R Program [2 - 1] End

R Program [2 - 2]

Let us confirm that the averages of the subsets of realizations of a distribution are distributed almost normally, even if the original distribution is far from a normal distribution.

```
norm20e()
  function (){
  # (1)
    set.seed(171)
  # (2)
    nd <- 50
    nt <- 1000
  # (3)
    xx2all <- rep(0, length = nd * nt)
    xx2mall <- NULL
    for(ii in 1:nt){
     xx <- rnorm(n = nd, mean = -2, sd = 3)
     xx2 <- xx^2
     xx2all[(ii * nd - nd + 1):(ii * nd)] <- xx2
     xx2mall[ii] <- mean(xx2)
    }
  # (4)
    par(mfrow = c(2,1), mai = c(1, 1, 0.5, 0.1),
     omi = c(0, 0, 0, 0))
  # (5)
    br1 <-  pretty(xx2all, n = 20)
    bw1 <- br1[2] - br1[1]
    xx2allh <- floor(xx2all / bw1) * bw1 + 0.01 * bw1
    hist(xx2allh, breaks = br1, main="", xlab = "x",
     ylab = "frequency")

    br2 <-  pretty(xx2mall, n = 20)
    bw2 <- br2[2] - br2[1]
    xx2mallh <- floor(xx2mall / bw2) * bw2 + 0.01 * bw2
    hist(xx2mallh, breaks = br2, main="", xlab = "x",
     ylab = "frequency")
  }
```

(1) `set.seed()` sets an initial value of pseudo-random numbers.
(2) `nd` is the number of data belonging to a subset of the data. `nt` is the number of subsets. Hence, `nd` multiplied by `nt` is the total number of original data.
(3) `rnorm()` produces the `nd` realizations of the normal distribution with mean (-2) and standard deviation 3. They are stored in `xx`. `xx` is obtained `nt` times. All of the given data are stored in `xx2all`. The average of each `nd` data is taken. These are stored in `xx2mall`.
(4) `par()` sets the graphics area.
(5) `pretty(xx2all, n = 20)` results in equally-spaced values that cover the range of all of the elements in `xx2all`. The interval of these equally-spaced values is written as $r \cdot 10^m$ (r is one of $\{1, 2, 5\}$, m is an integer), and hence the interval is something like 0.02, 10, or 5, 000. The number of these equally-spaced values is approximately 1 plus the value set by `n =`. The values of the end points of these intervals are round numbers. `hist()` draws histograms of `xx2all` and `xx2mall`. Before performing `hist()`, some preprocessing is carried out using `floor()`. This is a necessary step for illustrating a histogram in the usual sense (page 67 in [3]). These processes result in Fig. 2.3 (page 54).

R Program [2-2] End

2.4 Interval Estimation by *t* Distribution

It is assumed that n data ($\{x_i\}$ $(1 \leq i \leq n)$) are obtained by sampling from a population. Each x_i is a realization of a normal distribution. The average of these data is written as $\bar{x}$, and is calculated as

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i. \tag{2.12}$$

If each x_i is regarded as a random variable (X_i), their average is also a random variable ($\bar{X}$). Then, we have

$$\bar{X} = \frac{1}{n}\sum_{i=1}^{n} X_i. \tag{2.13}$$

The average of the population (μ) is called the population mean. It is an intrinsic and fixed value belonging to the population. On the other hand, the value of $\bar{x}$ differs between samplings. Thus, it is difficult to calculate an exact value for μ. A more realistic option is to estimate the approximate range of μ. This is known as an interval estimation of the population. In contrast, an estimation of the value of $\bar{x}$ is termed a point estimation of the population mean. Equation (2.12) is not the only point estimation method for the population mean. For example, the median is another point estimation of the population mean.

The data given by the 1st sampling are denoted by $\{x_1^{(1)}, x_2^{(1)}, x_3^{(1)}, \ldots, x_n^{(1)}\}$. Those given by the 2nd sampling are then $\{x_1^{(2)}, x_2^{(2)}, x_3^{(2)}, \ldots, x_n^{(2)}\}$, ..., and those given by the l-th sampling are $\{x_1^{(l)}, x_2^{(l)}, x_3^{(l)}, \ldots, x_n^{(l)}\}$. The averages of these data sets are specified as $\bar{x}^{(j)}$, $(1 \leq j \leq l)$. Thus, since $\bar{x}^{(j)}$ is considered a realization of a random variable, this random variable is called $\bar{X}$. $\bar{X}$ obeys a normal distribution because $\bar{X}$ is the (constant multiplied) sum of realizations of a normal distribution. It can be analytically proven that the sum of realizations of a normal distribution is regarded as a random variable obeying a normal distribution. This is confirmed by simulating a great many realizations of a normal distribution, and illustrating the distribution of the averages of their subsets. The resulting distribution approaches a normal distribution.

When the data yielded by the 1st sampling has been processed, the average given by the data is denoted by $\bar{x}^{(1)}$. The estimate of variance (σ^2) is called $(s^{(1)})^2$. It is defined as

$$(s^{(1)})^2 = \frac{1}{n-1} \sum_{i=1}^{n} (x_i^{(1)} - \bar{x}^{(1)})^2. \tag{2.14}$$

This value is called the unbiased estimator of variance, or the unbiased variance. To find the meaning of this value, a random variable that results in realizations $(\{(s^{(1)})^2, (s^{(2)})^2, \ldots, \})$ is named S^2. That is, we set

$$S^2 = \frac{1}{n-1} \sum_{i=1}^{n} (X_i - \bar{X})^2, \tag{2.15}$$

where we assume

$$X_i = E[X] + \epsilon_i. \tag{2.16}$$

X_i is a random variable producing x_i. $E[X]$ is the expectation (Eq. (2.6), page 52) of X. Since we assume that sampling is conducted with the same population and the same conditions, we have

$$E[X_i] = E[X]. \tag{2.17}$$

$\bar{X}$ is obtained by Eq. (2.13). X is a random variable that yields the realization $\{x_1, x_2, \ldots, \}$. Furthermore, $\{\epsilon_i\}$ are random variables satisfying the conditions below, and they are independent of one another. These random variables do not need to obey a normal distribution.

$$E[\epsilon_i] = 0, \tag{2.18}$$

$$E[\epsilon_i \epsilon_j] = \begin{cases} \sigma^2 & \text{if } i = j \\ 0 & \text{if } i \neq j. \end{cases} \tag{2.19}$$

σ^2 is the variance of X. Then, the following equation holds.

$$\begin{aligned}
E[S^2] &= \frac{1}{n-1}E\Big[\sum_{i=1}^{n}(X_i-\bar{X})^2\Big] \\
&= \frac{1}{n-1}E\Big[\sum_{i=1}^{n}\Big(X_i-\sum_{j=1}^{n}\frac{X_j}{n}\Big)^2\Big] \\
&= \frac{1}{n-1}E\Big[\sum_{i=1}^{n}\Big(E[X]+\epsilon_i-\sum_{j=1}^{n}\frac{E[X]+\epsilon_j}{n}\Big)^2\Big] \\
&= \frac{1}{n-1}E\Big[\sum_{i=1}^{n}\Big(E[X]+\epsilon_i-\frac{\epsilon_i}{n}-\sum_{j=1(j\neq i)}^{n}\frac{\epsilon_j}{n}-n\cdot\frac{E[X]}{n}\Big)^2\Big] \\
&= \frac{1}{n-1}E\Big[\sum_{i=1}^{n}\Big(\frac{(n-1)\epsilon_i}{n}-\sum_{j=1(j\neq i)}^{n}\frac{\epsilon_j}{n}\Big)^2\Big] \\
&= \frac{1}{n-1}E\Big[\sum_{i=1}^{n}\Big(\frac{(n-1)^2\epsilon_i^2}{n^2}-2\frac{(n-1)\epsilon_i}{n} \\
&\qquad \sum_{j=1(j\neq i)}^{n}\frac{\epsilon_j}{n}+\frac{1}{n^2}\Big(\sum_{j=1(j\neq i)}^{n}\epsilon_j\Big)\Big(\sum_{k=1(k\neq i)}^{n}\epsilon_k\Big)\Big)\Big] \\
&= \frac{1}{n-1}\sum_{i=1}^{n}\Big(\frac{(n-1)^2\sigma^2}{n^2}+\frac{(n-1)\sigma^2}{n^2}\Big) \\
&= \frac{1}{n-1}\sum_{i=1}^{n}\frac{(n-1)\sigma^2}{n} \\
&= \sigma^2,
\end{aligned} \tag{2.20}$$

where $\sum_{j=1(j\neq i)}^{n}$ indicates that the sum is taken from $j = 1$ to n, but that element i is excluded. The equality between the second and third lines is derived from Eq. (2.16), and that between the sixth and seventh lines is obtained from Eq. (2.19).

Equation (2.20) shows that when Eq. (2.14) (page 58) is used as the estimate of $E[S^2]$, its expectation is σ^2 (the variance of X). The name of the unbiased estimator of variance (Eq. (2.14), page 58) serves to contrast it with the maximum likelihood estimator of variance (i.e., the maximum likelihood variance) as below.

$$(s^{(1)'})^2 = \frac{1}{n}\sum_{i=1}^{n}(x_i^{(1)}-\bar{x}^{(1)})^2. \tag{2.21}$$

When the number of data is large, the difference between the unbiased estimator of variance and the maximum likelihood estimator of variance is negligible. Moreover, if not otherwise specified, the positive square root of the unbiased estimator of variance is called the standard deviation.

Next, let us examine the characteristics of the average given by the samples obtained by sampling from a population ($\bar{X}$ defined in Eq. (2.13), page 57). First, the expectation of $\bar{X}$ is

$$E[\bar{X}] = \sum_{i=1}^{n} E\Big[\frac{E[X] + \epsilon_i}{n}\Big] = \frac{1}{n}\sum_{i=1}^{n} E[X] = E[X], \tag{2.22}$$

using Eq. (2.18) (page 58). Writing the variance of $\bar{X}$ as $\text{var}(\bar{X})$, we have

$$\begin{aligned}
\text{var}(\bar{X}) &= E[(\bar{X} - E[\bar{X}])^2] \\
&= E\Big[\Big(\frac{1}{n}\sum_{i=1}^{n}(E[X] + \epsilon_i) - E[X]\Big)^2\Big] \\
&= E\Big[\Big(\frac{1}{n}\sum_{i=1}^{n}\epsilon_i\Big)^2\Big] \\
&= \frac{1}{n^2}E\Big[(\epsilon_1 + \epsilon_2 + \cdots + \epsilon_n)(\epsilon_1 + \epsilon_2 + \cdots + \epsilon_n)\Big] \\
&= \frac{1}{n^2}n\sigma^2 \\
&= \frac{\sigma^2}{n},
\end{aligned} \tag{2.23}$$

where the equality between the first and second lines is derived from Eqs. (2.22) and (2.16) (page 58). The equality between the fourth and fifth lines is obtained using Eq. (2.19) (page 58).

Therefore, the mean of the normal distribution producing $\bar{X}$ is $E[X]$. Its variance is $\frac{\sigma^2}{n}$. Hence, if the value of σ^2 is known, we can define a variable Z as below such that it obeys a standard normal distribution with mean 0 and variance 1.

$$Z = \frac{\bar{X} - \mu}{\frac{\sigma}{\sqrt{n}}}, \tag{2.24}$$

where μ is identical to $E[X]$. That is, it is the mean of the population.

Then, the probability density function of Z is illustrated by the solid line in Fig. 2.4. The solid arrow indicates the central range covering 95 % of this probability density function. This range indicated by the arrow is written as

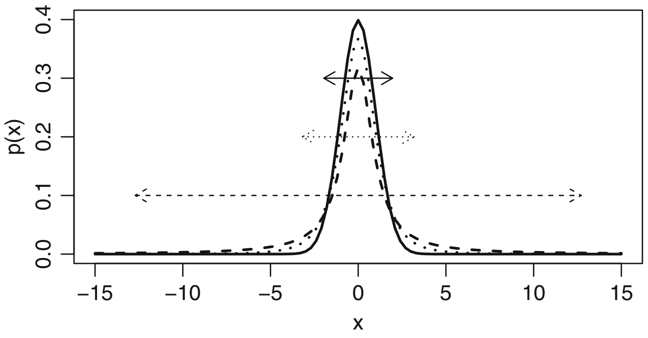


Fig. 2.4 Probability density function of the t-distribution with 1 degree of freedom (ϕ) is shown as a *dashed line*. Probability density function of the t-distribution with 3 degrees of freedom (ϕ) is shown as a *dotted line*. Probability density function of a normal distribution with mean 0 and variance 1 is shown as a *solid line*; this probability density function is identical to the t-distribution with $\phi = \infty$ degrees of freedom. The *dashed*, *dotted*, and *solid arrows* indicate the central ranges, which covers 95 % of the probability density function of the respective distributions

$$-1.96 < \frac{\bar{X} - \mu}{\frac{\sigma}{\sqrt{n}}} < 1.96. \tag{2.25}$$

In this equation, $\bar{X}$ is a random variable. Replacing $\bar{X}$ with a realization yields

$$\begin{aligned}
-1.96 &< \frac{\bar{x}^{(1)} - \mu}{\frac{\sigma}{\sqrt{n}}} < 1.96,\\
-1.96 &< \frac{\bar{x}^{(2)} - \mu}{\frac{\sigma}{\sqrt{n}}} < 1.96,\\
\vdots \quad & \qquad \vdots \qquad\quad \vdots\\
-1.96 &< \frac{\bar{x}^{(l)} - \mu}{\frac{\sigma}{\sqrt{n}}} < 1.96.
\end{aligned} \tag{2.26}$$

Equation (2.25) indicates that the 95 % range of these equations holds. That is, when $\{x_1^{(1)}, x_2^{(1)}, x_3^{(1)}, \ldots, x_n^{(1)}\}$ is obtained and used to calculate $\bar{x}^{(1)}$, Eq. (2.26) is satisfied with 95 % probability.

If $l = 1$, Eq. (2.26) becomes

$$\bar{x}^{(1)} - 1.96\frac{\sigma}{\sqrt{n}} < \mu < \bar{x}^{(1)} + 1.96\frac{\sigma}{\sqrt{n}}. \tag{2.27}$$

This equation is satisfied with 95 % probability. However, in practice, sampling is usually conducted only once. Hence, the expression "with 95 % probability" is sometimes paraphrased as "the confidence coefficient is 95 % (or 0.95)." The range of Eq. (2.27) is called the confidence interval. Although the confidence interval is defined as the central range covering 95 % of the probability density function, other percentages such as 99 % are sometimes employed. $\bar{x}^{(1)} - 1.96\frac{s^{(1)}}{\sqrt{n}}$ is called the lower confidence limit, and $\bar{x}^{(1)} + 1.96\frac{s^{(1)}}{\sqrt{n}}$ is called the upper confidence limit.

Furthermore, Eq. (2.27) becomes

$$\mu - 1.96\frac{\sigma}{\sqrt{n}} < \bar{x}^{(1)} < \mu + 1.96\frac{\sigma}{\sqrt{n}}. \tag{2.28}$$

Therefore, if further sampling is carried out under the same conditions, the resultant $\bar{x}^{(j)}(2 \le j \le l)$ lies in the range described below with 95 % probability.

$$\mu - 1.96\frac{\sigma}{\sqrt{n}} < \bar{x}^{(j)} < \mu + 1.96\frac{\sigma}{\sqrt{n}}. \tag{2.29}$$

Since Eq. (2.27) is equivalent to Eq. (2.28), the following descriptions are equivalent:

(1) If repeat sampling is conducted under the same conditions, the resultant $\bar{x}^{(j)}(2 \le j \le l)$ lies in the range given by Eq. (2.28) with 95 % probability.
(2) The confidence interval of μ is written as Eq. (2.27).

However, the value of σ is usually unknown. In such an event, Z (Eq. (2.24), page 60) obeys the t-distribution. This fact is expressed as

$$\frac{\bar{X} - \mu}{\dfrac{S}{\sqrt{n}}} \sim t(\phi), \tag{2.30}$$

where S is a random variable that provides $\{s^{(j)}\}$ $(1 \le j \le l)$ (Eq. (2.14), page 58). ϕ denotes the number of degrees of freedom. Although $t(\phi)$ is written in lower case, it is a random variable. The probability density function obeyed by $t(\phi)$ is called the t-distributionwith ϕ degrees of freedom. A realization of $t(\phi)$ is termed the t-value. It is defined as $\phi = n - 1$ (where n is the number of data). The shape of the t-distribution depends on the number of degrees of freedom. Figure 2.4 illustrates the probability density function of the t-distribution with $\phi = 1$ degree of freedom (dashed line) and $\phi = 3$ degrees of freedom (dotted line). The t-distribution with $\phi = \infty$ is identical to the normal distribution with mean 0 and variance 1. That is, if the number of data is infinite ($n = \infty$), $\sigma = s^{(1)} = s^{(2)} = \cdots = s^{(l)}$ holds exactly. Hence, we can use Eq. (2.25) (page 61).

The central 95 % range of the probability density function of the t-distribution with $\phi = 1$ degree of freedom is described as

$$-12.7 < \frac{\bar{X} - \mu}{\frac{S}{\sqrt{n}}} < 12.7. \tag{2.31}$$

The central 95 % range of the probability density function of the t-distribution with $\phi = 3$ degrees of freedom (Fig. 2.4) is described as

$$-3.18 < \frac{\bar{X} - \mu}{\frac{S}{\sqrt{n}}} < 3.18. \tag{2.32}$$

Equations such as Eqs. (2.31) and (2.32) are summarized as

$$t(\phi, 0.025) < \frac{\bar{X} - \mu}{\frac{S}{\sqrt{n}}} < t(\phi, 0.975), \tag{2.33}$$

where $t(\phi, 0.025)$ (a nonrandom variable) is the t that satisfies the following equation (where $den_t(x)$ denotes the probability density function of the t-distribution with ϕ degrees of freedom):

$$\int_{-\infty}^{t} den_t(x)dx = 0.025, \tag{2.34}$$

and $t(\phi, 0.975)$ (nonrandom variable) is the t that satisfies:

$$\int_{-\infty}^{t} den_t(x)dx = 0.975. \tag{2.35}$$

Figure 2.5 illustrates the probability density function of the t-distribution ($den_t(t)$) with $\phi = 3$ degrees of freedom. The t-value that satisfies Eq. (2.34) is $t(\phi, 0.025)$ and the t-value satisfying Eq. (2.35) is $t(\phi, 0.975)$. Hence, the following equation holds.

$$t(\phi, 0.025) = -t(\phi, 0.975). \tag{2.36}$$

The simulation below confirms that the probability density function drawn in Fig. 2.5 does actually result from Eq. (2.30) (page 62). The simulation data ($\{x_i\}$, $(1 \leq i \leq 4)$) are 4 realizations of $N(-2, 3^2)$ (a normal distribution with mean -2 and variance 3^2). The t-values are calculated using the following equation.

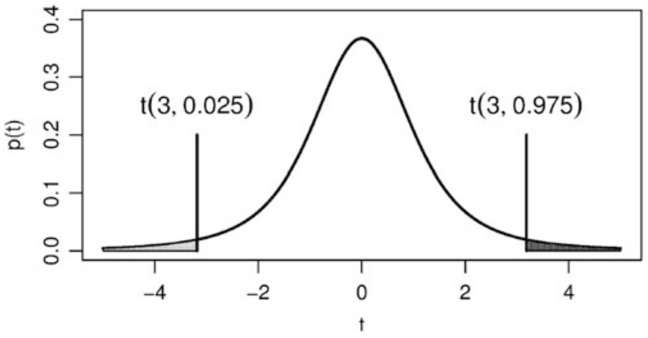


Fig. 2.5 Probability density function of the t-distribution with $\phi = 3$ degrees of freedom ($den_t(t)$). Because $\int_{-\infty}^{t(\phi,0.025)} den_t(t)dt = 0.025$ holds, the *thin* area shown in *gray* is 0.025 units. Since $\int_{t(\phi,0.975)}^{\infty} den_t(t)dt = 0.025$, the *thick gray* area is also 0.025 units

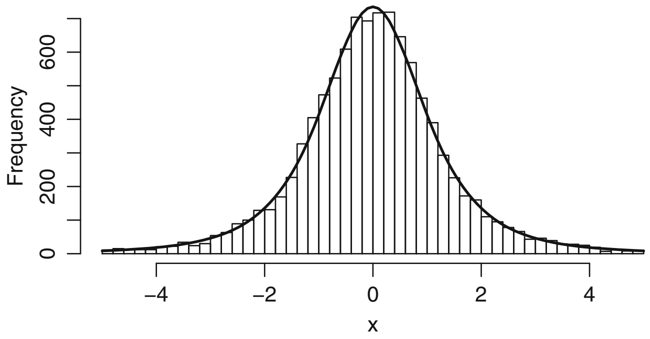


Fig. 2.6 Histogram of the t-values of 10, 000 data sets, each of which consists of 4 data, and the constant-multiplied probability density function of the t-distribution with 3 degrees of freedom (*solid line*)

$$t = \frac{\bar{x} - (-2)}{\frac{s}{\sqrt{4}}}. \tag{2.37}$$

s (Eq. (2.14), page 58) and $\bar{x}$ are defined as

$$s = \sqrt{\frac{(x_i - \bar{x})^2}{3}}, \qquad \bar{x} = \frac{\sum_{i=1}^{4} x_i}{4}. \tag{2.38}$$

By altering the initial value of the pseudo-random numbers, 10, 000 simulation data are produced, yielding the 10, 000 t-values. Figure 2.6 is a histogram illustrating their distribution. The constant-multiplied probability density function of the t-distribution with 3 degrees of freedom is also shown. The two distributions are almost the same.

When Eq. (2.33) (page 63) is represented using realizations, we have Eq. (2.26) (page 61). However, $s^{(1)} = s^{(2)} = \cdots = s^{(l)}$ does not hold here. By transforming Eq. (2.33) (page 63) into a form similar to that of Eq. (2.27) (page 61), we have

$$\bar{x}^{(1)} - t(\phi, 0.975)\frac{s^{(1)}}{\sqrt{n}} < \mu < \bar{x}^{(1)} - t(\phi, 0.025)\frac{s^{(1)}}{\sqrt{n}}. \tag{2.39}$$

This equation holds with 95 % probability. In common with the above-mentioned terminology, the expression "with 95 % probability" is sometimes paraphrased as "the confidence coefficient is 95 % (or 0.95)." The terms of the lower confidence limit and the upper confidence limit are defined in the same manner as in Eq. (2.27) (page 61).

Equation (2.36) also gives

$$\bar{x}^{(1)} - t(\phi, 0.975)\frac{s^{(1)}}{\sqrt{n}} < \mu < \bar{x}^{(1)} + t(\phi, 0.975)\frac{s^{(1)}}{\sqrt{n}}. \tag{2.40}$$

Because $t(\phi, 0.975)$ is positive, Eq. (2.40) has more intuitive appeal than Eq. (2.39) in representing a confidence interval. It should be noted that the value of $t(\phi, 0.975)$ is not listed on the t-distribution table. Instead, the value of $t(\phi, 0.025)$ that satisfies the equation below is listed.

$$\int_t^{\infty} den_t(x) = 0.025. \tag{2.41}$$

This value is the same as $t(\phi, 0.975)$ (Eq. (2.35), page 63).

Furthermore, Eq. (2.33) (page 63) is transformed into a similar form to Eq. (2.28) (page 62). Then, we have

$$\mu - t(\phi, 0.975)\frac{s^{(1)}}{\sqrt{n}} < \bar{x}^{(1)} < \mu - (\phi, 0.025)\frac{s^{(1)}}{\sqrt{n}}. \tag{2.42}$$

Using Eq. (2.36) (page 63), we derive

$$\mu - t(\phi, 0.975)\frac{s^{(1)}}{\sqrt{n}} < \bar{x}^{(1)} < \mu + t(\phi, 0.975)\frac{s^{(1)}}{\sqrt{n}}. \tag{2.43}$$

Hence, if sampling is carried out under the same conditions hereafter, the resultant $\bar{x}^{(j)}$ $(2 \le j \le l)$ has a 95 % probability of lying in the interval:

$$\mu - t(\phi, 0.975)\frac{s^{(j)}}{\sqrt{n}} < \bar{x}^{(j)} < \mu + t(\phi, 0.975)\frac{s^{(j)}}{\sqrt{n}}. \tag{2.44}$$

$\bar{x}^{(1)}$ in Eqs. (2.42) and (2.43) denotes a realization of $\bar{X}$. Therefore, $\bar{X}$ falls in this interval with 95 % probability. It may seem strange that the t-distribution is used to

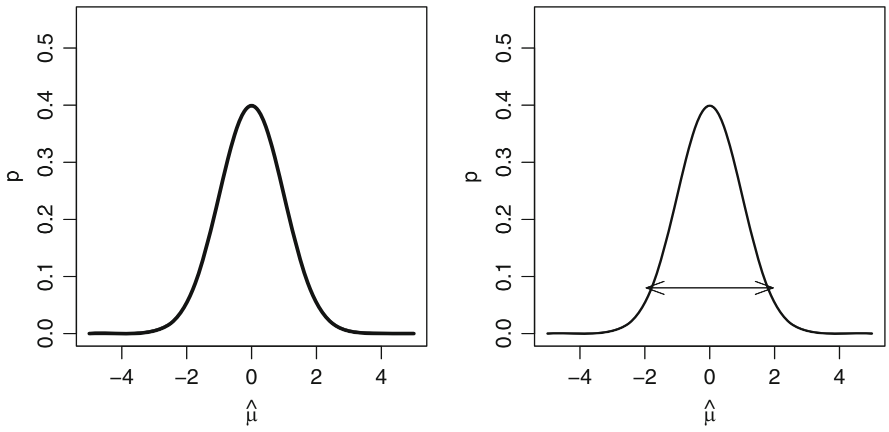


Fig. 2.7 Distribution of $\hat{\mu}$ when $\bar{x}^{(1)}$ (the average of the data) is 0. It is assumed that σ^2 in the population is 1 (*left*). The *arrow* indicates the 95 % confidence interval (*right*)

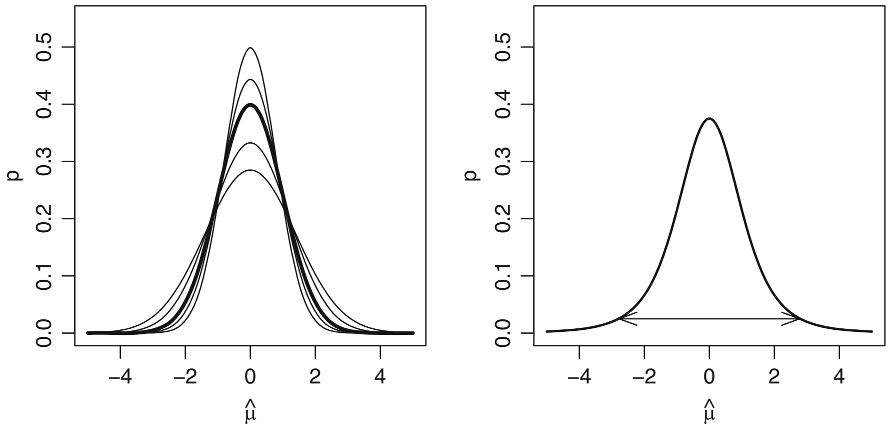


Fig. 2.8 Distribution of $\hat{\mu}$ when $\bar{x}^{(1)}$ (the average of the data) is 0. It is assumed that σ^2 in the population is unknown. The *heavy solid line* shows the probability density function with $\sigma^2 = (s^{(1)})^2$. As the *thin solid lines* indicate, the value of σ^2 is not limited to $(s^{(1)})^2$ but takes various values around it (*left*). The *arrow* shows the 95 % confidence interval. The *heavy solid line* indicates the probability density function obtained by averaging various probability density functions illustrated in this figure (*left*) (*right*)

indicate the central 95 % range, although $\bar{X}$ in Eqs. (2.42) and (2.43) obeys a normal distribution. The reason can be understood intuitively by considering Figs. 2.7 and 2.8. Values of $\hat{\mu}$ are the estimates (realizations of $\bar{X}$) yielded by estimating the value of μ using n data several times. If the variance of the population (σ^2) is known, $\hat{\mu}$ is assumed to be distributed normally. Although μ is a unique value, its estimate is represented by a probability density function (Fig. 2.7 (left)). If the probability density function obeyed by $\hat{\mu}$ can be identified, its 95 % confidence interval is obtained (Fig. 2.7 (right)). On the other hand, if the variance of a population (σ^2) is unknown, its value must be estimated. Then, $(s^{(1)})^2$ (Eq. (2.14), page 58) can be used as the value of σ^2. However, $(s^{(2)})^2, (s^{(3)})^2, \ldots$ take various values close to $(s^{(1)})^2$. Each value of the variance yields a different normal distribution (Fig. 2.8 (left)). Therefore, by averaging these normal distributions, the probability density function that $\hat{\mu}$ obeys can be derived. The resultant probability density function is not distributed normally. The 95 % confidence interval is usually somewhat wider than that of a normal distribution, although the probability density function has a similar form to a normal distribution. That is, despite the normal distribution of $\bar{X}$, its confidence interval is not represented by a normal distribution because it is constructed by combining diverse confidence intervals.

R Program [2 - 3]

The central 95 % range of the probability density function of the t-distribution is derived and compared with that of a normal distribution.

```
ttest12()
  function (){
  # (1)
    p1 <- qt(p = 0.025, df = 3)
    print("qt(p = 0.025, df = 3)")
    print(p1)
    p2 <- qt(p = 0.975, df = 3)
    print("qt(p = 0.975, df = 3)")
    print(p2)
  # (2)
    p1 <- qt(p = 0.025, df = 10000)
    print("qt(p = 0.025, df = 10000)")
    print(p1)
    p2 <- qt(p = 0.975, df = 10000)
    print("qt(p = 0.975, df = 10000)")
    print(p2)
  # (3)
    p1 <- qnorm(p = 0.025, mean = 0, sd = 1)
    print("qnorm(p = 0.025, mean = 0, sd = 1)")
    print(p1)
    p2 <- qnorm(p = 0.975, mean = 0, sd = 1)
    print("qnorm(p = 0.975, mean = 0, sd = 1)")
    print(p2)
  }
```

(1) `qt()` calculates the value of t that satisfies Eq. (2.34) (page 63) given by the t-distribution with $\phi = 3$ degrees of freedom. The result is stored in `p1`, and is output. The value of t that satisfies Eq. (2.35) (page 63) is obtained. The result is stored in `p2`, and is output. Then, the output is:

```
  "qt(p = 0.025, df = 3)"
  -3.182446
  "qt(p = 0.975, df = 3)"
  3.182446
```

(2) `qt()` calculates the value of t that satisfies Eq. (2.34) (page 63) given by the t-distribution with $\phi = 10,000$ degrees of freedom; the result is stored in `p1`, and is output. The value of t that satisfies Eq. (2.35) (page 63) is obtained. The result is stored in `p2`, and is output. Then, the output is:

```
  "qt(p = 0.025, df = 10000)"
  -1.960201
  "qt(p = 0.975, df = 10000)"
  1.960201
```

(3) `qnorm()` calculates the value of t given by replacing $den_t(\cdot)$ in Eq. (2.34) (page 63) with a probability density function of a normal distribution with mean 0 and variance 1^2; the result is stored in `p1`, and is output. The value of t given by replacing $den_t(\cdot)$ in Eq. (2.35) (page 63) with a normal distribution with mean 0 and variance 1^2 is stored in `p2`, and is output.

```
  "qnorm(p = 0.025, mean = 0, sd = 1)"
  -1.959964
  "qnorm(p = 0.975, mean = 0, sd = 1)"
  1.959964
```

The values yielded by the t-distribution with the $10,000$ degrees of freedom are very close to those given by a normal distribution with mean 0 and variance 1^2.

R Program [2-3] End

R Program [2-4]

The central 95 % range of the t-distribution is obtained by realizations of pseudo-random numbers.

```
ttest21e()
  function (){
  # (1)
    nd <- 4
    nt <- 10000
    set.seed(176)
  # (2)
    tt <- NULL
    for(ii in 1:nt){
      xx <- rnorm(n = nd, mean = -2, sd = 3)
      xxav <- mean(xx)
      xxvar <- var(xx)
      tt[ii] <- (xxav + 2)/(sqrt(xxvar)/sqrt(nd))
    }
  # (3)
```

```
    xxs <- sort(tt)
    p1 <- xxs[ceiling(nt * 0.025)]
    p2 <- xxs[ceiling(nt * 0.975)]
    print("p1")
    print(p1)
    print("p2")
    print(p2)
  # (4)
    par(mfrow = c(1,1), mai = c(2, 1, 1.5, 0.1),
     omi = c(0, 0, 0, 0))
  # (5)
    tt2 <- tt[tt >= -5 & tt <= 5]
    br1 <-  pretty(tt2, n = 40)
    bw1 <- br1[2] - br1[1]
    tt2h <- floor(tt2/bw1) * bw1 + 0.01 * bw1
    hist1 <- hist(tt2h, breaks = br1, main = "",
     xlab = "x", ylab = "Frequency")
  # (6)
    curve(dt(x, df = nd - 1) * bw1 * nt, min(br1),
     max(br1), xlab = "x", ylab = "p(x)", lwd = 2,
     xlim = c(min(br1), max(br1)),
     ylim = c(0, max(hist1$counts)), add = T)
  }
```

(1) The number of data (`nd`) is given. The number of simulations (`nt`) is given.
(2) `set.seed()` sets an initial value for the pseudo-random numbers. `tt` is prepared for storing realizations of the *t*-distribution. `nt` samplings are carried out. The respective *t*-values are calculated and stored in `tt`. `var()` calculates the unbiased estimator of variance (Eq. (2.14), page 58). `sqrt()` returns a positive square root.
(3) `sort()` arranges `xx` in ascending order. The result is saved as `xxs`. The value located closest to the 2.5 % point in `xxs` is denoted by `p1`. The value located closest to the 97.5 % point in `xxs` is named `p2`. `p1` and `p2` are displayed. `ceiling()` gives the smallest integers not less than all the arguments in parentheses (`()`).
(4) `par()` sets the graphics area.
(5) Values between −5 and 5 are extracted from `tt`, and these are named `tt2`. The histogram of `tt2` is drawn. `floor()` returns the largest integers not greater than all the arguments.
(6) `dt()` gives the values of the probability density function of the *t*-distribution with (`nd-1`) degrees of freedom. The values are multiplied by a constant and the resultant values are plotted by `curve()`.

The results of `ttest21e()` are as follows.

```
[1] "p1"
[1] -3.111366
[1] "p2"
[1] 3.16264
```

These are rather close to the results of `ttest12()` (−3.182318, 3.182318). In addition, `ttest21e()` also outputs Fig. 2.6 (page 64).

R Program [2-4] End

R Program [2-5]

Let us confirm that Eq. (2.40) (page 65) holds with 95 % probability.

`ttest22()`

```
function (){
# (1)
  nd <- 10
  nt <- 10000
  tta <- qt(0.975, df = nd - 1)
  mu1 <- 6
# (2)
  ct1 <- 0
  for(ii in 1:nt){
# (3)
    set.seed(180 + ii * 3)
    xx <- rnorm(n = nd, mean = mu1, sd = 2)
    xxav <- mean(xx)
    xxvar <- var(xx)
    uplim <- xxav + tta * sqrt(xxvar) / sqrt(nd)
    lowlim <- xxav - tta * sqrt(xxvar) / sqrt(nd)
# (4)
    if (uplim < mu1 | lowlim > mu1){
       ct1  <- ct1 + 1
    }
  }
# (5)
  print("p =")
  print(ct1 / nt)
}
```

(1) The number of data (`nd`) is given. The number of simulations (`nt`) is given. `qt()` calculates the value of $t(\phi, 0.975)$ (Eq. (2.35), page 63) with (`nd - 1`) degrees of freedom, and the value is saved as `tta`.

(2) `ct1` is prepared for recording the number of data that do not satisfy Eq. (2.40) (page 65). Then, the simulation is executed `nt` times.

(3) Simulation data (`xx`) are realizations of the normal distribution with mean `mu1` and variance 2^2. The average of `xx` is stored in `xxav`. The unbiased estimator of the variance of `xx` is saved as `xxvar`. The upper confidence limit of Eq. (2.40) is denoted by `uplim`. The lower confidence limit of Eq. (2.40) is named `lowlim`.
(4) If Eq. (2.40) (page 65) is not satisfied, 1 is added to `ct1`. "|" means "or".
(5) The probability that Eq. (2.40) is not satisfied is displayed.

`ttest22()` outputs:

```
"p ="
0.0495
```

This is approximately 0.05.

R Program [2-5] End

2.5 t-Test

As well as estimating confidence intervals, one application of the t-distribution is the t-test. The t-test is a category of hypothesis testing; it is also simply referred to as the test. Hypothesis testing uses data to determine whether a null hypothesis holds. An example of a null hypothesis is:

$$H_0 : \mu = -4.5. \tag{2.45}$$

That is, we set a hypothesis that the mean of a population is equal to -4.5. An alternative hypothesis is the hypothesis that holds when the null hypothesis is not satisfied. A possible alternative hypothesis (H_1) to H_0 is:

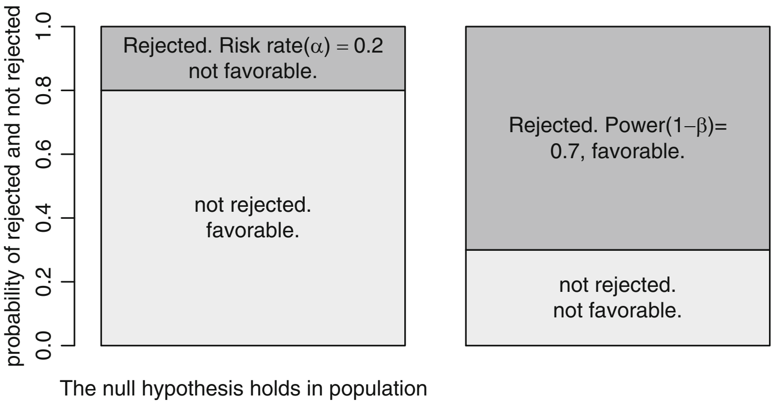
$$H_1 : \mu \neq -4.5. \tag{2.46}$$

In the process of testing a null hypothesis, we first assume that the null hypothesis holds before proceeding with the data analysis. If the data analysis concludes that the acquisition of these data should be regarded as a very rare event on the assumption of the null hypothesis, the null hypothesis is rejected, and hence the alternative hypothesis is accepted. In a reduction to absurdity, if theoretical developments on the basis of a hypothesis lead to an unacceptable deduction, the hypothesis is concluded to be wrong. On the other hand, the test does not declare that a null hypothesis is wrong. It just concludes that the data at hand is generated with only a low probability on the assumption of the null hypothesis. Then, when the phenomenon that generates the data occurs with only a low probability, we should not suppose that the event actually happened, but rather that the null hypothesis does not hold, and hence the null hypothesis is rejected. However, even if we conclude that the null hypothesis is not rejected because we cannot say that the phenomenon that generates the data at hand is a rare event, it does not indicate that we have firm grounds to prove that the null hypothesis is correct. Therefore, when the null hypothesis is not rejected, we do not state that "the null hypothesis is proven correct" or "the null hypothesis is adopted." Rather, we just say that the null hypothesis is not rejected. This situation is similar to that of reduction to absurdity: when theoretical developments on the basis of a hypothesis do not result in an unacceptable deduction, it does not show that the hypothesis is proven correct.

Moreover, even when it is concluded that the data at hand are generated with only a low probability if a null hypothesis is correct, this rare event may have actually occurred and such scarce data may have been obtained. When this happens, the null hypothesis should not have been rejected. This sort of error is called a "type I error". The probability of a type I error (α), that is to say, the probability of concluding that the data at hand have only a low probability of being generated, in spite of the correctness of the null hypothesis, is termed the risk rate. That is, the risk rate indicates the probability that the null hypothesis is rejected when it should not be. Hence, the risk rate of a test should be low. The risk rate is also known as the level of significance (significant level). Whereas the "risk rate" focuses on whether a type I error occurs, the "level of significance" focuses on the firmness of the grounds for rejecting the null hypothesis. A high risk rate means that the risk of rejecting the null hypothesis erroneously is high; this is an intuitive wording. On the other hand, the expression "the level of significance is high" gives the impression that firm grounds are required to reject the null hypothesis. However, firm grounds for rejecting the null hypothesis are needed when the level of significance is low.

On the other hand, the null hypothesis may not actually hold, but we do not conclude that the data at hand is generated with only a low probability. This type of error is called a "type II error". The probability of concluding that the null hypothesis cannot be rejected although it does not hold is denoted by β; hence, $(1-\beta)$ indicates the probability of obtaining the conclusion that the null hypothesis should be rejected when the null hypothesis does not hold. We call $(1-\beta)$ the power of the test. A lower risk rate and a larger power of test are desirable. The concepts of the risk rate and the power of a test are illustrated in Fig. 2.9. When the null hypothesis holds in a population, the null hypothesis should not be rejected. In contrast, when the null hypothesis does not hold in a population, the null hypothesis should be rejected. Therefore, the power of a test plays an important role in the development of an experimental design.

Let us consider a problem where the correctness of H_0 (Eq. (2.45), page 71) is tested when n data ($\{x_i\}$ $(1 \leq i \leq n)$) are obtained from a population that obeys a normal distribution. The alternative hypothesis is set as H_1 (Eq. (2.46), page 71). When $\mu = -4.5$ is assigned in Eq. (2.33) (page 63) and the inequality does not hold, we conclude that the $\{x_i\}$ were obtained with a low probability. Therefore, if this inequality does not hold, H_0 should be rejected. On the contrary, when the inequality holds, H_0 cannot be rejected. When H_0 is rejected and the average of the data is less than -4.5, we can say that the average of the data is "significantly" less than -4.5. If the average of the data is larger than -4.5, we can say that the average of the data is significantly larger than -4.5.



probability of rejected and not rejected
0.0
0.2
0.4
0.6
0.8
1.0
Rejected. Risk rate(α) = 0.2 not favorable.
not rejected. favorable.
Rejected. Power(1−β)= 0.7, favorable.
not rejected. not favorable.
The null hypothesis holds in population

Fig. 2.9 Concepts of the risk rate and the power of a test. Risk rate is the probability of rejecting the null hypothesis when the null hypothesis holds in a population. The power of a test is the probability of rejecting the null hypothesis when the null hypothesis does not hold in a population

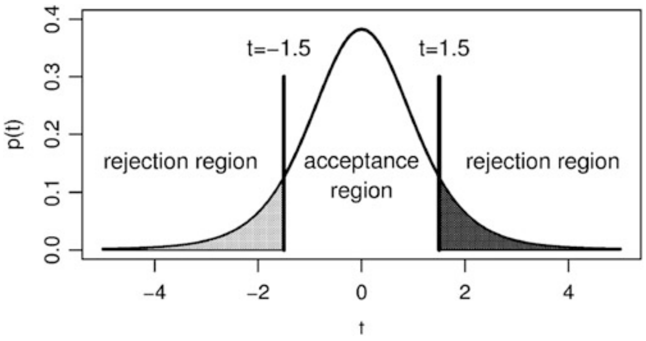


Fig. 2.10 t-distribution with 6 degrees of freedom. The region where t is larger than 1.5 is colored *dark gray*; the area is 0.09214037. The region where t is less than -1.5 is colored *light gray*; this area is 0.09214037. In the context of a two-sided test, the p-value is the area colored *dark gray* plus that colored *light gray*

In addition to deciding whether or not to reject the null hypothesis, the certainty of rejecting the null hypothesis should be quantified in order to evaluate the grounds for rejection. The value for this purpose is called the p-value. The p of p-value means the probability. For example, the curve in Fig. 2.10 depicts the t-distribution when the number of data is 7 and there are 6 degrees of freedom. When the t-value given by these 7 data is 1.5, the area over which t is larger than 1.5 is calculated in a similar way to Eq. (2.34) (page 63). That is,

$$\int_{1.5}^{\infty} den_t(x) = 0.09214037. \tag{2.47}$$

The area over which t is less than -1.5 in this t-distribution is calculated in the same way:

$$\int_{-\infty}^{-1.5} den_t(x) = 0.09214037. \tag{2.48}$$

When a two-sided test is carried out, that is, H_1 (Eq. (2.46)) is employed as the alternative hypothesis, the sum of the two values (Eqs. (2.47) and (2.48)), i.e., 0.1842807, is the probability that an absolute t-value larger than that at hand is obtained when the null hypothesis holds. This value is called the p-value. If the p-value is small, it is concluded that we have good grounds for rejecting the null hypothesis. When the result of this hypothesis testing is shown, the p-value tells us how good are the grounds for rejecting the null hypothesis as well as whether the null hypothesis is rejected.

Assume that a specific value (e.g., 0.1842807) is employed as the p-value and the central range of the t-values (the t-values between -1.5 and 1.5 in this example) is set so that the sum of the areas under both tails of the t-distribution is equal to this value. Then, if the t-value given by the data lies in either tail, we can say that the t-value is located in the rejection region (critical region). On the other hand, if the t-value falls in the central range, we say that the t-value is placed in the acceptance region (Fig. 2.10). When the t-value is located in the acceptance region, it indicates that this is a usual value if the null hypothesis holds. However, as previously discussed, this does not mean that we have a result supporting the null hypothesis and that we should adopt the null hypothesis. In that regard, the term "acceptance region" is rather questionable. On the other hand, if the t-value falls in the rejection region, it indicates that this is not a usual value if the null hypothesis holds. When this occurs, we should conclude that the null hypothesis does not hold, rather than that the unusual t-value is obtained although the null hypothesis holds. Hence, we reject the null hypothesis. The p-value sets the boundary between the rejection region and the acceptance region. We often adopt a p-value of 0.05. That is, the risk rate is usually set at 5 %. Since we are using the t-distribution, this test is called the t-test.

We sometimes set the alternative hypothesis such that μ deviates in only one direction from the value assumed in the null hypothesis. For example, the following H_1' is set as the alternative hypothesis to H_0 (Eq. (2.45)).

$$H_1' : \mu > -4.5. \tag{2.49}$$

A test in which the tails of both sides are taken into account, such as Eq. (2.46), is called a two-sided test. On the other hand, a test in which the tail of one side is taken into account is called a one-sided test. When our present knowledge of the phenomenon that generates the data shows that μ of a population cannot be less than -4.5, a one-sided test is employed. In this circumstance, the p-value is half of that of the two-sided test, because the tail on the right-hand side is the only rejection region. Even if μ cannot be less than -4.5, the average of the data can be less than

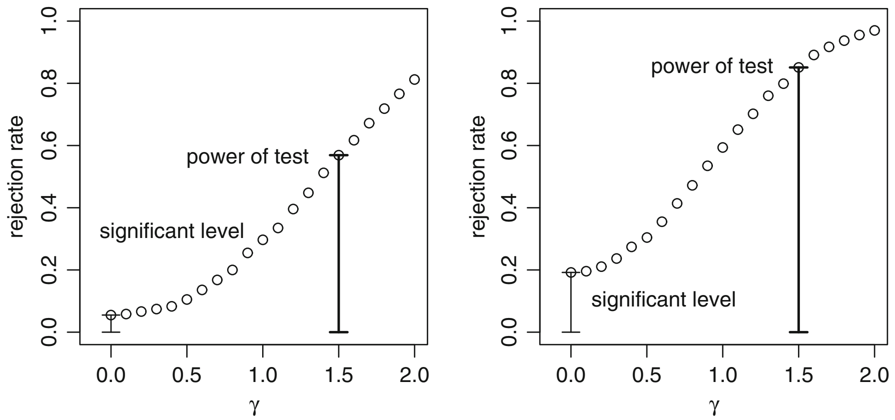


Fig. 2.11 The proportion for which the null hypothesis is rejected when a (two-sided) t-test is conducted with a 5 % risk rate (the number of data is 10) (*left*). The proportion for which the null hypothesis is rejected when a (two-sided) t-test is conducted with a 20 % risk rate (the number of data is 10) (*right*)

-4.5. In this event, it cannot be concluded that the average of the data is significantly smaller than -4.5, whatever p-value may be given. Hence, we conclude that the null hypothesis is not rejected without needing to conduct a test.

On the other hand, suppose that our concern is whether or not the average of the data is significantly larger than -4.5 when we do not know whether the mean of the population is more or less than -4.5. In this situation, if the average of the data is larger than -4.5, we carry out a two-sided test, and if the average of the data is smaller than -4.5, our conclusion is that we cannot say that the mean of the population is larger than -4.5. If we conduct a test to determine whether or not the mean of the population is equal to -4.5 when the average of the data is less than -4.5, the result is either: (1) we can say that the average of data is significantly less than -4.5, or (2) we cannot say that the average of the data is significantly less than -4.5. Whichever result is obtained, the conclusion is the same: we cannot say that the average of the data is significantly larger than -4.5. Therefore, we do not have to carry out the test when the average of the data is less than -4.5.

Let us conduct a simulation to illustrate the risk rate and the power of the test. Simulation data ($\{x_i\}$ $(1 \le i \le n)$) are generated using

$$x_i = 2.5 + \gamma + \epsilon_i, \tag{2.50}$$

where $\{\epsilon_i\}$ are realizations of $N(0, 2^2)$ (a normal distribution with mean 0 and variance 2^2). γ is one of $\{0, 0.1, 0.2, \ldots, 2\}$. 1,000 sets of simulation data (the number of data (n) is 10) are produced and a two-sided test is carried out with the following null hypothesis (H_0) and alternative hypothesis (H_1).

$$H_0 : \mu = 2.5, \tag{2.51}$$

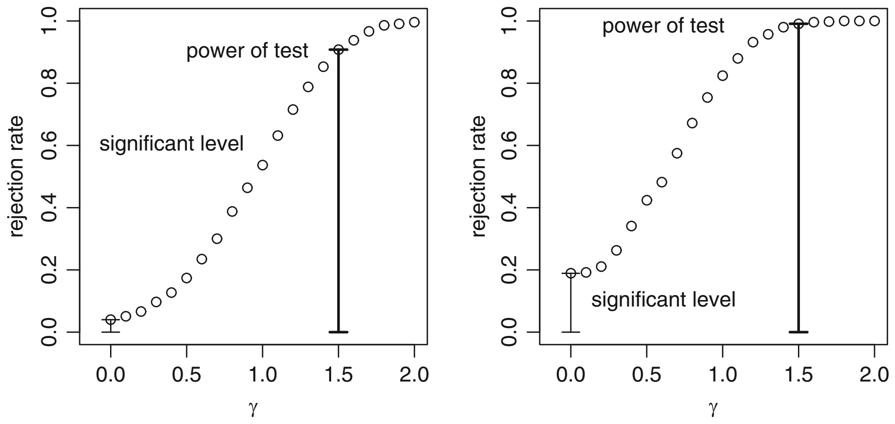


Fig. 2.12 The proportion for which the null hypothesis is rejected when a (two-sided) t-test is conducted with a 5 % risk rate (the number of data is 20) (*left*). The proportion for which the null hypothesis is rejected when a (two-sided) t-test is conducted with a 20 % risk rate (the number of data is 20) (*right*)

$$H_1 : \mu \neq 2.5. \tag{2.52}$$

When the risk rate is set at 5 %, Fig. 2.11 (left) is obtained. $\gamma = 0$ indicates that the null hypothesis holds. Hence, if the rejection proportion is 5 %, it means that the risk rate (significant level) of this test is 5 %. That is, the probability that a type I error will occur is 5 %. When γ is positive, we should reject the null hypothesis because it does not hold. For example, assume a value of $\gamma = 1.5$. Since 51.2 % of the simulation data are rejected when $\gamma = 1.5$, the power of the test is 51.2 %. That is, if the data are generated by Eq. (2.50) with $\gamma = 1.5$, the null hypothesis is rejected with a probability of 51.2 %. This also implies that a type II error occurs with a probability of 48.8 %. When the risk rate is set at 20 %, we have Fig. 2.11 (right). This figure shows that the risk rate is actually about 20 %. Although a test with a high risk rate is not desirable, the power of the test is around 79.9 % when $\gamma = 1.5$. In this sense, the test has become better. That is, there is a tradeoff between a high risk rate and a high power of test.

Figure 2.12 shows the result when the number of data is 20. The risk rate is 5 % and the power of the test with $\gamma = 1.5$ is 85.3 % in Fig. 2.12 (left). On the contrary, the risk rate is 20 % and the power of the test with $\gamma = 1.5$ is 98 % in Fig. 2.12 (right). The power of the tests in these figures is higher than in Fig. 2.11. This means that the number of data must be increased if we wish to enhance the power of the test without increasing the risk rate.

The power of the test is small when γ is small in both Figs. 2.11 and 2.12. That is, when the difference between the mean of a population and the mean in the null hypothesis is small, the probability of missing the difference is augmented. To prevent this mistake, the number of data must be increased. However, when the difference between the mean of a population and the mean in the null hypothesis is small, correct rejection of the null hypothesis is not usually of high value.

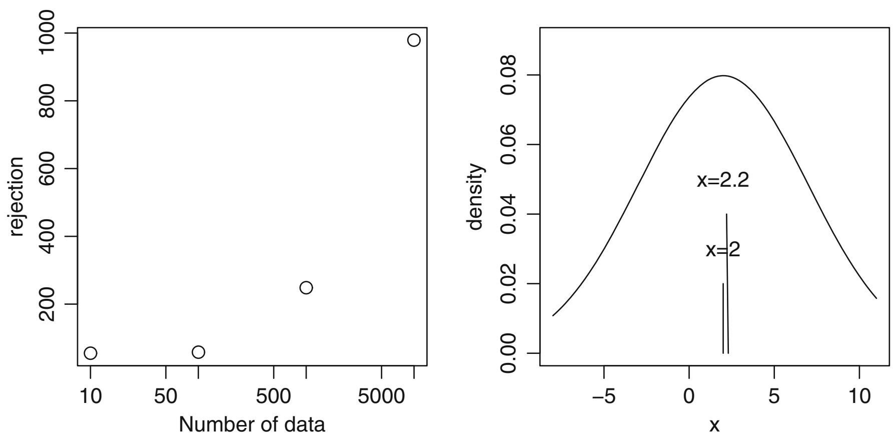


Fig. 2.13 The horizontal axis indicates the number of data and the vertical axis indicates the number of data sets for which the null hypothesis is rejected; the total number of data sets is $1,000$ (*left*). The *curve* shows the probability density function (a normal distribution with mean 2 and variance 5^2) obeyed by the population; the mean in the null hypothesis ($= 2.2$) is also shown (*right*)

Let us use a simulation to realize this. The simulation data $\{x_i\}$ ($1 \le i \le n$, n is the number of simulation data) are realizations of $N(2, 5^2)$ (a normal distribution with mean 2 and variance 5^2). These data are subjected to a t-test. The null hypothesis is "$\mu = 2.2$" (μ is the mean of the population). The alternative hypothesis is "$\mu \neq 2.2$". The number of data (n) is chosen from $\{10, 100, 1,000, 10,000\}$. This simulation was conducted $1,000$ times by changing the initial value of the pseudo-random numbers. The number of times the null hypothesis was rejected with a 5 % risk rate was counted. The result is illustrated in Fig. 2.13 (left).

When $n = 10$, the null hypothesis is rejected 55 times out of $1,000$. This means that the power of the test is 5.5 %. On the other hand, when $n = 10,000$, the null hypothesis is rejected 979 times out of $1,000$, which indicates that the power of the test is 97.9 %. When the number of data is $10,000$, the null hypothesis "$\mu = 2.2$" is rejected using realizations of $N(2, 5^2)$ in most data sets. However, referring to Fig. 2.13 (right), which shows the distribution of the data and the appearance of the null hypothesis, the difference between the mean in the null hypothesis and the average of the data is much smaller than the dispersion of data.

In cases like this, the importance of rejecting the null hypothesis depends on the nature of the data. If an exact theory concerning the phenomenon that generates the data gives "$\mu = 2.2$," the possibility of a phenomenon that denies the hypothesis is of great value. Hence, the result provides incentives for further studies. For example, if the null hypothesis that an object moves slower than the speed of light is rejected, it has serious consequences, even if the speed of the object is only a little faster than light speed. However, in some situations, even if the null hypothesis is rejected, the result does not affect the understanding of the phenomenon, because of the small difference between μ in the null hypothesis and the average of the data. The rejection of the null hypothesis is not of high value for these data. In a psychological test, for example, some test subjects may know the purpose of the test in advance and give suitable responses in line with the purpose. This could cause the null hypothesis to be rejected. We should take such effects into account. Furthermore, even if a new mean value is employed for prediction or control, the adoption of the new mean value is not of great use if the results of the prediction or the control remain almost unchanged. Additionally, various costs of introducing this new mean value should occasionally be considered.

In view of such circumstances, whether a null hypothesis with a risk rate of 5 % is rejected is not the only thing that matters. If rejection of the null hypothesis does not affect the understanding of the phenomenon that generates the data, or has no impact on the prediction and the control, the test does not need to be carried out if the difference between the mean in the null hypothesis and the data is small—at least, the importance of the test result is small. Various concepts of the effect size have been suggested to consider this (e.g., [1]).

R Program [2-6]

Let us investigate the meaning of the risk rate and the power of the test in the *t*-test.

```
ttest61e()
  function (){
  # (1)
    nd <- 10
    mu1 <- 2.5
    nt <- 1000
    gamma1 <- seq(from = 0, to = 2, by = 0.1)
  # (2)
    reject1 <- NULL
    for(ii in 1:length(gamma1)){
      reject1[ii] <- 0
  # (3)
      for (kk in 1:nt){
        set.seed(kk*915)
        d1 <- rnorm(n = nd, mean = 0, sd = 2) + mu1 +
         gamma1[ii]
        tt1 <- t.test(x = d1, mu = mu1)
        if(tt1$p.value <= 0.05){
          reject1[ii] <- reject1[ii] + 1
        }
      }
    }
    reject1 <- reject1/nt
  # (4)
```

```
    nd <- 10
    reject2 <- NULL
    for(ii in 1:length(gamma1)){
      reject2[ii] <- 0
      for (kk in 1:nt){
        set.seed(kk*915)
        d1 <- rnorm(n = nd, mean = 0, sd = 2) + mu1 +
         gamma1[ii]
        tt1 <- t.test(x = d1, mu = mu1)
        if(tt1$p.value <= 0.2){
          reject2[ii] <- reject2[ii] + 1
        }
      }
    }
   reject2 <- reject2/nt
  # (5)

    par(mfrow = c(1,2), mai = c(2, 1, 1.5, 0.1),
     omi = c(0, 0, 0, 0))
    plot(gamma1, reject1,  xlab = expression(gamma),
     ylab = "rejection rate",xlim = c(-0.12,2),
     ylim = c(0,1))
    arrows(0, 0, 0,reject1[1], angle = 90, code = 3,
     length = 0.07)
    text(0.39, 0.32, "significant level")
    arrows(1.5, 0, 1.5,reject1[16], angle = 90,
     code = 3, length = 0.07, lwd = 2)
    text(0.89, 0.56, "power of test")
  # (6)
    plot(gamma1, reject2,  xlab = expression(gamma),
     ylab = "rejection rate", xlim = c(-0.12,2),
     ylim=c(0,1))
    arrows(0, 0, 0,reject2[1], angle = 90, code = 3,
     length = 0.07)
    text(0.6, 0.1, "significant level")
    arrows(1.5, 0, 1.5,reject2[16], angle = 90,
     code = 3, length = 0.07, lwd = 2)
    text(0.92, 0.85, "power of test")
  }
```

(1) The number of data (`nd`), the constant in Eq. (2.50) (page 75) (2.5, `mu1`), and the number of simulations (`nt`) are given. `gamma1` ($\{0, 0.1, 0.2, \ldots, 2\}$) is prepared to provide values of γ (Eq. (2.50), page 75).

(2) The vector `reject1` is set to store the number of rejections of the null hypothesis. The simulation is carried out using one of the elements of `gamma1`.

(3) Simulation data, in which one of the elements of `gamma1` is used as γ, are produced. The t-test is carried out using `t.test()`. The result is named `tt1`. `tt1$p.value` is the p-value. Then, if `tt1$p.value` is less than or equal to 0.05, 1 is added to `reject1[ii]`. The resultant `reject1` is divided by `nt`. This yields the proportion of times the null hypothesis is rejected.
(4) The risk rate for rejecting the null hypothesis is set to 0.2, and the same simulation is conducted. Thus, `reject2` is obtained.
(5) The result given by the risk rate of 0.05 is illustrated in a graph. `arrows()` draws an arrow. Since `angle = 90` is designated here, the angle from the shaft of the arrow to the edge of the arrow head is 90°; it constructs a flat arrow tip. Since `code = 3` is assigned here, arrows are drawn on both ends of the straight line. `length = 0.07` specifies the length of the edges of the arrow head (in inches). `text()` writes some text. The arguments are the X-and Y-coordinate values, and the text to be written, in this order. We obtain Fig. 2.11 (right) (page 75).
(6) The results of the simulation with a risk rate of 0.2 are illustrated in a graph. We obtain Fig. 2.11 (right) (page 75).

R Program [2-6] End

R Program [2-7]

The power of the test is calculated using `power.t.test()`.

The results given by `ttest61e()` are obtained analytically, rather than by simulation. For example, the following R program is used.

```
ttest66()
  function (){
  # (1)
    tp <- power.t.test(n = 10, sd = 2,sig.level = 0.05,
     type = "one.sample", delta = 1.5, alternative =
     "two.sided", strict = T)
  # (2)
    print(tp)
  }
```

(1) `power.t.test()` outputs the power of the test. `n =` specifies the number of data. `sd =` assigns the standard deviation (the positive square root of the variance) of a population. `type = ''one.sample''` specifies the type of the test (`''one.sample''` sets a one-sample test in which a null hypothesis such as Eq. (2.51) (page 75) is used). `delta = 1.5` provides the difference between the mean in the null hypothesis and that of a population. `alternative = ''two.sided''` denotes a two-sided test. `strict = T` indicates that the exact calculation is carried out.
(2) The result of (1) is displayed.

`ttest66()` outputs:

```
One-sample t test power calculation
n = 10
delta = 1.5
sd = 2
sig.level = 0.05
power = 0.5619533
alternative = two.sided
```

R Program [2-7] End

R Program [2-8]

Let us confirm that the proportion of null hypothesis rejections is high when the number of data is large by conducting a *t*-test in R.

```
ttest71e()
  function (){
  # (1)
   par(mfrow = c(1,2), mai = c(2, 1, 1.5, 0.1),
    omi = c(0, 0, 0, 0))
  # (2)
    ndv <- 10^seq(from = 1, to = 4, by = 1)
  # (3)
    mu1 <- 2.2
    reject1 <- NULL
    for(ii in 1:length(ndv)){
      reject1[ii] <- 0
      for (kk in 1:1000){
        set.seed(kk*915)
        d1 <- rnorm(n = ndv[ii], mean = 2, sd = 5)
        tt1 <- t.test(x = d1, mu = mu1)
        if(tt1$p.value <= 0.05){
          reject1[ii] <- reject1[ii] + 1
        }
      }
    }
  # (4)
     plot(ndv, reject1, log = "x", xlab =
      "Number of data", ylab = "rejection", cex = 1.3)
  # (5)
     curve(dnorm(x, mean = 2, sd = 5), from = -8,
      to = 11, n = 101, ylim = c(0, 0.09), xlab = "x",
      ylab = "density")
     lines(c(2, 2), c(0, 0.02))
     text(2, 0.03,"x=2")
     lines(c(2.3, 2.2), c(0, 0.04))
     text(2, 0.05,"x=2.2")
  }
```

(1) `par()` sets the graphics area.
(2) The number of data is chosen as one of $\{10, 100, 1,000, 10,000\}$ and stored in `ndv`.
(3) The null hypothesis that "the mean of the population is 2.2" is set and the t-test is conducted. The data set is `d1`. Every time the number of data is set, $1,000$ realizations are produced. The number of data sets in which the null hypothesis is rejected is stored in `reject1`.
(4) The relationship between the total number of data sets and the number of data sets in which the null hypothesis is rejected is illustrated in a graph. Figure 2.13 (left) (page 77) is obtained.
(5) The probability density function of the population which generates data is drawn. When the mean of the probability density function is compared with the hypothesis that "the mean is 2.2," we find that the difference is very small when compared with the variance of the data (Fig. 2.13 (right)).

R Program [2 - 8] End

2.6 Interval Estimation of Population Variance and the χ^2 Distribution

We obtain n data ($\{x_i^{(1)}\}$ $(1 \le i \le n)$) by sampling from a population. The respective elements of $\{x_i^{(1)}\}$ are realizations of the same normal distribution. The χ^2-value of these data is written as χ^2 and is defined as

$$\chi^2 = \sum_{i=1}^{n} \frac{(x_i^{(1)} - \bar{x}^{(1)})^2}{\sigma^2} = \frac{(n-1)(s^{(1)})^2}{\sigma^2}, \tag{2.53}$$

where $\bar{x}^{(1)}$ is written as

$$\bar{x}^{(1)} = \frac{1}{n} \sum_{i=1}^{n} x_i^{(1)}. \tag{2.54}$$

$(s^{(1)})^2$ is defined in Eq. (2.14) (page 58). σ^2 is called the population variance. That is, it gives the variance of the normal distribution that the population obeys (not the estimate given by the data but the parameter of the population). When m samplings result in $\{x_i^{(j)}\}$ $(1 \le i \le n, 1 \le j \le m)$, the m values of χ^2 obey the χ^2-distribution

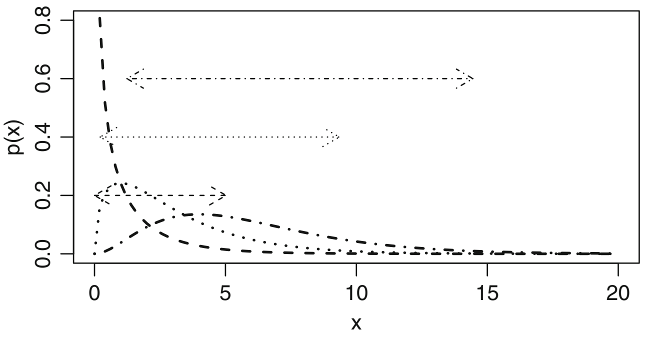


Fig. 2.14 The probability density function of the χ^2-distribution with $\phi = 1$ degrees of freedom is shown by a *dashed line*. The probability density function of the χ^2-distribution with $\phi = 3$ degrees of freedom is shown by a *dotted line*. The probability density function of the χ^2-distribution with $\phi = 6$ degrees of freedom is shown by a *chain line*. The *dashed arrows* show the acceptance region of the probability density function of the χ^2-distribution with $\phi = 1$ when the risk rate is set to 5 %; although the tip of the *left arrow* appears to point to $x = 0.0$, it is actually located near $x = 0.001$. The *dotted* and *chain-line arrows* show the acceptance region of the probability density function of the χ^2-distribution with $\phi = 3$ and $\phi = 6$, respectively, when the risk rate is 5 %

with ϕ degrees of freedom ($\phi = n - 1$). The expectation of the χ^2-distribution with ϕ degrees of freedom is written as $E[\chi^2_\phi]$ (χ^2_ϕ is a random variable). The definition of expectation is given by Eq. (2.6) (page 52). Then, using Eq. (2.20) (page 59), $E[\chi^2_\phi]$ is written as

$$E[\chi^2_\phi] = E\left[\frac{(n-1)S^2}{\sigma^2}\right] = n - 1 = \phi, \tag{2.55}$$

where S^2 is a random variable yielded by Eq. (2.15) (page 58).

Figure 2.14 illustrates the probability density of the χ^2-distribution with $\phi = 1, 3, 6$ degrees of freedom. The figure also shows the acceptance regions of these probability density functions for a risk rate of 5 %. Using these regions, the confidence interval given by the χ^2-value for n data (Eq. (2.53), page 82) is written as

$$\chi^2(\phi, 0.025) < \frac{(n-1)(s^{(1)})^2}{\sigma^2} < \chi^2(\phi, 0.975), \tag{2.56}$$

where $\chi^2(\phi, 0.025)$ is the x value indicated by the tip of the left arrow in Fig. 2.14, and $\chi^2(\phi, 0.975)$ is the x value indicated by the tip of the right arrow in the same figure. The probability density function of the χ^2-distribution denoted by $\chi^2(\phi, 0.025)$ (a nonrandom variable) is the value of χ^2 that satisfies the following equation (this definition is similar to Eq. (2.34), page 63):

$$\int_{-\infty}^{\chi^2} den_{\chi^2}(x)dx = 0.025. \tag{2.57}$$

As with Eq. (2.35) (page 63), $\chi^2(\phi, 0.975)$ (a nonrandom variable) is χ^2 satisfying

$$\int_{-\infty}^{\chi^2} den_{\chi^2}(x)dx = 0.975. \tag{2.58}$$

Equation (2.56) becomes

$$\frac{(n-1)(s^{(1)})^2}{\chi^2(\phi, 0.025)} < \sigma^2 < \frac{(n-1)(s^{(1)})^2}{\chi^2(\phi, 0.975)}. \tag{2.59}$$

This equation represents the confidence interval of σ^2 (the population variance). This equation corresponds to Eq. (2.40) (page 65) in the context of the t-distribution.

We can realize a test of the population variance using the null hypothesis and the alternative hypothesis. This is analogous to Eqs. (2.45) (page 71) and (2.46) (page 71). The null hypothesis, for example, is

$$H_0 : \sigma^2 = 9. \tag{2.60}$$

When a two-sided test is conducted using this null hypothesis, the alternative hypothesis is

$$H_1 : \sigma^2 \neq 9. \tag{2.61}$$

The p-value is defined in a similar way to that of the t-distribution. This test is denoted by the "χ^2-test for homogeneity" or the "χ^2-test for variance in a normal population". The χ^2-test is also applied to test the goodness of fit and the independence of a contingency table.

Next, let us perform a simulation to show the distribution of the χ^2-value (Eq. (2.53), page 82). Simulation data ($\{x_i\}$ $(1 \leq i \leq 4)$) are prepared, where $\{x_i\}$ are 4 realizations of $N(-2, 3^2)$ (a normal distribution with mean -2 and variance 3^2). The procedure of calculating χ^2-values for these 4 data is repeated $10,000$ times by changing the initial value of the pseudo-random numbers. The distribution of the resulting $10,000$ χ^2-values is shown in Fig. 2.15. The constant-multiplied probability density function of the χ^2-distribution with 3 degrees of freedom is superimposed.

Furthermore, let us conduct a simulation to illustrate the implications of the risk rate and the power of the test in the context of the χ^2-test. The simulation data ($\{x_i\}$ $(1 \leq i \leq n)$) are realizations of $N(0, \gamma^2)$ (a normal distribution with mean 0 and variance γ^2). γ^2 is set to one of $\{1, 2, 3, \ldots, 20\}$. A χ^2-test is carried out with the null hypothesis that "the variance is 9." The risk rate is assigned as either 5 or 20 %, and $1,000$ simulations are performed with each setting. The results are shown

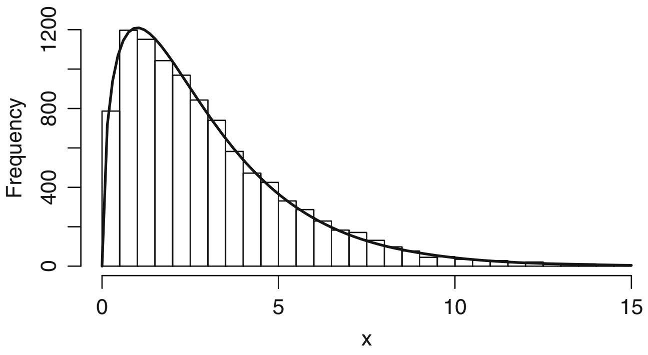


Fig. 2.15 Histogram of the χ^2-values of 10,000 data sets, each consisting of 4 data, and the probability density function of the χ^2-distribution with 3 degrees of freedom. The constant-multiplied probability density function (*solid line*)

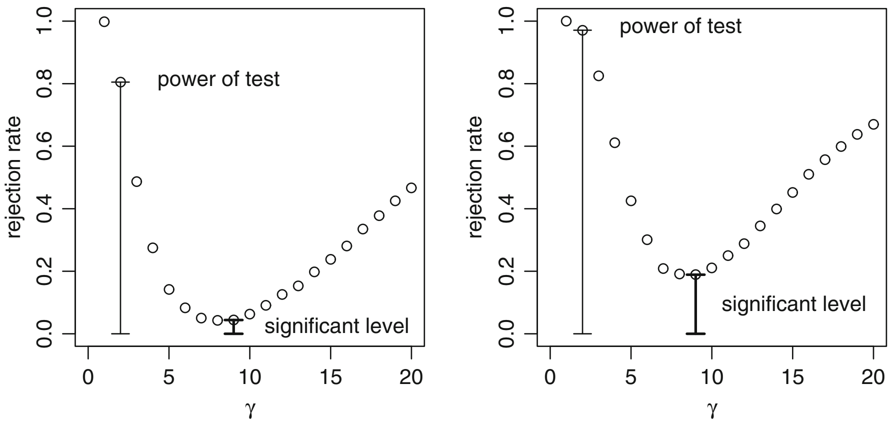


Fig. 2.16 Proportion of times the null hypothesis is rejected when the (two-sided) χ^2-test is conducted with risk rate 5 % using 10 data (*left*). Proportion of times the null hypothesis is rejected when the (two-sided) χ^2-test is conducted with risk rate 20 % using 10 data (*right*)

in Fig. 2.16. When γ^2 is 9, the null hypothesis holds in the population. Hence, the proportion of times the null hypothesis is rejected represents the risk rate (significant level). The results are in line with the settings of 5 and 20 % for the risk rate. When γ^2 differs from 9, the null hypothesis should be rejected. Then, the proportion of times the null hypothesis is rejected is considered the power of the test. This shows that the power of the test is higher the more γ^2 deviates from 9. A comparison between Fig. 2.16 (left) and (right) shows that the power of the test is enhanced by the higher risk rate in this case.

R Program [2 - 9]

Realizations given by pseudo-random numbers result in the probability density function of the χ^2-distribution and the acceptance region when the risk rate is 5 %.

```
chi21e()
  function (){
  # (1)
    par(mfrow = c(1,1), mai = c(2, 1, 1.5, 0.1),
     omi = c(0, 0, 0, 0))
  # (2)
    nd <- 4
    nt <- 10000
  # (3)
    set.seed(181)
    chi <- NULL
    for(ii in 1:nt){
      xx <- rnorm(n = nd, mean = -2, sd = 3)
      xxvar <- var(xx)
      chi[ii] <- (nd - 1) * xxvar/(nd - 1)^2
    }
  # (4)
    xxs <- sort(chi)
    p1 <- xxs[ceiling(nt * 0.025)]
    p2 <- xxs[ceiling(nt * 0.975)]
    print("p1")
    print(p1)
    print("p2")
    print(p2)
  # (5)
    print("qchisq(0.025, df = nd - 1)")
    print(qchisq(0.025, df = nd - 1))
    print("qchisq(0.975, df = nd - 1)")
    print(qchisq(0.975, df = nd - 1))
  # (6)
    chi2 <- chi[ chi <= 15]
    br1 <-  pretty(chi2, n=40)
    bw1 <- br1[2] - br1[1]
    chi2h <- floor(chi2/bw1) * bw1 + 0.01 * bw1
    hist1 <- hist(chi2h, breaks = br1, main = "",
     xlab = "x", ylab = "Frequency")
  # (7)
    curve(dchisq(x, df = nd - 1) * bw1 * nt, min(br1),
     max(br1), xlab = "x", ylab = "p(x)", lwd = 2,
     xlim = c(min(br1), max(br1)),
     ylim = c(0, max(hist1$counts)), add = T)
  }
```

(1) `par()` sets the graphics area.
(2) The number of data in one sampling is denoted by `nd`. This sampling is conducted `nt` times.
(3) `set.seed()` sets the initial value for the pseudo-random numbers. `chi` is prepared in order to store realizations of the χ^2-distribution. Sampling is carried out `nt` times. The χ^2-values of the respective data are calculated. These χ^2-values are stored in `chi`.
(4) `sort()` arranges the elements of `xx` in ascending order. The result is named `xxs`. The value at the 2.5 % position in `xxs` is named `p1`. The value at the 97.5 % position in `xxs` is named `p2`. The values of `p1` and `p2` are displayed.
(5) `qchisq()` calculates the value of $\chi^2(\phi, 0.025)$ (Eq. (2.56), page 83) based on the χ^2-distribution. The resultant value is displayed. `qchisq()` calculates the value of $\chi^2(\phi, 0.975)$ (Eq. (2.56), page 83) based on the χ^2-distribution. The resultant value is displayed.
(6) Values less than or equal to 15 are extracted from `chi`. The resultant values are named `chi2`. `hist()` draws the histogram of `chi2`.
(7) The probability density function of the χ^2-distribution with (`nd-1`) degrees of freedom is plotted.

The results of `chi21e()` are:

```
"p1"
0.2271126
"p2"
9.373766
"qchisq(0.025, df = nd - 1)"
0.2157953
"qchisq(0.975, df = nd - 1)"
9.348404
```

The values obtained by this simulation (i.e., `p1` and `p2`) are close to those obtained from the probability density function (`qchisq(0.025, df = nd - 1)` and `qchisq(0.975, df = nd - 1)`). `chi21e()` also outputs Fig. 2.15 (page 85).

R Program [2-9] End

R Program [2-10]

Let us investigate the meaning of the χ^2-test and the power of the test.

```
chi61e()
  function (){
  # (1)
    nd <- 10
    gamma1 <- seq(from = 1, to = 20, by = 1)
    nt <- 1000
  # (2)
    q1 <- qchisq(0.025, df = nd -1)
    q2 <- qchisq(0.975, df = nd -1)
    v1 <- 9
    reject1 <- NULL
  # (3)
    for(ii in 1:length(gamma1)){
      reject1[ii] <- 0
  # (4)
      for (kk in 1:nt){
        set.seed(kk*915 + 3)
        d1 <- rnorm(n = nd, mean = 0, sd =
         sqrt(gamma1[ii]))
        chi2 <- (nd - 1) * var(d1)/v1
        if(chi2 < q1 | chi2 > q2){
          reject1[ii] <- reject1[ii] + 1
        }
      }
    }
    reject1 <- reject1/nt
  # (5)
    q1 <- qchisq(0.1, df = nd -1)
    q2 <- qchisq(0.9, df = nd -1)
    reject2 <- NULL
    for(ii in 1:length(gamma1)){
      reject2[ii] <- 0
  # (6)
      for (kk in 1:nt){
        set.seed(kk*915 + 3)
        d1 <- rnorm(n = nd, mean = 0, sd =
         sqrt(gamma1[ii]))
        chi2 <- (nd - 1) * var(d1)/v1
        if(chi2 < q1 | chi2 > q2){
          reject2[ii] <- reject2[ii] + 1
        }
      }
    }
    reject2 <- reject2/nt
  # (7)
    par(mfrow = c(1,2), mai = c(1, 1, 1, 0.1),
     omi = c(0, 0, 0, 0))
    plot(gamma1, reject1,  xlab = expression(gamma),
     ylab = "rejection rate",xlim = c(0,20),
     ylim = c(0,1))
    arrows(2, 0, 2,reject1[2], angle = 90, code = 3,
     length = 0.07)
```

```
    text(15.3, 0.02, "significant level")
    arrows(9, 0, 9, reject1[9], angle = 90, code = 3,
     length = 0.07, lwd = 2)
    text(8, 0.81, "power of test")
  # (8)
    plot(gamma1, reject2,  xlab = expression(gamma),
     ylab = "rejection rate", xlim = c(0,20),
     ylim = c(0,1))
    arrows(2, 0, 2,reject2[2], angle = 90, code = 3,
     length = 0.07)
    text(15, 0.09, "significant level")
    arrows(9, 0, 9, reject2[9], angle = 90, code = 3,
     length = 0.07, lwd = 2)
    text(8, 0.98, "power of test")
  }
```

(1) The number of data (`nd`) is given. Values of $\{1, 2, 3, \ldots, 20\}$ are used for γ. The number of simulations (`nt`) is given.
(2) `qchisq()` calculates the value of χ^2 that satisfies Eq. (2.57) (page 84) (i.e., $\chi^2(\phi, 0.025)$). The result is stored in `p1`. `qchisq()` calculates the value of χ^2 that satisfies Eq. (2.58) (page 84) (i.e., $\chi^2(\phi, 0.975)$). The result is stored in `p2`. `v1` is set to 9 so that the null hypothesis is "the variance is 9." `reject1` is prepared in order to store the number of times the null hypothesis is rejected.
(3) Simulations are carried out using the selected value of γ.
(4) The simulation is conducted `nt` times. The simulation data (`d1`) is constructed using one value of γ from `gamma1`. `d1` consists of realizations of a normal distribution with mean 0 and variance `gamma1[ii]`. The χ^2-value is calculated and assigned to `chi2`. If the value of `chi2` falls in the rejection region of the χ^2-test (the two-sided test), 1 is added to `reject1[ii]`. `reject1` is divided by `nt` to give the proportion of rejections.
(5) $\chi^2(\phi, 0.1)$ is calculated and the result is stored in `p1`. $\chi^2(\phi, 0.9)$ is calculated and the result is stored in `p2`. Simulations are carried out with one value extracted from `gamma1`.
(6) The simulation is performed `nt` times. If the simulation data (`d1`) lie in the rejection region of the χ^2-test (two-sided test), 1 is added to `reject2[ii]`. `reject2` is divided by `nt`. Then, we have the proportion of times the null hypothesis has been rejected.
(7) The relationship between γ and the proportion of rejections when the risk rate is 5 % is plotted and explanations are added. Figure 2.16 (left) (page 85) is obtained.
(8) The relationship between γ and the proportion of rejections when the risk rate is 20 % is plotted and explanations are added. Figure 2.16 (right) (page 85) is obtained.

R Program [2 - 10] End

2.7 *F* Distribution and *F*-Test

Suppose that sampling from two populations is carried out. Let m data ($\{x_i\}$ $(1 \leq i \leq m)$) be derived from population-A and let n data ($\{y_i\}$ $(1 \leq i \leq n)$) be derived from population-B. Each $\{x_i\}$ is a realization of a normal distribution ($N(\mu, \sigma^2)$ (a normal distribution with mean μ and variance σ^2)). Each $\{y_i\}$ is a realization of the same normal distribution. The averages of these two data sets are named $\bar{x}$ and $\bar{y}$, respectively. They are defined as follows.

$$\bar{x} = \frac{1}{m}\sum_{i=1}^{m} x_i, \qquad \bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i, \tag{2.62}$$

$$s_x^2 = \frac{1}{m-1}\sum_{i=1}^{m}(x_i - \bar{x})^2, \qquad s_y^2 = \frac{1}{n-1}\sum_{i=1}^{n}(y_i - \bar{y})^2. \tag{2.63}$$

In addition, F_0 is defined as

$$F_0 = \frac{s_x^2}{s_y^2}. \tag{2.64}$$

Another definition of F_0 is

$$F_0 = \begin{cases} \dfrac{s_x^2}{s_y^2} & \text{if } s_x^2 \geq s_y^2 \\[2ex] \dfrac{s_y^2}{s_x^2} & \text{if } s_x^2 < s_y^2. \end{cases} \tag{2.65}$$

F_0 is termed the F-value, the unbiased variance ratio, or the variance ratio. It should be noted that F_0 is a nonrandom variable, despite being represented in upper case. The definition in Eq. (2.65) is of great use when a numerical table is referenced for carrying out the F-test. However, Eq. (2.64) is preferable in the current era. The default `var.test()` implemented in R is used without regard to the magnitude relationship between the two variances.

When $\{x_i\}$ and $\{y_i\}$ in Eq. (2.62) are regarded as random variables, s_x and s_y are considered to be random variables. Then, F_0 also comes to be a random variable. When F_0 is treated as a random variable, we write F_0 as F. The distribution obeyed by F is called the F-distribution. When we set $\phi_1 = m - 1$ and $\phi_2 = n - 1$, the use of Eq. (2.53) (page 82) transforms Eq. (2.64) into

$$F = \frac{\dfrac{\chi^2_{\phi_1}}{\phi_1}}{\dfrac{\chi^2_{\phi_2}}{\phi_2}}, \tag{2.66}$$

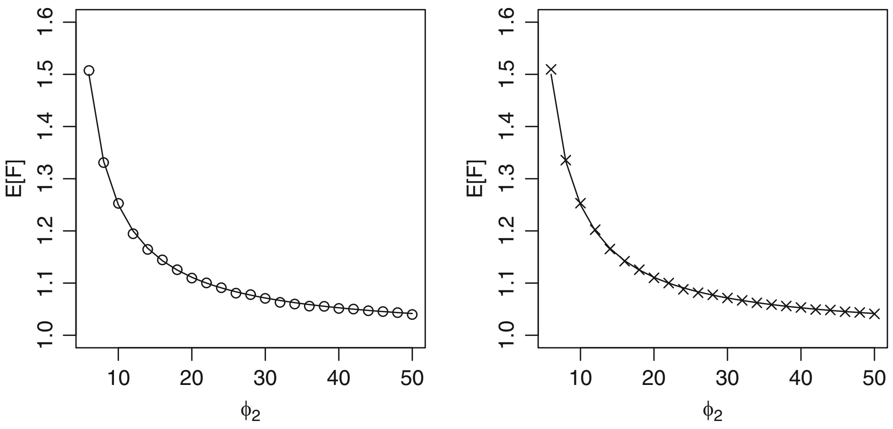


Fig. 2.17 Relationship between ϕ_2 and the average of 100, 000 realizations of the F-distribution with $\phi_1 = 7$. The *solid line* indicates $\frac{\phi_2}{\phi_2 - 2}$ (*left*). Result yielded by the F-distribution with $\phi_1 = 200$ (*right*)

where $\chi^2_{\phi_1}$ (a random variable) and $\chi^2_{\phi_2}$ (a random variable) are given by:

$$\chi^2_{\phi_1} = \sum_{i=1}^{m} \frac{(x_i - \bar{x})^2}{\sigma^2} = \frac{(m-1)s_x^2}{\sigma^2}, \qquad \chi^2_{\phi_2} = \sum_{i=1}^{n} \frac{(y_i - \bar{y})^2}{\sigma^2} = \frac{(n-1)s_y^2}{\sigma^2}. \tag{2.67}$$

$\chi^2_{\phi_1}$ and $\chi^2_{\phi_2}$ are independent of each other. The probability density function obeyed by F in Eq. (2.66) is called the F-distribution with the first degree of freedom (numerator degree of freedom, degree of freedom in the numerator) of ϕ_1 and the second degree of freedom (denominator degree of freedom, degree of freedom in the denominator) of ϕ_2. The expectation of F (i.e., $E[F]$) is as follows (it is independent of ϕ_1):

$$E[F] = \frac{\phi_2}{\phi_2 - 2} \tag{2.68}$$

As the derivation of Eq. (2.68) is not simple, its validity is examined using a simulation. We set $\phi_1 = 7$ and ϕ_2 to one of $\{6, 8, 10, \ldots, 50\}$, and average 100, 000 realizations of the F-distribution. The result of this simulation is illustrated in Fig. 2.17 (left). These averages agree well with the curve of $\frac{\phi_2}{\phi_2 - 2}$. When we set $\phi_1 = 200$, Fig. 2.17 (right) is obtained. These averages also correspond well to the curve of $\frac{\phi_2}{\phi_2 - 2}$. Therefore, Eq. (2.68) appears to be valid.

If Eq. (2.68) is correct, F_0 is close to $\frac{\phi_2}{\phi_2 - 2}$ when the population variance of population-A is equal to that of population-B. If F_0 is very far from $\frac{\phi_2}{\phi_2 - 2}$, we must conclude that the population variance of population-A is different from that of population-B. When this procedure is represented in a similar way to that of Eqs. (2.45) (page 71) and (2.46) (page 71), the null hypothesis is

$$H_0 : \sigma_a^2 = \sigma_b^2, \tag{2.69}$$

and the alternative hypothesis is

$$H_1 : \sigma_a^2 \neq \sigma_b^2. \tag{2.70}$$

In these hypotheses, σ_a^2 stands for the population variance of population-A and σ_b^2 stands for that of population-B. This test is called the "F-test of equality of variances."

In the context of analysis of variance, the subsequent analysis varies only when it is concluded that σ_b^2 is larger than σ_a^2. Hence, we employ the alternative hypothesis:

$$H_1' : \sigma_a^2 < \sigma_b^2. \tag{2.71}$$

For this situation, Eq. (2.64) (page 90) should be used; Eq. (2.65) (page 90) is not suitable.

In a similar way to the t-test, the F-test proceeds on the assumption that the null hypothesis is correct. Then, if we reach a result that makes us think that the occurrence of a low probability event should be assumed, the null hypothesis is rejected. If H_0 (Eq. (2.69), page 92) holds, F_0 obeys the F-distribution. Figure 2.18 illustrates the probability density functions given by a couple of F-distributions. The acceptance region of the 5 % risk rate yielded by the null hypothesis (Eq. (2.69), page 92) and the alternative hypothesis (Eq. (2.70), page 92) is superimposed. When $den_f(x)$ is defined as the probability density function of an F-distribution, the upper limit of the left rejection region is the F (a nonrandom variable) that satisfies

$$\int_{-\infty}^{F} den_f(x)dx = 0.025. \tag{2.72}$$

The lower limit of the right rejection region is the F (a nonrandom variable) that satisfies

$$\int_{-\infty}^{F} den_f(x)dx = 0.975. \tag{2.73}$$

Next, let us conduct a simulation to confirm that the F-value (Eq. (2.64), page 90) obeys the F-distribution if the null hypothesis (Eq. (2.69)) holds. Assume that the sample size (the number of data) of one sampling from population-A is 20 and that from population-B is 10. Both populations obey $N(2, 3^2)$ (a normal distribution with mean 2 and variance 3^2). The F-value (Eq. (2.64), page 90) is calculated $10,000$ times by altering the initial value of the pseudo-random numbers. The histogram of

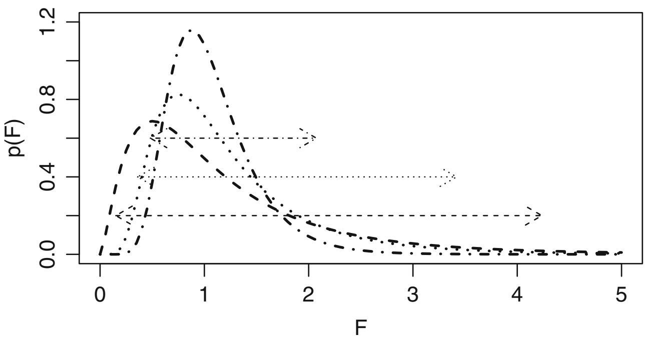


Fig. 2.18 The probability density function of the F-distribution with first degree of freedom $\phi_1 = 5$ and the second degree of freedom $\phi_2 = 10$ is shown by a *dashed line*. The probability density function of the F-distribution with $\phi_1 = 20$ and $\phi_2 = 10$ is shown by a *dotted line*. The probability density function of the F-distribution with $\phi_1 = 30$ and $\phi_2 = 30$ is shown by a *chain line*. The *dashed*, *dotted*, and *chain-line arrows* indicate the acceptance region with a 5 % risk rate for the respective probability density functions

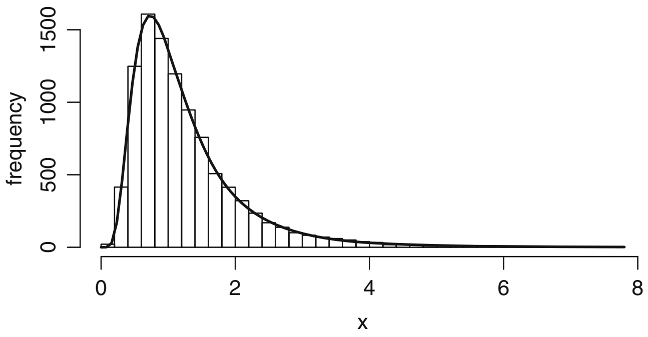


Fig. 2.19 Histogram of the F-values (Eq. (2.64), page 90) when the sample size (the number of data) of one sampling from population-A is 20 and the sample size of one sampling from population-B is 10. Both populations obey a normal distribution with mean 2 and variance 3^2. The constant-multiplied probability density function of the F-distribution with $\phi_1 = 19$ and $\phi_2 = 9$ is also drawn

the resultant distribution is shown in Fig. 2.19. The distribution of these F-values should follow the F-distribution with the first degree of freedom 19 and the second degree of freedom 9. The constant-multiplied probability density function of this F-distribution is superimposed in Fig. 2.19. The two distributions agree well.

Next, let us conduct a simulation to examine the meaning of the risk rate and a null hypothesis in carrying out the F-test. Assume that the size of the sample (the number of data) in one sampling from population-A is 30 ($n = 30$) and the size of the sample in one sampling from population-B is 40 ($m = 40$). The variance of the data sampled from population-A (Var_a) is written as

$$Var_a = \frac{1}{n-1}\Big(\sum_{i=1}^{n} x_i - \frac{1}{n}\sum_{j=1}^{n} x_j\Big)^2 + \gamma. \tag{2.74}$$

The $\{x_i\}$ ($1 \le i \le 30$) are realizations of $N(0, 2^2)$ (a normal distribution with mean 0 and variance 2^2). γ is chosen to be one of $\{0, 0.5, 1.0, \ldots, 5.0\}$. When $\gamma = 0$, the variance of $\{x_i\}$ is that of the normal distribution ($N(0, 2^2)$). When $\gamma \neq 0$, the variance of $\{x_i\}$ differs from that of the normal distribution ($N(0, 2^2)$). This implies that $\{x_i\}$ are realizations of a different distribution to $N(0, 2^2)$. The variance of the data sampled from population-B (Var_b) is written as

$$Var_b = \frac{1}{m-1}\Big(\sum_{i=1}^{m} y_i - \frac{1}{m}\sum_{j=1}^{m} y_j\Big)^2. \tag{2.75}$$

The $\{y_i\}$ ($1 \le i \le 40$) are the realizations of $N(0, 2^2)$ (a normal distribution with mean 0 and variance 2^2). The variance of $\{y_i\}$ is that of the normal distribution ($N(0, 2^2)$). The F-value for these data is defined as

$$F_0 = \frac{Var_a}{Var_b}. \tag{2.76}$$

Then, the null hypothesis is set as "F_0 obeys the F-distribution with the first degree of freedom $\phi_1 = 29$ and the second degree of freedom $\phi_2 = 39$." The alternative hypothesis is set as "F_0 does not obey the F-distribution with the first degree of freedom $\phi_1 = 29$ and the second degree of freedom $\phi_2 = 39$." Using these hypotheses, the F-test (two-sided test) is conducted. This simulation is repeated 1,000 times by altering the initial value of the pseudo-random numbers. The results given by a risk rate (significant level) of 5 % are illustrated in Fig. 2.20 (left) and those given by a risk rate of 20 % are illustrated in Fig. 2.20 (right). This simulation also shows that a higher risk rate results in a higher power of the test.

R Program [2-11]

The acceptance region yielded by the probability density function of the F-distribution with a 5 % risk rate is obtained by realizations given by pseudo-random numbers.

```
f21e()
  function (){
  # (1)
    nd1 <- 20
    nd2 <- 10
    nt <- 10000
  # (2)
    set.seed(199)
    ff <- NULL
```

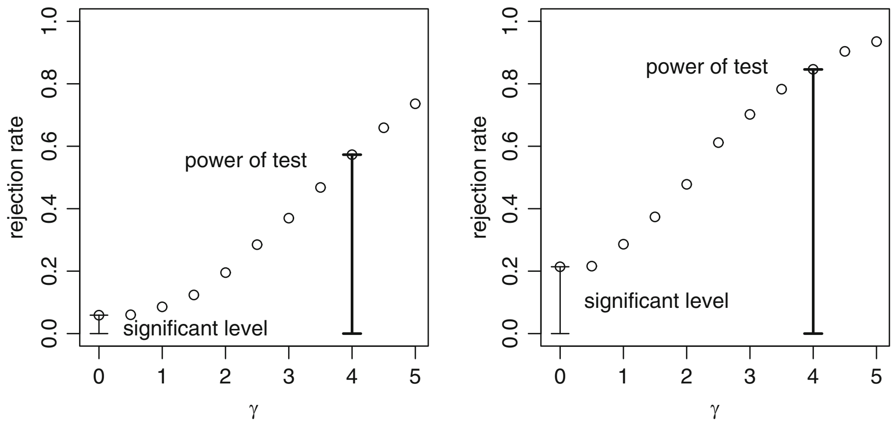


Fig. 2.20 Proportion of times the null hypothesis is rejected when the (two-sided) F-test with a 5 % risk rate is carried out after 30 simulation data are sampled from population-A and 40 simulation data are sampled from population-B (*left*). Results of a similar simulation with a 20 % risk rate (*right*)

```
    for(ii in 1:nt){
      xx <- rnorm(n = nd1, mean = 2, sd = 3)
      xxvar <- var(xx)
      yy <- rnorm(n = nd2, mean = 2, sd = 3)
      yyvar <- var(yy)
      ff[ii] <- xxvar/yyvar
    }
  # (3)
    xxs <- sort(ff)
    p1 <- xxs[ceiling(nt * 0.025)]
    print("p1")
    print(p1)
    p2 <- xxs[ceiling(nt * 0.975)]
    print("p2")
    print(p2)
  # (4)
    print("qf(p = 0.025, df1 = nd1 - 1, df2 = nd2 -1)")
    print(qf(p = 0.025, df1 = nd1 - 1, df2 = nd2 -1))
    print("qf(p = 0.975, df1 = nd1 - 1, df2 = nd2 -1)")
    print(qf(p = 0.975, df1 = nd1 - 1, df2 = nd2 -1))
  # (5)
    par(mfrow = c(1,1), mai = c(2, 1, 1.5, 0.1),
     omi = c(0, 0, 0, 0))
    ff2 <- ff[ ff <= 8]
    br1 <-  pretty(ff2, n = 40)
    bw1 <- br1[2] - br1[1]
    ff2h <- floor(ff2/bw1) * bw1 + 0.01 * bw1
    hist1 <- hist(ff2h, breaks = br1, main="", xlab="x",
     ylab = "Frequency")
  # (6)
    curve(df(x, df1 = nd1-1, df2 = nd2-1) * bw1 * nt,
     min(br1), max(br1), xlab = "x", ylab = "p(x)",
     lwd = 2, xlim = c(min(br1), max(br1)),
     ylim = c(0, max(hist1$counts)), add = T)
  }
```

(1) The sample size (the number of data) of one sampling from population-A is `nd1`. The sample size of one sampling from population-B is `nd2`. This simulation is repeated `nt` times.
(2) `set.seed()` sets an initial value for the pseudo-random numbers. `ff` is prepared in order to store the realizations of the F-distribution. The simulation is repeated `nt` times, with the F-values calculated and accumulated in `ff`.
(3) `sort()` arranges `xx` in ascending order. The result is stored in `xxs`. The value located closest to the 2.5 % point in `xxs` is named `p1`. `p1` is displayed. The value located closest to the 97.5 % point in `xxs` is named `p2`. `p2` is displayed.
(4) `qf()` calculates the value of F that satisfies Eq. (2.72) (page 92), and this value is displayed. The value that satisfies Eq. (2.73) (page 92) is calculated and displayed.
(5) Values less than or equal to 8 in `ff` are named `ff2`. The histogram of `ff2` is drawn.
(6) The constant-multiplied probability density function of the F-distribution with the first degree of freedom `nd1-1` and the second degree of freedom `nd2-1` is illustrated.

`f21e()` outputs:

```
"p1"
0.3400989
"p2"
3.674442
"qf(p = 0.025, df1 = nd1 - 1, df2 = nd2 -1)"
0.3472159
"qf(p = 0.975, df1 = nd1 - 1, df2 = nd2 -1)"
3.683338
```

The results of the simulation using pseudo-random numbers are close to the actual values. `f21e()` also outputs Fig. 2.19 (page 93).

R Program [2 - 11] End

R Program [2 - 12]

Using a simulation, we can also appreciate the risk rate and the power of the test in the F-test.

```
f61e()
  function (){
  # (1)
    nd1 <- 30
    nd2 <- 40
    gamma1 <- seq(from = 0, to = 5, by = 0.5)
    nt <- 1000
  # (2)
    q1 <- qf(0.025, df1 = nd1 - 1, df2 = nd2 - 1)
    q2 <- qf(0.975, df1 = nd1 - 1, df2 = nd2 - 1)
  # (3)
    reject1 <- NULL
    for(ii in 1:length(gamma1)){
      reject1[ii] <- 0
  # (4)
      for (kk in 1:nt){
        set.seed(kk*915 + 7)
        d1 <- rnorm(n = nd1, mean = 0, sd = 2)
        v1 <- var(d1) + gamma1[ii]
        d2 <- rnorm(n = nd2, mean = 0, sd = 2)
        v2 <- var(d2)
        ff <- v1/v2
  # (5)
        if(ff < q1 | ff > q2){
          reject1[ii] <- reject1[ii] + 1
        }
      }
    }
  # (6)
    reject1 <- reject1/nt
  # (7)
    q1 <- qf(0.1, df1 = nd1 -1, df2 = nd2 -1)
    q2 <- qf(0.9, df1 = nd1 -1, df2 = nd2 -1)
    reject2 <- NULL
    for(ii in 1:length(gamma1)){
      reject2[ii] <- 0
      for (kk in 1:nt){
        set.seed(kk*915 + 7)
        d1 <- rnorm(n = nd1, mean = 0, sd = 2)
        v1 <- var(d1) + gamma1[ii]
        d2 <- rnorm(n = nd2, mean = 0, sd = 2)
        v2 <- var(d2)
        ff <- v1/v2
        if(ff < q1 | ff > q2){
          reject2[ii] <- reject2[ii] + 1
        }
      }
    }
    reject2 <- reject2/nt
  # (8)
    par(mfrow = c(1,2), mai = c(2, 1, 1.5, 0.1),
     omi = c(0, 0, 0, 0))
    plot(gamma1, reject1,  xlab = expression(gamma),
     ylab = "rejection rate", xlim = c(-0.1, 5),
     ylim = c(0,1))
    arrows(0, 0, 0, reject1[1], angle = 90, code = 3,
     length = 0.07)
    text(1.5, 0.012, "significant level")
    arrows(4, 0, 4, reject1[9], angle = 90, code = 3,
    length = 0.07, lwd = 2)
    text(2.3, 0.55, "power of test")
  # (9)
    plot(gamma1, reject2,  xlab = expression(gamma),
     ylab = "rejection rate", xlim = c(-0.1, 5),
     ylim = c(0,1))
    arrows(0, 0, 0, reject2[1], angle = 90, code = 3,
     length = 0.07)
    text(1.5, 0.1, "significant level")
    arrows(4, 0, 4, reject2[9], angle = 90, code = 3,
     length = 0.07, lwd = 2)
    text(2.3, 0.85, "power of test")
  }
```

(1) The number of data sampled from population-A is given as `nd1`. That sampled from population-B is given as `nd2`. Values of `gamma1` are selected from $\{0, 0.5, 1.0, \ldots, 5.0\}$. The number of simulations is set as `nt`.
(2) `qf()` calculates the F value that satisfies Eq. (2.72) (page 92). The result is stored in `q1`. `qf()` calculates the F values that satisfy Eq. (2.73) (page 92). The result is saved in `q2`.
(3) Using one value in `gamma1` as γ (Eq. (2.76), page 94), the number of data falling in the rejection region is counted.
(4) After sampling from population-A and population-B, the F-value (Eq. (2.76), page 94) is calculated. The result is stored in `ff`.
(5) If the F-value lies in the rejection region, 1 is added to `reject1[ii]`.
(6) `reject1` is divided by `nt` to give the proportion of rejections.
(7) The risk rate is set to 0.2, and procedures (2)–(6) are carried out. The proportion of rejections (`reject2`) is obtained.
(8) The result from setting the risk rate to 0.05 is plotted. Figure 2.20 (left) (page 95) is displayed.

(9) The result from setting the risk rate to 0.2 is plotted. Figure 2.20 (right) (page 95) is displayed.

R Program [2 - 12] End

2.8 Wilcoxon Signed-Rank Sum Test

The statistical tests treated so far are used when data obey a normal distribution. However, data are not always normally distributed. In addition, if the data contain outliers, simply fitting a normal distribution may not represent the data appropriately. To cope with such situations, it is possible to conduct tests for another specific distribution or eliminate the outliers. However, it is often difficult to find the specific probability distribution that the data obey. The identification of outliers is also not easy in most situations. In such cases, nonparametric tests provide a convenient way to deal with this diversity. Nonparametric tests do not assume that the data obey a normal distribution or another specific distribution.

The simplest nonparametric test is the Wilcoxon signed-rank sum test. This test is created by improving the t-test to cope with a population that does not obey a normal distribution. However, this test does assume that the distribution obeyed by the population is bilaterally symmetric. Let us take an example: consider the data $\{-1.8, 1.5, 1.6, 0.5, -0.4, 1.2, -1.8, 1.6\}$ and the null hypothesis that the mean of the population is equal to -0.4. The algorithm for the Wilcoxon signed-rank sum test is described as follows.

(1) Data values of -0.4 are eliminated from the data set. This yields $\{-1.8, 1.5, 1.6, 0.5, 1.2, -1.8, 1.6\}$. The number of data given by this procedure is named n. We have $n = 7$ in this example.
(2) -0.4 is subtracted from each of the data in the set yielded by (1). Then, we have $\{-1.4, 1.9, 2.0, 0.9, 1.6, -1.4, 2.0\}$.
(3) The absolute values of the data given by (2) are taken. The result is $\{1.4, 1.9, 2.0, 0.9.1.6.1.4, 2.0\}$.
(4) The data set given by (3) is ranked in ascending order. The ranks of tied data values are averaged. Thus, we have $\{2.5, 5, 6.5, 1, 4, 2.5, 6.5\}$.
(5) Data are removed from the set given by (4) if the corresponding values in the set given by (2) are negative (that is, those ranks corresponding to positive values in (2) are retained). This results in the set $\{5, 6.5, 1, 4, 6.5\}$. The summation of these values is named v. Then, we have $v = 23$.
(6) When the mean of the population is -0.4, v given by (5) approximately obeys a normal distribution. The mean (μ) and variance (σ^2) of this normal distribution are:

$$\mu = \frac{1}{4}n(n+1), \tag{2.77}$$

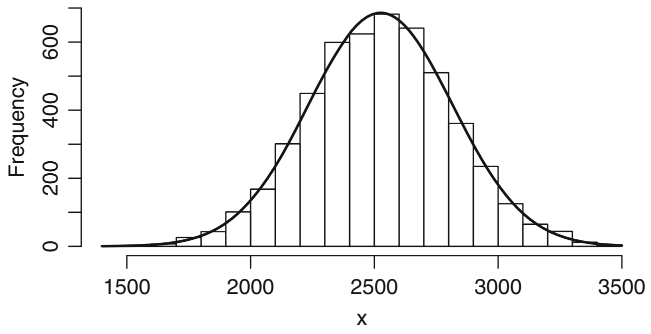


Fig. 2.21 Histogram of v given by 5,000 data sets, each consisting of 100 data. The constant-multiplied probability density function of the normal distribution given by Eqs. (2.77) and (2.78) (*solid line*) is superimposed

$$\sigma^2 = \frac{n(n+1)(2n+1)}{24}. \tag{2.78}$$

(7) If v is located in the rejection region, the null hypothesis is rejected.

Let us conduct a simulation to ascertain that the values of v yielded by (5) in the above algorithm are close to the realizations of the normal distribution defined by Eqs. (2.77) and (2.78). The simulation data are 100 realizations of uniform random numbers between -2 and 8. The mean in the null hypothesis is set to 3. The procedure of calculating v using this simulation data is repeated 5,000 times by changing the initial value of the pseudo-random numbers. The distribution of the resultant values of v is illustrated in Fig. 2.21. The constant-multiplied probability distribution function yielded by Eqs. (2.77) and (2.78) is superimposed. These two distributions agree well.

However, when the number of data is small, we cannot say that v approximately obey a normal distribution even if the null hypothesis holds. In this case, the exact probability should be calculated. Furthermore, when the distribution of v on the assumption of the null hypothesis is approximated by a normal distribution, a correction known as a continuity correction is sometimes applied.

The Wilcoxon signed-rank sum test does not assume that data obey a normal distribution. However, this test does not always surpass the t-test when data form a distribution that is considerably different from a normal distribution. To investigate this point, let us show the results of a simulation to compare the Wilcoxon signed-rank sum test and the t-test. This simulation uses 20 data. These data are realizations of the probability density function obtained by averaging the two probability density functions of $N(0.0, 3+\beta)$ and $N(0.0, -3+\beta)$. The value of β is selected from $\{0, 0.2, 0.4, \ldots, 2.0\}$. An example histogram when $\beta = 1.0$ is shown in Fig. 2.22. For each value of β, 10,000 data sets are produced. The Wilcoxon signed-rank sum test and t-test are carried out using these data sets to determine whether or not the mean is 0. Both tests are two-sided, and the null hypothesis is that the mean of

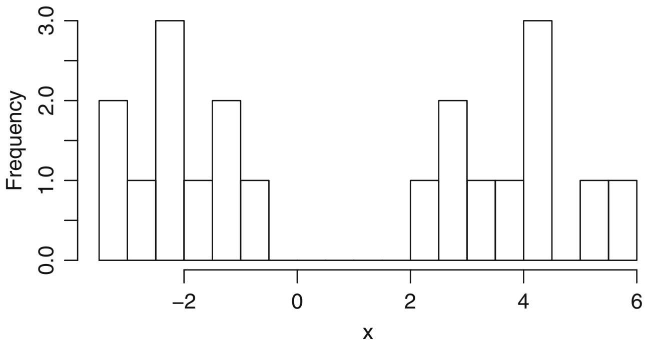


Fig. 2.22 Histogram of simulation data. These data are used to examine characteristics of the Wilcoxon signed-rank sum test when data form a distribution considerably different from a normal distribution ($\beta = 1.0$)

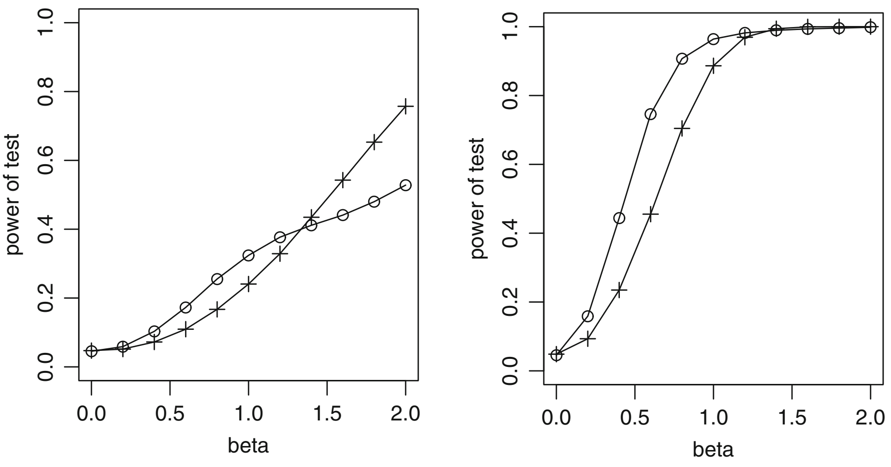


Fig. 2.23 Result of the numerical simulation for determining the nature of the Wilcoxon signed-rank sum test and the t-test when data form a considerably different distribution from a normal distribution. *Open circle* represent the rejection rate (the risk rate and the power of the test) given by the Wilcoxon signed-rank sum test. *Plus symbol* represent the rejection rate (the risk rate and the power of the test) given by the t-test. The number of data is 20 (*left*) and 100 (*right*)

the population is 0. The rates of data that make the p-value lower than 0.05 in the $10,000$ data sets are shown in Fig. 2.23 (left). When $\beta = 0$, this value is the risk rate. When $\beta \neq 0$, this value is the power of the test. The power of the test for the t-test is larger for some values of β. The risk rate is about 0.05, as expected. This result shows that if the distribution of data appears to be far from a normal distribution, we cannot always conclude that the Wilcoxon signed-rank sum test is the better choice. On the other hand, Fig. 2.23 (right) shows the result when the number of data is 100. The power of the test for the Wilcoxon signed-rank sum test is higher in this case.

R Program [2 - 13]

A Wilcoxon signed-rank sum test can be carried out using R.

```
wil1()
  function (){
  # (1)
    set.seed(148)
    xx <- runif(20, min = -4, max = 8)
    mu1 <- 1.9
  # (2)
    xx2 <- xx[xx != mu1]
  # (3)
    rank1 <- rank(abs(xx2 - mu1))
    rank2 <- rank1[xx2 > mu1]
    vv <- sum(rank2)
    print("vv")
    print(vv)
  # (4)
    wtest1 <- wilcox.test(xx, mu = mu1, correct = F,
     exact = F)
    print(wtest1)
    v1 <- wtest1$statistic
  # (5)
    nd <- length(xx2)
    m1 <- 0.25 * nd * (nd + 1)
    sd1 <- sqrt(nd * (nd + 1) * (2 * nd + 1)/24)
    p1 <- (1 - pnorm(v1, mean=m1, sd = sd1)) * 2
    p2 <- pnorm(v1, mean = m1, sd = sd1) * 2
    p3 <- min(c(p1, p2))
    print("--- p3 ---")
    print(p3)
  }
```

(1) `runif()` produces realizations of uniform random numbers. The results are stored in `xx`. The number of data is 20. The minimum value is −4, the maximum value is 8. We test the null hypothesis that the mean of the population that generates these data is `mu1`.
(2) Elements that are identical to `mu1` are deleted from `xx`. The result is stored in `xx2`.
(3) The absolute values of the differences between the values of elements of `xx2` and the value of `mu1` are given an order; the resultant ranks are stored in `rank1`. If there are equal values, the average of the ranks is assigned to all equal values. Elements for which the corresponding `xx2` values are greater than `mu1` are

chosen from the ranked values. The selected values are summed, and the result is named `vv` and is displayed.

(4) `wilcox.test()` conducts a Wilcoxon signed-rank sum test. `correct = F` is set up; a continuity correction is not applied. Since `exact = F` is assigned, it is assumed that v (given in Step (5) in the above algorithm, page 99) obeys a normal distribution, defined by Eqs. (2.77) and (2.78). Since `alternative =` is not assigned, a two-sided test is carried out. The results of the test are saved in `wtest1`. `wtest1` is displayed. Statistics for the use of the Wilcoxon signed-rank sum test, which corresponds to `vv` above, are extracted from `wtest1`; the result is named `v1`.

(5) The number of elements in `xx2` is named `nd`. The p-value (`p3`) of the two-sided test corresponds to the position of `v1` within the normal distribution defined by Eqs. (2.77) and (2.78). `p3` is displayed. The value that appears on the screen is close to that given by (4). `wil1()` outputs:

```
"vv"
138
Wilcoxon signed rank test
data:  xx
V = 138, p-value = 0.218
alternative hypothesis: true location is not
equal to 1.9
"--- p3 ---"
0.2179573
```

R Program [2-13] End

R Program [2-14]

Let us confirm that when the number of data is large, the value of v is close to a realization of the normal distribution defined by Eqs. (2.77) and (2.78).

`wil21e()`

```
function (){
# (1)
  nd <- 100
  nt <- 5000
# (2)
  ff <- NULL
  vvt <- NULL
  for(ii in 1:nt){
    set.seed(144 + ii*6)
    xx <- runif(nd, min = -2, max = 8)
    mu1 <- 3
# (3)
    xx2 <- xx[xx != mu1]
# (4)
    rank1 <- rank(abs(xx2 - mu1))
    rank2 <- rank1[xx2 > mu1]
# (5)
    vvt[ii] <- sum(rank2)
  }
# (6)
  par(mfrow = c(1,1), mai = c(2, 1, 1.5, 0.1),
   omi = c(0, 0, 0, 0))
  br1 <-  pretty(vvt, n = 20)
  bw1 <- br1[2] - br1[1]
  vvth <- floor(vvt/bw1) * bw1 + 0.01 * bw1
  hist1 <- hist(vvth, breaks = br1, main="", xlab="x",
   ylab = "Frequency")
# (7)
  curve(dnorm(x, mean = 0.25 * nd * (nd + 1), sd =
   sqrt(nd * (nd + 1) * (2 * nd + 1)/24) ) * bw1 *
   nt, min(br1), max(br1), xlab = "x", ylab = "p(x)",
   lwd = 2, xlim = c(min(br1), max(br1)),
   ylim = c(0, max(hist1$counts)), add = T)
}
```

(1) The number of data (`nd`) and the number of simulations (`nt`) are given.
(2) `runif(nd, min = -2, max = 8)` generates uniform random numbers between −2 and 8; these are simulation data named `xx`. The mean of the null hypothesis (`mu1`) is set to 3.
(3) Values that are identical to `mu1` are deleted from the data.
(4) `rank()` yields the ranks of the data. These are saved in `rank1`. Values corresponding to larger values than `mu1` are extracted from `rank1`. The obtained values are named `rank2`.
(5) The sum of `rank2` is saved as `vvt[ii]`.
(6) `hist()` draws a histogram of `vvt`.
(7) `curve()` plots the curve of the constant-multiplied probability density function defined by Eqs. (2.77) (page 99) and (2.78) (page 100).

`wil21e()` outputs Fig. 2.21 (page 100).

R Program [2 - 14] End

R Program [2 - 15]

The Wilcoxon signed-rank sum test is carried out when the data obey a distribution that is far from a normal distribution.

```
wil71e()
  function (){
  # (1)
  nb <- 11
  nd <- 20
  nt <- 10000
  pt1t <- NULL
  pt2t <- NULL
  be1t <- NULL
# (2)
  for(jj in 1:nb){
    pt1 <- 0
    pt2 <- 0
    be1 <- (jj - 1) * 0.2
    be1t[jj] <- be1
# (3)
    for(kk in 1:nt){
      set.seed(904 + kk * 144)
      uni1 <- runif(nd, min = 0, max = 1)
      n1 <- length(uni1[uni1 >= 0.5])
      n2 <- nd -n1
      xx <- rnorm(n1, mean = 3 + be1, sd = 1)
      xx <- c(xx, rnorm(n2, mean = -3 + be1, sd = 1))
# (4)
      wtest1 <- wilcox.test(xx, mu = 0, correct = T,
       exact = T)
      p1 <- wtest1$p.value
      if (p1 < 0.05){
        pt1 <- pt1 +1
      }
# (5)
      ttest1 <- t.test(xx, mu = 0)
      p2 <- ttest1$p.value
      if (p2 < 0.05){
        pt2 <- pt2 +1
      }
    }
# (6)
    pt1t[jj] <- pt1/nt
    pt2t[jj] <- pt2/nt
  }
# (7)
  par(mfrow = c(1, 1), mai = c(1, 1, 0.2, 0.2),
   oma = c(1, 1, 1, 1))
  plot(be1t, pt1t , type = "n", xlab = "beta",
   ylab = "power of test", ylim = c(0,1))
  lines(be1t, pt1t )
  points(be1t, pt1t, pch = 1)
```

```
    lines(be1t, pt2t )
    points(be1t, pt2t, pch = 3)
  }
```

(1) It is assumed that β takes `nb` different values. The number of data is set to `nd`. Using `nt` sets of data, the simulations are conducted. `pt1t` is prepared in order to store the values of the risk rate and the power of the test given by the Wilcoxon signed-rank sum test for a value of β. `pt2t` is prepared in order to store the risk rate and the power of the test given by the t-test for each value of β. `be1t` is set in order to save the values of β.
(2) The value of `be1` (β) is set to one of $\{0, 0.2, 0.4, \ldots, 2.0\}$ to generate the simulation data (`xx`). `pt1` is prepared in order to store the number of simulation data for which the null hypothesis is rejected by the Wilcoxon signed-rank sum test. `pt2` is prepared in order to store the number of simulation data for which the null hypothesis is rejected by the t-test. The value of β is given in `be1`. All values used as `be1` are collected in `be1t`.
(3) Simulation data that obey the distribution illustrated in Fig. 2.22 (page 101) are generated and named `xx`.
(4) `wilcox.test()` executes the Wilcoxon signed-rank sum test. The results are stored in `wtest1`. Since `alternative =` is not assigned, a two-sided test is carried out. Since `correct = T` is set, a continuity correction is applied. Furthermore, since `exact = T` is assigned, the exact p-value is calculated. Even if `exact = T` is not assigned and there are no tied data, the exact p-value is calculated if the number of data is less than 50. If the number of data is equal to or greater than 50, the approximate p-value is obtained using the normal distribution defined by Eqs. (2.77) (page 99) and (2.78) (page 100). When `exact = F` is set, the approximate p-value using the normal distribution is calculated at all times. The component of "`p.value`" contained in `wtest1` is the p-value. This value is extracted and named `p1`. When the null hypothesis is rejected with a 5 % risk rate, 1 is added to `pt1`.
(5) `t.test()` carries out the t-test. The result is stored in `ttest1`. Since `alternative =` is not specified, a two-sided test is executed. The component of `p.value` contained in `ttest1` is the p-value. This value is extracted and named `p2`. When the null hypothesis is rejected with a 5 % risk rate, 1 is added to `pt2`.
(6) Values of `pt1` divided by the number of simulation data sets are collected in `pt1t`. Values of `pt2` divided by the number of simulation data sets are collected in `pt2t`.
(7) The impact of the value of β on the risk rate and the power of the test is illustrated in Fig. 2.23 (left).

R Program [2 - 15] End

References

1. Kline RB (2004) Beyond significance testing: reforming data analysis methods in behavioral research. American Psychological Association, Washington
2. Konishi S, Kitagawa G (2008) Information criteria and statistical modeling. Springer, New York
3. Takezawa K (2012) Guidebook to R graphics using Microsoft windows. Wiley, New York

Chapter 3
Simple Regression

3.1 Derivation of Regression Coefficients

When the data $\{(x_i, y_i)\}$ $(1 \leq i \leq n)$ are given, a_0 and a_1 are derived by minimizing the residual sum of squares (RSS) in a procedure called a simple regression:

$$RSS = \sum_{i=1}^{n}(y_i - a_0 - a_1 x_i)^2 = \sum_{i=1}^{n} e_i^2, \tag{3.1}$$

where $(y_i - a_0 - a_1 x_i)$ $(= e_i)$ is a residual. This process yields the regression equation:

$$y = \hat{a}_0 + \hat{a}_1 x, \tag{3.2}$$

where a_0 is the intercept and a_1 is the gradient (slope). Each data point is represented as

$$y_i = \hat{a}_0 + \hat{a}_1 x_i + e_i. \tag{3.3}$$

Values such as a_0 and a_1 are called regression coefficients. The "$\widehat{\ }$" (hat) of $\hat{a}_0$ and $\hat{a}_1$ indicates that these values are estimates.

Estimates are calculated using data $(\{(x_i, y_i)\})$. Therefore, the estimated values are different every time a sampling (trial) is carried out. A variable such as x is called a predictor variable (predictor). A variable such as y, which is yielded using a predictor variable and regression coefficients, is called a target variable. The method of minimizing RSS and deriving regression coefficients is termed the least squares method. Using Eq. (3.2), Eq. (3.1) becomes

$$RSS = \sum_{i=1}^{n}(y_i - \hat{a}_0 - \hat{a}_1 x_i)^2. \tag{3.4}$$

The regression coefficients given by the least squares method ($\hat{a}_0$ and $\hat{a}_1$ here) are described using matrix forms. Data (the number of data points is n) are represented using $\mathbf{X}$ and $\mathbf{y}$ as

$$\mathbf{X} = \begin{pmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_n \end{pmatrix}, \qquad \mathbf{y} = \begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_n \end{pmatrix} = \begin{pmatrix} a_0 + a_1 x_1 \\ a_0 + a_1 x_2 \\ a_0 + a_1 x_3 \\ \vdots \\ a_0 + a_1 x_n \end{pmatrix} + \begin{pmatrix} \epsilon_1 \\ \epsilon_2 \\ \epsilon_3 \\ \vdots \\ \epsilon_n \end{pmatrix}, \tag{3.5}$$

where the column consisting of 1s in $\mathbf{X}$ is used to derive the constant term ($\hat{a}_0$ in Eq. (3.2)) in a simple regression equation. The matrix $\mathbf{X}$ is called a design matrix. The design matrix plays an important role in multiple regression by the least squares method. A design matrix is also called a model matrix. The $\{\epsilon_i\}$ $(1 \leq i \leq n)$ are errors. These errors are realizations of a population with a mean of 0. a_0 and a_1 (Eq. (3.5)) indicate the values (parameters) of a population (i.e., true values). A residual is calculated by subtracting a corresponding estimate given by a regression equation from the data value of the target variable. In contrast, an error is obtained by subtracting the value of the target variable given by the true values of a_0 and a_1 from the data value of the target variable.

Thus, the following equation can be derived:

$$\mathbf{y} = \tilde{\mathbf{y}} + \boldsymbol{\epsilon}, \tag{3.6}$$

where $\tilde{\mathbf{y}}$ and $\boldsymbol{\epsilon}$ are defined as

$$\tilde{\mathbf{y}} = \begin{pmatrix} a_0 + a_1 x_1 \\ a_0 + a_1 x_2 \\ a_0 + a_1 x_3 \\ \vdots \\ a_0 + a_1 x_n \end{pmatrix}, \qquad \boldsymbol{\epsilon} = \begin{pmatrix} \epsilon_1 \\ \epsilon_2 \\ \epsilon_3 \\ \vdots \\ \epsilon_n \end{pmatrix}, \tag{3.7}$$

where a_0 and a_1 are true values (parameters).

$\mathbf{a}$ and $\hat{\mathbf{a}}$ are defined as

$$\mathbf{a} = \begin{pmatrix} a_0 \\ a_1 \end{pmatrix}, \qquad \hat{\mathbf{a}} = \begin{pmatrix} \hat{a}_0 \\ \hat{a}_1 \end{pmatrix}. \tag{3.8}$$

Using Eqs. (3.4) and (3.8), regression coefficients are calculated by minimizing RSS in Eq. (3.1) (page 109). Then, we have

$$\mathbf{X}^t \mathbf{X} \mathbf{a} = \mathbf{X}^t \mathbf{y}. \tag{3.9}$$

Equation (3.9) is termed a normal equation. "^{t}" denotes the transpose of a matrix. Then, $\hat{\mathbf{a}}$ (an estimate of $\mathbf{a}$) is written as

$$\hat{\mathbf{a}} = (\mathbf{X}^t\mathbf{X})^{-1}\mathbf{X}^t\mathbf{y}. \tag{3.10}$$

Hence, the following equation is derived:

$$\hat{\mathbf{y}} = \mathbf{X}\hat{\mathbf{a}} = \mathbf{X}(\mathbf{X}^t\mathbf{X})^{-1}\mathbf{X}^t\mathbf{y}. \tag{3.11}$$

$\hat{\mathbf{y}}$ is described as

$$\hat{\mathbf{y}} = \begin{pmatrix} \hat{y}_1 \\ \hat{y}_2 \\ \hat{y}_3 \\ \vdots \\ \hat{y}_n \end{pmatrix} = \begin{pmatrix} \hat{a}_0 + \hat{a}_1 x_1 \\ \hat{a}_0 + \hat{a}_1 x_2 \\ \hat{a}_0 + \hat{a}_1 x_3 \\ \vdots \\ \hat{a}_0 + \hat{a}_1 x_n \end{pmatrix}. \tag{3.12}$$

Therefore, Eq. (3.11) is represented as

$$\hat{\mathbf{y}} = \mathbf{H}\mathbf{y}. \tag{3.13}$$

The hat matrix $\mathbf{H}$ is written as

$$\mathbf{H} = \mathbf{X}(\mathbf{X}^t\mathbf{X})^{-1}\mathbf{X}^t. \tag{3.14}$$

The term "hat matrix" means that this matrix leads to estimates, and is derived from the fact that an estimate is usually represented by adding a hat ($\widehat{\ }$).

$\mathbf{H}$ is a symmetric matrix. This is proved by transposing the right-hand side of Eq. (3.14) to give

$$\begin{aligned}(\mathbf{X}(\mathbf{X}^t\mathbf{X})^{-1}\mathbf{X}^t)^t &= ((\mathbf{X}^t\mathbf{X})^{-1}\mathbf{X}^t)^t\mathbf{X}^t \\ &= \mathbf{X}((\mathbf{X}^t\mathbf{X})^{-1})^t\mathbf{X}^t \\ &= \mathbf{X}((\mathbf{X}^t\mathbf{X})^t)^{-1}\mathbf{X}^t \\ &= \mathbf{X}(\mathbf{X}^t\mathbf{X})^{-1}\mathbf{X}^t, \end{aligned} \tag{3.15}$$

where Eq. (1.39) (page 30) is used to derive the equations on the second and third lines. The equations on the third and fourth lines are a direct result of $\mathbf{X}^t\mathbf{X}$ being a symmetric matrix. The notion that $\mathbf{X}^t\mathbf{X}$ is a symmetric matrix is proved by writing $\mathbf{X}^t\mathbf{X}$ as

$$\mathbf{X}^t\mathbf{X} = \begin{pmatrix} n & \sum_{i=1}^n x_i \\ \sum_{i=1}^n x_i & \sum_{i=1}^n x_i^2 \end{pmatrix}. \tag{3.16}$$

Furthermore, because $\mathbf{H}$ is given by the data of the predictor variable, it does not depend on the data of the target variable. Then, let us assume that all values of $\{y_i\}$ $(1 \le i \le n)$ are 1s. $\mathbf{1}$ is defined as a column vector in which the number of elements is n and all elements are 1. That is,

$$\mathbf{1} = \begin{pmatrix} 1 \\ 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix}. \tag{3.17}$$

Hence, when $\mathbf{y} = \mathbf{1}$, Eq. (3.13) yields

$$\hat{\mathbf{y}} = \mathbf{H}\mathbf{1}. \tag{3.18}$$

When all the data of the target variable are 1s, all the estimates given by the regression using Eq. (3.1) (page 109) are 1s. Hence, we obtain

$$\mathbf{1} = \mathbf{H}\mathbf{1}. \tag{3.19}$$

This equation indicates that

$$\sum_{j=1}^{n} [\mathbf{H}]_{ij} = 1 \qquad (1 \le i \le n). \tag{3.20}$$

Therefore, regardless of the values of the elements of $\mathbf{y}$, each element of $\hat{\mathbf{y}}$ is a weighted average of the elements of $\mathbf{y}$. That is, the following equations are obtained:

$$\hat{y}_i = \sum_{j=1}^{n} w_{ij} y_j \qquad (1 \le i \le n), \tag{3.21}$$

$$\sum_{j=1}^{n} w_{ij} = 1 \qquad (1 \le i \le n). \tag{3.22}$$

Although $0 \le w_{ij}$ is satisfied in an ordinal weighted average, w_{ij} in Eq. (3.21) can be negative. As the hat matrix of a simple regression is symmetric, Eq. (3.20) (page 112) results in

$$\sum_{j=1}^{n} [\mathbf{H}]_{ij} = \sum_{j=1}^{n} [\mathbf{H}]_{ji} = \sum_{i=1}^{n} [\mathbf{H}]_{ij} = 1, \tag{3.23}$$

where the second equality is obtained by exchanging i and j.

Equation (3.23) gives the relationship between $\{y_i\}$, $\{\hat{y}_i\}$, and $\bar{y}$ (an average of $\{y_i\}$) as follows:

$$\sum_{i=1}^{n} \hat{y}_i = \sum_{i=1}^{n}\sum_{j=1}^{n}[\mathbf{H}]_{ij}\, y_j = \sum_{j=1}^{n} 1 \cdot y_j = \sum_{j=1}^{n} y_j . \tag{3.24}$$

That is, the average $\{y_i\}$ is equal to the average $\{\hat{y}_i\}$. This relationship is satisfied in a more general regression equation (page 41 in [4]). Equation (3.24) can be transformed into

$$\sum_{i=1}^{n}(y_i - \hat{y}_i) = 0. \tag{3.25}$$

Hence, we obtain

$$\sum_{i=1}^{n} e_i = 0. \tag{3.26}$$

That is, the sum of the residuals is 0.

Using Eq. (3.26), the following equation, which is equivalent to Eq. (3.10) (page 111), is derived as follows: First, $\{x_i'\}$ and $\{y_i'\}$ are defined as

$$x_i' = x_i - \bar{x}, \qquad y_i' = y_i - \bar{y}, \tag{3.27}$$

where $\bar{x}$ and $\bar{y}$ are defined as

$$\bar{x} = \frac{\sum_{i=1}^{n} x_i}{n}, \qquad \bar{y} = \frac{\sum_{i=1}^{n} y_i}{n}. \tag{3.28}$$

The transformation of $\{x_i\}$ into $\{x_i'\}$ using Eq. (3.27) is termed centering (pages 11 and 124 in [1], and page 128 in [3]). Then, Eq. (3.1) (page 109) is written as

$$\begin{aligned}
RSS &= \sum_{i=1}^{n}\left(y_i' + \frac{\sum_{i=1}^{n} y_i}{n} - a_0 - a_1\left(x_i' + \frac{\sum_{i=1}^{n} x_i}{n}\right)\right)^2 \\
&= \sum_{i=1}^{n}\left(y_i' - a_0 - a_1 x_i' + \frac{\sum_{i=1}^{n} y_i}{n} - a_1\frac{\sum_{i=1}^{n} x_i}{n}\right)^2 \\
&= \sum_{i=1}^{n}\left(y_i' - a_0 - a_1 x_i' + \frac{\sum_{i=1}^{n}(y_i - a_1 x_i)}{n}\right)^2 \\
&= \sum_{i=1}^{n}\left(y_i' - a_0 - a_1 x_i' + \frac{\sum_{i=1}^{n}(a_0 + e_i)}{n}\right)^2
\end{aligned}$$

$$
\begin{aligned}
&= \sum_{i=1}^{n}\Big(y'_i - a_0 - a_1 x'_i + \frac{\sum_{i=1}^{n} a_0}{n}\Big)^2 \\
&= \sum_{i=1}^{n}\Big(y'_i - a_0 - a_1 x'_i + \frac{n a_0}{n}\Big)^2 \\
&= \sum_{i=1}^{n}(y'_i - a_1 x'_i)^2, \qquad (3.29)
\end{aligned}
$$

where the equality on the fourth and fifth lines are derived using Eq. (3.26). The RSS given by Eq. (3.29) is differentiated with respect to a_1. We obtain

$$
\begin{aligned}
\frac{\partial}{\partial a_1}\sum_{i=1}^{n}(y'_i - a_1 x'_i)^2 &= \sum_{i=1}^{n}\frac{\partial}{\partial a_1}(y'_i - a_1 x'_i)^2 \\
&= \sum_{i=1}^{n} x'_i (y'_i - a_1 x'_i) \\
&= 0. \qquad (3.30)
\end{aligned}
$$

Then, $\hat{a}_1$ is represented as

$$
\begin{aligned}
\hat{a}_1 &= \frac{\sum_{i=1}^{n} x'_i y'_i}{\sum_{i=1}^{n} x'_i x'_i} \\
&= \frac{S_{xy}}{S_{xx}}, \qquad (3.31)
\end{aligned}
$$

where S_{xx} and S_{xy} are defined as

$$
S_{xx} = \sum_{i=1}^{n}(x_i - \bar{x})^2, \qquad S_{xy} = \sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y}). \qquad (3.32)
$$

Substituting Eq. (3.31) into Eq. (3.4) yields

$$
e_i = y_i - \hat{a}_0 - \frac{S_{xy}}{S_{xx}} x_i \qquad (1 \le i \le n). \qquad (3.33)
$$

From Eq. (3.26) (page 113), summation with respect to i results in

$$
\sum_{i=1}^{n} y_i - \hat{a}_0 n - \frac{S_{xy}}{S_{xx}} \sum_{i=1}^{n} x_i = 0. \qquad (3.34)
$$

Hence, we have

$$
\hat{a}_0 = \bar{y} - \frac{S_{xy}}{S_{xx}} \bar{x}. \qquad (3.35)
$$

This equation leads to

$$\bar{y} = \hat{a}_0 + \hat{a}_0\bar{x}, \tag{3.36}$$

which gives the relationship between the average of the predictor variable data and that of the target variable data.

Using Eqs. (3.4) and (3.14), the expectation of Eq. (3.4) (page 109) can be written as

$$\begin{aligned} E[RSS] &= E[(\mathbf{y}-\hat{\mathbf{y}})^t(\mathbf{y}-\hat{\mathbf{y}})] \\ &= E[\mathbf{y}^t(\mathbf{I}-\mathbf{H})^t(\mathbf{I}-\mathbf{H})\mathbf{y}] \\ &= E[\mathbf{y}^t(\mathbf{I}-\mathbf{H})\mathbf{y}] \\ &= E[(\tilde{\mathbf{y}}^t+\boldsymbol{\epsilon}^t)(\mathbf{I}-\mathbf{H})(\tilde{\mathbf{y}}+\boldsymbol{\epsilon})] \\ &= E[\boldsymbol{\epsilon}^t\boldsymbol{\epsilon}-\boldsymbol{\epsilon}^t\mathbf{H}\boldsymbol{\epsilon}], \end{aligned} \tag{3.37}$$

where $\mathbf{I}$ is an identity matrix of size $n \times n$. The above equation comes from the fact that $\mathbf{H}$ is a symmetric matrix (i.e., $\mathbf{H}^t = \mathbf{H}$) (Eq. (3.38)), and the use of the true values ($\tilde{\mathbf{y}}$) as the values of the target variable gives the same true values as the estimates (i.e., $\mathbf{H}\tilde{\mathbf{y}} = \tilde{\mathbf{y}}$).

We can intuitively understand that, when the true values ($\tilde{\mathbf{y}}$) are used as target variables, the same true values are obtained as estimates (i.e., $\mathbf{H}\tilde{\mathbf{y}} = \tilde{\mathbf{y}}$). That is, using the true values as data yields a regression equation to which the data are perfectly fitted. Hence, the estimated values of the target variable using the predictor variable data are identical to the target variable data. This fact can be derived analytically (page 40 in [4]).

Equation (3.37) can be extended using:

$$E[\epsilon_i\epsilon_j] = \begin{cases} \sigma^2 & if\, i = j \\ 0 & if\, i \neq j. \end{cases} \tag{3.38}$$

This equation leads to

$$E[\boldsymbol{\epsilon}^t\boldsymbol{\epsilon}] = E\left[\sum_{i=1}^{n}\epsilon_i\epsilon_i\right] = n\sigma^2, \tag{3.39}$$

$$E[\boldsymbol{\epsilon}^t\mathbf{H}\boldsymbol{\epsilon}] = E\left[\sum_{i=1}^{n}\sum_{j=1}^{n}\epsilon_i[\mathbf{H}]_{ij}\epsilon_j\right] = \text{trace}(\mathbf{H})\sigma^2 = 2\sigma^2, \tag{3.40}$$

where trace($\mathbf{H}$) denotes the trace (the sum of diagonal elements (on-diagonal elements) of a matrix) of $\mathbf{H}$. In a simple regression, trace($\mathbf{H}$) $= 2$ holds (page 43 in [4]). Equations (3.17), (3.39), and (3.40) lead to

$$E[RSS] = (n-2)\sigma^2. \tag{3.41}$$

Hence, the following value is employed as the estimate of σ^2 (i.e., $\hat{\sigma}^2$), given by the data at hand:

$$\hat{\sigma}^2 = \frac{RSS}{n-2}. \tag{3.42}$$

However, this equation is valid only if Eqs. (3.6) (page 110) and (3.7) (page 110) are satisfied. Equation (3.42) cannot be used if the data are not represented as values given by a linear equation plus errors with identical variances, even approximately. This is because $\mathbf{H}\tilde{\mathbf{y}} = \tilde{\mathbf{y}}$ does not hold, meaning that Eq. (3.37) cannot be obtained. On the other hand, if a_1 is 0, that is, the equation generating the data is constant, Eq. (3.41) can be used. Thus, if the equation producing the data is linear or contains a linear equation as a special case, Eq. (3.42) is useful.

Equation (3.41) is the expectation of RSS. This is given by the values of the target variable from a regression equation and the target variable data used to derive the regression equation. If RSS is calculated using the values of the target variable given by a regression equation and the target variable of new data, which are not used for deriving the regression equation, the resultant expectation takes another value. The new data (hereafter termed "data in the future") $\mathbf{y}^*$ are defined as

$$\mathbf{y}^* = \tilde{\mathbf{y}} + \boldsymbol{\epsilon}^*, \tag{3.43}$$

where $\boldsymbol{\epsilon}^*$ can be described as

$$\boldsymbol{\epsilon}^* = \begin{pmatrix} \epsilon_1^* \\ \epsilon_2^* \\ \epsilon_3^* \\ \vdots \\ \epsilon_n^* \end{pmatrix}. \tag{3.44}$$

The $\{\epsilon_i^*\}$ $(1 \leq i \leq n)$ are errors given by the same probability density function as that generating $\{\epsilon_i\}$ $(1 \leq i \leq n)$.

The residual sum of squares for this situation, RSS^*, is written as

$$\begin{aligned} E[RSS^*] &= E[(\mathbf{y}^* - \hat{\mathbf{y}})^t(\mathbf{y}^* - \hat{\mathbf{y}})] \\ &= E[(\tilde{\mathbf{y}} + \boldsymbol{\epsilon}^* - \mathbf{H}(\tilde{\mathbf{y}} + \boldsymbol{\epsilon}))^t(\tilde{\mathbf{y}} + \boldsymbol{\epsilon}^* - \mathbf{H}(\tilde{\mathbf{y}} + \boldsymbol{\epsilon}))] \\ &= E[(\tilde{\mathbf{y}}(\mathbf{I} - \mathbf{H}) + \boldsymbol{\epsilon}^* - \mathbf{H}\boldsymbol{\epsilon})^t(\tilde{\mathbf{y}}(\mathbf{I} - \mathbf{H}) + \boldsymbol{\epsilon}^* - \mathbf{H}\boldsymbol{\epsilon})] \\ &= E[\boldsymbol{\epsilon}^{*t}\boldsymbol{\epsilon}^* + \boldsymbol{\epsilon}\mathbf{H}\boldsymbol{\epsilon}] \\ &= n\sigma^2 + \text{trace}(\mathbf{H})\sigma^2 \\ &= (n+2)\sigma^2. \end{aligned} \tag{3.45}$$

R Program [3 - 1]

Let us confirm that the regression coefficients given by Eq. (3.10), Eqs. (3.31) (page 114) and (3.35) (page 114), and `lm()` are identical.

```
simple()
  function ()
  {
  # (1)
    set.seed(813)
    nd <- 20
    xx <- seq(from = 1, to = nd, by = 1)
    yy <- xx *2.5 -14 + rnorm(nd, mean = 0, sd = 3)
  # (2)
    xxm <- matrix( c(rep(1, length = nd), xx), ncol = 2)
    yym <- matrix(yy, ncol = 1)
  # (3)
    print("Result of Eq.(3.10)")
    print("aa")
    print(aa)
    ey <- aa[1] + aa[2] * xx
  # (4)
    aa1 <- sum((xx - mean(xx)) * (yy - mean(yy))) /
     sum((xx - mean(xx))^2)
    print("Result of Eq.(3.31)")
    print("aa1")
    print(aa1)
    aa0 <- mean(yy) - aa1 * mean(xx)
    print("Result of Eq.(3.35)")
    print("aa0")
    print(aa0)
  # (5)
    data1 <- data.frame(x = xx, y = yy)
    lm1 <- lm(y~x, data = data1)
    print(lm1)
  # (6)
    par(mfrow = c(1,1), mai = c(2, 1, 1.5, 0.1),
     omi = c(0, 0, 0, 0))
    plot(xx, yy, xlab = "x", ylab = "y", type = "n")
    points(xx, yy)
    lines(xx, ey, lwd = 1)
  }
```

(1) `set.seed(813)` sets an initial value for the generation of pseudo-random numbers. The number of data points (`nd`) is given. The predictor variable data are stored in `xx` and the target variable data are saved in `yy`.

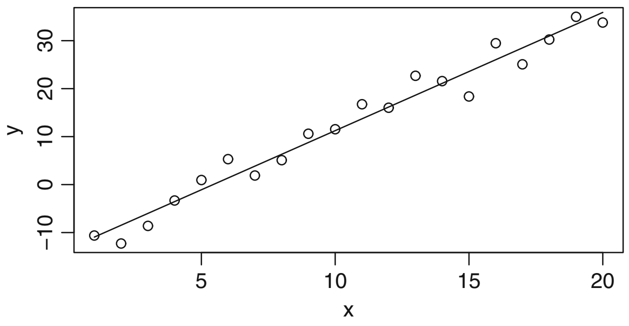


Fig. 3.1 Result using simple regression. *Circle* denotes data and the *solid line* depicts estimates

(2) `xx` is combined with $\mathbf{1}$ (Eq. (3.17) (page 112)) to produce $\mathbf{X}$ (Eq. (3.5) (page 110). The resultant matrix is called `xxm`. `yy` is transformed into a matrix form to give $\mathbf{y}$ (Eq. (3.5) (page 110)); the result is named `yym`.
(3) The regression coefficients (a_0, a_1) are calculated using Eq. (3.10) (page 111). The values of $\hat{\mathbf{a}}$ (Eq. (3.8) (page 110)) are stored in `aa`. Using `aa`, estimates corresponding to the respective elements of `xx` are calculated and saved as `ey`.
(4) $\hat{a}_1$, derived using Eq. (3.31) (page 114), is named `aa1`. This value is displayed in the console window. $\hat{a}_0$ from Eq. (3.35) (page 114) is termed `aa0`. This value is also displayed in the console window.
(5) `xx` and `yy` are combined in the data frame named `data1`. In this data frame, `xx` is called `x` and `yy` is called `y`. `lm()` carries out a simple regression using `data1` as data. The result is stored in `lm1`. `print()` displays the contents of `lm1` in the console window.
(6) `par()` sets the graphics area. `plot()` draws coordinate axes. `type = "n"` specifies that only coordinate axes are drawn. `points()` graphs the data points. `lines()` draws estimates given by a simple regression, and Fig. 3.1 (page 118) is obtained.

`simple()` also outputs:

```
"Result of Eq.(3.10)"
"aa"
[,1]
-13.406846
2.464767
"Result of Eq.(3.31)"
"aa1"
2.464767
"Result of Eq.(3.35)"
"aa0"
-13.40685
Call:
```

```
lm(formula = y ~ x, data = data1)
Coefficients:
(Intercept)           x
    -13.407       2.465
```

R Program [3 - 1] End

3.2 Exchange Between Predictor Variable and Target Variable

When a simple regression using the least squares method is carried out, we may sometimes be unsure which variable should be adopted as the predictor variable. For example, $\{x_i\}$ $(1 \le i \le n)$ is assumed to be a series of observations by a measurement device, and $\{y_i\}$ $(1 \le i \le n)$ is assumed to be those by another device. x_i and y_i are observations of the same object. If a linear relationship is found between the two series of observations, we think of the two regression equations as follows:

$$y_i = a_0 + a_1 x_i + \epsilon_i, \tag{3.46}$$

$$x_i = b_0 + b_1 y_i + \epsilon'_i. \tag{3.47}$$

It is not easy to choose one regression equation from the above two equations to cope with this type of data.

For example, 40 sets of simulation data are produced using

$$y_i = 2 + x_i + \epsilon_i. \tag{3.48}$$

Here, $\{x_i\}$ $(1 \le i \le 40)$ are $\{1, 2, 3, \ldots, 40\}$. The $\{\epsilon_i\}$ $(1 \le i \le 20)$ are realizations of $N(0, 5^2)$ (normal distribution; mean of 0 and variance of 5^2). Figure 3.2 shows the distributions of $\hat{a}_0$ and $\hat{a}_1$ yielded by $2,000$ simulations using pseudo-random numbers with varying the initial value. The mean $\hat{a}_0$ is 2.015736 and the mean $\hat{a}_1$ is 0.9999637. Figure 3.3 illustrates the distribution of $\hat{b}_0$ and $\hat{b}_1$ given by $2,000$ simulations using pseudo-random numbers with varying the initial value. The mean $\hat{b}_0$ is 1.352609 and the mean $\hat{b}_1$ is 0.8504109. These outcomes indicate that, when the predictor and target variables are exchanged, the resulting regression is not simply inverted. Therefore, when a simple regression using two variables is carried out, the choice of predictor variable can have a considerable impact on the results and their interpretation.

R Program [3 - 2]

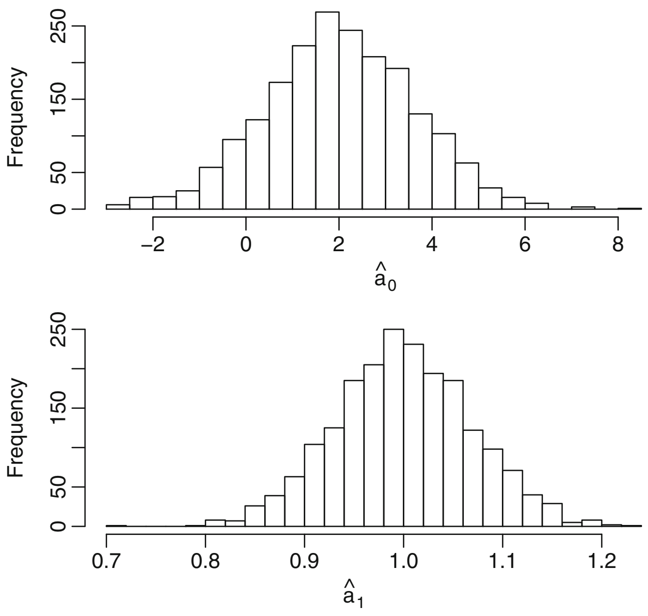


Fig. 3.2 Distributions of $\hat{a}_0$ and $\hat{a}_1$. The mean $\hat{a}_0$ is 2.015736. The mean $\hat{a}_1$ is 0.9999637. Simulation data were produced using Eq. (3.48). $2,000$ simulations to obtain a regression equation in the form of Eq. (3.46) were carried out

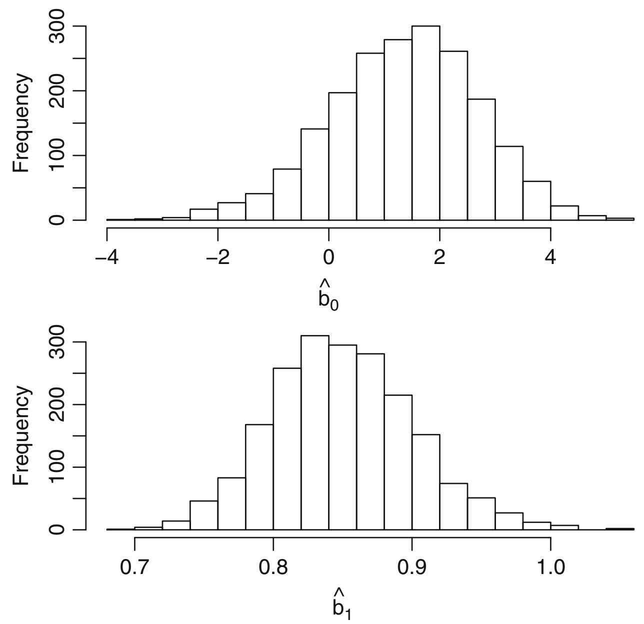


Fig. 3.3 Distributions of $\hat{b}_0$ and $\hat{b}_1$. The mean $\hat{b}_0$ is 1.352609. The mean $\hat{b}_1$ is 0.8504109. Simulation data were produced using Eq. (3.48). $2,000$ simulations to obtain a regression equation in the form of Eq. (3.47) were carried out

When a predictor variable and a target variable are exchanged, the resultant regression equation is not an inverse function.

```
simp3e()
  function ()
  {
  # (1)
    nd <- 40
    xx <- seq(from = 1, to = nd, by = 1)
    bb0v <- NULL
    bb1v <- NULL
  # (2)
    for(kk in 1:2000){
      set.seed(813 + kk * 7)
  # (3)
      yy <- 2 + xx + rnorm(nd, mean = 0, sd = 5)
      data1 <- data.frame(x = yy, y = xx)
      lm1 <- lm(y~x, data = data1)
      bb0v[kk] <- lm1$coef[1]
      bb1v[kk] <- lm1$coef[2]
    }
  # (4)
    par(mfrow = c(2,1), mai = c(1, 1, 0.5, 0.1),
     omi = c(0, 0, 0, 0))
  # (5)
    br1 <-  pretty(bb0v, n = 20)
    bw1 <- br1[2] - br1[1]
    bb0vh <- floor(bb0v / bw1) * bw1 + 0.01 * bw1
    hist(bb0vh, breaks = br1, main = "", xlab =
     expression(hat(b)[0]))
    br2 <-  pretty(bb1v, n=20)
    bw2 <- br2[2] - br2[1]
    bb1vh <- floor(bb1v / bw2) * bw2 + 0.01 * bw2
    hist(bb1vh, breaks = br2, main = "", xlab =
     expression(hat(b)[1]))
  # (6)
    print(mean(bb0v))
    print(mean(bb1v))
  }
```

(1) The number of data points (`nd`) is given. The values of the predictor variable data (`xx`) are set as $\{1, 2, 3, \ldots, 40\}$. `bb0v` is used to store $\hat{b}_0$ (Eq. (3.47) (page 119)), and `bb1v` is used to store $\hat{b}_1$ (Eq. (3.47)).

(2) 2, 000 simulations are carried out using pseudo-random numbers with varying the initial value.

(3) The values of the target variable are saved in `yy`. The results of the simple regression are saved as `lm1`. Regression coefficients are stored in `bb0v` and `bb1v`.
(4) `par()` sets the graphics area.
(5) `hist()` draws a histogram illustrating the distribution of `bb0v`. Figure 3.3 (page 120) is obtained.
(6) The averages of `bb0v` and `bb1v` are displayed in the console window.

`simp3()` also outputs:

```
1.352609
0.8504109
```

R Program [3-2] End

3.3 Regression to the Mean

Let us assume that we have a set of plants or animals. The number of individuals is n. Their weights and sizes ($\{x_i^{<1>}\}$ $(1 \le i \le n)$) are assumed to remain constant over time. Each individual produces a child, and the sizes or weights of the children are denoted by $\{x_i^{<2>}\}$ $(1 \le i \le n)$. Thus, n weights or sizes of k generations ($\{x_i^{<k>}\}$ $(1 \le i \le n, 1 \le k \le K)$) are observed. Assume the following equation for describing this data:

$$x_i^{<k+1>} = \gamma x_i^{<k>} + \frac{(1-\gamma)\sum_{j=1}^n x_j^{<k>}}{n} + \epsilon_i^{<k>}, \tag{3.49}$$

where γ is a positive constant and $\{\epsilon_i^{<k>}\}$ are realizations of $N(0, \sigma^2)$ (normal distribution; mean of 0 and variance of σ^2). We assume that there is no error in the observation of an individual. Hence, $\{\epsilon_i^{<k>}\}$ are due to random impacts from genetic uncertainty and environmental disorder. The term $\dfrac{(1-\gamma)\sum_{j=1}^n x_j < k >}{n}$ works to make the average $\{x_i^{<k>}\}$ in terms of i independent of k. Then, the summation of Eq. (3.49) in terms of i leads to

$$\begin{aligned}
\sum_{i=1}^n x_i^{<k+1>} &= \sum_{i=1}^n \gamma x_i^{<k>} + \sum_{i=1}^n \frac{(1-\gamma)\sum_{j=1}^n x_j^{<k>}}{n} + \sum_{i=1}^n \epsilon_i^{<k>} \\
&= \sum_{i=1}^n \gamma x_i^{<k>} + (1-\gamma)\sum_{j=1}^n x_j^{<k>} + \sum_{i=1}^n \epsilon_i^{<k>} \\
&= \sum_{i=1}^n x_i^{<k>} + \sum_{i=1}^n \epsilon_i^{<k>}
\end{aligned}$$

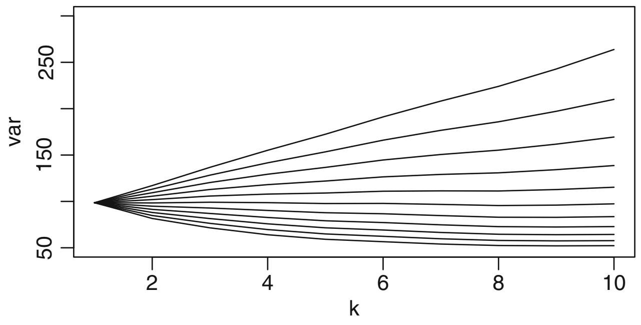


Fig. 3.4 Relationship between k and $\text{var}(\{x_i^{<k>}\})$ for a fixed γ. Each curve corresponds, in ascending order, to $\gamma = \{0.8, 0.82, 0.84, \ldots, 1.0\}$

$$\approx \sum_{i=1}^{n} x_i^{<k>}. \tag{3.50}$$

The approximation from the third to the fourth line is based on the equation $E[\epsilon_i^{<k>}] = 0$, where $\epsilon_i^{<k>}$ is regarded as a random variable. Equation (3.50) indicates that the average of $\{x_i^{<k>}\}$ in terms of i is independent of k. Next, the variance of $\{x_i^{<k>}\}$ ($\text{var}(\{x_i^{<k>}\})$) for a fixed k is defined as

$$\text{var}(\{x_i^{<k>}\}) = \frac{1}{n-1}\sum_{i=1}^{n}(x_i^{<k>} - \frac{1}{n}\sum_{j=1}^{n} x_j^{<k>})^2. \tag{3.51}$$

Because Eq. (3.49) is assumed to hold, the average $\{x_i^{<k>}\}$ in each generation remains the same. Therefore, if $\text{var}(\{x_i^{<k>}\})$ remains almost constant, the characteristics as a set remain substantially the same, because the mean and variance are nearly independent of k.

Now, let us conduct a simulation to find γ (Eq. (3.49) (page 122)) such that the variance is independent of k. The number of data points (n) is set at $10,000$. $\{x_i^{<1>}\}$ are realizations of $N(50, 10^2)$ (normal distribution; mean of 50 and variance of 10^2). $\{\epsilon_i^{<k>}\}$ are realizations of $N(0, 4.3^2)$ (normal distribution; mean of 0 and variance of 4.3^2). γ is set to one of the 11 numbers in the set $\{0.8, 0.82, 0.84, \ldots, 1.0\}$. The simulated results from $k = 2$ through $k = 10$ are shown in Fig. 3.4. This figure shows that $\text{var}(\{x_i^{<k>}\})$ remains almost constant when $\gamma = 0.9$. Hence, assigning $\gamma = 0.9$ in Eq. (3.49) ensures that the characteristics as a set remain virtually constant.

When $\gamma = 1$ in Eq. (3.49), $E[x_i^{<k+1>}] = x_i^{<k>}$ holds. This indicates that no hereditary direction is observed. The variance of the set, however, becomes larger. Therefore, when $\gamma = 0.9$, that is, when there is a hereditary direction towards the mean, an equilibrium set is realized. Focusing on the case $\gamma = 0.9$, it shows that a larger individual brings a smaller child than his or her parent to the world, and a smaller individual brings a larger child than his or her parent to the world. Hence, members of the set appear to get closer to the mean. However, the diversification effect brought about by $\{\epsilon_i^{<k>}\}$ cancels this out. Therefore, when $\gamma = 0.9$, a set of almost constant variance is realized. This phenomenon is called "regression to the mean" (regression towards the mean). This regression does not refer to the production of a regression equation; it means "going back".

In addition, when the characteristics of a set are almost independent of k, Eq. (3.49) (page 122) is transformed into

$$x_i^{<k-1>} = \gamma x_i^{<k>} + \frac{(1-\gamma)\sum_{j=1}^{n} x_j^{<k>}}{n} + \epsilon_i^{<k>\prime}, \tag{3.52}$$

where $\epsilon_i^{<k>\prime}$ and $\epsilon_i^{<k>}$ are realizations of $N(0, \sigma^2)$ (normal distribution; mean of 0 and variance of σ^2). When the data of the parents are represented using those of the children, we obtain a linear equation with a slope of less than 1 plus a random number obeying a normal distribution. That is, while some hereditary information is lost from parents to children, some information that the children have is lost in the parents.

Next, let $\{x_i^{<k>}\}$ be the score from the k-th examination of n students in a school. The difficulty level of the examinations is assumed to be kept the same. The problem is to predict the scores of the $(k+1)$-th examination when the 1st to k-th examinations have finished. The mean and variance of $\{x_i^{<k>}\}$ are assumed to be identical to those of $\{x_i^{<k+1>}\}$. The above simulation indicates that, if $x_i^{<k>}$ is larger than the mean, the estimate of $x_i^{<k+1>}$ should be less than $x_i^{<k>}$. However, if the i-th student is a member of another school in which the average score is higher than $x_i^{<k>}$, the estimate of $x_i^{<k+1>}$ should be larger than $x_i^{<k>}$. A similar phenomenon is often observed when the Stein estimator (James–Stein estimator) is employed. Whether or not this sort of estimation is justified is not a mathematical problem, but is instead judged from the standpoint of the characteristics of the data and the purpose of the estimation.

When it comes to examination results, a slightly different form may be appropriate. We represent $x_1^{<k>}$ as U_1 (random variable) and assume

$$U_i = v_i + \epsilon_i, \tag{3.53}$$

where v_i indicates the intellectual ability of the i-th student. It is assumed that the intellectual ability of the i-th student has a unique value depending only on i. In other words, $\{v_i\}$ takes the same value for every trial. However, $\{v_i\}$ $(1 \leq i \leq n)$ are realizations of $N(0, 1^2)$ (normal distribution; mean of 0 and variance of 1^2). $\{\epsilon_i\}$ $(1 \leq i \leq n)$ are random errors caused by the characteristics of a test. They are realizations of $N(0, \sigma_2^2)$ (normal distribution; mean of 0 and variance of σ_2^2). Hence, the $\{\epsilon_i\}$ take different values in each trial. That is, they are random variables. Therefore, the probability density function followed by ϵ_i is as below.

$$den(\epsilon_i) = \left(\frac{1}{\sqrt{2\pi\sigma_2^2}}\right)\exp\left(-\frac{1}{2\sigma_2^2}\epsilon_i^2\right). \tag{3.54}$$

Let us assume that $\mu_1 = 50$ and $U_1 = 65$. As the average of $\{\epsilon_i\}$ is roughly 0, the average score of this test is about 50 and the score of the first student is 65. Thus, we have

$$65 = v_1 + \epsilon_1. \tag{3.55}$$

Then, the following equation is obtained.

$$\epsilon_1 = 65 - v_1. \tag{3.56}$$

Therefore, the expectation of v_1 is written as

$$E[v_1] = \frac{\int_{-\infty}^{\infty} v_1\left(\frac{1}{\sqrt{2\pi\sigma_1^2}}\right)\exp\left(-\frac{1}{2\sigma_1^2}(v_1-50)^2\right)\left(\frac{1}{\sqrt{2\pi\sigma_2^2}}\right)\exp\left(-\frac{1}{2\sigma_2^2}(65-v_1)^2\right)dv_1}{\int_{-\infty}^{\infty}\left(\frac{1}{\sqrt{2\pi\sigma_1^2}}\right)\exp\left(-\frac{1}{2\sigma_1^2}(v_1-50)^2\right)\left(\frac{1}{\sqrt{2\pi\sigma_2^2}}\right)\exp\left(-\frac{1}{2\sigma_2^2}(65-v_1)^2\right)dv_1}. \tag{3.57}$$

When we assume $\sigma_1^2 = 100$ and $\sigma_2^2 = 36$, we have $E[v_1] = \frac{2075}{34}(\approx 61.02941)$; this value was derived using Mathematica 3.0. However, when $\mu_1 = 70$ and the other settings remain the same, $E[v_1] = \frac{2255}{34}(\approx 66.32353)$ is obtained; this value was also derived using Mathematica 3.0. That is, whereas the intellectual ability of student $i = 1$ is fixed at v_1, the value of $E[v_1]$ depends on which group student $i = 1$ belongs to. This is another result of regression to the mean.

It may appear odd that, while v_1 is assumed to take the same value in every trial, this value is regarded as a random variable in Eq. (3.57). However, an unknown nonrandom variable is often treated as a random variable in Bayesian Inference. On the other hand, when it is emphasized that v_1 is not a random variable, the value 65 is obtained by adding errors with mean 0 (ϵ_1) to v_1. Then, the estimate of v_1 comes to be 65. In this situation, the group to which student $i = 1$ belongs to does not affect the estimate of v_1.

Estimation methods such as Eq. (3.57) are the origin of the linear mixed model (page 95 in [2]). Hence, when the linear mixed model is used practically, the estimates are affected by the group to which the individual or individuals who generate the data belong. This point should be taken into account when interpreting the results of the linear mixed model in practical situations.

R Program [3 - 3]

Estimation of γ (Eq. (3.49) (page 122)) such that the variance of a set remains constant.

```
simp5()
  function()
  {
  # (1)
    nd <- 10000
    nt <- 10
  # (2)
    v1m <- matrix(rep(0, length = 11 * nt), ncol = 11)
  # (3)
    for(jj in c(1:11)){
      gamma1 <- 0.78 + jj * 0.02
      set.seed(815)
      xx <- rnorm(nd, mean = 50, sd = 10)
      v1 <- NULL
      v1[1] <- var(xx)
  # (4)
      for(ii in 1:(nt-1)){
        xx2 <-  gamma1 * xx + (1 - gamma1) *
         sum(xx) / nd  + rnorm(nd, mean = 0,
         sd = 4.3)
        v1[ii + 1] <- var(xx2)
        xx <- xx2
      }
  # (5)
      v1m[,jj] <- v1
    }
  # (6)
    par(mfrow = c(1,1), mai = c(2, 1, 1.5, 0.1),
     omi = c(0, 0, 0, 0))
    yrange <- range(as.vector(v1m))
    yrange1 <- min(pretty(yrange))
    yrange2 <- max(pretty(yrange))
    plot(c(1:nt), v1, pch = 1, type="n", xlab = "k",
     ylab = "var", ylim=c(yrange1, yrange2))
    for (jj in 1:11){
      lines(c(1:nt), v1m[,jj])
    }
  }
```

(1) The number of data points (`nd`) and the number of simulations (the range of k, `nt`) are given.

(2) A matrix (`v1m`) for storing variances is prepared.
(3) γ (Eq. (3.49) (page 122)) is set to one of $\{0.8, 0.82, 0.84, \ldots, 1.0\}$ to carry out a simulation. `xx` contains the data ($\{x_i^{<k>}\}$). A vector (`v1`) for storing variances is prepared. The variance of $\{x_i^{<1>}\}$ when k is 1 is saved as `v1[1]`.
(4) Using Eq. (3.49), the values of $\{x_i^{<k>}\}$ when k is more than 1 are calculated and saved as `v1[ii + 1]`.
(5) The values in `v1` are organized in `v1m`.
(6) The values of `v1m` are drawn in a graph (Fig. 3.4 (page 123)).

R Program [3-3] End

R Program [3-4]

The value of γ (Eq. (3.49) (page 122)) for which the variance of a set remains constant is obtained.

```
simp12()
  function(){
  # (1)
    int1 <- function(x){
     x <- 1 / sqrt(2 * pi * 100) * 1/sqrt(2 * pi *
       36) * x * exp(-1/(2 * 100) * (x-50)^2) *
       exp(-1/(2 * 36) * (x - 65)^2)
    return(x)
    }
  # (2)
    int2 <- function(x){
     x <- 1 / sqrt(2 * pi * 100) * 1 /sqrt(2 * pi *
       36) * exp(-1/(2 * 100) * (x-50)^2) *
       exp(-1/(2 * 36) * (x - 65)^2)
    return(x)
    }
  # (3)
    an1 <- integrate(int1, lower = -Inf,
     upper = Inf)$value
    an2 <- integrate(int2, lower = -Inf,
     upper = Inf)$value
    an3 <- an1/an2
    print(an3)
  }
```

(1) The integrand of the numerator in Eq. (3.57) (page 125) is denoted by `int1`.
(2) The integrand of the denominator in Eq. (3.57) is denoted by `int2`.
(3) `integrate()` calculates the value of the numerator in Eq. (3.57). The result is saved as `an1`. `integrate()` also calculates the value of the denominator.

The result is saved as `an2`. The value of Eq. (3.57) is saved as `an3`. This value is displayed in the console window.

`simp12()` outputs:

```
61.02941
```

This value is identical to that described below Eq. (3.57) (page 125); the latter value is given by Mathematica 3.0.

R Program [3-4] End

3.4 Confidence Interval of Regression Coefficients in Simple Regression

When sampling is carried out many times from the same population under identical conditions, the values of $\hat{\mathbf{a}}$ given by Eq. (3.10) (page 111) are different each time. In this sense, the values of $\hat{\mathbf{a}}$ are random variables. Part of the appearance of the distribution of $\hat{\mathbf{a}}$ is observed using the variance-covariance matrix. The variance-covariance matrix of $\hat{\mathbf{a}}$, called $\mathbf{C}$, is defined as

$$\mathbf{C} = \begin{pmatrix} E\left[\left(\hat{a}_0 - E[\hat{a}_0]\right)\left(\hat{a}_0 - E[\hat{a}_0]\right]\right] & E\left[\left(\hat{a}_0 - E[\hat{a}_0]\right]\left(\hat{a}_1 - E[\hat{a}_1]\right)\right] \\ E\left[\left(\hat{a}_1 - E[\hat{a}_1]\right)\left(\hat{a}_0 - E[\hat{a}_0]\right)\right] & E\left[\left(\hat{a}_1 - E[\hat{a}_1]\right)\left(\hat{a}_1 - E[\hat{a}_1]\right)\right] \end{pmatrix}, \quad (3.58)$$

where $\hat{a}_0$ and $\hat{a}_1$ are random variables. $E[\cdot]$ denotes the expectation (expected value). For example, $E[\hat{a}_0]$ is the mean value of the values of $\hat{a}_0$ obtained by many samplings. However, sampling is usually carried out only once to obtain n data, and then $\hat{a}_0$ and $\hat{a}_1$ (which are nonrandom variables here) are calculated using the data. Distributions of $\hat{a}_0$ and $\hat{a}_1$, which would be the result of many samplings, cannot be produced. Accordingly, $\mathbf{C}$ should be estimated using $\hat{a}_0$ and $\hat{a}_1$ (nonrandom variables) yielded by one sampling of n data. If this is realized, we will know to what extent the resultant $\hat{a}_0$ and $\hat{a}_1$ are reliable. The estimate of $\mathbf{C}$ given by one sampling of n data points is written as

$$\mathbf{C} = \sigma^2(\mathbf{X}^t\mathbf{X})^{-1}, \quad (3.59)$$

where it is assumed that the following equation, in which the predictor variable is represented by x and the target variable is represented by y, holds in a population. Let ϵ be a random variable representing an error. It obeys a normal distribution in which the mean is 0 and the variance is σ^2.

$$y = a_0 + a_1 x + \epsilon. \tag{3.60}$$

Equation (3.59) is derived as follows:
First, Eqs. (4.96) (page 128) and (3.10) (page 111) lead to

$$\begin{aligned}
\mathbf{C} &= E[(\hat{\mathbf{a}} - E[\hat{\mathbf{a}}])(\hat{\mathbf{a}} - E[\hat{\mathbf{a}}])^t] \\
&= E[((\mathbf{X}^t\mathbf{X})^{-1}\mathbf{X}^t\mathbf{y} - E[(\mathbf{X}^t\mathbf{X})^{-1}\mathbf{X}^t\mathbf{y}])((\mathbf{X}^t\mathbf{X})^{-1}\mathbf{X}^t\mathbf{y} - E[(\mathbf{X}^t\mathbf{X})^{-1}\mathbf{X}^t\mathbf{y}])^t] \\
&= E[((\mathbf{X}^t\mathbf{X})^{-1}\mathbf{X}^t\mathbf{y} - (\mathbf{X}^t\mathbf{X})^{-1}\mathbf{X}^t E[\mathbf{y}])((\mathbf{X}^t\mathbf{X})^{-1}\mathbf{X}^t\mathbf{y} - (\mathbf{X}^t\mathbf{X})^{-1}\mathbf{X}^t E[\mathbf{y}])^t] \\
&= E[((\mathbf{X}^t\mathbf{X})^{-1}\mathbf{X}^t(\mathbf{X}\mathbf{a} + \boldsymbol{\epsilon}) - (\mathbf{X}^t\mathbf{X})^{-1}\mathbf{X}^t\mathbf{X}\mathbf{a})((\mathbf{X}^t\mathbf{X})^{-1}\mathbf{X}^t(\mathbf{X}\mathbf{a} + \boldsymbol{\epsilon}) \\
&\quad -(\mathbf{X}^t\mathbf{X})^{-1}\mathbf{X}^t\mathbf{X}\mathbf{a})^t] \\
&= E[((\mathbf{X}^t\mathbf{X})^{-1}\mathbf{X}^t\boldsymbol{\epsilon})((\mathbf{X}^t\mathbf{X})^{-1}\mathbf{X}^t\boldsymbol{\epsilon})^t].
\end{aligned} \tag{3.61}$$

We give the following definition.

$$\mathbf{U} = (\mathbf{X}^t\mathbf{X})^{-1}\mathbf{X}^t. \tag{3.62}$$

As a result, Eq. (3.61) becomes

$$\begin{aligned}
\mathbf{C} &= E[(\mathbf{U}\boldsymbol{\epsilon})(\mathbf{U}\boldsymbol{\epsilon})^t] \\
&= E[(\mathbf{U}\boldsymbol{\epsilon}\boldsymbol{\epsilon}^t\mathbf{U}^t].
\end{aligned} \tag{3.63}$$

The il element of the matrix defined in Eq. (3.63) is

$$\begin{aligned}
E\left[\sum_j \sum_k [\mathbf{U}]_{ij}[\boldsymbol{\epsilon}]_j[\boldsymbol{\epsilon}]_k[\mathbf{U}]_{lk}\right] &= \sigma^2 \sum_j [\mathbf{U}]_{ij}[\mathbf{U}]_{lj} \\
&= \sigma^2[\mathbf{U}\mathbf{U}^t]_{il}.
\end{aligned} \tag{3.64}$$

Equation (3.38) (page 115) is used to derive the equation on the first line. Using Eq. (3.64), Eq. (3.62) becomes

$$\begin{aligned}
\mathbf{C} &= \sigma^2\mathbf{U}\mathbf{U}^t \\
&= \sigma^2((\mathbf{X}^t\mathbf{X})^{-1}\mathbf{X}^t)((\mathbf{X}^t\mathbf{X})^{-1}\mathbf{X}^t)^t \\
&= \sigma^2(\mathbf{X}^t\mathbf{X})^{-1}\mathbf{X}^t\mathbf{X}((\mathbf{X}^t\mathbf{X})^{-1})^t \\
&= \sigma^2((\mathbf{X}^t\mathbf{X})^{-1})^t \\
&= \sigma^2((\mathbf{X}^t\mathbf{X})^t)^{-1} \\
&= \sigma^2(\mathbf{X}^t\mathbf{X})^{-1}.
\end{aligned} \tag{3.65}$$

Equation (1.39) (page 30) is used to obtain the equality between the fourth and fifth lines. Equation (3.65) yields Eq. (3.59) (page 128).

Using Eq. (3.16) (page 111), Eq. (3.59) (page 128) becomes

$$\mathbf{C} = \sigma^2 \begin{pmatrix} n & \sum_{i=1}^n x_i \\ \sum_{i=1}^n x_i & \sum_{i=1}^n x_i^2 \end{pmatrix}^{-1}. \tag{3.66}$$

Equation (1.30) (page 28) results in

$$\mathbf{C} = \frac{\sigma^2}{n\sum_{i=1}^n x_i^2 - \sum_{i=1}^n x_i \sum_{i=1}^n x_i} \begin{pmatrix} \sum_{i=1}^n x_i^2 & -\sum_{i=1}^n x_i \\ -\sum_{i=1}^n x_i & n \end{pmatrix}. \tag{3.67}$$

Next, S_{xx} is defined as

$$S_{xx} = \sum_{i=1}^n (x_i - \bar{x})^2, \tag{3.68}$$

where $\bar{x}$ is the average of $\{x_i\}$ $(1 \leq i \leq n)$ (Eq. (3.28) (page 113)). Hence, Eq. (3.68) is written as

$$\begin{aligned} S_{xx} &= \sum_{i=1}^n \Bigl(x_i - \frac{1}{n}\sum_{i=1}^n x_i\Bigr)^2 \\ &= \sum_{i=1}^n x_i^2 - \frac{2}{n}\sum_{i=1}^n x_i \sum_{i=1}^n x_i + \frac{2}{n}\Bigl(\sum_{i=1}^n x_i\Bigr)^2 \\ &= \sum_{i=1}^n x_i^2 - \frac{1}{n}\sum_{i=1}^n x_i \sum_{i=1}^n x_i. \end{aligned} \tag{3.69}$$

Using this equation, Eq. (3.67) (page 130) becomes

$$\mathbf{C} = \frac{\sigma^2}{nS_{xx}} \begin{pmatrix} \sum_{i=1}^n x_i^2 & -\sum_{i=1}^n x_i \\ -\sum_{i=1}^n x_i & n \end{pmatrix}. \tag{3.70}$$

Because σ^2 is unknown in this equation, it is replaced by the estimate of σ^2 (i.e., $\hat{\sigma}^2$) given by n data. $\hat{\sigma}^2$ is written as follows (Eq. (3.42) (page 116)).

$$\hat{\sigma}^2 = \frac{\sum_{i=1}^n (y_i - a_0 - a_1 x_i)^2}{n-2} = \frac{RSS}{n-2}, \tag{3.71}$$

where RSS is defined in Eq. (3.4). The value of $\hat{\sigma}^2$ given by Eq. (3.71) is termed an unbiased estimator of σ^2.

Substituting Eq. (3.71) into Eq. (3.59) (page 128) yields

$$\mathbf{C} = (\mathbf{X}^t\mathbf{X})^{-1}\hat{\sigma}^2$$

$$= \frac{(\mathbf{X}^t\mathbf{X})^{-1}\sum_{i=1}^{n}(y_i - a_0 - a_1 x_i)^2}{n-2}. \tag{3.72}$$

All the elements of $(\mathbf{X}^t\mathbf{X})^{-1}\mathbf{X}^t$ on the right-hand side of Eq. (3.10) (page 111) are nonrandom variables. All the elements of $\mathbf{y}$ are considered to be random variables that obey a normal distribution. Hence, both of the elements of $\hat{\mathbf{a}}$ (a_0 and a_1), which are written as the sum of random variables multiplied by constants (i.e., a linear combination of random variables), are random variables obeying normal distributions. As the variance of a_1 is $[\mathbf{C}]_{22}$, the result in the last chapter (Fig. 2.27 (page 61)) shows that the confidence interval of a_1 (a nonrandom variable, Eq. (3.69)) in the population is represented as

$$\hat{a}_1 - 1.96\sqrt{[\mathbf{C}]_{22}} < a_1 < \hat{a}_1 + 1.96\sqrt{[\mathbf{C}]_{22}}. \tag{3.73}$$

Using Eq. (3.70) (page 130), we obtain

$$\hat{a}_1 - 1.96\sqrt{\frac{\sigma^2}{S_{xx}}} < a_1 < \hat{a}_1 + 1.96\sqrt{\frac{\sigma^2}{S_{xx}}}. \tag{3.74}$$

$\hat{\sigma}^2$ given by Eq. (3.71) is usually employed as σ^2.

3.5 t-Test in Simple Regression

A t-test is used to carry out a hypothesis test to determine whether a_1 in a population is regarded as a specific value ($a_{1,0}$ here). The null hypothesis for this test is set as

$$H_0 : a_1 = a_{1,0}. \tag{3.75}$$

The alternative hypothesis is set as

$$H_1 : a_1 \neq a_{1,0}. \tag{3.76}$$

In particular, when $a_{1,0} = 0$ is employed, the null hypothesis becomes

$$H_0 : a_1 = 0. \tag{3.77}$$

The corresponding alternative hypothesis becomes

$$H_1 : a_1 \neq 0. \tag{3.78}$$

If this null hypothesis (Eq. (3.77)) is rejected, and hence the alternative hypothesis is adopted, it invalidates the assumption that the population is described using the

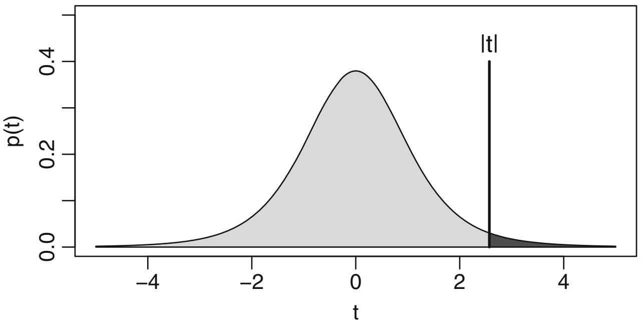


Fig. 3.5 The absolute value of the t-value (Eq. (3.81)) is calculated. The p-value is defined as twice the area (filled in *dark gray*) where the value of the horizontal axis is greater than $|t|$

regression equation $y = a_0$. This leads to the conclusion that the population is represented using the regression equation $y = a_0 + a_1 x$.

The t-value required to conduct a test using Eqs. (3.75) and (3.76) is defined as

$$t = \frac{\hat{a}_1 - a_{1,0}}{\sqrt{[\mathbf{C}]_{22}}} \sim t(\phi). \tag{3.79}$$

When the null hypothesis (Eq. (3.75)) holds, t obeys the t-distribution with $(n-2)$ degrees of freedom in the same manner as Eq. (2.30) (page 62). That is, we write the equation:

$$t \sim t(n-2). \tag{3.80}$$

As a simple regression involves two regression coefficients, the value of ϕ (degrees of freedom) in a simple regression is $(n-2)$. Hence, the t value is obtained as

$$\begin{aligned} t &= \frac{\hat{a}_1 - a_{1,0}}{\sqrt{[\mathbf{C}]_{22}}} \\ &= \frac{(\hat{a}_1 - a_{1,0})\sqrt{S_{xx}}}{\sqrt{\sigma^2}} \\ &\approx \frac{(\hat{a}_1 - a_{1,0})\sqrt{S_{xx}}}{\sqrt{\hat{\sigma}^2}}. \end{aligned} \tag{3.81}$$

The p-value is obtained by comparing this t (nonrandom variable) with the probability density function of the t-distribution with $(n-2)$ degrees of freedom. As shown in Fig. 3.5, because this is a two-sided test, the p-value is given by twice the area (filled in dark gray) for which the value of the horizontal axis is more than $|t|$. For example, we may take the stance that the null hypothesis is rejected if the p-value is less than 0.05. Then, if the t-value is

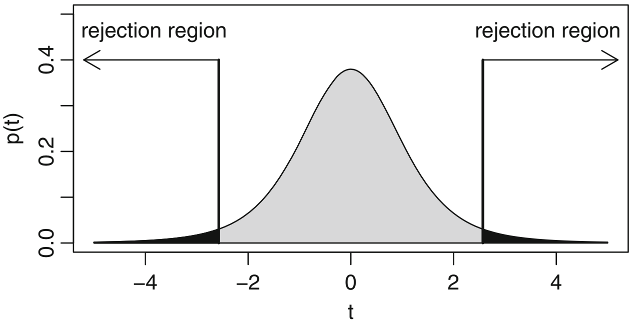


Fig. 3.6 Rejection region when null hypothesis is rejected by the p-value of less than 0.05. The sum of the *dark gray* areas at both ends is 0.05

placed in the rejection regions (critical regions) in Fig. 3.6, the null hypothesis is rejected. When the null hypothesis holds, the t-value has a 5 % probability of being located in the rejection regions. Hence, the sum of the dark gray areas at both ends of the probability density function (Fig. 3.6) is 0.05. In addition, if the confidence interval given by Eq. (3.73) (page 131) does not contain 0, then it is equivalent to finding that the p-value is less than 0.05.

The simulation data obtained using the following equation are generated to determine the distributions of the regression coefficients given by a simple regression:

$$y_i = -5 + 4x_i + \epsilon_i. \tag{3.82}$$

$\{\epsilon_i\}$ $(1 \leq i \leq n)$ are the realizations of $N(0, 3^2)$ (normal distribution; mean of 0 and variance of 3^2). The values of the predictor variable ($\{x_i\}$ $(1 \leq i \leq 4)$) are $\{1, 2, 3, 4\}$. Using Eq. (3.82), four simulation data points are produced. The simulation was repeated 10,000 times with a different initial value for the pseudo-random numbers. This yields the t-value distribution in Fig. 3.7 (left). The null hypothesis (H_0) and the alternative hypothesis (H_1) for this simulation data are as follows:

$$H_0 : a_1 = 4, \tag{3.83}$$

$$H_1 : a_1 \neq 4. \tag{3.84}$$

The t-values are calculated using

$$t = \frac{(\hat{a}_1 - 4)\sqrt{S_{xx}}}{\sqrt{\hat{\sigma}^2}}. \tag{3.85}$$

The probability density function of the t distribution with 2 $(= 4 - 2)$ degrees of freedom, assuming that the null hypothesis (Eq. (3.83)) holds, multiplied by a constant is also drawn in Fig. 3.7 (left). The two distributions are almost identical.

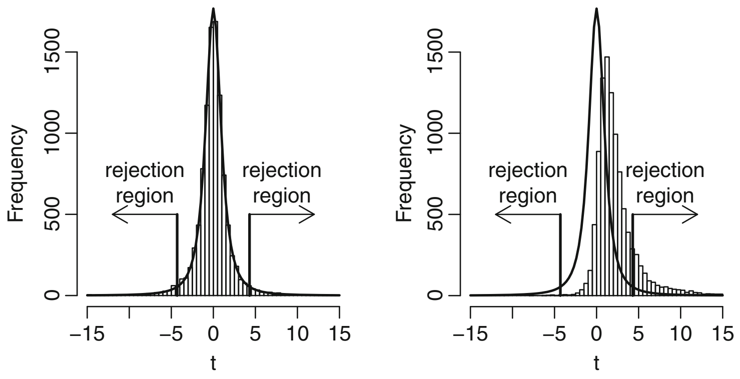


Fig. 3.7 Distribution of the t-value of the regression coefficient ($\hat{a}_1$) calculated by simple regression using simulation data given by Eq. (3.82), and the probability density function of the t distribution, assuming that the null hypothesis (Eq. (3.83)) holds, multiplied by a constant (rejection regions given by 5 % risk rate are also shown) (*left*). Distribution of the t-value of the regression coefficient ($\hat{a}_1$) calculated by simple regression using simulation data given by Eq. (3.86), and the probability density function of the t distribution, assuming that the null hypothesis (Eq. (3.83)) holds, multiplied by a constant (rejection regions given by 5 % risk rate are also shown) (*right*)

The rejection regions given by a 5 % risk rate are also illustrated. The number of simulation data points located in the left rejection region is 271, and that located in the right rejection region is 236. All 507 simulation data points are positioned in the rejection regions. Therefore, although the simulation data satisfy the null hypothesis (Eq. (3.83)), the null hypothesis is rejected for 5 % of the data. In other words, the type I error occurs with a probability of approximately 5 %.

However, if we use

$$y_i = -5 + 6x_i + \epsilon_i, \tag{3.86}$$

to generate the simulation data, instead of Eq. (3.82), the result is as shown in Fig. 3.7 (right). The t-value is calculated using Eq. (3.85). The curve representing the t distribution, assuming that the null hypothesis holds, is the same as that drawn in Fig. 3.7 (left). As the simulation data does not obey the null hypothesis, the resulting distribution of the t-value deviates considerably from the t distribution yielded using the null hypothesis. When a 5% risk rate for the null hypothesis is set, 16 simulation data points are located in the left rejection region and 1,424 simulation data points are located in the right one; all 1,440 data fall within the rejection regions. Therefore, the null hypothesis is not rejected by the remaining 8,560 simulation data points, although the null hypothesis is not satisfied. This means that the power of the test is about 15% and the probability of occurrence of the type II error is roughly 85%.

R Program [3 - 5]

The t-value and the p-value calculated using `lm()` are identical to those derived according to their definitions.

```
simp13()
  function()
  {
  # (1)
    set.seed(814)
    nd <- 20
    xx <- seq(from = 1, to = nd, by = 1)
    yy <- xx *0.2 -14 + rnorm(nd, mean = 0, sd = 3)
  # (2)
    data1 <- data.frame(x = xx, y = yy)
    lm1 <- lm(y~x, data = data1)
    suma1 <- summary(lm1)
    print(suma1)
  # (3)
    tt1 <- suma1$coef[2,3]
    print("tt1")
    print(tt1)
    pval1 <- suma1$coef[2,4]
    print("pval1")
    print(pval1)
  # (4)
    xxm <- matrix( c(rep(1, length=nd), xx), ncol = 2)
    yym <- matrix(yy, ncol = 1)
    ey <- aa[1] + aa[2]*xx
  # (5)
    sig2 <- sum( (yy - ey)^2 ) /(nd-2)
  # (6)
    tt2 <- aa[2] / sqrt(cc[2,2])
    print("tt2")
    print(tt2)
  # (7)
    pval2 <- (1 - pt(abs(tt2), df = nd-2)) * 2
    print("pval2")
    print(pval2)
  }
```

(1) `set.seed(814)` sets an initial value for the pseudo-random numbers. The number of data points (`nd`) is given. The simulation data of the predictor variable are stored in `xx`, and the target variable data are saved in `yy`.

(2) `xx` and `yy` are organized in the data frame named `data1`. In this data frame, `xx` is called `x` and `yy` is called `y`. Then, `lm()` carries out a simple regression.

The result is stored in `lm1`. `summary()` takes the summary from `lm1`. The result is named `suma1`, and `suma1` is displayed in the console window.

(3) The t-value (Eq. (3.81) (page 132)) for $\hat{a}_1$ is obtained as `suma1$coef[2,3]` (the $(2, 3)$ element of a matrix named `coef`, which is a component of `suma1`) from `suma1`. This is named `tt1` and is displayed in the console window. The p-value (Eq. (3.81)) for $\hat{a}_1$ is taken as `suma1$coef[2,4]` (the $(2, 4)$ element of the matrix `coef`, which is a component of `suma1`) from `suma1`. This is named `pval1` and is displayed in the console window.

(4) According to the definition described in the R program `simp1()`, a simple regression is conducted. The resultant regression coefficients are called `aa`. Estimates corresponding to `xx` are calculated using `aa`. The estimates are named `ey`.

(5) $\hat{\sigma}^2$ calculated using Eq. (3.71) (page 130) is named `sig2`. The matrix of $\mathbf{C}$ derived using Eq. (3.72) (page 130) is called `cc`.

(6) The t-value calculated using Eq. (3.81) is named `tt2` and is displayed in the console window.

(7) `pt()` calculates the p-value using `tt2` given in (6). The resultant value is named `pval2` and is displayed in the console window.

`simp13()` outputs:

```
Call:
lm(formula = y ~ x, data = data1)
Residuals:
Min        1Q        Median     3Q       Max
-5.1932    -1.5410   0.3925     1.6462   5.5685

Coefficients:
Estimate       Std.        Error      t value   Pr(>|t|)
(Intercept) -13.5994    1.3822     -9.839    1.15e-08 ***
x              0.1874      0.1154     1.625     0.122
---
Signif. codes: 0'***'0.001'**'0.01'*'0.05'.'0.1' '1

Residual standard error:2.975 on 18 degrees of freedom
Multiple R-squared:0.1279, Adjusted R-squared: 0.07941
F-statistic: 2.639 on 1 and 18 DF,  p-value: 0.1217

"tt1"
1.624502
"pval1"
0.1216501
"tt2"
1.624502
"pval2"
0.1216501
```

The *t*-value and the *p*-value yielded by `lm()` are identical to those according to their definitions.

R Program [3 - 5] End

R Program [3 - 6]

The regression coefficients of the simple regression obey the *t*-distribution.

```
simp22e()
  function ()
  {
  # (1)
    set.seed(814)
    nt <- 10000
    tt <- NULL
    nd <- 4
  # (2)
    for (jj in 1:nt){
      xx <- seq(from=1, to=nd, by=1)
      yy <-  -5 + xx * 6 + rnorm(nd, mean = 0, sd = 3)
  # (3)
      xxm <- matrix(c(rep(1, length = nd), xx),
       ncol = 2)
      yym <- matrix(yy, ncol=1)
      ey <- aa[1] + aa[2]*xx
      sig2 <- sum( (yy - ey)^2 ) /(nd-2)
  #    tt[jj] <- aa[2]/sqrt(cc[2,2])
      tt[jj] <- (aa[2] - 4)/sqrt(cc[2,2])
    }
  # (4)
    qt1 <- qt(0.025, nd-2)
    nr1 <- length(tt[qt1 > tt])
    qt2 <- qt(0.975, nd-2)
    nr2 <- length(tt[qt2 < tt])
    print("qt1")
    print(qt1)
    print("qt2")
    print(qt2)
    print("nr1")
    print(nr1)
    print("nr2")
    print(nr2)
  # (5)
    par(mfrow = c(1, 1), mai = c(1, 1, 0.2, 0.2),
     oma = c(1, 1, 1, 1))
    tt <- tt[tt <=15 & tt >= -15]
    br1 <-  pretty(c(-15, 15), n=100)
    bw1 <- br1[2] - br1[1]
    tth <- floor(tt/bw1) * bw1 + 0.01 * bw1
    hist1 <- hist(tth, breaks=br1, main="", xlab="t",
     xlim = c(-15, 15), ylim = c(0, 1700))
  # (6)
    curve(dt(x, df = nd - 2) * bw1 * nt, min(br1),
     max(br1), xlab = "x", ylab = "p(x)", lwd = 2,
     xlim = c(min(br1), max(br1)),
     ylim = c(0, max(hist1$counts)), add = T)
    arrows(qt1, 500, -12, 500, code = 2, lty = 1,
     length = 0.15)
    lines(c(qt1, qt1), c(0, 500), lwd = 3)
    text(qt1 - 4.8, 760, "rejection")
    text(qt1 - 4.8, 650, "region")
    arrows(qt2, 500, 12, 500, code = 2, lty = 1,
     length = 0.15)
    lines(c(qt2, qt2), c(0, 500), lwd = 3)
    text(qt2 + 5.1, 760, "rejection")
    text(qt2 + 5.1, 650, "region")
  }
```

(1) The number of data points (`nd`) and the number of simulations (`nt`) are given.
(2) Simulation data are generated using Eq. (3.86) (page 134). Simulation data of the predictor variable are stored in `xx`. Target variable data are stored in `yy`.
(3) A simple regression is carried out using the method of `simp1()`. The resulting regression coefficients are named `aa`. `solve()` calculates the inverse matrix. Estimates corresponding to `xx` are derived using `aa`. The resultant estimates are called `ey`. $\hat{\sigma}^2$ is estimated using Eq. (3.71) (page 130). The result is named `sig2`. The matrix **C**, obtained using Eq. (3.72) (page 130), is called `cc`. The t-value calculated using Eq. (3.81) (page 132) is named `tt[jj]`.
(4) `qt()` yields the rejection region. The result is stored in `qt1`. The number of data falling in the rejection region is saved in `nr`. `qt1` and `nr` are displayed in the console window.
(5) Values that are greater than or equal to -8 or less than or equal to 12 are retrieved from `tt` and used to form an updated `tt`. `hist()` draws a histogram of `tt`.
(6) The probability density function of the t-distribution with 2 degrees of freedom (`nd-2`) (2 in this example) multiplied by a constant is superimposed on the histogram drawn in (5). Figure 3.7 (right) (page 134) is obtained.

`simp22e()` also outputs:

```
"qt1"
-4.302653
"qt2"
```

```
4.302653
"nr1"
16
"nr2"
1424
```

R Program [3 - 6] End

3.6 F-Test on Simple Regression

When a statistical test employs H_0 (Eq. (3.75) (page 131)) as a null hypothesis and H_1 (Eq. (3.76) (page 131)) as an alternative hypothesis, the use of Eq. (3.81) (page 132) is based on the characteristic that a_1 obeys the t-distribution with certain degrees of freedom. Another method, in which the residual sum of squares given by the regression equation of $y = a_0$ and that given by the regression equation of $y = a_0 + a_1 x$ are compared, can also be used. If the RSS given by $y = a_0 + a_1 x$ is considerably smaller, $y = a_0 + a_1 x$ is determined to be an appropriate regression equation. Hence, H_0 is rejected and H_1 is adopted. This method is called the F-test. The RSS given by $y = a_0 + a_1 x$ is obtained from Eq. (3.4) (page 109). The RSS given by $y = a_0$ is represented as follows.

$$RSS(a_0) = \sum_{i=1}^{n} (y_i - a_0)^2. \tag{3.87}$$

Furthermore, RSS defined in Eq. (3.4) is denoted by $RSS(a_0, a_1)$. The F-value for this setting is

$$F(H_0, H_1) = \frac{\dfrac{RSS(a_0) - RSS(a_0, a_1)}{1}}{\dfrac{RSS(a_0, a_1)}{n-2}}. \tag{3.88}$$

This (nonrandom variable) F is called the F-value. This value cannot be negative. The division by 1 in the numerator indicates that the difference between the number of regression coefficients of $y = a_0 + a_1 x$ and that of $y = a_0$ is 1. The division by $(n - 2)$ in the denominator comes from the number of data (n) minus the number of regression coefficients ($= 2$) of $y = a_0 + a_1 x$. The F-value derived from one sampling is accumulated by many repetitions of the sampling. Such F-values constitute a distribution. This distribution is the F-distribution with the first degree of freedom (or the numerator degree of freedom, degree of freedom in the numerator) ($\phi_1 = 1$) and the second degree of freedom (or the denominator degree of freedom, degree of freedom in the denominator) ($\phi_2 = n - 2$) (Eq. (2.66)

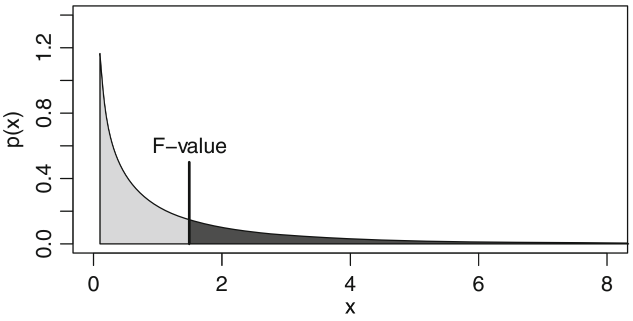


Fig. 3.8 Probability density function of the F-distribution with the first degree of freedom 1 and the second degree of freedom 10. The space of the *dark gray* area is the p-value

(page 90)) when the population obeys $y = a_0$, that is, the null hypothesis (Eq. (3.75) (page 131)) holds. This situation is represented by

$$F(H_0, H_1) \sim F_{1,n-2}. \tag{3.89}$$

Figure 3.8 illustrates an example of the probability density function obeyed by $F_{1,n-2}$. Then, the following value is calculated.

$$p = \int_{F(H_0,H_1)}^{\infty} den(1, n-2)dx, \tag{3.90}$$

where $den(1, n-2)$ is the probability density function of the F-distribution with the first degree of freedom 1 and the second degree of freedom $(n-2)$. The value of p corresponding to the F-value given by Eq. (3.88) is the space of the dark gray area in Fig. 3.8. For example, if the value of p is less than 0.05, the data rarely seems to be obtained under the assumption that the null hypothesis holds. Then, the null hypothesis is rejected. Therefore, the area in which the value of p is less than, for example, 0.05 is set as the rejection region (Fig. 3.9). This is an example of a one-sided test. When the F-value is a positive number close to 0, $(RSS(a_0) - RSS(a_0, a_1))$ given by Eq. (3.88) takes a positive value close to 0. That is, the RSS given by $y = a_0 + a_1x$ is close to that given by $y = a_0$. In such an event, we have little grounds for adopting $y = a_0 + a_1x$. Therefore, when the F-value is located in the region where it takes a positive value close to 0, H_0 (Eq. (3.75) (page 131) cannot be rejected. Hence, a one-sided test is carried out. This test is considered to be a type of analysis of variance (ANOVA). Furthermore, a test using the F-value (Eq. (3.88)) yields exactly the same result as when a test using the t-value (Eqs. (3.73) (page 131) and (3.81) (page 132)) is conducted. That is, the p-value given by the t-value is identical to that given by the F-value. This can be rigorously proved.

Let us conduct a simulation to illustrate that the test based on Eq. (3.90) is rational. This test uses the F-values given by Eq. (3.88). First, 20 data are generated

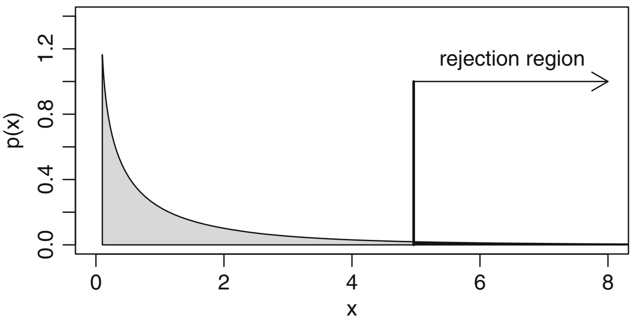


Fig. 3.9 Probability density function of the F-distribution with the first degree of freedom 1 and the second degree of freedom 10. The region indicated by the *arrow* is the rejection region when the risk rate is set at 5 %.

using the following equation.

$$y_i = 3 + \epsilon_i, \tag{3.91}$$

where the $\{\epsilon_i\}$ $(1 \leq i \leq n)$ are realizations of $N(0, 0.2^2)$ (normal distribution; mean of 0 and variance of 0.2^2). Then, the following values are calculated.

$$RSS(a_0) = \sum_{i=1}^{n}(y_i - a_0)^2, \tag{3.92}$$

$$RSS(a_0, a_1) = \sum_{i=1}^{n}(y_i - a_0 - a_1 x_i)^2. \tag{3.93}$$

We generate $2,000$ sets of simulation data from a different initial value for the pseudo-random numbers. The F-values (Eq. (3.88)) of the respective data are calculated. The histogram of the distribution of these values is shown in Fig. 3.10 (left). The probability density function of the F-distribution with the first degree of freedom 1 and the second degree of freedom $(n - 2)$ (Eq. (3.90)) is superimposed. The frequencies are normalized so that the sums of the respective frequencies are equal. The F-values are distributed almost according to the probability density function of the F-distribution with the first degree of freedom 1 and the second degree of freedom $(n - 2)$. The rejection region for a 5 % risk rate is superimposed The number of simulation data that fall within the rejection region is 123. This is roughly 5 % of the total number of data. Moreover, Fig. 3.10 (right) shows a scatter graph illustrating the relationship between the numerator and the denominator in the right-hand side of Eq. (3.88). The relationship between the numerator and the denominator in the right-hand side of Eq. (3.88) is also drawn using the first 50 of the $2,000$ data sets. This graph shows the relationship between the two sets of 50 values. The first set consists of 50 realizations of the χ^2-distribution with 1 degree

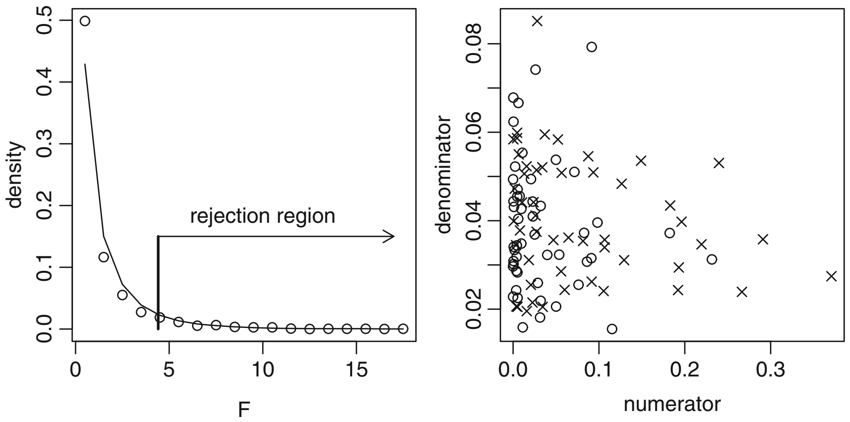


Fig. 3.10 Distribution of the F-value (Eq. (3.88)) given by 2,000 sets of simulation data (*open circle*) and the probability density function of the F-distribution with the first degree of freedom 1 and the second degree of freedom $(n-2)$ (Eq. (3.90)) (*solid line*). The rejection region for a 5 % risk rate is superimposed (*left*). The relationship (*open circle*) between the numerator and the denominator in the right-hand side of Eq. (3.88). This relationship is obtained from the first 50 of the 2,000 data sets. The relationship (*crosses*) between two independent realizations of the χ^2-distribution is superimposed (*right*)

of freedom multiplied by the variance of $\{\epsilon_i\}$ $(= 0.2^2)$. The second set consists of 50 realizations of the chi-squared distribution with $(n-2)$ degrees of freedom multiplied by the variance of $\{\epsilon_i\}$ $(= 0.2^2)$ and divided by $(n-2)$. These two sets of realizations of the chi-squared distribution are independent. The distribution given by the relationship between the numerator and the denominator in the right-hand side of Eq. (3.88) looks close to that given by the two chi-squared distributions. That is, when the null hypothesis (Eq. (3.75) (page 131)) holds, the F-value (Eq. (3.88)) forms a random variable that obeys the probability density function defined by Eq. (2.66) (page 90). This example shows the appropriateness of the test using the F-value.

Another statistic may be used to determine whether a constant or simple regression equation is desirable. For example, the following statistic can be conceived.

$$F^{+}(H_0, H_1) = \frac{\dfrac{RSS(a_0)}{n-1}}{\dfrac{RSS(a_0, a_1)}{n-2}}. \tag{3.94}$$

When the null hypothesis defined in Eq. (3.75) (page 131) holds, the numerator is the realization of the χ^2-distribution with $(n-1)$ degrees of freedom multiplied by the variance of $\{\epsilon_i\}$ and divided by $(n-1)$. Moreover, the denominator is the realization of the chi-squared distribution with $(n-2)$ degrees of freedom multiplied by the variance of $\{\epsilon_i\}$ and divided by $(n-2)$. Therefore, when the

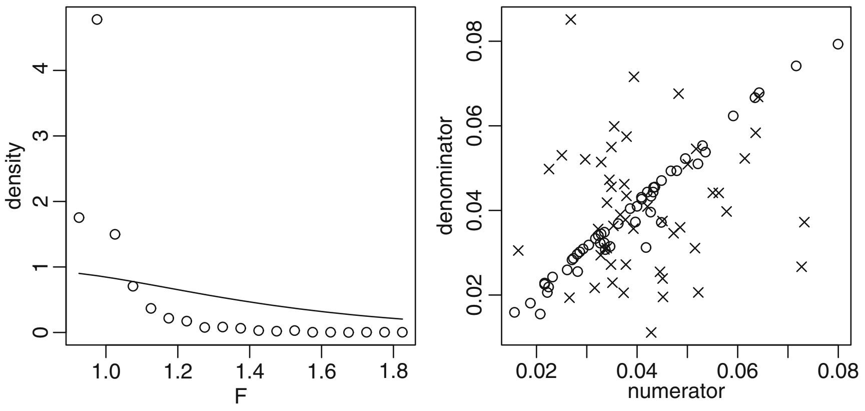


Fig. 3.11 Distribution of $F^+(H_0, H_1)$ (Eq. (3.94)) (*open circle*) yielded by the 2,000 sets of simulation data and the probability density function of the F-distribution with the first degree of freedom $(n-1)$ and the second degree of freedom $(n-2)$ (Eq. (3.90)) (*solid line*) (*left*). The relationship (*open circle*) between the numerator and the denominator in the right-hand side of Eq. (3.94). This relationship is obtained from the first 50 of the 2,000 data sets. The relationship (*crosses*) between two independent realizations of the χ^2-distribution is superimposed (*right*)

data satisfies the null hypothesis defined by Eq. (3.77), $F^+(H_0, H_1)$ may obey the F-distribution with the first degree of freedom $(n-1)$ and the second degree of freedom $(n-2)$. However, a simulation similar to Fig. 3.10 but using $F^+(H_0, H_1)$ results in Fig. 3.11. The distribution of $F^+(H_0, H_1)$ (Eq. (3.94)) differs greatly from that of the F-distribution with the first degree of freedom $(n-1)$ and the second degree of freedom $(n-2)$. In addition, the relationship between the numerator and the denominator in Eq. (3.94) varies substantially from that given by the chi-squared distribution. This implies that, as the numerator and the denominator in the right-hand side of Eq. (3.94) are highly dependent, the data does not form an F-distribution.

Next, let us estimate the rate of rejection of the null hypothesis on the basis of the null hypothesis (Eq. (3.77) (page 131)) and the alternative hypothesis (Eq. (3.78) (page 131)) when the simulation data ($\{y_i\}$ $(1 \leq i \leq n)$) are given by the following equation.

$$y_i = 3 + \gamma x_i + \epsilon_i, \tag{3.95}$$

where the number of data (n) is 20. The $\{\epsilon_i\}$ are realizations of $N(0, 0.2^2)$ (normal distribution; mean of 0 and variance of 0.2^2). γ is one of $\{0.1, 0.15\}$ and $\{x_i\}$ $(1 \leq i \leq n)$ is set as $\{0.1, 0.2, 0.3, \ldots, 2\}$. $F(H_0, H_1)$ (Eq. (3.88)) given by the simulation data with $\gamma = 0.1$ is calculated 2,000 times by altering the initial value of the pseudo-random numbers. Figure 3.12 (left) shows the distribution of the resulting F-values ($F(H_0, H_1)$). The probability density function of an F-distribution with the first degree of freedom (ϕ_1) 1 and the second degree of

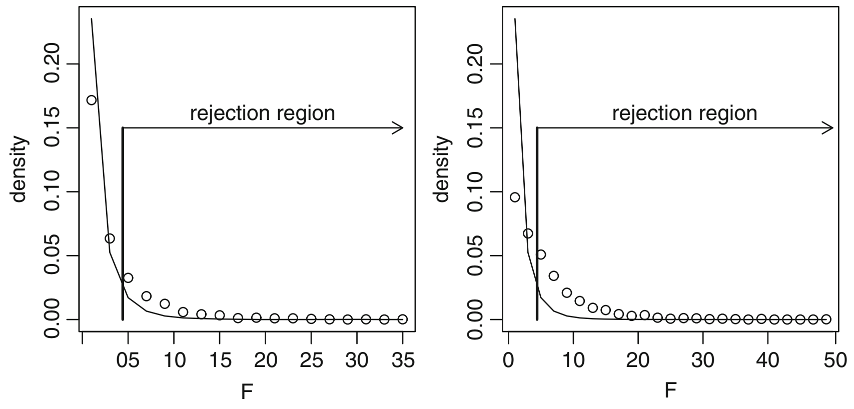


Fig. 3.12 Distribution of the F-values (Eq. (3.88)) given by 2,000 simulation data generated by Eq. (3.95) with $\gamma = 0.1$ (*open circle*). Probability density function of the F-distribution with the first degree of freedom 1 and the second degree of freedom $(n-2)$ (Eq. (3.90)) is superimposed (*solid line*) (*left*). Results given by setting $\gamma = 0.15$ (*right*)

freedom (ϕ_2) $(n-2)$ (Fig. 3.8 (page 140)) is superimposed. In common with Fig. 3.10 (left) (page 142), the distribution of the F-values is standardized to allow a comparison with the probability density function of the F-distribution. The arrow indicates the region where the value of p (Eq. (3.90)) is smaller than 0.05 (the rejection region) in the probability density function of the F-distribution. The number of simulation data that fall within the rejection region is 458. In principle, the null hypothesis should be rejected for each of the data sets because all of the 2,000 sets of the data do not satisfy the null hypothesis. However, the null hypothesis is rejected for 458 data sets only. This indicates that the power of test is 22.9 %. That is, the probability of occurrence of the type II error is 77.1 %.

When $\gamma = 0.15$, we have Fig. 3.12 (right). The number of simulation data that fall within the rejection region is 885. In this case, the power of test is 44.25 % and the probability of occurrence of the type II error is 55.75 %.

R Program [3-7]

The result of `anova()`, which carries out the F-test to select between $y = a_0 + a_1 x$ and $y = a_0$, is identical to that of the selection according to the definition. It is also identical to the result of the t-test.

```
simp31()
  function()
  {
  # (1)
    set.seed(815)
    nd <- 20
    xx <- seq(from = 1, to = nd, by = 1)
    yy <- xx * 0.2 - 14 + rnorm(nd, mean = 0, sd = 3)
  # (2)
    data1 <- data.frame(x = xx, y = yy)
    lm0 <- lm(y~1, data = data1)
    lm1 <- lm(y~x, data = data1)
  # (3)
    anova1 <- anova(lm0, lm1)
    print(anova1)
  # (4)
    ff <- (sum(lm0$residuals^2) - sum(lm1$residuals^2))/
    ((sum(lm1$residuals^2))/(nd-2))
    pval1 <- 1 - pf(ff, df1 = 1, df2 = nd-2)
    print("pval1")
    print(pval1)
  # (5)
    pval2 <- summary(lm1)$coef[2,4]
    print("pval2")
    print(pval2)
  }
```

(1) The number of data (`nd`) is given. Simulation data (`xx`, `yy`) are generated.
(2) `lm()` carries out regression using the regression equation of $y = a_0$. The result is saved as `lm0`. `lm()` carries out regression using the equation $y = a_0 + a_1 x$. The result is saved as `lm1`.
(3) `anova()` conducts an analysis of variance to compare `lm0` and `lm1`. The result is saved as `anova1`. `anova1` is output.
(4) The F-values are calculated using Eq. (3.88) (page 139). The result is saved as `ff`. `pf()` calculates the integral of the probability density function from $-\infty$ to `ff`. This probability density function is given by an F-distribution with the first degree of freedom `1` and the second degree of freedom `nd-2`. As this is a one-sided test, the p-value is obtained by subtracting this value from 1. The resulting p-value is saved as `pval1` and displayed in the console window.
(5) The p-value of the t-test on a_1 is extracted from `lm1` and is saved as `pval2`. This value is displayed in the console window.

The results of `simp31()` are as follows.

```
Analysis of Variance Table
Model 1: y ~ 1
Model 2: y ~ x
   Res.Df     RSS   Df   Sum of Sq      F      Pr(>F)
1      19   212.723
2      18   163.691  1    49.032     5.3917  0.03216 *
---
Signif. codes:0'***'0.001'**'0.01'*'0.05'.'0.1' '1
"pval1"
0.03216104
```

```
  "pval2"
  0.03216104
```

The value `0.03216` described below `Pr(>F)` is the *p*-value given by `anova(lm0, lm1)`. The value described below `pval1` is the *p*-value resulting from the F-value, which is calculated according to its definition. The value below `pval2` is the *p*-value given by `lm1`. These values are identical.

R Program [3-7] End

R Program [3-8]

When the null hypothesis (Eq. (3.77) (page 131)) holds, the values of $F(H_0, H_1)$ (Eq. (3.88) (page 139)) forms a distribution close to the F-distribution.

```
simp151e()
  function()
  {
  # (1)
    nd <- 20
    nt <- 2000
    xx <-seq(from = 0.1, to = 2, length = nd)
    sd1 <- 0.2
  # (2)
    ffv <- NULL
    v1 <- NULL
    v2 <- NULL
  # (3)
    for (kk in 1:nt){
      set.seed(348 + kk * 5)
      yy <- 3 + rnorm(nd, mean = 0, sd = sd1)
      rss1 <- sum((yy - mean(yy))^2)
      data1 <- data.frame(x = xx, y = yy)
      lm1 <- lm(y~x, data = data1)
      rss2 <- sum(lm1$residual^2)
      v1[kk] <- rss1 - rss2
      v2[kk] <- rss2/(nd-2)
      ffv[kk] <- v1[kk]/v2[kk]
    }
  # (4)
    br1 <- pretty(range(ffv), n=20)
    bw1 <- br1[2] - br1[1]
    ffh <- floor(ffv / bw1) * bw1 + 0.1 * bw1
    his1 <- hist(ffh, breaks = br1, right = F, plot = F)
    counts1 <- his1$counts
    br2 <- br1 + bw1 * 0.5
    br2 <- br2[1:(length(br2) - 1)]
  # (5)
```

```
    fdis <- NULL
    for (kk in 1:length(br2)){
      fdis[kk] <- df(br2[kk], 1, nd - 2)
    }
  # (6)
    qq1 <- qf(0.95, 1, nd - 2)
    nr <- length(ffv[ffv > qq1])
    print(nr)
  # (7)
    chi1 <- rchisq(50, 1) * sd1^2
    chi2 <- rchisq(50, nd - 2) * sd1^2/(nd - 2)
  # (8)
   par(mfrow = c(1,2), mai = c(2, 1, 1.5, 0.1),
    omi = c(0, 0, 0, 0))
    cden1 <- counts1 / nt * sum(fdis)
    plot(br2, cden1, type = "n", ylim = c(0,
     max(cden1, fdis)), xlab = "F",
     ylab = "density")
    lines(br2, fdis)
    points(br2, cden1)
    lines(c(qq1, qq1), c(0, 0.15), lwd=2)
    arrows(qq1, 0.15, 17, 0.15, angle = 30, code = 2,
     length = 0.1)
    text(10, 0.18, "rejection region")
  # (9)
    v1 <- v1[1:50]
    v2 <- v2[1:50]
    plot(v1, v2, xlim=range(v1, chi1),ylim = range(v2,
     chi2), xlab = "numerator", ylab = "denominator")
    points(chi1, chi2, pch = 4)
  }
```

(1) The number of data (`nd`), the number simulations (`nt`), the values of the predictor (`xx`), and the standard deviation of the errors contained in the data (`sd1`) are given.
(2) The vectors `ffv`, which stores values of $F(H_0, H_1)$, `v1`, which stores values of the numerator of $F(H_0, H_1)$, and `v2`, which stores values of the denominator of $F(H_0, H_1)$, are prepared.
(3) Using Eq. (3.91) (page 141), the values of the target variable (`yy`) are calculated. The residual sum of squares yielded by regression to a constant is saved as `rss1`. The residual sum of squares yielded by simple regression is saved as `rss2`. The numerator of the right-hand side of Eq. (3.88) (page 139) is saved as `v1[kk]`. The denominator of the fraction is saved as `v2[kk]`. The value of $F(H_0, H_1)$ (Eq. (3.88)) is saved as `ffv[kk]`.
(4) The break points of the bins for drawing a histogram of `ffv[kk]` are stored in `br1`. `hist()` derives the frequencies of `ffv[kk]` using `br1`. The result

is saved as `counts1`. The locations of the central points of the respective bins are stored in `br2`.

(5) The probability density function of an F-distribution with the first degree of freedom 1 and the second degree of freedom $(n-2)$ is obtained using `df()`. The result is saved as `fdis`.

(6) `qf()` yields the rejection region of the F-distribution. The number of simulation data that fall within the rejection region is saved as `nr`. `nr` is displayed in the console window.

(7) The equivalent value of the numerator of the right-hand side of Eq. (3.88) (page 139) is calculated using realizations of the χ^2-distribution. These realizations are given by `rchisq()`. The result is saved as `chi1`. The equivalent value of the denominator is calculated. The result is saved as `chi2`.

(8) `counts1` is normalized for comparison with `fdis`. The result is saved as `cden1`. `cden1` and `fdis` are drawn in a graph. When such data are graphed, the distribution of two random variables and the distributions of one random variable are drawn together to illustrate the features of the distribution more clearly (page 89 in [5]).

(9) The first 50 data are extracted from each of `v1` and `v2`. The relationship between the two data is drawn as a scatter graph. To show the relationship between `chi1` and `chi2`, the points are superimposed. Figure 3.10 (page 142) is drawn.

`simp151e()` also outputs:

```
123
```

This is the number of data located in the rejection region.

R Program [3-8] End

3.7 Selection Between Constant and Nonconstant Regression Equations

A method using the F-test is widely applied to choose between $y = a_0 + a_1 x$ and $y = a_0$ in practical data analysis. However, when we process actual data, we rarely know whether either of $y = a_0 + a_1 x$ or $y = a_0$ is the correct regression equation. The relationship between x and y is possibly more complex. Therefore, when the hypothesis test using the F-value given by Eq. (3.88) determines that the null hypothesis "$y = a_0$ is right" is rejected, and hence "$y = a_0 + a_1 x$" is adopted, this conclusion may lead to apparently inappropriate results.

For example, the data in Fig. 3.13 show a markedly upward trend accompanied by periodic variation and random errors. However, when the F-test with a 5 % risk rate is conducted, the null hypothesis "$y = a_0$ is right" is not rejected. This seems counterintuitive to the behavior of the data. This is because the two possibilities of

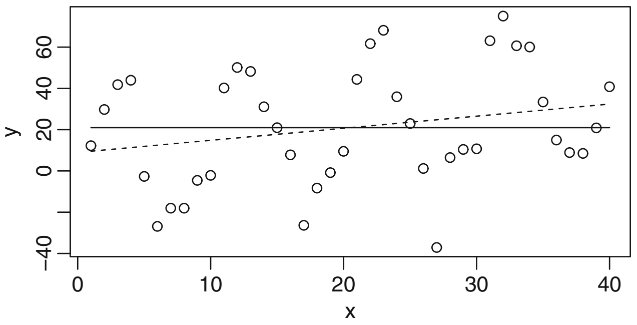


Fig. 3.13 Polynomial equations fitted to 40 data (*open circle*). The *solid line* shows a zero degree polynomial ($y = a_0$). The *dashed line* shows a one degree polynomial ($y = a_0 + a_1 x$)

$y = a_0 + a_1 x$ and $y = a_0$ are inadequate for this data. Therefore, when the test to choose between $y = a_0 + a_1 x$ and $y = a_0$ is used to determine whether an upward trend is present or not, an obvious upward trend can be overlooked. If an analyst determines the behavior of the data visually, as well as by conducting the F-test, this type of oversight is generally avoided. However, when large amounts of data are processed automatically, such a problem may be unavoidable.

R Program [3 - 9]

The test for choosing between $y = a_0 + a_1 x$ and $y = a_0$ is not always useful for determining whether there is an upward or downward trend.

```
simp41()
  function()
  {
  # (1)
    set.seed(815)
    nd <- 40
    xx <- seq(from = 1, to = nd, by = 1)
    yy <- xx + 40 * sin(pi * xx * 0.2) + 2 +
     rnorm(nd, mean = 0, sd = 10)
  # (2)
    data1 <- data.frame(x = xx, y = yy)
    lm0 <- lm(y~1, data = data1)
    lm1 <- lm(y~x, data = data1)
  # (3)
    anova1 <- anova(lm0, lm1)
    print("----- anova(lm0, lm1) -----")
    print(anova1)
  # (4)
    par(mfrow = c(1,1), mai = c(2, 1, 1.5, 0.1),
     omi = c(0, 0, 0, 0))
    plot(xx, yy, xlab = "x", ylab = "y")
    lines(xx, lm0$fitted, lty = 1)
    lines(xx, lm1$fitted, lty = 2)
  }
```

(1) The values of the predictor variable of the simulation data (`xx`) and those of the target variable (`yy`) are given. The number of data (`nd`) is given.
(2) The data are organized in the data frame of `data1`. `lm()` fits the data to the zero degree polynomial (constant). The result is saved as `lm0`. `lm()` fits the data to a one degree polynomial. The result is saved as `lm1`.
(3) `anova()` carries out the F-test (analysis of variance). The result is output.
(4) The estimates given by zero degree polynomial regression and those given by one degree polynomial regression are drawn in a graph (Fig. 3.13 (page 149)).

`simp41()` also outputs:

```
"----- anova(lm0, lm1) -----"
Analysis of Variance Table
Model 1: y ~ 1
Model 2: y ~ x
  Res.Df   RSS Df Sum of Sq      F Pr(>F)
1     39 31677
2     38 29849  1      1827 2.3264 0.1355
```

R Program [3-9] End

3.8 Prediction Error of Simple Regression

Statistical tests are not the only way to assess the appropriateness of a regression equation. The standpoint of the predictions is also important. From the viewpoint of prediction, when we use data in the future on the grounds of a regression equation, that which results in a small value of the prediction error is considered to be highly appropriate. However, as data in the future are not at hand, the prediction error given by possible data in the future is estimated using only the data at hand. One method for overcoming this problem is Cross-Validation (CV). When the prediction error of the regression equation $\hat{m}(x)$ is estimated using cross-validation, the result is denoted by $CV[\hat{m}(x)]$. $CV[\hat{m}(x)]$ is defined as

$$CV[\hat{m}(x)] = \frac{\sum_{k=1}^{n}(y_k - \hat{m}^{-k}(x_k))^2}{n}, \tag{3.96}$$

where n is the number of data. $\hat{m}^{-k}(x_k)$ is an estimate for the predictor of x_k when the regression equation is constructed by eliminating the k-th data point. Eq. (3.96) is not only useful for simple regression and linear regression, but also for other regression equations and models.

The n regression equations ($\hat{m}^{-k}(x_k)$ $(1 \leq k \leq n)$) are produced to calculate the prediction error using Eq. (3.96). When the number of data is large, the computational cost is reasonably large. However, when we use linear regression, which is defined as regression with the use of the hat matrix (Eq. (3.14) (page 111)) for estimation, the prediction error given by the cross-validation is obtained with a small amount of computation. The equation below is used for this purpose.

$$CV[\hat{m}(x)] = \sum_{i=1}^{n} \frac{(y_i - \hat{m}(x_i))^2}{n \cdot (1 - [\mathbf{H}]_{ii})^2}. \tag{3.97}$$

As the estimates are obtained using a hat matrix, Eq. (3.97) is derived. It should be noted that Eq. (3.97) is not an approximation of Eq. (3.96). The values given by the two equations are exactly the same (page 117 in [4]).

It was realized that the prediction error given by this cross-validation can be unsuitable, and so a Generalized Cross-Validation (GCV) was developed to alleviate this problem. This is defined as

$$GCV = \frac{\sum_{i=1}^{n}(y_i - \hat{m}(x_i))^2}{n \cdot \left(1 - \frac{\sum_{i=1}^{n}[\mathbf{H}]_{ii}}{n}\right)^2}. \tag{3.98}$$

When a simple regression equation, multiple regression equation, or polynomial equation is fitted by the least squares method, the following equation holds (page 43 in [4]).

$$\sum_{i=1}^{n}[\mathbf{H}]_{ii} = q + 1, \tag{3.99}$$

where q is the number of regression coefficients, omitting a constant term. We have $q = 1$ for simple regression.

R Program [3 - 10]

The value of CV given by Eq. (3.96) is identical to that given by Eq. (3.97). Although the value of GCV given by Eq. (3.98) is slightly different from that of CV, the difference is fairly small.

```
simp101()
  function()
  {
  # (1)
    set.seed(815)
    nd <- 40
    xx <- seq(from = 1, to = nd, by = 1)
    yy <- xx + 2 + rnorm(nd, mean = 0, sd = 10)
  # (2)
    cv1 <- 0
    for (kk in 1:nd){
      xxd <- xx[-kk]
      yyd <- yy[-kk]
      datad <- data.frame(x = xxd, y = yyd)
      lmd <- lm(y~x, data = datad)
      levd <- lm.influence(lmd)$hat
      datad2 <- data.frame(x=xx[kk])
      eyd <- as.numeric(predict(lmd, newdata = datad2))
      cv1 <- cv1 + (yy[kk]-eyd)^2
    }
    cv1 <- cv1/nd
  # (3)
    data1 <- data.frame(x = xx, y = yy)
    lm1 <- lm(y~x, data = data1)
    lev1 <- lm.influence(lm1)$hat
    ey1 <- fitted(lm1)
    fr1 <- sum(lev1)
    cv2 <- sum(((yy - ey1)/(1 - lev1) )^2)/nd
    gcv1 <- sum( (yy - ey1)^2 )/(nd * (1 - fr1 / nd)^2)
  # (4)
    print("cv1")
    print(cv1)
    print("cv2")
    print(cv2)
    print("gcv1")
    print(gcv1)
  }
```

(1) The number of data (`nd`) is given. The values of the predictor variable (`xx`) of the simulation data and those of the target variable (`yy`) of the simulation data are given.
(2) Values of CV are obtained using Eq. (3.96) (page 150). The result is saved as `cv1`.
(3) The component of `hat` contained in the output of `lm.influence(lm1)` stores the diagonal elements of the hat matrix. The diagonal elements of the hat matrix are called leverages. `fitted(lm1)` yields the estimates corresponding to the values of the target variable of the data. Using these values, CV (Eq. (3.97) (page 151)) corresponding to `lm1` is calculated. The result is saved as `cv2`. GCV (Eq. (3.98) (page 151)) is calculated. The result is saved as `gcv1`.
(4) `cv1`, `cv2`, and `gcv1` are output.

`simp101()` outputs:

```
"cv1"
105.2033
"cv2"
105.2033
"gcv1"
105.7198
```

R Program [3 - 10] End

R Program [3 - 11]

Let us choose between $y = a_0 + a_1 x$ and $y = a_0$ by GCV (Eq. (3.98) (page 151)) using the same data as in `simp41()`.

```
simp61()
  function()
  {
  # (1)
    set.seed(815)
    nd <- 40
    xx <- seq(from = 1, to = nd, by = 1)
    yy <- xx + 40*sin(pi * xx * 0.2) + 2 +
     rnorm(nd, mean = 0, sd = 10)
  # (2)
    data1 <- data.frame(x = xx, y = yy)
    lm0 <- lm(y~1, data = data1)
    lm1 <- lm(y~x, data = data1)
  # (3)
    lev0 <- lm.influence(lm0)$hat
    ey0 <- fitted(lm0)
    fr0 <- sum(lev0)
    gcv0 <- sum( (yy - ey0)^2 )/(nd * (1 - fr0 / nd)^2)
  # (4)
    lev1 <- lm.influence(lm1)$hat
    ey1 <- fitted(lm1)
    fr1 <- sum(lev1)
    gcv1 <- sum( (yy - ey1)^2 )/(nd * (1 - fr1 / nd)^2)
  # (5)
    print("gcv0")
    print(gcv0)
    print("gcv1")
    print(gcv1)
  }
```

(1) The number of data (`nd`) is given. The values of the predictor variable (`xx`) of the simulation data and those of the target variable (`yy`) of the simulation data are given.
(2) `lm()` carries out regression using the equation $y = a_0$. The result is saved as `lm0`. `lm()` carries out regression using the equation $y = a_0 + a_1 x$. The result is saved as `lm1`.
(3) The component of `hat` contained in the output of `lm.influence(lm1)` stores the diagonal elements of the hat matrix. `fitted(lm0)` yields estimates corresponding to the values of the target variable of the data. Using these values, GCV (Eq. (3.98) (page 151)) corresponding to `lm0` is calculated. The result is saved as `gcv0`.
(4) GCV corresponding to `lm1` is calculated. The result is saved as `gcv1`.
(5) `gcv0` and `gcv1` are output.

`simp61()` outputs:

```
"gcv0"
833.0509
"gcv1"
826.8523
```

The result above shows that the one degree polynomial equation is preferable because it gives a smaller prediction error. This conclusion is different from that of `simp41()`. This is because GCV accommodates more predictors than the F-test with a 5 % risk rate (page 250).

R Program [3 - 11] End

3.9 Weighted Regression

Even if the errors $\{\epsilon_i\}$ $(1 \leq i \leq n)$ (Eq. (3.7) (page 110)) contained in the data are considered to obey a normal distribution and have mean 0, the variance of the errors can be nonconstant. This situation implies that the variance should be represented as $\{\sigma_i^2\}$ $(1 \leq i \leq n)$ instead of σ^2. If this is the case, the least squares method should not be used in the form we presented earlier. Equation (3.1) (page 109) should be replaced with the following equation.

$$RSS = \sum_{i=1}^{n} \frac{1}{\sigma_i^2}(y_i - a_0 - a_1 x_i)^2 = \sum_{i=1}^{n} e_i^2. \tag{3.100}$$

When the squared residuals are weighted and their sum is minimized, the regression is called a weighted regression.

Let us conduct a simulation to estimate $\{\sigma_i^2\}$ $(1 \leq i \leq n)$ (Eq. (3.100)) using a smoothing spline (smoothing splines) (a method of nonparametric regression). The

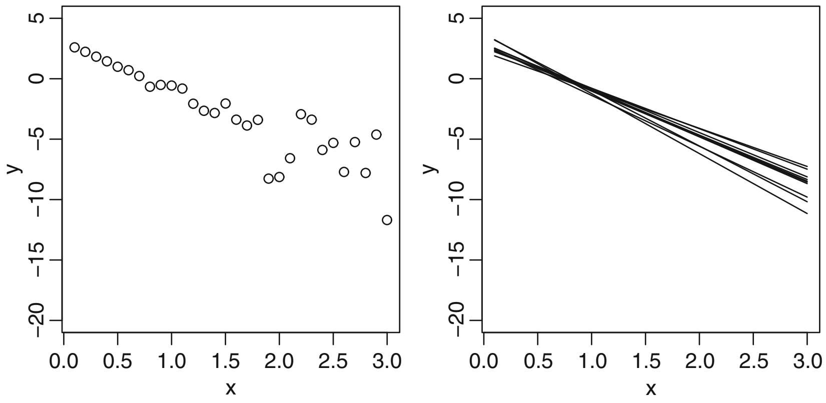


Fig. 3.14 An example of the simulation data given by Eq. (3.101) (*left*). Results of ordinary simple regression using Eq. (3.1) (page 109) for 10 sets of simulation data

number of data are set to 30. The values of the predictor variable ($\{x_i\}$ $(1 \le i \le n)$) are set to $\{0.1, 0.2, 0.3, \ldots, 3.0\}$. The values of the target variable ($\{y_i\}(1 \le i \le n)$) are generated using the equation below.

$$y_i = -4x_i + 3 + \epsilon_i x_i^2, \tag{3.101}$$

where $\{\epsilon_i\}$ are realizations of $N(0, 0.5^2)$ (normal distribution; mean of 0 and variance of 0.5^2). Hence, the absolute values of the errors increase with i. The squared residual for each i multiplied by a constant ($\frac{n}{n-2}$), which provides unbiasedness, is represented as the following equation. Multiplication by $\frac{n}{n-2}$ does not affect the final results.

$$r_i^2 = \frac{n}{n-2}(y_i - \hat{a}_0 + \hat{a}_1 x_i)^2. \tag{3.102}$$

r_i^2 is regarded as a function of i. The natural logarithms of $\{r_i^2\}$ are subjected to smoothing by the smoothing spline. The resulting estimates are transformed by exponential transformation to be used as $\{\sigma_i^2\}$. These values are then applied to a simple regression by Eq. (3.100). Thus, there are three steps to estimating $\{\sigma_i^2\}$: (1) transformation by natural logarithm, (2) smoothing, and (3) exponential transformation. This is because σ_i^2 should not be negative, and even if the estimate of σ_i^2 varies dramatically with i, problematic values should be dealt with appropriately.

An example of the simulation data ($\{x_i\}$ and $\{y_i\}$) is shown in Fig. 3.14 (left). We generate 10 sets of simulation data by varying the initial value of the pseudo-random numbers. The regression equations yielded by ordinary simple regression using Eq. (3.1) (page 109) are depicted in Fig. 3.14 (right). On the other hand, the values of $\{\sigma_i^2\}$ estimated using a smoothing spline are shown in Fig. 3.15 (left).

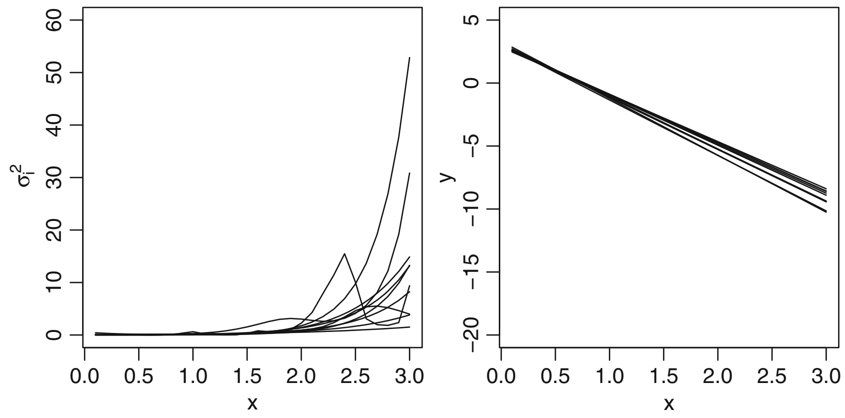


Fig. 3.15 Values of $\{\sigma_i^2\}$ given by the 10 sets of simulation data (*left*). Results of weighted simple regression (Eq. (3.100)) for the 10 sets of simulation data

Using these $\{\sigma_i^2\}$, simple regression using Eq. (3.100) is carried out. The results are drawn in Fig. 3.15 (right). The variation of the regression equations is smaller than that in Fig. 3.14 (right). This result indicates that a reliable regression equation is obtained by weighting the data appropriately.

The transformation of variables is a conventional method for coping with inhomogeneous errors contained in data. The Box-Cox transformation is a typical example. Although it is true that such a method handles the inhomogeneity of errors by transforming variables, it may lead to a regression that does not give a one degree polynomial equation, or it may assume that errors do not follow a normal distribution (pages 286 and 293 in [1]). Therefore, the inhomogeneity of errors and use of regression equations that are not linear should be treated separately. In this way, we can cope with the situation in which a linear equation is preferable but some inhomogeneity of errors is present, or that in which regression equations other than a linear equation are preferable but there is no inhomogeneity of errors.

Then, if a regression equation other than one degree polynomial is desirable, this problem is addressed by choosing one with an appropriate form or using nonparametric regression. Moreover, if there is some inhomogeneity of errors, it is dealt with using RSS, such as in Eq. (3.100) (page 154). Such strategies enable independent processes to handle these two problems. Therefore, we can facilitate the solution of one or both problems. On the other hand, when the distribution obeyed by the errors is not a normal distribution, generalized linear regression can be used successfully in most situations. Broadly speaking, generalized linear regression is a type of weighted regression.

$\boxed{\text{R Program [3 - 12]}}$

Weighted regression is conducted by calculating weights using a smoothing spline.

```
simp182()
  function()
  {
  # (1)
    nd <- 30
    nt <- 10
  # (2)
    xx <- seq(from = 0.1, to = 3, length = nd)
  # (3)
    yym <- matrix(rep(0, length = nd * nt), ncol = nt)
    sig2sm <- matrix(rep(0, length = nd * nt),
     ncol = nt)
    ey2m <- matrix(rep(0, length = nd * nt), ncol = nt)
  # (4)
    for(jj in 1:nt){
      set.seed(6964 + jj * 17)
      yy <- - xx * 4 + 3 + rnorm(nd, mean = 0,
       sd = 0.5) * xx^2
      yym[, jj] <- yy
      data1 <- data.frame(x = xx, y = yy)
      lm1 <- lm(y~x, data = data1)
      ey <- lm1$fitted
  # (5)
      sig2 <- (yy - ey)^2 * nd / (nd-2)
      sig2s <- exp(smooth.spline(xx, log(sig2))$y)
      sig2sm[, jj] <- sig2s
      ww2 <- 1/sig2s
      data2 <- data.frame(x = xx, y = yy)
      lm2 <- lm(y~x, data = data1, weights = ww2)
      ey2 <- lm2$fitted
      ey2m[, jj] <- ey2
    }
  # (6)
    par(mfrow = c(1,2), mai = c(2, 1, 1.5, 0.1),
     omi = c(0, 0, 0, 0))
    yrange <- range(as.vector(sig2sm))
    yrange1 <- min(pretty(yrange))
    yrange2 <- max(pretty(yrange))
    plot(xx, yy, pch = 1, type="n", xlab = "x",
     ylab = expression(sigma[i]^2), ylim=c(yrange1,
     yrange2))
    for(jj in 1:nt){
      lines(xx, sig2sm[,jj])
    }
    yrange <- range(as.vector(yym))
    yrange1 <- min(pretty(yrange))
    yrange2 <- max(pretty(yrange))
    plot(xx, yy, pch = 1, type="n", xlab = "x",
     ylab = "y", ylim = c(yrange1, yrange2))
    for(jj in 1:nt){
      lines(xx, ey2m[,jj])
    }
  }
```

(1) The number of data (`nd`) and the number of simulations (`nt`) are given.
(2) The values of the predictor variable of the simulation data (`xx`) are given.
(3) The matrix (`yym`) for storing all values of the predictor variable of the simulation data, the matrix (`sig2sm`) for storing the estimates of $\{\epsilon_i\}$, and the matrix (`ey2m`) for storing estimates given by the regression equation based on RSS (Eq. (3.100) (page 154)) are prepared.
(4) `lm()` carries out unweighted simple regression (Eq. (3.1) (page 109)). The result is stored in `lm1`. Estimates corresponding to the data are saved as `ey`.
(5) The $\{r_i^2\}$ are estimated using Eq. (3.102) (page 155). Then, the natural logarithms of these values are smoothed by `smooth.spline()`. The resultant values are transformed by exponential transformation. The result is saved as `sig2s`. `sig2s` is organized in the form of `sig2sm[, jj]`. Simple regression (Eq. (3.100) (page 154)) is carried out using `sig2s`. The result is stored in `lm2`. Estimates corresponding to the data are saved as `ey2`. `ey2` is organized in the form of `ey2m[, jj]`.
(6) Values of `sig2s` corresponding to the 10 simulations are drawn.
(7) The regression equations given by the 10 simulations are drawn.

`simp182()` outputs Fig. 3.15 (page 156).

R Program [3 - 12] End

3.10 Least Squares Method and Prediction Error

When the errors contained in the data are close to an independent and identically distributed (i.i.d.) normal distribution, the least squares method is most often used. The least squares method is backed by the method of maximum likelihood (page 121 in [4]). However, when the purpose of producing a regression equation is to estimate values of the target variable with minimum prediction error using the specific values of the predictors, the method of maximum likelihood is not always the best choice. This is because, although the least squares method or the method of maximum likelihood yields a regression equation that fits the data well, the resulting regression equation is not guaranteed to give the best fit to data in the future.

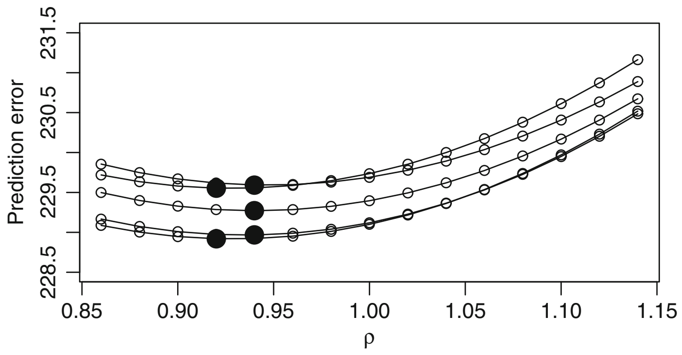


Fig. 3.16 Effect of biasing the slope of the prediction error. The slope is that of the simple regression equation given by the least squares method if the bias is not present. *Open circle* is the average of S. *Filled circle* indicates the minimum value. $\rho = 1$ corresponds to the regression without this bias

We conduct the following simulation. When the data $\{(x_i, y_i)\}$ $(1 \leq i \leq n)$ is given, the resultant regression coefficients given by the least squares method are denoted by $\hat{a}_0$ and $\hat{a}_1$. The other realizations sampled from the population that gave $\{(x_i, y_i)\}$ are represented as $\{(x_i, y_i^*)\}$ $(1 \leq i \leq n)$. K sets of these realizations are generated, and the following value is calculated.

$$S = \frac{1}{K \cdot n} \sum_{k=1}^{K} \sum_{i=1}^{n} (y_i^* - \hat{y}_i)^2, \tag{3.103}$$

where the $\{\hat{y}_i\}$ are defined as

$$\hat{y}_i = \hat{a}_0 - \hat{a}_1 x_i. \tag{3.104}$$

In Eq. (3.103), S is the prediction error obtained from the simple regression equation given by the least squares method.

Next, an alternative regression equation given by $\{(x_i, y_i)\}$ is depicted as

$$y = \rho \hat{a}_1 x + \hat{a}_0 + \frac{(1-\rho)\sum_{j=1}^{n} x_j}{n} \hat{a}_1, \tag{3.105}$$

where ρ is a constant satisfying $0 \leq \rho \leq 1$. This value is generally close to 1. The regression equation represented by Eq. (3.105) is yielded by giving some bias to the simple regression equation by the least squares method. This bias reduces the absolute value of the slope. The value of S (Eq. (3.103)) is calculated using the $\{\hat{y}_i\}$ obtained by substituting $\{x_i\}$ into Eq. (3.105). S is the prediction error of a regression equation in which the slope is biased.

Then, simulation data are generated using the following equation.

$$y_i = 2x_i - 4 + \epsilon_i. \tag{3.106}$$

Here, $\{\epsilon_i\}$ $(1 \le i \le 100)$ are realizations of $N(0, 15^2)$ (normal distribution; mean of 0 and variance of 15^2) and the $\{x_i\}$ $(1 \le i \le 100)$ are $\{0.1, 0.2, 0.3, \ldots, 10\}$. The procedure of calculating S with $K = 10$ is carried out 500 times by altering the initial value of the pseudo-random numbers. The relationship between ρ and S is examined 5 times using one value from among $\{0.86, 0.88, \ldots, 1.14\}$ for ρ in Eq. (3.105). The result is shown in Fig. 3.16. The prediction error takes a smaller value when ρ is smaller than 1. This instance shows that the regression coefficients yielded by the least squares method are not considered to be optimal in terms of prediction.

R Program [3 - 13]

On some occasions, the optimal coefficients of a simple regression in terms of prediction are not obtained by the least squares method.

```
simp201e()
  function()
  {
  # (1)
    nd <- 100
    sd1 <- 15
    nk <- 10
    nt <- 500
    nl <- 5
  # (2)
    xxa <- seq(from = 0.1, to = 10, length = nd)
    rho1t <- seq(from = 0.86, to = 1.14, by = 0.02)
    rssam <- NULL
  # (3)
    for (ll in 1:nl){
      rssa <- NULL
  # (4)
      for (ii in 1:length(rho1t)){
  # (5)
        rho1 <- rho1t[ii]
        rss1 <- 0
  # (6)
        for (jj in 1:nt){
  # (7)
          set.seed(6964*ll + jj * 14)
          yya <- xxa * 2 - 4 + rnorm(nd, mean = 0,
           sd = sd1)
          sxy <- sum((xxa - mean(xxa)) *
           (yya - mean(yya)))
          sxx <- sum((xxa - mean(xxa))^2)
          aa <- sxy/sxx
          bb <- mean(yya) - aa * mean(xxa)
          bbd <- (1 - rho1) * sum(xxa) / nd
          ey <- rho1 * aa * xxa + bb + bbd * aa
# (8)
          for(kk in 1:nk){
            set.seed(3201*ll + kk * 48 + jj * 13)
            yyb <- xxa * 2 - 4 + rnorm(nd, mean = 0,
             sd = sd1)
            rss1 <-  rss1 + sum((yyb - ey)^2)/(nk * nd)
          }
        }
# (9)
        rssa[ii] <- rss1/nt
      }
     rssam <- cbind(rssam, rssa)
    }
# (10)
    par(mfrow = c(1,1), mai = c(2, 1, 1.5, 0.1),
     omi = c(0, 0, 0, 0))
    yrange <- range(as.vector(rssam))
    yrange1 <- min(pretty(yrange))
    yrange2 <- max(pretty(yrange))
    plot(rho1t, rssam[,1], pch = 1, type="n", xlab =
     expression(rho), ylab = "Prediction error",
     ylim = c(yrange1, yrange2))
    for (ll in 1:nl){
      lines(rho1t, rssam[,ll], lwd = 1)
      points(rho1t, rssam[,ll])
      wh2 <- which(rssam[,ll] == min(rssam[,ll]))
      rh2 <- rho1t[wh2]
      min2 <- rssam[wh2,ll]
      points(rh2, min2, pch=16, cex=2)
    }
}
```

(1) The number of data (`nd`), standard deviation of errors (`sd1`), value of K in Eq. (3.103) (page 159) (`nk`), number of times the procedure is repeated to calculate S (Eq. (3.103)) (`nt`), and the number of times the relationship between ρ and S is examined (`nl`) are given.

(2) The values of the predictor variable (`xxa`) are given. The values used as ρ (`rho1t`) are given. The matrix `rssam` for storing the results of the numerical simulation is prepared.

(3) The numerical simulation is carried out `nl` times. The vector `rssa` for storing the values of S corresponding to respective values of ρ is prepared.

(4) The value of S corresponding to each value of ρ is calculated.

(5) The value of ρ is saved as `rho1`.
(6) The procedure of calculating S (Eq. (3.103) (page 159)) is carried out `nt` times.
(7) `mean(yya)` gives the average of `yya`. The estimates given by Eq. (3.105) (page 159) are calculated. The result is saved as `ey`.
(8) The value of S is obtained using the `nk` sets of future data ($\{(x_i, y_i^*)\}$). The result is saved as `rss1`.
(9) The average of the `nt` values of `rss1` is saved as `rssa[ii]`. The values of `rssa` are organized in `rssam`.
(10) `range(rssam)` yields the minimum and maximum values of `rssam`. The relationship between `rho1t` and `rssam` is illustrated in a graph.

R Program [3 - 13] End

References

1. Myers RH (1990) Classical and modern regression with applications (Duxbury Classic). Duxbury, North Scituate
2. Ruppert D, Wand MP, Carroll RJ (2003) Semiparametric regression. Cambridge University Press, Cambridge
3. Ryan TP (1996) Modern regression methods. Wiley-Interscience, New York
4. Takezawa K (2006) Introduction to nonparametric regression. Wiley, New York
5. Takezawa K (2012) Guidebook to R graphics using Microsoft windows. Wiley, New York

Chapter 4
Multiple Regression

4.1 Derivation of Regression Coefficients

When data $\{(x_{i1}, x_{i2}, \ldots, x_{iq}, y_i)\}$ $(1 \leq i \leq n)$ are given, multiple regression derives the values of $\{a_j\}(1 \leq j \leq q)$ by minimizing

$$RSS = \sum_{i=1}^{n}(y_i - a_0 - \sum_{j=1}^{q} a_j x_{ij})^2 = \sum_{i=1}^{n} e_i^2. \tag{4.1}$$

Here $e_i = y_i - a_0 - \sum_{j=1}^{q} a_j x_{ij}$ are called residuals. The acronym RSS stands for the residual sum of squares. Equation (4.1) yields the regression equation:

$$\hat{y} = \hat{a}_0 + \sum_{j=1}^{q} \hat{a}_j x_j, \tag{4.2}$$

where $\{\hat{a}_j\}$ are estimates of the regression coefficients, $\{x_j\}$ the predictor variables, and $\hat{y}$ the estimates of target variables. Using Eq. (4.2), Eq. (4.1) is transformed into

$$RSS = \sum_{i=1}^{n}(y_i - \hat{a}_0 - \sum_{j=1}^{q} \hat{a}_j x_{ij})^2. \tag{4.3}$$

The procedures described above can be represented in a matrix form as follows. First, the n sets of data are arrayed as matrices denoted by $\mathbf{X}$ and $\mathbf{y}$,

$$\mathbf{X} = \begin{pmatrix} 1 & x_{11} & x_{12} & \ldots & x_{1q} \\ 1 & x_{21} & x_{22} & \ldots & x_{2q} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n1} & x_{n2} & \ldots & x_{nq} \end{pmatrix}, \mathbf{y} = \begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_n \end{pmatrix}. \tag{4.4}$$

K. Takezawa, *Learning Regression Analysis by Simulation*,
DOI 10.1007/978-4-431-54321-3_4, © Springer Japan 2014

Here $\mathbf{X}$ is called the design matrix. Next, we set

$$\mathbf{y} = \tilde{\mathbf{y}} + \boldsymbol{\epsilon}, \tag{4.5}$$

where $\tilde{\mathbf{y}}$ and $\boldsymbol{\epsilon}$ are defined as

$$\tilde{\mathbf{y}} = \begin{pmatrix} a_0 + \sum_{j=1}^{q} a_j x_{1j} \\ a_0 + \sum_{j=1}^{q} a_j x_{2j} \\ a_0 + \sum_{j=1}^{q} a_j x_{3j} \\ \vdots \\ a_0 + \sum_{j=1}^{q} a_j x_{nj}, \end{pmatrix}, \quad \boldsymbol{\epsilon} = \begin{pmatrix} \epsilon_1 \\ \epsilon_2 \\ \epsilon_3 \\ \vdots \\ \epsilon_n \end{pmatrix}. \tag{4.6}$$

These $\{a_j\}$ $(0 \le j \le q)$ are values which represent a population (a.k.a. parameters or true values).

Then, $\mathbf{a}$ and $\hat{\mathbf{a}}$ are defined as

$$\mathbf{a} = \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_q \end{pmatrix}, \quad \hat{\mathbf{a}} = \begin{pmatrix} \hat{a}_0 \\ \hat{a}_1 \\ \vdots \\ \hat{a}_q \end{pmatrix}. \tag{4.7}$$

Regression coefficients are derived by minimizing the *RSS* (Eq. (4.1)) in a manner similar to that for simple regression. Thus, we have

$$\mathbf{X}^t\mathbf{X}\mathbf{a} = \mathbf{X}^t\mathbf{y}, \tag{4.8}$$

$$\hat{\mathbf{a}} = (\mathbf{X}^t\mathbf{X})^{-1}\mathbf{X}^t\mathbf{y}, \tag{4.9}$$

$$\hat{\mathbf{y}} = \mathbf{X}\hat{\mathbf{a}} = \mathbf{X}(\mathbf{X}^t\mathbf{X})^{-1}\mathbf{X}^t\mathbf{y}, \tag{4.10}$$

where $\hat{\mathbf{y}}$ is defined as

$$\hat{\mathbf{y}} = \begin{pmatrix} \hat{y}_1 \\ \hat{y}_2 \\ \hat{y}_3 \\ \vdots \\ \hat{y}_n \end{pmatrix} = \begin{pmatrix} \hat{a}_0 + \sum_{j=1}^{q} \hat{a}_j x_{1j} \\ \hat{a}_0 + \sum_{j=1}^{q} \hat{a}_j x_{2j} \\ \hat{a}_0 + \sum_{j=1}^{q} \hat{a}_j x_{3j} \\ \vdots \\ \hat{a}_0 + \sum_{j=1}^{q} \hat{a}_j x_{qj} \end{pmatrix}. \tag{4.11}$$

Equation (4.10) is rewritten as

$$\hat{\mathbf{y}} = \mathbf{H}\mathbf{y}, \tag{4.12}$$

with $\mathbf{H}$, called a hat matrix, which is defined as

$$\mathbf{H} = \mathbf{X}(\mathbf{X}^t\mathbf{X})^{-1}\mathbf{X}^t. \tag{4.13}$$

H is a symmetric matrix for the same reason leading to Eq. (3.15) (page 111).

Furthermore, following a similar discussion to that of simple regression (Eq. (3.26) (page 113)), we have

$$\sum_{i=1}^{n} e_i = 0. \tag{4.14}$$

That is, the sum of the residuals ($\{e_i\}$ $(1 \le i \le n)$) is 0.

Using Eq. (4.14), the equation equivalent to Eq. (4.9) can be derived: For simplicity let $q = 2$ here. First, $\{x'_{i1}\}$, $\{x'_{i2}\}$, and $\{y'_i\}$ are defined as

$$x'_{i1} = x_{i1} - \bar{x}_1, \qquad x'_{i2} = x_{i2} - \bar{x}_2, \qquad y'_i = y_i - \bar{y}, \tag{4.15}$$

where $\bar{x}_1$, $\bar{x}_2$, and $\bar{y}$ are

$$\bar{x}_1 = \frac{\sum_{i=1}^{n} x_{i1}}{n}, \qquad \bar{x}_2 = \frac{\sum_{i=1}^{n} x_{i2}}{n}, \qquad \bar{y} = \frac{\sum_{i=1}^{n} y_i}{n}. \tag{4.16}$$

The transformation of $\{x_{1i}\}$ into $\{x'_{1i}\}$ and that of $\{x_{2i}\}$ into $\{x'_{2i}\}$ using Eq. (4.15) are called centering. Using Eq. (4.15), Eq. (4.1) (page 163) becomes

$$\begin{aligned}
RSS &= \sum_{i=1}^{n}\left(y'_i + \frac{\sum_{i=1}^{n} y_i}{n} - a_0 - a_1\left(x'_{i1} + \frac{\sum_{i=1}^{n} x_{i1}}{n}\right) - a_2\left(x'_{i2} + \frac{\sum_{i=1}^{n} x_{i2}}{n}\right)\right)^2 \\
&= \sum_{i=1}^{n}\left(y'_i - a_0 - a_1 x'_{i1} - a_1 x'_{i2} + \frac{\sum_{i=1}^{n} y_i}{n} - a_1\frac{\sum_{i=1}^{n} x_{i1}}{n} - a_2\frac{\sum_{i=1}^{n} x_{i2}}{n}\right)^2 \\
&= \sum_{i=1}^{n}\left(y'_i - a_0 - a_1 x'_{i1} - a_2 x'_{i2} + \frac{\sum_{i=1}^{n}(y_i - a_1 x_{i1})}{n} + \frac{\sum_{i=1}^{n}(y_i - a_1 x_{i2})}{n}\right)^2 \\
&= \sum_{i=1}^{n}\left(y'_i - a_0 - a_1 x'_{i1} - a_2 x'_{i2} + \frac{\sum_{i=1}^{n}(a_0 + e_i)}{n}\right)^2 \\
&= \sum_{i=1}^{n}\left(y'_i - a_0 - a_1 x'_{i1} - a_1 x'_{i2} + \frac{\sum_{i=1}^{n} a_0}{n}\right)^2 \\
&= \sum_{i=1}^{n}\left(y'_i - a_0 - a_1 x'_{i1} - a_2 x'_{i2} + \frac{n a_0}{n}\right)^2 \\
&= \sum_{i=1}^{n}(y'_i - a_1 x'_{i1} - a_2 x'_{i2})^2.
\end{aligned} \tag{4.17}$$

Differentiation of this *RSS* with respect to a_1 results in

$$\sum_{i=1}^{n} x'_{i1}(y'_i - a_1 x'_{i1} - a_2 x'_{i2}) = 0. \tag{4.18}$$

Conversely, differentiation with respect to a_2 yields

$$\sum_{i=1}^{n} x'_{i2}(y'_i - a_1 x'_{i1} - a_2 x'_{i2}) = 0. \tag{4.19}$$

That is, we have

$$S_{1y} - a_1 S_{11} - a_2 S_{12} = 0, \qquad S_{2y} - a_1 S_{21} - a_2 S_{22} = 0, \tag{4.20}$$

where S_{11}, S_{12}, S_{22}, S_{1y}, and S_{2y} are defined as

$$\begin{aligned} S_{11} &= \sum_{i=1}^{n}(x_{i1} - \bar{x}_1)^2, & S_{12} = S_{21} &= \sum_{i=1}^{n}(x_{i1} - \bar{x}_1)(x_{i2} - \bar{x}_2), \\ S_{22} &= \sum_{i=1}^{n}(x_{i2} - \bar{x}_2)^2, & S_{1y} &= \sum_{i=1}^{n}(x_{i1} - \bar{x}_1)(y_i - \bar{y}), \\ S_{2y} &= \sum_{i=1}^{n}(x_{i2} - \bar{x}_2)(y_i - \bar{y}). \end{aligned} \tag{4.21}$$

Equation (4.20) is represented in a matrix form as

$$\begin{pmatrix} S_{11} & S_{12} \\ S_{21} & S_{22} \end{pmatrix} \begin{pmatrix} a_1 \\ a_2 \end{pmatrix} = \begin{pmatrix} S_{1y} \\ S_{2y} \end{pmatrix}. \tag{4.22}$$

If $\begin{pmatrix} S_{11} & S_{12} \\ S_{21} & S_{22} \end{pmatrix}$ has an inverse matrix; that is, it is a regular matrix, we obtain

$$\begin{pmatrix} a_1 \\ a_2 \end{pmatrix} = \begin{pmatrix} S_{11} & S_{12} \\ S_{21} & S_{22} \end{pmatrix}^{-1} \begin{pmatrix} S_{1y} \\ S_{2y} \end{pmatrix}. \tag{4.23}$$

Using Eq. (1.30) (page 28), we have

$$\begin{pmatrix} a_1 \\ a_2 \end{pmatrix} = \frac{1}{S_{11}S_{22} - S_{12}S_{21}} \begin{pmatrix} S_{22} & -S_{12} \\ -S_{21} & S_{11} \end{pmatrix} \begin{pmatrix} S_{1y} \\ S_{2y} \end{pmatrix}. \tag{4.24}$$

Extraction of the a_1 component of the above equation gives

$$\hat{a}_1 = \frac{S_{22}S_{1y} - S_{12}S_{2y}}{S_{11}S_{22} - S_{12}S_{21}}, \tag{4.25}$$

where a_1 becomes $\hat{a}_1$ because a_1 is estimated. When $S_{12} = S_{21} = 0$ is satisfied, we obtain

$$\hat{a}_1 = \frac{S_{1y}}{S_{11}}. \tag{4.26}$$

This equation is the same as Eq. (3.31) (page 114), and shows that when there is no correlation between $\{x_{1i}\}$ $(1 \leq i \leq n)$ and $\{x_{2i}\}$ $(1 \leq i \leq n)$, a regression coefficient for each predictor of a multiple regression equation is identical to that of simple regression. The same argument holds when q is larger than 2.

Moreover, the following relationship is obtained using the properties of the hat matrix (page 40 in [4]).

$$\begin{aligned}\hat{\mathbf{y}}^t(\mathbf{y} - \hat{\mathbf{y}}) &= (\mathbf{Hy})^t(\mathbf{y} - \mathbf{Hy}) \\ &= \mathbf{y}^t\mathbf{H}^t(\mathbf{y} - \mathbf{Hy}) \\ &= \mathbf{y}^t(\mathbf{Hy} - \mathbf{HHy}) \\ &= \mathbf{y}^t(\mathbf{Hy} - \mathbf{Hy}) \\ &= 0,\end{aligned} \tag{4.27}$$

where the following equation is used to proceed between the third identity to the fourth.

$$\mathbf{HHy} = \mathbf{Hy}. \tag{4.28}$$

The left-hand side of this equation is $\mathbf{H}\hat{\mathbf{y}}$. Hence, the elements of this vector are estimates of a regression equation which is obtained when estimates by the original regression equation are employed as data. Because the estimates ($\mathbf{H}\hat{\mathbf{y}}$) are intuitively identical to $\hat{\mathbf{y}}$ (i.e., $\mathbf{Hy}$), Eq. (4.28) holds. However, Eq. (4.28) can be derived from Eq. (4.10) (page 164).

Using Eqs. (4.14) (page 165) and (4.27), we have

$$\begin{aligned}&\sum_{i=1}^{n}(\hat{y}_i - \bar{y})^2 + \sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \\ &= \sum_{i=1}^{n}(\hat{y}_i^2 - 2\hat{y}_i\bar{y} + \bar{y}^2 + y_i^2 - 2y_i\hat{y}_i + \hat{y}_i^2) \\ &= \sum_{i=1}^{n}(y_i^2 - 2\hat{y}_i\bar{y} + \bar{y}^2 + 2\hat{y}_i^2 - 2y_i\hat{y}_i) \\ &= \sum_{i=1}^{n}(y_i^2 - 2y_i\bar{y} + \bar{y}^2)\end{aligned}$$

$$= \sum_{i=1}^{n}(y_i - \bar{y})^2, \tag{4.29}$$

where the relationship $-2\bar{y}\sum_{i=1}^{n}\hat{y}_i = -2\bar{y}\sum_{i=1}^{n}y_i$, which is used to derive the fourth line from the third line, follows from Eq. (4.14) (page 165). The equation $2\sum_{i=1}^{n}\hat{y}_i^2 - 2\sum_{i=1}^{n}y_i\hat{y}_i = 0$ is obtained from Eq. (4.27). On the one hand, $\sum_{i=1}^{n}(y_i - \bar{y})^2$ in Eq. (4.29) denotes the total variability in target variable. On the other hand, $\sum_{i=1}^{n}(\hat{y}_i - \bar{y})^2$ represents the variability explained by regression equation. Thus, $\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$ is interpreted as the variability not explained by the regression equation; that is, it is the residual sum of squares. Equation (4.29) holds between these three values. It should be noted, however, that this relationship does not always hold for all regressions; it may not hold even for regressions using least squares. Even if the regression equation is obtained by least squares, the sum of residuals can be nonzero. That is, Eq. (4.14) might not be satisfied. If it does, Eq. (4.29) usually does not hold.

When both sides of Eq. (4.29) are divided by $\sum_{i=1}^{n}(y_i - \bar{y})^2$, we obtain

$$\frac{\sum_{i=1}^{n}(\hat{y}_i - \bar{y})^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} + \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} = 1. \tag{4.30}$$

$\dfrac{\sum_{i=1}^{n}(\hat{y}_i - \bar{y})^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$ is represented as R^2. The following equation is then obtained;

$$\begin{aligned} R^2 &= \frac{\sum_{i=1}^{n}(\hat{y}_i - \bar{y})^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \\ &= 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y})^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}. \end{aligned} \tag{4.31}$$

This R^2 is called the coefficient of determination. The right-hand side of the first line of Eq. (4.31) signifies the proportion of variability explained by the regression equation. As both denominator and numerator in this fraction are positive, we have $0 \leq R^2$. Furthermore, as both denominator and numerator of the second term on the right-hand side of the second line of Eq. (4.31) are also positive, $R^2 \leq 1$ holds. Hence, we obtain the inequality $0 \leq R^2 \leq 1$. The coefficient of determination in multiple regression can be geometrically interpreted (page 41 in [4]), which makes this concept clearer. Even if Eqs. (4.14) and (4.27) do not hold, the values using the definitions of the first line and that of the second line can be calculated. The two values are, however, not necessarily identical.

R Program [4 - 1]

The regression coefficients of multiple regression are calculated.
`mul1e()`

```
function ()
{
# (1)
  set.seed(813)
  nd <- 5
  xx1 <- runif(n = nd, min = 0, max = 10)
  xx2 <- runif(n = nd, min = -5, max = 5)
  yy <- xx1 * 2.5 - xx2 * 4 + 7.7 + rnorm(nd,
   mean = 0, sd = 3)
# (2)
  xxm <- matrix( c(rep(1, length=nd), xx1, xx2),
   ncol = 3)
  print("---------- (2) ----------")
  print("xxm(design matrix)")
  print(xxm)
  yym <- matrix(yy, ncol = 1)
  print("yym")
  print(yym)
# (3)
  print("---------- (3) ---------")
  print("Regression coefficient given by Eq.(4.9)")
  aa <- solve(t(xxm) %*% xxm) %*% t(xxm) %*% yym
  print("aa")
  print(aa)
  hat1 <- xxm %*% solve(t(xxm) %*% xxm) %*% t(xxm)
  print("hat1(hat matrix)")
  print(hat1)
# (4)
  print("---------- (4) ----------")
  print("Regression coefficient given by Eq.(4.24)")
  s11 <- sum((xx1 - mean(xx1))^2)
  s12 <- sum((xx1 - mean(xx1)) * (xx2 - mean(xx2)))
  s21 <- s12
  s22 <- sum((xx2 - mean(xx2))^2)
  smat <- matrix(c(s11, s12, s21,s22), ncol = 2)
  s1y <- sum((xx1 - mean(xx1)) * (yy - mean(yy)))
  s2y <- sum((xx2 - mean(xx2)) * (yy - mean(yy)))
  svec <- c(s1y, s2y)
  aas <- solve(smat, svec)
  print(aas)
# (5)
  print("---------- (5) ----------")
  data1 <- data.frame(x1 = xx1, x2 = xx2, y = yy)
  lm1 <- lm(y~x1 + x2, data = data1)
  print("Regression coefficient given by lm()")
```

```
    print(lm1)
    hat1 <- influence(lm1)$hat
    print("leverages(the on-diagonal element")
    print("(diagonal element) of the hat matrix given")
    print("by lm() and influence()")
    print(hat1)
  # (6)
    print("---------- (6) ----------")
    xxm2 <- matrix(c(xx1, xx2), ncol = 2)
    print("xxm2")
    print(xxm2)
    ls1 <- lsfit(xxm2, yy)
    print("Regression coefficient given by lsfit()")
    print(ls1$coef)
  }
```

(1) The number of data (`nd`) is given. Data of the predictors (`xx1`, `xx2`) and that of the target variable (`yy`) are generated.
(2) The design matrix (Eq. (4.4) (page 163)) is produced as `xxm`; it is output. `yy` is transformed into a matrix form and is denoted by `yym`; it is output.
(3) The regression coefficients are obtained using Eq. (4.9) (page 164) and are output. `solve()` calculates the inverse matrix. The hat matrix is derived using Eq. (4.13) (page 164) and is output.
(4) The regression coefficients (except a_0) are derived using Eq. (4.24) (page 166). The result is output.
(5) `lm()` constructs a multiple regression equation. The regression coefficients and the leverages (the on-diagonal element (diagonal element) of the hat matrix) are calculated and output.
(6) The design matrix to be used in `lsfit()`, in which the column corresponding to the constant term is deleted, is constructed; it is named `xxm2`. `lsfit()` pruduces a multiple regression equation. The resultant regression coefficients are output.

`mul1e()` outputs:

```
"---------- (2) ----------"
"xxm(design matrix)"
     [,1]     [,2]       [,3]
[1,]    1 6.140967  2.1283981
[2,]    1 2.102733  0.9101128
[3,]    1 1.377456  0.1709478
[4,]    1 5.444335  2.9343503
[5,]    1 2.389563 -1.1456652
"yym"
          [,1]
[1,] 18.839410
[2,]  7.682364
```

```
[3,]  9.547659
[4,] 11.680424
[5,] 18.811897
"---------- (3) ----------"
"Regression coefficient given by Eq.(4.9)"
"aa"
          [,1]
[1,]  4.171011
[2,]  3.974723
[3,] -4.736220
"hat1(hat matrix)"
            [,1]        [,2]        [,3]        [,4]
      [,5]
[1,]  0.63548334 -0.08884695 -0.15599454  0.38761732
 0.2217408
[2,] -0.08884695  0.45039481  0.44451482  0.20477932
-0.0108420
[3,] -0.15599454  0.44451482  0.49221212  0.06506651
 0.1542011
[4,]  0.38761732  0.20477932  0.06506651  0.56477325
-0.2222364
[5,]  0.22174083 -0.01084200  0.15420109 -0.22223640
 0.8571365
"---------- (4) ----------"
"Regression coefficient given by Eq.(4.24)"
3.974723 -4.736220

"---------- (5) ----------"
"Regression coefficient given by lm()"
Call:
lm(formula = y ~ x1 + x2, data = data1)
Coefficients:
(Intercept)           x1           x2
      4.171        3.975       -4.736
"leverages(the on-diagonal element"
"(diagonal element) of the hat matrix given"
"by lm() and influence()"
        1         2         3         4         5
0.6354833 0.4503948 0.4922121 0.5647732 0.8571365
"---------- (6) ----------"
"xxm2"
         [,1]      [,2]
[1,] 6.140967  2.1283981
[2,] 2.102733  0.9101128
[3,] 1.377456  0.1709478
```

```
  [4,] 5.444335  2.9343503
  [5,] 2.389563 -1.1456652
  "Regression coefficient given by lsfit()"
  Intercept        X1        X2
   4.171011  3.974723 -4.736220
```

R Program [4 - 1] End

R Program [4 - 2]

When Eq. (4.14) (page 165) does not hold, Eq. (4.29) (page 167) does not hold, either. The following R program exemplifies this point.

```
mul3()
  function ()
  {
  # (1)
    set.seed(3967)
    nd <- 10
    xx1 <- runif(n = nd, min = 0, max = 10)
    xx2 <- runif(n = nd, min = -5, max = 5)
    yy <- -xx1 *5 + xx2*4 + 9.4 + rnorm(nd,
     mean = 0, sd = 2)
  # (2)
    data1 <- data.frame(x1 = xx1, x2 = xx2, y = yy)
    lm1 <- lm(y~x1 + x2, data = data1)
    print("lm1$coef")
    print(lm1$coef)
    print("sum(lm1$residuals)")
    print(sum(lm1$residuals))
  # (3)
    sa1 <- sum((yy - mean(yy))^2)
    sb1 <- sum((lm1$fitted - mean(yy))^2)
    sc1 <- sum(lm1$residuals^2)
    print("sa1")
    print(sa1)
    print("sb1 + sc1")
    print(sb1 + sc1)
  # (4)
    lm2 <- lm(y~x1 + x2 - 1, data = data1)
    print("lm2$coef")
    print(lm2$coef)
    print("sum(lm2$residuals)")
    print(sum(lm2$residuals))
  # (5)
    sa2 <- sum((yy - mean(yy))^2)
    sb2 <- sum((lm2$fitted - mean(yy))^2)
    sc2 <- sum(lm2$residuals^2)
    print("sa2")
    print(sa2)
    print("sb2 + sc2")
    print(sb2 + sc2)
  }
```

(1) The number of data (`nd`) is given. The simulation data (values of the predictors are saved as `xx1` and `xx2`, values of the target variable are as `yy`) are generated.
(2) `lm()` conducts multiple regression. The result is stored in `lm1`. The regression equation is described as $(a_0 + a_1x_1 + a_2x_2)$. The resultant regression coefficients (`lm1$coef`) are displayed. The sum of the residuals (`sum(lm1$residuals)`) is displayed.
(3) The value of the right-hand side of Eq. (4.29) (page 167) (the total variability in target variable) is calculated, here denoted by `sa1`. The first term of the left-hand side of Eq. (4.29) (the variability explained by regression equation) is calculated, here denoted by `sb1`. The second term of the left-hand side of Eq. (4.29) (the variability not explained by regression equation; i.e., the residual sum of squares) is calculated, here denoted by `sc1`. `sa1` is displayed as well as `sb1+sc1`.
(4) `lm()` carries out the multiple regression with the result saved in `lm2`. The regression equation is $(a_1x_1 + a_2x_2)$ (a constant term is excluded). The resulting regression coefficients (`lm2$coef`) are displayed. The sum of residuals (`sum(lm2$ residuals)`) is displayed.
(5) The total variability in target variable is calculated and saved as `sa2`. The variability explained by regression equation is calculated and saved as `sb2`. The variability not explained by regression equation (i.e., residual sum of squares) is calculated and saved as `sc2`. `sa2` is displayed along with `sb2+sc2`.

`mul3()` outputs:

```
"lm1$coef"
(Intercept)          x1          x2
   8.100117   -4.785944    4.001354
"sum(lm1$residuals)"
-5.551115e-17
"sa1"
4242.92
"sb1 + sc1"
4242.92

"lm2$coef"
       x1        x2
-3.663849  4.519819
"sum(lm2$residuals)"
16.36483
```

```
"sa2"
4242.92
"sb2 + sc2"
3655.579
```

When the regression equation $(a_0 + a_1x_1 + a_2x_2)$ is employed, Eqs. (4.14) (page 165) and (4.29) (page 167) hold. However, if the regression equation $(a_1x_1 + a_2x_2)$ is adopted, neither of the two equations hold.

R Program [4-2] End

4.2 Test on Multiple Regression

Using Eqs. (4.3) and (4.13), the expectation of Eq. (4.3) is represented as

$$\begin{aligned} E[RSS] &= E[(\mathbf{y}-\hat{\mathbf{y}})^t(\mathbf{y}-\hat{\mathbf{y}})] \\ &= E[\mathbf{y}^t(\mathbf{I}-\mathbf{H})^t(\mathbf{I}-\mathbf{H})\mathbf{y}] \\ &= E[\mathbf{y}^t(\mathbf{I}-\mathbf{H})\mathbf{y}] \\ &= E[(\tilde{\mathbf{y}}^t+\boldsymbol{\epsilon}^t)(\mathbf{I}-\mathbf{H})(\tilde{\mathbf{y}}+\boldsymbol{\epsilon})] \\ &= E[\boldsymbol{\epsilon}^t\boldsymbol{\epsilon}-\boldsymbol{\epsilon}^t\mathbf{H}\boldsymbol{\epsilon}], \end{aligned} \tag{4.32}$$

where $\mathbf{I}$ is an identity matrix (the size is $n \times n$). This equation is obtained using the properties: $\mathbf{H}$ is a symmetric matrix ($\mathbf{H}^t = \mathbf{H}$), and if the values of the target variable are true values ($\tilde{\mathbf{y}}$), the estimates are also true values ($\mathbf{H}\tilde{\mathbf{y}} = \tilde{\mathbf{y}}$). Equation (3.38) (page 115) yields

$$E[\boldsymbol{\epsilon}^t\boldsymbol{\epsilon}] = E\left[\sum_{i=1}^{n}\epsilon_i\epsilon_i\right] = n\sigma^2. \tag{4.33}$$

The equation below is obtained in a way analogous to Eq. (3.40) (page 115).

$$E[\boldsymbol{\epsilon}^t\mathbf{H}\boldsymbol{\epsilon}] = E\left[\sum_{i=1}^{n}\sum_{j=1}^{n}\epsilon_i[\mathbf{H}]_{ij}\epsilon_j\right] = \text{trace}(\mathbf{H})\sigma^2 = (q+1)\sigma^2, \tag{4.34}$$

where trace($\mathbf{H}$) stands for the trace (sum of the diagonal elements of a matrix) of $\mathbf{H}$. Equations (4.32)–(4.34) yield

$$E[RSS] = (n-q-1)\sigma^2. \tag{4.35}$$

Lying behind this equation is the following probability density function which RSS/σ^2 obeys.

$$RSS/\sigma^2 \sim \chi^2_{n-q-1}. \tag{4.36}$$

This equation indicates that RSS/σ^2 obeys the χ^2-distribution with $(n - q - 1)$ degrees of freedom. However, Eqs. (4.35) and (4.36) can be used when Eq. (4.6) (page 164) holds. When the data do not even approximately satisfy the condition that the data are the sum of the value given by Eq. (4.6) and the equal variance error, Eqs. (4.35) and (4.36) cannot be used. On the other hand, if one or more elements of $\{a_j\}$ $(1 \leq j \leq q)$ are 0, Eqs. (4.35) and (4.36) can be used. That is, if the form of the equation which generated the data is as in Eq. (4.6) or contains Eq. (4.6) as a special case, Eqs. (4.35) and (4.36) are valid.

Equation (4.3) (page 163) defines the residual sum of squares given by estimates of the target variable calculated by a regression equation and values of the target variable of the data used for constructing the regression equation. We also need another residual sum of squares given by estimates of the target variable calculated by a regression equation and values of the target variable of the new data which was not used for constructing the regression equation. The values of the target variable of the new data are denoted by $\mathbf{y}^*$ and are defined as

$$\mathbf{y}^* = \tilde{\mathbf{y}} + \boldsymbol{\epsilon}^*, \tag{4.37}$$

where $\boldsymbol{\epsilon}^*$ is given as

$$\boldsymbol{\epsilon}^* = \begin{pmatrix} \epsilon_1^* \\ \epsilon_2^* \\ \epsilon_3^* \\ \vdots \\ \epsilon_n^* \end{pmatrix}. \tag{4.38}$$

$\{\epsilon_i^*\}$ $(1 \leq i \leq n)$ are errors which are generated by the same probability density function as that of $\{\epsilon_i\}$ $(1 \leq i \leq n)$.

The residual sum of squares for this occasion is denoted by RSS^*. RSS^* is written as

$$\begin{aligned} E[RSS^*] &= E[(\mathbf{y}^* - \hat{\mathbf{y}})^t(\mathbf{y}^* - \hat{\mathbf{y}})] \\ &= E[(\tilde{\mathbf{y}} + \boldsymbol{\epsilon}^* - \mathbf{H}(\tilde{\mathbf{y}} + \boldsymbol{\epsilon}))^t(\tilde{\mathbf{y}} + \boldsymbol{\epsilon}^* - \mathbf{H}(\tilde{\mathbf{y}} + \boldsymbol{\epsilon}))] \\ &= E[(\tilde{\mathbf{y}}(\mathbf{I} - \mathbf{H}) + \boldsymbol{\epsilon}^* - \mathbf{H}\boldsymbol{\epsilon})^t(\tilde{\mathbf{y}}(\mathbf{I} - \mathbf{H}) + \boldsymbol{\epsilon}^* - \mathbf{H}\boldsymbol{\epsilon})] \\ &= E[\boldsymbol{\epsilon}^{*t}\boldsymbol{\epsilon}^* + \boldsymbol{\epsilon}\mathbf{H}\boldsymbol{\epsilon}] \\ &= n\sigma^2 + \text{trace}(\mathbf{H})\sigma^2 \\ &= (n + q + 1)\sigma^2. \end{aligned} \tag{4.39}$$

The background of this equation is that the probability density function which RSS^*/σ^2 obeys is represented as

$$RSS^*/\sigma^2 \sim \chi^2_{n+q+1}. \tag{4.40}$$

This equation indicates that RSS^*/σ^2 obeys the χ^2-distribution with $(n + q + 1)$ degrees of freedom.

As for simple regression (Eq. (3.58) (page 128)), the variance-covariance matrix of $\hat{\mathbf{a}}$ (the elements of $\hat{\mathbf{a}}$ are regarded as random variables) is called $\mathbf{C}$. Then, $\mathbf{C}$ is written as

$$\mathbf{C} = \begin{pmatrix} E\Big[\Big(\hat{a}_0 - E[\hat{a}_0]\Big)\Big(\hat{a}_0 - E[\hat{a}_0]\Big]\Big] & \dots & E\Big[\Big(\hat{a}_0 - E[\hat{a}_0]\Big]\Big(\hat{a}_q - E[\hat{a}_q]\Big)\Big] \\ \vdots & \ddots & \vdots \\ E\Big[\Big(\hat{a}_q - E[\hat{a}_q]\Big)\Big(\hat{a}_0 - E[\hat{a}_0]\Big)\Big] & \dots & E\Big[\Big(\hat{a}_q - E[\hat{a}_q]\Big)\Big(\hat{a}_q - E[\hat{a}_q]\Big)\Big] \end{pmatrix}. \tag{4.41}$$

In a manner analogous to that of simple regression, $\mathbf{C}$ is estimated using the following:

$$\begin{aligned} \mathbf{C} &= (\mathbf{X}^t\mathbf{X})^{-1}\hat{\sigma}^2 \\ &= \frac{(\mathbf{X}^t\mathbf{X})^{-1}\sum_{i=1}^{n}(y_i - a_0 - \sum_{j=1}^{q} a_j x_{ij})^2}{n-q-1}. \end{aligned} \tag{4.42}$$

One of the properties of the variance-covariance matrix is that the variance-covariance matrix is a symmetric matrix, and all of the eigenvalues (characteristic values) are positive or 0. This property of the variance-covariance matrix is called positive semidefiniteness; refer to the description associated with Eq. (1.77) (page 43). The proof is as follows.

$\mathbf{C}$ is written as

$$\mathbf{C} = E\left[\begin{pmatrix} \hat{a}_0 - E[\hat{a}_0] \\ \vdots \\ \hat{a}_q - E[\hat{a}_q] \end{pmatrix} \Big((\hat{a}_0 - E[\hat{a}_0]) \quad \dots \quad (\hat{a}_q - E[\hat{a}_q]) \Big)\right]. \tag{4.43}$$

An arbitrary vector $\mathbf{v}$ is defined as

$$\mathbf{v} = \begin{pmatrix} v_0 \\ v_1 \\ \vdots \\ v_q \end{pmatrix}. \tag{4.44}$$

Then, $\mathbf{v}^t\mathbf{C}\mathbf{v}$ becomes

$$\begin{aligned}
&\mathbf{v}^t\mathbf{C}\mathbf{v} = \\
&\begin{pmatrix} v_0 & \dots & v_q \end{pmatrix} E\left[\begin{pmatrix} \hat{a}_0 - E[\hat{a}_0] \\ \vdots \\ \hat{a}_q - E[\hat{a}_q] \end{pmatrix} \begin{pmatrix} (\hat{a}_0 - E[\hat{a}_0]) & \dots & (\hat{a}_q - E[\hat{a}_q]) \end{pmatrix}\right] \begin{pmatrix} v_0 \\ \vdots \\ v_q \end{pmatrix} \\
&= E\left[\begin{pmatrix} v_0 & \dots & v_q \end{pmatrix} \begin{pmatrix} \hat{a}_0 - E[\hat{a}_0] \\ \vdots \\ \hat{a}_q - E[\hat{a}_q] \end{pmatrix} \begin{pmatrix} (\hat{a}_0 - E[\hat{a}_0]) & \dots & (\hat{a}_q - E[\hat{a}_q]) \end{pmatrix} \begin{pmatrix} v_0 \\ \vdots \\ v_q \end{pmatrix}\right] \\
&= E\left[\sum_{j=0}^{q} v_j^2 (\hat{a}_j - E[\hat{a}_j])^2\right]. \qquad (4.45)
\end{aligned}$$

As the content of $E[\cdot]$ is positive or 0, $\mathbf{v}^t\mathbf{C}\mathbf{v}$ is positive or 0, indicating that when $\mathbf{v}^t\mathbf{C}\mathbf{v}$ is transformed into diagonal form as in Eq. (1.76) (page 41), all of eigenvalues are positive or 0. Then, $\mathbf{C}$ is positive semidefinite (Q.E.D).

When we choose which regression equation we should use: $y = a_0 + a_1x_1 + a_2x_2$ or $y = a_0 + a_1x_1$, the t-test is applicable. That is, the null hypothesis is set as

$$H_0 : a_2 = 0, \qquad (4.46)$$

with the alternative hypothesis set as

$$H_1 : a_2 \neq 0. \qquad (4.47)$$

If the null hypothesis is rejected and the alternative hypothesis is adopted, the null hypothesis that $a_2 = 0$ holds in the population is then denied. That is, $y = a_0 + a_1x_1 + a_2x_2$ is used as a regression equation. To carry out this test of hypothesis, the t-value is defined in a manner analogous to Eq. (3.79) (page 132), that is to say,

$$t = \frac{\hat{a}_2}{\sqrt{[\mathbf{C}]_{33}}}. \qquad (4.48)$$

If the null hypothesis (Eq. (4.46)) is satisfied, the t-value obeys the t-distribution with ϕ degrees of freedom in a fashion similar to Eq. (3.80) (page 132). This is described as

$$t \sim t(\phi). \qquad (4.49)$$

ϕ (degrees of freedom) in this equation is $(n - 3)$ because this regression equation uses three regression coefficients: a_0, a_1, and a_2.

The F-test is carried out in a way similar to that for simple regression. The multiple regression equation which we assume is set as

$$\begin{aligned} y &= a_0 + \sum_{j=1}^{r} a_j x_j + \sum_{j=r+1}^{q} a_j x_j \\ &= a_0 + \mathbf{a_1}^t \cdot \mathbf{x_1} + \mathbf{a_2}^t \cdot \mathbf{x_2}, \end{aligned} \tag{4.50}$$

where $r < q$. $\mathbf{a_1}$, $\mathbf{x_1}$, $\mathbf{a_2}$, and $\mathbf{x_2}$ are defined as follows.

$$\mathbf{a_1} = \begin{pmatrix} a_1 \\ a_2 \\ \vdots \\ a_r \end{pmatrix}, \quad \mathbf{a_2} = \begin{pmatrix} a_{r+1} \\ a_{r+2} \\ \vdots \\ a_q \end{pmatrix}, \quad \mathbf{x_1} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_r \end{pmatrix}, \quad \mathbf{x_2} = \begin{pmatrix} x_{r+1} \\ x_{r+2} \\ \vdots \\ x_q \end{pmatrix}. \tag{4.51}$$

The null hypothesis (H_0) and the alternative hypothesis (H_1) are set as below.

$$H_0 : \mathbf{a_1} = \mathbf{0}, \tag{4.52}$$

$$H_1 : \mathbf{a_1} \neq \mathbf{0}, \tag{4.53}$$

where $\mathbf{0}$ is a r-length column vector in which all elements are 0. As seen above, if F-test is used for multiple regression, the null hypothesis and the alternative hypothesis can be set up as plural predictors. This is different from the t-test for multiple regression. F-value for this setting is obtained as

$$F(H_0, H_1) = \frac{\dfrac{RSS(\mathbf{a_2}) - RSS(\mathbf{a_1}, \mathbf{a_2})}{r}}{\dfrac{RSS(\mathbf{a_1}, \mathbf{a_2})}{n - q - 1}}, \tag{4.54}$$

where $RSS(\mathbf{a_1}, \mathbf{a_2})$ is the residual sum of squares when $y = a_0 + \mathbf{a_1}^t \mathbf{x_1} + \mathbf{a_2}^t \mathbf{x_2}$ is employed. $RSS(\mathbf{a_2})$ is the residual sum of squares when $y = a_0 + \mathbf{a_2}^t \mathbf{x_2}$ is used.

When the null hypothesis (Eq. (4.52)) holds, we have

$$F(H_0, H_1) \sim F_{r,n-q-1}, \tag{4.55}$$

which is an F-distribution with the first degree of freedom (r) and the second degree of freedom ($n - q - 1$). Next, we calculate

$$p = \int_{F(H_0,H_1)}^{\infty} den(x, r, n - q - 1) dx, \tag{4.56}$$

where $den(x, r, n - q - 1)$ is the probability density function $F_{r,n-q-1}$. If the p-value is less than 0.05, the null hypothesis is rejected and the alternative hypothesis is adopted.

However, problems can arise if the t-test or F-test is carried out to produce a multiple regression equation. This is easily demonstrated when the degrees of the polynomial equation is selected as we will discuss later.

R Program [4-3]

One predictor of a multiple regression is tested using the t-test.

```
mul6()
  function ()
  {
  # (1)
    set.seed(813)
    nd <- 100
    xx1 <- runif(nd, min = -2, max = 3)
    xx2 <- runif(nd, min = -1, max = 5)
    yy <- xx1*0.4 - xx2*0.07 -8 + rnorm(nd, mean = 0,
     sd = 1)
  # (2)
    xxm <- matrix(c(rep(1, length = nd), xx1, xx2),
     ncol = 3)
    yym <- matrix(yy, ncol = 1)
  # (3)
    aa <- solve(t(xxm) %*% xxm) %*% t(xxm) %*% yym
    ey <- aa[1] + aa[2]*xx1 + aa[3]*xx2
  # (4)
    data1 <- data.frame(x1 = xx1, x2 = xx2, y = yy)
    lm2 <- lm(y~., data = data1)
    sum2 <- summary(lm2)
    print(sum2)
    ey <- lm2$fitted
  # (5)
    sig2 <- sum((yy - ey)^2)/(nd-3)
    cc <- sig2 * solve(t(xxm) %*% xxm)
  # (6)
    tt2 <- aa[3]/sqrt(cc[3,3])
    print("tt2")
    print(tt2)
  # (7)
    pval2 <- (1- pt(abs(tt2), df = nd-3))*2
    print("pval2")
    print(pval2)
  }
```

(1) The number of data (`nd`) is given. The simulation data for the first predictor are stored in `xx1`. Those for the second predictor are stored in `xx2`. The data for the target variable are stored in `yy`.
(2) The data are transformed into matrix form as in Eq. (4.4) (page 163). The design matrix is denoted by `xxm`. The matrix of the target variable is denoted by `yym`.
(3) The regression coefficients ($\hat{\mathbf{a}}$) are calculated using Eq. (4.9) (page 164). The result is denoted by `aa`. Estimates corresponding to respective elements of `yy` are derived using `aa`. The estimates are named `ey`.
(4) `xx1`, `xx2`, and `yy` are organized in the data frame of `data1`. In the data frame, `xx1` is called `x1`, `xx2` is called `x2`, and `yy` is called `y`. Then, `lm()` conducts a multiple regression. The result is saved in `lm2`. `summary()` extracts the outline of `lm2`. The summary is called `suma2` which is then displayed.
(5) **C** (Eq. (4.42) (page 176)) is obtained and denoted by `cc`.
(6) The t-value is calculated using Eq. (4.48) (page 177). The result is stored in `tt2`.
(7) `pt()` yields the p-value corresponding to `tt2`, and the result is then saved as `pval2` and displayed.

`mul6()` outputs:

```
Call:
lm(formula = y ~ ., data = data1)

Residuals:
Min      1Q       Median   3Q       Max
-1.72155 -0.69734 -0.07087  0.76642  2.06612

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -8.04273 0.14459    -55.623 < 2e-16 ***
x1           0.46549 0.06000     7.759  8.7e-12 ***
x2          -0.09178 0.05311    -1.728  0.0872 .
---
Signif. codes:0 '***'0.001'**'0.01'*'0.05'.'0.1' '1

Residual standard error: 0.8875 on 97 degrees
 of freedom
Multiple R-squared:0.3895, Adjusted R-squared:0.3769
F-statistic:30.94 on 2 and 97 DF, p-value:4.045e-11

"tt2"
-1.728031
"pval2"
0.08716502
```

The value (`-1.251`) on the row of `x2` and the column of `t value` is identical to `tt2`. The value (`0.872`) on the row of `x2` and the column of `Pr(>|t|)` is the same as `pval2`.

R Program [4-3] End

R Program [4-4]

The one predictor of a multiple regression equation is tested by the F-test.

```
mul11()
  function ()
  {
  # (1)
    set.seed(813)
    nd <- 100
    xx1 <- runif(nd, min = -2, max = 3)
    xx2 <- runif(nd, min = -1, max = 5)
    yy <- xx1*0.4 - xx2*0.07 -8 + rnorm(nd, mean = 0,
     sd = 1)
  # (2)
    data1 <- data.frame(x1=xx1, x2=xx2, y=yy)
    lm2 <- lm(y~., data=data1)
    lm1 <- lm(y~x1, data=data1)
  # (3)
    anova1 <- anova(lm1, lm2)
    print("anova1")
    print(anova1)
  # (4)
    ff3 <- (sum(lm1$residuals^2) - sum(lm2$residuals^2
     ))/1/((sum(lm2$residuals^2))/(nd-3))
    print("ff3")
    print(ff3)
  # (5)
    pval3 <- 1- pf(ff3, df1=1, df=nd-3)
    print("pval3")
    print(pval3)
  }
```

(1) Simulation data are generated. The simulation data for the first predictor are denoted by `xx1`, those for the second are denoted by `xx2`, and those for the target variable are denoted by `yy`.
(2) `xx1`, `xx2`, and `yy` are organized in the data frame of `data1`. `lm()` carries out the regression to obtain regression equation $y = a_0 + a_1x_1 + a_2x_2$. The result is stored in `lm2`. Then, `lm()` conducts a regression to obtain regression equation $y = a_0 + a_1x_1$. The result is saved as `lm1`.

(3) `anova()` carries out the F-test (another saying is analysis of variance) for comparing `lm1` and `lm2`.
(4) The F-value (Eq. (4.54) (page 178)) is calculated. The result is saved as `ff3`.
(5) The p-value corresponding to `ff3` is derived using Eq. (4.56) (page 178).

`mul11()` outputs:

```
"anova1"
Analysis of Variance Table
Model 1: y ~ x1
Model 2: y ~ x1 + x2
     Res.Df \mathit{RSS}    Df  Sum of Sq  F       Pr(>F)
1    98     78.748
2    97     76.396 1   2.3518     2.9861 0.08717.
---
Signif. codes: 0'***'0.001'**'0.01'*'0.05'.'0.1' '1
"ff3"
2.986089
"pval3"
0.08716502
```

The value (`2.9861`) on the row of `2` and the column of `F` in `anova1` is the same as `ff3`. The value (`0.08717`) on the row of `2` and the column of `Pr(>F)` in `anova1` is the same as `pval3`. `pval3` is identical to `pval2` resulted from `mul6()`.

R Program [4-4] End

R Program [4-5]

Two predictors of the multiple regression equation are tested using F-test.

`mul12()`

```
function ()
{
# (1)
  set.seed(813)
  nd <- 100
  xx1 <- runif(nd, min = -2, max = 3)
  xx2 <- runif(nd, min = -1, max = 5)
  yy <- xx1*0.4 - xx2*0.07 -8 + rnorm(nd, mean = 0,
   sd = 1)
# (2)
  data1 <- data.frame(x1=xx1, x2=xx2, y=yy)
  lm2 <- lm(y~., data=data1)
  lm0 <- lm(y~1, data=data1)
# (3)
  anova1 <- anova(lm0, lm2)
  print("anova1")
  print(anova1)
# (4)
  ff4 <- (sum(lm0$residuals^2) - sum(lm2$residuals^2))/
   2/((sum(lm2$residuals^2))/(nd-3))
  print("ff4")
  print(ff4)
# (5)
  pval4 <- 1- pf(ff4, df1=2, df=nd-3)
  print("pval4")
  print(pval4)
}
```

(1) Simulation data are generated. The simulation data of the first predictor are given as `xx1`, those of the second predictor as `xx2`, and those of the target variable as `yy`.
(2) `xx1`, `xx2`, and `yy` are organized in the data frame of `data1`. Then, `lm()` carries out the regression to obtain regression equation $y = a_0 + a_1x_1 + a_2x_2$. The result is stored as `lm2`. Next, `lm()` carries out regression to obtain regression equation of $y = a_0$. The result is stored in `lm0`.
(3) `anova()` conducts an F-test (that is to say, a variance analysis) for comparing `lm0` and `lm2`.
(4) F-value (Eq. (4.54) (page 178)) is calculated; the result is denoted by `ff4`.
(5) p-value corresponding to `ff4` is obtained using Eq. (4.56) (page 178).

`mul12()` outputs:

```
"anova1"
Analysis of Variance Table
Model 1: y ~ 1
Model 2: y ~ x1 + x2
    Res.Df \mathit{RSS}     Df Sum of Sq F       Pr(>F)
1   99     125.130
2   97     76.396   2  48.734    30.939 4.045e-11 ***
---
Signif. codes:  0'***'0.001'**'0.01'*'0.05'.'0.1' '1
"ff4"
30.93876
"pval4"
4.044831e-11
```

The value (`30.939`) on the row of `2` and the column of `F` in `anova1` is identical to `ff4`. The value (`4.045e-11`) on the row of `2` and the column of `Pr(>F)` in `anova1` is identical to `pval4`. `ff4` and `pval4` coincides with the content below yielded by `mul6()`.

```
F-statistic: 30.94 on 2 and 97 DF, p-value: 4.045e-11
```

R Program [4-5] End

4.3 Prediction Error on Multiple Regression

Predictors in statistics are chosen on the ground of prediction error in multiple regression in a manner similar to that of simple regression. One such predictor for giving prediction error is GCV (Generalized Cross-Validation). GCV is defined in a way similar to that for simple regression:

$$GCV = \frac{\sum_{i=1}^{n}(y_i - \hat{m}(x_{i1}, x_{i2}, \ldots, x_{iq}))^2}{n \cdot \left(1 - \frac{\sum_{i=1}^{n}[\mathbf{H}]_{ii}}{n}\right)^2}, \tag{4.57}$$

where $\hat{m}(x_{i1}, x_{i2}, \ldots, x_{iq})$ is the estimate corresponding to y_i (the result of the multiple regression equations). $\mathbf{H}$ is the hat matrix (Eq. (4.13) (page 164)). The value of $\sum_{i=1}^{n}[\mathbf{H}]_{ii}$ in multiple regression is obtained from Eq. (3.99) (page 151). The all-possible-regression procedure (all-possible-subsets-regression procedure) calculates statistics such as the GCV for all combinations of predictors and compares the resultant statistical values in selecting the best combination of predictors.

Two methods are available for choosing predictors of a regression equation such as a multiple regression equation: one is the method using a statistical test such as the F-test; the other is the method using statistics like GCV. Because methods using statistical tests do not check out all combinations of predictors, we have no certainty that the resultant regression equation is actually optimal despite the computational cost being relatively small. Furthermore, the procedure for proceeding with a selection of predictors varies with methodology such as the forward selection method, the backward selection method, and the forward and backward selection method. It can lead to differences in results. Conversely, the method to select a regression equation by minimizing statistics such as GCV assures the optimality of the chosen regression equation with respect to the standard of the criterion if the values of statistics, such as GCV, are calculated for all combinations of the predictors. Indeed, this method chooses the optimal regression equation in terms of a specific statistics by covering every conceivable regression equations. Although there are diverse statistics available for this purpose, selection of the statistics usually does not affect the result substantially. Therefore, if the computational capability permits, the all-possible-regression procedure is usually a preferable option; it calculates the values of statistics such as GCV for all $((2^q - 1))$ combinations of predictors. If this method is not practical because of heavy computational load, one idea is that predictors which are most likely needed are always selected. Alternatively, when correlation between two predictors is high, the number of applicant predictors is reduced by adopting one of these two predictors. It should be noted, however, that the regression equation given by the all-possible-regression procedure using statistics such as GCV may contain non-essential predictors; this possibility should not be neglected in some cases. This point is covered in the next section.

R Program [4-6]

The R program below constructs multiple regression equations for all combinations of predictors and chooses the optimal combination of predictors by GCV.

```
mul31()
  function ()
  {
  # (1)
    set.seed(45)
    np <- 4
    nd <- 50
    xxa <- runif(nd, min = -3, max = 4)
    xxb <- runif(nd, min = -2, max = 1)
    xxc <- runif(nd, min = 0, max = 4)
    xxd <- runif(nd, min = -4, max = -1)
    xxm <- cbind(xxa, xxb, xxc, xxd)
    yy <- -xxa *0.001 + xxb*2.1 -xxc *0.03 +
     xxd *3.3 + 4 + rnorm(nd, mean = 0, sd = 3)
  # (2)
    ns <- 2^np-1
    aint <- ns
    cd <- rep(0, length = np)
    cdm <- NULL
    for (aint in 1:ns){
      for(kk in 1:1E7){
        amod <- aint %% 2
        aint <- aint %/% 2
        cd[kk] <- amod
        if (aint==0) break
      }
    cd <- cd == 1
    cdm <- cbind(cdm, cd)
    }
    cdm <- cbind(cdm, rep(FALSE, length = np) )
  # (3)
    gcvt <- NULL
    for(ii in 1:ns){
      cd <- cdm[, ii]
      xxs <- xxm[, cd, drop = F]
      data1 <- data.frame(x = xxs, y = yy)
      lm.out <- lm(y~., data=data1)
      lev1 <- lm.influence(lm.out)$hat
      ey <- fitted(lm.out)
      fr <- sum(lev1)
      gcvt[ii] <- sum( (yy - ey)^2 )/(nd * (1 - fr /
      nd)^2)
    }
  # (4)
      gcvcon <- sum((yy-mean(yy))^2 )/
       (nd * (1 - 1/nd)^2)
      gcvt <- c(gcvt, gcvcon)
  # (5)
    bestcd <- cdm[, gcvt ==  min(gcvt)]
    if (is.vector(bestcd) == TRUE){
       nbest <- 1
       bestcd <- as.matrix(bestcd)
    }
    else
    {
      nbest <- dim(bestcd)[2]
    }
  # (6)
    for(kk in 1:nbest){
      bestcd2 <- bestcd[,kk]
      print( bestcd2 )
      if(all(bestcd2 == rep(FALSE, length = np))){
        print("function = constant")
        print(mean(yy))
      }
      else
      {
        xxt <- xxm[, bestcd2]
        data2 <- data.frame(x = xxt, y = yy)
        lm.out <- lm(y~., data = data2)
        print(lm.out$coefficients)
      }
    }
  }
```

(1) Simulation data are generated. As each of `xxa`, `xxb`, `xxc`, and `xxd` are data corresponding to one of the predictors, `xxm` obtained by combining these using `cbind()` is the design matrix. The values of the target variable is given as `yy`. The number of predictors is given as `np`. The number of data is given as `nd`.

(2) The number of all combinations of preditors (except the regression equation of a constant function) is saved as `ns`. `%%` yields a remainder of the division. `%/%` performs integer division in which the remainder (fractional part) is truncated. In this example, both `%%` and `%/%` treat positive values and hence the results follow intuitively. When negative values are dealt, however, be forewarned that `-14 %% 5` results in `1`, `-14 %% -5` results in `-4`, `-14 %/% 5` results in `-3`, and `-14 %/% -5` results in `2`, for example. All combinations of

the predictors, including the regression equation of a constant function, are described in `cdm`.

(3) The values of GCV are calculated as to all combinations of predictors obtained in (2) (the regression equation of a constant function is omitted). The results are stored in `gcvt`.

(4) The values of GCV for the regression equation of a constant function are stored in `gcvt`.

(5) The optimal combination of predictors which corresponds to the minimal element (or elements) of `gcvt` is selected. It is saved as `bestcd`. The number of the minimal element (or elements) of `gcvt` is saved as `nbest`. If `nbest` is 1, `bestcd` has a vector form and hence it is transformed into a matrix form.

(6) If the selected regression equation is the regression equation of a constant function, the constant (i.e., the average of the values of the target variable) is calculated and displayed. If the selected regression equation is not the regression equation of a constant function, a regression equation is constructed using `bestcd2`. The resultant regression equation is stored in `lm.out`. The regression coefficients of the regression equation are displayed.

The result of `mul31()` is:

```
FALSE  TRUE FALSE  TRUE
(Intercept)       x.xxb       x.xxd
   3.977132    1.966888    3.388904
```

The predictors, which are designed to have a large impact on the target variable when the simulation data were generated, are selected here.

R Program [4-6] End

4.4 Notes on Model Selection Using Prediction Error

When the selection of predictors using GCV in the manner of `mul31()` (page 185) are performed, we basically believe that there are functional relationships represented by a multiple linear regression between the predictors and the target variable. However, we have to think of the possibility that the predictors are chosen for use in the multiple linear regression even though the selected predictors actually do not have functional relationships with the target variable. The simulation for estimating this possibility is shown below.

The number of data is 100. All of the five predictors are realizations of uniform random numbers between 0 and 1. Values of the target variable are derived using the equation,

$$y_i = \epsilon_i, \tag{4.58}$$

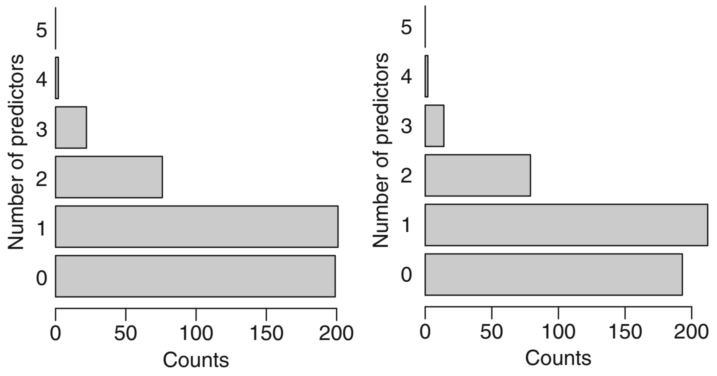


Fig. 4.1 Number of predictors selected by the all-possible-regression procedure using GCV when the predictors have no functional relationships with the target variable (*left*). Number of predictors selected by the forward and backward selection method using AIC (*right*)

where $\{\epsilon_i\}$ are realizations of $N(0, 2^2)$ (normal distribution; the mean is 0 and the variance is 2^2). Using these data, all-possible-regression procedure chooses predictors by GCV. The simulation was repeated 500 times by changing the initial value of the pseudo-random numbers. The result is shown in Fig. 4.1 (left). Although there are no functional relationships between the predictors and the target variable in these data, only 199 data sets resulted in a constant function as an appropriate regression equation while 301 data sets chose one or more predictors. With the same simulation data, the forward and backward selection method using Akaike's Information Criterion (AIC) were preformed. The result is Fig. 4.1 (right). 193 data sets selected a constant function. Therefore, we should not assume that the selection of a multiple regression equation other than a constant function by the all-possible-regression procedure using GCV directly prove the presence of functional relationships between the predictors and the target variable. A new method is clearly needed for identifying the existence of functional relationships [5].

R Program [4-7]

Using simulation data in which no relationships is conceivable between the predictors and the target variable, the R program below performs the forward and backward selection method using AIC for choosing the optimal combination of predictors.

```
mul511e()
  function ()
  {
  # (1)
    library(MASS)
  # (2)
    nd <- 100
    np <- 5
    nt <- 500
  # (3)
    bestp <- rep(0, np+1)
    for(kk in 1:nt){
  # (4)
      set.seed(1400 + kk * 8)
      xx1 <- runif(nd, min = 0, max = 1)
      xx2 <- runif(nd, min = 0, max = 1)
      xx3 <- runif(nd, min = 0, max = 1)
      xx4 <- runif(nd, min = 0, max = 1)
      xx5 <- runif(nd, min = 0, max = 1)
      xxm <- cbind(xx1, xx2, xx3, xx4,xx5)
      yy <- rnorm(nd, mean = 0, sd = 2)
  # (5)
      data1 <- data.frame(x = xxm, y = yy)
      lm1 <- lm(y~., data = data1)
      r1 <- stepAIC(lm1, trace = 0)
      fr <- length(r1$coef)
      bestp[fr] <- bestp[fr] + 1
    }
  # (6)
    par(mfrow = c(1, 1), mai = c(1.5, 1.5, 0.5, 0.5),
     oma = c(1, 1, 1, 1))
    barplot(bestp, xlab = "Counts", ylab = "Number of
     predictors", names.arg = as.character(seq(from = 0,
     to = np)), horiz = T, las = 1)
  }
```

(1) The use of package “MASS” is described.
(2) The number of data (`nd`), the number of predictor (`np`), and the number of times of simulation (`nt`) are given.
(3) A matrix form of `bestp` is prepared for storing the number of the selected matrix. Simulation is carried out `nt` times.
(4) The simulated values of predictors (`xx1`, `xx2`, `xx3`, `xx4`, and `xx4`) are given by uniform random numbers between 0 and 1. The values are organized in `xxm`. The values of the target variable are saved as `yy`.
(5) `lm()` derives a multiple linear regression with all predictors. The result is saved as `lm1`. `lm1` is entered in `stepAIC()` to select the predictors using the forward and backward selection method.
(6) `barplot()` draws a bar chart (bar graph). Setting of `horiz = T` specifies horizontal bars. Setting of `las = 1` indicates that the numbers of the scale along the abscissa axis are written horizontally. Setting of `las = 2` makes the numbers of the scale vertical.

R Program [4-7] End

4.5 Polynomial Regression

When the data $\{(x_i, y_i)\}$ $(1 \leq i \leq n)$ are given, polynomial regression derives the values of $\{a_j\}$ $(1 \leq j \leq q)$ by minimizing the value below.

$$RSS_{poly} = \sum_{i=1}^{n}(y_i - a_0 - \sum_{j=1}^{q} a_j x_i^j)^2. \tag{4.59}$$

This procedure gives the following regression equation:

$$y = \hat{a}_0 + \sum_{j=1}^{q} \hat{a}_j x^j. \tag{4.60}$$

Comparison with Eq. (4.2) (page 163) indicates that a multiple regression equation is transformed into a polynomial equation by altering x_{ij} into x_i^j. As the functional relationship between x and y can be other than $y = a_0$ and $y = a_0 + a_1 x$, the use of polynomial equations allows us to express diverse functional relationships.

Selection of the degrees of the polynomial equation (q in Eqs. (4.59) and (4.60)) can use a test of hypothesis. For example, when we select between $y = a_0$ and $y = a_0 + a_1 x$ as a regression equation, the null hypothesis and the alternative hypothesis are set as follows.

$$H_0 : a_1 = 0, \tag{4.61}$$

$$H_1 : a_1 \neq 0. \tag{4.62}$$

As this is the same as the null hypothesis ((Eq. 3.77) (page 131)) and the alternative hypothesis (Eq. (3.78) (page 131)) for simple regression, the procedure is identical to the t-test for simple regression or the F-test for it.

However, various forms of tests are possible for a polynomial equation. One of these is the following settings of the null hypothesis and the alternative hypothesis.

$$H_0 : a_2 = 0, \tag{4.63}$$

$$H_1 : a_2 \neq 0. \tag{4.64}$$

That is, these aim to compare a linear equation ($y = a_0 + a_1 x$) and a quadratic equation ($y = a_0 + a_1 x + a_2 x^2$). The corresponding F-value (Eq. (4.54) (page 178)) is

$$F(H_0, H_1) = \frac{\dfrac{RSS_{lin} - RSS_{quad}}{1}}{\dfrac{RSS_{quad}}{n-3}}, \tag{4.65}$$

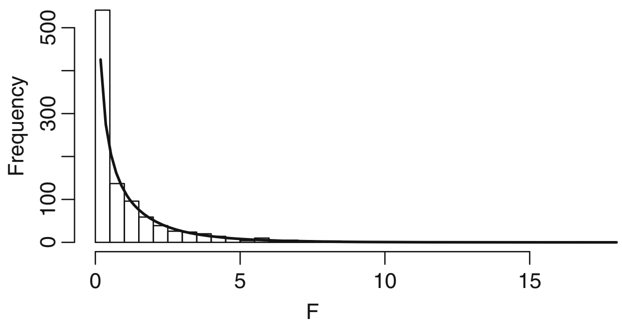


Fig. 4.2 Distribution of F-values given by Eq. (4.65) and the probability density function of F-distribution with the first degrees of freedom of 1 and the second degrees of freedom of $(40 - 3)$

where RSS_{lin} is the residual sum of squares given by a linear equation whereas RSS_{quad} is the residual sum of squares given by a quadratic equation.

The simulation to be described below shows that when the null hypothesis (Eq. (4.63)) holds, the F-value (Eq. (4.65)) obeys

$$F(H_0, H_1) \sim F_{1,n-3}. \tag{4.66}$$

First, values of the target variable for 40 data are generated using

$$y_i = 3 - 4x_i + \epsilon_i, \tag{4.67}$$

where $\{\epsilon_i\}$ $(1 \leq i \leq n)$ are realizations of $N(0, 0.3^2)$ (normal distribution; the mean is 0 and the variance is 0.3^2). The values of the predictors ($\{x_i\}$ $(1 \leq i \leq n)$) are given as $\{1, 2, \ldots, 40\}$. Next, values of RSS_{lin} and RSS_{quad} are calculated, and as a result, the value of $F(H_0, H_1)$ (Eq. (4.65)) are obtained. The distribution of $F(H_0, H_1)$ calculated using $1,000$ sets of simulation data given by changing the initial value of pseudo-random numbers is drawn in Fig. 4.2. The probability density function of the F-distribution with the first degrees of freedom (1) and the second degrees of freedom $(40 - 3)$ is also illustrated.

Alternatively, other settings of the null hypothesis and the alternative hypothesis from Eqs. (4.63) (page 190) and (4.64) (page 190) are possible. For example, the following setting is one such choice:

$$H_0 : a_1 = 0 \quad \text{and} \quad a_2 = 0, \tag{4.68}$$

$$H_1 : a_1 \neq 0 \quad \text{or} \quad a_2 \neq 0. \tag{4.69}$$

That is, a zero degree polynomial equation (a constant function) ($y = a_0$) and a quadratic equation ($y = a_0 + a_1 x + a_2 x^2$) are compared. The F-value (Eq. (4.54) (page 178)) for this comparison is

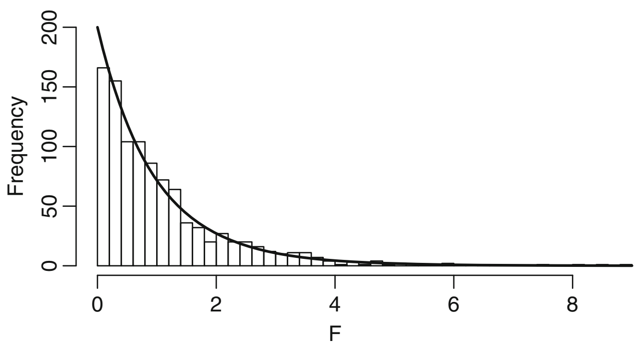


Fig. 4.3 Distribution of F-values given by Eq. (4.70) and the probability density function of F-distribution with the first degrees of freedom of 2 and the second degrees of freedom of (40 − 3)

$$F(H_0, H_1) = \frac{\dfrac{RSS_{const} - RSS_{quad}}{2}}{\dfrac{RSS_{quad}}{n-3}}, \tag{4.70}$$

where RSS_{const} is the residual sum of squares given by a zero degree polynomial equation, whereas RSS_{quad} is the residual sum of squares given by a quadratic equation as defined above.

Let us perform the simulation that confirms the F-value (Eq. (4.70)) follows

$$F(H_0, H_1) \sim F_{1,n-3}, \tag{4.71}$$

when the null hypothesis (Eq. (4.68)) holds. First, the values of the target variable for 40 data are generated using the following equation.

$$y_i = -14 + \epsilon_i, \tag{4.72}$$

where $\{\epsilon_i\}$ $(1 \le i \le n)$ are realizations of $N(0, 0.3^2)$ (normal distribution; the mean is 0 and the variance is 0.3^2). The values of the predictors ($\{x_i\}$ $(1 \le i \le n)$) are set as $\{1, 2, \ldots, 40\}$. Next, the values of RSS_{const} and RSS_{quad} are calculated. Using these values, the value of $F(H_0, H_1)$ (Eq. (4.70)) is derived. The distribution of $F(H_0, H_1)$ calculated using $1,000$ sets of simulation data given by changing the initial value of pseudo-random numbers is drawn in Fig. 4.3. The constant-multiplied probability density function of the F-distribution with the first degree of freedom (2) and the second degree of freedom (40 − 3) is superimposed.

However, the selection of the degrees of a polynomial equation is based on the following:

(1) The F-test (i.e., the test by Eqs. (4.61) (page 190) and (4.62) (page 190)) is conducted to compare the zero degree polynomial equation (a constant function) and the linear equation.

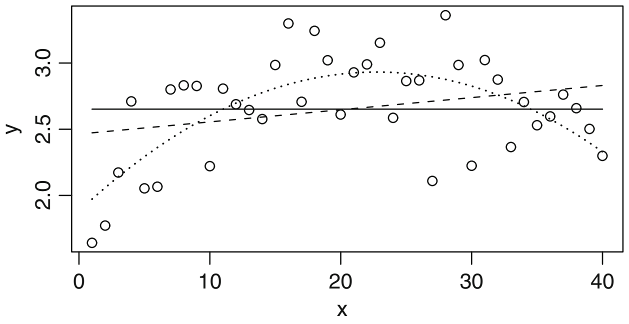


Fig. 4.4 Polynomial equations are fitted to the 40 data (*open circle*). The *solid line* indicates the zero degree polynomial equation ($y = a_0$). The *dashed line* indicate a linear equation ($y = a_0 + a_1x$, the p-value, obtained by comparing with the zero degree polynomial equation, is 0.09144). The *dotted line* indicates the quadratic equations ($y = a_0 + a_1x + a_2x^2$, the p-value, similarly obtained, is $1.399 \cdot 10^{-5}$)

(2) If the linear equation is selected, the F-test (i.e., the test given by Eqs. (4.63) (page 190) and (4.64) (page 190)) is conducted to compare the linear equation and the quadratic equation.

That is, the degree of the polynomial equation is increased one by one. A test to compare a polynomial equation with another of two or more degrees (i.e., using Eqs. (4.68) (page 191) and (4.69) (page 191)) is seldom performed.

Nevertheless, the simulation below shows that a method involving increasing the degrees of the polynomial equation one by one possibly results in an unlikely optimal polynomial equation. Figure 4.4 illustrates the result of fitting a zero degree polynomial equation, a linear equation, and a quadratic equation to the 40 data using least squares. The F-value based on the null hypothesis "$H_0 : y = a_0$" and the alternative hypothesis "$H_1 : y = a_0 + a_1x$" is calculated using Eq. (3.88) (page 139) and the corresponding p-value is calculated to be 0.09144. Therefore, we cannot deny the hypothesis that $a_1 = 0$ should be used as a regression equation because the null hypothesis is not rejected. Alternatively, if we assume the null hypothesis "$H_0 : y = a_0$" and the alternative hypothesis "$H_1 : y = a_0 + a_1x + a_2x^2$", the F-value (Eq. (4.70) (page 192)) can be calculated and the corresponding p-value found to be $1.399 \cdot 10^{-5}$. That is, the null hypothesis is rejected. We conclude that the quadratic equation should be employed.

As seen above, although the null hypothesis "$y = a_0$ should be used" is rejected when a specific alternative hypothesis is assigned, this null hypothesis is not rejected when another alternative hypothesis is assigned. Let us suppose, therefore, that we adopt the policy that although we should use a regression equation with as small number of regression coefficients as possible, if the comparison between null hypothesis "a regression equation with a small number of regression coefficients should be employed" and alternative hypothesis "a regression equation with a larger number of regression coefficients should be employed" rejects the null hypothesis, the alternative hypothesis should be adopted. If the test with the null hypothesis $y = a_0$ and the alternative hypothesis $y = a_0 + a_1x$ is conducted, we conclude that $y = a_0$ should be employed because the null hypothesis is not rejected. Hence, we need not examine the regression equation $y = a_0 + a_1x + a_2x^2$, which contains a larger number of regression coefficients than $y = a_0 + a_1x$. This means that we usually pay no attention to the truth that use of the null hypothesis $y = a_0$ and the alternative hypothesis $y = a_0 + a_1x + a_2x^2$ leads to the conclusion that $y = a_0 + a_1x + a_2x^2$ should be employed. The method to select a regression equation that minimizes GCV is desirable in this respect.

R Program [4-8]

The degree of the polynomial equation is selected using the F-test.

```
poly41()
  function ()
  {
  # (1)
    par(mai = c(1, 1, 1, 1), omi = c(0, 0, 0, 0))
  #(2)
    set.seed(815)
    nd <- 40
    xx <- seq(from=1, to=nd, by=1)
    yy <- sin(xx*0.07) + 2 + rnorm(nd, mean=0, sd=0.3)
  # (3)
    data1 <- data.frame(x=xx, y=yy)
    lm0 <- lm(y~1, data=data1)
    lm1 <- lm(y~x, data=data1)
    lm2 <- lm(y~poly(x, degree=2), data=data1)
  # (4)
    anova1 <- anova(lm0, lm1)
    print("----- anova(lm0, lm1) -----")
    print(anova1)
    anova2 <- anova(lm0, lm2)
    print("----- anova(lm0, lm2) -----")
    print(anova2)
  # (5)
    ff1 <- (sum(lm0$residuals^2) - sum(lm1$residuals^2
     ))/((sum(lm1$residuals^2))/(nd-2))
    pval1 <- 1 - pf(ff1, df1=1, df2=nd-2)
    print("pval1")
    print(pval1)
  # (6)
    ff2 <- (sum(lm0$residuals^2) - sum(lm2$residuals^2
     ))/2/((sum(lm2$residuals^2))/(nd-3))
    pval2 <- 1 - pf(ff2, df1=2, df2=nd-3)
    print("pval2")
    print(pval2)
  # (7)
    plot(xx, yy, xlab="x", ylab="y")
    lines(xx, lm0$fitted, lty=1)
    lines(xx, lm1$fitted, lty=2)
    lines(xx, lm2$fitted, lty=3)
  }
```

(1) `par()` sets the graphics area.
(2) The number of data (`nd`) is given. The values of the predictors of the simulation data are given as `xx`. Those of the target variable are given as `yy`.
(3) `xx` and `yy` are organized in a data frame of `data1`. `xx` is called `x` and `yy` is called `y` in the data frame. Next, the result of the regression to the zero degree regression equation using `lm()` is stored in `lm0`. The result of the simple regression using `lm()` is stored in `lm1`. The result of the quadratic regression using `lm()` is stored in `lm2`. `poly(x, degree=2)` stands for a quadratic equation. When a polynomial equation is specified in `lm()`, `poly()` specifies the use of an orthogonal polynomial to increase the accuracy of the calculation.
(4) `anova(lm0, lm1)` carries out an analysis of variance for comparing the result of the regression to the zero degree regression equation and that of simple regression. The result is stored in `anova1` and is output. `anova(lm0, lm2)` carries out analysis of variance for comparing the result of the regression to the zero regression equation and that of quadratic regression. The result is stored in `anova2` and is output.
(5) The F-value for comparing the result of the regression to the zero degree regression equation and that of simple regression are saved as `ff1`. The p-value is calculated using `ff1`. The result is saved as `pval1` and is displayed.
(6) The F-value for comparing the result of the regression to the zero degree regression equation and that of quadratic regression are saved as `ff2`. The p-value is calculated using `ff2`. The result is saved as `pval2` and is displayed.
(7) The original data, the estimates given by the regression to the zero degree regression equation, those given by simple regression, and those given by quadratic regression are drawn in a graph.

The result of `poly41()` is:

```
[1] "----- anova(lm0, lm1) -----"
Analysis of Variance Table
Model 1: y ~ 1
Model 2: y ~ x
   Res.Df  \mathit{RSS}     Df  Sum of Sq  F       Pr(>F)
1  39      6.1290
2  38      5.6807  1   0.4483     2.9988 0.09144 .
---
Signif. codes: 0'***'0.001'**'0.01'*'0.05'.'0.1' '1
```

```
[1] "----- anova(lm0, lm2) -----"
Analysis of Variance Table
Model 1: y ~ 1
Model 2: y ~ poly(x, degree = 2)
   Res.Df \mathit{RSS}     Df Sum of Sq F       Pr(>F)
1  39     6.1290
2  37     3.3496  2  2.7794    15.351 1.399e-05 ***
---
Signif. codes: 0'***'0.001'**'0.01'*'0.05'.'0.1' '1
[1] "pval1"
[1] 0.09143803
[1] "pval2"
[1] 1.398627e-05
```

`poly41()` also outputs Fig. 4.4 (page 193).

R Program [4-8] End

R Program [4-9]

When the null hypothesis is "$y = a_0$" and the alternative hypothesis is "$y = a_0+a_1x+a_2x^2$", the distribution of the F-values (Eq. (4.70) (page 192)) is graphed.

`poly57e()`

```
function ()
{
# (1)
  set.seed(815)
  nd <- 40
  nt <- 1000
# (2)
  ff2 <- NULL
  for (jj in 1:nt){
# (3)
    xx <- seq(from = 1, to = nd, by = 1)
    yy <-  -14 + rnorm(nd, mean = 0, sd = 0.3)
# (4)
    data1 <- data.frame(x = xx, y = yy)
    lm0 <- lm(y~1, data = data1)
    lm2 <- lm(y~poly(x, degree = 2), data = data1)
# (5)
    ff2[jj] <- (sum(lm0$residuals^2) -
     sum(lm2$residuals^2)) / 2 /
     ((sum(lm2$residuals^2))/(nd - 3))
  }
# (6)
  par(mfrow = c(1,1), mai = c(2, 1, 1.5, 0.1),
   omi = c(0, 0, 0, 0))
# (7)
  br1 <-  pretty(ff2, n = 40)
  bw1 <- br1[2] - br1[1]
  ff2h <- floor(ff2 / bw1) * bw1 + 0.01 * bw1
  hist1 <- hist(ff2h, breaks = br1, main = "",
   xlab = "F", ylab = "Frequency", ylim = c(0, 200))
# (8)
  xxa <- seq(from = min(xx), to = max(xx),
   length = 1000)
  curve(df(x, df1 = 2, df2 = nd - 3) * bw1 * nt,
   min(br1), max(br1), xlab = "", ylab = "p(x)",
   lwd = 2, xlim = c(min(br1), max(br1)),
   ylim = c(0, max(hist1$counts)), add = T)
}
```

(1) The initial value of the pseudo-random numbers is given. The number of data (`nd`) is set. The number of times of the simulation (`nt`) is specified.
(2) `ff2` is prepared for storing the F-values. Sampling is conducted `nt` times.
(3) Simulation data are generated. The values of the predictors of the simulation data are stored in `xx`. Those of the target values are stored in `yy`.
(4) `xx` and `yy` are organized in the data frame of `data1`. `xx` is called `x` and `yy` is called `y` in the data frame. Then, the result of the regression to a zero degree polynomial equation (a constant function) using `lm()` is stored in `lm0`. The result of regression to the quadratic equation using `lm()` is stored in `lm2`.
(5) The F-value is calculated using Eq. (4.70) (page 192). The result is saved as `ff2`.
(6) `par()` sets the graphics area.
(7) `hist()` draws the histogram of `ff2`.
(8) `curve()` superimposes the graph of the probability density function of the F-distribution with the first degree of freedom of 5 and the second degree of freedom of (40 − 6). `df(x, df1 = 2, df2 = nd-3)` gives the values of the probability density function of the F-distribution. `bw1 * nt` is the constant to assist the comparison with the histogram. `poly57e()` outputs Fig. 4.3.

R Program [4-9] End

4.6 Variance of Regression Coefficient and Multicollinearity

One of the things that should be taken into consideration when constructing a multiple regression equation is multicollinearity. This situation occurs when the predictors are not fully independent of one another. The term "multicollinearity" combines "multi-" implying large number and "-collinearity" meaning linear dependency (page 123 in [2]).

To understand the properties concerning multicollinearity, the following $\{x'_{ij}\}$ $(1 \le i \le n, 1 \le j \le q)$ are used instead of $\{x_{ij}\}$ $(1 \le i \le n, 1 \le j \le q)$; this transformation is similar to Eq. (4.15) (page 165),

$$\begin{aligned} x'_{ij} &= x_{ij} - \frac{\sum_{i=1}^{n} x_{ij}}{n} \qquad (1 \le i \le n, 1 \le j \le q), \\ y'_i &= y_i - \frac{\sum_{i=1}^{n} y_i}{n} \qquad (1 \le i \le n). \end{aligned} \tag{4.73}$$

Note, the summation of $\{x'_{ij}\}$ with respect to i gives 0. Similarly summation of $\{y'_{ij}\}$ with respect to i also gives 0.

The definition of the multiple regression equation yields the following equation:

$$\begin{aligned} y_i &= \hat{a}_0 + \sum_{j=1}^{q} \hat{a}_j x_{ij} + e_i \\ &= \hat{y}_i + e_i, \end{aligned} \tag{4.74}$$

where $\{e_i\}$ $(1 \le i \le n)$ are residuals (refer to Eq. (4.1) (page 163) and the explanations below the equation). Summing Eq. (4.74) with respect to i and dividing by n gives

$$\hat{a}_0 = \frac{1}{n}\sum_{i=1}^{n} y_i - \frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{q} \hat{a}_j x_{ij}, \tag{4.75}$$

where Eq. (4.14) (page 165) is used. The term $\frac{1}{n}\sum_{i=1}^{n} y_i$ is subtracted from the both sides of Eq. (4.74) giving the result:

$$\begin{aligned} y_i - \frac{1}{n}\sum_{i=1}^{n} y_i &= \hat{a}_0 + \sum_{j=1}^{q} \hat{a}_j x_{ij} - \frac{1}{n}\sum_{i=1}^{n} y_i + e_i \\ &= \hat{a}_0 + \sum_{j=1}^{q} \hat{a}_j x_{ij} - \frac{1}{n}\sum_{i=1}^{n}(\hat{a}_0 + \sum_{j=1}^{q} \hat{a}_j x_{ij} + e_i) + e_i \\ &= \hat{a}_0 + \sum_{j=1}^{q} \hat{a}_j x_{ij} - \hat{a}_0 - \frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{q} \hat{a}_j x_{ij} + e_i \end{aligned}$$

$$= \sum_{j=1}^{q} \hat{a}_j x_{ij} - \frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{q} \hat{a}_j x_{ij} + e_i$$

$$= \sum_{j=1}^{q} \hat{a}_j \left(x_{ij} - \frac{1}{n} \sum_{i=1}^{n} x_{ij} \right) + e_i. \tag{4.76}$$

In deriving the third line from the second, $\sum_{i=1}^{n} e_i = 0$ is used. Then, by Eq. (4.73), we obtain

$$y'_i = \sum_{j=1}^{q} \hat{a}_j x'_{ij} + e_i. \tag{4.77}$$

Hence, when a regression equation expressed in terms of $\{x'_{ij}\}$ $(1 \le i \le n, 1 \le i \le q)$ and $\{y'_i\}$ $(1 \le i \le n)$ is compared with that given by $\{x_{ij}\}$ and $\{y_i\}$, we find that the regression coefficients, with the exception of the constant term of the regression equation, are the same whereas the constant term derived from $\{x'_{ij}\}$ and $\{y'_i\}$ is 0. Thus, the properties of the regression equation given by $\{x'_{ij}\}$ and $\{y'_i\}$ are identical to those given by $\{x_{ij}\}$ and $\{y_i\}$.

Therefore, the interpretation and properties of multicollinearity is shown below using $\{x'_{ij}\}$ and $\{y'_i\}$. First, linear dependence of $\{\mathbf{x}'_j\}$ $(1 \le j \le q)$ is defined by:

$$\sum_{j=1}^{q} c_j \mathbf{x}'_j = \mathbf{0}, \tag{4.78}$$

where $\mathbf{x}'_j$ $(1 \le j \le q)$ is

$$\mathbf{x}'_j = \begin{pmatrix} x'_{1j} \\ x'_{2j} \\ \vdots \\ x'_{nj} \end{pmatrix} \quad (1 \le j \le q). \tag{4.79}$$

Furthermore, one or more of the $\{c_j\}$ $(1 \le j \le q)$ are nonzero. Based on the properties of linear independency in linear algebra, Eq. (4.78) indicates that $\{\mathbf{x}'_j\}$ $(1 \le j \le q)$ are linearly dependent.

Equation (4.78) is rewritten as

$$\mathbf{X}'\mathbf{c} = \mathbf{0}, \tag{4.80}$$

where $\mathbf{X}'$ is defined as

$$\mathbf{X}' = \begin{pmatrix} x_{11} & x_{12} & \ldots & x_{1q} \\ x_{21} & x_{22} & \ldots & x_{2q} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \ldots & x_{nq} \end{pmatrix}, \tag{4.81}$$

and **c** is a column vector defined as

$$\mathbf{c} = \begin{pmatrix} c_1 \\ c_2 \\ \vdots \\ c_q \end{pmatrix}. \tag{4.82}$$

Multiplying Eq. (4.80) by $\mathbf{X}'^t$ from the left results in

$$\mathbf{X}'^t\mathbf{X}'\mathbf{c} = \mathbf{0}. \tag{4.83}$$

As $\mathbf{X}'^t\mathbf{X}'$ is a square matrix, then under the condition that at least one of the $\{c_j\}$ $(1 \leq j \leq q)$ is nonzero, Eq. (4.83) is equivalent to the following equation.

$$|\mathbf{X}'^t\mathbf{X}'| = 0, \tag{4.84}$$

where $|\mathbf{X}'^t\mathbf{X}'|$ is the determinant (Eq. (1.60) (page 36)) of $\mathbf{X}'^t\mathbf{X}'$. That is, when Eq. (4.78) holds, the regression coefficients ($\{\hat{a}_1, \hat{a}_2, \ldots, \hat{a}_q\}$) cannot be obtained using the following equation (refer to Eq. (4.9) (page 164)) because $(\mathbf{X}'^t\mathbf{X}')^{-1}$ is not unique:

$$\hat{\mathbf{a}}' = (\mathbf{X}'^t\mathbf{X}')^{-1}\mathbf{X}'^t\mathbf{y}', \tag{4.85}$$

where $\hat{\mathbf{a}}'$ and $\mathbf{y}'$ are defined as

$$\hat{\mathbf{a}}' = \begin{pmatrix} \hat{a}_1 \\ \hat{a}_2 \\ \vdots \\ \hat{a}_q, \end{pmatrix}, \quad \mathbf{y}' = \begin{pmatrix} \hat{y}'_1 \\ \hat{y}'_2 \\ \hat{y}'_3 \\ \vdots \\ \hat{y}'_n \end{pmatrix}. \tag{4.86}$$

Conversely, when there are multiple regression equations that minimize Eq. (4.1) (page 163), Eq. (4.78) (page 199) is satisfied. Then, for the two sets of values of the regression coefficients here denoted by $\hat{\mathbf{a}}'^{(1)}$ and $\hat{\mathbf{a}}'^{(2)}$ ($\hat{\mathbf{a}}'^{(1)} \neq \hat{\mathbf{a}}'^{(2)}$), we have

$$\| \mathbf{y}' - \mathbf{X}'\hat{\mathbf{a}}'^{(1)} \|^2 = \| \mathbf{y}' - \mathbf{X}'\hat{\mathbf{a}}'^{(2)} \|^2 . \tag{4.87}$$

This equation indicates:

$$\begin{aligned}
&\parallel \mathbf{y}' - \mathbf{X}'\hat{\mathbf{a}}'^{(1)} \parallel^2 - \parallel \mathbf{y}' - \mathbf{X}'\hat{\mathbf{a}}'^{(2)} \parallel^2 \\
&= (\mathbf{y}' - \mathbf{X}'\hat{\mathbf{a}}'^{(1)t})(\mathbf{y}' - \mathbf{X}'\hat{\mathbf{a}}'^{(1)}) - (\mathbf{y}' - \mathbf{X}'\hat{\mathbf{a}}'^{(2)})^t(\mathbf{y}' - \mathbf{X}'\hat{\mathbf{a}}'^{(2)}) \\
&= -\hat{\mathbf{a}}'^{(1)t}\mathbf{X}'^t\mathbf{y}' - \mathbf{y}'^t\mathbf{X}'\hat{\mathbf{a}}'^{(1)} + \hat{\mathbf{a}}'^{(1)t}\mathbf{X}'^t\mathbf{X}'\hat{\mathbf{a}}'^{(1)t} + \hat{\mathbf{a}}'^{(2)t}\mathbf{X}'^t\mathbf{y}' \\
&+ \mathbf{y}'^t\mathbf{X}'\hat{\mathbf{a}}'^{(2)} - \hat{\mathbf{a}}'^{(2)t}\mathbf{X}'^t\mathbf{X}'\hat{\mathbf{a}}'^{(2)} \\
&= -\hat{\mathbf{a}}'^{(1)t}\mathbf{X}'^t\mathbf{X}'\hat{\mathbf{a}}'^{(1)} - \mathbf{y}'^t\mathbf{X}'\hat{\mathbf{a}}'^{(1)t} + \hat{\mathbf{a}}'^{(1)t}\mathbf{X}'^t\mathbf{X}'\hat{\mathbf{a}}'^{(1)} + \hat{\mathbf{a}}'^{(2)t}\mathbf{X}'^t\mathbf{X}\hat{\mathbf{a}}'^{(2)} \\
&\quad + \mathbf{y}'^t\mathbf{X}'\hat{\mathbf{a}}'^{(2)} - \hat{\mathbf{a}}'^{(2)t}\mathbf{X}'^t\mathbf{X}'\hat{\mathbf{a}}'^{(2)} \\
&= -\hat{\mathbf{a}}'^{(1)t}\mathbf{X}'^t\mathbf{X}'\hat{\mathbf{a}}'^{(2)} - \mathbf{y}'^t\mathbf{X}'\hat{\mathbf{a}}'^{(1)} + \hat{\mathbf{a}}'^{(1)t}\mathbf{X}'^t\mathbf{X}'\hat{\mathbf{a}}'^{(1)} + \mathbf{y}'^t\mathbf{X}'\hat{\mathbf{a}}'^{(2)} \\
&= -\hat{\mathbf{a}}'^{(1)t}\mathbf{X}'^t\mathbf{X}'\hat{\mathbf{a}}'^{(2)} - \hat{\mathbf{a}}'^{(2)t}\mathbf{X}'^t\mathbf{X}'\hat{\mathbf{a}}'^{(1)} + \hat{\mathbf{a}}'^{(1)t}\mathbf{X}'^t\mathbf{X}'\hat{\mathbf{a}}'^{(1)} + \hat{\mathbf{a}}'^{(2)t}\mathbf{X}'^t\mathbf{X}'\hat{\mathbf{a}}'^{(2)} \\
&= \parallel \mathbf{X}'(\hat{\mathbf{a}}'^{(2)} - \hat{\mathbf{a}}'^{(1)}) \parallel^2 \\
&= 0,
\end{aligned} \tag{4.88}$$

where $\parallel \; \parallel$ represents the Euclidean length of a vector (the square root of the sum of squares of the elements of a vector). The following two equations, which are obtained in a way similar to that of Eq. (4.8) (page 164), are used in deriving line eight from line seven:

$$\mathbf{X}'^t\mathbf{y}' = \mathbf{X}'^t\mathbf{X}'\hat{\mathbf{a}}'^{(1)}, \qquad \mathbf{X}'^t\mathbf{y}' = \mathbf{X}'^t\mathbf{X}'\hat{\mathbf{a}}'^{(2)}. \tag{4.89}$$

The equality of Eq. (4.88) ($\parallel \mathbf{X}'(\hat{\mathbf{a}}'^{(2)} - \hat{\mathbf{a}}'^{(1)}) \parallel^2 = 0$) is equivalent to that of the equation below.

$$\mathbf{X}'(\hat{\mathbf{a}}'^{(2)} - \hat{\mathbf{a}}'^{(1)}) = \mathbf{0}. \tag{4.90}$$

As $\hat{\mathbf{a}}'^{(1)} \neq \hat{\mathbf{a}}'^{(2)}$ holds, the column vectors constituting $\mathbf{X}$ are linearly dependent (Eq. (4.80) (page 199)). Thus, it is proved that when Eq. (4.82) (page 200) holds, multicollinearity (Eq. (4.78) (page 199)) exists.

In the context of multiple regression equations, however, when Eq. (4.78) holds approximately, we say that the predictors have multicollinearity. That is, when the following equation holds approximately, the data has multicollinearity.

$$\sum_{j=1}^{q} c_j \mathbf{x}'_j \approx \mathbf{0}. \tag{4.91}$$

This approximate equation is equivalent to

$$|\mathbf{X}'^t\mathbf{X}'| \approx 0. \tag{4.92}$$

One problem that multicollinearity of a multiple regression equation causes is that it augments the variances of the regression coefficients. Therefore, if multicollinearity exists, slight variations in errors contained in the values of the target variables in data can change the values of the regression coefficients significantly. Hence, statistics are developed to quantify these variations:

$$\frac{E[(\hat{\mathbf{a}}' - \mathbf{a}')^t(\hat{\mathbf{a}}' - \mathbf{a}')]}{\sigma^2} = \frac{\sum_{j=0}^{q} E[(\hat{a}_j - a_j)^2]}{\sigma^2}, \tag{4.93}$$

where $\mathbf{a}'$ is defined as

$$\mathbf{a}' = \begin{pmatrix} a_1 \\ a_2 \\ \vdots \\ a_q \end{pmatrix}. \tag{4.94}$$

That is, $E[(\hat{\mathbf{a}}' - \mathbf{a}')^t(\hat{\mathbf{a}}' - \mathbf{a}')]$ represents the expectation of the sum of the variances of q regression coefficients. Equation (4.42) (page 176) yields

$$\mathbf{C}' = \sigma^2(\mathbf{X}''^t\mathbf{X}')^{-1}, \tag{4.95}$$

where $\mathbf{C}'$ is the variance-covariance matrix of $\hat{\mathbf{a}}'$, which is written in the form:

$$\mathbf{C}' = \begin{pmatrix} E\Big(\big(\hat{a}_1 - E(\hat{a}_1)\big)\big(\hat{a}_1 - E(\hat{a}_1)\big)\Big) & \ldots & E\Big(\big(\hat{a}_1 - E(\hat{a}_1)\big)\big(\hat{a}_q - E(\hat{a}_q)\big)\Big) \\ \vdots & \ddots & \vdots \\ E\Big(\big(\hat{a}_q - E(\hat{a}_q)\big)\big(\hat{a}_1 - E(\hat{a}_1)\big)\Big) & \ldots & E\Big(\big(\hat{a}_q - E(\hat{a}_q)\big)\big(\hat{a}_q - E(\hat{a}_q)\big)\Big) \end{pmatrix}. \tag{4.96}$$

Hence, the ij-element of $\mathbf{C}'$ is $E[(\hat{a}_i - a_i)(\hat{a}_j - a_j)]$. Then, using Eq. (4.95), the variance of $\hat{a}_j$ ($E[(\hat{a}_j - a_j)(\hat{a}_j - a_j)]$) is obtained. However, this does not show the direct relationship between multicollinearity (Eq. (4.92) (page 201)) and the variances of the regression coefficients. Then, using Eq. (4.94), Eq. (4.93) is rewritten as

$$\frac{E[(\hat{\mathbf{a}}' - \mathbf{a}')^t(\hat{\mathbf{a}}' - \mathbf{a}')]}{\sigma^2} = \mathrm{trace}((\mathbf{X}''^t\mathbf{X}')^{-1}). \tag{4.97}$$

As $\mathbf{X}''^t\mathbf{X}'$ is a symmetric matrix, there exists an orthogonal matrix $\mathbf{U}$ that performs diagonalization in such a way that

$$(\mathbf{X}''^t\mathbf{X}')^{-1} = (\mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^{-1})^{-1}$$

$$
\begin{aligned}
&= (\mathbf{U}^{-1})^{-1}(\mathbf{U}\boldsymbol{\Lambda})^{-1} \\
&= \mathbf{U}\boldsymbol{\Lambda}^{-1}\mathbf{U}^{-1}, \qquad (4.98)
\end{aligned}
$$

where $\boldsymbol{\Lambda}$ is given as

$$
\boldsymbol{\Lambda} = \begin{pmatrix} \lambda_1 & 0 & 0 \ldots & 0 \\ 0 & \lambda_2 & 0 \ldots & 0 \\ \vdots & \vdots & \vdots \ddots & \vdots \\ 0 & 0 & 0 \ldots & \lambda_q \end{pmatrix}. \qquad (4.99)
$$

As $\boldsymbol{\Lambda}$ is a diagonal matrix, $\boldsymbol{\Lambda}^{-1}$ is represented as

$$
\boldsymbol{\Lambda}^{-1} = \begin{pmatrix} \frac{1}{\lambda_1} & 0 & 0 \ldots & 0 \\ 0 & \frac{1}{\lambda_2} & 0 \ldots & 0 \\ \vdots & \vdots & \vdots \ddots & \vdots \\ 0 & 0 & 0 \ldots & \frac{1}{\lambda_q} \end{pmatrix}. \qquad (4.100)
$$

Thus, the trace of $(\mathbf{X}''^t\mathbf{X}')^{-1}$ is as follows:

$$
\begin{aligned}
\mathrm{trace}((\mathbf{X}''^t\mathbf{X}')^{-1}) &= \mathrm{trace}(\mathbf{U}\boldsymbol{\Lambda}^{-1}\mathbf{U}^{-1}) \\
&= \sum_{i=1}^{q}\sum_{j=1}^{q}\sum_{k=1}^{q}[\mathbf{U}]_{ij}[\boldsymbol{\Lambda}^{-1}]_{jk}[\mathbf{U}^{-1}]_{ki} \\
&= \sum_{i=1}^{q}\sum_{j=1}^{q}[\mathbf{U}]_{ij}[\boldsymbol{\Lambda}^{-1}]_{jj}[\mathbf{U}^{-1}]_{ji} \\
&= \sum_{i=1}^{q}\sum_{j=1}^{q}[\mathbf{U}]_{ij}[\mathbf{U}^{-1}]_{ji}[\boldsymbol{\Lambda}^{-1}]_{jj} \\
&= \sum_{i=1}^{q}\sum_{j=1}^{q}\delta_{ij}[\boldsymbol{\Lambda}^{-1}]_{jj} \\
&= \sum_{i=1}^{q}[\boldsymbol{\Lambda}^{-1}]_{ii} \\
&= \sum_{i=1}^{q}\frac{1}{\lambda_i}. \qquad (4.101)
\end{aligned}
$$

Therefore, we have

$$\frac{E((\hat{\mathbf{a}}' - \mathbf{a}')^t(\hat{\mathbf{a}}' - \mathbf{a}'))}{\sigma^2} = \sum_{j=1}^{q} \frac{1}{\lambda_j}. \tag{4.102}$$

This equation gives the sum of the variances of the regression coefficients with the exception of the constant term of the regression equation. Equation (4.45) (page 177) shows that all of $\{\lambda_j\}$ $(1 \leq j \leq q)$ are positive. Therefore, when the sum of the variances of the regression coefficients (not including the constant term) is small, one or more among the $\{\lambda_j\}$ take small positive values. Implications of this finding are shown by deriving the equation below from Eq. (4.98) (page 202).

$$\begin{aligned}
|\mathbf{X}'^t\mathbf{X}'| &= |\mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^{-1}| \\
&= |\mathbf{U}||\boldsymbol{\Lambda}||\mathbf{U}^{-1}| \\
&= |\mathbf{U}\mathbf{U}^{-1}||\boldsymbol{\Lambda}| \\
&= |\boldsymbol{\Lambda}| \\
&= \prod_{j=1}^{q} \lambda_j,
\end{aligned} \tag{4.103}$$

where δ_{ij} is the Kronecker delta defined as

$$\delta_{ij} = \begin{cases} 1 \text{ if } i = j \\ 0 \text{ if } i \neq j. \end{cases} \tag{4.104}$$

Hence, when Eq. (4.92) (page 201) holds, one or more among $\{\lambda_j\}$ takes small positive values. Therefore, when multicollinearity is present, variances of one or more regression coefficients among $\{\hat{a}_j\}$ $(1 \leq j \leq q)$ are large.

However, it should be noted that Eq. (4.102) yields the sum of variances of the regression coefficients but not including the constant term. For example, when $\hat{a}_1 = 200$ and $\hat{a}_2 = -0.3$ are assumed, then a variance of $\hat{a}_1$ of 10 differs in meaning from the variance of $\hat{a}_2$ of 10. Nevertheless, the value that Eq. (4.102) gives does not reflect the difference in meaning between the two things. To handle this problem, the $\{x'_{ij}\}$ with specified value j are multiplied by a constant to make the variance of $\{x'_{ij}\}$ equal $\frac{1}{n}$. This transformation is referred to as scaling (page 124 in [2], page 124 in [3]). Scaling makes $\mathbf{X}'^t\mathbf{X}'$ a correlation matrix of $\mathbf{X}'$; all of its diagonal elements are unity. However, as the predictors of a multiple regression equation are usually not regarded as random variables but fixed values, the word correlation is to some extent inappropriate in the situation (page 125 in [2]).

Figure 4.5 shows regression planes given by a regression equation when multicollinearity occurs. The five regression planes are the results of constructing

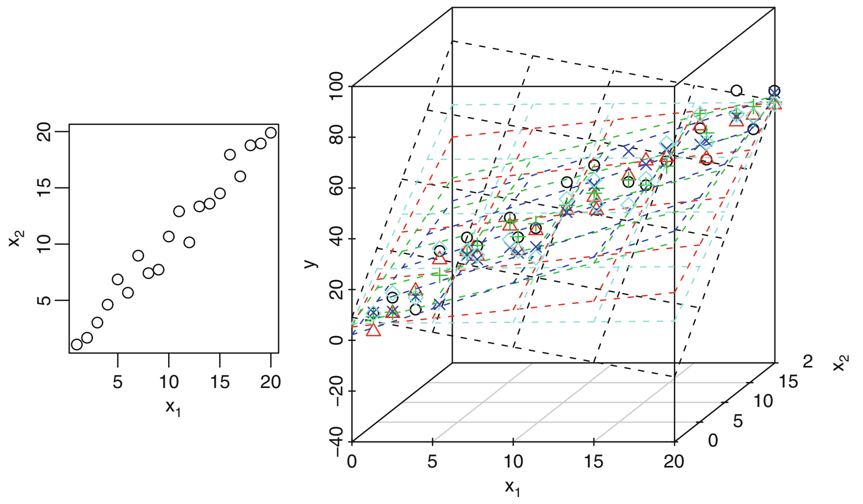


Fig. 4.5 Values of predictors of the simulation data; $\{\epsilon_i^b\}$ $(1 \le i \le 20)$ are realizations of the uniform distribution between −2 and 2 (*left*). Regression planes each of which is produced by one set of simulation data out of five sets of simulation data (*right*)

multiple regression equations by the least squares method. The simulation data for these regression equations are generated by

$$y_i = x_{i1} + 2x_{i2} + 5 + \epsilon_i^a, \tag{4.105}$$

where $\{\epsilon_i^a\}$ $(1 \le i \le 20)$ are realizations of the normal distribution $N(0, 4.9^2)$; the mean is 0 and the variance is 4.9^2. $\{x_{i1}\}$ $(1 \le i \le 20)$ is set as $\{1, 2, 3, \ldots, 20\}$. $\{x_{i2}\}$ $(1 \le i \le 20)$ are generated using

$$x_{i2} = x_{i1} + \epsilon_i^b. \tag{4.106}$$

$\{\epsilon_i^b\}$ $(1 \le i \le 20)$ are realizations of the uniform distribution between −2 and 2. Five sets of simulation data are generated by altering the initial value of the pseudo-random numbers. Figure 4.5 which shows multiple regression equations constructed by these simulation data indicates that the values of the regression coefficients vary dramatically.

In distinct contrast, Fig. 4.6 shows the results when $\{\epsilon_i^b\}$ $(1 \le i \le 20)$ are realizations of a uniform distribution between −9 and 9. As the correlation between the two predictors is large in Fig. 4.5, Eq. (4.91) (page 201) holds with high accuracy. That is, multicollinearity is present. This augments the variances of the regression coefficients. Conversely, as the correlation between the two predictors is small in Fig. 4.6, Eq. (4.91) (page 201) does not hold very much. Therefore, the variances in the regression coefficients are small. These results lead to the prediction that the value of Eq. (4.102) in Fig. 4.5 is large and the value of Eq. (4.102) in Fig. 4.6

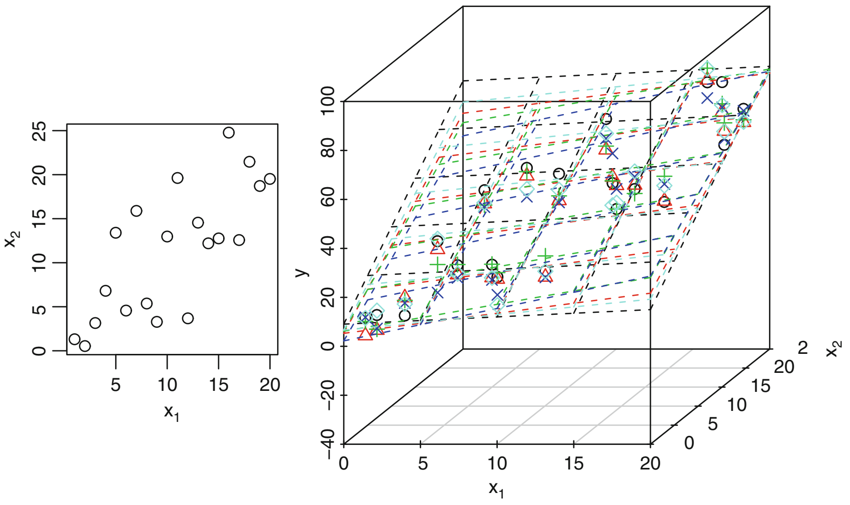


Fig. 4.6 Values of predictors of the simulation data ($\{\epsilon_i^b\}$ $(1 \le i \le 20)$ are realizations of the uniform distribution between -9 and 9) (*left*). Regression planes each of which is produced by one set of simulation data out of five sets of simulation data (*right*)

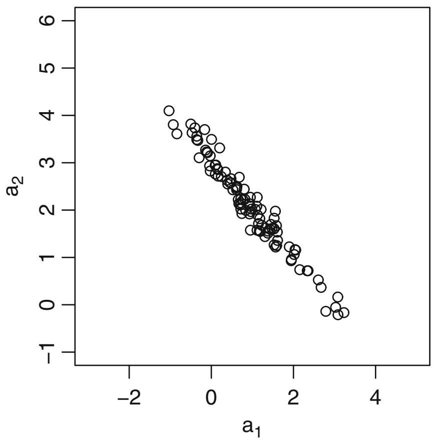


Fig. 4.7 Distribution of the regression coefficients when the simulation data used in Fig. 4.5 are generated 100 times by changing the initial value of the pseudo-random numbers

is small. Indeed, the value of Eq. (4.102) corresponding to Fig. 4.5 is 2.138052 but for Fig. 4.6 is 0.1340991.

The simulation data used in Fig. 4.5 are generated 100 times by changing the initial value of the pseudo-random numbers. The distribution of the values of $\hat{a}_1$ and $\hat{a}_2$ given by each simulation data yields Fig. 4.7. This graph indicates that there is a large negative correlation between $\hat{a}_1$ and $\hat{a}_2$ and the sum of the variance of $\hat{a}_1$ and that of $\hat{a}_2$ are almost identical to 2.138052 obtained from Eq. (4.102).

R Program [4 - 10]

The theoretical value of the variance of each regression coefficient (Eq. (4.95) (page 202)) nearly equals the value given by simulation, as does obviously the sum of the variances of the regression coefficients not including the constant term (Eq. (4.102) (page 204)).

```
col21e()

  function ()
  {
  # (1)
    nd <- 20
    xx1 <- seq(from = 1, to = nd, by = 1)
    set.seed(911)
    xx2 <- xx1 + runif(nd, min = -2, max = 2)
  # (2)
    par(mai = c(1, 1, 1, 1), omi = c(0, 0, 0, 0))
  # (3)
    var0 <- 0
    var1 <- 0
    var2 <- 0
    nt <- 20000
    sd1 <- 4.9
  # (4)
    for(kk in 1:nt){
      set.seed(kk * 3 + 429)
      aa0 <- 5
      aa1 <- 1
      aa2 <- 2
      yy <- aa1 * xx1 + aa2 * xx2 + aa0 +
       rnorm(nd, mean = 0, sd = sd1)
      data1 <- data.frame(x1 = xx1, x2 = xx2, y = yy)
      lm1 <- lm(y ~ x1 + x2, data = data1)
      coef1 <- lm1$coef
  # (5)
      if (kk ==1) {
        plot(coef1[2], coef1[3], xlab = expression(a[1]),
         ylab = expression(a[2]), type = "n",
         xlim = c(-3,5), ylim = c(-1,6))
      }
      if (kk <= 100) {
        points(coef1[2], coef1[3])
      }
```

```
# (6)
    var0 <- var0 + (aa0 - coef1[1])^2
    var1 <- var1 + (aa1 - coef1[2])^2
    var2 <- var2 + (aa2 - coef1[3])^2
  }
# (7)
  var0 <- var0/nt
  var1 <- var1/nt
  var2 <- var2/nt
  print("Variance of each regression coefficient")
  print("(simulation)")
  print(var0)
  print(var1)
  print(var2)
  print("Sum of the variaces of the regression")
  print("coefficients except the regression")
  print("coefficient of the constant")
  print("term (simulation)")
  print(var1+var2)
# (8)
  xx1d <- xx1 - mean(xx1)
  xx2d <- xx2 - mean(xx2)
  xx12d <- cbind(xx1d, xx2d)
  xx12di <- solve(t(xx12d) %*% xx12d)
  cc1 <- sd1^2 * xx12di
  var1a <- cc1[1, 1]
  var2a <- cc1[2, 2]
  print("Variance of each regression coefficient")
  print("except the regression coefficient of")
  print("the constant term (theoretical values")
  print("given by the variance-covariance matrix)")
  print(var1a)
  print(var2a)
  print("Sum of the variaces of the regression")
  print("coefficients except the regression")
  print("coefficient of the constant")
  print("term (theoretical values given by")
  print("the variance-covariance matrix)")
  print(var1a + var2a)
# (9)
  xxad <- t(xx12d) %*% xx12d
  eigen1d <- eigen(xxad)$values
  vv1d <- sum(1/eigen1d)*sd1^2
  print("Sum of the variaces of the regression")
  print("coefficients except the regression")
  print("coefficient of the constant term")
  print("(theoretical values given by")
  print("the eigenvalues)")
  print(vv1d)
}
```

(1) The number of data (nd) is given. The values of predictors of the simulation data (xx1 and xx2) are generated.
(2) par() sets the graphics area.
(3) For storing the variances of regression coefficients given by simulation data, var0 (variance of a_0), var1 (variance of a_1), and var2 (variance of a_2) are prepared. The number of simulations (nt) is given. The standard deviation (sd1) of the errors contained in the predictor values in the simulation data is set at 4.9.
(4) The values of the target variable (yy) of the simulation data are generated. lm() constructs a multiple regression equation. The resultant regression coefficients are stored in lm().
(5) The values of the regression coefficients yielded by the first 100 sets of simulation data are graphed. Figure 4.7 (page 206) is obtained.
(6) The sum of the variances of the regression coefficients given by the simulation data is calculated.
(7) The average of the variances of the regression coefficients given by the simulation data is calculated. The average of variances of the regression coefficients and their sum are output.
(8) The variances of the regression coefficients are calculated using Eq. (4.95) (page 202). These values and their sum are displayed. solve() yields the inverse matrix.
(9) The sum of the variances of the regression coefficients are calculated using Eq. (4.102) (page 204). The result is displayed. eigen() uses the eigenequation of hh to derive eigenvalues and eigenvectors. The component of values in the output from eigen() stores the eigenvalues.

col21e() also outputs:

```
"Variance of each regression coefficient"
"(simulation)"
(Intercept)
5.729675
       x1
0.7881854
       x2
0.8117826
"Sum of the variaces of the regression"
"coefficients except the regression"
"coefficient of the constant"
"term (simulation)"
      x1
1.599968
"Variance of each regression coefficient"
```

```
"except the regression coefficient of"
"the constant term (theoretical values"
"given by the variance-covariance matrix)"
0.783627
0.8056918
"Sum of the variaces of the regression"
"coefficients except the regression"
"coefficient of the constant"
"term (theoretical values given by"
"the variance-covariance matrix)"
1.589319
"Sum of the variaces of the regression"
"coefficients except the regression"
"coefficient of the constant term"
"(theoretical values given by"
"the eigenvalues)"
1.589319
```

The result of Eq. (4.95) (page 202) is completely identical to that of Eq. (4.102) (page 204). It is similar to the result produced by the simulation data.

R Program [4 - 10] End

4.7 Detection of Multicollinearity Using Variance Inflation Factors

To detect the presence and degree of multicollinearity, Variance Inflation Factor (VIF) is another option defined as

$$VIF_j = \frac{1}{1 - R_j^2} \quad (1 \leq j \leq q), \tag{4.107}$$

where R_j^2 is the coefficient of determination (Eq. (4.31) (page 168)) of the multiple regression equation given by the least squares method when the j-th predictor is used as the target variable and the remaining predictors are used as predictors. For example, the following multiple regression equation is constructed for calculating R_2^2 when the number of predictors is 3 ($q = 3$).

$$x_2 = b_0 + b_1 x_1 + b_3 x_3, \tag{4.108}$$

where b_0, b_1, and b_3 are regression coefficients produced by the least squares method. We are in habit of supposing that, if VIF for a specific value of j is equal to

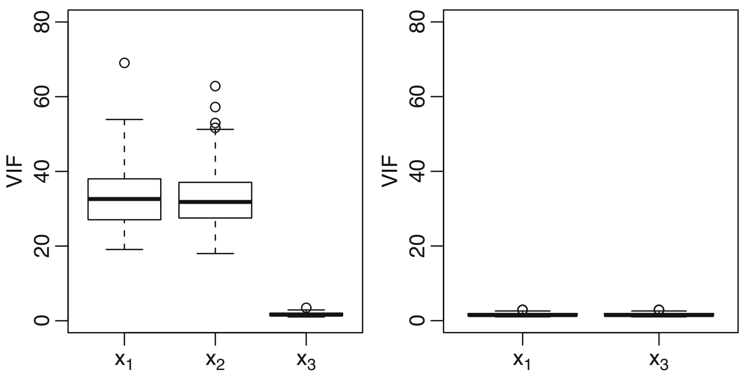


Fig. 4.8 Boxplot showing the values of VIF when the first and the second predictors among the three predictors are involved in bringing about multicollinearity (*left*). Boxplot when the second predictor is removed (*right*)

or larger than 10, the j-th predictor is involved in bringing about multicollinearity. However, this criterion is not always reliable (page 369 in [2]).

To determine the importance of VIF, the values of VIF for the simulation data are calculated. The result is illustrated in Fig. 4.8. The values of the target variable of the simulation data are generated using

$$y_i = 2x_{i1} - 3x_{i2} - 6x_{i3} + 0.5 + \epsilon_i^a, \tag{4.109}$$

where $\{\epsilon_i^a\}$ $(1 \leq i \leq 20)$ are realizations of a normal distribution; the mean is 0 and the variance is 4.9^2. $\{x_{i1}\}$ $(1 \leq i \leq 20)$ is set as $\{1, 2, 3, \ldots, 20\}$. $\{x_{i2}\}$ $(1 \leq i \leq 20)$ are given by the equation below.

$$x_{i2} = x_{i1} + \epsilon_i^b, \tag{4.110}$$

where $\{\epsilon_i^b\}$ $(1 \leq i \leq 20)$ are random values obtained from a uniform distribution over the range -2 and 2. $\{x_{i3}\}$ $(1 \leq i \leq 20)$ are obtained using

$$x_{i3} = x_{i1} + \epsilon_i^c, \tag{4.111}$$

where $\{\epsilon_i^c\}$ $(1 \leq i \leq 20)$ take values between 10 and 40 following a uniform distribution. 100 sets of simulation data are generated by altering the initial value of the pseudo-random numbers. A multiple regression equation is produced using each one set of the simulation data. $\{VIF_j\}$ $(1 \leq j \leq 3)$ are calculated for the respective multiple regression equations. The respective distributions of resulting $\{VIF_j\}$ are shown as a boxplot (Fig. 4.8 (left)). This boxplot reflects the involvement of the first and the second predictors in multicollinearity. When multiple regression equations without the second predictor are constructed, the resultant $\{VIF_1, VIF_3\}$ gives Fig. 4.8 (right). Apparently, multicollinearity is removed.

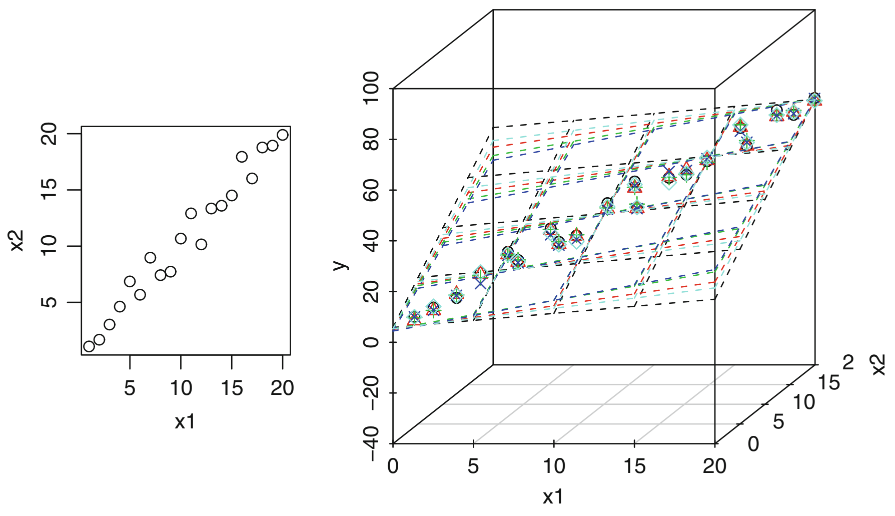


Fig. 4.9 The values for the predictors of the simulation data ($\{\epsilon_i^b\}$ $(1 \leq i \leq 20)$ are distributed over −2 and 2 following a uniform distribution) (*left*). Regression planes are each produced by one set of simulation data out of five sets of simulation data when the errors contained in the target variable are smaller than those of Fig. 4.5 (*right*)

However, as the definition of VIF (Eq. (4.106) (page 205)) shows, VIF is calculated using the values of the predictors of data. Hence, the values of the target variable have no impact on it. Conversely, Eq. (4.95) (page 202) and Eq. (4.97) (page 202) indicate that the variances of regression coefficients are influenced by the amount of errors contained in the values of the target variable in data. For example, whereas Fig. 4.5 (page 205) is given if the errors contained in the values of the target variable in the simulation data are obtained from a normal distribution (with mean 0 and variance 4.9^2), Fig. 4.9 is obtained if the errors are realizations of the normal distribution (mean 0 and variance 1^2). Although the values of the predictors of data are the same as those for Fig. 4.5, the variances of the regression coefficients are small. That is, as the variances of the regression coefficients are not determined by the values of VIF only, we are unable to know the degree of multicollinearity simply by the VIF values if the variances of the regression coefficients are the main concern.

R Program [4 - 11]

The definition of VIF (Eq. (4.106) (page 205)) is confirmed.

```
col31()
  function ()
  {
  # (1)
    library(HH)
  # (2)
    nd <- 20
    xx1 <- seq(from = 1, to = nd, by = 1)
    set.seed(911)
    xx2 <- xx1 + runif(nd, min = -2, max = 2)
    xx3 <- xx1 + runif(nd, min = 10, max = 40)
    yy <- 2 * xx1 - 3 * xx2 - 6 * xx3 + 0.5 + rnorm(nd,
     mean = 0, sd = 4.9)
    data0 <- data.frame(x1 = xx1, x2 = xx2, x3 = xx3,
     y = yy)
  # (3)
    lm0 <- lm(y ~ x1 + x2 + x3, data = data0, x = T)
    vif0 <- vif(lm0)
    print("vif0")
    print(vif0)
  # (4)
    lm1 <- lm(x1 ~ x2 + x3, data = data0)
    r1 <- summary(lm1)$r.squared
    vif1 <- 1/(1-r1)
    print("vif1")
    print(vif1)
  # (5)
    lm2 <- lm(x2 ~ x1 + x3, data = data0)
    r2 <- summary(lm2)$r.squared
    vif2 <- 1/(1-r2)
    print("vif2")
    print(vif2)
  # (6)
    lm3 <- lm(x3 ~ x1 + x2, data = data0)
    r3 <- summary(lm3)$r.squared
    vif3 <- 1/(1-r3)
    print("vif3")
    print(vif3)
  }
```

(1) The use of package "HH" is described. It aims to use `vif()`.

(2) The number of data (`nd`) is given. The values of the predictors of the simulation data are given as `xx1`, `xx2`, and `xx3`. The values of the target variable of the simulation data are given. The simulation data are organized in `data0`.

(3) `lm()` conducts multiple regression using x_1 and x_2, and x_3 as predictors and y as the target variable. The result is stored in `lm0`. `x = T` is set in `lm()` because this setting makes `lm0` contain the design matrix of this multiple regression as its component. This assignment is requisite in using `vif()`. Using this `lm0`, `vif()` calculates the value of VIF. The result is stored in `vif0` and then output.

(4) `lm()` carries out multiple regression using x_2 and x_3 as predictors and x_1 as the target variable. The result is stored in `lm1`. VIF_1 is calculated using Eq. (4.106) (page 205). The result is stored in `vif1` and is output.
(5) `lm()` conducts multiple regression using x_1 and x_3 as predictors and x_2 as the target variable. The result is stored in `lm2`. VIF_2 is calculated using Eq. (4.106). The result is stored in `vif2` and then output.
(6) `lm()` conducts multiple regression using x_1 and x_2 as predictors and x_3 as the target variable. The result is stored in `lm3`. VIF_3 is calculated using Eq. (4.106). The result is stored in `vif3` and then output.

`col31()` outputs:

```
"vif0"
x1        x2         x3
23.484225 22.082298  1.555468
"vif1"
23.48422
"vif2"
22.0823
"vif3"
1.555468
```

Values given by `vif()` are identical to those obtained by Eq. (4.106) (page 205).

R Program [4 - 11] End

R Program [4 - 12]

VIF (Eq. (4.106) (page 205)) indicates the presence of multicollinearity. `col32()`

```
function ()
{
# (1)
  library(HH)
# (2)
  nd <- 20
  xx1 <- seq(from = 1, to = nd, by = 1)
# (3)
  nt <- 100
# (4)
  vif0m <- matrix(rep(0, length = nt * 3), ncol = nt)
  for (kk in 1:nt){
    set.seed(kk*2 + 23)
    xx2 <- xx1 + runif(nd, min = -2, max = 2)
    xx3 <- xx1 + runif(nd, min = 10, max = 40)
```

```
      yy <- 2 * xx1 -3 * xx2 - 6 * xx3  + 0.5 +
       rnorm(nd, mean = 0, sd = 4.9)
      data0 <- data.frame(x1 = xx1, x2 = xx2,
       x3 = xx3, y = yy)
  # (5)
      lm0 <- lm(y ~ x1 + x2 + x3, data = data0, x = T)
      vif0 <- vif(lm0)
      vif0m[, kk] <- vif0
    }
  # (6)
    par(mai = c(1, 1, 1, 1), omi = c(0, 0, 0, 0))
    boxplot(t(vif0m), names = c("x1", "x2", "x3"),
     ylab = "VIF", ylim = c(0,80))
  }
```

(1) The use of package "HH" is described. It aims to use `vif()`.
(2) The number of data (`nd`) is given. The values of the first predictor of the simulation data (`xx1`) are generated.
(3) The number of simulations (`nt`) is given.
(4) `vif0m` in the form of matrix is prepared for storing the values of VIF. The values of the second predictor (`xx2`), those of the third predictor (`xx3`), and those of the target variable (`yy`) are generated by changing the initial value of the pseudo-random numbers. These values are organized in `data0`.
(5) `lm()` produces a multiple regression equation. The result is stored in `lm0`. `vif()` calculates $\{VIF_j\}$ $(1 \leq j \leq 3)$ using `lm0`. The resultant values are stored in `vif0m`.
(6) Distributions of the values of VIF of respective predictors are graphed in the form of the boxplot using `boxplot()`. Figure 4.8 (page 211) is obtained.

R Program [4 - 12] End

4.8 Hessian Matrix of Log-Likelihood

When a random variable of Y follows a normal distribution, the associated probability density function (Eq. (2.7) (page 52)) is written as

$$f(y) = \left(\frac{1}{\sqrt{2\pi\sigma^2}}\right) \exp\left(-\frac{1}{2\sigma^2}(y-\mu)^2\right). \tag{4.112}$$

A variable of y in this probability density function is a nonrandom variable (nonstochastic variable). If this is the case, we assume that we derive one sample (y_1; y_1 is also a nonrandom variable) from the population. Our problem is whether we can regard Eq. (4.112) as the probability density function of the population. One the one hand, when y_1 is substituted into y in Eq. (4.112), the value given by Eq. (4.112) is large if y_1 is close to μ. The large value given by Eq. (4.112) indicates that y_1 is located where the value of the probability density function is large. That is, y_1 is an all-too-common value in this probability density function. Therefore, we can fairly believe the assumption that y_1 is derived from this probability density function. In other words, this assumption is likely. On the other hand, when y_1 is far from μ, the value given by Eq. (4.112) is small. This means that y_1 is placed where the value of the probability density function is small. Hence, the assumption that y_1 is derived from this probability density function is not very likely. Therefore, when y_1 is given as a value of y in $f(y)$ (Eq. (4.112)), the values given by diverse values of μ show the likelihoods of the respective μ. Hence, when y in $f(y)$ is fixed at y_1 and μ is varied, $f(y_1)$ is considered a function of μ. If we enlarge the value of $f(y_1)$ by adjusting μ, the probability density function which makes y_1 look likely is obtained.

As just described, when we assume that the data are fixed but the parameters are varied in the probability density function, the function is called the likelihood function or simply the likelihood. As the natural logarithm of the likelihood is usually easier to deal with than the likelihood as it is, we refer to the logarithm (natural logarithm) of the likelihood as the log-likelihood.

Moreover, generalization of Eq. (4.112) yields the probability density function of a normal distribution for the random variable of the vector of $(Y_1, Y_2, \ldots, Y_n)$. When random variables are independent of one another and the variances are equal, we have

$$\begin{aligned} f(y_1, y_2, \ldots, y_n) &= \left(\frac{1}{\sqrt{2\pi\sigma^2}}\right)^n \prod_{i=1}^{n} \exp\left(-\frac{1}{2\sigma^2}(y_i-\mu)^2\right) \\ &= \left(\frac{1}{\sqrt{2\pi\sigma^2}}\right)^n \exp\left(-\frac{1}{2\sigma^2}\sum_{i=1}^{n}(y_i-\mu)^2\right). \end{aligned} \tag{4.113}$$

The variable of this probability density function is the vector of $(y_1, y_2, \ldots, y_n)$ and this vector is a nonrandom variable. In this situation, when a sampling gives n data and $(y_1, y_2, \ldots, y_n)$ are replaced with real data, the large value given by Eq. (4.113) with the data of $(y_1, y_2, \ldots, y_n)$ indicates a large likelihood of the assumption that the data of $(y_1, y_2, \ldots, y_n)$ are obtained with this density function. Then, we derive the probability density function which makes $(y_1, y_2, \ldots, y_n)$ look likely by adjusting μ with fixed values of $(y_1, y_2, \ldots, y_n)$. Furthermore, when σ^2 is varied, the probability density function which makes $(y_1, y_2, \ldots, y_n)$ more likely is produced by adjusting σ^2 as well as μ.

Next, we assume that n data sets of $\{(x_{11}, \ldots, x_{1q}, y_1), \ldots, (x_{n1}, \ldots, x_{nq}, y_n)\}$ are sampled from a population and the data satisfy the following equation.

$$\tilde{y}_i = a_0 + \sum_{j=1}^{q} a_j x_{ij} + \epsilon_i, \tag{4.114}$$

where $\{x_{ij}\}$ $(0 \leq j \leq q)$ represent the conditions of the experiment or the observation when $\tilde{y}_i$ are obtained. $\{a_j\}$ $(0 \leq j \leq q)$ are regression coefficients to determine μ_i using the conditions of the experiment or the observation. $\{\epsilon_i\}$ $(1 \leq i \leq n)$ are errors. When values for $\{a_j\}$ $(0 \leq j \leq q)$ are set, we have

$$\mu_i = a_0 + \sum_{j=1}^{q} a_j x_{ij}. \tag{4.115}$$

Then, when $\{\epsilon_i\}$ in Eq. (4.114) obeys Eq. (4.113), we obtain

$$\begin{aligned} f(\tilde{y}_1, \tilde{y}_2, \ldots, \tilde{y}_n) &= \left(\frac{1}{\sqrt{2\pi\sigma^2}}\right)^n \exp\left(-\frac{1}{2\sigma^2}\prod_{i=1}^{n}\epsilon_i^2\right) \\ &= \left(\frac{1}{\sqrt{2\pi\sigma^2}}\right)^n \exp\left(-\frac{1}{2\sigma^2}\prod_{i=1}^{n}(\tilde{y}_i - a_0 - \sum_{j=1}^{q} a_j x_{ij})^2\right). \end{aligned} \tag{4.116}$$

If the value of this equation given by specific values of $\{a_j\}$ and σ^2 is large, the assumption that $\{y_i\}$ is calculated using the values, given by substituting σ^2 and $\{a_j\}$ into Eq. (4.114), is very likely. Hence, Eq. (4.116) is considered to be a function which gives the likelihood of σ^2 and $\{a_j\}$. We then write

$$L(\{\hat{a}_j\}, \sigma^2 | \{(x_i, y_i)\}) = \left(\frac{1}{\sqrt{2\pi\sigma^2}}\right)^n \exp\left(-\frac{1}{2\sigma^2}\sum_{i=1}^{n}(y_i - a_0 - \sum_{j=1}^{q} a_j x_{ij})^2\right). \tag{4.117}$$

This is the likelihood function when we assume Eq. (4.114) in which the mean of $\{\epsilon_i\}$ is 0 and the variance is σ^2. $L(\{\hat{a}_j\}, \sigma^2 | \{(x_i, y_i)\})$ indicates the likelihood of $\{\hat{a}_j\}$ and σ^2 when $\{(x_i, y_i)\}$ are fixed. The log-likelihood is then written as

$$l(\{a_j\}, \sigma^2 | \{(x_i, y_i)\}) = -\frac{n}{2}\log(2\pi) - \frac{n}{2}\log(\sigma^2) - \frac{1}{2\sigma^2}\sum_{i=1}^{n}(y_i - a_0 - \sum_{j=1}^{q} a_j x_{ij})^2. \tag{4.118}$$

To obtain $\{a_j\}$ which maximizes this value, this equation is differentiated with respect to a_j and the result is set to 0 as

$$\frac{\partial l(\{a_j\}, \sigma^2 | \{(x_i, y_i)\})}{\partial a_j} = -\frac{n}{2}\log(2\pi) - \frac{n}{2}\log(\sigma^2) - \frac{1}{2\sigma^2}\sum_{i=1}^{n}(y_i - a_0 - \sum_{j=1}^{q} a_j x_{ij})^2 = 0. \tag{4.119}$$

In this equation, $-\sum_{i=1}^{n}(y_i - a_0 - \sum_{j=1}^{q} a_j x_{ij})^2$ is the only part which depends on $\{a_j\}$. This is identical to *RSS* (Eq. (4.1) (page 163)) multiplied by -1. Therefore, $\{a_j\}$ which maximizes Eq. (4.119) is given by Eq. (4.9) (page 164).

When $j = 0$ holds, Eq. (4.119) is written as

$$\frac{\partial l(\{a_j\}, \sigma^2 | \{(x_i, y_i)\})}{\partial a_0} = -\frac{1}{\sigma^2}\sum_{i=1}^{n}(y_i - a_0 - \sum_{j=1}^{q} a_j x_{ij}) = 0. \tag{4.120}$$

Otherwise, when $j \neq 0$ holds, we have

$$\frac{\partial l(\{a_j\}, \sigma^2 | \{(x_i, y_i)\})}{\partial a_j} = -\frac{1}{\sigma^2}\sum_{i=1}^{n}x_{ij}(y_i - a_0 - \sum_{j=1}^{q} a_j x_{ij}) = 0. \tag{4.121}$$

The second derivatives of $l(\{a_j\}, \sigma^2 | \{(x_i, y_i)\})$ (Eq. (4.118) (page 217)) are given as

$$\begin{aligned}
\frac{\partial^2 l(\{a_j\}, \sigma^2 | \{(x_i, y_i)\})}{\partial a_0^2} &= -\frac{n}{\sigma^2} \\
\frac{\partial^2 l(\{a_j\}, \sigma^2 | \{(x_i, y_i)\})}{\partial a_0 \partial a_j} &= -\frac{1}{\sigma^2}\sum_{i=1}^{n} x_{ij} \qquad (j \neq 0) \\
\frac{\partial^2 l(\{a_j\}, \sigma^2 | \{(x_i, y_i)\})}{\partial a_j \partial a_0} &= -\frac{1}{\sigma^2}\sum_{i=1}^{n} x_{ij} \qquad (j \neq 0) \\
\frac{\partial^2 l(\{a_j\}, \sigma^2 | \{(x_i, y_i)\})}{\partial a_j \partial a_k} &= -\frac{1}{\sigma^2}\sum_{i=1}^{n} x_{ij}x_{ik} \qquad (j \neq 0, k \neq 0).
\end{aligned} \tag{4.122}$$

When $q = 2$ and $l(\{a_j\}, \sigma^2 | \{(x_i, y_i)\})$ is denoted more simply by l, we have

$$\begin{aligned}
\begin{pmatrix}
\frac{\partial^2 l}{\partial a_0^2} & \frac{\partial^2 l}{\partial a_0 \partial a_1} & \frac{\partial^2 l}{\partial a_0 \partial a_2} \\
\frac{\partial^2 l}{\partial a_1 \partial a_0} & \frac{\partial^2 l}{\partial a_1^2} & \frac{\partial^2 l}{\partial a_1 \partial a_3} \\
\frac{\partial^2 l}{\partial a_2 \partial a_0} & \frac{\partial^2 l}{\partial a_2 \partial a_1} & \frac{\partial^2 l}{\partial a_2^2}
\end{pmatrix}
&= -\frac{1}{\sigma^2}
\begin{pmatrix}
n & \sum_{i=1}^{n} x_{i1} & \sum_{i=1}^{n} x_{i2} \\
\sum_{i=1}^{n} x_{i1} & \sum_{i=1}^{n} x_{i1}^2 & \sum_{i=1}^{n} x_{i1}x_{i2} \\
\sum_{i=1}^{n} x_{i2} & \sum_{i=1}^{n} x_{i1}x_{i2} & \sum_{i=1}^{n} x_{i2}^2.
\end{pmatrix} \\
&= -\frac{1}{\sigma^2}\mathbf{X}^t\mathbf{X}.
\end{aligned} \tag{4.123}$$

This **X** is a design matrix (Eq. (4.4) (page 163)). The second line of Eq. (4.123) equals the product of the inverse matrix of Eq. (4.42) (page 176) and (-1). It indicates that this matrix is closely bound up with the reliability of the regression coefficients. It is further explained in Fig. 4.10 (page 221) and surrounding text. The matrix in which the jk-element is $\frac{\partial^2 l}{\partial a_j \partial a_k}$ is referred to as the Hessian matrix. It is not limited for use as a multiple regression equation. In general, even if l does not stand for the log-likelihood, the matrix constructed by the second derivatives of a scalar function is called the Hessian matrix in mathematics.

Moreover, $\{a_j\}$ which satisfies Eqs. (4.120) (page 218) and (4.121) (page 218) (that is, $\{a_j\}$ are obtained by the least squares method) are represented as $\{\hat{a}_j\}$. When $q = 2$, l is expanded in a Taylor series at $\{\hat{a}_j\}$ yielding:

$$\begin{aligned}
&l(a_0, a_1, a_2) \approx l(\hat{a}_0, \hat{a}_1, \hat{a}_2)\\
&+ (a_0 - \hat{a}_0)\frac{\partial l}{\partial a_0'}\bigg|_{a_0'=a_0} + (a_1 - \hat{a}_1)\frac{\partial l}{\partial a_1'}\bigg|_{a_1'=a_1} + (a_2 - \hat{a}_2)\frac{\partial 2}{\partial a_2'}\bigg|_{a_2'=a_2}\\
&+ \frac{\partial^2 l}{\partial a_0'^2}\bigg|_{a_0'=a_0}(a_0 - \hat{a}_0)^2 + \frac{\partial^2 l}{\partial a_1'^2}\bigg|_{a_1'=a_1}(a_1 - \hat{a}_1)^2 + \frac{\partial^2 l}{\partial a_2'^2}\bigg|_{a_2'=a_2}(a_2 - \hat{a}_2)^2\\
&+ 2\frac{\partial^2 l}{\partial a_0' \partial a_1'}\bigg|_{a_0'=a_0, a_1'=a_1}(a_0 - \hat{a}_0)(a_1 - \hat{a}_1)\\
&+ 2\frac{\partial^2 l}{\partial a_1' \partial a_2'}\bigg|_{a_1'=a_1, a_2'=a_2}(a_1 - \hat{a}_1)(a_2 - \hat{a}_2)\\
&+ 2\frac{\partial^2 l}{\partial a_2' \partial a_0'}\bigg|_{a_2'=a_2, a_0'=a_0}(a_2 - \hat{a}_2)(a_0 - \hat{a}_0).
\end{aligned} \tag{4.124}$$

Equations (4.120) and (4.121) implies

$$\frac{\partial l}{\partial a_0'}\bigg|_{a_0'=a_0} = 0, \qquad \frac{\partial l}{\partial a_1'}\bigg|_{a_1'=a_1} = 0, \qquad \frac{\partial 2}{\partial a_2'}\bigg|_{a_2'=a_2} = 0. \tag{4.125}$$

Substitution of Eqs. (4.123) (page 218) and (4.125) (page 219) into Eq. (4.124) gives

$$\begin{aligned}
&l(a_0, a_1, a_2)\\
&\approx l(\hat{a}_0, \hat{a}_1, \hat{a}_2) - \frac{1}{\sigma^2}\Bigg(n(a_0 - \hat{a}_0)^2 + (a_1 - \hat{a}_1)^2\sum_{i=1}^{n} x_{i1}^2 + (a_2 - \hat{a}_2)^2\sum_{i=1}^{n} x_{i2}^2\\
&+ 2(a_0 - \hat{a}_0)(a_1 - \hat{a}_1)\sum_{i=1}^{n} x_{i1} + 2(a_1 - \hat{a}_1)(a_2 - \hat{a}_2)\sum_{i=1}^{n} x_{i1}x_{i2}\\
&+ 2(a_2 - \hat{a}_2)(a_0 - \hat{a}_0)\sum_{i=1}^{n} x_{i2}\Bigg).
\end{aligned} \tag{4.126}$$

Use of Eq. (4.123) (page 218) leads to

$$l(a_0, a_1, a_2) \approx l(\hat{a}_0, \hat{a}_1, \hat{a}_2) - \frac{(\mathbf{a} - \hat{\mathbf{a}})^t \mathbf{X}^t \mathbf{X} (\mathbf{a} - \hat{\mathbf{a}})}{\sigma^2}, \tag{4.127}$$

where $\mathbf{a}$ and $\hat{\mathbf{a}}$ are defined as

$$\mathbf{a} = \begin{pmatrix} a_0 \\ a_1 \\ a_2 \end{pmatrix}, \quad \hat{\mathbf{a}} = \begin{pmatrix} \hat{a}_0 \\ \hat{a}_1 \\ \hat{a}_2 \end{pmatrix}. \tag{4.128}$$

Equation (1.77) (page 43) shows that $\mathbf{X}^t\mathbf{X}$ is a positive semidefinite matrix. Hence, diagonalization of $\mathbf{X}^t\mathbf{X}$ is realized by an orthogonal mtrix U (Eq. (1.51) (page 35)).

$$\mathbf{X}^t\mathbf{X} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^{-1}. \tag{4.129}$$

All of the diagonal elements of $\boldsymbol{\Lambda}$ are positive or 0. Then, we have

$$l(a_0, a_1, a_2) \approx l(\hat{a}_0, \hat{a}_1, \hat{a}_2) - \frac{\mathbf{b}^t \boldsymbol{\Lambda} \mathbf{b}}{\sigma^2}, \tag{4.130}$$

where $\mathbf{b}$ is defined as

$$\mathbf{b} = \mathbf{U}^{-1}(\mathbf{a} - \hat{\mathbf{a}}). \tag{4.131}$$

When $\mathbf{a} = \hat{\mathbf{a}}$, the derivative of $l(a_0, a_1, a_2)$ with respect to a_0 is 0 (Eq. (4.120) (page 218)), as for a_1 (Eq. (4.121) (page 218)) and a_2 (Eq. (4.121)). Hence, if diagonal elements of $\boldsymbol{\Lambda}$ are positive, that is, $\mathbf{X}^t\mathbf{X}$ is a positive definite matrix, the value of Eq. (4.130) takes the maximum value when $\mathbf{a} = \hat{\mathbf{a}}$ is assigned (refer to Eq. (1.76) (page 41) and surrounding text).

The Fisher information matrix (or simply information matrix) is a similar construct like the Hessian matrix. Denoting the jk-element of Fisher information matrix as In_{jk}, it is defined as

$$In_{jk} = -E\left[\frac{\partial^2 l}{\partial\theta_j \partial\theta_k}\right], \tag{4.132}$$

or equivalently (page 185 in [1])

$$In_{jk} = E\left[\frac{\partial l}{\partial\theta_j}\frac{\partial l}{\partial\theta_k}\right]. \tag{4.133}$$

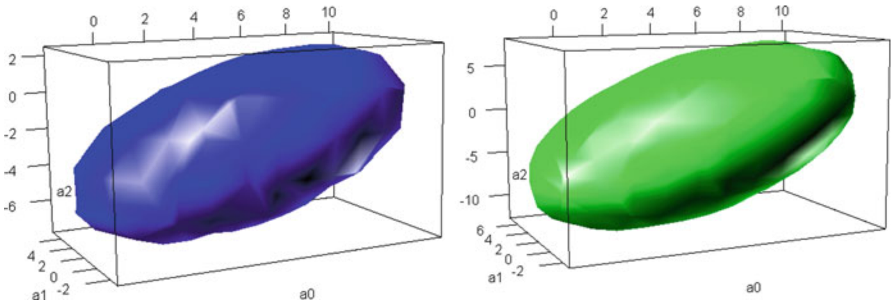


Fig. 4.10 Constant-height surface of −450 of the log-likelihood given by Eq. (4.118) when the simulation data generated by Eq. (4.134) is used (*left*). Constant-height surface with the same settings except that $\{x_{i2}\}$ are uniform random numbers between −1 and 0.5; note that the scale of a_2 is different from that of Fig. 4.10 (*left*) (*right*)

$\{\theta_j\}$ are regression coefficients contained in l (log-likelihood). With $\dfrac{\partial^2 l}{\partial\theta_j\partial\theta_k}$ for a multiple regression defined as in Eq. (4.122) (page 218), it does not contain random variables. Hence, the following equation holds:

$$In_{jk} = -E\left[\frac{\partial^2 l}{\partial a_j \partial a_k}\right] = -\frac{\partial^2 l}{\partial a_j \partial a_k}. \tag{4.134}$$

That is, the Hessian matrix multiplied by (−1) is Fisher information matrix. However, Eq. (4.134) does not hold exactly in a generalized linear regression because the Hessian matrix depends on $\{y_i\}$.

To understand the behavior of Eq. (4.118) (page 217), simulation data are generated using

$$y_i = 2x_{i1} - 3x_{i2} + 4 + \epsilon_i, \tag{4.135}$$

where $\{x_{i1}\}$ $(1 \le i \le 100)$ are uniform random numbers between −1 and 2, $\{x_{i2}\}$ $(1 \le i \le 100)$ uniform random numbers between −2 and 1, and $\{\epsilon_i\}$ are obtained from a normal distribution (with mean 0 and variance 2^2). With $\sigma^2 = 4$ in Eq. (4.118), various values are assigned as a_0, a_1, and a_2. The constant-height surface of −450 of l is illustrated in Fig. 4.10 (left); the value of l is larger than −450 inside this constant-height surface. Next, $\{x_{i2}\}$ are changed to uniform random numbers between −1 and 0.5 and the same simulation is performed. The result is illustrated in Fig. 4.10 (right). The range of the constant-height surface turns out to be larger in direction a_2, implying that the area where l is close to maximum l spreads out in that direction. This is an accurate reflection of the difficulty in estimating a_2 when the variation of $\{x_{i2}\}$ is small. In other words, when the range of x_2 is narrow, the amount of information we have about a_2 is small. Thus, the information matrix indicates how much information about the regression

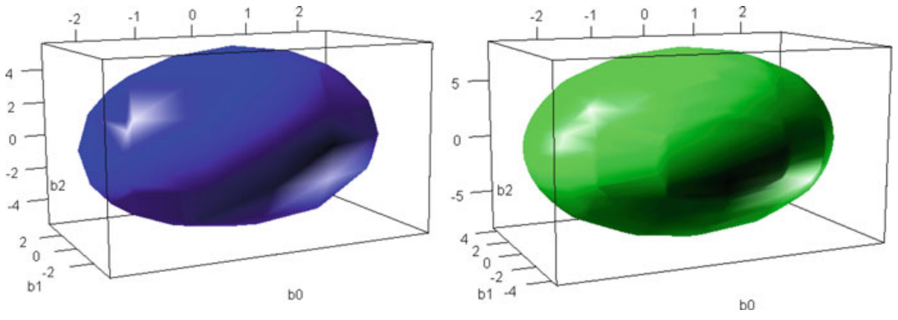


Fig. 4.11 Result of orthogonal transformation of Fig. 4.10

coefficients is contained in the data. This is what “information” signifies here and is different from that of Akaike’s Information Criterion (AIC). It is also different to that in Shannon’s Information Theory. For this reason, each diagonal element of the information matrix is called Fisher information.

Using the same data as those used to generate Fig. 4.10 (left), the constant-height surface of -450 of l is drawn with various values of b_0, b_1, and b_2 in Eq. (4.130) (page 220) (Fig. 4.11 (left)). When the data in Fig. 4.10 (right) are used, the transformation generates the illustration in Fig. 4.11 (right). The constant-height surfaces drawn in Fig. 4.11 show that the orthogonal transformation of the coordinates used in Fig. 4.10 constructs the spheroid for which the axes of the rotation become new coordinates. The transformation of Fig. 1.17 (left) (page 41) into Fig. 1.17 (right) is extended to the three-dimensional case in this manner.

R Program [4 - 13]

Constant-height surface of the log-likelihood of multiple regression is illustrated in three-dimensional graph (Fig. 4.10 (left)).

```
hess1()
  function() {
  # (1)
    library(misc3d)
    library(rgl)
  # (2)
    fun1 <- function(af0, af1, af2) {
  # (3)
      nn <- length(af0)
  # (4)
      set.seed(1526)
      nd <- 100
      xx1 <- runif(n=nd, min=-1, max=2)
      xx2 <- runif(n=nd, min=-2, max=1)
      yy <- 2*xx1 - 3*xx2 + 4 +rnorm(n = nd,
       mean = 0, sd = 2)
      llf <- NULL
      for(ii in 1:nn){
        ss <- sum((yy - af0[ii] - af1[ii]*xx1 -
         af2[ii]*xx2)^2)
        llf[ii] <- -nd*0.5*log(2*pi) -
         nd*0.5*log(sig2) - ss*0.5/sig2
      }
      return(llf)
    }
  # (5)
    th1 <- -450
    sig2 <<- 4
  # (6)
    a0 <- seq(from = -30, to = 40, by = 1)
    a1 <- seq(from = -8, to = 14, by = 2)
    a2 <- seq(from =  -10, to = 6, by = 2)
  # (7)
    clear3d(type = "all")
    rgl.light(theta = -40, phi = 50,
     viewpoint.rel = T, ambient = "white",
     diffuse = "white", specular = "white")
    rgl.light(theta = -60, phi = 10,
     viewpoint.rel = T, ambient = "white",
     diffuse = "white", specular = "white")
    rgl.light(theta = 130, phi = -30,
     viewpoint.rel = T, ambient = "white",
     diffuse = "white", specular = "white")
  # (8)
    contour3d(fun1, th1, a0, a1, a2,
     color = "blue",  engine = "rgl", scale = T,
     color2 = "red", fill = T, smooth=T)
  # (9)
    axes3d()
    title3d(xlab = "a0", ylab = "a1", zlab = "a2")
    box3d()
  # (10)
    aspect3d(x = 3.5, y = 2.2, z = 2)
  # (11)
    rm(sig2, envir = .GlobalEnv)
  }
```

(1) The use of package “misc3d” and the package “rgl” is described (page 226 in [6]).

(2) The function (`fun1()`) which gives log-likelihood is defined. `af0`, `af1`, and `af2` are values of the regression coefficients.

(3) The number of lattice points where the value of log-likelihood is calculated is given as `nn`.
(4) `nd` data (`xx1`, `xx2`, and `yy`) are given. The values of log-likelihood are calculated at lattice points. The results are stored in `llf`.
(5) The values on the constant-height surface are saved as `th1`. The variance (`sig2`) of the errors which obey a normal distribution is set as 4. Because `sig2` is referred to in `fun1()`, `<<-` is used here.
(6) The value of equally-spaced points (`a0`, `a1`, and `a2`) for constructing lattice points are given.
(7) `clear3d(type = ''all'')` deletes all the graphs. `rgl.light()` sets the positions and colors of lights.
(8) `contour3d()` draws a constant-height surface. As this three-dimensional graph can be rotated using a mouse, this constant-height surface is observable from various directions.
(9) `axes3d()` adds coordinate axes to the three-dimensional graph. `title3d()` writes explanations of the coordinate axes. `box3d()` draws a box which surrounds the area of the three-dimensional graph.
(10) `aspect3d()` adjusts the ratio of the lengths of the three directions of the three-dimensional graph.
(11) `sig2` is deleted from the global environment.

R Program [4 - 13] End

References

1. Bickel PJ, Doksum KA (2000) Mathematical statistics: basic ideas and selected topics, vol I, 2nd edn. Prentice Hall, Englewood Cliffs
2. Myers RH (1990) Classical and modern regression with applications (Duxbury Classic). Duxbury, North Scituate
3. Ryan TP (1996) Modern regression methods. Wiley-Interscience, New York
4. Takezawa K (2006) Introduction to nonparametric regression. Wiley, New York
5. Takezawa K (2012) Flexible model selection criterion for multiple regression. Open J Stat 2(4):401–407
6. Takezawa K (2012) Guidebook to R graphics using microsoft. Wiley, New York

Chapter 5
Akaike's Information Criterion (AIC) and the Third Variance

5.1 C_p and FPE

One statistic in selecting a multiple regression equation is Mallows' C_p, which is an approximation of the error variance of the estimates given by a multiple regression equation. This error variance multiplied by n is written as

$$
\begin{aligned}
E\Big[\sum_{i=1}^{n}(\hat{y}_i - E[y_i])^2\Big] &= E\Big[\big(\mathbf{H}(\tilde{\mathbf{y}}+\boldsymbol{\epsilon})-\tilde{\mathbf{y}}\big)^t\big(\mathbf{H}(\tilde{\mathbf{y}}+\boldsymbol{\epsilon})-\tilde{\mathbf{y}}\big)\Big] \\
&= E\Big[\big((\mathbf{H}-\mathbf{I})\tilde{\mathbf{y}}+\mathbf{H}\boldsymbol{\epsilon}\big)^t\big((\mathbf{H}-\mathbf{I})\tilde{\mathbf{y}}+\mathbf{H}\boldsymbol{\epsilon}\big)\Big] \\
&= \tilde{\mathbf{y}}^t(\mathbf{I}-\mathbf{H})\tilde{\mathbf{y}} + E[\boldsymbol{\epsilon}^t\mathbf{H}\boldsymbol{\epsilon}] \\
&= \tilde{\mathbf{y}}^t(\mathbf{I}-\mathbf{H})\tilde{\mathbf{y}} + (q+1)\sigma^2,
\end{aligned}
\tag{5.1}
$$

where $\mathbf{H}$ is a hat matrix (Eq. (4.13) (page 164)) and $\tilde{\mathbf{y}}$ are true values, that is, $\tilde{y}_i = E[y_i]$ holds. The relation $\mathbf{H}^2 = \mathbf{H}$ (Eq. (4.28) (page 167)) is used here, along with Eq. (4.34) (page 174). In deriving Eq. (4.32) (page 174), $\mathbf{H}\tilde{\mathbf{y}} = \tilde{\mathbf{y}}$ was assumed. However, this relationship is not assumed here.

On the other hand, the expectation of the residual sum of squares is written as

$$
\begin{aligned}
E\Big[\sum_{i=1}^{n}(\hat{y}_i - y_i)^2\Big] &= E\Big[\big(\mathbf{H}(\tilde{\mathbf{y}}+\boldsymbol{\epsilon})-(\tilde{\mathbf{y}}+\boldsymbol{\epsilon})\big)^t\big(\mathbf{H}(\tilde{\mathbf{y}}+\boldsymbol{\epsilon})-(\tilde{\mathbf{y}}+\boldsymbol{\epsilon})\big)\Big] \\
&= \Big[\big((\mathbf{H}-\mathbf{I})(\tilde{\mathbf{y}}+\boldsymbol{\epsilon})\big)^t\big((\mathbf{H}-\mathbf{I})(\tilde{\mathbf{y}}+\boldsymbol{\epsilon})\big)\Big] \\
&= \tilde{\mathbf{y}}^t(\mathbf{I}-\mathbf{H})\tilde{\mathbf{y}} + E[\boldsymbol{\epsilon}^t(\mathbf{I}-\mathbf{H})\boldsymbol{\epsilon}] \\
&= \tilde{\mathbf{y}}^t(\mathbf{I}-\mathbf{H})\tilde{\mathbf{y}} + (n-q-1)\sigma^2.
\end{aligned}
\tag{5.2}
$$

The relation $\mathbf{H}\tilde{\mathbf{y}} = \tilde{\mathbf{y}}$ is not assumed here, either. The following equation is obtained from Eq. (5.2).

$$\tilde{\mathbf{y}}^t(\mathbf{I}-\mathbf{H})\tilde{\mathbf{y}} = E\Big[\sum_{i=1}^{n}(\hat{y}_i - y_i)^2\Big] - (n-q-1)\sigma^2. \tag{5.3}$$

Substitution of this equation into Eq. (5.1) gives

$$\begin{aligned} E\Big[\sum_{i=1}^{n}(\hat{y}_i - E[y_i])^2\Big] &= E\Big[\sum_{i=1}^{n}(\hat{y}_i - y_i)^2\Big] - (n-q-1)\sigma^2 + (q+1)\sigma^2 \\ &= E\Big[\sum_{i=1}^{n}(\hat{y}_i - y_i)^2\Big] - (n-2q-2)\sigma^2 \\ &\approx \sum_{i=1}^{n}(\hat{y}_i - y_i)^2 - (n-2q-2)\sigma^2. \end{aligned} \tag{5.4}$$

Dividing the previous equation by σ^2 determines Mallows' C_p [5] which is defined as

$$C_p = \frac{\sum_{i=1}^{n}(\hat{y}_i - y_i)^2}{\sigma^2} - n + 2q + 2. \tag{5.5}$$

As a regression equation that yields a small error variance of the estimates is considered to be beneficial, we choose a regression equation with a small value of C_p. However, σ^2 is needed to estimate C_p. When σ^2 is unknown, the multiple regression equation with all predictors is constructed and $\hat{\sigma}^2$ given by the following

$$\hat{\sigma}^2 = \frac{\sum_{i=1}^{n}(\hat{y}_i - y_i)^2}{n - q_{max} - 1} \tag{5.6}$$

is used. Here q_{max} is the number of all predictors. When Eq. (5.6) is used instead of σ^2 in Eq. (5.5), C_p no longer is precisely an unbiased estimator of the value of Eq. (5.4) divided by σ^2 because $\frac{1}{\hat{\sigma}^2}$ is not an unbiased estimator of $\frac{1}{\sigma^2}$ (20 page in [6]).

Alternatively, if $\mathbf{H}\tilde{\mathbf{y}} = \tilde{\mathbf{y}}$ holds, Eq. (5.1) (page 225) becomes

$$E\Big[\sum_{i=1}^{n}(\hat{y}_i - E[y_i])^2\Big] = (q+1)\sigma^2. \tag{5.7}$$

Therefore, when we only use regression equations that satisfy $\mathbf{H}\tilde{\mathbf{y}} = \tilde{\mathbf{y}}$ in the selection of regression equation, estimates closer to the true values are obtained by a smaller value of q. For example, when the true values follow a quadratic equation, whichever one of the quadratic equation and the cubic equation is used, $\mathbf{H}\tilde{\mathbf{y}} = \tilde{\mathbf{y}}$ holds. However, a quadratic equation gives closer estimates to the true values.

One other statistic used in selecting a multiple regression equation is the final prediction error (FPE) ([1], 247 page of [4]). This is an approximation of the residual sum of squares for data in the future ($\{y_i^*\}$ $(1 \le i \le n)$, i.e., $\{\tilde{y}_i + \epsilon^*\}$). That is, the following values are obtained in a manner analogous to Eq. (3.45) (page 116).

$$\begin{aligned} E\Big[\sum_{i=1}^{n}(y_i^* - \hat{y}_i)^2\Big] &= E\Big[\big(\tilde{\mathbf{y}} + \boldsymbol{\epsilon}^* - \mathbf{H}(\tilde{\mathbf{y}} - \boldsymbol{\epsilon})\big)^t\big(\tilde{\mathbf{y}} + \boldsymbol{\epsilon}^* - \mathbf{H}(\tilde{\mathbf{y}} - \boldsymbol{\epsilon})\big)\Big] \\ &= E\Big[\tilde{\mathbf{y}}^t(\mathbf{I} - \mathbf{H})\tilde{\mathbf{y}} + \boldsymbol{\epsilon}^{*t}\boldsymbol{\epsilon}^* + \boldsymbol{\epsilon}^t\mathbf{H}\boldsymbol{\epsilon}\Big] \\ &= (n + q + 1)\sigma^2. \end{aligned} \tag{5.8}$$

In deriving the expression on the third line from the preceding, $\mathbf{H}\tilde{\mathbf{y}} = \tilde{\mathbf{y}}$ is assumed. This assumption is not set in the derivation of C_p (Eq. (5.5)).

$\mathbf{H}\tilde{\mathbf{y}} = \tilde{\mathbf{y}}$ is equivalent to the condition that the form of a regression equation either is identical to that of the equation that generates the data or contains that of the equation that generates the data as a special case. As a simple example to illustrate the point, let us assume that the elements of $\tilde{\mathbf{y}}$ obey a quadratic equation. That is, the elements of $\tilde{\mathbf{y}}$ ($\{\tilde{y}_i\}$ $(1 \le i \le n)$) satisfy

$$\tilde{y}_i = \tilde{a}_0 + \tilde{a}_1 x_i + \tilde{a}_2 x_i^2, \tag{5.9}$$

where $\{x_i\}$ $(1 \le i \le n)$ are values of predictors of the data, and $\tilde{a}_0$, $\tilde{a}_1$, and $\tilde{a}_2$ are constants. In this situation, the following cubic equation

$$y = a_0 + a_1 x + a_2 x^2 + a_3 x^3 \tag{5.10}$$

is fitted to the data of $\{(x_i, \tilde{y}_i)\}$ $(1 \le i \le n)$ by the least squares method. Then, we have $\hat{a}_0 = \tilde{a}_0$, $\hat{a}_1 = \tilde{a}_1$, $\hat{a}_2 = \tilde{a}_2$, and $\hat{a}_3 = 0$. As a result, the estimates given by this regression equation are identical to the data ($\{\tilde{y}_i\}$). Hence, $\mathbf{H}\tilde{\mathbf{y}} = \tilde{\mathbf{y}}$ holds because the form of the cubic equation is identical to the form of the equation (the quadratic equation) that generates the data or contains the equation as a special case.

Conversely, when a regression is performed using a cubic equation, $\{\tilde{y}_i\}$ is represented using a cubic equation if $\mathbf{H}\tilde{\mathbf{y}} = \tilde{\mathbf{y}}$ holds. In this instance, the form of cubic equation is identical to that of the equation that generates $\{\tilde{y}_i\}$ or contains the equation as a special case.

Therefore, when a regression is performed by the least squares method, the validity of $\mathbf{H}\tilde{\mathbf{y}} = \tilde{\mathbf{y}}$ is equivalent to the condition that the form of cubic equation is identical to that of the equation that generates the data or contains the equation as a special case. However, as $\tilde{\mathbf{y}}$ is unknown in normal data, it is difficult to determine whether $\mathbf{H}\tilde{\mathbf{y}} = \tilde{\mathbf{y}}$ is satisfied.

If σ^2 is known, the value given by Eq. (5.8) is reduced when q is smaller. This equation requires the same condition as that for Eq. (5.7) (page 226). Hence, when the form of the regression equation is identical to that of the equation that generates the data, errors of the estimates given by data in the future are smaller than those given by a more complicated regression equation that contains the former regression equation as a special case. While Eq. (5.7) is derived with the objective of bringing the estimates close to their true values, Eq. (5.8) is intended to bring the estimates close to the data in the future. These two objectives lead to the same conclusion though. This does not mean, however, that when we know the form of the equation that generates the data, the form of the correct equation is guaranteed to yield the best result in terms of prediction. This conclusion is valid only if $\mathbf{H}\tilde{\mathbf{y}} = \tilde{\mathbf{y}}$ holds. We cannot rule out the possibility that, when we use a regression equation that does not satisfy this equation, we could obtain a better regression equation, in terms of prediction, than the regression equation with the form of the equation that generates the data. In fact, it may happen that when we use data that a 7th-degree polynomial generates, fitting with a quadratic equation gives better results in terms of prediction than fitting with a 7th-degree polynomial (16 page in [11]).

On the other hand, when σ^2 is unknown, the following equation is substituted into Eq. (5.8);

$$\hat{\sigma}^2 = \frac{\sum_{i=1}^{n}(\hat{y}_i - y_i)^2}{n-q-1}. \tag{5.11}$$

The statistic given by this equation is called FPE (247 page in [4]), and is defined as

$$FPE = \frac{n+q+1}{n-q-1}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2. \tag{5.12}$$

FPE assumes that σ^2 is obtained using the regression equation for which we estimate the errors for data in the future; the multiple regression equation with all predictors is not used. This point also differentiates FPE from C_p. Equation (5.12) is useful for automating for example the selection of the order of time-series model. However, we usually do not know whether $\mathbf{H}\tilde{\mathbf{y}} = \tilde{\mathbf{y}}$ holds. Indeed, we find it strange that FPE is used with σ^2 unknown without concern for the validity of this equation in yielding results of practical value. However, when $n \gg q$ holds, an approximate value for FPE can be obtained,

$$\begin{aligned} FPE &= \frac{1+\dfrac{q+1}{n}}{1-\dfrac{q+1}{n}}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2 \\ &\approx \frac{1}{1-\dfrac{2q+2}{n}}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2 \end{aligned}$$

$$\approx \frac{1}{\left(1 - \frac{q+1}{n}\right)^2} \sum_{i=1}^{n} (\hat{y}_i - y_i)^2. \tag{5.13}$$

That is, FPE takes values close to the Generalized Cross-Validation (GCV) (Eqs. (3.98) (page 151) and (4.57) (page 184)) multiplied by n. Hence, FPE works well for the selection of the regression equation regardless of whether $\mathbf{H}\tilde{\mathbf{y}} = \tilde{\mathbf{y}}$ holds.

5.2 AIC of a Multiple Regression Equation with Independent and Identical Normal Distribution

We assume that the data of $\mathbf{X}$ and $\mathbf{y}$ (Eq. (4.4) (page 163)) are given, and the errors in the target variable are realizations of $N(0, \sigma^2)$ (normal distribution; the mean is 0 and the variance is σ^2. σ^2 is also known as the error variance). The likelihood of the multiple regression equation ($a_0 + \sum_{j=1}^{q} a_j x_{ij}$ with $\{a_j\}$ $(0 \leq j \leq q)$ the regression coefficients) in light of $\mathbf{X}$ and $\mathbf{y}$ is expressible as

$$L(\{a_j\}, \sigma^2 | \mathbf{X}, \mathbf{y}) = \left(\frac{1}{\sqrt{2\pi\sigma^2}}\right)^n \exp\left(-\frac{1}{2\sigma^2} \sum_{i=1}^{n} (a_0 + \sum_{j=1}^{q} a_j x_{ij} - y_i)^2\right), \tag{5.14}$$

where $a_0 + \sum_{j=1}^{q} a_j x_{ij}$ gives an estimate corresponding to y_i. $L(\{a_j\}, \sigma^2 | \mathbf{X}, \mathbf{y})$ indicates how likely the regression equation is in light of the data $\mathbf{X}$ and $\mathbf{y}$ when the regression equation is represented using $\{a_j\}$ and σ^2. Large values mean that there is a high possibility that the data are obtained by this regression equation; small values mean that there is a low possibility that the data are obtained by this regression equation. In data analysis, we should not consider the data at hand to have been given by just a coincidental accident but we should consider it to have been given by common practice. Hence, the values of $\{a_j\}$ and σ^2 which lead to a high likelihood are desirable. Therefore, we adjust the values of $\{a_j\}$ and σ^2 to maximize $L(\{a_j\}, \sigma^2 | \mathbf{X}, \mathbf{y})$. This method is called the maximum likelihood procedure (maximum likelihood method).

Because the derivation of conditions to maximize a function is equivalent to that in maximizing the natural logarithm of the function, the natural logarithm of Eq. (5.14) denoted by $l(\{a_j\}, \sigma^2 | \mathbf{X}, \mathbf{y})$ and called the log-likelihood, is more convenient. It is defined as

$$l(\{a_j\}, \sigma^2 | \mathbf{X}, \mathbf{y}) = -\frac{n}{2}\log(2\pi) - \frac{n}{2}\log(\sigma^2) - \frac{1}{2\sigma^2} \sum_{i=1}^{n} (a_0 + \sum_{j=1}^{q} a_j x_{ij} - y_i)^2. \tag{5.15}$$

It is maximized by differentiating with respect to σ^2 and setting the result to zero. We then have

$$\frac{\partial l(\{a_j\}, \sigma^2|\mathbf{X}, \mathbf{y})}{\partial \sigma^2} = -\frac{n}{2\sigma^2} + \frac{1}{2(\sigma^2)^2}\sum_{i=1}^{n}(a_0 + \sum_{j=1}^{q} a_j x_{ij} - y_i)^2 = 0. \quad (5.16)$$

As a result, $\hat{\sigma}^2$ (the estimate of σ^2) is written as

$$\hat{\sigma}^2 = \frac{1}{n}\sum_{i=1}^{n}(a_0 + \sum_{j=1}^{q} a_j x_{ij} - y_i)^2. \quad (5.17)$$

This $\hat{\sigma}^2$ is the result of estimating error variance using the maximum likelihood procedure; that is, this is the maximum likelihood estimator of error variance. This is different from the unbiased estimator (Eq. (3.71) (page 130)). Hence, Eq. (5.15) becomes

$$\begin{aligned} l(\{a_j\}, \hat{\sigma}^2|\mathbf{X}, \mathbf{y}) &= -\frac{n}{2}\log(2\pi) - \frac{n}{2}\log(\hat{\sigma}^2) - \frac{n}{2} \\ &= -\frac{n}{2}\log(2\pi) - \frac{n}{2}\log\Big(\frac{1}{n}\sum_{i=1}^{n}(a_0 + \sum_{j=1}^{q} a_j x_{ij} - y_i)^2\Big) - \frac{n}{2}. \quad (5.18) \end{aligned}$$

Maximizing $l(\{a_j\}, \hat{\sigma}^2|\mathbf{X}, \mathbf{y})$ is equivalent to minimizing $\hat{\sigma}^2$. That is, when the likelihood is defined as in Eq. (5.14) (page 229), the calculation of $\{a_j\}$ using the maximum likelihood procedure is the same as using the least squares method. Moreover, $\hat{\sigma}^2$ given by $\{\hat{a}_j\}$ (estimates of $\{a_j\}$) which are obtained by the least squares method is the maximum likelihood estimator of σ^2. Therefore, denoting by $l(\{\hat{a}_j\}, \hat{\sigma}^2|\mathbf{X}, \mathbf{y})$ those values for $l(\{a_j\}, \hat{\sigma}^2|\mathbf{X}, \mathbf{y})$ calculated using $\{\hat{a}_j\}$ that have been obtained by the least squares method, then $l(\{\hat{a}_j\}, \hat{\sigma}^2|\mathbf{X}, \mathbf{y})$ can be written as

$$l(\{\hat{a}_j\}, \hat{\sigma}^2|\mathbf{X}, \mathbf{y}) = -\frac{n}{2}\log(2\pi) - \frac{n}{2}\log\Big(\frac{1}{n}\sum_{i=1}^{n}(\hat{a}_0 + \sum_{j=1}^{q} \hat{a}_j x_{ij} - y_i)^2\Big) - \frac{n}{2}, \quad (5.19)$$

which represents the log-likelihood of $\{\hat{a}_j\}$ and $\hat{\sigma}^2$ in light of the data at hand (data in hand; $\mathbf{X}, \mathbf{y}$); $\{\hat{a}_j\}$ and $\hat{\sigma}^2$ are obtained using data at hand $(\mathbf{X}, \mathbf{y})$. However, when the data of the same number of those at hand are obtained by sampling from the same population with the same conditions, the resultant log-likelihood is not likely to be close to $l(\{\hat{a}_j\}, \hat{\sigma}^2|\mathbf{X}, \mathbf{y})$. Whereas the distribution of the population does not change, the data at hand can vary from one sampling to another. Hence, these regression coefficients ($\{\hat{a}_j\}$ and $\hat{\sigma}^2$) are pulled toward the direction in which the data at hand look likely from $\{a_j\}$ and σ^2 in population. Hence, the likelihood of $\{\hat{a}_j\}$ and $\hat{\sigma}^2$ in light of the emerging data derived from the same population is usually poorer than that in light of the original data at hand. Therefore, the log-likelihood of $\{\hat{a}_j\}$ and $\hat{\sigma}^2$ in light of the emerging data is usually smaller than that in light of the original data at hand.

Then, the data in the future, labeled $\mathbf{X}$ and $\mathbf{y}^*$ (the number of data being n), are assumed to be obtained using the function $y_i^* = a_0 + \sum_{j=1}^{q} a_j x_{ij} + \epsilon^*$ (ϵ^* is a realization of $N(0.0, \sigma^2)$) given in the same manner as that of the data at hand. The values of predictors of the data are the same as those of the data at hand ($\mathbf{X}$). If this is the case, the likelihood of $\{\hat{a}_j\}$ and $\hat{\sigma}^2$ in light of $\mathbf{X}$ and $\mathbf{y}^*$ is represented by the following equation in a fashion similar to Eq. (5.15) (page 229);

$$l(\{\hat{a}_j\}, \hat{\sigma}^2|\mathbf{X}, \mathbf{y}^*) = -\frac{n}{2}\log(2\pi) - \frac{n}{2}\log(\hat{\sigma}^2) - \frac{1}{2\hat{\sigma}^2}\sum_{i=1}^{n}(\hat{a}_0 + \sum_{j=1}^{q}\hat{a}_j x_{ij} - y_i^*)^2. \tag{5.20}$$

Multiplying both sides by (−2), we then have

$$-2l(\{\hat{a}_j\}, \hat{\sigma}^2|\mathbf{X}, \mathbf{y}^*) = n\log(2\pi) + n\log(\hat{\sigma}^2) + \frac{1}{\hat{\sigma}^2}\sum_{i=1}^{n}(\hat{a}_0 + \sum_{j=1}^{q}\hat{a}_j x_{ij} - y_i^*)^2. \tag{5.21}$$

Future data are required for obtaining $l(\{\hat{a}_j\}, \hat{\sigma}^2|\mathbf{X}, \mathbf{y}^*)$; the data at hand alone cannot give this value. However, if there is simple relationship between $l(\{\hat{a}_j\}, \hat{\sigma}^2|\mathbf{X}, \mathbf{y}^*)$ and $l(\{\hat{a}_j\}, \hat{\sigma}^2|\mathbf{X}, \mathbf{y})$, $l(\{\hat{a}_j\}, \hat{\sigma}^2|\mathbf{X}, \mathbf{y}^*)$ is estimated using $l(\{\hat{a}_j\}, \hat{\sigma}^2|\mathbf{X}, \mathbf{y})$. Then, we assume

$$d = 2\Bigl(l(\{\hat{a}_j\}, \hat{\sigma}^2|\mathbf{X}, \mathbf{y}) - E[l(\{\hat{a}_j\}, \hat{\sigma}^2|\mathbf{X}, \mathbf{y}^*)]\Bigr). \tag{5.22}$$

The factor of 2 arises from convention. The historical background for this is that as the constant part of the essential term given by the Taylor expansion of the log-likelihood is $\left(-\frac{1}{2}\right)$, this factor is removed in the subsequent development. $E[l(\{\hat{a}_j\}, \hat{\sigma}^2|\mathbf{X}, \mathbf{y}^*))]$ indicates the mean of various values of $l(\{\hat{a}_j\}, \hat{\sigma}^2|\mathbf{X}, \mathbf{y}^*))$, each of which results from one set of data in the future. Specifically, it is the expectation (expected value) of $l(\{\hat{a}_j\}, \hat{\sigma}^2|\mathbf{X}, \mathbf{y}^*))$. The expectation is calculated here because the errors in $\mathbf{y}^*$ (data in the future) take various values. The likelihood of the resultant regression equation in light of the data in the future is usually considered to be smaller than that in light of the data at hand. Therefore, this d is expected to be positive in many situations. When the form of the regression equation is identical to that of the equation that generates the data or contains the equation as a special case (or such relations approximately hold), the approximate equation below holds (proof to be given later);

$$d \approx 2(q + 2). \tag{5.23}$$

This relation holds even when a regression equation is not a multiple regression equation or the data obey a distribution other than a normal distribution so long as the number of regression coefficients contained in a regression equation is $(q + 1)$ [2, 4]. In fact, when the data follow a distribution other than a normal distribution, the estimation of the error variance or similar procedure might not be needed. That is, whereas the shape of the probability density function of a normal distribution is determined by two values, the mean and the variance, that of another distribution is determined by one value. In this situation, the number of regression coefficients to be estimated is smaller by one; that is,

$$d \approx 2(q+1). \tag{5.24}$$

Equations (5.22) and (5.23) lead to

$$-2E[l(\{\hat{a}_j\}, \hat{\sigma}^2|\mathbf{X}, \mathbf{y}^*))] \approx -2l(\{\hat{a}_j\}, \hat{\sigma}^2|\mathbf{X}, \mathbf{y}) + 2(q+2). \tag{5.25}$$

As with $E[l(\{\hat{a}_j, \hat{\sigma}^2|\mathbf{X}, \mathbf{y}^*))]$, if the log-likelihood of the regression equation obtained by data at hand is estimated in light of the data in the future, the log-likelihood is called the expected log-likelihood (page 35 in [4]).

The left hand side of Eq. (5.25) determines the definition of the Akaike's Information Criterion (AIC). It can also be written as

$$AIC = -2l(\{\hat{a}_j\}, \hat{\sigma}^2|\mathbf{X}, \mathbf{y}) + 2(q+2). \tag{5.26}$$

Substitution of Eq. (5.18) (page 230) into this equation results in

$$AIC = n\log(2\pi) + n\log\Big(\frac{1}{n}\sum_{i=1}^{n}(\hat{a}_0 + \sum_{j=1}^{q}\hat{a}_j x_{ij} - y_i)^2\Big) + n + 2(q+2). \tag{5.27}$$

Another definition of AIC found in older literature is

$$AIC = -2l(\{\hat{a}_j\}, \hat{\sigma}^2|\mathbf{X}, \mathbf{y}) + 2(q+1). \tag{5.28}$$

As this definition is obtained using a less precise approximation, Eq. (5.26) is preferable.

Let us conduct a simulation to ascertain whether Eq. (5.23) actually holds. First, let us assume that data are given by a constant plus an error; the error obeys a normal distribution. The simulation data are realizations of $N(1, 5^2)$ (normal distribution with mean 1 and variance 5^2). The number of data is 100. These data are fitted to a constant. Namely, the average of these data is the estimate. The log-likelihood in light of data at hand for this situation is given by Eq. (5.18) (page 230). To calculate the expectation of the value yielded by Eq. (5.20) (page 231), 100 sets of data in the future are generated in the same manner as that of the simulation data (data at hand) described above. The value given by the following equation is calculated for each set of data in the future. The average of the resultant values is regarded as the approximation of the expectation.

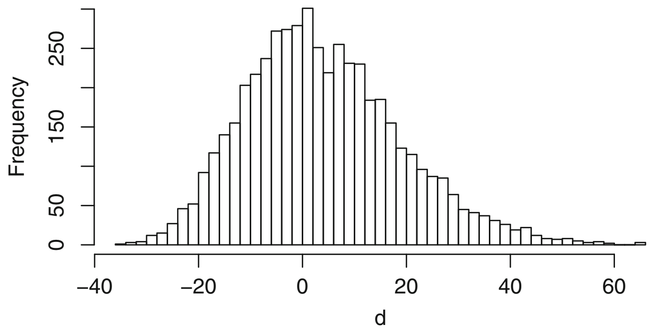


Fig. 5.1 Distribution of d (Eq. (5.22)) given by a simulation performed 5,000 times in which 100 simulation data are obtained using $y = 1 + \epsilon$ and regression is conducted with constant regression equation $y = a_0$. The mean of the distribution is 4.042123

$$l(\hat{a}_0, \hat{\sigma}^2 | \mathbf{X}, \mathbf{y}^*) = -\frac{n}{2}\log(2\pi) - \frac{n}{2}\log(\hat{\sigma}^2) - \frac{1}{2\hat{\sigma}^2}\sum_{i=1}^{n}(\hat{a}_0 - y_i^*)^2. \tag{5.29}$$

This simulation is repeated 5,000 times by varying the initial values of the pseudo-random numbers. The resulting distribution of d (Eq. (5.22) (page 231)) is shown in Fig. 5.1. As the average of the d values turned out to be 4.042123, Eq. (5.23) (page 231) with $q = 0$ holds.

Next, 100 simulation data ($\{(x_i, y_i)\}$ $(1 \leq i \leq n)$) were generated; they consist of the values of the predictors $\{0.1, 0.2, 0.3, \ldots, 10\}$ and those of the target variables subject to the following

$$y_i = 3.9x_i + 1 + \epsilon_i, \tag{5.30}$$

where $\{\epsilon_i\}$ are realizations of $N(0, 1^2)$ (normal distribution with mean 0 and variance 1^2). A linear equation is fitted to these data. The log-likelihood for this case is represented by Eq. (5.18) (page 230) with setting $q = 1$. To calculate the expectation of the value given by Eq. (5.20) (page 231), 100 sets of data in the future were generated in the same manner as that of the data at hand noted previously and values from Eq. (5.20) (page 231) were obtained. The average of these values is used as an approximation of the expectation. The distribution of d (Eq. (5.22) (page 231)), drawn in Fig. 5.2, was given by a simulation that was performed 5,000 times by varying the initial values of the pseudo-random numbers. As the average of the values of d turned out to be 6.127781, Eq. (5.23) (page 231) holds assuming $q = 1$.

Alternatively, the simulation data used in Fig. 5.2 can be fitted to a constant instead of a linear equation. The result, illustrated in Fig. 5.3, gives an average of 0.0572068 for this distribution. Clearly, Eq. (5.23) (page 231) with $q = 0$ does not hold, as the form of the regression equation is not identical to that of the equation that generated the data and does not contain that of the equation that generated the data as a special case, either.

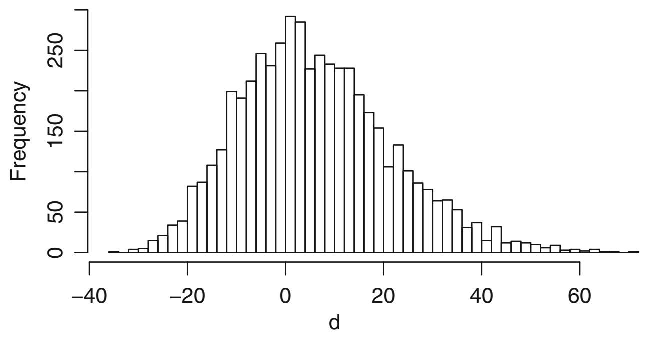


Fig. 5.2 Distribution of d (Eq. (5.22)) given by a simulation performed 5,000 times in which 100 simulation data are obtained using $y = 3.9x + 1 + \epsilon$ and regression is performed with the regression equation ($y = a_0 + a_1x$). The mean of the distribution is 6.127781

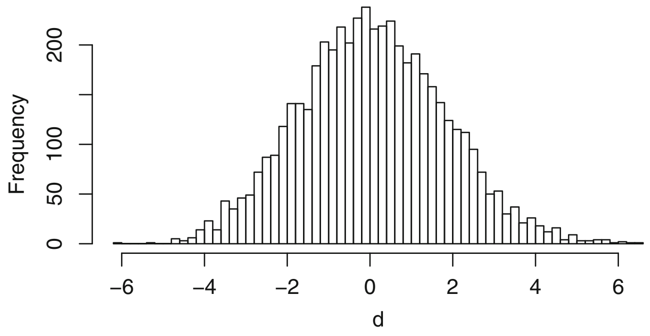


Fig. 5.3 Distribution of d (Eq. (5.22)) given by 5,000 times simulation in which 100 simulation data are obtained using $y = 3.9x + 1 + \epsilon$ and the data are modeled by the regression equation ($y = a_0$). The mean of the distribution is 0.0572068

When the error variance of the population (σ^2) is known, Eq. (5.15) (page 229) is written as

$$l(\{\hat{a}_j|\mathbf{X},\mathbf{y}\}) = -\frac{n}{2}\log(2\pi) - \frac{n}{2}\log(\sigma^2) - \frac{1}{2\sigma^2}\sum_{i=1}^{n}(\hat{a}_0 + \sum_{j=1}^{q}\hat{a}_j x_{ij} - y_i)^2, \quad (5.31)$$

where σ^2 is a parameter representing error variance. Furthermore, Eq. (5.20) (page 231) is represented as

$$l(\{\hat{a}_j\}|\mathbf{X},\mathbf{y}^*) = -\frac{n}{2}\log(2\pi) - \frac{n}{2}\log(\sigma^2) - \frac{1}{2\sigma^2}\sum_{i=1}^{n}(\hat{a}_0 + \sum_{j=1}^{q}\hat{a}_j x_{ij} - y_i^*)^2. \quad (5.32)$$

Multiplying Eqs. (5.31) and (5.32) by factor −2 gives

$$-2l(\{\hat{a}_j\}|\mathbf{X},\mathbf{y}) = n\log(2\pi) + n\log(\sigma^2) + \frac{1}{\sigma^2}\sum_{i=1}^{n}(\hat{a}_0 + \sum_{j=1}^{q}\hat{a}_j x_{ij} - y_i)^2, \quad (5.33)$$

$$-2l(\{\hat{a}_j\}|\mathbf{X},\mathbf{y}^*) = n\log(2\pi) + n\log(\sigma^2) + \frac{1}{\sigma^2}\sum_{i=1}^{n}(\hat{a}_0 + \sum_{j=1}^{q}\hat{a}_j x_{ij} - y_i^*)^2. \quad (5.34)$$

Therefore, Eq. (5.22) (page 231) yields

$$\begin{aligned} d &= -2E[l(\{\hat{a}_j\}|\mathbf{X},\mathbf{y}^*)] + 2l(\{\hat{a}_j\}|\mathbf{X},\mathbf{y}) \\ &= E\Big[\frac{1}{\sigma^2}\sum_{i=1}^{n}(\hat{a}_0 + \sum_{j=1}^{q}\hat{a}_j x_{ij} - y_i^*)^2\Big] - \frac{1}{\sigma^2}\sum_{i=1}^{n}(\hat{a}_0 + \sum_{j=1}^{q}\hat{a}_j x_{ij} - y_i)^2 \\ &\approx (n+q+1) - (n-q-1) \\ &= 2q+2. \end{aligned} \quad (5.35)$$

The first term in the third line of this derivation comes from Eq. (4.35) (page 174). The last term is derived from Eq. (4.39) (page 175). We then obtain

$$-2E[l(\{\hat{a}_j\}|\mathbf{X},\mathbf{y}^*)] \approx -2l(\{\hat{a}_j\}|\mathbf{X},\mathbf{y}) + 2(q+1). \quad (5.36)$$

Hence, using Eq. (5.33) (page 235), AIC for multiple regression is written as

$$\begin{aligned} AIC &= -2l(\{\hat{a}_j\},\hat{\sigma}^2|\mathbf{X},\mathbf{y}) + 2(q+1) \\ &= n\log(2\pi) + n\log(\sigma^2) + \frac{1}{\sigma^2}\sum_{i=1}^{n}(\hat{a}_0 + \sum_{j=1}^{q}\hat{a}_j x_{ij} - y_i)^2 + 2(q+1). \end{aligned} \quad (5.37)$$

Comparison between Eqs. (5.26) and (5.37) implies that $2(q+2)$ on the right-hand side of Eq. (5.25) is based on the fact that σ^2 is considered to be an unknown regression coefficient.

R Program [5 - 1]

The simulation data given by $y = 1 + \epsilon$ (ϵ representing a normally distributed error) are modeled by the regression equation $y = a_0$ in deriving the mean of d (Eq. (5.22) (page 231)) on the assumption that the error variance is unknown.

```
aic41e()
  function ()
  {
  # (1)
```

```
    nd <- 100
    nt <- 5000
  # (2)
    dif <- NULL
  # (3)
    set.seed(2494)
    for(kk in 1:nt){
      yya <- 1 + rnorm(nd, mean = 0, sd = 5)
      ava <- mean(yya)
      sig2 <- sum((ava - yya)^2) / nd
      la <- nd * log(2*pi) + nd * log(sig2) + nd
  # (4)
      lb <- 0
      for(ii in 1:100){
        yyb <- 1 + rnorm(nd, mean = 0, sd = 5)
        sig2b <- sum((ava - yyb)^2) / nd
        lb <- nd * log(2 * pi) + nd * log(sig2) +
         sig2b * nd / sig2 + lb
      }
      lb <- lb/100
  # (5)
      dif[kk] <- lb - la
    }
  # (6)
    print("mean(dif)")
    print(mean(dif))
  # (7)
    par(mfrow = c(1, 1), mai = c(1.5, 1.5, 0.5, 0.5),
     oma = c(1, 1, 1, 1))
    br1 <-  pretty(dif, n=50)
    bw1 <- br1[2] - br1[1]
    difh <- floor(dif/bw1) * bw1 + 0.01 * bw1
    hist(difh, breaks = br1, main = "", xlab = "d",
     ylab = "Frequency")
  }
```

(1) The number of data (`nd`) and the number of times the simulation (`nt`) is performed are given.
(2) `dif` is prepared to store the value d (Eq. (5.22) (page 231)).
(3) After setting the initial value of the pseudo-random numbers, the simulation is performed `nt` times. `yya` stores values of the target variable of the data. `ava` stores $\hat{a}_0$. As this is regression to a constant, $\hat{a}_0$ is the average of the values of the target variable of the data. `sig2` stores $\hat{\sigma}^2$. $-2l(\{\hat{a}_j\}\}, \hat{\sigma}^2|\mathbf{X}, \mathbf{y})$ (Eq. (5.15) (page 229)) is stored in `la`.

(4) The average of the values of $-2l(\{\hat{a}_j\}, \hat{\sigma}^2|\mathbf{X}, \mathbf{y}^*)$ (Eq. (5.21) (page 231)) is stored in `lb`.
(5) The value of d (Eq. (5.22) (page 231)) is stored in `dif[kk]`.
(6) The average of the elements of `dif` is displayed.
(7) The histogram of `dif` is drawn (Fig. 5.1 (page 233)).

`aic41e()` also outputs:

```
"mean(dif)"
4.042123
```

R Program [5 - 1] End

R Program [5 - 2]

The simulation data given by $y = 3.9x + 1 + \epsilon$ (ϵ representing a normally distributed error) are modeled by the regression equation of $y = a_0 + a_1 x$ to derive the mean d (Eq. (5.22) (page 231)) on the assumption that the error variance is unknown. `aic51e()`

```
function ()
{
# (1)
  nd <- 100
  nt <- 5000
# (2)
  dif <- NULL
# (3)
  xx <- seq(from = 0.1, to = 10, length = nd)
  xxm <- matrix( c(rep(1, length = nd), xx), ncol = 2)
  xxq <- xxm %*% solve(t(xxm) %*% xxm) %*% t(xxm)
# (4)
  set.seed(2494)
  for(kk in 1:nt){
    yya <- xx * 3.9 + 1 + rnorm(nd, mean = 0, sd = 1)
    yyam <- matrix(yya, ncol = 1)
    yyhat <- xxq %*% yyam
    sig2 <- sum((yya - yyhat)^2)/nd
    la <- nd * log(2 * pi) + nd * log(sig2) + nd
# (5)
    lb <- 0
    for(ii in 1:100){
      yyb <- xx * 3.9 + 1 + rnorm(nd, mean = 0,
       sd = 1)
      sig2b <- sum((yyhat - yyb)^2) / nd
      lb <- nd * log(2 * pi) + nd * log(sig2)+
       sig2b * nd / sig2 + lb
    }
    lb <- lb/100
# (6)
    dif[kk] <- lb - la
  }
# (7)
  print("mean(dif)")
  print(mean(dif))
# (8)
  par(mfrow = c(1, 1), mai = c(1.5, 1.5, 0.5, 0.5),
   oma = c(1, 1, 1, 1))
  br1 <-  pretty(dif, n = 50)
  bw1 <- br1[2] - br1[1]
  difh <- floor(dif / bw1) * bw1 + 0.01 * bw1
  hist1 <- hist(difh, breaks = br1, main = "",
   xlab = "d", ylab = "Frequency")
}
```

(1) The number of data (nd) and the number of times the simulation (nt) is performed are given,
(2) dif is prepared to store the value d (Eq. (5.22) (page 231)).
(3) The values of the predictor of the data are stored in xx. Transforming it into the form of column vector produces xxm. xxq is defined by removing $\mathbf{y}$ from the right-hand side of Eq. (4.9) (page 164).
(4) After the initial values of the pseudo-random numbers are set, the simulation is performed nt times. The elements of yyhat are estimates corresponding to the target variable of the data. sig2 stands for $\hat{\sigma}^2$. The value of $-2l(\{\hat{a}_j\}, \hat{\sigma}^2|\mathbf{X}, \mathbf{y})$ (Eq. (5.15) (page 229)) is saved as la.
(5) The average of the values of $-2l(\{\hat{a}_j\}, \hat{\sigma}^2|\mathbf{X}, \mathbf{y}^*)$ (Eq. (5.21) (page 231)) is saved as lb.
(6) The value of d (Eq. (5.22) (page 231)) given by each simulation is saved as dif[kk].
(7) The average of the elements of dif is displayed.
(8) The histogram of the elements of dif is drawn (Fig. 5.2 (page 234)).

aic51e() also outputs:

```
  "mean(dif)"
  6.127781
```

R Program [5 - 2] End

R Program [5 - 3]

The simulation data given by $y = 3.9x + 1 + \epsilon$ (ϵ representing a normally distributed error) are modeled by the regression equation of $y = a_0$ to derive the mean of d (Eq. (5.22) (page 231)) on the assumption that the error variance is unknown.

```
aic56e()
  function ()
  {
  #(1)
    nd <- 100
    nt <- 5000
  # (2)
    dif <- NULL
  # (3)
    xx <- seq(from = 0.1, to = 10, length = nd)
    set.seed(2494)
    for(kk in 1:nt){
      yya <- xx * 3.9 + 1 + rnorm(nd, mean = 0, sd = 1)
      ava <- mean(yya)
      sig2 <- sum((ava - yya)^2) / nd
      la <-  nd*log(2 * pi) + nd * log(sig2) + nd
  # (4)
      lb <- 0
      for(ii in 1:100){
        yyb <- xx * 3.9 + 1 + rnorm(nd, mean = 0,
         sd = 1)
        sig2b <- sum((ava - yyb)^2)/nd
        lb <-  nd * log(2 * pi) +nd * log(sig2) +
         sig2b * nd / sig2 + lb
      }
      lb <- lb / 100
  # (5)
      dif[kk] <- lb - la
    }
  # (6)
    print("mean(dif)")
    print(mean(dif))
  # (7)
    par(mfrow = c(1, 1), mai = c(1.5, 1.5, 0.5, 0.5),
     oma = c(1, 1, 1, 1))
    br1 <-  pretty(dif, n = 50)
    bw1 <- br1[2] - br1[1]
    difh <- floor(dif/bw1) * bw1 + 0.01 * bw1
    hist1 <- hist(difh, breaks = br1, main = "",
     xlab = "d", ylab = "Frequency")
  }
```

(1) The number of data (`nd`) and the number of times the simulation (`nt`) is performed are given,
(2) `dif` is initialized to store d (Eq. (5.22) (page 231)).
(3) After the initial values of the pseudo-random numbers are set, the simulation is performed `nt` times. The elements of `yya` are values of the target variable. `ava` receives values of $\hat{a}_0$ and `sig2` that of $\hat{\sigma}^2$. The value of $-2l(\{\hat{a}_j\}\}, \hat{\sigma}^2|\mathbf{X}, \mathbf{y})$ (Eq. (5.15) (page 229)) is stored in `la`.
(4) The average of the values of $-2l(\{\hat{a}_j\}\}, \hat{\sigma}^2|\mathbf{X}, \mathbf{y}^*)$ (Eq. (5.21) (page 231)) is saved as `lb`.
(5) The value of d (Eq. (5.22) (page 231)) is saved as `dif[kk]`.
(6) The average of the elements of `dif` is displayed.
(7) The histogram of the elements of `dif` is drawn (Fig. 5.3 (page 234)).

`aic56e()` also outputs:

```
  "mean(dif)"
  0.0572068
```

R Program [5 - 3] End

5.3 Derivation of *AIC* for Multiple Regression

Equation (4.32) (page 174) shows that the expectation given in Eq. (4.3) (page 163) is

$$E[RSS] = E[(\mathbf{y} - \hat{\mathbf{y}})^t(\mathbf{y} - \hat{\mathbf{y}})] \\ = E[\boldsymbol{\epsilon}^t\boldsymbol{\epsilon} - \boldsymbol{\epsilon}^t\mathbf{H}\boldsymbol{\epsilon}]. \tag{5.38}$$

Then, Eq. (4.36) (page 175) gives the following relation,

$$RSS \sim \chi^2_{n-q-1}, \tag{5.39}$$

where χ^2_{n-q-1} is a random variable which obeys the χ^2-distribution with $(n-q-1)$ degrees of freedom.

On the other hand, $-2l(\{\hat{a}_j\}, \hat{\sigma}^2|\mathbf{X}, \mathbf{y}^*)$ (Eq. (5.21) (page 231)) for the multiple regression is expressed as

$$-2l(\{\hat{a}_j\}, \hat{\sigma}^2|\mathbf{X}, \mathbf{y}^*) = n\log(2\pi) + n\log(\hat{\sigma}^2) + \frac{1}{\hat{\sigma}^2}\sum_{i=1}^{n}\left(\hat{a}_0 + \sum_{j=1}^{q}\hat{a}_j x_{ij} - y_i^*\right)^2 \\ = n\log(2\pi) + n\log\left(\frac{RSS}{n}\right) + \frac{nRSS^*}{RSS}, \tag{5.40}$$

where RSS^* is defined as

$$RSS^* = \sum_{i=1}^{n}(\hat{a}_0 + \sum_{j=1}^{q}\hat{a}_j x_{ij} - y_i^*)^2. \tag{5.41}$$

Therefore, in view of Eqs. (5.39) (page 240) and (4.40) (page 176), the third term on the right-hand side of Eq. (5.40) is written as

$$\frac{nRSS^*}{RSS} = \frac{n\chi^2_{n+q+1}}{\chi^2_{n-q-1}} = n\frac{n+q+1}{n-q-1}F_{n+q+1,n-q-1}, \tag{5.42}$$

where χ^2_{n+q+1} is a random variable which obeys the χ^2-distribution with $(n+q+1)$ degrees of freedom. As the errors of the data at hand are independent of those of the data in the future, χ^2_{n-q-1} is independent of χ^2_{n+q+1} in Eq. (5.42). $F_{n+q+1,n-q-1}$ is a random variable which obeys the F-distribution with the first degree of freedom of $(n+q+1)$ and the second degree of freedom of $(n-q-1)$. Using Eq. (2.68) (page 91), the expectation of the random variable given by Eq. (5.42) is represented as

$$E\left[n\frac{n+q+1}{n-q-1}F_{n+q+1,n-q-1}\right] = n\frac{n+q+1}{n-q-1}\cdot\frac{n-q-1}{n-q-1-2} = n\frac{n+q+1}{n-q-3}. \tag{5.43}$$

Hence, Eq. (5.40) is written as

$$\begin{aligned} -2E[l(\{\hat{a}_j\},\hat{\sigma}^2|\mathbf{X},\mathbf{y}^*)] &= n\log(2\pi)+n\log(\hat{\sigma}^2)+\frac{1}{\hat{\sigma}^2}\sum_{i=1}^{n}\left(a_0+\sum_{j=1}^{q}a_j x_{ij}-y_i^*\right)^2 \\ &= n\log(2\pi)+n\log\left(\frac{RSS}{n}\right)+n\frac{n+q+1}{n-q-3}. \end{aligned} \tag{5.44}$$

This expression on the right-hand side is denoted by AIC_c (the subscripted “c” meaning “corrected”). Thus, we have [3, 10]

$$AIC_c = n\log(2\pi) + n\log\left(\frac{RSS}{n}\right) + n\frac{n+q+1}{n-q-3}. \tag{5.45}$$

If n is large, the third term is approximated as

$$n\frac{n+q+1}{n-q-3} = n\frac{1+\dfrac{q+1}{n}}{1-\dfrac{q+3}{n}} \approx n\left(1+\frac{q+1}{n}+\frac{q+3}{n}\right) \approx n\left(1+\frac{2q+4}{n}\right). \tag{5.46}$$

Now we have Eq. (5.25) (page 232). Therefore, using Eq. (5.18) (page 230), Eq. (5.40) (page 240) is approximately

$$-2E[l(\{a_j\}, \hat{\sigma}^2|\mathbf{X}, \mathbf{y}^*)] \approx -2l(\{a_j\}, \hat{\sigma}^2|\mathbf{X}, \mathbf{y}) + 2q + 4. \tag{5.47}$$

This equation is identical to Eq. (5.26) (page 232). Hence, the AIC defined below is approximately the expectation of minus two times the log-likelihood in light of the data in the future.

$$AIC = n\log(2\pi) + n\log\left(\frac{RSS}{n}\right) + n + 2q + 4. \tag{5.48}$$

Note, however, that this equation is derived on the assumption that the form of the regression equation either is identical to the equation that generates the data or contains the equation as a special case. This does not mean that, when several regression equations are compared using AIC, the containment relation between these is required. It actually means that the containment relation between the form of the regression equation and that of the equation that generates the data must satisfy the condition on which AIC is based.

The significance of AIC lies in the fact that the expectation of the log-likelihood in light of the data in the future is approximated using only data at hand. Its validity though is limited to situations where the form of the regression equation either is identical to the equation that generates the data or contains the equation as a special case. When this condition is not satisfied, even approximately, or the fulfillment of the condition is not established, we do not know whether AIC is a good approximation to minus-two times the expectation of the log-likelihood in light of the data in the future. Then, AIC seems tricky to use unless we have a criterion for knowing how this condition is met and the criterion lets us determine whether Eq. (5.25) holds with high accuracy. In practice, if data are generated by what we consider to be major candidates among regression equations given by the data at hand, the distribution of the resultant data is usually similar to that of the data at hand and that of the data in the future. It is, however, difficult to be sure of this. Alternatively, selection of a regression equation using AIC is asymptotically equivalent to that using GCV regardless of whether AIC works as an approximation of minus-two times the expectation of the log-likelihood in light of the data in the future. Hence, the two statistics lead to similar results (page 121 in [11]). Therefore, regardless of whether AIC is a good approximation of minus-two times the expectation of the log-likelihood in light of the data in the future, AIC (Eq. (5.26)) is useful in selecting a regression equation when errors are normally distributed and the error variance is unknown.

As long as AIC and GCV lead approximately to the same results, the major focus will be on which statistics delivers more valid results. It is true that GCV is derived from weaker conditions. It does not indicate, however, that the practical value of GCV is always higher than that of AIC. For example, when a nonparametric probability density function is produced using generalized linear model (GLM), AIC is sometimes a better statistic than GCV in optimizing smoothness.

Furthermore, much literature on AIC states that the measurement of the distance between the true probability density function and that derived from data using the Kullback-Leibler information divergence (Kullback-Leibler divergence) to derive AIC is the key methodology underlying AIC. However, this does not mean that, although there are many methods for measuring the distance between probability density functions, selection of the Kullback-Leibler information divergence for this purpose gives birth to AIC. As is described above, the basic idea in developing AIC is:

(1) We would like to estimate the log-likelihood in light of the data in the future using the log-likelihood given by the probability density function which is derived from the data at hand by the maximum likelihood procedure.
(2) Unfortunately, we do not have the data in the future.
(3) Then, the value is approximately estimated using only the data at hand.

The equation with the same form as that of the Kullback-Leibler information divergence emerges necessarily by the definition of log-likelihood.

Moreover, comparison between Eqs. (5.5) (page 226) and (5.37) (page 235) shows that when the error variance is known, AIC is equivalent to C_p. This relation implies that even if AIC does not work effectively as the approximation of the expectation of minus-two times the log-likelihood in light of the data in the future, we can use AIC for selecting a regression equation.

That is, even when AIC does not function effectively as the approximation of the expectation of minus-two times the log-likelihood in light of the data in the future, AIC works for selection of a regression function because the effectiveness of AIC is backed by GCV when error variance is unknown and it is backed by C_p when error variance is known. Because we are not sure even of the approximate validity of the assumption that the form of a regression equation is equivalent to that of the equation that generates the data or contains the equation which generates the data as a special case, the theory of AIC alone does not explain its effectiveness for selection of a regression equation when errors are normally distributed.

5.4 *AIC* with Unbiased Estimator for Error Variance

The previous sections use the maximum likelihood estimator of σ^2 (Eq. (5.17) (page 230)) as error variance. The unbiased estimator of σ^2 ($\hat{\sigma}^2_{ub}$), however, is available for this purpose. It is defined as

$$\hat{\sigma}^2_{ub} = \frac{RSS}{n-q-1}. \tag{5.49}$$

Substitution of $\hat{\sigma}^2_{ub}$ into σ^2 in Eq. (5.15) (page 229) yields

$$\begin{aligned} l(\{\hat{a}_j\}, \hat{\sigma}^2_{ub}|\mathbf{X}, \mathbf{y}) &= -\frac{n}{2}\log(2\pi) - \frac{n}{2}\log(\hat{\sigma}^2_{ub}) - \frac{n-q-1}{2} \\ &= -\frac{n}{2}\log(2\pi) - \frac{n}{2}\log\left(\frac{RSS}{n-q-1}\right) - \frac{n-q-1}{2}. \end{aligned} \tag{5.50}$$

Hence, the expectation of minus-two times the log-likelihood in light of the data in the future is

$$\begin{aligned} &-2E[l(\{\hat{a}_j\}, \hat{\sigma}^2_{ub}|\mathbf{X}, \mathbf{y}^*)] \\ &= E\left[n\log(2\pi) + n\log\left(\frac{RSS}{n-q-1}\right) + \frac{(n-q-1)RSS^*}{RSS}\right]. \end{aligned} \tag{5.51}$$

Then, the third term of the right-hand side of Eq. (5.51) becomes

$$\begin{aligned} &E\left[\frac{(n-q-1)RSS^*}{RSS}\right] \\ &= (n-q-1)\frac{n+q+1}{n-q-1}E\left[F_{n+q+1,n-q-1}\right] \\ &= (n-q-1)\frac{n+q+1}{n-q-1}\cdot\frac{n-q-1}{n-q-3} \\ &= (n-q-1)\frac{n+q+1}{n-q-3}. \end{aligned} \tag{5.52}$$

Therefore, Eq. (5.51) is written as

$$\begin{aligned} &-2E[l(\{\hat{a}_j\}, \hat{\sigma}^2_{ub}|\mathbf{X}, \mathbf{y}^*)] \\ &= E\left[n\log(2\pi) + n\log\left(\frac{RSS}{n-q-1}\right) + \frac{(n-q-1)RSS^*}{RSS}\right] \\ &= n\log(2\pi) + n\log\left(\frac{RSS}{n-q-1}\right) + \frac{(n-q-1)(n+q+1)}{n-q-3}. \end{aligned} \tag{5.53}$$

As this is equivalent to AIC_c when the unbiased estimator is used as error variance, it is denoted by $AIC_{c'}$ and defined as

$$AIC_{c'} = n\log(2\pi) + n\log\left(\frac{RSS}{n-q-1}\right) + \frac{(n-q-1)(n+q+1)}{n-q-3}. \tag{5.54}$$

For large n, $AIC_{c'}$ approximates to

$$\begin{aligned}
AIC_{c'} &= n\log(2\pi) + n\log\left(\frac{RSS}{n-q-1}\right) + \frac{(n-q-1)(n+q+1)}{n-q-3} \\
&= n\log(2\pi) + n\log\left(\frac{RSS}{n-q-1}\right) + n\frac{\left(1-\dfrac{q+1}{n}\right)\left(1+\dfrac{q+1}{n}\right)}{1-\dfrac{q+3}{n}} \\
&\approx n\log(2\pi) + n\log\left(\frac{RSS}{n-q-1}\right) + n\left(\left(1-\frac{(q+1)^2}{n^2}\right)+\frac{q+3}{n}\right) \\
&= n\log(2\pi) + n\log\left(\frac{RSS}{n-q-1}\right) + n\left(1+\frac{q+3}{n}\right) \\
&= n\log(2\pi) + n\log\left(\frac{RSS}{n-q-1}\right) + n + q + 3. \qquad (5.55)
\end{aligned}$$

Hence, denoting $AIC_{c'}$ for large n as AIC', AIC' becomes

$$AIC' = n\log(2\pi) + n\log\left(\frac{RSS}{n-q-1}\right) + n + q + 3. \qquad (5.56)$$

Substitution of Eq. (5.50) (page 244) into this equation gives

$$AIC' = -2l(\{\hat{a}_j\}, \hat{\sigma}^2_{ub}|\mathbf{X}, \mathbf{y}) + 2q + 4. \qquad (5.57)$$

Note that although the form of Eq. (5.47) is the same as that of Eq. (5.57), their interpretation differs because $l(\{\hat{a}_j\}, \hat{\sigma}^2|\mathbf{X}, \mathbf{y})$ for AIC is different from $l(\{\hat{a}_j\}, \hat{\sigma}^2_{ub}|\mathbf{X}, \mathbf{y})$ for AIC'.

5.5 Error Variance by Maximizing Expectation of Log-Likelihood in Light of the Data in the Future and the "Third Variance"

As described above, whereas AIC and AIC_c use the maximum likelihood estimator as error variance, AIC' and $AIC_{c'}$ use the unbiased estimator. Therefore, let us entertain the possibility that another value is employed as error variance. This value of error variance, denoted by σ^2_{AIC}, is defined as

$$\sigma^2_{AIC} = \frac{RSS}{n-\alpha}, \qquad (5.58)$$

where α is a constant. Substitution of this σ^2_{AIC} into σ^2 in Eq. (5.15) (page 229) yields

$$l(\{\hat{a}_j\}, \hat{\sigma}^2_{AIC}|\mathbf{X}, \mathbf{y}) = -\frac{n}{2}\log(2\pi) - \frac{n}{2}\log(\hat{\sigma}^2_{AIC}) - \frac{n-\alpha}{2}$$
$$= -\frac{n}{2}\log(2\pi) - \frac{n}{2}\log\left(\frac{RSS}{n-\alpha}\right) - \frac{n-\alpha}{2}. \quad (5.59)$$

Using this result, a similar calculation to that for the derivation of AIC_c (Eq. (5.45) (page 241)) is conducted. Then, we have AIC^a_c (the "adjustable AIC"), defined as

$$AIC^a_c = n\log(2\pi) + n\log\left(\frac{RSS}{n-\alpha}\right) + (n-\alpha)\frac{n+q+1}{n-q-3}. \quad (5.60)$$

Hence, the $\hat{\alpha}$ that minimizes AIC^a_c is

$$\hat{\alpha} = n\left(1 - \frac{n-q-3}{n+q+1}\right). \quad (5.61)$$

Substitution of this $\hat{\alpha}$ into Eq. (5.60) gives AIC^u_c (the "ultimate AIC") [12], which is defined as

$$AIC^u_c = n\log(2\pi) + n\log\left(\frac{RSS}{n - n\left(1 - \frac{n-q-3}{n+q+1}\right)}\right)$$
$$+\left(n - n\left(1 - \frac{n-q-3}{n+q+1}\right)\right)\frac{n+q+1}{n-q-3}$$
$$= n\log(2\pi) + n\log\left(\frac{RSS(n+q+1)}{n-q-3}\right) + n. \quad (5.62)$$

The essential part of AIC^u_c is $\frac{RSS(n+q+1)}{n-q-3}$ which is similar to FPE (Eq. (5.12) (page 228)).

Furthermore, substitution of Eq. (5.61) into Eq. (5.58) leads to

$$\hat{\sigma}^2_{AIC} = \frac{RSS}{n - n\left(1 - \frac{n-q-3}{n+q+1}\right)}$$
$$= \frac{RSS(n+q+1)}{n(n-q-3)}. \quad (5.63)$$

If n (the number of data) is large, $\hat{\sigma}^2_{AIC}$ becomes

$$\begin{aligned}\hat{\sigma}^2_{AIC} &= \frac{RSS\left(1+\dfrac{q+1}{n}\right)}{n\left(1-\dfrac{q+3}{n}\right)} \\ &\approx \frac{RSS}{n\left(1-\dfrac{q+3}{n}-\dfrac{q+1}{n}\right)} \\ &= \frac{RSS}{n\left(1-\dfrac{2q+4}{n}\right)} \\ &= \frac{RSS}{n-2q-4}. \end{aligned} \tag{5.64}$$

Setting $q = 0$, namely, performing a fit of a normal distribution to several data, the resultant $\hat{\sigma}^2_{AIC}$ denoted by $\hat{\sigma}^2_{third}$, is written as

$$\hat{\sigma}^2_{third} = \frac{RSS}{n-4}. \tag{5.65}$$

This $\hat{\sigma}^2_{third}$ for historical reasons is called the "third variance" [12]; its discovery followed error variances of the maximum likelihood estimator and the unbiased estimator.

Whereas the maximum likelihood estimator of error variance is based on the maximization of the log-likelihood in light of the data at hand, the unbiased estimator of error variance is based on the intention that the average of the variances given by several samples from a population should be equal to the true variance. In contrast, whereas the third variance is on equal footing with the unbiased estimator of error variance in the sense that another sampling from the population is assumed, it pays little attention to unbiasedness but is produced by maximization of the expectation of the log-likelihood in light of the data in the future given by other samplings. The variance derived by this methodology is a little larger than the unbiased estimator of error variance.

Let us describe a simulation to establish that the third variance (Eq. (5.65)) minimizes the expectation of minus-two times the log-likelihood in light of the data in the future. As this simulation fits the data to a normal distribution, Eq. (5.40) (page 240) is replaced with the following:

$$-2l(\hat{a}_0, \hat{\sigma}^2 | \{y_i^*\}) = n\log(2\pi) + n\log(\hat{\sigma}^2) + \frac{1}{\hat{\sigma}^2}\sum_{i=1}^{n}(\hat{a}_0 - y_i^*)^2$$

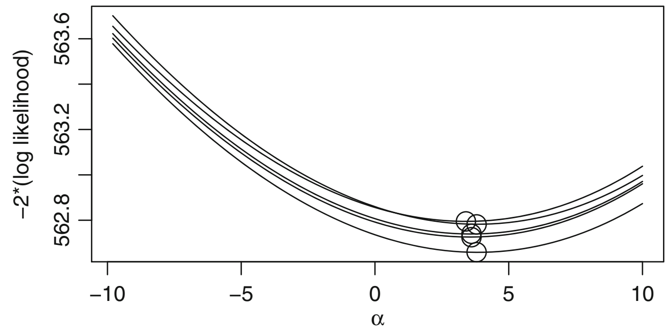


Fig. 5.4 Relationship between α and the average of $-2l(\hat{a}_0, \hat{\sigma}^2|\mathbf{X}, \mathbf{y}^*)$ in this simulation. The symbol *open circle* shows the minimum points of the averages of $-2l(\hat{a}_0, \hat{\sigma}^2|\mathbf{X}, \mathbf{y}^*)$ for each of the five simulations

$$= n\log(2\pi) + n\log\left(\frac{RSS}{n-\alpha}\right) + \frac{(n-\alpha)RSS^*}{RSS}. \quad (5.66)$$

The data at hand in this simulation are $\{y_i\}$ $(1 \le i \le 100)$ (realizations of $N(-13.0, 4^2)$); the data in the future are $\{y_i^*\}$ $(1 \le i \le 100)$ (realizations of $N(-13.0, 4^2)$). $\hat{a}_0$, $\hat{\sigma}^2$, RSS, and RSS^* are defined as

$$\hat{a}_0 = \frac{\sum_{i=1}^{n} y_i}{n}, \qquad \hat{\sigma}^2 = \frac{RSS}{n-\alpha}, \quad (5.67)$$

$$RSS = \sum_{i=1}^{n} \frac{(y_i - \hat{a}_0)^2}{n}, \qquad RSS^* = \sum_{i=1}^{n} \frac{(y_i^* - \hat{a}_0)^2}{n}, \quad (5.68)$$

where in this simulation $n = 100$, and 2,000 sets $\{y_i\}$ and $\{y_i^*\}$ are obtained by varying the initial values of the pseudo-random numbers. Using these data, $2,000$ values of $-2l(\hat{a}_0, \hat{\sigma}^2|\{y_i^*\})$ are calculated. Then, 2,000 values of $-2l(\hat{a}_0, \hat{\sigma}^2|\{y_i^*\})$ are averaged. This task is performed using α (Eq. (5.58) (page 245)) which is set to one value among the sequence $-9.8, -9.6, -9.4, \ldots, 10$. In this way, we obtain the value of α (Eq. (5.58) (page 245)) that minimizes the expectation of minus-two times the log-likelihood in light of the data in the future. The results from five simulations are graphed in Fig. 5.4. When α is approximately 4, the expectation of minus-two times the log-likelihood in light of the data in the future takes a minimum value. This is a convincing result because Eq. (5.61) (page 246) yields $\alpha = 3.96$.

R Program [5-4]

The existence of the third variance is confirmed using a simulation.

```
aic106()
  function ()
  {
```

```
# (1)
  nd <- 100
  ne <- 100
  nt <- 2000
# (2)
  like2m <- matrix(rep(0, length = 500), ncol = 5)
  like2al <- NULL
  like2min <- NULL
  for (jj in 1:5){
    like2t <- NULL
    alt <- NULL
    for(ii in 1:100){
      alt[ii] <- (ii- 50) * 0.2
      al <- alt[ii]
# (3)
      like2 <- 0
      for (kk in 1:nt){
        set.seed(6961 + kk * 14 + jj * 4)
        yya <- rnorm(nd, mean = -13, sd = 4)
        av1 <- mean(yya)
        sig2a <- sum((yya - av1)^2)/(nd - al)
        set.seed(6201 + kk * 97)
        yyf <- rnorm(ne, mean = -13, sd = 4)
        rss1 <- sum((yyf - av1)^2)
        like2 <-  ne * log(2 * pi) + ne * log(sig2a) +
         rss1 / sig2a + like2
      }
      like2t[ii] <- like2/nt
    }
# (4)
     like2wh <- which(like2t == min(like2t))
     like2al[jj] <- alt[like2wh]
     like2min[jj] <- like2t[like2wh]
     like2m[,jj] <-  like2t
  }
# (5)
  par(mfrow = c(1, 1), mai = c(1, 1, 0.2, 0.2),
   oma = c(1, 1, 1, 1))
  ymin1 <- min(as.vector(like2m))
  ymax1 <- max(as.vector(like2m))
  plot(alt, like2m[,1] , type = "n", xlab =
   expression(alpha), ylab = "-2*(log likelihood)",
   ylim = c(ymin1, ymax1))
  for (jj in 1:5){
    lines(alt, like2m[,jj])
```

```
      points(like2al[jj], like2min[jj], pch = 1,
       cex = 2)
    }
  }
```

(1) The number of simulation data (data at hand) (`nd`) is given. The number of simulation data (data in the future) (`ne`) is given. `nt` (the number of times the simulation is performed) is set to be $2,000$.
(2) A matrix (`like2m`) for storing the average of minus-two times the log-likelihood in light of the data in the future is initialized. A vector (`like2al`) for storing the value of α that minimizes the average of minus-two times the log-likelihood in light of the data in the future is initialized. A vector (`like2min`) for storing the minimum value of the average of minus-two times the log-likelihood in light of the data in the future is initialized. `alt` (the value of α) is set to one of the values $-9.8, -9.6, -9.4, \ldots, 10$.
(3) Realizations of $N(-13.0, 4^2)$ are generated and stored in `yya` ($\{y_i\}$) and `yyf` ($\{y_i^*\}$). The value of $-2l(\hat{a}_0, \hat{\sigma}^2|\{y_i^*\})$ (Eq. (5.66) (page 247)) is calculated and stored as `like2`. The average of the values of `like2` is saved as `like2t`.
(4) The value of `alt` (α) which gives the minimum value of `like2t` is saved as `like2al[jj]`. The minimum value of `like2t` is saved as `like2min[jj]`. The value of `like2t` is stored as `like2m[,jj]`.
(5) The relationship between `alt` and `like2m[,jj]` are graphed. The position given by `like2al[jj]` and `like2min[jj]` is plotted (Fig. 5.4 (page 248)).

R Program [5-4] End

5.6 Relationship Between AIC (or GCV) and F-Test

The selection of a regression equation using a test such as F-test is closely related to that using statistics such as GCV. This relationship reveals some of the characteristics of these statistics.

Let us consider whether the q-th predictor should be added to the multiple regression equation with $(q-1)$ predictors (i.e., $r = 1$ in Eq. (4.54) (page 178)). The F-value for this situation, here denoted by $F(n, q)$, is defined as

$$\begin{aligned} F(n,q) &= \frac{\dfrac{RSS(q-1) - RSS(q)}{1}}{\dfrac{RSS(q)}{n-q-1}} \\ &= (n-q-1)\left(\frac{RSS(q-1)}{RSS(q)} - 1\right), \end{aligned} \tag{5.69}$$

where $RSS(q)$ is defined as

$$RSS(q) = \begin{cases} \sum_{i=1}^{n}(y_i - a_0)^2 & \text{if } q = 0 \\ \sum_{i=1}^{n}(y_i - a_0 - \sum_{j=1}^{q} a_j x_{ij})^2 & \text{if } q \geq 1. \end{cases} \tag{5.70}$$

Equation (5.69) leads to

$$\frac{RSS(q-1)}{RSS(q)} = \frac{F(n,q)}{n-q-1} + 1. \tag{5.71}$$

Furthermore, GCV (Eqs. (3.98) (page 151) and (4.57) (page 184)) is defined as

$$GCV(q) = \frac{RSS(q)}{n\left(1 - \dfrac{q+1}{n}\right)^2}. \tag{5.72}$$

This equation yields

$$\frac{GCV(q)}{GCV(q-1)} = \frac{RSS(q)\cdot\left(1 - \dfrac{q}{n}\right)^2}{RSS(q-1)\cdot\left(1 - \dfrac{q+1}{n}\right)^2}. \tag{5.73}$$

Substitution of Eq. (5.71) into this equation gives

$$\frac{GCV(q)}{GCV(q-1)} = \left(\frac{F(n,q)}{n-q-1} + 1\right)^{-1} \frac{(n-q)^2}{(n-q-1)^2}. \tag{5.74}$$

Thus, when we have a multiple regression equation with $(q-1)$ predictors, the condition determining whether the q-th predictor should be added is written as

$$\left(\frac{F(n,q)}{n-q-1} + 1\right)^{-1} \frac{(n-q)^2}{(n-q-1)^2} < 1. \tag{5.75}$$

Rearranging, we have

$$F(n,q) > (n-q-1)\left(\frac{(n-q)^2}{(n-q-1)^2} - 1\right). \tag{5.76}$$

When the inequality sign is replaced with an equal sign, the graph of $F(n,q)$ can be drawn; Figs. 5.5 (left top) and 5.6 (left top) illustrate F-values for $n = 100$ and $n = 20$, respectively. We find that when we use GCV to determine whether the q-th predictor should be added to the regression equation with $(q-1)$ predictors, the corresponding $F(n,q)$ is almost independent of q.

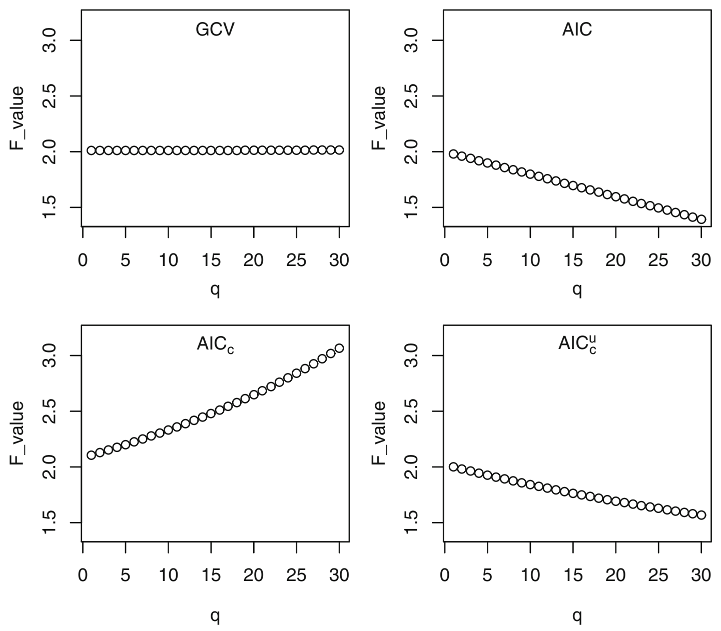


Fig. 5.5 Relationship between q and $F(100, q)$ corresponding to GCV, AIC, AIC_c, or AIC_c^u

When the regression equation with $(q-1)$ predictors is a true regression function, $F(n, q)$ for this situation is written as (Eq. (5.69) (page 250))

$$F(n, q) = (n - q - 1)\left(\frac{RSS(q-1)}{RSS(q)} - 1\right) \sim F_{1,n-q-1}. \tag{5.77}$$

$F_{1,n-q-1}$ stands for the F-distribution with the first degree of freedom set to 1 and the second degree of freedom set to $(n - q - 1)$. Then, p can be derived from

$$p = \int_{F(n,q)}^{\infty} f(1, n - q - 1, x)dx, \tag{5.78}$$

where $f(1, n-q-1, x)$ is the probability density function of the F-distribution with the first degree of freedom set to 1 and the second degree of freedom set to $(n-q-1)$. Here, p stands for the value of integral of the probability density function over the region where x is larger than $F(n, q, p)$. This p is the probability that the F-value takes a larger value than $F(n, q)$ when the regression equation with $(q-1)$ predictor is the true regression equation. Substitution of the values of $F(n, q)$ graphed in Fig. 5.5 (left top) into Eq. (5.78) gives the values of p shown in Fig. 5.7 (left top). The values of p which correspond to the values of $F(n, q)$ graphed in Fig. 5.7 (left top) are shown in Fig. 5.8 (left top). These values of p give the probability that the p-th predictor is erroneously added when the regression equation with $(q - 1)$

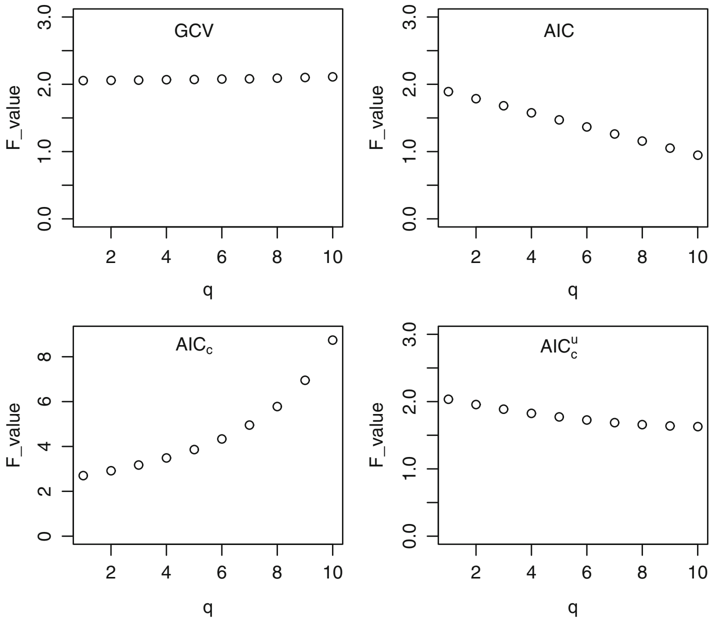


Fig. 5.6 Relationship between q and $F(20, q)$ corresponding to GCV, AIC, AIC_c, or AIC_c^u

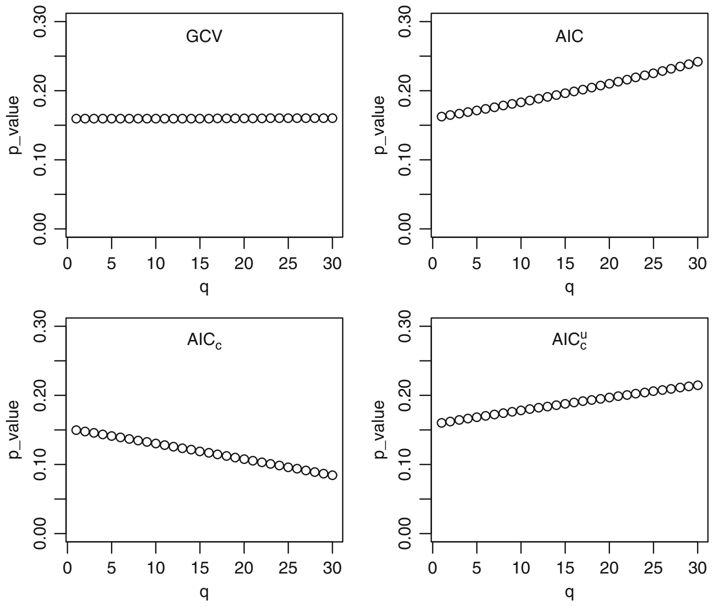


Fig. 5.7 Relationship between q and p corresponding to GCV, AIC, AIC_c or AIC_c^u ($n = 100$)

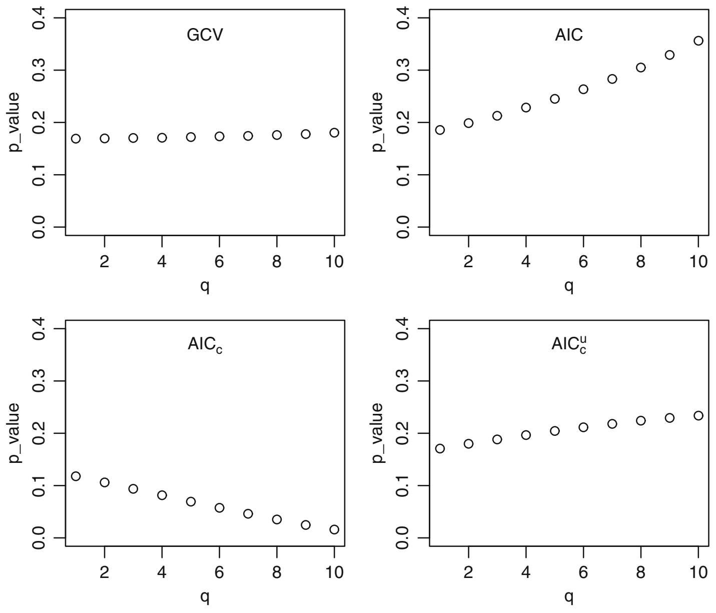


Fig. 5.8 Relationship between q and p corresponding to GCV, AIC, AIC_c, and AIC_c^u ($n = 20$)

predictors is the true regression equation. In other words, it is the probability that the type I error occurs. This probability is fixed in the process of the forward and backward selection method. For example, this value is between 0.25 and 0.5 (e.g., page 188 of [8]), between 0.15 and 0.2, or 0.05 (314 page of [7]). Therefore, because the values of p in Figs. 5.7 (left top) and 5.8 (left top) are almost independent of q, the selection characteristics for predictors using GCV are similar to the forward and backward selection method using the F-value. Figure 5.9 compares the two rejection regions (critical region) : one is the area where the selection of predictors using GCV erroneously adds the p-th predictor to the true regression equation with $(q - 1)$ predictors, and the other is that of the F-test when $p = 0.05$ (p indicates the risk rate) is set.

Next, AIC (Eq. (5.25) (page 232)) for this situation is written as

$$AIC(q) = n\log(2\pi) + n\log\left(\frac{RSS(q)}{n}\right) + n + 2q + 4. \tag{5.79}$$

Hence, we obtain

$$\begin{aligned} AIC(q) - AIC(q-1) &= n\log\left(\frac{RSS(q)}{n}\right) - n\log\left(\frac{RSS(q-1)}{n}\right) + 2 \\ &= n\log\left(\frac{RSS(q)}{RSS(q-1)}\right) + 2. \end{aligned} \tag{5.80}$$

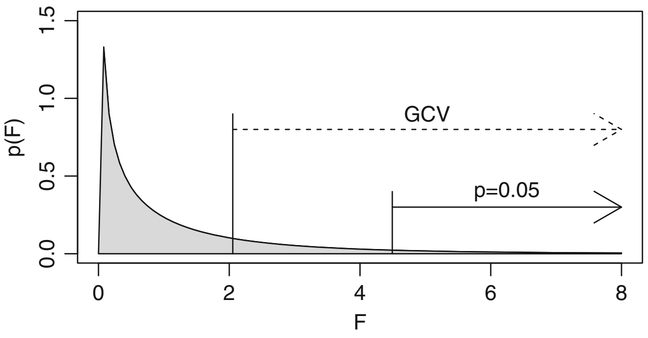


Fig. 5.9 Plot of the probability density function of the F-distribution with the first degree of freedom (ϕ_1) set to 1 and the second degree of freedom (ϕ_1) set to 16. Specifically, the distribution of the F-values on the assumption that the null hypothesis "the number of the predictor is 3" is true (i.e., $F(20, 3)$) is drawn (the number of data (n) is 20). The regions where GCV and the F-test reject this null hypothesis are shown by a *dashed* and *solid arrows*, respectively

Substitution of Eq. (5.71) (page 251) yields

$$AIC(q) - AIC(q-1) = n\log\left(\left(\frac{F(n,q)}{(n-q-1)} + 1\right)^{-1}\right) + 2 < 0. \tag{5.81}$$

Thus, when we have a regression equation with $(q-1)$ predictors, the condition to determine whether the q-predictor should be added is

$$\left(\frac{F(n,q)}{n-q-1} + 1\right)^{-1} < \exp\left(\frac{-2}{n}\right). \tag{5.82}$$

Rearranging gives

$$F(n,q) > (n-q-1)\left(\exp\left(\frac{2}{n}\right) - 1\right). \tag{5.83}$$

Replacing the inequality with an equal sign, values of $F(n,q)$ can be plotted. For $n = 100$ and $n = 20$, the corresponding graphs of $F(n,q)$ are presented in Figs. 5.5 (right top) and 5.6 (right top). The values of p corresponding to values in these graphs are illustrated in Figs. 5.7 (right top) and 5.8 (right top). When AIC is adopted, the value of p increases as q becomes larger; the value of p determines whether the q-th predictor should be added given a regression equation with $(q-1)$ predictors. This tendency shows that the selection of a regression equation using AIC takes a slightly different tack from that in the forward and backward selection method with a fixed value of p.

With AIC_c (Eq. (5.45) (page 241)) defined as

$$AIC_c(q) = n\log(2\pi) + n\log\left(\frac{RSS}{n}\right) + n\frac{n+q+1}{n-q-3}, \tag{5.84}$$

we have

$$\begin{aligned} AIC_c(q) - AIC_c(q-1) &= n\log\left(\frac{RSS(q)}{n}\right) - n\log\left(\frac{RSS(q-1)}{n}\right) \\ &+n\frac{n+q+1}{n-q-3} - n\frac{n+q}{n-q-2} \\ &= n\log\left(\frac{RSS(q)}{RSS(q-1)}\right) + n\frac{n+q+1}{n-q-3} - n\frac{n+q}{n-q-2}. \end{aligned} \tag{5.85}$$

Substitution of Eq. (5.71) (page 251) leads to

$$\begin{aligned} &AIC_c(q) - AIC_c(q-1) \\ &= n\log\left(\left(\frac{F(n,q)}{n-q-1}+1\right)^{-1}\right) + n\frac{n+q+1}{n-q-3} - n\frac{n+q}{n-q-2} < 0. \end{aligned} \tag{5.86}$$

Then, the condition for determining whether the q-th predictor should be added given a regression equation with $(q-1)$ predictors is

$$F(n,q) > \left(\exp\left(\frac{n+q+1}{n-q-3} - \frac{n+q}{n-q-2}\right) - 1\right)(n-q-1). \tag{5.87}$$

Replacing the inequality with an equal sign, values of $F(n,q)$ with the setting of $n = 100$ and $n = 20$ are graphed in Figs. 5.5 (left bottom) and 5.6 (left bottom), respectively. The p values corresponding to the values in these graphs are illustrated in Figs. 5.7 (left bottom) and 5.8 (left bottom). When AIC_c is adopted, the value of p decreases as q becomes larger; the value of p determines whether the q-th predictor should be added given a regression equation with $(q-1)$ predictors. This phenomenon conforms with the well-known tendency that when AIC_c is employed, the criterion for adding a new predictor to a regression equation with many predictors is strict. Moreover, AIC_c take a slightly different tack from that by the forward and backward selection method with p fixed.

Adopting $AIC_c^u(q)$ which is defined as (also refer to Eq. (5.62) (page 246))

$$AIC_c^u(q) = \log(2\pi) + n\log\left(\frac{RSS(q)(n+q+1)}{n-q-3}\right) + n, \tag{5.88}$$

we obtain the following

$$AIC_c^u(q) - AIC_c^u(q-1) = n\log\left(\frac{RSS(q)(n+q+1)}{n-q-3}\right) - n\log\left(\frac{RSS(q-1)(n+q)}{n-q-2}\right). \tag{5.89}$$

That is,

$$AIC_c^u(q) - AIC_c^u(q-1) = n\log\left(\frac{RSS(q)(n+q+1)(n-q-2)}{RSS(q-1)(n-q-3)(n+q)}\right). \tag{5.90}$$

Substitution of Eq. (5.71) leads to

$$AIC_c^u(q) - AIC_c^u(q-1) = n\log\left(\left(\frac{F(n,q)}{n-q-1}+1\right)^{-1}\frac{(n+q+1)(n-q-2)}{(n-q-3)(n+q)}\right). \tag{5.91}$$

Then, given a multiple regression equation with $(q-1)$ predictors, the condition for determining whether to add the q-th predictor is written as

$$F(n,q) > \left(\frac{(n+q+1)(n-q-2)}{(n-q-3)(n+q)} - 1\right)(n-q-1). \tag{5.92}$$

Replacing the inequality with an equal sign, graphs of $F(n,q)$ with $n = 100$ and $n = 20$ are plotted in Figs. 5.5 (right top) and 5.6 (right bottom). The values of p corresponding to the values in these graphs are presented in Figs. 5.7 (right bottom) and 5.8 (right bottom). When AIC_c^u is adopted, the value of p increases as q becomes larger; the value of p determines whether the q-th predictor should be added given a regression equation with $(q-1)$ predictors. This tendency, however, is not as strong as found for AIC but does make a difference for AIC_c. These considerations indicate that the considerable change with AIC_c^u results from the modification of AIC_c by introducing the "third variance".

To understand certain characteristics of GCV, AIC, AIC_c, and AIC_c^u, let us develop an example of a simulation which chooses between a multiple regression equation with one predictor (i.e., simple regression equation) and that with two predictors. The number of data is assumed to be 20 and the values of the both of the predictors ($\{x_{i1}, x_{i2}\}$ $(1 \le i \le n)$) are generated from a distribution of uniform random numbers between -1 and 1. The values of the target variable are calculated using

$$y_i = 5x_{i1} + \gamma x_{i2} + 1 + \epsilon_i, \tag{5.93}$$

where $\{\epsilon_i\}$ $(1 \le i \le n)$ are generated from $N(0, 8^2)$ (normal distribution with mean 0 and variance 8^2). γ is chosen from the set $\{0, 0.1, 0.2, \ldots, 5\}$. An appropriate

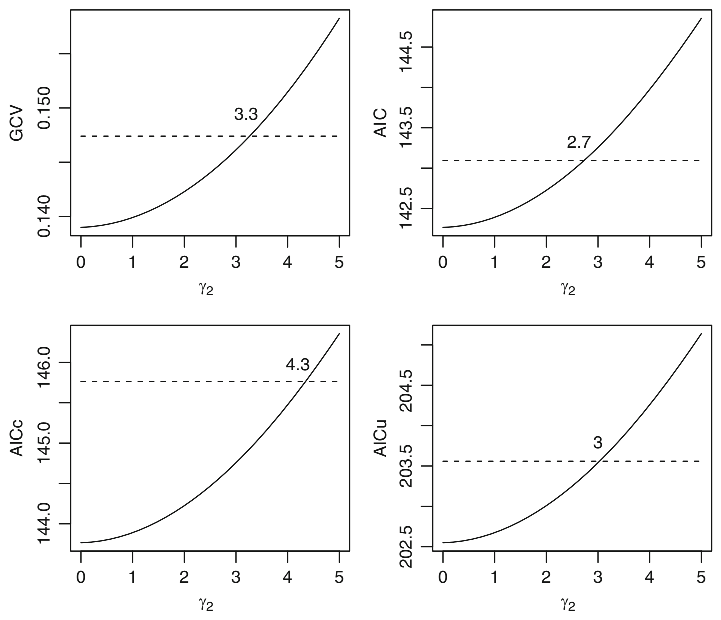


Fig. 5.10 Results of the selection of predictors using GCV, AIC, AIC_c, and AIC_c^u when the number of data is 20. The *solid lines* indicate the values of respective statistics when the number of predictors is one. The *dashed lines* indicate the values of the respective statistics when the number of predictors is two. The approximate values of γ are shown at the intersection of the two lines

regression equation is then selected between that with one predictor and that with two predictors using one of GCV, AIC, AIC_c, and AIC_c^u. This simulation is performed 500 times by varying the initial values of the pseudo-random numbers. The result, shown in Fig. 5.10, implies that AIC adopts a permissive policy with respect to addition of the second predictor, whereas AIC_c takes a strict policy. GCV and AIC_c^u fall somewhere in between the two.

R Program [5 - 5]

A graph is drawn to show how GCV, AIC, AIC_c, and AIC_c^u are related to the value of p of the F-test.

```
aic254e()
  function ()
  {
  # (1)
    nd <- 20
    qq <- seq(from = 1, to = 10, by = 1)
    nq <- length(qq)
  # (2)
    gcv1p <- NULL
    aic1p <- NULL
    aicc1p <- NULL
    aicu1p <- NULL
  # (3)
    for(ii in 1:nq){
      fr1 <- 1
      fr2 <- nd - qq[ii] - 1
  # (4)
      gcv1f <- (nd - qq[ii] - 1) * ((nd - qq[ii])^2 /
       (nd - qq[ii] - 1)^2 - 1)
      gcv1p[ii] <- 1 - pf(gcv1f, fr1, fr2)
      aic1f <- (nd - qq[ii] - 1) * (exp(2 / nd) - 1)
      aic1p[ii] <- 1 - pf(aic1f, fr1, fr2)
      aicc1f <- (exp((nd + qq[ii] + 1)/(nd - qq[ii] -
       3) - (nd + qq[ii])/(nd - qq[ii] - 2)) - 1) *
       (nd - qq[ii] - 1)
      aicc1p[ii] <- 1 - pf(aicc1f, fr1, fr2)
      aicu1f <- ((nd + qq[ii] + 1) * (nd - qq[ii] - 2) /
       ((nd - qq[ii] - 3) * (nd + qq[ii])) - 1) *
       (nd - qq[ii] - 1)
      aicu1p[ii] <- 1 - pf(aicu1f, fr1, fr2)
    }
  # (5)
    par(mfrow = c(2, 2), mai = c(1, 1, 0.2, 0.2),
     oma = c(1, 1, 1, 1))
    plot(qq, gcv1p , xlab = expression(q), ylab =
     "p_value", ylim = c(0,0.4))
    text(5, 0.37, "GCV")
    plot(qq, aic1p , xlab = expression(q), ylab =
     "p_value", ylim = c(0,0.4))
    text(5, 0.37, "AIC")
    plot(qq, aicc1p , xlab = expression(q), ylab =
     "p_value", ylim = c(0, 0.4))
    text(5, 0.37, expression(AIC[c]))
    plot(qq, aicu1p , xlab = expression(q), ylab =
     "p_value", ylim = c(0,0.4))
    text(5, 0.37, expression(AIC[c]^u))
  }
```

(1) The number of data (`nd`), the numbers of predictors (`qq`), and the number of elements of `qq` (`nq`) are given.
(2) `gcv1p` is initialized to store values of p corresponding to $GCV(q)$. `aic1p` is initialized to store values of p corresponding to $AIC(q)$. `aicc1p` is initialized to store values of p corresponding to $AIC_c(q)$. `aicu1p` is initialized to store values of p corresponding to $AIC_c^u(q)$.

(3) q is set to a value in the range 1 to `nq`; the corresponding value of p is calculated. The first degree of freedom is set to `fr1`. The second degree of freedom is set to `fr2`.

(4) The F-value given by Eq. (5.76) is saved as `gcv1f`. Using this value, `gcv1p[ii]` (the value of p) is calculated. Values of $F(n,q)$ are stored in `gcv1f`, df_1 in `fr1`, and df_2 in `fr2`; `pf(gcv1f, fr1, fr2)` takes the value calculated from

$$\tilde{p} = \int_0^{F(n,q)} f(x, df_1, df_2)dx, \tag{5.94}$$

where $f(x, df_1, df_2)$ is the probability density function of the F-distribution with the first degree of freedom of df_1 and the second degree of freedom of df_2. Hence, when $df_1 = 1$ and $df_2 = n - q - 1$ are set, this value along with that of p given by Eq. (5.78) (page 252) satisfies

$$p + \tilde{p} = 1. \tag{5.95}$$

Therefore, `1 - pf(gcv1f, fr1, fr2)` yields the value of p given by Eq. (5.78) (page 252). If `lower.tail = FALSE` is added to the arguments of `pf()`, the value of p is obtained directly. In a similar fashion, the F-value calculated by Eq. (5.83) (page 255) is saved as `aic1f`. Using this value, `aic1p[ii]` (the value of p) is derived. The F-value calculated by Eq. (5.87) (page 256) is saved as `aicc1f`. Using the value, `aicc1p[ii]` (the value of p) is derived. The F-value calculated by Eq.(5.92) (page 257) is saved as `aicu1f`. Using the value, `aicu1p[ii]` (the value of p) is derived.

(5) The values of p obtained in (4) are graphed (Fig. 5.8 (page 254)).

R Program [5 - 5] End

R Program [5 - 6]

GCV yields the boundary to choose between a regression equation with a_1 and a regression equation with a_1 and a_2.

```
aic313()
  function ()
  {
  # (1)
    nd <- 20
    nt <- 500
    nc <-51
    gam1v <-  seq(from = 0, to = 5, length = nc)
  # (2)
    gcv1v <- NULL
    gcv2v <- NULL
  # (3)
    for(jj in 1:nc){
      gcv1 <- 0
      gcv2 <- 0

  # (4)
      for(kk in 1:nt){
        set.seed(237 + kk*14)
        xx1 <- runif(nd, min = -1, max = 1)
        xx2 <- runif(nd, min = -1, max = 1)
        yy <- 5 * xx1 + gam1v[jj] * xx2+ 1 +
         rnorm(nd, mean = 0, sd = 8)
  # (5)
        data1 <- data.frame(x1 = xx1,x2 = xx2, y = yy)
        lm1 <- lm(y~x1, data = data1)
        rss1 <- sum(lm1$residuals^2)
        lm2 <- lm(y~x1 + x2, data = data1)
        rss2 <- sum(lm2$residuals^2)
  # (6)
        gcv1 <- gcv1 + rss1 / (nd * (1 - 2 / nd)^2) / nt
        gcv2 <- gcv2 + rss2 / (nd * (1 - 3 / nd)^2) / nt
    }
  # (7)
    gcv1v[jj] <- gcv1 / nt
    gcv2v[jj] <- gcv2 / nt
  }
  # (8)
    par(mfrow = c(1, 1), mai = c(1, 1, 0.2, 0.2),
     oma = c(1, 1, 1, 1))
    wh1 <- which(abs(gcv1v - gcv2v) ==
     min(abs(gcv1v - gcv2v)) )
    gam2 <- gam1v[wh1]
    plot(gam1v, gcv1v , type = "n", xlab =
     expression(gamma[2]), ylab = "GCV",
     ylim = range(gcv1v, gcv2v))
    lines(gam1v, gcv1v)
    lines(gam1v, gcv2v, lty = 2)
    len1 <- (range(gcv1v)[2] - range(gcv1v)[1]) * 0.1
    text(gam2, gcv1v[wh1] + len1, as.character(gam2))
  }
```

(1) The number of data (`nd`), the number of times the simulation (`nt`) is performed, the number of γ values used in the simulation (`nc`), and the values of γ (`gam1v`) are set.

(2) `gcv1v` receives the values of GCV when a regression equation constructed with only a_1 is prepared. Similarly, `gcv2v` receives the values of GCV when a regression equation constructed with a_1 and a_2 is prepared.

(4) Simulation data are generated. The values of both of the predictors (`xx1`, `xx2`) are generated from a distribution of uniform random numbers between -1 and 1. The uniform random numbers are given by `runif()`. The values of the target variable are calculated using Eq. (5.93) (page 257).

(5) The resulting regression equation with a_1 is stored in `lm1`. The resulting residual sum of squares is saved as `rss1`. The resulting regression equation with a_1 and a_2 is stored in `lm2`. The resulting residual sum of squares is saved as `rss2`.

(6) The values of GCV yielded by the regression equation with a_1 are summed up in `gcv1`. The values of GCV yielded by the regression equation with a_1 and a_2 are summed up in `gcv2`.

(7) The average of the values of GCV given by the regression equation with a_1 is saved as `gcv1v[jj]`. The average of the values of GCV given by the regression equation with a_1 and a_2 is saved as `gcv2v[jj]`.

(8) The averages of the values of GCV when only a_1 is employed and those when a_1 and a_2 are employed are graphed with the respective values of `gam1v`. One value of the elements of `gam1v` is plotted. This value provides the closest estimates of the averages of GCV for the two regression equations. Figure 5.10 (page 258) (left top) is obtained.

R Program [5 - 6] End

5.7 AIC on Poisson Regression

The random variable of Y which obeys the Poisson distribution is given. Its probability density function is written as

$$\begin{aligned} f(y) &= \frac{\exp(-\mu)\mu^y}{y!} \\ &= \exp(y\log(\mu) - \mu - \log(y!)), \end{aligned} \tag{5.96}$$

where "!" stands for factorial and μ denoted by the mean of the Poisson distribution (e.g., page 160 of [9]), that is,

$$E[y] = \mu. \tag{5.97}$$

The data of $\{(x_i, y_i)\}$ $(1 \le i \le n)$ are given. Each y_i is assumed to be a realization of the Poisson distribution. We then have the following:

$$f(y_i) = \exp(y_i \log(\mu_i) - \mu_i - \log(y_i!)), \tag{5.98}$$

where μ_i is the mean of the Poisson distribution which generates the values of y_i.

Next, let us assume that μ_i is written in the form:

$$\mu_i = \exp(a_0 + a_1 x_i). \tag{5.99}$$

Substitution of this equation into Eq. (5.98) results in

$$f(y_i) = \exp(y_i(a_0 + a_1 x_i) - \exp(a_0 + a_1 x_i) - \log(y_i!)). \tag{5.100}$$

Then, the discrete probability density function (also known as the probability mass function or the probability function) which the set $\{y_1, y_2, y_3, \ldots, y_n\}$ obeys is given as

$$f(\{y_1, y_2, y_3, \ldots, y_n\}) = \prod_{i=1}^{n} \exp(y_i(a_0 + a_1 x_i) - \exp(a_0 + a_1 x_i) - \log(y_i!)). \tag{5.101}$$

When $\hat{a}_0$ and $\hat{a}_1$, which are estimates of a_0 and a_1, respectively, are obtained, we have the regression equation:

$$E[y] = \mu = \exp(\hat{a}_0 + \hat{a}_1 x). \tag{5.102}$$

This equation allows us to calculate the expectation of the target variable when an arbitrary value of x is set. Using Eq. (5.101), the log-likelihood of a_0 and a_1 in light of $\{(x_i, y_i)\}$ is written as (e.g., page 132 of [9])

$$l(a_0, a_1 | \{(x_i, y_i)\}) = \sum_{i=1}^{n} (y_i(a_0 + a_1 x_i) - \exp(a_0 + a_1 x_i) - \log(y_i!)). \tag{5.103}$$

The values of a_0 and a_1 are optimized to maximize the above value. As exemplified in the above example, regression based on a Poisson distribution is called Poisson regression. Poisson regression is a sort of generalized linear regression (e.g., [9]).

The values of a_0 and a_1 which maximize Eq. (5.103) are denoted by $\hat{a}_0$ and $\hat{a}_1$, respectively. The value given by substituting $\hat{a}_0$ and $\hat{a}_1$ into Eq. (5.103) is represented as

$$l(\hat{a}_0, \hat{a}_1 | \{(x_i, y_i)\}) = \sum_{i=1}^{n} (y_i(\hat{a}_0 + \hat{a}_1 x_i) - \exp(\hat{a}_0 + \hat{a}_1 x_i) - \log(y_i!)). \tag{5.104}$$

$\{(x_i, y_i)\}$ used in the above equation are the data which are used for obtaining $\hat{a}_0$ and $\hat{a}_1$; namely, these are the data at hand. However, a regression equation that generates highly likely data in the future is usually considered to be of more practical use. With the data in the future denoted by $\{(x_i, y_i^*)\}$ $(1 \leq i \leq n)$, the log-likelihood which shows the likelihood of $\hat{a}_0$ and $\hat{a}_1$ in light of the data in the future is written as

$$l(\hat{a}_0, \hat{a}_1|\{(x_i, y_i^*)\}) = \sum_{i=1}^{n}(y_i^*(\hat{a}_0 + \hat{a}_1 x_i) - \exp(\hat{a}_0 + \hat{a}_1 x_i) - \log(y_i^*!)). \quad (5.105)$$

Hence, if the difference between Eqs. (5.104) and (5.105) is estimated, the log-likelihood in light of the data in the future is estimated using only the data at hand. Then, d which has meaning similar to that in Eq. (5.22) (page 231) is defined as

$$d = 2\Big(l(\hat{a}_0, \hat{a}_1|\{(x_i, y_i)\}) - E[l(\hat{a}_0, \hat{a}_1|\{(x_i, y_i^*)\})]\Big). \quad (5.106)$$

A simulation for estimating this d is shown below. The simulation data are $\{(x_i, y_i)\}$ $(1 \leq i \leq n)$. The number of data (n) is 30. The values of the predictor is given by

$$x_i = i. \quad (5.107)$$

The values of the target variable are the realizations of the Poisson distributions in which the mean is written as

$$\mu_i = \exp(0.2x_i - 4). \quad (5.108)$$

Rearranging gives

$$\log(\mu_i) = 0.2x_i - 4, \quad (5.109)$$

where the logarithmic function transforms the mean into a linear equation $(0.2x_i - 4)$. In the context of generalized linear regression, the function which undertakes this role is called the link function. The values of $\hat{a}_0$ and $\hat{a}_1$ are then estimated using $\{(x_i, y_i)\}$ which are generated by the procedure above. To estimate the value of $E[l(\hat{a}_0, \hat{a}_1|\{(x_i, y_i^*)\})]$ (Eq. (5.106)) using the values available, 100 sets of data in the future ($\{(x_i, y_i)\}$ $(1 \leq i \leq n)$) are generated in the same manner as for $\{(x_i, y_i)\}$. The values of $l(\hat{a}_0, \hat{a}_1|\{(x_i, y_i^*)\})$ given by these 100 sets of data in the future are averaged. The result is used as $E[l(\hat{a}_0, \hat{a}_1|\{(x_i, y_i^*)\})]$.

This simulation given by this procedure is conducted $1,000$ times by varying the initial values of the pseudo-random numbers. The resultant distribution of d is illustrated in Fig. 5.11. The mean of this distribution is 3.893834. As the Poisson distribution does not need the equivalent estimation of variance, Eq. (5.24) (page 232) gives AIC. The result of the simulation setting $q = 1$ indicates $d = 3.893834$ inferring that AIC works appropriately.

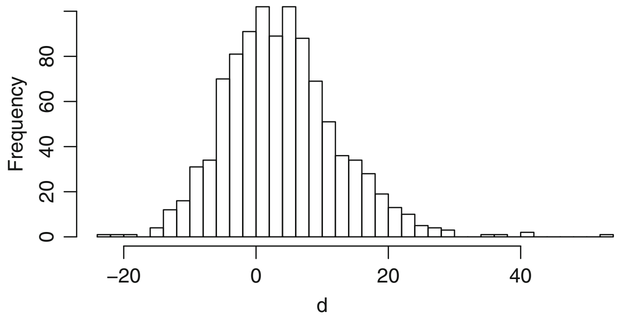


Fig. 5.11 After a regression by maximizing the value of Eq. (5.103) has been performed using 30 sets of simulation data, the value of d (Eq. (5.106)) is estimated using the data in the future. This distribution of d is obtained by conducting this simulation 1,000 times. The mean of this distribution is 3.893834

R Program [5 - 7]

AIC approximately holds for the Poisson distribution if the form of the regression equation is identical to that of the equation that either generates the data or contains the equation as a special case.

```
aic341e()
  function ()
  {
  # (1)
    nd <- 30
    nt <- 1000
  # (2)
    dif <- NULL
    for(kk in 1:nt){
      set.seed(2494 + kk * 29)
      xxa <- seq(from = 1, to = 30, length = nd)
      yya <- NULL
      for(ii in 1:nd){
        yya[ii] <- rpois(1, lambda = exp(xxa[ii] *
         0.2 - 4))
      }
  # (3)
      data1 <- data.frame(x = xxa, y = yya)
      glm1 <- glm(y~x, data=data1, family =
       poisson(log))
      ey <- glm1$fitted
      la <- -2*(sum( -ey + yya * log(ey) -
       lgamma(yya + 1) ))
  # (4)
      lb <- 0
      for(jj in 1:100){
        yyb <- NULL
        for(ii in 1:nd){
          yyb[ii] <- rpois(1, lambda = exp(xxa[ii] *
           0.2 - 4))
        }
        lb <- lb - 2 * (sum( -ey + yyb * log(ey) -
         lgamma(yyb + 1) ) )
     }
      lb <- lb/100
  # (5)
      dif[kk] <- lb - la
    }
  # (6)
    print("mean(dif)")
    print(mean(dif))
  # (7)
    par(mfrow = c(1, 1), mai = c(1.5, 1.5, 0.5, 0.5),
     oma = c(1, 1, 1, 1))
    br1 <-  pretty(dif, n = 40)
    bw1 <- br1[2] - br1[1]
    difh <- floor(dif/bw1) * bw1 + 0.01 * bw1
    hist(difh, breaks = br1, main = "", xlab = "d",
     ylab = "Frequency")
  }
```

(1) The number of data (`nd`) is given. The number of times of the simulation (`nt`) is given.
(2) The values of the predictors of the simulation data (`xxa`) are given. The values of the target variable of the simulation data (`yya`) are generated by the Poisson distribution defined in Eq. (5.99); `rpois()` is used for this purpose.
(3) Using the simulation data generated in (2), `glm()` performs the generalized linear regression. As `family = poisson(log)` is assigned in this example, the regression is conducted assuming a Poisson distribution and logarithmic function used as link function; that is, a Poisson regression assuming Eq. (5.102) (page 263) is performed. The estimates corresponding to the data are obtained using the resultant regression equation. The estimates are stored as `ey` and yield the value of $l(\hat{a}_0, \hat{a}_1|\{(x_i, y_i)\})$ (Eq. (5.104) (page 263)). This value is multiplied by (−2) with the result saved as `la`.
(4) The expectation of minus-two times the $l(\hat{a}_0, \hat{a}_1|\{(x_i, y_i^*)\})$ (Eq. (5.105) (264 page)) is approximated by generating 100 sets of data in the future in the same manner as that of the data at hand.
(5) Using the results of (3) and (4), an approximation of d (Eq. (5.106) (page 264)) is calculated and the result saved as `dif[kk]`.
(6) The average of `dif` is obtained with the result then displayed.

(7) The histogram presenting the distribution of `dif` is drawn (Fig. 5.11 (page 265)).

`aic341e()` also outputs:

```
"mean(dif)"
3.893834
```

R Program [5 - 7] End

References

1. Akaike H (1969) Fitting autoregressive models for prediction. Ann Inst Stat Math 21(1): 243–247
2. Burnham KP, Anderson DR (2002) Model selection and multi-model inference: a practical information-theoretic approach, 2nd edn. Springer, Berlin
3. Hurvich CM, Tsai C-L (1989) Regression and time series model selection in small samples. Biometrika 76(2):297–307
4. Konishi S, Kitagawa G (2007) Information criteria and statistical modeling. Springer, New York
5. Mallows CL (1973) Some comments on Cp. Technometrics 15(4):661–675
6. McQuarrie ADR, Tsai C-L (1998) Regression and time series model selection. World Scientific, Singapore
7. Montgomery DC, Peck EA, Vining GG (2001) Introduction to linear regression analysis, 3rd edn. Wiley, New York
8. Myers RH (2000) Classical and modern regression with applications (Duxbury Classic), 2nd edn. Duxbury, North Scituate
9. Myers RH, Montgomery DC, Vining GG (2001) Generalized linear models: with applications in engineering and the sciences. Wiley-Interscience, New York
10. Sugiura N (1978) Further analysts of the data by Akaike's information criterion and the finite corrections. Comm Stat Theor Meth 7(1):13–26
11. Takezawa K (2006) Introduction to nonparametric regression. Wiley, New York
12. Takezawa K (2012) A revision of AIC for normal error models. Open J Stat 2(3):309–312. http://www.scirp.org/journal/ojs/

Chapter 6
Linear Mixed Model

6.1 Random-Effects Model

The simplest form of the linear mixed model is the random-effects model, which represents data using the regression equation:

$$\mathbf{y}_i = \boldsymbol{\alpha} + \mathbf{b}_i + \boldsymbol{\epsilon}_i \qquad (1 \le i \le m), \tag{6.1}$$

where $\boldsymbol{\alpha}$, $\mathbf{y}_i$, $\mathbf{b}_i$, and $\boldsymbol{\epsilon}_i$ are column matrices for which the lengths are n_i and can be expressed in the form:

$$\boldsymbol{\alpha} = \begin{pmatrix} \alpha \\ \alpha \\ \alpha \\ \vdots \\ \alpha \end{pmatrix}, \quad \mathbf{y}_i = \begin{pmatrix} y_{1i} \\ y_{2i} \\ y_{3i} \\ \vdots \\ y_{n_i i} \end{pmatrix}, \quad \mathbf{b}_i = \begin{pmatrix} b_i \\ b_i \\ b_i \\ \vdots \\ b_i \end{pmatrix}, \quad \boldsymbol{\epsilon}_i = \begin{pmatrix} \epsilon_{1i} \\ \epsilon_{2i} \\ \epsilon_{3i} \\ \vdots \\ \epsilon_{n_i i} \end{pmatrix}. \tag{6.2}$$

Here, $\{y_{ji}\}$ $(1 \le j \le n_i)$ are observations of the i-th treatment $(1 \le i \le m)$; $\{b_i\}$ $(1 \le i \le m)$ are realizations from $N(0, d^2)$ (normal distribution with mean 0 and variance d^2). To specify the i-th treatment, one has to fix the value of i. Because all elements of $\mathbf{b}_i$ for the i-th treatment take the same value, b_i is intrinsic to a specific treatment. The values of b_i, however, vary among the treatments, although these are not independent of one another because $\{b_i\}$ are realizations of B (random variable (stochastic variable)). The distribution that B follows is $N(0, d^2)$ (normal distribution with mean 0 and variance d^2). $\{\epsilon_{ji}\}$ $(1 \le i \le m, 1 \le j \le n_i)$ are realizations of $N(0, \sigma^2)$ (normal distribution with mean 0 and variance σ^2). That is, an experiment with m treatments is conducted with the i-th treatment repeated n_i times. The j-th observation in the i-th treatment is denoted by y_{ji}. For example, consider an experiment where crop yields per area are recorded for m successive years. In the i-th year, n_i repetitions are performed and the crop yield for each

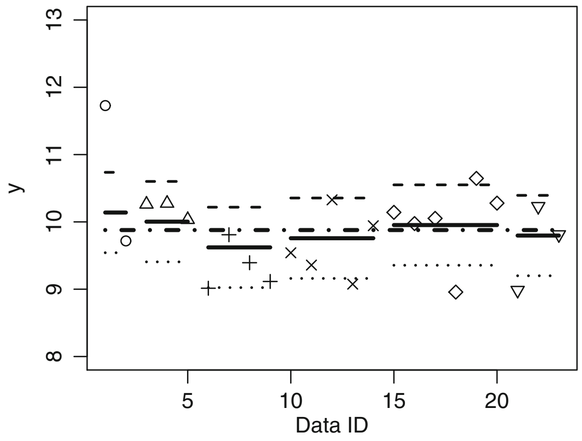


Fig. 6.1 Random-effects model (Eq. (6.1) (page 269) i.e., Eq. (6.3) (page 270)) obtained for simulation data ($m = 6$ and $\{n_i\} = \{2, 3, 4, 5, 6, 3\}$). The six symbols sort data by year. The *solid*, *dashed*, *dotted* and *dot-dash lines* for each year show the respective values $\{\hat{\alpha} + \hat{b}_i\}$, $\{\hat{\alpha} + \hat{b}_i + \hat{\sigma}\}$, $\{\hat{\alpha} + \hat{b}_i - \hat{\sigma}\}$, and $\hat{\alpha}$ for $(1 \le i \le 6)$

repetition is recorded. In such circumstances, the crop yield for the j-th repetition of the i-th year (y_{ji}) is written as

$$y_{ji} = \alpha + b_i + \epsilon_{ji}. \tag{6.3}$$

Here, b_i takes a specific value (nonrandom variable (nonstochastic variable)) for each year. Although b_i depends on i, b_i is not just an intrinsic value for the i-th year. $\{b_i\}$ are realizations of $N(0, d^2)$ (normal distribution with mean 0 and variance d^2). That is, this $\{b_i\}$ is based on the assumption that the influence of meteorological phenomena and others on crop varies yearly at random and it obeys $N(0, d^2)$. In contrast, ϵ_{ji} takes a different value for each observation (y_{ji}). It is a realization of $N(0, \sigma^2)$ (normal distribution with mean 0 and variance σ^2). Therefore, $\{\epsilon_{ji}\}$ is the random error contained in corresponding observation ($\{y_{ji}\}$).

Figure 6.1 shows the results of constructing the random-effects model in the form of Eq. (6.1) (i.e., Eq. (6.3)) using the simulation data, with $m = 6$ and $\{n_i\} = \{2, 3, 4, 5, 6, 3\}$. The values of the target variable are generated using

$$y_{ji} = 9.9 + b_i + \epsilon_{ji}. \tag{6.4}$$

$\{b_i\}$ are generated using $N(0, 0.5^2)$ (normal distribution with mean 0 and variance 0.5^2). $\{\epsilon_{ji}\}$ are generated using $N(0, 0.7^2)$ (normal distribution with mean 0 and variance 0.7^2). In Fig. 6.1, the values of $(\hat{\alpha} + \hat{b}_i)$ for each i are not identical to the averages of the data belonging to the i-th group. This is because the value of b_i (Eq. (6.3)) is not determined by only the data belonging to the i-th group but the entire data.

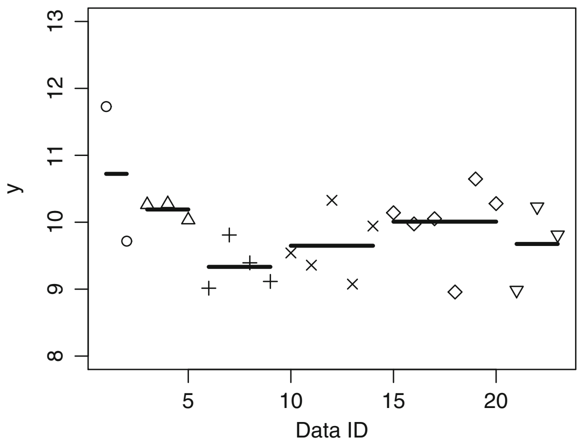


Fig. 6.2 Result of the simulation data used in Fig. 6.1 when Eq. (6.5) (page 271) is adopted. The *solid line* for each year indicates the value of $\{\hat{c}_i\}$ $(1 \le i \le 6)$

Using the same data as that used in Fig. 6.1, Fig. 6.2 shows the result of fitting the following regression equation:

$$y_{ji} = c_i + e_{ji}, \tag{6.5}$$

where $\{c_i\}$ $(1 \le i \le 6)$ denotes the annual average of the data of the corresponding year. That is, we have

$$\hat{c}_i = \frac{\sum_{j=1}^{n_i} y_{ji}}{n_i}. \tag{6.6}$$

Hence, the values of $\hat{c}_i$ in Fig. 6.2 are identical to the averages of the data belonging to the i-th group. $\{e_{ji}\}$ $(1 \le j \le n_i)$ are realizations of $N(0, \sigma_i^2)$ $(1 \le i \le m)$ (normal distribution with mean 0 and variance σ_i^2).

To see the difference between the results shown in Fig. 6.1 and those in Fig. 6.2, $\{\hat{b}_i\}$ and $\{\hat{c}_i\}$ are compared using the bootstrap method. That is, $\{\hat{\alpha} + \hat{b}_i\}$ and $\{\hat{c}_i\}$ are calculated using bootstrap data produced by resampling data of the corresponding year. This procedure is repeated 100 times by altering the initial value of the pseudo-random numbers. Distributions of the respective $\{\hat{\alpha} + \hat{b}_i\}$ are then obtained. The distributions are shown in Fig. 6.3 using a boxplot. Respective distributions of $\{\hat{c}_i\}$ are drawn in Fig. 6.4. Comparison between Figs. 6.3 and 6.4 shows that the variations in $\{\hat{c}_i\}$ are larger than those in $\{\hat{\alpha} + \hat{b}_i\}$. This behavior reflects the fact that whereas $\{b_i\}$ $(1 \le i \le m)$ are realizations of $N(0, d^2)$ in Eq. (6.1) (page 269) (i.e., Eq. (6.3) (page 270)), $\{c_i\}$ $(1 \le i \le m)$ are determined independently of one another in Eq(6.5) (page 271). Small variations of the respective $\{\hat{b}_i\}$ indicate that the reliabilities of the estimates are high. This is one of the significances for

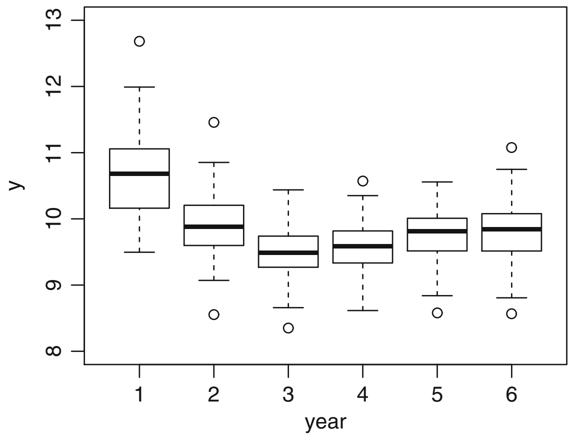


Fig. 6.3 Result of the bootstrap method by resampling the data of each year using the same simulation data as that used in Fig. 6.1. The distributions of $\{\hat{\alpha} + \hat{b}_i\}$ given by the random-effects model (Eq. (6.1) (page 269) (i.e., Eq. (6.3) (page 270))) are illustrated by a boxplot

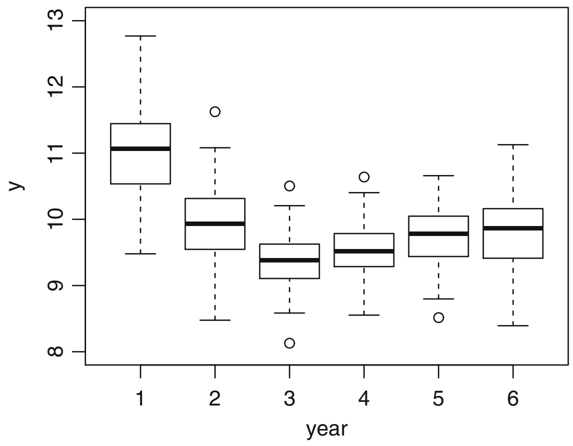


Fig. 6.4 Result of the bootstrap method by resampling the data of each year using the same simulation data as that used in Fig. 6.1. The distributions of $\{\hat{c}_i\}$ given by Eq. (6.5) (page 271) are represented by a boxplot

using the random-effects model. The small variations in $\{\hat{\alpha} + \hat{b}_i\}$, however, imply that the regression equation used here is based on a strong assumption. In practical situations, we rarely know before-hand whether the $\{\hat{b}_i\}$ are realizations of a normal distribution. Hence, if a regression equation in the form of Eq. (6.1) (page 269) is constructed when this assumption does not hold, we cannot avoid producing results that do not mirror reality due to the unreasonable constraints, although the resultant regression equation might be preferable to use given small variations in $\{\hat{\alpha} + \hat{b}_i\}$.

R Program [6 - 1]

Random-effects model in the form of Eq. (6.1) (page 269) (i.e., Eq. (6.3) (page 270)) is produced.

```
rem1()
  function (){
  # (1)
    library(nlme)
  # (2)
    al <- 9.9
    ni <- c(2, 3, 4, 5, 6, 3)
    nyear <- length(ni)
    set.seed(198)
    bbi <- rnorm(nyear, mean = 0, sd = 0.5)
  # (3)
    yyall <- NULL
    yearall <- NULL
    for (ii in 1:nyear){
      yy <- rep(al + bbi[ii], length = ni[ii]) +
       rnorm(ni[ii], mean = 0, sd = 0.7)
      yyall <- c(yyall, yy)
      year <- rep(ii, length = ni[ii])
      yearall <- c(yearall, year)
    }
  # (4)
    data1 <- data.frame(yyall = yyall, yearall =
     yearall)
  # (5)
    lme1 <- lme(yyall ~ 1, random=~1 | yearall,
     data = data1)
    print(summary(lme1))
  # (6)
    coef1 <- lme1$coef[[1]]
    coef2 <- lme1$coef[[2]]$yearall
    sigma1 <- lme1$sigma
  # (7)
    par(mai = c(1, 1, 1, 1), omi = c(0, 0, 0, 0))
    plot(c(1:length(yyall)), yyall, xlab = "Data ID",
     ylab = "y", type = "n", ylim = c(8,13))
    ct1 <- 0
    for(kk in 1:nyear){
      points(c((ct1 + 1):(ct1 + ni[kk])),
       yyall[(ct1 + 1):(ct1 + ni[kk])], pch = kk)
      lines(c((ct1 + 1),(ct1 + ni[kk])),
       c(coef1 + coef2[kk], coef1 + coef2[kk]),
       lwd = 3)
      lines(c((ct1 + 1),(ct1 + ni[kk])),
       c(coef1 + coef2[kk] + sigma1, coef1 +
       coef2[kk] + sigma1), lwd = 2, lty = 2)
      lines(c((ct1 + 1),(ct1 + ni[kk])),
       c(coef1 + coef2[kk] - sigma1, coef1
       + coef2[kk] - sigma1), lwd = 2, lty = 3)
      ct1 <- ct1 + ni[kk]
    }
    lines(c(1,length(yyall)), c(coef1, coef1),
     lwd = 3, lty = 4)
  }
```

(1) The use of the package “nlme” is described.
(2) The values of `a1` (α (Eq. (6.1) (page 269)), `ni` ($\{n_i\}$ $(1 \leq i \leq m)$, `nyear` (m), and `bbi` ($\{b_i\}$) are given.
(3) `yyall` is obtained by combining all values of $\{y_{ji}\}$. The values of years corresponding to `yyall` are set as `yearall`.
(4) `yearall` and `yyall` are organized in a data frame of `data1`.
(5) `lme()` produces a random-effects model using `data1` as data. The result is stored in `lme1`. Setting `yyall ~ 1` in `lme()` indicates that a random-effects model is constructed. `random=~1 | yearall` signifies that the constant part ($\{b_i\}$ (Eq. (6.3) (page 270)) in the regression equation obeys a normal distribution and the values of multiple b_i's are identical when their corresponding elements `yearall` are equal, although b_i is random. Furthermore, `print(summary(lme1))` displays the contents of `lme1` using `summary()` on a screen.
(6) As $\hat{\alpha}$ is contained in `lme1$coef[[1]]`, the value is saved as `coef1`. Similarly, as the $\{\hat{b}_i\}$ are contained in `lme1$coef[[2]]$yearall`, the values are stored in `coef2`. As $\hat{\sigma}$ is contained in `lme1$sigma`, the value is saved as `sigma1`.
(7) The resultant random-effects model is displayed (Fig. 6.1).

`rem1()` also outputs:

```
Linear mixed-effects model fit by REML
 Data: data1
       AIC      BIC    logLik
  51.87572 55.14884 -22.93786

Random effects:
 Formula: ~1 | yearall
        (Intercept)  Residual
StdDev:   0.2822163 0.5972811 Fixed effects: yyall ~ 1
```

```
                Value Std.Error DF  t-value p-value
(Intercept) 9.878649 0.1723041 17 57.33263       0

Standardized Within-Group Residuals:
        Min          Q1         Med          Q3
-1.66686214 -0.68671152  0.05043881  0.44364603
        Max
 2.65994762

Number of Observations: 23
Number of Groups: 6
```

This output shows that the restricted maximum likelihood estimation (REML) is used for obtaining the regression equation. Moreover, Akaike's Information Criterion (AIC) and Bayesian Information Criterion (BIC) given by this random-effects model are also displayed in this output. We also know from this output that the estimate of the variance of $\{b_i\}$ ($\hat{d}^2$) is 0.07964604 ($= 0.2822163^2$).

To obtain the reliability of the result produced by `rem1()`, the number of simulation data is increased drastically. For example, `(2)` in `rem1()` is replaced with the following `(2)'`.

```
# (2)'
  al <- 9.9
  ni <- c(20, 30, 40, 50, 60, 30)
  ni <- rep(ni, times = 100)
  nyear <- length(ni)
  set.seed(198)
  bbi <- rnorm(nyear, mean = 0, sd = 0.5)
  yyall <- NULL
  yearall <- NULL
```

Then, the part of the output on a screen is replaced with:

```
Random effects:
Formula: ~1 | yearall
        (Intercept)  Residual
StdDev:    0.494839 0.6938775
```

This result reflects the settings of 0.5 as the standard deviation for $\{b_i\}$ and 0.7 for the standard deviation of $\{\epsilon_{ji}\}$.

R Program [6-1] End

R Program [6-2]

The variations for $\{\hat{\alpha} + \hat{b}_i\}$ are estimated by the bootstrap method.

```
rem3()
  function (){
```

```
# (1)
  library(nlme)
# (2)
  nb <- 100
# (3)
  al <- 9.9
  ni <- c(2, 3, 4, 5, 6, 3)
  nyear <- length(ni)
  set.seed(198)
  bbi <- rnorm(nyear, mean = 0, sd = 0.5)
  yyall <- NULL
  yearall <- NULL
# (4)
  coef1v <- NULL
  coef2v <- NULL
  for(jj in 1:nb){
    yyall <- NULL
    yearall <- NULL
    for (ii in 1:nyear){
      yy <- rep(al + bbi[ii], length = ni[ii]) +
       rnorm(ni[ii], mean = 0, sd = 0.7)
      yyb <- sample(yy, size = length(yy),
       replace = T)
      yyall <- c(yyall, yyb)
      year <- rep(ii, length = ni[ii])
      yearall <- c(yearall, year)
    }
    data1 <- data.frame(yyall = yyall,
     yearall = yearall)
    lme1 <- lme(yyall ~ 1, random = ~1 | yearall,
     data = data1)
    coef1 <- lme1$coef[[1]]
    coef1v <- c(coef1v, coef1)
    coef2 <- lme1$coef[[2]]$yearall
    coef2v <- cbind(coef2v, coef2)
  }
# (5)
  par(mai = c(1, 1, 1, 1), omi = c(0, 0, 0, 0))
  boxplot(coef1v + coef2v[1,], coef1v + coef2v[2,],
   coef1v + coef2v[3,], coef1v + coef2v[4,],
   coef1v + coef2v[5,], coef1v + coef2v[6,],
   ylim = c(8,13), xlab="year", names =
    as.character(c(1:6)), ylab="y" )
}
```

(1) The use of the package “nlme” is described.
(2) The number of times of the regression using bootstrap data is performed is given as `nb`.
(3) Simulation data are generated.
(4) The procedure generating the bootstrap data and constructing a random-effects model is performed `nb` times. The `nb` values of `coef1` ($\hat{\alpha}$) are combined to produce the vector of `coef1v`. The `nb` values of `coef2` ($\{\hat{b}_i\}$) are combined to produce the vector of `coef2v`.
(5) Using a boxplot, `coef1v` and `coef2v` draw the respective variations for $\{\hat{\alpha} + \hat{b}_i\}$ (Fig. 6.3 (page 272)).

R Program [6-2] End

6.2 Random Intercept Model

Instead of Eq. (6.1), the random intercept model represents data using the following regression equation,

$$\mathbf{y}_i = \boldsymbol{\alpha} + a\mathbf{x}_i + \mathbf{b}_i + \boldsymbol{\epsilon}_i \qquad (1 \le i \le m), \tag{6.7}$$

where a is not a vector but a scalar regression coefficient and $\mathbf{x}_i$ is defined as

$$\mathbf{x}_i = \begin{pmatrix} x_{1i} \\ x_{2i} \\ x_{3i} \\ \vdots \\ x_{n_i i} \end{pmatrix}. \tag{6.8}$$

As in Eq. (6.1) (page 269), all elements of $\mathbf{b}_i$ take the same value (i.e., b_i). Moreover, $\{b_i\}$ $(1 \le i \le m)$ are realizations of $N(0, d^2)$ (normal distribution with mean 0 and variance d^2). $\{x_{ji}\}$ $(1 \le i \le m, 1 \le j \le n_i)$ are values of the predictors of the data. For example, we assume that a certain crop is grown in a number n_i of fields over a period of m years with different amounts of fertilizer applied to each field. Crop yields are measured each year. Thus this experiment is performed m times with the amounts of fertilizer set at x_{ji} $(1 \le i \le m, 1 \le j \le n_i)$. That is, we have n_i treatments recording crop yields in the i-th year. Then, with the amount of fertilizer in the j-th treatment of the i-th year denoted by x_{ji}, the crop yield (y_{ji}) is written as

$$y_{ji} = \alpha + a x_{ji} + b_i + \epsilon_{ji}. \tag{6.9}$$

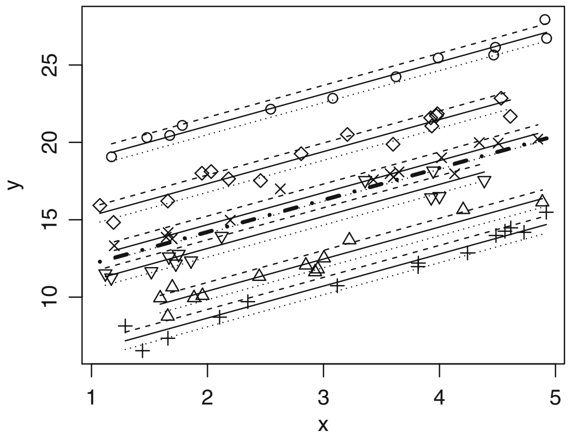


Fig. 6.5 Random intercept model given by the simulation data setting $m = 6$ and $\{n_i\} = \{12, 13, 14, 15, 16, 13\}$ (Eq. (6.7), i.e., Eq. (6.9)). The six symbols represent data of different years. The *solid*, *dashed*, *dotted*, and *dot-dash lines* of successive years show the respective linear trends in $\{\hat{\alpha} + \hat{a}x_{ji} + \hat{b}_i\}$ $(1 \leq i \leq 6)$, $\{\hat{\alpha} + \hat{a}x_{ji} + \hat{b}_i + \hat{\sigma}\}$, $\{\hat{\alpha} + \hat{a}x_{ji} + \hat{b}_i - \hat{\sigma}\}$, and $\{\hat{\alpha} + \hat{a}x_{ji}\}$

Figure 6.5 presents data analyzed with respect to the random intercept model (Eq. (6.7), specifically, Eq. (6.9)) given by simulation data with $m = 6$ and $\{n_i\} = \{12, 13, 14, 15, 16, 13\}$. The values of the predictors are determined from uniform random numbers between 1 and 5. α is set at 9.9. The simulation data is generated using the equation:

$$y_{ji} = 9.9 + 2x_{ji} + b_i + \epsilon_{ji}. \tag{6.10}$$

The distribution of $\{b_i\}$ is $N(0, 10^2)$ (normal distribution with mean 0 and variance 10^2); that of $\{\epsilon_{ji}\}$ is $N(0, 0.5^2)$ (normal distribution with mean is 0 and variance 0.5^2).

R Program [6-3]

A random intercept model as expressed by Eq. (6.7) (page 277) (i.e., Eq. (6.9) (page 277)) is constructed.

```
rem21()
  function (){
  # (1)
    library(nlme)
  # (2)
    al <- 9.9
    ni <- c(12, 13, 14, 15, 16, 13)
    nyear <- length(ni)
    set.seed(205)
    xx <- matrix(rep(0, length=max(ni) * nyear),
     ncol = nyear)
    for (ii in 1:nyear){
      xx[1:ni[ii], ii] <- runif(ni[ii], min = 1,
       max = 5)
      xx[1:ni[ii], ii] <- sort(xx[1:ni[ii], ii])
    }
    bbi <- rnorm(nyear, mean = 0, sd = 10)
  # (3)
    xxall <- NULL
    yyall <- NULL
    yearall <- NULL
    for (ii in 1:nyear){
      xxall <- c(xxall, xx[1:ni[ii], ii])
      yy <- rep(al + bbi[ii], length = ni[ii]) +
       2 * xx[1:ni[ii],ii] +
       rnorm(ni[ii], mean = 0, sd = 0.5)
      yyall <- c(yyall, yy)
      year <- rep(ii, length = ni[ii])
      yearall <- c(yearall, year)
    }
  # (4)
    data1 <- data.frame(yyall = yyall, yearall =
     yearall, xxall = xxall)
  # (5)
    lme1 <- lme(yyall ~ xxall, random=~1 | yearall,
     data = data1)
    print(summary(lme1))
  # (6)
    coef1a <- lme1$coef$fixed[1]
    coef1b <- lme1$coef$fixed[2]
    coef2 <- lme1$coef$random$yearall
    sigma1 <- lme1$sigma
  # (7)
    par(mai = c(1, 1, 1, 1), omi = c(0, 0, 0, 0))
    plot(xxall, yyall, xlab = "x", ylab = "y",
     type = "n")
    ct1 <- 0
    for(kk in 1:nyear){
      points(xxall[(ct1 + 1):(ct1 + ni[kk])],
       yyall[(ct1 + 1):(ct1 + ni[kk])], pch = kk)
      lines(xxall[(ct1 + 1):(ct1 + ni[kk])],
       coef1a + coef2[kk] + coef1b *
       xxall[(ct1 + 1):(ct1 + ni[kk])], lwd = 1)
      lines(xxall[(ct1 + 1):(ct1 + ni[kk])],
       coef1a + sigma1 + coef2[kk] + coef1b *
       xxall[(ct1 + 1):(ct1 + ni[kk])] , lwd = 1,
       lty = 2)
      lines(xxall[(ct1 + 1):(ct1 + ni[kk])],
       coef1a - sigma1 + coef2[kk] + coef1b *
       xxall[(ct1 + 1):(ct1 + ni[kk])] , lwd = 1,
       lty = 3)
      ct1 <- ct1 + ni[kk]
    }
    xxalls <- sort(xxall)
    lines(xxalls, coef1a + coef1b * xxalls,
    lwd = 3, lty = 4)
  }
```

(1) The use of the package "nlme" is described.
(2) The values of `a1` (α (Eq. (6.9)), `ni` ($\{n_i\}$ $(1 \leq i \leq m)$), `nyear` (m), `xx` (x_{ji} (Eq. 6.9) (page 277)), and `bbi` ($\{b_i\}$) are given.
(3) The set of all elements of $\{x_{ji}\}$ gives the vector `xxall`. Similarly, the set of all the elements of $\{y_{ji}\}$ gives the vector `yyall`. The index values for the years corresponding to the elements of `yyall` are saved as `yearall`.
(4) `yearall`, `yyall`, and `xxall` are organized in the data frame of `data1`.
(5) `lme()` produces the random intercept model using `data1` as data. The result is stored in `lme1`. `yyall ~ xxall` indicates that a linear equation is constructed. `random=~1 | yearall` shows that the constant ($\{b_i\}$ (Eq. (6.9) (page 277))) in the regression equation obeys a normal distribution and although the value of the constant (b_i) is random, the values of multiple b_i's are identical if their corresponding elements of `yearall` are equal. `print(summary(lme1))` displays the summarized contents of `lme1` using `summary()`.
(6) Since `lme1$coef$fixed[1]` contains the value of $\hat{\alpha}$, the value is saved as `coef1a`. Since `lme1$coef$fixed[2]` contains the value of $\{\hat{a}\}$, the value is saved as `coef1b`. Since `lme1$coef$random$yearall` contains the value of $\{\hat{b}_i\}$, the value is saved as `coef2`. Since `lme1$sigma` contains the value of $\hat{\sigma}$, the value is saved as `sigma1`.
(7) The resultant analysis for this random intercept model is displayed (Fig. 6.5).

`rem21()` also outputs:

```
  Linear mixed-effects model fit by REML
   Data: data1
         AIC      BIC    logLik
    187.1187 196.6965 -89.55934

  Random effects:
   Formula: ~1 | yearall
          (Intercept)  Residual
  StdDev:     4.55324 0.5597792 Fixed effects: yyall ~ xxall
```

```
                Value Std.Error DF  t-value p-value
(Intercept) 10.069990 1.8662977 76  5.39570       0
xxall        2.069596 0.0519939 76 39.80463       0
Correlation:
      (Intr)
xxall -0.083

Standardized Within-Group Residuals:
        Min          Q1         Med          Q3
-1.98630124 -0.71470221 -0.09567211  0.57004773
        Max
 2.83644093

Number of Observations: 83
Number of Groups: 6
```

`Correlation` represents correlation coefficient between the intercept and the slope.

R Program [6-3] End

6.3 Random Intercept and Slope Model

The random intercept and slope model represents data using the following regression equation instead of Eq. (6.7) (page 277).

$$\mathbf{y}_i = \alpha + (\beta + a_i)\mathbf{x}_i + \mathbf{b}_i + \boldsymbol{\epsilon}_i \qquad (1 \le i \le m), \tag{6.11}$$

where β is a regression coefficient. $\{(a_i, b_i)^t\}$ $(1 \le i \le m)$ are realizations of $(A, B)^t$; both A and B are random variables in the form of scalars. $(A, B)^t$ obeys $N((0,0)^t, \mathbf{D})$ (a normal distribution in which the mean is $(0,0)^t$ and the variance-covariance matrix is $\mathbf{D}$).

As used in Eq. (6.7), $\{x_{ji}\}$ $(1 \le i \le m, 1 \le j \le n_i)$ is the data. For example, we assume that a crop has been grown with various amounts of fertilizer applied and the crop yields (per area) had been recorded each year. This experiment is performed m times (i.e., m years). The amount of fertilizer applied over the i-th year is set to x_{ji} $(1 \le i \le m, 1 \le j \le n_i)$. That is, a certain crop is grown in n_i identical fields under different fertilizer applications and crop yields for respective conditions are measured in the i-th year. Then, given the amount of fertilizer is x_{ji} measured in the i-th year, the crop yield (y_{ji}) is written as

$$y_{ji} = \alpha + (\beta + a_i)x_{ji} + b_i + \epsilon_{ji}. \tag{6.12}$$

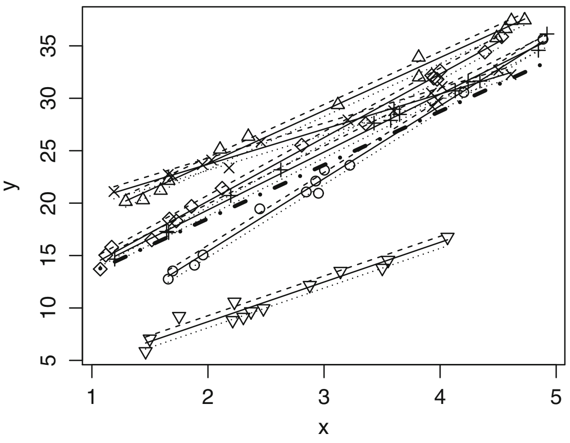


Fig. 6.6 Random intercept and slope model (Eq. (6.11), i.e., Eq. (6.12)) by the simulation data with the setting of $m = 6$ and $\{n_i\} = \{12, 13, 14, 15, 16, 13\}$. The six symbols represent data from different years. The *solid*, *dashed*, *dotted*, and *dot-dash lines* for each year show the trend values for $\{\hat{\alpha} + (\hat{\beta} + \hat{a}_i)x_{ji} + \hat{b}_i\}$ $(1 \le i \le 6)$, $\{\hat{\alpha} + (\hat{\beta} + \hat{a}_i)x_{ji} + \hat{b}_i + \hat{\sigma}\}$, $\{\hat{\alpha} + (\hat{\beta} + \hat{a}_i)x_{ji} + \hat{b}_i - \hat{\sigma}\}$, and $\hat{\alpha} + \hat{\beta}x_{ji}$

Figure 6.6 displays the data as analyzed using the random intercept and slope model (Eq. (6.11), i.e., Eq. (6.12)) given with simulation data setting $m = 6$ and $\{n_i\} = \{12, 13, 14, 15, 16, 13\}$. The values of the predictors are uniform random numbers between 1 and 5. The values of the target variable are given by

$$y_{ji} = 9.9 + a_i x_{ji} + b_i + \epsilon_{ji}, \tag{6.13}$$

where $\{a_i\}$ are realizations from $N(2.5, 0.5^2)$ (normal distribution with mean 2.5 and variance 0.5^2). $\{b_i\}$ are realizations from $N(0, 10^2)$ (normal distribution with mean 0 and variance 10^2). Therefore, $\mathbf{D}$ is a diagonal matrix. The distribution of $\{\epsilon_{ji}\}$ is $N(0, 0.5^2)$ (normal distribution with mean 0 and variance 0.5^2).

R Program [6-4]

The following R program produces a random intercept and slope model described by Eq. (6.11) (i.e., Eq. (6.12)).

```
rem31()
  function (){
  # (1)
    library(nlme)
  # (2)
    al <- 9.9
    ni <- c(12, 13, 14, 15, 16, 13)
    nyear <- length(ni)
    set.seed(205)
    aai <- rnorm(nyear, mean = 2.5, sd = 0.5)
    xx <- matrix(rep(0, length = max(ni) * nyear),
     ncol = nyear)
    for (ii in 1:nyear){
      xx[1:ni[ii], ii] <- runif(ni[ii], min = 1,
       max = 5)
      xx[1:ni[ii], ii] <- sort(xx[1:ni[ii], ii])
    }
    bbi <- rnorm(nyear, mean = 0, sd = 10)
    xxall <- NULL
    yyall <- NULL
    yearall <- NULL
  # (3)
    for (ii in 1:nyear){
      xxall <- c(xxall, xx[1:ni[ii], ii])
      yy <- rep(al + bbi[ii], length = ni[ii]) +
       2 * aai[ii] * xx[1:ni[ii],ii] +
       rnorm(ni[ii], mean = 0, sd = 0.5)
      yyall <- c(yyall, yy)
      year <- rep(ii, length = ni[ii])
      yearall <- c(yearall, year)
    }
  # (4)
    data1 <- data.frame(yyall = yyall,
     yearall = yearall, xxall = xxall)
  # (5)
    lme1 <- lme(yyall ~ xxall, random=~xxall | yearall,
     data = data1)
    print(summary(lme1))
  # (6)
    coef1b <- lme1$coef$fixed[1]
    coef1a <- lme1$coef$fixed[2]
    coef2b <- lme1$coef$random$yearall[,1]
    coef2a <- lme1$coef$random$yearall[,2]
    sigma1 <- lme1$sigma
  # (7)
    par(mai = c(1, 1, 1, 1), omi = c(0, 0, 0, 0))
    plot(xxall, yyall, xlab = "x", ylab = "y",
     type = "n")
    ct1 <- 0
    for(kk in 1:nyear){
      points(xxall[(ct1 + 1):(ct1 + ni[kk])],
       yyall[(ct1 + 1):(ct1 + ni[kk])], pch = kk)
      lines(xxall[(ct1 + 1):(ct1 + ni[kk])],
       coef1b + coef2b[kk] + (coef1a + coef2a[kk]) *
       xxall[(ct1 + 1):(ct1 + ni[kk])], lwd = 1)
      lines(xxall[(ct1 + 1):(ct1 + ni[kk])],
        coef1b + sigma1 + coef2b[kk] +
       (coef1a + coef2a[kk]) *
        xxall[(ct1 + 1):(ct1 + ni[kk])], lwd = 1,
        lty = 2)
      lines(xxall[(ct1 + 1):(ct1 + ni[kk])],
       coef1b - sigma1 + coef2b[kk] + (coef1a +
       coef2a[kk]) * xxall[(ct1 + 1):(ct1 + ni[kk])],
       lwd = 1, lty = 3)
      ct1 <- ct1 + ni[kk]
    }
    xxalls <- sort(xxall)
    lines(xxalls, coef1b + coef1a * xxalls,
    lwd = 3, lty = 4)
  }
```

(1) The use of the package "nlme" is described.
(2) The values of `a1` (α (Eq. (6.12) (page 281)), `ni` ($\{n_i\}$ $(1 \leq i \leq m)$), `nyear` (m), `xx` (x_{ji} (Eq. (6.9) (page 277)), and `bbi` ($\{b_i\}$) are given.
(3) `xxall` is formed from all the $\{x_{ji}\}$ values. `yyall` is formed from all the $\{y_{ji}\}$ values. The index values of the respective years corresponding to `yyall` are assigned to `yearall`.
(4) `yearall`, `yyall`, and `xxall` are organized into data frame of `data1`.
(5) `lme()` produces a random intercept and slope model using `data1` as data. The result is stored in `lme1`. `yyall ~ xxall` indicates that a linear equation has been constructed. `random=~xxall | yearall` indicates that although each of a_i and b_i (Eq. (6.12) (page 281)) obey a normal distribution, the values of multiple a_i's are identical when their corresponding elements in `yearall` are equal, and similarly for the b_i's. Furthermore, `print(summary(lme1))` displays the contents of `lme1` on a screen using `summary()`.
(6) Since `lme1$coef$fixed[1]` contains $\hat{\alpha}$, the value is saved as `coef1b`. Since `lme1$coef$fixed[2]` contains $\hat{\beta}$, the value is saved as `coef1a`. `lme1$ coef$random$yearall[,1]` contains $\{\hat{a}_i\}$, the value is saved as `coef2b`. Since `lme1$coef$random$yearall[,2]` contains $\{\hat{b}\}$, the values are saved as `coef2a`. Since `lme1$sigma` contains $\hat{\sigma}$, the value is saved as `sigma1`.
(7) The resultant random intercept and slope model is illustrated (Fig. 6.6).

`rem31()` also outputs:

```
Linear mixed-effects model fit by REML
Data: data1
AIC      BIC      logLik
219.8885 234.2552 -103.9443

Random effects:
```

```
Formula: ~xxall | yearall
Structure: General positive-definite, Log-Cholesky
 parametrization
             StdDev      Corr
(Intercept)  6.3758133   (Intr)
xxall        1.3696860   -0.436
Residual     0.5747867

Fixed effects: yyall ~ xxall
                Value    Std.Error  DF  t-value   p-value
(Intercept) 8.283355  2.6095145  76  3.174290  0.0022
xxall       5.118007  0.5625145  76  9.098444  0.0000
Correlation:
(Intr)
xxall -0.439

Standardized Within-Group Residuals:
Min          Q1           Med            Q3
-1.82440197  -0.64977127  -0.05953078   0.58494805
Max
2.54107581
Number of Observations: 83
Number of Groups: 6
```

The output above includes the fact that the correlation coefficient between A and B is -0.436. Although the correlation between A and B is set to zero when the simulation data is generated, this correlation is not negligible because the number of data is small.

R Program [6 - 4] End

6.4 Generalized Linear Mixed Model

If the constant term is assumed to be random, the regression equation of the generalized linear mixed model (GLMM) is written as

$$E(\mathbf{Y}_i) = g^{-1}(\boldsymbol{\alpha} + a\mathbf{x}_i + \mathbf{b}_i) \quad (1 \le i \le m), \tag{6.14}$$

where $E(\cdot)$ stands for expectation (expected value), $\mathbf{Y}_i$ a random variable which is constructed by obtaining the data of $\mathbf{y}_i$ (vector) many times, and $g^{-1}(\cdot)$ is the inverse function of a link function. It is, for example, the inverse function of the logarithmic function (Eq. (5.109) (page 264)). This equation is similar to Eq. (6.7) (page 277). Whereas the regression equation is written as a linear equation plus $\{\epsilon_{ji}\}$, which obeys a normal distribution, if a normal distribution is assumed, the regression equation in general cannot be represented in this form if another distribution is assumed. A regression equation such as Eq. (6.14) is usually needed for such a situation.

For example, let us consider an experiment in which animals are given a stimulus with a specific level to observe the presence or absence of a response. If a stimulus of level x_{ji} $(1 \leq i \leq m, 1 \leq j \leq n_i)$ is applied, the result (y_{ji}) is represented as 1 (presence of the response), or 0 (absence of the response). A random variable constructed from the y_{ji}'s that are observed in many experimental runs is denoted by Y_{ji}. Then, we have

$$E(Y_{ji}) = g^{-1}(\alpha + a x_{ji} + b_i). \tag{6.15}$$

We denote the elements of $\mathbf{Y}_i$ are $\{Y_{ji}\}$ $(1 \leq j \leq n_i)$ and the elements of $\mathbf{x}_i$ are $\{x_{ji}\}$ $(1 \leq j \leq n_i)$. When the realization of Y_{ji} is written as y_{ji}, the value of y_{ji} is 1 or 0. Hence, Y_{ji} fulfils the Bernoulli distribution. As the Bernoulli distribution is a special case of the binomial distribution, the regression equation of Eq. (6.15) is obtained if we execute software developed for the generalized linear mixed model with the assumption that the target variable satisfies the binomial distribution. If the target variable fulfills the Bernoulli distribution, the logistic function is a common choice for $g(\cdot)$ (the link function). The logistic function is written as

$$g(p) = \log\left(\frac{p}{1-p}\right). \tag{6.16}$$

This transformation is also called the logit transformation. Setting $g(p)$ to η, we have

$$p = g^{-1}(\eta) = \frac{\exp(\eta)}{1+\exp(\eta)}. \tag{6.17}$$

We show a simulation to construct a generalized linear mixed model (Eq. (6.14), i.e., Eq. (6.15)) using simulation data with $m = 10$ and $\{n_i\} = \{12, 13, 14, 15, 16, 13, 11, 17, 13, 16\}$. The target variable of the simulation data is assumed to obey the Bernoulli distribution. The values of the predictors $(\{x_{ji}\})$ are realizations of the uniform random numbers between 1 and 5. The values of the target variable is obtained using the following equation.

$$\eta_{ji} = 2.1 x_{ji} - 6 + b_i, \tag{6.18}$$

where $\{b_i\}$ are realizations of $N(0, 1^2)$ (normal distribution with mean 0 and variance 1^2). If we define

$$p = \frac{\exp(\eta_{ji})}{1+\exp(\eta_{ji})}, \tag{6.19}$$

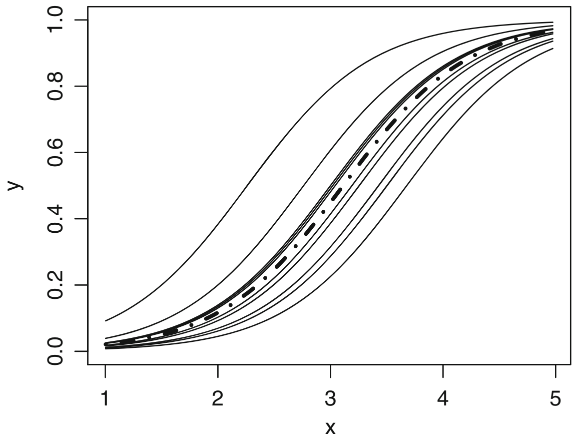


Fig. 6.7 Generalized linear mixed model (Eq. (6.14), i.e., Eq. (6.15)) for the simulation data with $m = 10$ and $\{n_i\} = \{12, 13, 14, 15, 16, 13, 11, 17, 13, 16\}$. The target variable for the simulation data obeys the Bernoulli distribution. The logistic function is used as the link function. The *dot-dash line* represents $g^{-1}(\hat{\alpha} + \hat{a}x)$. The 10 *solid lines* are graphs of $\{g^{-1}(\hat{\alpha} + \hat{a}x + \hat{b}_i)\}$ $(1 \le i \le 10)$

then Y_{ji} is the random variable which takes the value 1 with probability p and 0 with probability $(1 - p)$. The realization of Y_{ji} is denoted by y_{ji} (the value of the target variable). Figure 6.7 shows the result of constructing the generalized linear mixed model assuming a binomial distribution. The simulation data $(\{x_{ji}, y_{ji}\})$ obtained above and the logistic function as a link function are used.

R Program [6-5]

A random intercept model in the form of Eq. (6.14) (i.e., Eq. (6.15) is constructed.

```
rem41()
  function ()
  {
  # (1)
    library(lme4)
  # (2)
    set.seed(820)
    ni <- c(12, 13, 14, 15, 16, 13, 11, 17, 13, 16)
    ndset <- length(ni)
    xx <- matrix(rep(0, length = max(ni) * ndset),
     ncol = ndset)
    bbi <- rnorm(ndset, mean = 0, sd = 1)
    for (ii in 1:ndset){
      xx[1:ni[ii], ii] <- runif(ni[ii], min = 1,
       max = 5)
      xx[1:ni[ii], ii] <- sort(xx[1:ni[ii], ii])
    }
  # (3)
    xxall <- NULL
    yyall <- NULL
    zz <- NULL
    for (ii in 1:ndset){
      xxall <- c(xxall, xx[1:ni[ii], ii])
      yy <- NULL
      for(jj in 1:ni[ii]){
        eta1 <-  2.1 * xx[jj, ii] - 6 + bbi[ii]
        yy <- c(yy, rbinom(n = 1, size = 1,
         prob = exp(eta1)/(exp(eta1) + 1)))
      }
    yyall <- c(yyall, yy)
    zz <- c(zz, rep(paste(letters[ii], letters[ii],
     sep = ""), length = ni[ii]))
    }
  # (4)
    data1 <- data.frame(x = xxall, z = zz, y = yyall)
    lmer1 <- lmer(y~ x+(1|z), data = data1,
     family = binomial)
    print(summary(lmer1))
  # (5)
    coef1b <- fixef(lmer1)[1]
    coef1a <- fixef(lmer1)[2]
    coef2b <- ranef(lmer1)$z[,1]
  # (6)
    par(mai = c(1, 1, 1, 1), omi = c(0, 0, 0, 0))
    plot(xxall, yyall, xlab = "x", ylab = "y",
     type = "n")
    ct1 <- 0
    ex <- seq(from = min(xxall), to = max(xxall),
     length = 100)
    for (ii in 1:ndset){
      eta2 <- coef1b + coef2b[ii] + coef1a * ex
      ey <- exp(eta2)/(exp(eta2) + 1)
      lines(ex, ey)
      ct1 <- ct1 + ni[ii]
    }
    eta3 <- coef1b + coef1a* sort(xxall)
    ey <- exp(eta3)/(exp(eta3) + 1)
    lines(sort(xxall), ey, lwd = 3, lty = 4)
  }
```

(1) The use of the package “lme4” is described.
(2) The values of `ni` ($\{n_i\}$) are given. The number of elements in `ni` is given as `ndset`. α in Eq. (6.15) (page 286) is set to −6. `rnorm(ndset, mean = 0, sd = 1)` generates the values of $\{b_i\}$. a is set to 2.1. `rbinom()` generates realizations of the binomial distribution. `n = 1` is the number of data. `size = 1` is the number of times of the trial (when this value is 1, Bernoulli distribution is set).
(3) The simulation data given in (2) are reconstructed as the vector of the predictors of data (`xxall`), that of the target variable (`yyall`), and that which specifies the group (in the above example, it is an individual animal) which each data belongs to (`zz`. The element of `zz` is one of `"aa"`, `"bb"`, …, `"jj"`).
(4) `lmer()` performs the regression that constructs the generalized linear mixed model. The result is stored in `lmer1`. The assigning of `y~x + (1|z)` indicates the derivation of a linear equation. The constant part in the regression equation consists of a constant and a random value with mean 0. However, the constant part takes an identical value in the same group which is specified by `zz`. If `family = binomial` is set, the binomial distribution is asssumed. If the link function is not specified, the logistic function, which is the default for the binomial distribution, is employed. `print(summary(lmer1))` displays the content of `lmer1` using `summary()` in the console window.
(5) `fixef(lmer1)[1]` extracts the value of the intercept from `lmer1`. This intercept is that of the nonrandom part of the linear equation. `fixef(lmer1)[2]` extracts the slope from `lmer1`. This slope is that of the nonrandom part of the linear equation. `ranef(lmer1)$z[,1]` extracts the realizations of the random part of the regression equation from `lmer1`. The values of these realizations are given for the respective groups. The output of `lmer()` (`lmer1` in this example) belongs to the class of `mer`. Extracting the content of `lmer1` is explained by typing `help("mer-class", package="lme4")` to the console window.
(6) The result of the regression using a generalized linear mixed model is graphed (Fig. 6.7).

`rem41()` also outputs:

```
Generalized linear mixed model fit by the Laplace
approximation
Formula: y ~ x + (1 | z)
 Data: data1
 AIC    BIC    logLik  deviance
 134.4  143.2  -64.19  128.4
Random effects:
 Groups Name         Variance  Std.Dev.
 z      (Intercept)  0.80658   0.8981
Number of obs: 140,  groups: z, 10

Fixed effects:
```

```
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -5.6629  1.0001     -5.662  1.49e-08 ***
x            1.8196  0.2976      6.114  9.72e-10 ***
---
Signif. codes: 0'***'0.001'**'0.01'*'0.05'.'0.1' '1

Correlation of Fixed Effects:
  (Intr)
x -0.931
```

`Correlation of Fixed Effects` indicates the correlation coefficient between the intercept and the slope.

R Program [6-5] End

6.5 Generalized Additive Mixed Model

The generalized additive mixed model (GAMM) contains the category of models in which the intercept is assumed to be random; it constructs the following regression equation,

$$E(\mathbf{Y}_i) = g^{-1}(s(\mathbf{x}_i) + \mathbf{b}_i) \qquad (1 \le i \le m), \tag{6.20}$$

where $s(\cdot)$ signifies a nonparametric function. The spline function is often used for this function but, otherwise is the same as Eq. (6.14) (page 285). The elements of $\mathbf{Y}_i$ are represented as $\{Y_{ji}\}$ $(1 \le j \le n_i)$. Those of $\mathbf{x}_i$ are depicted as $\{x_{ji}\}$ $(1 \le j \le n_i)$. We then have

$$E(Y_{ji}) = g^{-1}(s(x_{ji}) + b_i). \tag{6.21}$$

We present a simulation to illustrate the construction of a generalized additive mixed model (Eq. (6.20), i.e., Eq. (6.21)). The simulation data inputs are $m = 10$ and $\{n_i\} = \{32, 33, 34, 35, 36, 33, 31, 37, 33, 36\}$. The target variable obeys the Bernoulli distribution. The values of the predictor ($\{x_{ji}\}$) form realizations of the uniform random numbers between 1 and 5. The values of the target variable are generated using the following equation.

$$\eta_{ji} = 2\sin(1.3x_{ji}) - 1 + b_i, \tag{6.22}$$

where $\{b_i\}$ are realizations of $N(0, 1.5^2)$ (normal distribution with mean 0 and variance 1.5^2). Setting

$$p = \frac{\exp(\eta_{ji})}{1 + \exp(\eta_{ji})}, \tag{6.23}$$

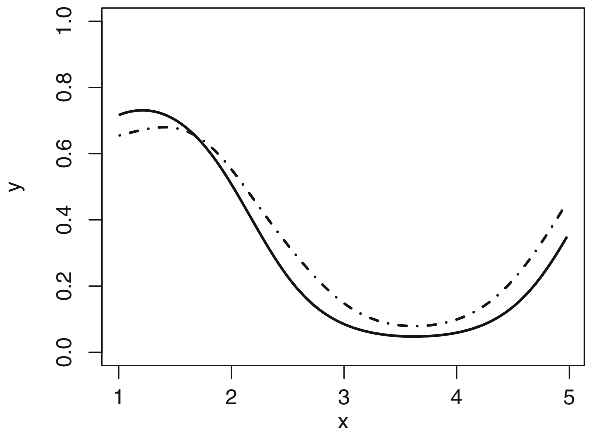


Fig. 6.8 Generalized additive mixed model (Eq. (6.20), i.e., Eq. (6.21)) given by the simulation data with $m = 10$ and $\{n_i\} = \{32, 33, 34, 35, 36, 33, 31, 37, 33, 36\}$. The target variable obeys the Bernoulli distribution. The link function is taken to be the logistic function. The *solid line* shows the values of $\{g^{-1}(s(x))\}$ ($s(\cdot)$ gives the values of the true function). The *dot-dash line* shows the values of $\{g^{-1}(\hat{s}(x))\}$ ($\hat{s}(\cdot)$ is a function estimated using data)

Y_{ji} is a random variable which takes 1 with probability p given by the above equation and 0 with probability $(1 - p)$; the realization of Y_{ji} is denoted by y_{ji} (the value of the target variable). The simulation data ($\{x_{ji}, y_{ji}\}$) generated by the procedure described above is used with the logistic function as link function. A generalized additive mixed model is constructed assuming a binomial distribution for which we obtain Fig. 6.8.

R Program [6-6]

A regression equation for the generalized additive mixed model in the form of Eq. (6.20) (i.e., Eq. (6.21)) is constructed.

```
rem51()
  function ()
  {
  # (1)
    library(mgcv)
  # (2)
    set.seed(816)
    ni <- c(32, 33, 34, 35, 36, 33, 31, 37, 33, 36)
    ndset <- length(ni)
    xx <- matrix(rep(0, length = max(ni) * ndset),
     ncol = ndset)
    bbi <- rnorm(ndset, mean = 0, sd = 1.5)
    for (ii in 1:ndset){
      xx[1:ni[ii], ii] <- runif(ni[ii], min = 1,
       max = 5)
      xx[1:ni[ii], ii] <- sort(xx[1:ni[ii], ii])
    }
  # (3)
    xxall <- NULL
    yyall <- NULL
    zz <- NULL
    for (ii in 1:ndset){
      xxall <- c(xxall, xx[1:ni[ii], ii])
      yy <- NULL
      for(jj in 1:ni[ii]){
        eta1 <- 2* sin(xx[jj, ii]*1.3) -1 + bbi[ii]
        yy <- c(yy, rbinom(1, size = 1, prob =
         exp(eta1)/(exp(eta1)+1)))
      }
      yyall <- c(yyall, yy)
      zz <- c(zz, rep(paste(letters[ii], letters[ii],
       sep = ""), length = ni[ii]))
    }
  # (4)
    data1 <- data.frame(x = xxall, z = zz, y = yyall)
    gamm1 <- gamm(y~s(x), random = list(z = ~1),
     data = data1, family = binomial)
    print(gamm1)
  # (5)
    par(mai = c(1, 1, 1, 1), omi = c(0, 0, 0, 0))
    plot(xxall, yyall, xlab = "x", ylab = "y",
     type = "n")
    ex <- seq(from = min(xxall), to = max(xxall),
     length = 100)
    data2 <- data.frame(x = ex, z = rep("aa",
     length = 100))
    ey <-  predict(gamm1$gam, newdata = data2,
     type = "response")
    lines(ex, ey, lwd = 2, lty = 4)
    eta2 <- 2* sin(ex * 1.3) -1
    yy <- exp(eta2)/(exp(eta2)+1)
    lines(ex, yy, lwd = 2)
  }
```

(1) The use of the package “mgcv” is described.
(2) Simulation data are generated.
(3) The simulation data obtained in (2) are arranged as the vector of the values of the predictor (`xxall`), that of the values of the target variable (`yyall`), and

that of the group names (individual names in the above example) of data (`zz`. Each element of `zz` is one of `"aa"`, `"bb"`, ..., `"jj"`).

(4) `gamm()` performs a regression based on a generalized additive mixed model. The result is stored in `gamm1`. Setting `y~s(x)` in `gamm()` indicates that the nonrandom part in the regression equation is a spline function given by the smoothing spline (smoothing splines) technique. Setting `random=list(z=~1)` indicates that the normally distributed part of the regression equation is a constant which is contained in x of the spline function ($s(x)$ (Eq. (6.21) (page 290))). The constant is a random value with mean 0 but takes the same value in a group specified by `z`. If `family = binomial` is specified, the binomial distribution is assumed. A link function is not specified here. Hence, the logistic function, which is the default for the binomial distribution, is employed. Moreover, `print(gamm1)` displays a summary of `gamm1` on the screen.

(5) The result of the regression using the generalized additive mixed model is drawn as a graph (Fig. 6.8).

`rem51()` also outputs:

```
Maximum number of PQL iterations:  20
iteration 1
iteration 2
iteration 3
iteration 4
iteration 5
$lme
Linear mixed-effects model fit by maximum likelihood
  Data: data
  Log-likelihood: -807.5422
  Fixed: fixed
X(Intercept)     Xs(x)Fx1
-0.8722551       1.5339778

Random effects:
 Formula: ~Xr.1 - 1 | g.1
 Structure: pdIdnot
        Xr.11    Xr.12    Xr.13    Xr.14    Xr.15
StdDev: 18.53512 18.53512 18.53512 18.53512 18.53512
 Xr.16    Xr.17
 18.53512 18.53512
           Xr.18
StdDev: 18.53512

Formula: ~1 | z %in% g.1
        (Intercept)  Residual
StdDev:    1.060032 0.9076563
```

```
Variance function:
 Structure: fixed weights
 Formula: ~invwt
Number of Observations: 340
Number of Groups:
       g.1 z  %in%  g.1
         1          10

$gam
Family: binomial
Link function: logit
Formula:
y ~ s(x)
<environment: 0x034f1e94>
Estimated degrees of freedom:
4.0548  total = 5.054785
```

R Program [6-6] End

Index