Digesting Gibbs Sampling Using R

Mahdi Teimouri Yanesari[1]

December 24, 2025

[1]profs.gonbad.ac.ir/fa/teimouri

Dedicated to my parents

Contents

List of Figures

List of Tables

Preface

In general, the statistical simulation approaches are referred to as the *Monte Carlo methods* as a whole. The broad class of the Monte Carlo methods involves the Markov chain Monte Carlo (MCMC) techniques that attract the attention of researchers from a wide variety of study fields. The main focus of this report is to provide a framework for all users who are interested in implementing the MCMC approaches in their investigations, especially the *Gibbs* sampling. I have tried, if possible, to eliminate the proofs, but reader is expected to know some topics in elementary calculus (including mathematical function, limit, derivative, partial derivative, simple integral) and statistics (including discrete and continuous random variables, probability distribution, expected value and variance, moment generating function, multivariate distribution, distribution of a functions of random variable, and the central limit theorem).

Almost all of researchers use statistical tools such as *confidence interval* or *hypothesis testing* for supporting their study results. Indeed, through both aforementioned tools, investigator can test a statement about the unknown population parameter such as population mean. Herein, the accuracy of decision made about the unknown parameter depends on a concept called *statistic* that is the sample counterpart of the unknown parameter under testing that is herein, for example, the sample mean or the sample median. The critical question is that which one is better the sample mean or the sample median ? The simulation process enables investigator to answer this question quite well. In fact, investigator will answer this question within a statistical laboratory by simulating sample evidence through the Monte Carlo methods, and then computing the sample mean or median based on simulated evidence. The accuracy of both statistics is accessible easily by repeating the simulation procedure. The Monte Carlo methods are widely used in several fields of study such as biology, engineering, econometrics, medicine, etc. The main concern of this report is placed on showing how user can apply the Monte Carlo methods in practice. I have tried to give the `R` codes accompanying with examples in order to help user for understanding the theoretical topics in the fields of statistics well. As the last, but not the least, I have used throughout the equivalent phrases "Gaussian" and "normal" interchangeably. I did my the best to avoid mistakes, specially for the formulas, however, I am sure that still there are several mistakes. Let me know please if you find anyone or have comments through the following email address.

Acknowledgements

- I would like to express my sincere thanks to my family and my colleagues that indirectly contributed to this work.
- I would like to express my sincere thanks to Amber Jain (`http://amberj.devio.us/`) for a free-to-edit LaTeX template.

Mahdi Teimouri Yanesari,
Assistant Professor in Statistics, Gonbad Kavous University, Gonbad Kavous, Iran.
Email:`mahdiba_2001@yahoo.com`

1

An introduction to R

The R (or R language) is an environment developed mainly for *statistical analysis*, *statistical computing*, and *data visualization*. In contrast to **SPSS** or **Minitab**, R lacks the graphical user interface (GUI) that allows user to accomplish statistical analysis using some graphical components such as button, dialog box, and icon. This means that R is a programming language or *command-line* interface in which user must type codes within some sessions called `console` or `script` for implementing the statistical methods. The first nonformal version on R was created by Ross Ihaka and Robert Gentleman, two statisticians from University of Auckland in August 1993 and then introduced formally in 1996 [Ihaka and Gentleman, 1996]. The first formal version of R, that is version 1.0, was created in February 29, 2000. This language has won a great deal of advocate among data scientists since it maintains some advantages among them we mention the following:

i. The R is loaded quickly.

ii. The R is user-friendly interface.

iii. The R is an open-source software that can be distributed freely.

iv. The R user can call (or interfaces) the C or Fortran for implementing a project faster.

v. The codes or programs in a new structure called R package that can be exported/imported simply while they are often accompanied by a long-form vignette (documentation) in order to help user for using it.

Another computational software written by data analysis researchers at Bell Labs is the S that has its origin from the joint project of Rick Becker and John M. Chambers on May 5 1976 Chambers [2020]. The S language was written to provide a framework for research in data analysis and collaborations to apply that research, rather than as a separate project to create a programming language Chambers [2020]. The S calls Fortran subroutines and has some extensions such as S3 and S4. The R and S3 are very close together. Herein, we do not deal with the differences between these extensions and their relations with R. Some redecorated and absolutely more attractive versions of R, called **R Studio** or **Visual R Studio**, are available for linting, compiling code, highlighting syntax, having a look at data, variables, packages, etc. Herein, we aim to provide a general, but not comprehensive, knowledge for users who are interested in working or being familiar with R. Since R is free for distributing or redistributing, the user is able to download it from `www.r-project.org`. To do this, the user just needs to enter `https://cran.r-project.org/mirrors.html`, and then download R, after selecting a mirror from a listing of mirrors that basically are servers located in different locations (countries) which host the same R and corresponding packages. It is suggested to select a close mirror in order to speed downloading. We point out that all notes given here can be found in more

details at `https://cran.r-project.org/manuals.html`. There is more detailed information on this in [Becker et al., 1988, Chambers and Hastie, 1992, Chambers, 1998, Nolan and Speed, 2000, Pinheiro and Bates, 2000, Therneau and Grambsch, 2000, Venables and Ripley, 2000, Harrell, 2001, Fox, 2002, Gareth et al., 2013, Hothorn and Everitt, 2014, Bloomfield, 2014, Daróczi, 2015, Blangiardo and Cameletti, 2015, Kohl, 2015, Sun, 2015, Dayal, 2015, Leemis, 2016, Murray, 2017, Hastie et al., 2017, Rahlf, 2017, Kelley, 2018, Fox and Weisberg, 2018].

1.1 Data and objects in R

There are, in general, five types of data in `R` including:

i. *character* (or *string*): the name of people, cities, mountains, or marital status, your gender, etc., that evidently cannot be recorded numerically.

ii. *complex*: a complex number such as $1 + 2i$ that is represented in `R` as `1+2i`.

iii. *integer*: a natural or whole number such as 2 that is represented in `R` when is followed by letter `L` as `2L`.

iv. *logical*: takes on only two states, that is, `TRUE` (or `T`) and `FALSE` (or `F`)

v. *numeric* (or *double*): a measurable quantity such as age, height, temperature, etc., that evidently can be recorded numerically, or the chemical, mathematical, or physical constants such as Avogadro's number (6.023×10^{23}), the ratio of a circle's circumference to its diameter $(\pi = 3.14159265\ldots)$, and the gravity on the Earth $(g = 9.8m/s^2)$.

The `R` user has not permission of access to the stored information directly, but `R` provides some data structure that is referred to as *object*. In point of view of some other programming languages *object* is a *variable*. Hence, everything in `R` is an *object*. All variables, regardless of their type, and all functions are objects. For example, the value such as 2 is a datum of type `"double"`, but if we have `a=2`, then `a` is an object. Each object in `R` belongs to a particular *class* that assigns some *attributes* or *properties*. These properties determine how the generic functions operate with the corresponding object. In summary, any object in `R` has a specific *type*, is stored in a specific *mode*, and has a particular class in order to determine the particular *object* can be used. These features are important when we are working on objects with more structures such as *vector*, *data frame*, *list*, *matrix*, etc.

R session

The `R` has two main sessions including *console* and *script* (or code editor). Once we run `R`, then the console session comes up in which a "$>$" prompt appears after which we must write the input command (call). One of the main objects in `R` is vector that has two types including: *atomic* and *nonatomic*. The elements of each atomic vector have the same type while elements of the latter are not of the same type. For example, suppose we would like to define an atomic vector consisting of numbers one, two, and three. To do this, we may proceed as follows.

```
R> v0 <- 1:3 # we may write v0 = 1:3
R> v0
```

and then press the Enter key "↵ " to see the output as follows.

```
[1] 1 2 3
```

Throughout, as seen in above, we show the `R` command and the `R` output in red- and blue-cored texts, respectively. As it is seen, we can insert an explanatory expression in front of vector `v0` in above after sharp symbol "`#`". We note that another way to define above vector is to run command

```
R> v1 <- c(1,2,3); v1
```

Of course, we can write several commands (calls) in one line provided that each command is separated with the next one by *just one* semicolon ";" symbol. As we see, objects `v0` and `v1`

R code	output
`R> mode(v0); typeof(v0); class(v0)`	`[1] "numeric"`
	`[1] "integer"`
	`[1] "integer"`
`R> mode(v1); typeof(v1); class(v1)`	`[1] "numeric"`
	`[1] "double"`
	`[1] "numeric"`

have the same `mode`, but different `type` and `class`. We note that, different objects may have the same class. In practice there is no difference between `v0` and `v1` defined in above and can be used interchangeably in many applications. Alternatively, we can produce the output given in above by typing `v1 <-c(1,2,3)` in the script session and then pressing Ctrl + R. The script is a new window that is appeared by following the track File ⟩ New script. Once we have created a script session, we follow the track File ⟩ Save as to save the script under an arbitrary name, for example "myfile.R" in which, as we see, suffix is `.R`. Care should be taken about the location in which the script will be saved. The default path can be changed straightforwardly. The default directory for saving `R` files is obtained by typing the following command in console or execute it from script session.

```
R> getwd()
```

The output is

```
[1] "C:/Users/NikPardaz/Documents"
```

If we no longer interested in directory given above, for changing it, we may put our desired address inside the command `setwd()`. For example, if we would like to change the path `C‣Users‣NikPardaz‣Documents` to `E‣my R work`, then we may use the command given below.

```
R> setwd("E:/my R work")
```

We note that `R` is case sensitive that implies, for example, letters "A" and "a" are different objects. There are different ways in `R` for getting help with functions and application of phrases. Herein, we suggest the most commonly used ways to get more information about phrase `apply`. The latter is a *built-in* function that is applied to the margins of an array or matrix. To do this, we may use either command

```
R> help(apply)
```

or

```
R> ?(apply)
```

The version of `R` that we are using is very important when installing the external packages. More detailed information on this and related features can be obtained using the command

```
R> R.version
```

We may use command

```
R> print(v1)
```

to observe the elements of vector v1. A nonatomic vector such as v2 may be defined in R as

```
R> v2 <- c(1, 2, 'male', 'female')
R> str(v2)

 chr [1:4] "1" "2" "male" "female"
```

As it is seen, two last elements of object v2 are of type character. If at least one of elements in a given object, such as v2, are character, then the given object is considered as a character shown by "chr". It is worth to note that objects in R whose type is character (nonumeric) must be surrounded by a single or double quotation marks. The elementary mathematical operations including addition, subtraction, and multiplication are meaningless for objects such as v2 defined in above. Sometimes, objects v1 and v2 are called the *numeric* and *non-numeric* vectors. Regardless of the its type, the element(s) of each vector can be found easily in R. For example, the second element of vector v1 is determined by command v1[2]. If one needs to determine more elements of a given vector, then the address of those elements must be put inside the square bracket. Let vector v3 is defined as

```
R> v3 <- 1:10 # equivalently we may write v3<-c(1,2,3,4,5,6,7,8,9,10)
```

For finding the first five elements of v3, we may use command

```
R> v3[1:5]
```

Equivalently, we may do this task through the command

```
R> v3[v3<=5]
```

Finally, if we want to save all elements of vector v3 that are between 3 and 8 (excluding ending points) in new vector called v4, we proceed as follows.

```
R> v4 <- v3[ v3<=7 & v3>=4 ]
```

If the desired elements are note well-addressed, then we have to determine the addresses through a new vector. For example, for picking up the elements of v3 whose addresses are odd (or even), we have

```
R> v.odd  <-v3[ c(1,3,5,7, 9) ]# elements of v3 whose indices are odd number
R> v.even1<-v3[ c(2,4,6,8,10) ]# elements of v3 whose indices are even number
```

Sometimes, we are interested eliminating one or mote items from a vector. For example, the last element of vector v3 is removed using command v3[-10]. Hence, another method for creating vector v.even in above is to use command

```
R> v.even2<-v3[-c(1,3,5,7,9)]# elements of v3 whose indices are even numbers
```

This means that vectors v.even1 and v.even2 are the same. The basic mathematical operations such as subtraction (addition) and multiplication (division) are applied to two vectors pointwise provided that the vectors are of the same lengths. For instance, we can compute v.odd/v_even1 as follows.

```
R> v.div <- v.odd/v.even1  # v.odd divided by v.even1
R> print(v.div)

0.5000000 0.7500000 0.8333333 0.8750000 0.9000000
```

The name assigned to a vector is quite optional, but we suggest the user to avoid using letters `c`, `D`, `F`, `I`, etc. Typically, an English letter, English letter followed by an integer number, English letter followed by English letter, an English letter followed by dot "·", or English letter followed by an underscore "_" is suggested. In above, we assigned the name `v.div` to vector `v.odd/v.even1`. The format and length of each element in output above can be determined by user. For example, we may use command `options(digits=4)` to see that

```
0.5000 0.7500 0.8333 0.8750 0.9000
```

As we see, the total digits after the decimal point is 4. If the output is greater than one, then the sum of total digits before and after the decimal point would be 4. If a vector multiplied by a numeric value, then all elements of the given vector is multiplied by that numeric value. This holds true also for subtraction, division, and addition operations. We have

```
R> v4 <- 1:3
R> v5 <- 3:5
R> v6 <- v4/v5

0.3333 0.5000 0.6000
```

Above three commands can be written in shorter form in order to save the space. For this purpose, all commands must be written sequentially, but each is separated from the others by symbol ";". We have

```
R> v4 <- 1:3; v5 <- 3:5; v6 <- v4/v5
```

Either both commands `ls()` or `objects()` can be used to show a listing of variables and objects created when working with `R`. Moreover, the objects created during an `R` session are stored in machine's memory and are ready when `R` is started at later time. We suggest to remove all variables at the end of each `R` session. For example, if we would like to remove either object `x` or both of objects `x` and `y` from the set of objects, we can use command

```
R> rm('x')
```

and

```
R> rm(list=c('x', 'y'))
```

respectively. In general command

```
R> rm(list=ls())
```

can be used to remove all objects from the current session. We note that, in general, `R` does not discriminate between commands `rm('x')` and `rm("x")`. It is worth to note that the `R` user if free to put blank or empty space between object(s) and round brackets `( )`, square brackets `[ ]`, and curly bracket `{ }`. For example, all commands `rm("x" )`, `rm( "x")`, `rm ("x")`, and `rm(list=c('x', 'y'))` are correct and do the same task. Furthermore, the `R` is case sensitive language. That is, the *capital* and *small* letters have different meaning and do different tasks. For instance, objects `x` and `X` are two different objects and command `Rm("x")` is meaningless. When using the latter, the output would be an error message as follows.

```
R> Rm("x")
Error in Rm("x") : could not find function "Rm"
```

The reminder of this subsection is devoted to introduce more complicated objects (or data structures) in `R` including sequence, data frame, matrix, array, and list.

Sequence

Herein, we introduce two important methods for creating vectors that have specified structures that typically are helpful in statistical analysis. For repeating some specified numeric values under some given pattern, we may use `rep(.)`. For example, in order to create sequence `1,1,2,2,3,3,4,4`, we may proceed as follows.

```
R> r1 <- rep(1:4, each = 2)
```

Furthermore, for creating sequence `1,2,3,4,1,2,3,4`, we have

```
R> r2 <- rep(1:4, 2)
```

and finally, we write

```
R> r3 <- rep(1:4, 1:4)
```

to produce sequence `1,2,2,3,3,3,4,4,4,4`.

For producing numeric values within an interval, $[a, b]$ say, with a given increment `delta` (or length `n`), we use command `seq(a,b,delta)` (or `seq(a,b,length=n)`). For example, in order to create vector `0,0.1,0.2,...,0.9,1`, we can use the following command.

```
R> s1 <- seq(0, 1, 0.1)
```

The output is

```
[1] 0.0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1.0
```

Alternatively, above output can be created using command

```
R> s2 <- seq(0, 1, length=11)
```

When working with vectors, we must take care about the observations that are recorded as missing or Not Available (`NA`), Not Available as Number (`NAN`), and infinity (`Inf`). In `R`, all observations that have been not recorded during the sampling stage are denoted by `NA`. The symbol `NAN` accounts for the undetermined (or undefined) cases such as is $0/0$ (or $\sqrt{-1}$), and hence is called *Not Available as Number*. Evidently, when a nonzero value is divided by zero, the result would be `Inf`. The mathematical operations cannot be applied on `NA`, `NAN`, and `Inf` cases. Hence, to avoid problems arising from these cases, we should have our diagnostics. One simple way, is the use of command `summary`. We have

```
R> z3 <- c(NA, 1:5)
R> summary(z3)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
   1.00    3.25    5.50    5.50    7.75   10.00       1
```

Based on above output, there is one missing value in vector `z3`. Typically, we need to apply the mathematical/statistical functions on elements of a vector. Table 1.1 shows a listing of statistical functions in `R` with a widespread use in applications.

Table 1.1: Some commands for numeric vector algebra in R.

Function	output/task
`all(x, na.rm=FALSE)`	`logical` is `TRUE` if all elements of vector `x` holds true for the given property, otherwise is `FALSE`.
`any(x, na.rm=FALSE)`	`logical` is `TRUE` if at least one elements of vector `x` holds true for some given property, otherwise is `FALSE`.
`as.vector(x)`	considering sequence `x` as a vector.
`cbind(x1,x2,...)`	creating a matrix whose columns are vectors `x1,x2,...`.
`ceiling(x)`	rounding elements of `x` to the smallest integer greater than `x`.
`cumsum(x)`	cumulative sum of elements of `x`.
`diff(x,lag=k)`	difference of `x` values with lag `k`.
`floor(x)`	rounding elements of `x` to the greatest integer less than `x`.
`intersect(x,y)`	intersect of `x` and `y`.
`is.numeric(x)`	logical: if `FALSE`, then `x` is not a numeric vector. and `TRUE` otherwise.
`length(x)`	length of `x`.
`max(x,na.rm=FALSE)`	maximum of `x`, by default `NA` is not removed.
`min(x,na.rm=FALSE)`	minimum of `x`, by default `NA` is not removed.
`mean(x)`	average of `x`.
`median(x,na.rm=FALSE)`	median of `x`.
`numeric(n)`	a numeric vector of length `n`.
`NULL`	a numeric vector regardless of its length.
`cumprod(x)`	cumulative product of elements of `x`.
`quantile(x,q)`	qth quantile of `x`.
`range(x,na.rm=FALSE)`	range of `x`, by default `NA` is not removed.
`rbind(x1,x2,...)`	creating a matrix whose rows are vectors `x1,x2,...`.
`rev(x)`	representing elements of `x` in reversed representation of `x`.
`round(x)`	rounding elements of `x` to the nearest integer.
`sd(x,na.rm=FALSE)`	standard deviation of `x`, by default `NA` is not removed.
`sum(x,na.rm=FALSE)`	sum of `x`, by default `NA` is not removed.
`summary(x)`	6-point summary of `x`.
`union(x,y)`	union of `x` and `y`.
`var(x,na.rm=FALSE)`	variance of `x`, by default `NA` is not removed.
`which.min(x)`	index in which `min(x)` occurs.
`which.max(x,na.rm=FALSE)`	index in which `max(x)` occurs.

Data frame

A rectangular structure of data with different types in R can be constructed using a concept called *data frame*. The data frame can be used to represent all types of data including numeric and non-numeric, but in a rectangular (balanced) form. The general command for data frame is

```
data.frame(name1=object1,name2=object2,...,row.names=NULL)
```

For example of data frame may be

```
R> name <- c('row1', 'row2', 'row3')
R> df1 <- data.frame(col1=1, col2=v4, col3=v5, row.names=name)
R> df1
```

The output becomes

```
     col1 col2 col3
row1    1    1    3
row2    1    2    4
row3    1    3    5
```

As another example, let we want to construct a *design of experiment* in which there are two factors (each of three levels) with one repetition. To do this, we may construct a data frame as follows.

R code

```
R> Level<-c('low','medium','high')
R> n.L<-length(Level); Rep<-1
R> factor1<-rep(Level,times=Rep,
+ each=n.L*n.L)
R> factor2<-rep(Level,times=n.L,
+ each=n.L)
R> df1<-data.frame(Factor1=
+ factor1, Factor2=factor2,
+ response=1:27)
```

output

```
    Factor1 Factor2 response
1       low     low        1
2       low     low        2
3       low     low        3
4       low  medium        4
5       low  medium        5
6       low  medium        6
7       low    high        7
8       low    high        8
...     ...     ...      ...
26     high    high       26
27     high    high       27
```

Sometimes we need to manipulate or organize some part of a data frame (or matrix) in terms of its other parts. A useful function for this purpose is `subset(df,subset,select)` in which, `df` denotes the data frame involving whole data, `subset` is the constraint we are willing to apply, and `select` is the name of column(s) to be select from a data frame. For example, in `airquality` data, suppose we are willing to selecting all levels of variables `Temp` and `Wind` for which variables `Ozone` and `Solar.R` are not missing. To do this is, we may use command `subset` as follows.

```
dd<-subset(airquality, !(Ozone=="NA") & !(Solar.R=="NA"), select=c(Wind,Temp))
```

When data frame is large, then we would prefer to manipulate the data frame in terms of its column's name rather than the column's order or number. In such a situation, one may use the R package `dplyr` available at R Comprehensive archive `https://cran.r-project.org/web/packages/dplyr/index.html`.

List

A useful and informative structure of data can be constructed in R using a concept called *list*. The list can be used to represent all types of data including matrix, numeric data, non-numeric data, and logical. To create a list (called here L1) consisting of vector v2 and data frame df1 defined earlier, we can write

```
R> L1 <- list(x1 = v2, x2 = df1); L1

$x1
[1] "1"      "2"      "male"   "female"

$x2
  x y z
1 1 1 4
2 1 2 5
3 1 3 6
```

To access to each part of data frame, for example the first part, we may write L$x1 and also for the second row of the second part we may write L$x2[2,], and finally elements consisting of digit 3 in last part of list L is accessible through command L$x3[4:6]. Of course, alternatively, the latter is accessible using L[[2]][2,]. We note that if we use command list(v2,df1), then names of two parts in the list L will be disappeared.

Matrix

One of the most commonly used structures in R is matrix. Recall that elements in each column of data frame are of the same type while different columns may have different types. The matrix is a rectangular form of data with the same type for which the numeric one is commonly used. In general, for constructing a matrix called A, we use the command given by the following.

```
R> A <- matrix(data=NA, nrow=1, ncol=1, byrow=FALSE, dimnames=NULL)
```

Herein data, nrow, ncol, byrow, and dimnames are arguments of function matrix. Table 1.2 below gives more explanations about the duty of each argument. It is worthwhile to note that

Table 1.2: Arguments of matrix in R.

Argument	output/task
data	a numeric vector that can include even NA, NAN, and Inf.
nrow	number of rows.
ncol	number of columns.
byrow	logical: if FALSE (by default), then the matrix is filled by columns, otherwise the matrix is filled by rows.
dimnames	a list of length 2 that gives the row and column names, respectively.

R automatically fills out the elements of matrix *column-by-column*. If user forgets the order of arguments in matrix function, it suffices to run command args(matrix) as

```
R> args(matrix)
```

to see the arguments of function matrix as follows.

```
function (data=NA, nrow=1, ncol=1, byrow=FALSE, dimnames=NULL)
NULL
```

Hence, if constructing a 2×2 matrix A represented as

$$A = \begin{bmatrix} 1 & 3 \\ 2 & 4 \end{bmatrix}.$$

is of interest, then we may use two commands to define matrix A as follows. If we want to

R code	output
`R>A<-matrix(c(1,2,3,4),nrow=2,ncol=2)# method 1`	`     [,1] [,2]`
`R>A<-matrix(1:4,nrow=2,ncol=2)# method 2`	`[1,]    1    3`
`R>A`	`[2,]    2    4`

assign names to the rows and columns, and further, fill out the matrix A by rows, we proceed as follows. To access (i, j)th element of matrix `A`, we must write `A[i,j]`. Moreover, the elements

R code	output
`R>L2<-list(c("row1","row2"),c("col1","col2"))`	`     col1 col2`
`R>A<-matrix(1:4,nrow=2,ncol=2,byrow=TRUE,`	`row1    1    2`
`+ dimnames=L2)`	`row2    3    4`

of ith row of matrix `A` are obtained by command `A[i, ]`, and accordingly, the elements of jth column of matrix `A` are obtained by command `A[ ,j]`. For instance, the output of commands `A[1,1]`, `A[1, ]`, and `A[, 2]` are given as follows.

```
R> A[1,1]     R> A[1,]     R> A[,2]
[1] 1         [1] 1  2     [1] 2  4
```

The matrix algebra typically is used in statistical methods. Table 1.3 lists the most commonly used functions (commands) associated with the matrix algebra.

Array

The array can be a one-, two-, or even three-dimensional object. The general command for array is `array(data=NA, dim=c(nr,nc,nz), dimnames=NULL)` in which `data` is the vector or sequence of array's elements, `nr` is the number of rows, `nc` is the number of columns, `nz` is the number of third coordinate, and `dimnames` is a list of names for three coordinates. There is also one- or two-dimensional array. The array can be reduced to construct a matrix. A simple example of three-, two-, and one-dimensional array can be constructed using vector `1:8`, `1:4`, and `1:2` as follows.

```
R> array(1:8,dim=c(2,2,2))  R> array(1:4,dim=c(2,2))   R> array(1:2,dim=2)

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

A useful summary about a data frame may be obtained though call `str(...)`. This some extension of command `type` defined formerly for vector object. The `str(...)` displays the

Table 1.3: Some commands for matrix algebra in R.

Function	output/task
A*B	pointwise product of the given matrices A and B.
A%*%B	product of the given matrices A and A.
as.matrix(df)	considers a data frame df as a matrix.
asplit(df,MARGIN)	splitting an array, data frame, list, or matrix by its margins (1 for rows, 2 for columns, and c(1,2) for rows and columns).
crossprod(A,B)	computing t(A)%*%B.
colMeans(A)	computing average of columns of A.
colnames(A)	assigning names to columns of A.
colSums(A)	computing sum of columns of A.
det(A)	computing determinant of matrix A.
dim(A)	dimensions of matrix A.
diag(A)	diagonal of matrix A.
diag(a)	constructing identity matrix whose diagonal elements are elements of vector a.
eigen(A)	eigen vectors and values of matrix A.
identical(A,B)	logical: TRUE if matrices A and B are identical; and FALSE otherwise.
is.matrix(df)	logical: TRUE is the given data frame df is a matrix, and FALSE otherwise.
isSymmetric(A)	logical: TRUE if matrix A is symmetric, and FALSE otherwise.
mahalanobis(x,mu,S)	Mahalanobis distance, that is (x-mu)$^\top$S^{-1}(x-mu).
ncol(A)	the number of columns of matrix (data frame) A.
qr(A)	QR decomposition of matrix A.
rowMeans(A)	computing average of rows of A.
rownames(A)	assigning names to rows of A.
rowSums(A)	computing sum of rows of A.
scale(A,center= TRUE,scale=TRUE)	matrix whose columns are standardized columns of matrix A.
solve(A)	inverse of matrix A.
t(A)	transpose of matrix A.

internal structure of an complicated object such as array, data frame, list, or matrix. For example, recall data frame df1, we have

```
R> str(df1)
'data.frame':  27 obs. of 3 variables:
 $Factor1:Factor w/ 3 levels "high","low","medium":2 2 2 2 2 2 2 2 2 3 ...
 $Factor2:Factor w/ 3 levels "high","low","medium":2 2 2 3 3 3 1 1 1 2 ...
 $response:int 1 2 3 4 5 6 7 8 9 10 ...
```

The phrase 'data.frame' in output indicates that the type of object is data frame that consists of 27 observations on three variables. Then, a summary is given on each of three variables Factor1, Factor2, and response.

1.2 Condition and repeat

In what follows, we deal with briefly some tools that are useful for creating desired data structure, data refinement, comparisons, etc. Some tools are needed to create a user-defined function well. These tools have been defined for R and include if(...), ifelse(...), while(...), an for(...). In what follows, we briefly discuss aforementioned tools.

1.2.1 Conditional decision using if(...)

For accomplishing some decision under some predetermined *condition* (or nature state), we use this command. In general, let nature space `Omega` consist of mutually disjoint states `State 1`, `State 2`, and `State 3` so that

$$\texttt{State 1} \cup \texttt{State 2} \cup \texttt{State 3} = \texttt{Omega}.$$

Furthermore, suppose we are willing to make `decision 1` (if `State 1` holds true), `decision 2` (if `State 2` holds true), and `decision 3` (if `State 3` holds true). For example, we can use the command `if(...)` for evaluating the sign of a real given number `x`. To this end, the pertinent pseudo and `R` codes are given by the following.

Pseudo code for command	`R` code for `if(...)` command
`if(State 1 holds true)`	`if(x==0)`
`{decision 1`	`{print(0)  # decision 1`
`}else if(State 2 holds true){`	`}else if(x>0){`
`decision 2`	`print(1)  # decision 2`
`}else{`	`}else{`
`decision 3}`	`print(-1) # decision 3`
`}`	`}`

If nature states consist of only two states, then we may use a simpler form of `if(...)` command called `ifelse(test,yes,no)`. Of course, the `if(...)` command still works well. Suppose, we are interested in turning nominal variable `x` that takes on either `'male'` or `'female'` into a numeric variable that takes on either `1` or `0`, respectively. The pertinent `if(...)` and `ifelse(test,yes,no)` commands are as follows.

```
R> if(x=='male')
+  {
+  x<-1
+  } else {
+  x<-0
+  }
```

```
R> x<-ifelse(x=='male',1,0)
```

While using `if(...)`, we may use some operators. Table 1.4 lists some commonly used operators. It is worth to note that operator `a||b` evaluates `b` only if `a` is `FALSE` [Fox and Weisberg, 2018]. Let `c1<- c(1,1,2,2,1,2,1)` and `c2<- c(1,2,2,2,1,1,1)` denote the clustering results of two populations each consisting of two clusters `1` and `2`. This fact that two clusters `c1` and `c2` are not fully matched can be shown by the following commands.

1.2.2 Repeat using while(...)

For repeating some task until some statement holds true, we may use command `while(...)`. This command essentially needs some initialization in relation to the given statement. In general, we should consider some counter

```
initialization
while(statement holds true)
{
body of while
}
```

Table 1.4: Some commonly used operators in R.

Operator	output/task
<=	less than or equal.
>=	greater than or equal.
!=	not equals.
a%%b	remainder of division a/b.
a%/%b	quotient of division a/b.
a==round(a)	logical: returns TRUE if a is an integer value, otherwise returns FALSE.
a|b	logical: it applies element-wise to vectors a and b, and returns a vector of logical values.
a||b	logical: it applies to numeric values a and b, and returns a single logical value. The output is TRUE if either a or b holds true.
a&b	logical: it applies element-wise to vectors a and b, and returns a vector of logical values. Each element of output is TRUE if either a or b holds true.
a&&b	logical: it applies to numeric values a and b, and returns a single logical value. The output is TRUE if both of a and b hold true.

```
R> a1<-ifelse(
+all(c1-c2 == 0),
+"full", "not full")
R> a1
[1] "not full"
```

```
R> a2<-ifelse(
+any(c1/c2 != 1),
+"not full", "full")
R> a2
[1] "not full"
```

```
R> if(all(c1-c2==0)
+|| all(c1/c2==1)
+a3<-"full" else a3<-"not full")
R> a3
[1] "not full"
```

Suppose we would like to print sum of nonnegative integer numbers provided that sum of these numbers does not exceed 10. Below, we give the command `while(...)` for this purpose.

R code	output
R> Sum <- 0; i <- 0 #initialization	[1] 0
R> while(Sum < 10){ #while(statement)	[1] 1
+ Sum <- Sum + i	[1] 3
+ print(Sum)	[1] 6
+ i <- i + 1	[1] 10
+ }	

1.2.3 Repeat using for(...)

For doing some task repeatedly for a given number, we may use command `for(...)`. Note that in contrast to `while(...)`, the number of iterations in `for(...)` is fixed and known. Let the non-numeric vector `sex` is defined as `sex=c('male','female','male','female','female','male')` that represents the sex of six persons. If we are willing to replace `'male'` and `'female'`, respectively, with `1` and `0`, then we can command `for(...)` as follows.

```
R> sex <- c('male','female','male','female','female','male')
R> n <- length(sex)
R> x <- rep(0, n) # creating a numeric vector called x of size n
R> for(i in 1:n)
+ {
+  if( sex[i] == 'female')
+  {
+   x[i] <- 0
+  }else{
+   x[i] <- 1
+  }
```

```
12 + }
```

We note that in order to constructing vector `x` in above, we may obtain `x` simply using command `x<-ifelse(sex=='female',0,1)`. The next example uses command `for(...)` to create a vector involving the names of rows (or columns) for a 10×10 matrix. The pertinent `R` code and output are as follows.

```
R> N1<-rep(NA,10)
R> for(i in 1:length(N1)){
+ N1[i]<-paste('a',
+ sep="-", i)}

 [1] "a-1"  "a-2"  "a-3"  "a-4"
 [5] "a-5"  "a-6"  "a-7"  "a-8"
 [9] "a-9"  "a-10"
```

```
R> N2<-rep(NA,10)
R> for(i in 1:length(N2)){
+ N2[i]<-paste(letters[i],
+ sep="", i) }

 [1] "a1"  "b2"  "c3"  "d4"
 [5] "e5"  "f6"  "g7"  "h8"
 [9] "i9"  "j10"
```

1.3 Data importing and exporting

Herein, we deal with reading *built-in* data sets existing in `R`and further discussing its nice property of exporting (writing) and importing (reading) data from other applications.

1.3.1 Reading built-in data

There are some applied data sets available in `R` that are known as *built-in* data. The list of data is ready for using by running command `R>data()`. There appears a list of several data sets. For calling some special data set, we need just to put the name of data inside the former command. Herein, we are willing to call two famous data sets known as `airquality` Chambers [2018], `iris` Fisher [1936] data. The first set is a data frame with 153 observations on 6 variables representing the daily air quality measurements in New York fro May to September 1973. The second set consists of 50 measurements of features sepal length, sepal width, petal length, and petal width sampled from species Setosa, Versicolor, and Virginica of iris flower. For example, we may run the command

```
R> data(airquality)
R> head(airquality,6) #we may use tail(airquality,10) for last 10 lines
```

to see the first six rows of `airquality` data as follows.

```
  Ozone Solar.R Wind Temp Month Day
1    41     190  7.4   67     5   1
2    36     118  8.0   72     5   2
3    12     149 12.6   74     5   3
4    18     313 11.5   62     5   4
5    NA      NA 14.3   56     5   5
6    28      NA 14.9   66     5   6
```

Alternatively, one may use command `View(airquality)` to have a look at data in a new frame. It is worth to note that if you have access to some built-in data called `x` that other do not, one way to move it is as follows. First run command `dput(x)` and then copy the blue-colored output that starts with `structure(list(...` and then insert it at a new variable called `y` or email the blue-colored output to any destination.

1.3.2 Importing data from R package

We can read data in R fro other applications. If data are appended to an R package, then we need to load the package that involves the data set. This is done through command `library(package name)` or `require(package name)`. For example, to access the `DBH` data existing in R package `ForestFit`, we proceed as follows.

```
R> library(ForestFit) # loading package
R> data(DBH)          # loading the pertinent data called DBH
R> head(DBH,2)        # representing first 2 lines of data
```

1.3.3 Importing data from other application

The prepared data for reading should be in tabular structure such as data frame or matrix and the acceptable formats are variants of `txt` including comma-, pipe-, or tab-delimited. Herein, the phrase delimiter refers to an object that separates the entries of two columns (fields). The comma-delimited format is also known as the comma separated values (CSV). Typically, the raw data are available form an external resource and reading data using command `read.table(...)` saves the user time and avoids some potential error arising from manual data entry. The main difference between all above mentioned types of `txt` data is the type of delimiter between columns. In general, command `read.table(...)` is a function that reads a tabular structure of data and converts it into a data frame. The arguments and the pertinent details are given in Table 1.5. Although `read.table(...)` can be used for all formats, but we may use commands

Table 1.5: Details about main arguments of `read.table(...)` in R.

Argument	output/task
`file`	the full address of file that is ready for reading. Each row of the tabular data would be a line of the output file.
`header`	logical: if `TRUE`, then the file contains the names of the variables as its first line; otherwise, a vector of letter `"V"` followed by column number will be assigned to columns as the name.
`row.names`	a vector of row names.
`col.names`	a vector of names for the variables. The default is a set consist of letters `"V"` followed by the column number.
`sep`	this parameter allows to set different delimiters including comma, pipe, semicolon tab, and one space for separating columns. It is assumed that the entries at each line of the imported file are separated by the character in front of `sep` parameter. If columns are comma-, pipe-, or tab-delimited, then `sep=","`, `sep="|"`, or `"\t"`, is recommended, respectively.
`na.strings`	a character or string that is recognized as missing value of input data. By default `na.strings="NA"`. If not, put the missing symbol instead of `NA` inside the quotation marks.
`dec`	a character assumed for decimal points.

`read.csv(...)`, `read.csv2(...)`, and `read.delim(...)` for reading tabular data with different delimiters as shown by Table 1.6.

For example, suppose we have a tabular data whose name is `A.csv` as

```
"" ,"a","b","c","d","e"
"a",  1,  2,  3,  4,  .
```

Table 1.6: Functions for importing data in R.

Function	input data specifications
read.csv(...)	comma delimited (separated), that is sep=',', and dec='.'
read.csv2(...)	semi colon separates, that is sep=';', and dec=','
read.delim(...)	tab separated, that is '\t', and dec='.'

```
"b",  6,  7,  8,  9, 10
"c", 11, 12, 13, 14, 15
"d", 16, 17, 18, 19,  *
```

existing at address E▸my R work▸A.csv. As it may be seen, two characters "." and "*" are denoting missing data and further letters a,b,c,d,e are assigned as names to the rows and columns. Using command read.csv(...), then the following output is obtained.

Command	output
R> out <- read.csv("E:/my R code/A.csv",	X a b c d e
+ header = T, na.strings=c('*', '.'))	1 a 1 2 3 4 NA
R> out	2 b 6 7 8 9 10
	3 c 11 12 13 14 15
	4 d 16 17 18 19 NA

1.3.4 Importing xls and xlsx data

Sometimes the original data are needed to read from Microsoft Excel application. One way to read such data in R is to install the package readxl [1] developed for R (>= 3.6). For instance, suppose the well known built-in data mtcars is available in xls format at path E▸my R work▸Mtcars.xls and we are willing to import it using package readxl. To do this, we proceed as follows.

```
R> library(readxl)
R> datxl1 <- read_excel("E:/my R work/Mtcars.xls", sheet=1, range="A1:L33")
R> datxl2 <- as.data.frame(datxl1)
```

All Mtcars.xls data are collected in just one sheet. If there are other sheets, then the argument sheet allows user to import another existing data sheet. If user interested in importing a predetermined range of data sheet, then the argument range can be changed appropriately. The range of Mtcars.xls data is "A1:L33". For example, some other types of importing a predetermined part of data sheet is given as follows.

```
read_excel("Mtcars.xls", range="mtcars!B1:D5")
read_excel("Mtcars.xls", range=cell_rows(1:4))
read_excel("Mtcars.xls", range=cell_cols("B:D"))
```

Alternatively, for xls format as above, the commands read_xls also works. Unless stated otherwise, any blank cell in the original Excel data sheet are considered as a missing value. Any change is permitted using argument na=" ". More detailed information on other arguments is accessible running args(read_excel). Moreover, for data of format xlsx, the command read_xlsx(...)

[1] https://cran.r-project.org/package=readxl.

1.3.5 Data exporting

Likewise, data can be exported fro R for other applications as an input. Table 1.7 shows the arguments of command `write.table(...)`. If the format of output data is `csv`, then the command `write.csv(...)` is preferable. For example, we may apply command `write.table(...)`

Table 1.7: Details about main arguments of `write.table(...)` in R.

Argument	output/task
`x`	the name of file to be exported.
`file=""`	the full address of the output file.
`append`	logical: By default is `FALSE` that means the old output is replaced with the new one, otherwise the new output will be added at the end of the old one.
`na="NA"`	the missing values in the output file is denoted by "`NA`" by default, otherwise the character must be determined by user.

to the first 6 lines of the `airquality` data to produce a `txt` file, namely `air.txt`, for which columns are tab-separated, decimal points are represented as slash "`/`", missing data are denoted by "`*`" rather than "`NA`", and finally row names are removed.

Command

```
R>write.table( head(airquality,6),
+"E:/my R code/air.txt", sep='\t',
+ dec='/', na='*')
```

output

```
"Ozone"  "Solar.R"  "Wind"  "Temp"  "Month"  "Day"
   41       190      7/4      67       5        1
   36       118       8       72       5        2
   12       149      12/6     74       5        3
   18       313      11/5     62       5        4
    *        *       14/3     56       5        5
   28        *      14/99     66       5        6
```

1.3.6 Apply family

A wide range of functions can be applied to the members of array, data frame, matrix, and vector through `apply` family of functions. Herein, we will briefly discuss these functions.

1. `apply(A, MARGIN, FUN,...)`: The data structure `A` can be matrix, array, or data frame (with numeric values). If `MARGIN=1 (or 2)`, then function `FUN` is applied to the rows (or columns) of `A`. The output of `apply` would be a vector. It it worth to note that "`...`" accounts for some additional arguments that are passed to the function `apply`. For example, we may add argument `na.rm=T` that means the missing vales are removed when applying function `FUN` to the coordinates of data structure `A`. For instance, the sum of rows, mean of columns, and variance of rows of matrix `A` denoted as above is obtained as follows.

```
R> apply(A, 1, sum)
[1] 4   6
```

```
R> apply(A, 2, mean)
[1] 1.5   3.5
```

```
R> apply(A, 1, var)
[1] 2   2
```

2. `lapply(A, FUN,...)`: The data structure `A` can be data frame, list, matrix, or vector. The command `lapply` applies function `FUN` to each element of object `A`. The output of `lapply` would be a list. For example, let function `f1` is defined as

```
R> f1<- function(x) 10*x
```

then the command `lapply(df2, f1)` applies function `f1` to the columns (members) of data frame `df2` defines as `df2 <- data.frame(x=1:2, y=c(-2,2))`. The output is

```
$x
[1] 10 20
$y
[1] -20  20
```

3. `sapply(A, INDEX, FUN,...)`: Both functions `lapply` and `sapply` do the same task, but the input of `sapply` is vector, data frame, or list and moreover, the output of the `lapply` is a list while the results of `sapply` is vector. Let we are interested in standardizing the columns of data frame `df1` defined earlier. To do this, we have

Command	output			
`R> sapply(df1, function(x)`		col1	col2	col3
`+ c(x-mean(x))/sd(x) )`	[1,]	NaN	-1	-1
	[2,]	NaN	0	0
	[3,]	NaN	1	1

Not surprisingly, the first column of output in above is `NAN` since we have undetermined case 0/0.

4. `tapply(A, INDEX, FUN,...)`: The function `tapply` in `R` can be used when we want to apply some function to a vector that is grouped by some other vector. The input of `tapply` can be array, data frame, matrix, or list. In what follows, suppose we would like to compute the variance of variable `Sepal.Length` within `iris` data in terms of variable `Species`. We have

```
R>data(iris)
R>tapply(iris$Petal.Length,
+ iris$Petal.Width<1,var)
FALSE        TRUE
0.68157980   0.03015918
```

```
R>data(iris)
R>tapply(iris$Petal.Length,
+ iris$Species=="setosa",var)
FALSE        TRUE
0.68157980   0.03015918
```

As it may be seen, not surprisingly, both parts in output above are the same due to the fact that `iris$Petal.Width<1` and `iris$Species=="setosa"` are equal sets. We conclude that the variance of `Petal.Length` when variable `Petal.Width` is less (more) than one is 0.03015918 (0.68157980). The function `tapply` can be employed for applying `FUN` on a vector that is grouped by two or more variables. For example, suppose we are willing to compute the average of `response` variable involved in data frame `df1` introduced earlier while two constraints including `Factor1='low'` and `Factor2='high'` are applied. The pertinent command and output are given by the following.

```
R>tapply(df1$response, list(
+ df1$Factor1=='low', df1$
+ Factor2=='high'), mean)
       FALSE  TRUE
FALSE   17.0  21.5
 TRUE    3.5   8.0
```

```
R>tapply(df1$response, list(
+ Factor1=df1$Factor1, Factor2=
+ df1$Factor2), mean)
          Factor2
Factor1    high low medium
  high       26  20     23
  low         8   2      5
  medium     17  11     14
```

1.4 Graphical summary

Data visualization is one of the most crucial elements in data analysis. A main part of statistical evaluations and comparisons is based on graphical summary. We may use different visualization facilities in `R`, depending on the type of data set. In general, graphical summaries are either high-level or low-level. High-level graphical facilities always create a new plot window and erase the current plot if necessary; rather the low-level ones, in general, do not create a new plot window. Among the high-level summaries, for example, we may mention the scatterplot for representing two sets of numeric data running `plot(...)`. If two data sets are of non-numeric or different types, then a contingency table may be suggested for graphical summary representation using `table(...)`. In what follows, we list some important high- and low-level graphical facilities.

i. **High-level**: `plot(...)`, `pers(...)`, `pairs(...)`, `image(...)`, `hist(...)`, `dotchart(...)`, `contour(...)`, `coplot(...)`, `boxplot(x~y,...)`, and `barplot(...)`.

ii. **Low-level**: `abline(...)`, `axis(...)`, `curve(...)`, `expression(...)`, `legend(...)`, `lines(...)`, `par(...)`, `segments(...)`, and `text(...)`.

In what follows we deal with the `plot(...)` among the other high-level graphical facilities.

1.4.1 Plot(...)

We may call `plot(x,y,...)` to display a two-dimensional visual summary of horizontal (`x`) and vertical (`y`) coordinates of numerical data points. The three dots `...` represent further arguments to be passed to object `plot(x,y,...)` that are optional. Table 1.8 lists some descriptions for some important arguments of this object. Herein, we call `trees` built-in data using command `data(trees)` for our graphical illustration. This data set consists of 31 measurements of felled black cherry trees on variables: girth (diameter), height, and volume[2]. Figure 1.1(a) shows the scatterplot of height versus girth and Figure 1.1(b) displays the scatterplot of height and volume versus girth simultaneously. The command for producing these figures are given as follows.

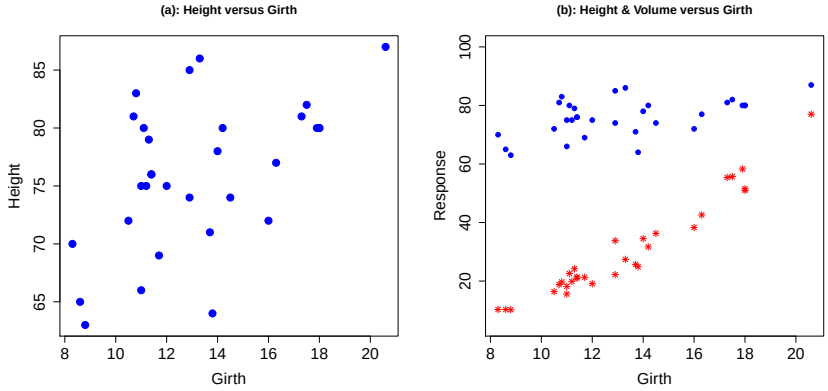


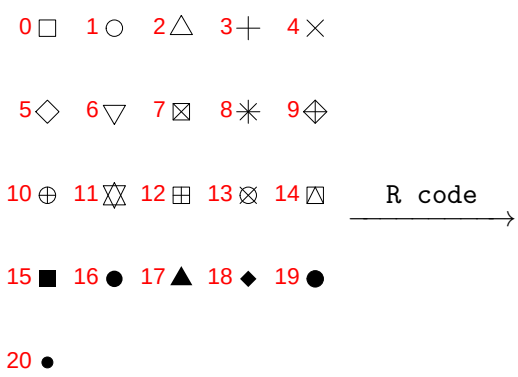
Figure 1.1: Scatterplot of (a): height vs. girth and (b): height and volume vs. girth.

```
R> # commands for producing Subfigure(a)
R>data(trees); x<-trees$Girth; y<-trees$Height; z<-trees$Volume; m<-1.5
R> plot(x, y, main="Height versus Girth", xlab="Girth", ylab="Height",
+ type="p", col="blue", pch=19, cex=m, lwd=m, cex.lab=m, cex.axis=m)
R> # commands for producing Subfigure(b)
R> plot(x, y, main="Height & Volume versus Girth", xlab="Girth", ylab=
+ "Response", type="p", col="blue", pch=19, ylim=c(5,100), lwd=m,
```

[2]Diameter is recorded at around 1.3 meters (or 4 to 6 feet) above the ground

Table 1.8: Details about main arguments of `plot` in `R`.

Argument	output/task
`cex`	the size of symbol suggested by `pch`.
`cex.axis`	the size of axes labels.
`cex.lab`	the size of axes title.
`cex.main`	the size of main title.
`col`	name of color defined for plot points inside the quotation mark.
`font`	the font type and font style of axes labels that is determined using object `Hershey`+ for font type and its style.
`lty`	an integer number representing the line type: 0=blank, 1=solid (default), 2=dashed, 3=dotted, 4=dotdash, 5=longdash, 6=twodash.
`lwd`	the thickness (width) of lines and symbols.
`mgp=c(a,b,c)`	it adjusts distance between axes lines and axes title (through `a`), axes labels (through `b`), and axes ticks (through `c`)
`pch`	an integer number from 0 to 20 denoting the symbol of data points shown as Figure 1.2.
`type`	it is `"b"` (for both points and lines), `"c"` (for empty points joined by lines), `"h"` (for histogram-like vertical lines), `"l"` (for lines), `"n"` (for any points or lines), `"o"` (for overplotted points and lines), `"p"` (for points), `"s"` (for a stair steps), and etc.
`xlab`	label for horizontal axis inside the quotation mark.
`ylab`	label for vertical axis inside the quotation mark.
`xlim`	`c(a,b)` that restricts the range of horizontal axis from `a` to `b`.
`ylim`	`c(a,b)` that restricts the range of vertical axis from `a` to `b`.
`yaxt="n"`	removes the labels of vertical axis.
`xaxt="n"`	removes the labels of horizontal axis.




```
R> par(mar=c(0,0,0,0))
R> w=c(rep('black',21),
+  rep('white',4))
R> y=rev(rep(1:5,each=5))
R> x=c(rep(1:5,5))
R> plot(x,y,pch=0:20,cex=4,
+  ylim=c(1,5),xlim=c(0,5),
+  axes=F,xlab=' ',ylab=' ',col=w)
R> text(x[1:21]-0.15,y[1:21],0:20,
+  pos=2,adj=1,col='red',cex=2.5)
```

Figure 1.2: Commonly used symbols for `pch` in `R` and the pertinent code.

```
+ cex.lab=m, cex.axis=m)
R> lines(x,z,type="p",col="red",pch=8)
```

After producing figure, user can save the produced figure freely in such formats as `bmp`, `jpeg`, `pdf`, `png`, and `tiff`. Herein, we use command `pdf(...)` for this end. Details about arguments of these calls are given using, for example, `?tiff`. Suppose the name of created figure is `FigA` and we would like to save it with portable document format (pdf) in track `E‣my R work`. To do this, we have

```
R> setwd("E:/my R work")
```

```
R> pdf(file="FigA.pdf",width=4,height=4) # width and height are in inches
R> # put here command plot(x,y,...)
R> # put here command line(x,y,...)
R> dev.off()
```

More manipulations can be made on produced plot using low-level graphical facilities. Table 1.9. Figure 1.3 shows two sample plots produced based on some supplementary arguments given in Table 1.9. The corresponding R codes are given as follows.

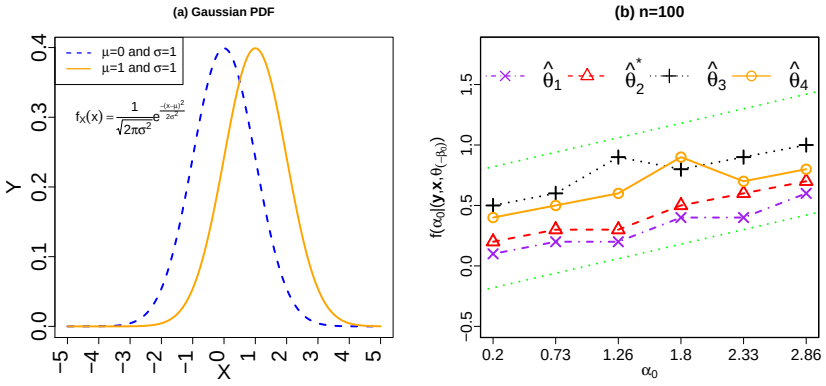


Figure 1.3: Two sample plots produced based on some supplementary calls of Table 1.3.

```
R># R code for producing Figure 1.2(a)
R>x<-seq(-5,5,.01); posi<-seq(-5,5,1)
R>plot(x,dnorm(x),col="blue",type='l',xlab='X',main="
+(a) Gaussian PDF",mgp=c(2.5,.5,0), ylab='Y',cex=1,
+lty=2,lwd=3,xaxt='n',cex.lab=2,cex.axis=2,pch=8)
R>lines(x,dnorm(x,1,1),col='orange',lwd=3, pch=19)
R>Leg1<-expression( paste(mu,"=0 and ",sigma,"=1"))
R>Leg2<-expression( paste(mu,"=1 and ",sigma,"=1"))
R>mytext<-expression(paste(f[X](x)==frac(1,sqrt(2*
+pi*sigma^2))*e^{frac(-(x-mu)^2,2*sigma^2)} ))
R>text(-3,0.3,mytext,cex=1.25)
R>axis(1,at=posi,lab=posi,hadj=1,padj=0,las=2,cex.axis=2)
R>legend('topleft',legend=c(Leg1,Leg2),lty=c(2,1),col=
+c('blue','orange'),cex=1.25,pch=c(8,19),lwd=2)
R># R code for producing Figure 1.2(b)
R>x1<-c(0.1,0.2,0.2,0.4,0.4,0.6);x2<-c(0.2,0.3,0.3,0.5,0.6,0.7)
R>x3<-c(0.5,0.6,0.9,0.8,0.9,  1);x4<-c(0.4,0.5,0.6,0.9,0.7,0.8)
R>vmle  <-expression(paste( hat(theta)[1]   ))
R>vstar <-expression(paste(hat(theta)[2]^'*'))
R>vhat1 <-expression(paste( hat(theta)[3]   ))
R>vhat2 <-expression(paste( hat(theta)[4]   ))
R>valpha<-expression(paste( alpha[0] ))
R>vylab <- expression(paste("f(",alpha[0],"|(", bold(y),",",
+bold(x),",", bold(theta)[(-beta[0])],")"))
R>x0 <- seq(0.2, 5, .5); v1<-c(vmle, vstar, vhat1, vhat2); m<-1.5;
R>plot(x1,cex.lab=m,cex.main=m,cex=2,main="(b) n=100",cex.axis=m,
+col="purple",lty=4,lwd=3,mgp=c(2.45,0.5,0),pch=4,type="o",
+xaxt='n',xlab=valpha,ylab=vylab,ylim=c(-0.5,1.8))
R>lines(x2, cex=2, col="red"   ,lty=2, lwd=3, pch=2 ,type="o")
R>lines(x3, cex=2, col="black" ,lty=3, lwd=3, pch=3 ,type="o")
R>lines(x4, cex=2, col="orange",lty=1, lwd=3, pch=1 ,type="o")
R>axis(1, at=seq(1,length(x0)), cex.axis=m,+labels=c("0.2","0.73",
+"1.26","1.8","2.33","2.86","3.4","3.93","4.46","5"))
```

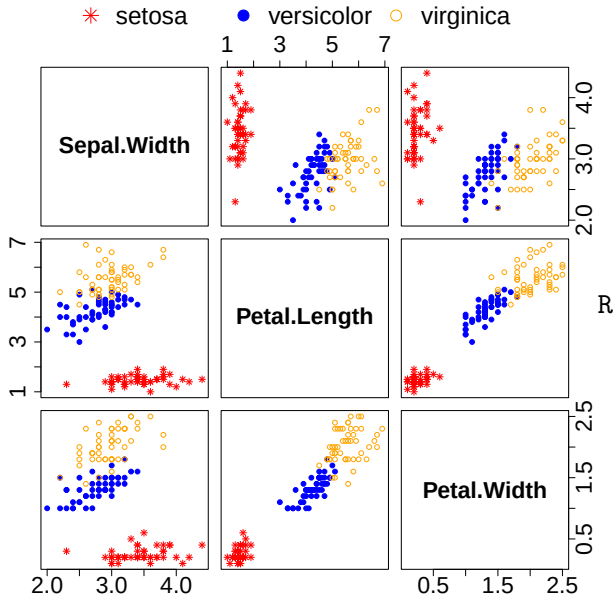
Table 1.9: Details about arguments of low-level graphical facilities.

Function/argument	output/task
`abline(a=NULL,b=NULL,h=NULL,v=NULL,...)`	it adds one or more straight lines to the plot. Parameters `a` and `b` are, accordingly, the intercept and slope of the fitted regression line. Parameters `h` and `v` are, respectively, the position of drawn horizontal and vertical lines to the plot.
`axis(side,at=NULL,labels=TRUE,...)`	
`side`	an integer number indicating the side of the plot we are willing to work on. That includes `1` (for `below`), `2` (for `left`), `3` (for `above`), and `4` (for `right`).
`at`	a vector of numerical values indicating the location at which the tick marks are shown.
`labels`	a vector of numeric or non-numeric data to be inserted at points determined in `at` argument.
`las`	an integer (`0`=always parallel to the axis [default] and `1`=always perpendicular to the axis) value.
`hadj`	a numeric number for moving the position of all axis labels simultaneously parallel to the reading direction.
`padj`	a numeric number for moving the position of all axis labels simultaneously perpendicular to the reading direction.
`curve(expr,from=NULL,to=NULL,n=101,add=FALSE)`	it works similar to call `plot(...)` for depicting a curve with formula `expr` over the given range starting at `from` and ending at `to`. If we want to add the output of call `curve(...)` to the output of plot, we must set `add=TRUE`.
`expression(paste(...))`	this command is useful when we want to add some annotations that typically involves mathematical formula. A full list of to a full list of mathematical symbols and formula run `demo(plotmath)`.
`legend('position',legend,lwd,col,horiz=FALSE,...)`	
`position`	one of `top`, `bottom`, `topleft`, `topright`, `bottomleft`, or `bottomright`. If some position else is desired, the `x` and `y` coordinates must be given instead.
`legend`	a vector of explanations for each line existing in the plot.
`lwd`	width of lines (or symbols) existing in argument legend.
`col`	color of lines existing in legend.
`horiz=FALSE`	logical: if `TRUE`, then the expressions are shown horizontally.
`lines(x,y,...)`	it adds a new curve to object `plot(...)` and its arguments are the same as `plot(...)`.
`par(...)`	it includes 72 graphical parameters or queries. The full list of these parameters is accessible by running `par(...)`. It is worth to note that if user changes the setting of object `par`, then these change will be applied to all graphs whenever the session is working. To restore the default settings, for other plots, calling `on.exit(...)` is suggested.
`segments(x0,y0,x1=x0,y1=y0,...)`	it draws a line from starting point whose `x` and `y` coordinates are, accordingly, `x0` and `y0` to end point whose `x` and `y` coordinates are, accordingly, `x1` and `y1`.
`text(a,b,'note',font,...)`	it puts a `note` at a position whose `x` and `y` coordinates are `a` and `b`, respectively. Parameter `font` determines the `note`, for which a `list` of fonts and styles can be found in object `Hershey`. Finally, three dots `...` represents further arguments that must be passed, for example `col` (for color), `cex` (for size), etc.

```
R>cols<-c("purple","red","black","orange")
R>legend("topleft",horiz=TRUE, legend=v1,bty="n",lwd=2,cex=2,
+col=cols,pch=c(4,2,3,1),lty=c(4,2,3,1))
R>abline(a=0.7,b=0.12,lty=3,lwd=3,col='green')
R>abline(a=-0.3,b=0.12,lty=3,lwd=3,col='green')
R>box(lwd=2)
```

1.4.2 Pairwise visualization

When there are more than two sets of numeric data, then a more useful graphical summary about the variables involved in study and associated relationships can be found using object `pairs(...)`. There is a long list of arguments for this object , among them in order to save space, we mention the `lower.panel`, `upper.panel`, and `diag.panel` that are more important than others. Suppose we are interested in relationship between last three variables, excluding `Species` `iris` data. A graphical summary may be obtained such as Figure 1.4 that displays the pairwise scatterplots of these three variables. More changes on produced pairwise scatterplot is possible by changing the arguments `lower.panel`, `upper.panel`, and `diag.panel` appropriately. Herein, we use our user-defined function by setting `diag.panel=panel.hist` for presenting the histogram of each marginal within the diagonal cells of pairwise scatterplots of Figure 1.5 while the superimposed appropriate Gaussian curve helps in discovering departure from normality.




```
R> c0<-unclass(iris$Species)
R> par(xpd=TRUE)
R> Col<-c("red","blue","orange")
R> pairs(iris[,2:4],main="",pch=
R> c(8,19,21)[c0],col=Col[c0],
+  labels=c("Sepal.Width",
+  "Petal.Length","Petal.Width"),
+  cex.labels=2.2, font.labels=2,
+  cex.axis=2.2)
R> L <-c("setosa","versicolor",
+  "virginica")
R> legend(0.1,1.05,bty="n",cex=
+  1.5,col=Col,pch=c(8,19,21),
+  legend=L,horiz=T)
R> dev.off()
```

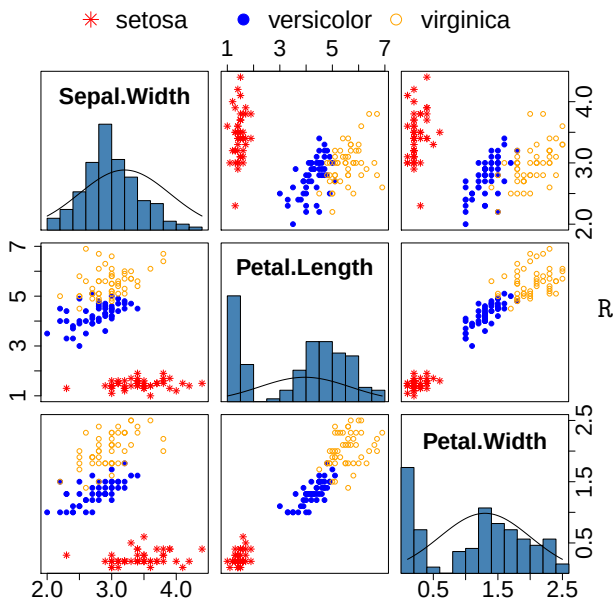
Figure 1.4: Pairwise scatterplots of `iris` data.

1.4.3 Contingency table

One useful graphical summary associated with both numeric and non-numeric data is contingency table. For constructing a contingency table we may use object `cut(...)` that splits numeric data into a number of mutually disjoint intervals (groups). Details about arguments of this object is given by Table 1.10. Recalling `women` data, we call `cut(...)` for grouping this

Table 1.10: Details about arguments of `cut(...)`.

Argument	output/task
`x`	a given numeric vector for grouping.
`breaks (br)`	either a numeric vector representing cut points or a single value>2 representing the number of groups.
`labels=NULL`	optional labels for the created groups.
`include.lowest`	`logical`: if `TRUE` and `x[i]` coincides with the lower bound of group, then group includes `x[i]`, otherwise `x[i]` belongs to another group.
`right`	`logical`: if `TRUE`, then each group takes the form `(a,b]`, otherwise it would be of the form `[a,b)`.
`...`	a number of arguments to be passed.



R code ⟶

```
R> panel.hist<-function(x,col,...){
+ usr<-par("usr");on.exit(par(usr))
+ par(usr=c(usr[1:2],0,1.5))
+ h<-hist(x, plot=FALSE)
+ breaks<-h$breaks
+ nB<-length(breaks)
+ y<-h$counts; y<-y/max(y)
+ rect(breaks[-nB],0,breaks[-1],
+ y,col="steelblue",...)
+ curve(dnorm(x,mean=mean(x),
+ sd=sd(x)),add=T)
+}
R> c0<-unclass(iris$Species)
R> par(xpd=TRUE)
R> Col<-c("red","blue","orange")
R> pairs(iris[,2:4],main="",pch=
R> c(8,19,21)[c0],col=Col[c0],
+   diag.panel =panel.hist,
+   labels=c("Sepal.Width",
+   "Petal.Length","Petal.Width"),
+   cex.labels=2.2, font.labels=2,
+   cex.axis=2.2)
R> L <-c("setosa","versicolor",
+   "virginica")
R> legend(0.1,1.05,bty="n",cex=
+   1.5,col=Col,pch=c(8,19,21),
+   legend=L,horiz=T)
R> dev.off()
```

Figure 1.5: Pairwise scatterplots of `iris` data with fitted Gaussian curve to the marginals.

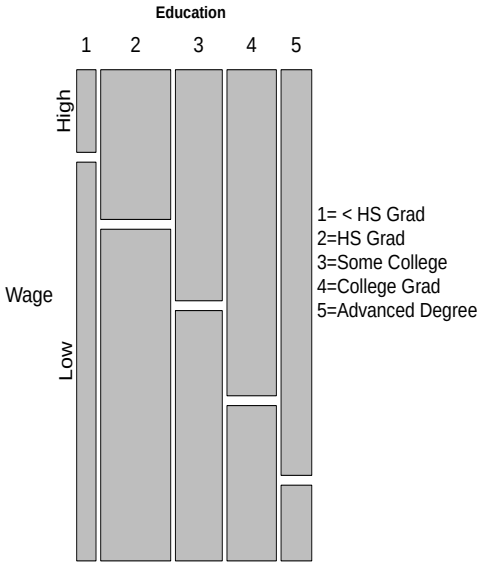
set of data as follows.

Command	output			
`R>br.h<-c(50,60,70,80)`		W		
`R>br.w<-c(110,130,150,170)`	H	(110,130]	(130,150]	(150,170]
`R>H<-cut(women$height,br=br.h)`	(50,60]	3	0	0
`R>W<-cut(women$weight,br=br.w)`	(60,70]	3	6	1
`R>table(H,W)`	(70,80]	0	0	2

The focus of the second example is in `wage` (wage and other information of 3000 male workers in the Mid-Atlantic region) data available at package `ISLR`. Suppose we are interested in relationship between marital status and wage variables. The former is non-numeric while the latter is numeric. One way to deal with relation between these two variables, first we may divide wage variable into two levels `High` (greater than median) and `Low` (less than median) and then investigating the chi-squared independence test. The output of chi-square independence test, after running `chisq.test(Table)`, is given by the following.

```
        Pearson's Chi-squared test
data:  Table
X-squared = 541.37, df = 4, p-value < 2.2e-16
```

Obviously, as it may be seen from Figure 1.6, we decide to reject the null hypothesis of H_0: "Wage and education variables are independent" against H_1: "Wage and education variables are not independent".




```
R> library(ISLR); data(Wage)
R> W.Level <-(ifelse(Wage$wage>
+  median(Wage$wage),"High","Low"))
R> n<- length(Wage$education)
R> E.Level <- unclass(Wage$education)
R> Table <- table(E.Level, W.Level)
R> par(mar=c(2, 4, 2, 11), xpd=TRUE)
R> mosaicplot(Table, cex.axis=1.5,
+  xlab="", ylab="",main="Education")
R> text(-.15,0.50,"Wage",cex=1.5,xpd=NA)
R> L <-c('1= < HS Grad','2=HS Grad','3=
+  Some College','4=College Grad','5=
+  Advanced Degree')
R> legend('topright',inset=c(-.6,0.3),L,
+  horiz=F, cex=1.4, bty="n")
R> dev.off()
```

Figure 1.6: Mosaicplot for `Wage` data.

1.5 Function in R

When we would like to accomplish some task(s) repeatedly, it is recommended to use some structure called *function* in order to saving space, reducing the computational costs, debugging errors and warnings easily, etc., within an `R` project. In general, an `R` function is either *built-in* or *user-defined*. Tables 1.1, 1.3, and 1.11 involve some built-in functions. The user-defined function are those created by user for some special purpose. Each function has four parts including *name*, *input*, *body*, and *output*. Two key points when creating a user-defined function are: i) to avoid constructing a function whose built-in counterpart does exist and ii) the name of the user-defined and existing built-in functions must be different; otherwise the user-defined function will *mask* the built-in function. The general structure of user-defined function is given as follows.

```
name <- function(input)
{
body of function
return(output) # we may write output instead
}
```

The input may be array, data frame, list, matrix, single value, vector, or a combination of these. The body of function is a set of `R` commands in order to achieve the goal (output). If the output has just one part, then user is free to write `output` rather than `return(output)` for returning the output of function. It sounds worthwhile to note that if the body of function consists of only one sentence then the curly brackets (braces) and `return(output)` can be removed in order to relax the created function. Table 1.11 lists with some popular primitive built-in functions.

For example, if we want repeatedly standardize a numeric vector, then we may create a function called `my.scale` for this purpose as follows.

```
R> my.scale <- function(v) (v - mean(v) )/sd(v)
```

Of course, we do not suggest to use the user-defined function `my.scale` for practical purposes since it built-in counterpart exists. Hence, the user-defined are mainly developed to perform

Table 1.11: Some primitive built-in functions in R.

Function	output/task
abs(x)	absolute value of x.
atan(x)	inverse of trigonometric function tan(x).
exp(x)	natural exponential of x.
Im(x)	imaginary part of x.
log(x)	logarithm of x based on number y=exp(1) by default.
lgamma(x)	logarithm of $\Gamma(x)$.
Re(x)	real part of complex number x.
sign(x)	takes on values -1, 0, or 1, if x<0, x=0, or, x>0, respectively.
sqrt(x)	square root of x.
x^y	x to power y (often symbol ^ is created by Shift + 6).

those tasks for which the built in function is not applicable. As another example, suppose we would like to create a function that computes the sign of a given real value. Recall that we have just computed the sign of vector sex. A little more work required to create function called my.sign for computing sign of given vector x. The function my.sign is given as follows.

```
R> my.sign <- function(x)
+{
+ n <- length(x)
+ y <- rep(0, n) # a numeric vector called y of size n as output
+ for(i in 1:n)
+ {
+  if( x[i] == 0)
+  {
+   y[i] <- 0
+  }else if (x[i] > 0){
+   y[i] <- 1
+  }else{
+   y[i] <- -1
+   }
+ }
+ return(y)
+}
```

It is worthwhile to note that each user-defined function must be checked to see does it work well or not, and next priorities are speed, memory usage, and etc.

1.5.1 Built-in function for character

Working with character or string is also important. The built-in functions have been developed effectively to deal with such type of variables in R. Herein, among them, we briefly introduce grep, grepl, gregexpr, letters, regexec, regexpr, sub, gsub, substr, strsplit, toupper, tolower. These functions, in general, have been developed for finding some given pattern within a non-numeric vector. Among aforementioned commands, we are willing to discuss briefly the commands grep and substr. Details about arguments of grep are given in Table 1.12. Suppose we would like to find pattern 'Bd' inside the non-numeric atomic vector

x <- c('abcd','Bdcd','abcdabcd','bd2'). Hence, we define

```
R> x <- c('abcd','Bdcd','abcdabcd','bd2'); pattern <- 'Bd'
```

We run command grep(...) for four different combinations of its arguments as follows.

```
R>grep(pattern,x,ignore.case=T,value=T) R>grep(pattern,x,ignore.case=T,value=F)
```

Table 1.12: Details about main arguments of `grep` in `R`.

Argument	output/task
`pattern`	a sequence of character(s) that we want to find it within some given non-numeric vector.
`x`	given non-numeric vector.
`ignore.case`	`logical`: if `TRUE`, the matched sequence will be found regardless of being upper or lowercase.
`value`	`logical`: if `TRUE`, then it returns the vector of matched sequence; otherwise, the indices vector is returned.

```
[1] "Bdcd" "bd2"
```

```
R>grep(pattern,x,ignore.case=F,value=T)
```

```
[1] "Bdcd"
```

```
[1] 2 4
```

```
R>grep(pattern,x,ignore.case=F,value=F)
```

```
[1] 2
```

Details about arguments of function `substr(...)` is given in Table 1.13. This command is mainly used to extract or replace substrings in a given non-numeric vector. For example, if

Table 1.13: Arguments of `substr(...)` in `R`.

Argument	output/task
`x`	a non-numeric vector.
`start`	first element to be extracted (or replaced).
`stop`	last element to be extracted (or replaced).
`value`	a string to be replaced.

we want to substitute the string `'sm'` into the second and third places of elements of vector `x` defined earlier, or further, extract the second and third elements of vector `x`. The corresponding commands and outputs are given as follows. In a further example, e are willing to separate digits

`R` code	output
`R>x<-c('abcd','Bdcd','abcdabcd','bd2')`	
`R>value<-'sm'; substr(x,2,3)<-value`	
`R>x`	`[1] "asmd" "Bsmd" "asmdabcd" "bsm"`
`R>x<-c('abcd','Bdcd','abcdabcd','bd2')`	
`R>substr(x,2,3)`	`[1] "bc" "dc" "bc" "d2"`

of a given number. To this end, we may use `gsub` command to separate digits of 2024 as follows.

`R` code	output
`R> val1 <- as.character(2024)`	`[1] 2 0 2 4`
`R> val2 <- gsub("(.)", "\\1 ", val1)`	
`R> val3 <- strsplit(val2, " ")[[1]]`	
`R> val4 <- as.numeric(val3); val4`	

1.6 External R package

1.6.1 External R package initialization

As mentioned before, attaching `R` packages is privileged to users for implementing the advanced or more technical statistical methods. Herein, we would like to deal with utilizing an

external R package. To this end, the first step is prepare the package's documentation (or Reference manual) that is given in pdf available at `https://cran.r-project.org/web/packages/pakagename`. The documentation gives necessary information on R package. There are several features about each R package that, for example, may be seen at `https://cran.r-project.org/doc/manuals/r-devel/R-exts.html#The-DESCRIPTION-file`. These features must be investigated whenever user wants to install and then use a package, among them we mention the following.

i. `Depends`: This field gives a numbers of comma-separated package names that the desired package depends on. The prerequisite packages will be installed while the current package is installed. This field also may specify a dependence on a certain version of R. For instance, if the desired package works only with version 4 or later, we may see `R (>= 4.0)` in the "Depends" field.

ii. `Suggests`: This field has the same characteristics as field "Depends" does. But, the list of packages in front of this field are not necessary to install. The prerequisite package(s) are just used for implementing examples, tests existing in the desired package.

iii. `Package source`: This fields contains the compressed source code of the desired package. For example, the compressed source code of package `nortest` is given by `nortest_1.0-4.tar.gz` in this field.

1.6.2 External R package installation

In this part, we are willing to deal with installing an external Rpackage. To this end, we select the `nortest` available at `https://cran.r-project.org/package=nortest`. The first step for installing an external Rpackage is study the documentation file of `nortest` as mentioned before. If all features hold true, then we use the steps given by the following for using this package for all practical purposes.

```
R > address = "E:/my R code"
R > install.packages("nortest", lib=address,
+   rpos="https://cran.r-project.org/package=nortest")
```

Running above two lines there appears a box entitles "Secure CRAN mirrors" that lists all available mirrors for installing the desired package. We suggest to select the nearest country, if you country is not listed there, and then approving to continue the installing process. Depending on the desired package's dependency on other packages, the installation process may take several seconds or minutes. Finally, a blue-colored message appears in console session indicating that the installation process is accomplished successfully. After installing, we must call the desired package through

```
R > library(nortest) # or require(nortest)
```

It is worth to mention that the argument `address` is quite optional and determined by user. The argument `rpos` is obtained from the last line in the package's page at CRAN. The above process must be done for using all external packages. Sometimes, due to some technical problems, user is not able to install the desired package. One way to avoid such a issue is to install the desired package manually. To this end, user just needs to download the package's compressed source code available at `Package source` part. Then, we determine the address of the package source through

```
install.packages(file.choose( ), repos = NULL, type="source")
```

to complete the installation process.

1.6.3 Normality test

Testing the normality of a data set is a vital statistical hypothesis in many parametric and non-parametric statistical methods [Thode, 2002, Stephens, 2017]. Though implementing the goodness-of-fit tests does not need an in-depth `R` programming skill, but we would prefer to use some external package for this purpose rather to create our user-defined function. After installing package `nortest`, having a look at topics, we can see that function `ad.test` has been developed for accomplishing the Anderson-Darling test of normality. There is more detailed information on this function in the pertinent page. Also, the package `nortest` allows user for running more normality test such as Cramer-von Mises, Lilliefors (or Kolmogorov-Smirnov), Pearson chi-square, and Shapiro-Wilk. In what follows, we give the pertinent commands and outputs when these goodness-of-fit tests are applied to a data set of size $n = 100$ randomly generated from Gaussian distribution with mean 5 and standard deviation 3. Each test consists of the *test statistic* and corresponding p-value. Since all p-values are greater than significance level, $\alpha = 0.05$ say, then *based on the sample evidence*, we decide to accept the null hypothesis of H_0: "data follow Gaussian distribution" against H_1: "data do not follow Gaussian distribution" at this level.

```
R> set.seed(20241204)
R> x<-rnorm(100,mean=5,sd=3)
R> #we use rnorm() to generates
R> #from Gaussian distribution
```

```
R> ad.test(x)
Anderson-Darling normality test
data:  x
A = 0.3784, p-value = 0.4007
```

```
R> cvm.test(x)
Cramer-von Mises normality test
data:  x
W = 0.055022, p-value = 0.4364
```

```
R> lillie.test(x)
Lilliefors (Kolmogorov-Smirnov) normality test
data:  x
D = 0.063051, p-value = 0.4239
```

```
R> pearson.test(x)
Pearson chi-square normality test
data:  x
P = 9.72, p-value = 0.4654
```

```
R> sf.test(x)
Shapiro-Francia normality test
data:  x
W = 0.98913, p-value = 0.5087
```

More diagnostics for assessing the normality are discussed through `R` package `EnvStats` available at `https://cran.r-project.org/package=EnvStats`. It is worthwhile to note that this package needs an `R(>= 3.5.0)` and imports packages `MASS`, `ggplot2`, `nortest`. Hence, installing `EnvStats`, we should take care about the imported packages and their dependencies. The package `EnvStats` is enable to draw quantile-quantile plot (q-q plot). Suppose we would like to display the q-q plot of the data already generated from Gaussian distribution. To this end, we run the commands as

```
R> library(EnvStats); set.seed(20241204); x<-rnorm(100,mean=5,sd=3)
R> qqPlot(x, dist = "norm", estimate.params = T, digits = 2,
+ points.col = "blue",  add.line = TRUE)
R> qqPlot(x, dist = "norm", param.list=list(mean=5, sd=3),
+ estimate.params = F, digits = 2, points.col = "blue",  add.line =
+ TRUE, plot.type="Tukey Mean-Difference Q-Q")
```

to display graphical summaries depicted in Figure 1.7. It is worthwhile to say that for producing the normal q-q plot (left subfigure), we set argument `estimate.params=T` and hence the package automatically sets `param.list=list(mean=mu, sd=sigma)` in which `mu=5.4` and `sigma=2.7` are the average and standard deviation of sample evidence `x`. For producing the

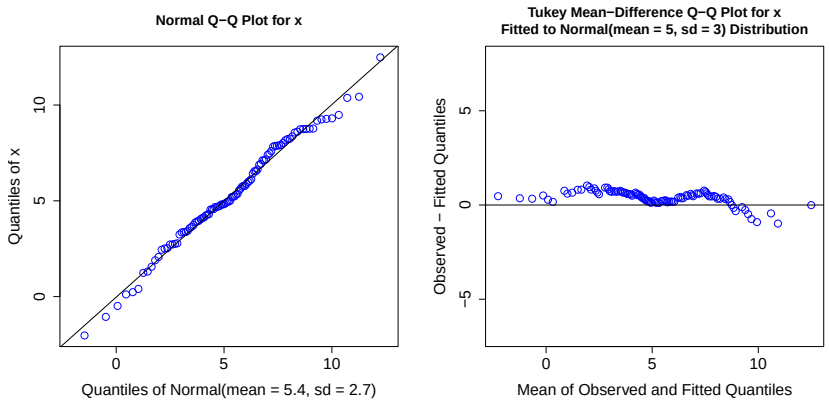


Figure 1.7: (left subfigure): q-q plot and (right subfigure): Tukey mean-difference q-q plot.

Tukey mean-difference q-q plot (right subfigure), we set `estimate.params=F` and hence we need to put the true mean and standard deviation using argument `param.list=list(mean=5, sd=3)`.

1.7 Regression analysis in R

The simple linear, multiple linear, and non-linear regression analysis are among applied topics is statistics with widespread use in other fields of study. Herein, we are willing to discuss how user can implement the regression analysis in `R`.

1.7.1 Linear regression

Suppose the amount of some variable that is known in the literature as *response* or *dependent* variable depends *linearly* on one (simple case) or more (multiple case) variable(s) namely *independent* variable(s) or *covariate*. The linear regression enables us to estimate the value of dependent variable y when a level of independent x is given. The set of dependent and independent variables should be provided in a data frame structure. For practical purposes, we will focus on *built-in* `women` data that consists of two variables average heights and weights for American women. Hence, for detecting how the dependent (weight) and independent (height) variables are linearly correlated, we proceed as follows.

```
R> data(women)
R> dat <- data.frame(y=women$weight, x=women$height)
R> model <- lm(y~x, data=dat) # lm accounts for linear model
R> out <- summary(model)
```

The summary of regression analysis such as estimated coefficients, estimated residuals, etc., are used for further analysis. Table 1.14 lists some commands to access more detailed summary related to the simple linear regression analysis. Figure 1.7 displays our user-defined function

created for depicting the dependent and independent scatterplot. The pertinent `R` function called `reg_plot(x,y,...)` is given as follows.

```
1 R> reg_plot <- function(x,y,PI=FALSE, interval=c("c","p"),...)
2 + {# ''c'' for confidence and ''p'' for prediction interval
3 + model <- lm(y~x)
4 + out <- summary(model)
5 + b0 <- round(out$coefficien[1,1],4)
6 + b1 <- round(out$coefficien[2,1],4)
```

Table 1.14: Details about output of `lm(...)` in `R`.

Command	output
`out$coefficients[1,1]`	estimated intercept.
`out$coefficients[2,1]`	estimated slope.
`out$coefficients[1,2]`	standard error of estimated intercept.
`out$coefficients[2,2]`	standard error of estimated slope.
`out$coefficients[1,4]`	p-value corresponds to null. hypothesis of intercept is zero.
`out$coefficients[2,4]`	p-value corresponds to null hypothesis of slope is zero.
`out$sigma`	estimated response variable standard error.
`out$r.squared`	coefficient of determination R^2.
`confint(model,level=0.95)`	0.95% confidence interval for regression coefficients.
`out$residuals`	residuals of fitted model.
`anova(model)`	analysis of variance table.
`predict(object,new,...)`	object is `lm(y x)` and `new` is a single or vector of values for which a confidence (`"c"`) or prediction (`"p"`) interval at `level=0.95` is constructed.

```
+ note1 <- paste("Y=", b0, "+", b1, "X")
+ note2 <- paste("R.squared=",round(out$r.squared,4))
+ n <- length(x)
+ new <- data.frame(x=seq(min(x), max(x), length =n))
+ if(PI == FALSE)
+ {
+  plot(x,y,...)
+  curve(b0 + b1*x, add=T)
+  lines(new$x, b0 + b1*new$x)
+  text(mean(x), max(y), note1)
+  text(mean(x), quantile(y, 0.90), note2)
+ }else if(PI == TRUE){
+  pred <- predict(model, new, interval = interval)
+  plot(x,y,...)
+  x0<-c(new$x[1:n], new$x[n:1])
+  y0<-c(pred[,2], pred[n:1,3])
+  polygon(x0,y0,col="steelblue", border="red")
+  text(mean(x), max(y), note1)
+  text(mean(x), quantile(y, 0.90), note2)
+  points(x,y,...)
+  curve(b0 + b1*x, add=T)
+ }
+}
```

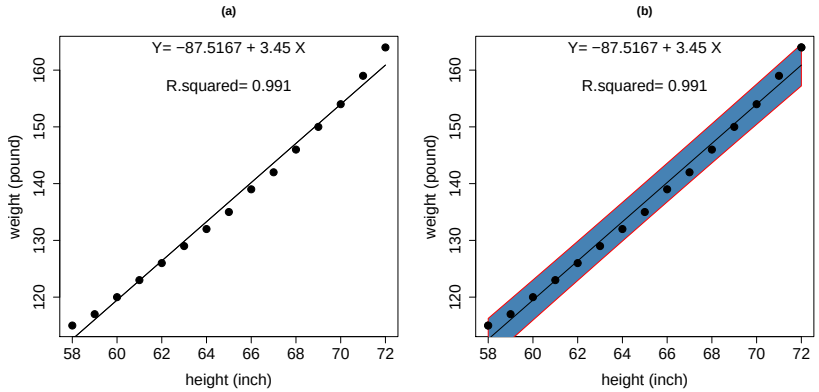


Figure 1.8: (a): Scatteplot of `women` data: (a) fitted regression line and (b): prediction interval. We run the `R` commands as follows for producing Figure 1.8.

```
R> data(women)
R> x <- women$height; y <- women$weight
R> reg_plot(x,y,PI=F,interval="p",pch=19,cex=1.5,
+ cex.lab=1.5,cex.axis=1.5,main="(a)",
+ xlab="height (inch)",ylab="weight (pound)")# for subfigure (a)
R> reg_plot(x,y,PI=T,interval="p",pch=19,cex=1.5,
+ cex.lab=1.5,cex.axis=1.5,main="(b)",
+ xlab="height (inch)",ylab="weight (pound)")# for subfigure (b)
```

1.7.2 Non-linear regression

In some applications relation between dependent and independent variables is not linear. There are infinite number of non-linear mathematical models for fitting to data. Herein, we would like to deal with the growth model. An application of such models can be found in forestry for describing the relation between tree's height (h in m/foot) and its diameter at breast height (dbh in cm/inch) Teimouri et al. [2020]. Among the most commonly used growth models, we focus on Weibull model whose relation is given by Yang et al. [1978]:

$$h = 1.3 + b1\left[1 - e^{-b2d^{b3}}\right], \tag{1.1}$$

where $b1 > 0$, $b2 > 0$ and $b3 > 0$ are the model's parameters [3]. There are different ways for estimating the model's parameters based on the nonlinear least squares method, among them herein, we suggest the use of function `nls(formula,data,start,...)`. Detail on main arguments of the former function is shown by Table 1.15.

Table 1.15: Details about main arguments of `nls(...)` in `R`.

Argument	output/task
`formula`	mathematical expression such as formula (1.1).
`data`	data frame consisting of dependent and independent data.
`start`	vector of starting (initial) values of the model's parameters.

The `nls(...)` uses some iterative algorithms to estimate the parameters. But, the performance (being convergent or not) of these algorithms in general is sensitive on the initial values. Not surprisingly, if starting values are close to the global maximum, then algorithm converges soon. Otherwise, the algorithm will not converge and `nls(...)` fails to estimate model's parameters. One way to deal this practical issue is the use of function `tryCatch(...)` that has mainly four parts described as Table 1.16. In what follows, we run function `nls(...)` for

Table 1.16: Details about main arguments of `tryCatch(...)` in `R`.

Argument	output/task
`expr`	the task to be accomplished.
`error`	the task to be done if the given task in part `expr` if an error is caught.
`warning`	the task to be done if accomplishing the given task in part `expr` does not fail, but a warning is caught.
`finally`	the task to be done regardless of accomplishing the given task in part `expr` is successful or not successful.

`DBH` data involved in package `ForestFit` under two scenarios including: i) true parameters are assumed to be known and function `tryCatch(...)` is not employed and ii) true parameters

[3]The constant 1.3 is model's intercept since the experimenter's height at breast level is on the average 1.3 m.

are assumed to be unknown and function `tryCatch(...)` is employed. We deal with these two scenarios as follows.

i. **First scenario**: We assume that true parameters are $b1 = 25.011$, $b2 = 0.0167$, and $b3 = 1.1561$. Using this setting, we use the command

```
R> library(ForestFit); data(DBH)
R> d <- DBH[DBH[,1]==55,10]; h <- DBH[DBH[,1]==55,11]
R> start <- c(25.011,0.016, 1.156)
R> relation <- as.formula(h~1.3+b1*(1-exp(-b2*d^b3)))
R> out <- summary( nls( relation, data=data.frame(h=h, d=d),
+ start = list( b1 = start[1], b2 = start[2], b3 = start[3] )) )
R> out
```

to see the output as follows.

```
Formula: h ~ 1.3 + b1 * (1 - exp(-b2 * d^b3))
Parameters:
    Estimate Std. Error t value Pr(>|t|)
b1 25.011868   2.400674  10.419 1.26e-14 ***
b2  0.016704   0.004924   3.392  0.00129 **
b3  1.156198   0.124108   9.316 6.63e-13 ***
---
Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
Residual standard error: 1.558 on 55 degrees of freedom
Number of iterations to convergence: 4
Achieved convergence tolerance: 8.725e-06
```

ii. **Second scenario**: We let the model's parameter are not precisely known. Hence, we use the function `tryCatch(...)` to implement the `nls(...)` function. The pertinent `R` code and output are given by the following.

```
R> library(ForestFit); data(DBH)
R> d <- DBH[DBH[,1]==55,10]
R> h <- DBH[DBH[,1]==55,11]
R> start <- c(1, 1, 1)
R> relation<-as.formula(h~1.3+b1*(1-exp(-b2*d^b3)))
R> f<-function(x, par)
+  {
+    b1<-par[1]; b2<-par[2]; b3<-par[3]; 1.3+b1*(1-exp(-b2*x^b3))
+  }
R> it <- 1; i <- 0
R> while(it <= 1)
+ {
+ r1<-runif(1,max(0,start[1]-10),start[1]+10) # b1 is selected randomly
+ r2<-runif(1,max(0,start[2]-10),start[2]+10) # b2 is selected randomly
+ r3<-runif(1,max(0,start[3]-10),start[3]+10) # b3 is selected randomly
+ out <- tryCatch(summary( nls( relation, data=data.frame(h=h, d=d),
+ start = list( b1 = r1, b2 = r2, b3 = r3 ) ) ),
+ error=function(e)( "fail" )  )
+   if( out[1] == "fail" )
+   {
```

```
+       it <- 1
+    }else{
+       it <- 2
+    }
+ i <- i + 1
+}
R> out
```

As it may be seen, we set the starting value as `start=c(1,1,1)` in second scenario and function `tryCatch(...)` allows to test different starting values during each test within `while(...)` function. As expected, the estimated regression coefficients under both both scenario are the same. We further applied the function `tryCatch(...)` to estimate three parameters of the logistic growth curve. It suffices to consider as follows.

```
R> relation<-as.formula(h~1.3+b1/(1+b2*exp(-b3*d)))
R> f<-function(x, par)
+ {
+    b1<-par[1]; b2<-par[2]; b3<-par[3]; 1.3+b1/(1+b2*exp(-b3*x))
+ }
```

Figure 1.9 displays the scatterplot of height and diameter variables while the estimated Weibull (left subfigure) and logistic (right subfigure) growth curves are added to the plot. It is worth to note that the output corresponds to fitting logistic growth curve is as follows.

```
Formula: h ~ 1.3 + b1/(1 + b2 * exp(-b3 * d))
Parameters:
    Estimate Std. Error t value Pr(>|t|)
b1 21.688329   0.814105  26.641  < 2e-16 ***
b2  6.878022   1.212616   5.672 5.42e-07 ***
b3  0.086529   0.009079   9.530 3.04e-13 ***
---
Signif. codes: 0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
Residual standard error: 1.802 on 55 degrees of freedom
Number of iterations to convergence: 9
Achieved convergence tolerance: 6.395e-06
```

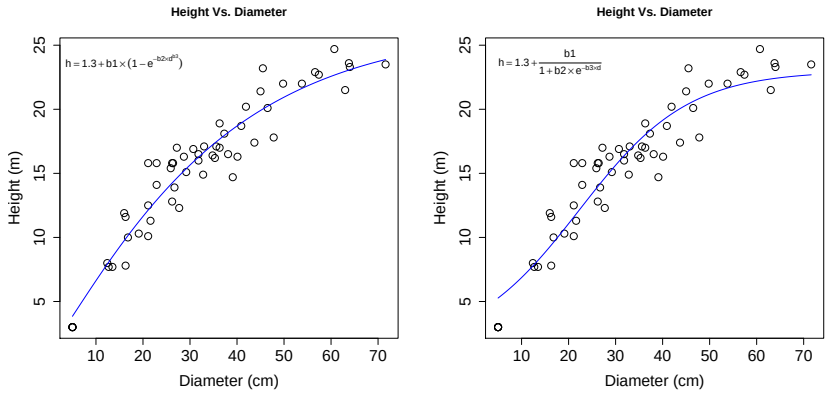


Figure 1.9: Scatterplot of height and diameter data with fitted Weibull (left subfigure) and logistic (right subfigure) growth curves.

2

Common simulation techniques

This section focuses on some important simulation approaches including: *Metropolis-Hastings algorithm*, *Inverse transform method*, *Distributional identity*, *Rejection sampling*, *Adaptive rejection sampling*, and *Ratio of uniforms method*, and *Generalized ratio of uniforms method*. We further deal with the methods for simulating from multivariate Gaussian distributions. Throughout, we assume that generation from uniform distribution on (0,1), that is $\mathcal{U}(0,1)$, is accomplished by the computer machine.

2.1 Metropolis-Hastings algorithm

In this part, we describe the Metropolis-Hastings algorithm. To this end, first, we review some preliminaries.

Definition 2.1.1. *A collection of sample evidence (random variables) indexed by time set $t \in \{0,1,2,\ldots\}$, represented as $\{X_t\}_t$, is called a stochastic or random process.*

Definition 2.1.2. *A stochastic process $\{X_t\}_{t\geq 0}$ is said to be Markov process if*

$$p(X_n = x_n | X_0 = x_0, \cdots, X_{n-1} = x_{n-1}) = p(X_n = x_n | X_{n-1} = x_{n-1}), \tag{2.1}$$

that means the position (or state) of a Markov process (chain) at time n just depends on its position at previous state $n-1$ for $n \geq 1$.

Definition 2.1.3. *For a given pair (i,j), if there is a positive probability that chain will ever visit state j while it starts with state i, then this chain is called irreducible.*

Let π_j denote the long-run proportion of times that an irreducible Markov chain visits state j. It can be shown that π_j exists, independent of the chain's initial state i, and is given as the solution of the equations

$$\pi_j = \sum_{i=1}^{N} \pi_i P_{ij}, \text{ for } j = 1, \cdots, N. \tag{2.2}$$

The system of equations in (2.2) can be represented in matrix form as

$$\boldsymbol{\pi}^\top = \begin{bmatrix} \pi_1 \\ \pi_2 \\ \vdots \\ \pi_j \\ \vdots \\ \pi_N \end{bmatrix}^\top = \begin{bmatrix} \pi_1 \\ \pi_2 \\ \vdots \\ \pi_j \\ \vdots \\ \pi_N \end{bmatrix}^\top \times \begin{bmatrix} P_{11} & P_{12} & \cdots & P_{1N} \\ P_{21} & P_{22} & \cdots & P_{2N} \\ \vdots & \vdots & \vdots & \vdots \\ P_{j1} & P_{j2} & \cdots & P_{jN} \\ \vdots & \vdots & \vdots & \vdots \\ P_{N1} & P_{N2} & \cdots & P_{NN} \end{bmatrix}. \tag{2.3}$$

The vector $\boldsymbol{\pi}$ is known as the *limiting distribution* or the *stationary distribution* and π_j is called the *limiting probability* that the Markov chain visits state j.

Definition 2.1.4. *An irreducible Markov chain is said to be aperiodic if we have*

$$P(X_n = j|X_0 = j) > 0, \text{ for } n \geq 0. \tag{2.4}$$

Definition 2.1.5. *A Markov chain with limiting probability π_j and transition probabilities P_{ij} is said to be time-reversible if we have $\pi_i P_{ij} = \pi_j P_{ji}$ for all $i \neq j$.*

Suppose we are willing to simulate from random variable for which we have $P(X = j) = \pi_j$ for $j = 1, \cdots, m$ (m is large). One way is the use of MH algorithm that constructs a time-reversible Markov chain $\{X_n\}_{n\geq 0}$ whose limiting probability is π_j. To this end, we consider an irreducible Markov chain with integer states $\{1, \cdots, m\}$ whose transition probability matrix is $\boldsymbol{Q}$, see Ross [2022]. Let the (i, j)-element of $\boldsymbol{Q}$ is denoted as $q(i, j)$. If $X_n = i$, then generation X is drawn from distribution with probability mass function $P(X = j) = q(i, j)$ for $j = 1, \cdots, m$. If $X = j$, then X_{n+1} would be i and j, accordingly, with probabilities $1 - \alpha(i, j)$ and $\alpha(i, j)$. The transition probabilities of the constructed Markov chain becomes

$$\begin{aligned} P_{i,j} =& q(i, j)\alpha(i, j), \text{ for } j \neq i, \\ P_{i,i} =& q(i, i) + \sum_{k\neq i} q(i, k)[1 - \alpha(i, k)]. \end{aligned}$$

The time-reversibility of this Markov chain with limiting distribution π_j holds true if

$$\pi_i P_{i,j} = \pi_j P_{j,i} \text{ for } j \neq i,$$

that implies

$$\pi_i q(i, j)\alpha(i, j) = \pi_j q(j, i)\alpha(j, i). \tag{2.5}$$

It is not hard to see that the identity in (2.5) is satisfied if we have

$$\alpha(i, j) = \min\left\{1, \frac{\pi_j q(j, i)}{\pi_i q(i, j)}\right\}. \tag{2.6}$$

The limiting probability π_j may be represented as $\pi_j = b_j/B$ in which $B = \sum_{j=1}^{m} b_j$ for $b_j > 0$ and sufficiently large m. Evidently, factor B plays the role of normalizing constant, and hence the right-hand side (RHS) of (2.6) can be represented as

$$\alpha(i, j) = \min\left\{1, \frac{b_j q(j, i)}{b_i q(i, j)}\right\}. \tag{2.7}$$

The steps of the MH algorithm is given by Algorithm 1. In practice, the MH algorithm is use frequently to simulate from continuous distribution with probability density function (PDF) $f(\cdot|\theta)$ that coincides with the limiting distribution π_j, of irreducible Markov chain. But, as it is seen from Algorithm 1, the MH algorithm produces dependent generations sine the next generation is created based on the current one. Hence, all estimators constructed based on the generated samples are biased and inefficient. To avoid this problem, the initial generated samples generated sample are removed from the whole [Robert et al., 2004]. The length L, of the removed initial samples is known as *burn-in* or *warm-up* for which some criteria has been determined [Hobert and Robert, 2004]. It can be shown that after sufficiently large number M of generations, the generated samples are statistically uncorrelated and converge to the chain's limiting distribution that coincides with distribution of PDF $f(\cdot|\theta)$. We follow the steps of Algorithm 2 for implementing the MH algorithm to simulate one generation from PDF $f(\cdot|\theta)$.

Algorithm 1 MH-algorithm for generation one sample from target PDF $\pi_j = b_j/B$

1: Construct an irreducible Markov chain with transition probabilities $q(i,j)$ (for $i,j = 1,\cdots,m$) and suggest an integer number c such that $1 \leq c \leq m$;
2: Set $n = 0$ and $x_0 = c$;
3: Generate x_{new} form probability mass function $P(X_{new} = j) = q(x_n, j)$;
4: Generate a random variable $u \sim \mathcal{U}(0,1)$;
5: **if** $u < [b_x q(x_{new}, x_n)]/[b_{x_n} q(x_n, x_{new})]$ **then**
6: $\quad x_{new} \leftarrow x$
7: **else**
8: $\quad x_{new} \leftarrow x_n$;
9: **end**
10: $n \leftarrow n+1$ and $x_n \leftarrow x_{new}$;
11: go to step 3.

It is worthwhile to mention that $q(\cdot|\cdot)$ is known as the *proposal* or *candidate* distribution and $f(\cdot|\theta)$ is also referred to as *target* distribution in the literature. The proposal must be easy to sampling from and moreover we have $\mathcal{S}_q = \mathcal{S}_f$ [1]. If the next state is obtained by adding some random variable to the current state then we have a *random walk* proposal and the algorithm is called sometimes random walk Metropolis algorithm. Furthermore, if the random variable follows a symmetric distribution, then the acceptance rate or transition probability takes a simpler form. In such a case, we have $q(x_{n+1}|x_n) = q(x_n|x_{n+1})$. The original Metropolis [Metropolis et al., 1953] algorithm was constructed based on a symmetric random walk proposal. For example, suppose that chain's current state is X_n and we use a standard Gaussian distribution, as the proposal, for determining the next state X_{n+1}. It follows that

$$X_{n+1} = X_n + Y,$$

where $Y \sim \mathcal{N}(\mu = 0, \sigma = 1)$. It is easy to check that PDF of the next state becomes $\mathcal{N}(x_{n+1}|\mu = x_n, \sigma = 1)$ or equivalently

$$q(x_{n+1}|x_n) = \mathcal{N}(x_{n+1}|\mu = x_n, \sigma = 1).$$

Obviously, since $\mathcal{N}(x_{n+1}|\mu = x_n, \sigma = 1) = \mathcal{N}(x_n|\mu = x_{n+1}, \sigma = 1)$, we have

$$q(x_{n+1}|x_n) = q(x_n|x_{n+1}). \tag{2.8}$$

The above argument holds true for any $\sigma > 0$. Therefore the acceptance rate given in Algorithm has the simpler form

$$A = \min\left\{1, \frac{f(x_{n+1}|\theta)}{f(x_n|\theta)}\right\}.$$

Another example for a random walk Metropolis algorithm is obtained by considering the uniform distribution on $(a-3b, a+3b)$ as the proposal in which $a \in \mathbb{R}$ and $b > 0$. In this case, we have $q(x_{n+1}|x_n) = q(x_n|x_{n+1}) = 1/(6\sigma)$. The algorithm proposed by Metropolis et al. [1953] was then developed by Hastings [1970] for general case including both symmetric and asymmetric $[q(x_{n+1}|x_n) \neq q(x_{n+1}|x_n)]$ cases. Today the latter case is known as the Metropolis-Hasting algorithm. A privilege of the MH algorithm is its capability to apply for the cases that the target distribution needs only to be specified up to proportionality constant. The care must taking into account that target PDF must never be zero at the initial state x_0 and the candidate

[1] $\mathcal{S}_f = \left\{x \in \mathbb{R} \middle| f(x|\theta) \geq 0, \int_{\mathcal{S}_f} f(x|\theta)dx = 1\right\}$.

$q(\cdot|\cdot)$ must have a broad enough support to visit any state of $\mathcal{S}_f$ with a positive probability while gets as simple as possible form. We give more details on finding a suitable proposal through the following example.

Algorithm 2 MH-algorithm for generation one sample from target PDF $f(\cdot|\theta)$

1: Set $n = 0$, read L, and choose the current state (initial generation) x_0;
2: **while** $n \leq L$ **do**
3: Sample x_{new} from the proposal distribution with PDF $q(\cdot|x_n)$;
4: Compute

$$A = \min\left\{1, \frac{f(x_{new}|\theta)q(x_n|x_{new})}{f(x_n|\theta)q(x_{new}|x_n)}\right\};$$

5: Generate $u \sim \mathcal{U}(0,1)$;
6: **if** $u < A$ **then**
7: Set $n \leftarrow n+1$ and $x_n \leftarrow x_{new}$;
8: **else**
9: $y \leftarrow x_n$, $n \leftarrow n+1$, and $x_n \leftarrow y$;
10: **end**
11: **end**
12: Accept x_L as a generation form target distribution f.

Example 2-1

Let $f(x|\theta) \propto (x+1)^{-2}x^{\theta}\exp\{-x^{\theta}\}$ for $x > 0$ denote the PDF of target distribution in which $\theta > 0$ is the family parameter. Considering the Weibull distribution with shape parameter θ and scale unity as the proposal distribution whose PDF is denoted as $\mathcal{W}(\cdot|\theta, 1)$, we represent the acceptance rate (transition probability) as

$$\begin{aligned} A &= \min\left\{1, \frac{(x_{n+1}+1)^{-2}x_{n+1}^{\theta}\exp\{-x_{n+1}^{\theta}\}\mathcal{W}(x_n|\theta,1)}{(x_n+1)^{-2}x_n^{\theta}\exp\{-x_n^{\theta}\}\mathcal{W}(x_{n+1}|\theta,1)}\right\}. \\ &= \min\left\{1, \frac{(x_n+1)^2 x_{n+1}}{(x_{n+1}+1)^2 x_n}\right\}. \end{aligned} \tag{2.9}$$

Fortunately, though using an asymmetric proposal, the acceptance rate has a simple form. Setting $\theta = 5$, we follow the steps given by the following for simulating 800 realizations from $f(x|\theta = 5)$. It is worth to highlight that we run Algorithm 2 for $Ls = 1000$ times of which first $Lb = 200$ (burn-in point) generations are removed.

- Set $n = 0$, $Lb = 200$, $Ls = 1000$, and the initial state (generation) $x_0 = 1$;
  1. Generate $x_{new} \sim \mathcal{W}(\theta = 5, 1)$
  2. Compute $A = \min\left\{1, \frac{(x_n+1)^2 x_{new}}{(x_{new}+1)^2 x_n}\right\}$;
  3. Generate $u \sim \mathcal{U}(0,1)$;
  4. If $u < A$, then $n \leftarrow n+1$ and $x_n \leftarrow x_{new}$;
  5. Repeat algorithm from step 1;
- Stop algorithm if $n = Lb$ and then accept x_{Lb} as a generation from $f(x|\theta = 5)$.

The `R` code, for generating from PDF $f(x|\theta = 5)$ given in Example 2-1 is given as follows.

```
R> set.seed(20240412)
R> theta <- 5
R> Ls <- 1000 # total number of iterations
R> Lb <- 200  # burn-in point
R> x <- rep(NA, Ls)
R> x0 <- 1; x[1]<- x0 # initial value
R> for(n in 1:Ls)
+ {
+  x.new <- rweibull(1, shape = theta, scale = 1)
+  A <- 2*log(x[n] + 1) + log(x.new) -
+       2*log(x.new + 1) - log(x[n])
+  if( runif(1) < exp( A ) )
+  {
+   x[n+1] <- x.new # x.new is accepted
+  }else{
+   x[n+1] <- x[n]  # x.new is rejected
+  }
+ }
R > hist(x[(Lb+1):Ls], prob=T, br=20)
```

As it is seen in example above, the Weibull choice of candidate yields a simple form of transition probability. Figure 2.1(b) shows the chain's motion across $Ls = 1000$ iterations while the initial state is $x_0 = 4$. As it is seen, the convergence occurs very soon. There have been introduced some extensions of the MH algorithm in the literature. We refer reader to for more details about extensions such as [Robert et al., 2004, Roberts and Rosenthal, 2009, Robert et al., 2010]. There have been introduced some extensions of the MH algorithm in the literature.

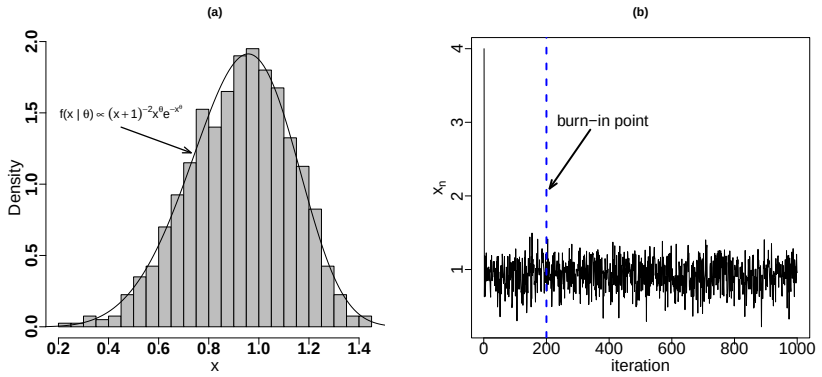


Figure 2.1: (a): Histogram (constructed based on chain's output between $201th$ up to $1000th$ iterations); superimposed is $f(x|\theta = 5)$ and (b): Chain's motion across iterations for sampling from $f(x|\theta = 5)$ with starting value $x_0 = 4$.

We refer reader to for more details about extensions such as [Robert et al., 2004, Roberts and Rosenthal, 2009, Robert et al., 2010].

2.2 Inverse transform method

Throughout this chapter let random variable X from which drawing sample is of interest has closed form cumulative distribution function (CDF), denoted herein by $F(\cdot)$, or is computed using accurate computer algorithms such as Gaussian distribution. In what follows $X \sim F(\cdot)$ or $X \sim f(\cdot)$ will denote the random variable X follows CDF $F(\cdot)$ or PDF $f(\cdot)$. The following proposition plays main role for simulating from X through the Inverse transform method.

Proposition 2.2.1. *Let $X \sim F(\cdot)$ and $U \sim \mathcal{U}(0,1)$. Then,*

$$X \overset{d}{=} F^{-1}(U),$$

where "$\overset{d}{=}$" denotes the identity in distribution and $F^{-1}(\cdot)$ is inverse of CDF or the quantile function.

Based on Proposition (2.2.1) in above, for simulating from random variable with CDF $F(\cdot)$ (or PDF $f(\cdot)$), first, one can simulate a generation such as u from $\mathcal{U}(0,1)$ and then a sample such as x is produced by computing $F^{-1}(u)$. In what follows, we give an example that accomplishes this method simply for drawing sample from exponential distribution.

Example 2-2

Let $X \sim \mathcal{EXP}(\lambda)$ with CDF given by

$$F(x) = 1 - \exp\{-\lambda x\}, \tag{2.10}$$

where $\lambda > 0$ is the family rate parameter. It easy to see that

$$F^{-1}(u) = -\frac{1}{\lambda}\log(1-u), 0 < u < 1. \tag{2.11}$$

Hence, a generation from exponential family with CDF in (2.10) becomes $x = -\log(1-u)/\lambda$ in which u is drawn from $\mathcal{U}(0,1)$. The `R` function `rexp0` given by

```
R> rexp0 <- function(n, lambda = 1)
+{
+x <- -1/lambda * log( 1- runif(n) )
+return(x)
+}
```

can be used for generating n independent realizations form $\mathcal{EXP}(\lambda)$. But, it should be noted that `R` has its own function `rexp(n, rate=lambda)` for generating from this family. So, we would prefer to use the latter in simulation setting rather than the former `R` code. ■

In order to simulate from some well-known statistical families of distributions in `R`, we may run the commands given in Table 2.1. When the CDF is a piecewise function, then the quantile function can be exploited by inverting the CDF on each piece for which $0 < F(\cdot) < 1$. Assuming that random variable X has a m-piece CDF represented by

$$F(x) = \begin{cases} 0, & \text{if} \quad x < a_0, \\ F_1(x), & \text{if} \quad a_0 \le x < a_1, \\ F_2(x), & \text{if} \quad a_1 \le x < a_2, \\ \vdots & \vdots \\ F_m(x), & \text{if} \ a_{m-1} \le x < a_m, \\ 1, & \text{if} \quad a_m \le x. \end{cases} \tag{2.12}$$

For example, schematic of a three-piece CDF

$$F(x) = \begin{cases} 0, & \text{if} \quad x < 0, \\ \frac{1}{6} \times \frac{1-\exp\{-2x\}}{1-\exp\{-2\}}, & \text{if} \quad 0 \le x < 1, \\ \frac{1}{6} + \frac{x-1}{6}, & \text{if} \quad 1 \le x < 2, \\ \frac{1}{3} + \frac{2}{3} \times \frac{\exp\{-x\}-\exp\{-2\}}{\exp\{-8\}-\exp\{-2\}}, & \text{if} \quad 2 \le x < 8, \\ 1, & \text{if} \quad 8 \le x, \end{cases} \tag{2.13}$$

is displayed in Figure 2.2, We emphasize that $F_i(x)$ (for $i = 1, \cdots, m$) are not CDF. Algorithm 3 describes how to simulate from a random variable that possesses a piecewise CDF.

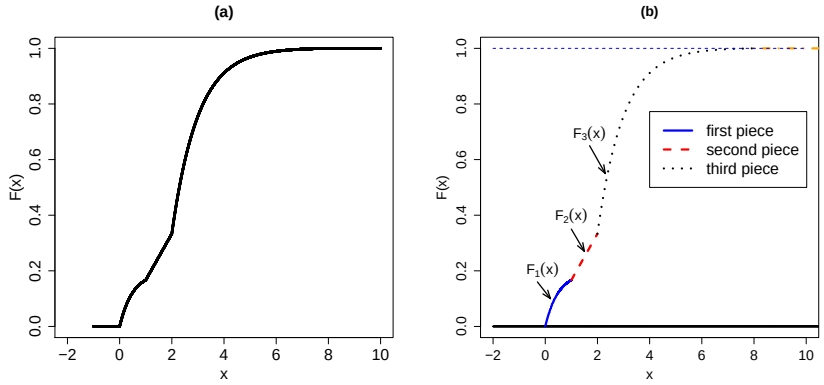


Figure 2.2: (a): a schematic of three-piece CDF and (b): the corresponding pieces.

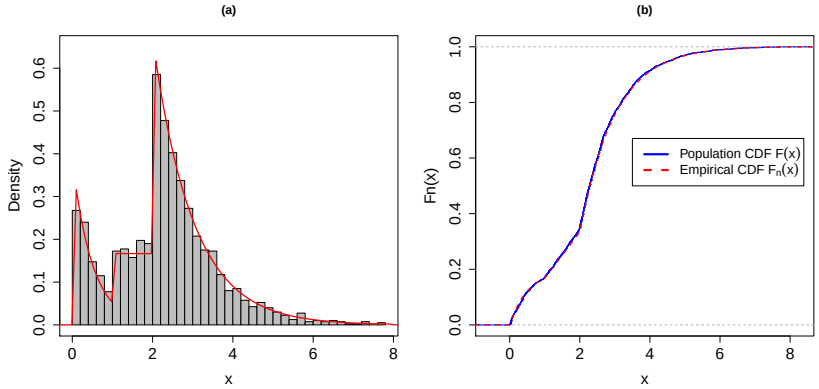


Figure 2.3: (a): Histogram of $n = 2000$ simulated data from family with CDF (2.13). The red-colored superimposed curve is the corresponding PDF. (b): The population CDF and empirical CDF.

2.3 Distributional identity

Herein, we present some examples in which generating from the distribution of interest is accomplished through its relation with other distribution(s) from which sampling is an easy or easier task. One of the most widely used distribution, that is the Gaussian, can be simulated through this approach. Although the quantile function of the Gaussian distribution can be computed numerically in `R` an then the inverse transform method can be applied to sample from, alternatively an efficient method known as the Box-Muller transform proposed by Box and Muller [1958] is suggested for this purpose as follows. It needs to generate independent random variables $U_1, U_2 \sim \mathcal{U}(0, 1)$ and then it can be proved that

$$Z_1 = \sqrt{-2 \log U_1} \cos(2\pi U_2),$$
$$Z_2 = \sqrt{-2 \log U_1} \sin(2\pi U_2),$$

independently follow a standard Gaussian distribution. Consequently, either $\mu + \sigma Z_1$ or $\mu + \sigma Z_2$ follows $\mathcal{N}(\mu, \sigma^2)$. If evaluating the sine or cosine functions wastes the computational time, one can use the method proposed by Marsaglia and Bray [1964] that is described below by Algorithm 4. The pertinent `R` function `rnorm0` given below can be used for generating Gaussian realizations based on Marsaglia-Bray method.

```
R> rnorm0 <- function(n, mu, sigma)
+{ # n is supposed to be an even number
```

Algorithm 3 Simulation from random variable with piecewise CDF

Read n and sequence $\{F(a_0), F(a_1), \cdots, F(a_m)\}$;
Generate u from $\mathcal{U}(0,1)$;
if $F(a_0) < u < F(a_1)$ **then**
    $x = F_1^{-1}(u)$;
if $F(a_1) < u < F(a_2)$ **then**
    $x = F_2^{-1}(u)$;
    $\vdots$
if $F(a_{m-2}) < u < F(a_{m-1})$ **then**
    $x = F_{m-1}^{-1}(u)$;
if $F(a_{m-1}) < u < F(a_m)$ **then**
    $x = F_m^{-1}(u)$;
return x as a generation from CDF $F(\cdot)$.

Algorithm 4 Marsaglia-Bray method for simulating Gaussian random variable

Generate U_1 and U_2 independently from $\mathcal{U}(-1,1)$;
Compute $W = U_1^2 + U_2^2$;
if $W < 1$ **then**
    $Y = W^{-1/2} \times \sqrt{-2\log W}$;
    $Z_1 = YU_1$ and $Z_2 = YU_2$;
else
    Come back to step 1 and repeat algorithm;
end
Return $\mu + \sigma Z_1$ and $\mu + \sigma Z_2$ as two independent generations from $\mathcal{N}(\mu, \sigma^2)$.

```
+i <- 1
+Z <- rep(NA, n)
+ while (i <= n/2)
+ {
+ U1 <- 2*(runif(1) - 0.5); U2 <- 2*(runif(1) - 0.5)
+ W <- U1^2 + U2^2
+  if(W < 1)
+  {
+  Y <- sqrt( -2*log(W)/W )
+  Z1 <- Y*U1; Z2 <- Y*U2;
+  Z[ (2*i - 1):(2*i) ] <- c(Z1, Z2)
+  i <- i + 1
+  }
+ }
+ return( mu + sigma*Z )
+}
```

2.3.1 Multinomial distribution

If random vector $\boldsymbol{X} = (X_1, \cdots, X_{\mathrm{K}})^\top$ follows a multinomial distribution, then its PDF is given by

$$\mathcal{MUL}(\boldsymbol{x}|N, \boldsymbol{\omega}) = \frac{\Gamma(N+1)}{\prod_{k=1}^{\mathrm{K}} \Gamma(N_k+1)} \omega_1^{x_1} \omega_2^{x_2} \cdots \omega_{\mathrm{K}}^{x_{\mathrm{K}}}, \tag{2.14}$$

where $N = \sum_{k=1}^{\mathrm{K}} x_k$ in which $x_k \geq 0$. The elements of family parameter $\boldsymbol{\omega} = (\omega_1, \cdots, \omega_{\mathrm{K}})^\top$ are constrained to be nonnegative and sum up to one, that is $\sum_{k=1}^{\mathrm{K}} \omega_k = 1$. We write $\boldsymbol{X} \sim \mathcal{MUL}(n, \boldsymbol{\omega})$ to indicate that random vector $\boldsymbol{X}$ follows a multinomial distribution with PDF given by (2.14). In order to simulate from $\mathcal{MUL}(\boldsymbol{x}|n, \boldsymbol{\omega})$ in `R`, we can use command `rmultinom(n, size, prob)` in which `n` is the sample size, `size` $= N$, and `prob` $= \boldsymbol{\omega}$. If `size` $= N = 1$, and K $= 2$ (or `prob`$=(\omega_1, \omega_2)^\top$), then multinomial distribution turns into a Bernoulli distribution with success probability ω_1.

2.3.2 Finite mixture model

The PDF of a K-component finite mixture model is given by

$$g(y|\boldsymbol{\Psi}) = \sum_{k=1}^{\mathrm{K}} \omega_k f(y|\boldsymbol{\theta}_k), \tag{2.15}$$

where $\boldsymbol{\Psi} = (\boldsymbol{\omega}^\top, \boldsymbol{\theta}_1, \cdots, \boldsymbol{\theta}_{\mathrm{K}})$ is the parameter space in which $\boldsymbol{\omega} = (\omega_1, \ldots, \omega_{\mathrm{K}})^\top$ is vector of mixing parameters such that $\sum_{k=1}^{\mathrm{K}} \omega_k = 1$. In (5.60), $f(y|\boldsymbol{\theta}_k)$ is the PDF of k-th component and hence the PDF $g(y|\boldsymbol{\Psi})$ is proper since

$$\int_{\mathbb{R}^p} g(\boldsymbol{y}|\boldsymbol{\Psi}) d\boldsymbol{y} = \sum_{k=1}^{\mathrm{K}} \omega_k \int_{\mathbb{R}^p} f(\boldsymbol{y}|\boldsymbol{\theta}_k) d\boldsymbol{y} = \sum_{k=1}^{\mathrm{K}} \omega_k = 1.$$

The weight parameter $\boldsymbol{\omega}$ provides the chance of observing sample from components. Sometimes, representation (5.60) is called the *finite mixture model* since the quantity K or number of clusters is assumed to be finite. Herein, we we suggest two approaches for generating from finite mixture model. Under the first method, for generating n realizations from a K-component mixture model, one can the multinomial distribution. In what follows, Algorithm 5 gives a pseudo code describing how to draw a sample of size n from a K-component Gaussian mixture model under the first method. Under the second method, we assume that the number of simulated

Algorithm 5 Simulating from K-component Gaussian mixture model: first method

1: Read n, K, and $\boldsymbol{\Psi} = (\boldsymbol{\omega}, \boldsymbol{\theta}_1^\top, \cdots, \boldsymbol{\theta}_{\mathrm{K}}^\top)^\top$;
2: Set Y a vector of length n;
3: Set $i = 1$;
4: **while** $i \leq n$ **do**
5: Sample $\boldsymbol{u} \sim \mathcal{MUL}(1, \boldsymbol{\omega})$;
6: Set $k_{max} = \{k | \boldsymbol{u}[k] = 1\}$ where $\boldsymbol{u}[k]$ is the kth element of vector $\boldsymbol{u}$ for
7: $k = 1, \cdots, \mathrm{K}$;
8: Sample y from PDF $f(\cdot|\boldsymbol{\theta}_{k_{max}})$
9: Set $Y[i] \leftarrow y$;
10: Set $i \leftarrow i + 1$;
11: **end**
12: Accept Y as sample of size n from K-component finite mixture model.

realization under each component, that is n_k, is known or is determined as $n_k = \lceil n \times \omega_k \rceil$ (for $k = 1, \cdots, \mathrm{K}-1$) provided that $n_{\mathrm{K}} = n - \sum_{k=1}^{\mathrm{K}-1} n_k$. Herein, the generic symbol $\lfloor n \times \omega_k \rfloor$ denotes the greatest integer value equal or less than $n \times \omega_k$. A sample of size n form K-component mixture model is obtained by merging all K simulated vectors each of size n_{K} from PDF $f(y|\boldsymbol{\theta}_k)$. Algorithm 6 gives a pseudo code describing how to accomplish this task.

Algorithm 6 Simulating from K-component Gaussian mixture model: second method

1: Read K, $n_1, \cdots, n_{\mathrm{K}}$, and $\boldsymbol{\Psi} = (\boldsymbol{\omega}, \boldsymbol{\theta}_1^\top, \cdots, \boldsymbol{\theta}_{\mathrm{K}}^\top)^\top$;
2: Set Y as a vector of length n;
3: Set $k = 1$;
4: **while** $k \leq \mathrm{K}$ **do**
5: Draw sample $\boldsymbol{y}_1$ of size n_k from PDF $f(\cdot|\boldsymbol{\theta}_k)$;
6: Set $k \leftarrow k + 1$;
7: **end**
8: Accept vector $(\boldsymbol{y}_1, \cdots, \boldsymbol{y}_{\mathrm{K}})^\top$ as sample of size n from K-component finite mixture model.

Example 2-3

Suppose we are interested in generating a sample of $n = 300$ observations from two-component Gaussian mixture model with PDF given by

$$g(y|\boldsymbol{\Psi}) = \omega_1 \times \mathcal{N}(y|\mu_1, \sigma_1^2) + \omega_2 \times \mathcal{N}(y|\mu_2, \sigma_2^2) \tag{2.16}$$

where $\boldsymbol{\Psi} = (\omega_1, \omega_2, \boldsymbol{\theta}_1^\top, \boldsymbol{\theta}_2^\top)^\top$ is whole parameter vector whose elements are

$$\begin{aligned} \omega_1 &= \frac{6}{10}, \boldsymbol{\theta}_1 = (\mu_1, \sigma_1)^\top = (-3, \sqrt{2})^\top, \\ \omega_2 &= \frac{4}{10}, \boldsymbol{\theta}_2 = (\mu_2, \sigma_2)^\top = (0, 1)^\top. \end{aligned} \tag{2.17}$$

The `R` functions `rmixnorm1` and `rmixnorm2` have been developed based on Algorithm 5 and Algorithm 6, respectively, to generate from a two-component Gaussian mixture model (2.16) with whole parameter given in (2.17). Replacing one of the commands given in Table 2.1 with `rnorm(n=1, mu=mu[[k]], sigma=sigma[[k]])` in line 17 of function `rminorm1` to simulate from other finite mixture models.

For example, the command `rgamma(n=1, shape=shape[[k]], rate=rate[[k]])` should be replaced with `rnorm(n=1, mu=mu[[k]], sigma=sigma[[k]])` for simulating from a two-component gamma mixture model with PDF given by

$$g(y|\boldsymbol{\Psi}) = \frac{6}{10} \times \mathcal{G}(y|a_1, b_1) + \frac{4}{10} \times \mathcal{G}(y|a_2, b_2), \tag{2.18}$$

if demanded. The corresponding whole parameter vector becomes

$$\begin{aligned} \omega_1 = \frac{6}{10}, \boldsymbol{\theta}_1 = (a_1, b_1)^\top = (6, 0.6)^\top, \\ \omega_2 = \frac{4}{10}, \boldsymbol{\theta}_2 = (a_2, b_2)^\top = (2, 1)^\top. \end{aligned} \tag{2.19}$$

Figure 2.4 displays the histogram of $n = 5000$ simulated data from two-component Gaussian (2.16) and gamma (2.18) mixture models. For producing Figure 2.4(a), we used function `rmixnorm1`. ■

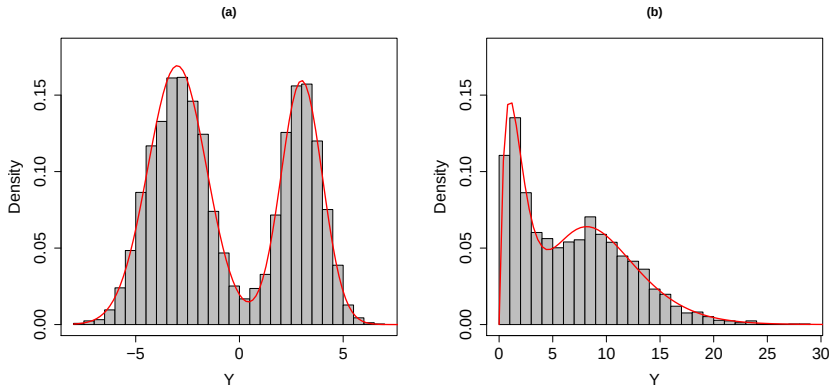


Figure 2.4: (a): histogram of $n = 5000$ simulated data from: (a) two-component Gaussian mixture model (2.16) and (b): two-component gamma mixture model (2.18). The red-colored superimposed curve is the corresponding PDFs given in (2.16) and (2.19), respectively.

```
R> set.seed(20240713)
R> n <- 5000; K <- 2; omega <- c(0.6, 0.4); mu1 <- -3; mu2 <- 0; sigma1 <- 1; sigma2 <- 2
R> mu <- list(mu1, mu2); sigma <- list(sigma1, sigma2)
R> rmixnorm1 <- function(n, mu, sigma, omega)
+ {
+ p <- length( mu[[1]] )
+ Y <- label <- rep(0, n)
+  for(i in 1:n)
+  {
+  r_MUL <- rmultinom(n = 1, size = 1, omega)
+  k <- apply(r_MUL, 2, which.max)
+  label[i] <- k
+  Y[i] <- rnorm(n = 1, mean = mu[[k]], sd = sigma[[k]])
+  }
+ ls <- list( "sample" = Y, "label" = label )
+ return(ls)
+ }
R> Y <- rmixnorm1(n, mu, sigma, omega)
R> plot( Y$sample, col = Y$label )
```

```
R> set.seed(20240713)
R> n <- 5000; K <- 2; omega <- c(0.6, 0.4); mu1 <- -3; mu2 <- 3; sigma1 <- sqrt(2); sigma2
    <- 1
R> mu <- list(mu1, mu2); sigma <- list(sigma1, sigma2)
R> rmixnorm2 <- function(n, mu, sigma, omega)
```

```
+ {
+  Y <- rep(0, n)
+  n_k <- rep(0, K)
+  n_k <- floor(n*omega)
+  cn_k <- cumsum(n_k)
+  n_k[K] <- n - sum(n_k[1:(K-1)])
+  Y[ 1: cn_k[1] ] <- rnorm(n = n_k[1], mean = mu[[1]], sd = sigma[[1]])
+   for(k in 2:K) Y[ ( (k - 1)*cn_k[k-1] + 1): cn_k[k] ] <-
+     rnorm(n = n_k[k], mean = mu[[k]], sd = sigma[[k]])
+  ls <- list( "sample" = Y, "label" = rep(1:K, n_k) )
+ return(ls)
+ }
```

2.3.3 Dirichlet distribution

If random vector $\boldsymbol{X} = (X_1, \cdots, X_{\mathrm{K}})^\top$ follows a Dirichlet distribution with parameter vector $\boldsymbol{\rho}$, say $\boldsymbol{X} \sim \mathcal{DIR}(\boldsymbol{\rho})$, then the PDF of $\boldsymbol{X}$ is

$$\mathcal{DIR}(\boldsymbol{x}|\boldsymbol{\rho}) = \frac{\Gamma(\rho_1 + \cdots + \rho_{\mathrm{K}})}{\prod_{k=1}^{\mathrm{K}} \Gamma(\rho_k)} x_1^{\rho_1 - 1} \times x_2^{\rho_2 - 1} \times \cdots \times x_{\mathrm{K}}^{\rho_{\mathrm{K}} - 1}, \tag{2.20}$$

where $0 < x_k < 1$ and $\rho_k > 0$ (for $k = 1, \cdots, \mathrm{K}$), provided that $\sum_{k=1}^{\mathrm{K}} x_k = 1$. If $\mathrm{K} = 2$, then Dirichlet distribution turns into Beta distribution with shape parameters $\rho_1 > 0$ and $\rho_2 > 0$, denoted by $\mathcal{BET}(\rho_1, \rho_2)$, whose PDF is

$$\mathcal{BET}(x|\rho_1, \rho_2) = \frac{\Gamma(\rho_1 + \rho_2)}{\Gamma(\rho_1)\Gamma(\rho_2)} x^{\rho_1 - 1}(1 - x)^{\rho_2 - 1}, \tag{2.21}$$

where $0 < x < 1$. Furthermore, it is worth to note that the ith marginal of random vector $\mathcal{DIR}(\boldsymbol{x}|\boldsymbol{\rho})$ follows $\mathcal{BET}(\rho_i, \sum_{k=1}^{\mathrm{K}} \rho_k - \rho_i)$.

For generating form $\mathcal{BET}(\rho_1, \rho_2)$ in `R`, we can use command `rbeta(n, shape1, shape2, ncp = 0)` in which `n` is sample size, `shape1` $= \rho_1$, and `shape2` $= \rho_2$. Let $X_k \sim \mathcal{G}(\rho_k, \beta)$ (for $k = 1, \cdots, \mathrm{K}$), then random vector $(X_1/Y, \cdots, X_{\mathrm{K}}/Y)^\top$ follows a Dirichlet distribution with parameter vector $\boldsymbol{\rho} = (\rho_1 \cdots, \rho_{\mathrm{K}})^\top$ in which $Y = \sum_{k=1}^{\mathrm{K}} X_k$.

```
R> rdirichlet <- function(n, omega)
+{
+ K <- length(omega); beta <- 1
+ X <- matrix(0, nrow = n, ncol = K)
+ for(k in 1:K) X[, k] <- rgamma(n, shape = omega[k], rate = beta)
+return( X/rowSums(X) )
+}
```

2.3.4 Chi-squared distribution

The random variable X is said to have chi-squared distribution with ν degrees of freedom if its PDF is given by

$$f(x|\nu) = \frac{2^{-\frac{\nu}{2}}}{\Gamma(\nu/2)} x^{\frac{\nu}{2} - 1} \exp\Big\{-\frac{x}{2}\Big\}, \tag{2.22}$$

where $\nu \in \mathbb{N}$ is the family parameter. The chi-squared distribution is a special case of gamma distribution. This means that if $X \sim \mathcal{G}(k/2, 1/2)$, then $X \sim \chi(k)$. For Simulating random variable X following a chi-squared distribution in `R`, we may use command `rchisq(n, df, ncp=0)` in which `n` is the sample size, `df` is degrees of freedom (here is parameter ν), and, by default, the non-centrality parameter `bncp` is set to be zero. In fact, X can be represented as the sum

of ν squared standard Gaussian random variables as $X = \sum_{i=1}^{\nu} Z_i^2$. Hence, as an elementary way, one can simulate a chi-squared random variable X through adding up ν number(s) independent standard Gaussian random variables. Of course, herein, we suggest to use either `rgamma(n, shape = nu/2, rate = 1/2)` or `rchisq(n, df = nu, ncp = 0)`. When the non-centrality parameter is nonzero, then the PDF of a chi-squared distribution can be written as

$$f(x|\nu,\lambda) = \frac{1}{2}\Big(\frac{x}{2}\Big)^{\frac{\nu}{4}-\frac{1}{2}} \exp\Big\{-\frac{x+\lambda}{2}\Big\} I_{\nu/2-1}(\sqrt{\lambda x}), \tag{2.23}$$

where $I_u(\cdot)$ is the modified Bessel function of the first kind that is given by

$$I_i(y) = \Big(\frac{y}{2}\Big)^i \sum_{j=0}^{\infty} \Big(\frac{y^2}{4}\Big)^j \frac{1}{\Gamma(j+1)\Gamma(i+j+1)}. \tag{2.24}$$

Alternatively, the PDF of each chi-squared distribution can be represented as

$$f(x|\nu,\lambda) = \sum_{i=0}^{\infty} \frac{\exp\{-\lambda/2\}}{\Gamma(i+1)} \Big(\frac{\lambda}{2}\Big)^i f(x|\nu+2i). \tag{2.25}$$

As it is seen from (2.25), when $\lambda \neq 0$, then PDF of a chi-squared can be expressed in terms a mixture model whose weights are equal to a Poisson mass function. Therefore, for simulating from a chi-squared distribution with nonzero non-centrality parameter, we adopt the following `R` code.

```
R> rchisq0 <- function(n, nu = 1, lambda = 0)
+ {
+ X <- rep(NA, n)
+ P <- rpois(n, lambda = lambda/2)
+ for(i in 1:n) X[i] <- rchisq(1, df = nu + 2*P[i])
+ return(X)
+ }
```

2.3.5 Inverse Gaussian distribution

For simulating from Inverse Gaussian distribution, denoted as $\mathcal{IG}(\lambda,\mu)$, we follow the method proposed by Michael et al. [1976]. The positive random variable X is said to follow $\mathcal{IG}(\lambda,\mu)$ if its PDF is given by

$$f(x|\boldsymbol{\theta}) = \sqrt{\frac{\lambda}{2\pi x^3}} \exp\Big\{-\frac{\lambda(x-\mu)^2}{2\mu^2 x}\Big\}, \tag{2.26}$$

where $\boldsymbol{\theta} = (\lambda > 0, \mu > 0)^\top$. Comparing by the RHS of (2.48) and (2.26), it is evident that generalized Inverse Gaussian distribution specializes to Inverse Gaussian distribution when $a = -1/2$ and $b = 0$. In words, $\mathcal{IG}(\lambda,\mu) = \mathcal{GIG}(a = -1/2, b = \lambda/\mu^2, c = \lambda)$. For simulating from $\mathcal{IG}(\lambda,\mu)$, first we note that Shuster [1968]:

$$Y = g(X) = \frac{\lambda(X-\mu)^2}{\mu^2 X} \sim \chi_{(1)}, \tag{2.27}$$

where $\chi_{(n)}$ accounts for a chi-square distribution with n degrees of freedom. So, we may think that generating from X is accomplished straightforwardly by drawing a sample from $\chi_{(1)}$ and then X is recovered by a simple inverse transformation. But, care must be taken into the fact that transformation in (2.28) may have two roots and the problem is here to choose among these two distinct roots x_1 and x_2. Overall, based on method of Michael et al. [1976], for a given

y in (2.28) there are k distinct roots $\{x_1, \cdots, x_k\}$ (here, we have $k = 2$). The problem turns into determining the multinomial probabilities associated with each of $x_i s$ (for $i = 1, \cdots, k$). Here, we investigate the case in which X is a continuous random variable. Let $f(\cdot)$ and $F(\cdot)$, respectively, denote the PDF and CDF of random variable X that we are interested in sampling from and $p_i(y)$ is the conditional (or multinomial) probability with which the ith root must be chosen. It is shown that Michael et al. [1976]:

$$p_i(y) = \left[1 + \sum_{i=1, i\neq j}^{k} \left|\frac{g'(x_i)}{g'(x_j)}\right| \times \frac{f(x_j)}{f(x_i)}\right]^{-1}, \tag{2.28}$$

where $g'(x)$ denotes the first derivative of transformation $g(x)$ with respect to x. Computing $p_i(y)$ in (2.28) for $i = 1, \cdots, k$, the ith root is chosen as a generation from X with probability $p_i(y)$. In what follows, we describe this method when $X \sim \mathcal{IG}(\lambda, \mu)$. Evidently, $g(x)$ given by (2.28) has two roots, namely x_1 and x_2. Then more algebra shows $f(x_2)/f(x_1) = (x_1/\mu)^3$ and $g'(x_i)/g'(x_j) = -(\mu/x_1)^2$. Hence, for a given $y \sim \chi_{(1)}$, the roots x_1 and x_2 are

$$x_1 = \mu + \frac{\mu^2 y}{2\lambda} - \frac{\mu}{2\lambda}\sqrt{4\mu\lambda y + \mu^2 y^2},$$

and $x_2 = \mu^2/x_1$ with probabilities

$$p_1(y) = \left[1 + \left|\frac{g'(x_1)}{g'(x_2)}\right| \times \frac{f(x_2)}{f(x_1)}\right]^{-1} = \frac{\mu}{\mu + x_1},$$

and $p_2(y) = 1 - p_1(y) = x_1/(\mu + x_1)$, respectively. We adopt the Algorithm 7 for simulating from $X \sim \mathcal{IG}(\lambda, \mu)$ as follows. The corresponding `R` function `rig(n, lambda, mu)`, based on

Algorithm 7 Simulating from Inverse Gaussian distribution

1: Read parameters λ and μ;
2: Generate u from $\mathcal{U}(0, 1)$;
3: Generate y from $\chi_{(1)}$;
4: Set $x_1 = \mu + \frac{\mu^2 y}{2\lambda} - \frac{\mu}{2\lambda}\sqrt{4\mu\lambda y + \mu^2 y^2}$;
5: **if** $u < \mu/(\mu + x_1)$ **then**
6: $\quad x = x_1$;
7: **else**
8: $\quad x = \mu^2/x_1$;
9: **end**
10: Accept x as a generation form $\mathcal{IG}(\lambda, \mu)$.

Algorithm 7, for generating n realizations from $\mathcal{IG}(\lambda, \mu)$ is given by the following.

```
R> rig <- function(n, lambda, mu)
+ {
+ y <- rchisq(n, df =1)
+ u <- runif(n)
+ x_1 <- mu + mu^2*y/(2*lambda) - mu/(2*lambda)*sqrt(4*mu*lambda*y + mu^2*y^2)
+ x_2 <- mu^2/x_1
+ x <- ifelse( u < mu/(mu + x_1), x_1, x_2 )
+ return(x)
+ }
```

2.3.6 α-Stable distribution

Herein, we briefly describe the family of α-stable distributions. This family of distributions was introduced by Paul Pierre Lévy, a French mathematician, in his work on sums of independent identically distributed random variables around 1924. Almost all of members of this

family either have not closed form for PDF or are only expressed in terms of the special functions [Zolotarev, 1986, Samorodnitsky and Taqqu, 1994]. So, this family of distributions are represented in terms of their characteristic function. The characteristic function of α-stable distribution has different parameterizations (forms). The most commonly used parameterizations for is given as follows [Nolan, 1998].

$$E(\exp\{jtX\}) = \begin{cases} \exp\left\{-|\sigma t|^{\alpha}\left[1 - j\beta\ \text{sign}(t)\tan(\frac{\pi\alpha}{2})\right] + jt\mu\right\}, \text{if } \alpha \neq 1, \\ \exp\left\{-|\sigma t|\Big[1 + j\beta\ \text{sign}(t)\frac{2}{\pi}\log|t|\Big] + jt\mu\right\}, \quad \text{if } \alpha = 1, \end{cases} \tag{2.29}$$

where $j^2 = -1$ and sign(u) is the sign function that takes values -1 and +1 for $u < 0$ and $u \geq 0$, respectively. Other forms for characteristic function of α-stable distributions are A, M, B, C, and E [Zolotarev, 1986]. The characteristic function given in (2.29) is known in the literature as S_1 parameterization that is a slightly different version of A parameterization. We write $X \sim S(\alpha, \beta, \sigma, \mu)$ to denote that random variable X follows an α-stable distribution with parameters α, β, σ, and μ in S_1 parameterization. Here, parameter $\alpha \in (0, 2]$ is the index of stability, which determines the tail thickness of the density function. When $\alpha = 2$, the tail thickness is at its thinnest while smaller values of α would yield thicker tails for the distribution. Similarly, the parameter $\beta \in [-1, 1]$ is the degree of skewness (-1 for totally skewed to the left, zero for symmetrical, and +1 for totally skewed to the right), $\sigma \in \mathbb{R}^+$, is the dispersion (scale) around the location parameter μ and $\mu \in \mathbb{R}$, is the horizontal shift of the density function. It is important to note that the parameters σ and μ, in general, are not synonymous with standard deviation and location parameter, as they are in the special case of Gaussian distribution. The only members of this class with closed form PDF are Lévy ($\alpha = 0.5$ and $\beta = 1$), Cauchy ($\alpha = 1$ and $\beta = 0$) and Gaussian ($\alpha = 2$). If $\beta = 0$, and $\mu = 0$, then the class of zero-location symmetric α-stable (SαS) distribution can be identified with the following characteristic function

$$\phi(t) = \exp\{-\sigma^{\alpha}|t|^{\alpha}\}. \tag{2.30}$$

The known members of SαS distribution with closed from density function are Cauchy ($\alpha = 1$) and Gaussian ($\alpha = 2$). Here, it is noteworthy to mention that the kth moment of each α-stable distribution is finite if $k < \alpha$. This means that variance of non-Gaussian α-stable distribution is not finite and its first moment is finite if $\alpha > 1$. For $\alpha > 1$ then μ becomes the expected value of distribution. For $\alpha < 1$, the support of class $S(\alpha, 1, \sigma, 0)$ only covers the positive semi-axis. For this reason, if $\alpha < 1$, $\beta = 1$, and $\mu = 0$, then the α-stable distribution family is also called positive α-stable distribution. An important member of the positive α-stable family is $S(\alpha/2, 1, [\cos(\pi\alpha/4)]^{2/\alpha}, 0)$ whose distribution function is given by [Chernin and Ibragimov, 1959]:

$$F_P(p|\alpha) = \frac{1}{\pi}\int_0^{\pi} \exp\Big\{-p^{-\frac{\alpha}{2-\alpha}}A(u)\Big\}du, \tag{2.31}$$

where $[A(u)]^{2/(2-\alpha)}$ in the right-hand side of (2.31) is called the Zolotarev's function Devroye [2009], defined for $0 < \alpha < 2$ as

$$A(u) = \frac{\Big\{\sin\big[(\frac{\alpha}{2})u\big]\Big\}^{\frac{\alpha}{2}}\Big\{\sin\Big[(1-\frac{\alpha}{2})u\Big]\Big\}^{1-\frac{\alpha}{2}}}{\sin(u)} \tag{2.32}$$

Let $W \sim \mathcal{EXP}(1)$ and are $U \sim \mathcal{U}(0, \pi)$ independent. It follows from (2.31) that

$$F_P(p|\alpha) = P\Big\{W > p^{-\frac{\alpha}{2-\alpha}}A(U)\Big\}. \tag{2.33}$$

A straightforward method was suggested by Kanter [1975] for simulating $P \sim S(\alpha/2, 1, [\cos(\pi\alpha/4)]^{2/\alpha}, 0)$ as follows.

$$P \overset{d}{=} \Big(\frac{A(U)}{W}\Big)^{\frac{2-\alpha}{\alpha}}. \tag{2.34}$$

Let $B(\alpha) = 1 - |1 - \alpha|$ and

$$a^\alpha = \sin\Big(\frac{\pi B(\alpha)(1+\beta)}{2}\Big)/\sin(\pi B(\alpha)),$$
$$b^\alpha = \sin\Big(\frac{\pi B(\alpha)(1-\beta)}{2}\Big)/\sin(\pi B(\alpha)).$$
(2.35)

Furthermore, suppose $Z \sim S(\alpha, \beta, 1, 0)$ and $P_1, P_2 \sim S(\alpha/2, 1, [\cos(\pi\alpha/4)]^{2/\alpha}, 0)$ are independent. A method for simulating Z in standard case ($\sigma = 1$ and $\mu = 0$) was proposed by Chambers et al. [1976] in B parameterization based on the fact that $Z \stackrel{d}{=} aP_1 + bP_2$. This method was elaborated to generate from Z in commonly used parameterization (2.29) with more detailed information by Weron [1996]. We refer reader to Nolan [2003] for a corrected version of the method for simulating $Z \sim S(\alpha, \beta, 1, 0)$. Let $\theta_0 = \arctan(\beta \tan(\pi\alpha/2))$. We have

$$Z \stackrel{d}{=} \begin{cases} \frac{2}{\pi}\Big[(\frac{\pi}{2} + \beta U_1)\tan(U_1) - \beta \log\Big(\frac{\frac{\pi}{2} W \cos(U_1)}{\frac{\pi}{2} + \beta U_1}\Big)\Big], & \text{if } \alpha = 1, \\ \frac{\sin[\alpha(\theta_0 + U_1)]}{[\cos(\alpha\theta_0)\cos(U_1)]^{1/\alpha}}\Big[\frac{\cos(\alpha\theta_0 + (1-\alpha)U_1)}{W}\Big]^{(1-\alpha)/\alpha}, & \text{if } \alpha \neq 1, \end{cases}$$
(2.36)

where $W \sim \mathcal{EXP}(1)$ is independent of $U_1 \sim \mathcal{U}(-\pi/2, \pi/2)$. For general case ($\sigma \neq 1$ and $\mu \neq 0$), we use the following transformation for generating from $X \sim S(\alpha, \beta, \sigma, \mu)$.

$$X \stackrel{d}{=} \begin{cases} \sigma Z + \mu + \beta\frac{2}{\pi}\sigma \log\sigma, & \text{if } \alpha = 1, \\ \sigma Z + \mu, & \text{if } \alpha \neq 1, \end{cases}$$
(2.37)

where $Z \sim S(\alpha, \beta, 1, 0)$ is defined as (2.36). The class of α-stable distributions possesses several nice properties. For instance, suppose $X_1, X_2, X_3 \sim S(\alpha, \beta, \sigma, \mu)$ are independent. Then, for any given positive constants such as a and b, we have

$$aX_1 + bX_2 \stackrel{d}{=} (a^\alpha + b^\alpha)^{1/\alpha} X_3.$$

Furthermore, if $X_1 \sim S(\alpha, \beta, \sigma, \mu)$ and $X_2 \sim S(\alpha, -\beta, \sigma, \mu)$, then $X_1 \stackrel{d}{=} -X_2$. The latter is called reflection property [Nolan, 2003]. The `R` function `stable(n, alpha, beta, sigma, mu)`, given below, can be used for simulating n realizations of $X \sim S(\alpha, \beta, \sigma, \mu)$ in S_1 parameterization.

```
R<- rstable <- function(n, alpha, beta, sigma, mu)
+{
+ u1 <- pi*( runif(n) - 1/2 )
+ theta0 <- atan( beta*tan(pi*alpha/2) )/alpha
+ w <- -log( runif(n) )
+  if (alpha == 1)
+  {
+  z <- 2/pi*( (pi/2 + beta*u1)*tan(u1) - beta*log( ( pi/2*w*cos(u1) )/(pi/2 + +beta*u1) ) )
+  x <- sigma*z + 2/pi*beta*sigma*log(sigma) + mu
+  }else{
+  z <- sin(alpha*(theta0 + u1))/(cos(alpha*theta0)*cos(u1))^(1/+alpha)*(cos(alpha*theta0 +
     (alpha - 1)*u1)/w)^( (1 - alpha)/alpha )
+  x <- sigma*z + mu
+  }
+return(x)
+}
```

2.3.7 Birnbaum-Saunders distribution

The Birnbaum-Saunders (BS) distribution has attracted much attention in life-testing and survival analysis. One interesting feature of the BS distribution is the role of the central limit theorem for constructing this distribution. Suppose a sequence of periodic loading are applied to a segment of metal, thereby a sequence of cracks with non-zero length in metal is produced. The total time T, until failure is called BS random variable. The CDF of the BS random variable T, is given by

$$F(t|\boldsymbol{\theta}) = \Phi\Big(\frac{1}{\alpha}\Big[\sqrt{\frac{t}{\beta}} - \sqrt{\frac{\beta}{t}}\Big]\Big), \tag{2.38}$$

where $t > 0$, $\Phi(\cdot)$ is the standard Gaussian CDF, and $\boldsymbol{\theta} = (\alpha, \beta)^\top$. Here, $\alpha > 0$ and $\beta > 0$ are the shape and scale parameters of the BS distribution, respectively. The BS distribution has received much attention in a wide range of fields. A review of its applications can be found in Ng et al. [2003], Teimouri [2023] and references therein. Differentiating the CDF given in (2.38), then the PDF of the BS distribution is given by

$$f(t|\boldsymbol{\theta}) = \frac{\sqrt{\frac{t}{\beta}} + \sqrt{\frac{\beta}{t}}}{2\sqrt{2\pi}\alpha t} \exp\Big\{-\frac{1}{2\alpha^2}\Big(\sqrt{\frac{t}{\beta}} - \sqrt{\frac{\beta}{t}}\Big)^2\Big\}. \tag{2.39}$$

We write $T \sim \mathcal{BS}(\alpha, \beta)$ to indicate that random Variable T follows a BS distribution with CDF (2.38). If random variable T follows the BS distribution with PDF given by (2.39) can be represented as a mixture of two GIG distributions as

$$f(t|\boldsymbol{\theta}) = \frac{1}{2}g(t|\boldsymbol{\theta}_1) + \frac{1}{2}g(t|\boldsymbol{\theta}_2), \tag{2.40}$$

where $g(\cdot|\boldsymbol{\theta}_i)$ (for $i = 1, 2$) is the PDF of GIG distribution[Hörmann and Leydold, 2014], $\boldsymbol{\theta}_1 = (1/2, 1/(\alpha^2\beta), \beta/\alpha^2)^\top$, and $\boldsymbol{\theta}_2 = (-1/2, 1/(\alpha^2\beta), \beta/\alpha^2)^\top$. This means that each BS random variable can be simulated through a two-component GIG mixture model with weight (or mixing) vector $\boldsymbol{\omega} = (0.5, 0.5)^\top$. It is not hard to check that

$$X \overset{d}{=} \frac{1}{2}\Big(\sqrt{\frac{T}{\beta}} - \sqrt{\frac{\beta}{T}}\Big), \tag{2.41}$$

where $X \sim \mathcal{N}(0, \alpha^2/4)$ or equivalently $T = \beta[1 + 2X^2 + 2X(1 + X^2)^{1/2}]$. Among distributional properties of BS distribution, we mention that if $T \sim \mathcal{BS}(\alpha, \beta)$, then $1/T \sim \mathcal{BS}(\alpha, 1/\beta)$. The `R` function `rbs(n, alpha, beta)`, given by the following, can be used for simulating n realizations of $T \sim \mathcal{BS}(\alpha, \beta)$.

```
R> rbs <- function(n, alpha, beta)
+{
+ X <- alpha/2*rnorm(n)
+ T <- beta*( 1 + 2*X^2 + 2*X*sqrt( 1 + X^2 ) )
+ return(T)
+}
```

2.4 Rejection sampling

Suppose sampling form PDF $f(\cdot|\theta)$ is not easy but it is dominated by another PDF such as $g(\cdot|\theta)$ so that

$$\sup_{x \in \mathcal{S}_f} \frac{f(x|\theta)}{g(x|\theta)} \leq M,$$

where $M \geq 1$ may depend on θ. A proposal with PDF $g(x|\theta)$ which is as possible as close to $f(x|\theta)$ and easy-to-sample from would be the key objective in each rejection sampling scheme. The following four-step algorithm shows how a rejection sampling scheme works [Devroye, 1986a]. It is proved that average numbers of trials for generating a sample in each rejection sampling scheme is M^{-1}, see Ross [2022]. Figure 2.5(b) shows the generated samples across 1000 iterations.

Algorithm 8 Rejection sampling technique

1: Choose suitable proposal with PDF $g(\cdot|\theta)$ and determine M;
2: Generate $y \sim g(\cdot|\theta)$;
3: Generate $u \sim \mathcal{U}(0,1)$;
4: Squeezing test: If $u < \frac{f(y|\theta)}{M \times g(y|\theta)}$, then accept y as a generation from $f(\cdot|\theta)$, otherwise return to step 2 and repeat algorithm.

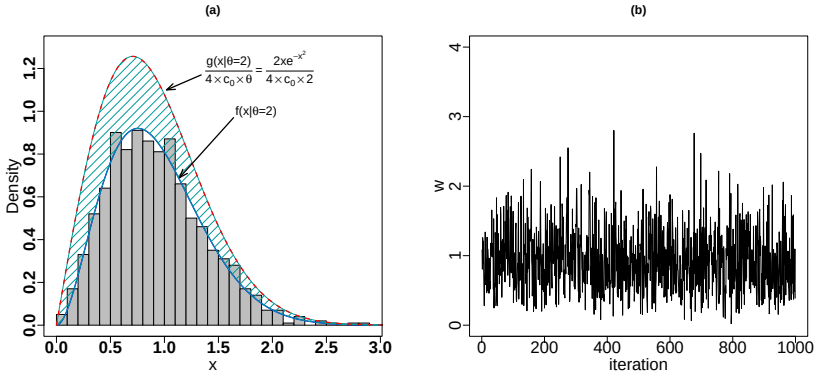


Figure 2.5: (a): Histogram constructed based on 1000 samples generated from $\mathcal{W}(\theta = 2, 1)$. Superimposed are $Mg(x|\theta)$ (red line) and $\mathcal{W}(x|\theta = 2, 1)$ (blue line). (b): Generated samples across iterations for sampling from $f(x|\theta = 2) = \mathcal{W}(x|\theta = 2, 1)$.

Example 2-4

Suppose $f(x|\theta)$ is defined as in Example 2-1. Herein, we can rewrite $f(x|\theta) = (x+1)^{-2}x^{\theta}\exp\{-x^{\theta}\}/c_0$ where c_0 is normalizing constant guarantees $\int_{\mathcal{S}_f} f(x|\theta)dx = 1$. Considering $g(x|\theta) = \mathcal{W}(x|\theta, 1)$ as the proposal, it is easy to check that

$$\frac{f(x|\theta)}{g(x|\theta)} = \frac{x}{c_0\theta(x+1)^2}. \tag{2.42}$$

Differentiating the RHS of (2.42) and setting the resultant to 0 reveals that the maximal value of RHS of (2.42) is $1/(4c_0\theta)$ that is obtained for $x = 1$. That is

$$\frac{f(x|\theta)}{g(x|\theta)} \leq \frac{1}{4c_0\theta} = M.$$

It follows that

$$\frac{f(x|\theta)}{M \times g(x|\theta)} = \frac{4x}{(x+1)^2}.$$

Based on Algorithm 8, the rejection sampling scheme for generating realization from $f(\cdot|\theta = 2)$ is described as follows.

1. Generate $y \sim \mathcal{W}(\theta = 2, 1)$;

2. Generate $u \sim \mathcal{U}(0,1)$;

3. If $u < 4y/(y+1)^2$, then accept y as a generation from $f(\cdot|\theta = 2)$; otherwise go to step 1 and repeat algorithm.

■

In Example 2-4 above, the usage time study for generating $n = 1000$ realizations is around 0.01 second. The pertaining `R` code, for producing Figure 2.5(b), is given as follows.

```
R > set.seed(20240519)
R > n <- 10000 # size of generations
R > theta <- 2
R > w <- rep(0, n) # vector of generated realizations
R > j <- 1
R > while(j <= n)
+ {
+  y <- rweibull(1, shape = theta, scale = 1)
+  u <- runif(1)
+  if( u < 4*y/(y+1)^2 )
+  {
+   w[j] <- y # y is accepted
+   j <- j + 1
+  }
+ }
R > plot(w)
```

Example 2-5

Let random variable X follow a distribution with PDF given by Azzalini [1985]:

$$f(x|\mu,\sigma,\lambda) = \frac{2}{\sigma}\phi\Big(\frac{x-\mu}{\sigma}\Big)\Phi\Big(\lambda\frac{x-\mu}{\sigma}\Big), \tag{2.43}$$

where $\phi(\cdot)$ and $\Phi(\cdot)$ represent accordingly the PDF and CDF of the standard Gaussian distribution. Herein, $\lambda \in \mathbb{R}$ is the skewness parameter (the cases $\lambda \to -\infty$, $\lambda = 0$, and $\lambda \to +\infty$ yield totally skewed to the left, symmetric (ordinary Gaussian), and totally skewed to the right distributions, respectively), $\mu \in \mathbb{R}$ is the location parameter, and $\sigma \in \mathbb{R}^+$ denotes the scale parameter. Hereafter, we write $X \sim \mathcal{SG}(\mu,\sigma,\lambda)$ to denote that X follows the distribution with PDF given by (2.43). Without loss of generality, let $\mu = 0$ and $\sigma = 1$. A direct method proposed by Henze [1986] for simulating from $\mathcal{SG}(0,1,\lambda)$ as follows. Let Z_1 and Z_2 denote two independent copies of the standard Gaussian random variate. Then, for random variable $X \sim \mathcal{SG}(0,1,\lambda)$ we can write

$$X \overset{d}{=} \frac{\lambda|Z_1| + Z_2}{\sqrt{1+\lambda^2}}. \tag{2.44}$$

For simulating from $\mathcal{SG}(0,1,\lambda)$ using the rejection method, let $f(x|0,1,\lambda = 4)$ to be the PDF of $\mathcal{SG}(x|0,1,\lambda = 4)$. We proceed by considering the fact that

$$f(x|0,1,\lambda) \le 2g(x|\theta) = 2\phi(x). \tag{2.45}$$

Hence, simulating from $\mathcal{SG}(0,1,\lambda)$, through the rejection sampling, is possible by considering the proposal PDF to be $\phi(x)$ and $M = 2$. Figure 2.6(a) shows histogram of generated realizations as well as the superimposed target and proposal PDFs for $\lambda = 4$. The sampler's motion across 500 iterations are displayed in Figure 2.5(b). ■

The pertaining `R` code, for producing Figure 2.6(b) is given as follows.

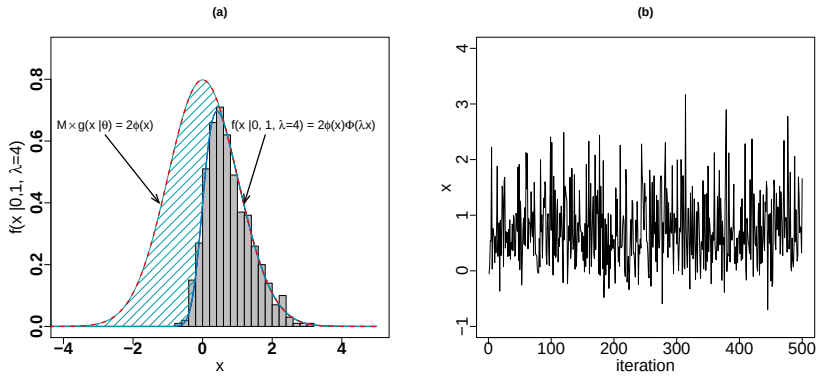


Figure 2.6: (a): Histogram constructed based on 500 samples generated from $\mathcal{SG}(0, 1, \lambda = 4)$. Superimposed are $Mg(x|\theta) = 2\phi(x)$ (red line) and $f(x|0, 1, \lambda = 4)$ (blue line). (b): Generations across iterations for sampling from $f(x|0, 1, \lambda = 4) \sim \mathcal{SG}(0, 1, \lambda = 4)$.

```
R> set.seed(20240225)
R> n <- 500
R> i <- 1; x <- rep(0, n); lambda <- 4
R> while(i <= n)
+ {
+ z <- rnorm(1);
+  if( runif(1) < pnorm(lambda*z) )
+  {
+  x[i] <- z; i <- i + 1
+  }
+ }
R > plot(x)
```

2.5 Adaptive rejection sampling

Suppose $f(x|\theta)$ is a log-concave PDF over $\mathcal{S}_f$ that equivalently means $\mathcal{H}(x) = \log f(x|\theta)$ is a concave function of x. A method was proposed by Devroye [1986b] for generating realization form log-concave distribution provided that the mode (a point at which derivative of $f(x|\theta)$ with respect to x is zero) of $f(x|\theta)$ to be known. More precisely, the proposed method is a rejection sampling scheme using a three-piece exponential proposal in which the center piece touches the target PDF $f(x|\theta)$ at its mode. While this method is not adaptive, the adaptive rejection sampling (ARS) scheme [Gilks and Wild, 1992] develops this method by improving the proposal distribution in each iteration such that the updated proposal proposes generation that passes the *squeezing* test with higher probability than the previous iteration. In each ARS scheme, $\mathcal{H}(x)$ can be bounded by piecewise linear upper hull in which each tangent line is defined as

$$U_k(x) = (x - a)\mathcal{H}'(a) + \mathcal{H}(a), \tag{2.46}$$

for $x \in [z_{i-1}, z_i]$ and $i = 1, \cdots, k$ where

$$z_i = \frac{\mathcal{H}(x_{i+1}) - \mathcal{H}(x_i) - x_{i+1}\mathcal{H}'(x_{i+1}) + x_i\mathcal{H}'(x_i)}{\mathcal{H}'(x_i) - \mathcal{H}'(x_{i+1})}.$$

We note that the tangent lines are constructed at abscissa (points) $T_k = \{x_1 \leq x_2 \leq \cdots \leq x_k \in \mathcal{S}_f\}$ and intersect at z_i where z_0 and z_k are the lower ($-\infty$ if $\mathcal{S}_f$ is not bounded from below) and the upper (∞ if $\mathcal{S}_f$ is not bounded from above) bounds of $\mathcal{S}_f$, accordingly. Likewise, we

can define a piecewise linear lower hull constructed based on chords between adjacent abscissae in T_k as

$$L_k(x) = \frac{\mathcal{H}(x_{i+1} - x)\mathcal{H}(x_i) - \mathcal{H}(x - x_i)\mathcal{H}(x_{i+1})}{x_{i+1}x_i}, \tag{2.47}$$

where $x \in [x_i, x_{i+1}]$ for $i = 1, \cdots, k-1$. If $x < x_1$(or $x > x_k$), then we set $L_k(x) = -\infty$. Notice that log-concavity of $f(x|\theta)$ ensures that $L_k(x) \leq \mathcal{H}(x) \leq U_k(x)$ for $x \in \mathcal{S}_f$. Each ARS algorithm proceeds three *initialization*, *sampling*, and *updating* steps for generating n realizations from PDF $f(x|\theta)$ given by the following.

- *initialization step*: Construct the abscissae in T_k. If $\mathcal{S}_f$ is left-unbounded, then set x_1 such that $\mathcal{H}'(x_1) < 0$ and if $\mathcal{S}_f$ is right-unbounded, then set x_k such that $\mathcal{H}'(x_k) < 0$. Furthermore, compute $L_k(x)$, $U_k(x)$, and $S_k(x)$ where

  $$s_k(x) = \frac{\exp\{U_k(x)\}}{\int \exp\{U_k(y)\}dy},$$

  and functions $L_k(x)$ and $U_k(x)$ as defined in (2.47) and (2.46), accordingly.

- *sampling step*: Generate x^* from $s_k(x)$ and $u \sim \text{Unif}(0, 1)$ independently; and then perform squeezing test that accepts x^* if $w \leq \exp\{L_k(x^*) - U_k(x^*)\}$; otherwise compute the rejection test that accepts x^* if $w \leq \exp\{\mathcal{H}(x^*) - U_k(x^*)\}$; otherwise reject x^*.

- *updating step*: If evaluating $\mathcal{H}(x^*)$ and $\mathcal{H}'(x^*)$ is necessary, then add x^* to T_k in order to form updated T_k as $T_k^* = \{\{x_1 \leq x_2 \leq \cdots \leq x_k\} \cup x^* \in \mathcal{S}_f\}$. Return to the *initialization* step while the set of $k+1$ abscissae in T_k^* is relabeled in ascending order and repeat these three steps if n realizations have been not yet obtained. We note that for initialization, a set of two starting abscissae ($k \geq 2$) is necessary.

We note that a given PDF $f(.|\theta)$ is unimodal if and only if $f(.|\theta)$ is log-concave. So, for simulating from a distribution with unimodal PDF $f(.|\theta)$, the ARS algorithm would be a suitable candidate provided that $\mathcal{H}(.)$ has closed form.

Example 2-6

Let $f(x|\boldsymbol{\theta})$ represent the PDF of generalized inverse Gaussian, denoted as $\mathcal{GIG}(\psi, \chi, \lambda)$, distribution defined as [Johnson et al., 1995, Jorgensen, 2012]:

$$f(x|\boldsymbol{\theta}) = \Big(\frac{\psi}{\chi}\Big)^{\frac{\lambda}{2}} \frac{x^{\lambda-1}}{2\mathcal{K}_\lambda(\sqrt{\psi\chi})} \exp\Big\{-\frac{\psi x}{2} - \frac{\chi}{2x}\Big\}, \tag{2.48}$$

where $\boldsymbol{\theta} = (\psi, \chi, \lambda)^\top$. It is easy to check that $f(x|\boldsymbol{\theta})$ is log-concave for $\lambda \geq 1$. Implementing the ARS algorithm is executable through the computer code. The package `ars` developed for this purpose that is available at `https://CRAN.R-project.org/package=ars`. ■

2.5.1 Two-dimensional single rejection sampling

Let $f(x|\theta)$, that my be known only up to a proportionality constant z, is an unimodal PDF of random variable $X \in \mathbb{R}$ from which sampling is of interest. If it is possible to consider an extra auxiliary variable Y, thereby the marginal PDF o X can be marginalized by integrating out Y as

$$f(x|\theta) = \int f(x, y|\theta)dy,$$

then one can use the *two-dimensional rejection sampling* scheme in order to generate sample x with PDF $f(x|\theta)$. For this purpose, let $l(x, y|\theta)$ denote the joint PDF of the proposal (candidate) such that

$$\sup_{(x,y)\in\mathcal{S}_{f(x,y|\theta)}} \frac{f(x, y|\theta)}{l(x, y|\theta)} = c.$$

We use the following algorithm for simulating from $f(x|\theta)$ through the *two-dimensional single rejection* sampling scheme.

1. Generate pair (X, Y) or the joint PDF $l(x, y\theta)$;
2. Generate $u \sim \text{Unif}(0, 1)$;
3. If $u < f(x, y|\theta)/[c\times l(x, y|\theta)]$, then accept X as a generation from $f(x|\theta)$; otherwise return to step 1 and repeat the algorithm.

The efficiency of a *two-dimensional single rejection* sampling scheme depends on quantity c and the complexity of sampling from two-dimensional candidate $l(x, y|\theta)$. If $l(x, y|\theta)$ is easy to sampling from and c is as small as possible, then this algorithm works efficiently.

Example 2-7

Let X denotes a random variable following an exponentially tilted α-stable distribution with tilting parameter $\lambda > 0$ if it follows a distribution whose PDF is given by

$$f(x|\boldsymbol{\theta}) = \exp\{-x\lambda + \lambda^{\frac{\alpha}{2}}\}d(x|\alpha), \tag{2.49}$$

where $\boldsymbol{\theta} = (\alpha, \lambda)^\top$ for $0 < \alpha \leq 2$ is the family parameter vector and $d(x|\alpha)$ is the PDF of a positive α-stable distribution, see Devroye [2009]. The PDF $d(x|\alpha)$ has not closed form, but can be represented as

$$d(x|\alpha) = \frac{1}{\pi}\int_0^\pi \exp\Big\{-x^{-\frac{\alpha}{2-\alpha}}A(y)\Big\}dy, \tag{2.50}$$

where

$$A(y) = \frac{\Big\{\sin\big[(\frac{\alpha}{2})y\big]\Big\}^{\frac{\alpha}{2}}\Big\{\sin\Big[(1-\frac{\alpha}{2})y\Big]\Big\}^{1-\frac{\alpha}{2}}}{\sin(y)}.$$

We note that $A(y)^{2/(2-\alpha)}$ is called the Zolotarev's function [Devroye, 2009]. From (2.49) and (2.50), it can be easily seen that $f(x|\boldsymbol{\theta})$ is the marginal of a joint PDF. Herein, we do not aim to discuss this example in details and just mention that for simulating from $f(x, y|\boldsymbol{\theta})$, a bivariate distribution with PDF

$$l(x, y|\boldsymbol{\theta}) = \frac{1}{\pi} \times \frac{x^{m-1}}{\Gamma(m)}\exp\{-x\}, \tag{2.51}$$

is chosen as candidate. Depending on values of α and λ, the gamma PDF with shape parameter $m = \alpha\lambda^\alpha$ in RHS of (2.51) can be replaced with a gamma PDF with shape parameter $m = (1-\alpha)\lambda^\alpha + 1$ or truncated Gaussian PDF. The reader is referred to [Hofert, 2011, Qu et al., 2021] for a thorough treatment of this example. For simulating from PDF (2.49), there is an open source code called `copula` has been developed for `R` language that is available at `https://CRAN.R-project.org/package=copula`. ■

Example 2-8

Let $\mathcal{N}_p(\boldsymbol{\mu}, \Sigma)$ denote a p-dimensional random vector $\boldsymbol{X} = (X_1, \cdots, X_p)^\top$ following a Gaussian distribution with mean vector $\boldsymbol{\mu}$ and covariance matrix Σ whose PDF and CDF are shown by $\boldsymbol{\phi}_p(\cdot|\boldsymbol{\mu}, \Sigma)$ (relation 2.64) and $\boldsymbol{\Phi}_p(\cdot|\boldsymbol{\mu}, \Sigma)$, respectively. Furthermore, write $\mathcal{N}_p(\boldsymbol{\mu}, \Sigma, \mathcal{R}^p)$ to denote a p-dimensional random vector $\boldsymbol{X} = (X_1, \cdots, X_p)^\top$ following a Gaussian distribution truncated on positive Lebesgue measure region $\mathcal{R}^p$ denoted as

$$\mathcal{R}^p = \{\boldsymbol{x} \in \mathbb{R}^p | \boldsymbol{a} \leq A\boldsymbol{x} \leq \boldsymbol{b}\}, \tag{2.52}$$

where $\boldsymbol{a} = (a_1, \cdots, a_p)^\top$ and $\boldsymbol{b} = (b_1, \cdots, b_p)^\top$ are vectors of finite or infinite constants. Moreover, matrix A is a $p \times p$ full rank matrix that constructs a set of p independent linear constraints. As pointed out by [Geweke, 1991] the marginals of $\boldsymbol{X}$ with distribution given by (2.52) are not truncated Gaussian, but the full conditionals, that is distribution of X_i (for $i = 1, \cdots, p$) *conditional* on all of other members of $\boldsymbol{X}$ are truncated Gaussian. To verify this claim, we shall confine ourselves here to suppose $p = 2$ and $A = \boldsymbol{I}_p$ in which $\boldsymbol{I}_2$ accounts for a 2×2 identity matrix. Therefore (2.52) can be rewritten as

$$\mathcal{N}_2(\boldsymbol{\mu}, \Sigma, \mathcal{R}^2), \tag{2.53}$$

where $\mathcal{R}^2 = \{\boldsymbol{x} = (x_1, x_2)^\top \in \mathbb{R}^2 | a_1 \leq x_1 \leq b_1, a_2 \leq x_2 \leq b_2\}$, $\boldsymbol{\mu} = (\mu_1, \mu_2)^\top$, and $\Sigma = [(\sigma_{11}, \sigma_{21})^\top, (\sigma_{12}, \sigma_{22})^\top]$. In univariate case, we may write $\mathcal{N}(\mu, \sigma, \mathcal{R})$ to denote the family of univariate Gaussian distributions with mean μ and variance σ truncated on interval $\mathcal{R} = \{x \in \mathbb{R} | a \leq x \leq b\}$. For computing the marginal PDF of (3.270), e.g. PDF of X_1, we have

$$\begin{aligned} f_{X_1}(x_1|\boldsymbol{\mu}, \Sigma) =& \frac{1}{\mathcal{P}_G} \int_{a_2}^{b_2} \boldsymbol{\phi}_2(\boldsymbol{x}|\boldsymbol{\mu}, \Sigma) dx_2, \\ =& \frac{1}{\mathcal{P}_G} \phi(x_1|\mu_1, \sigma_{11}) \int_{a_2}^{b_2} \phi(x_2|\mu_{2.1}, \sigma_{2.1}) dx_2, \\ =& \frac{1}{\mathcal{P}_G} \phi(x_1|\mu_1, \sigma_{11}) \Big[\Phi(b_2|\mu_{2.1}, \sigma_{2.1}) - \Phi(a_2|\mu_{2.1}, \sigma_{2.1})\Big], \end{aligned} \tag{2.54}$$

where $\mathcal{P}_G = \boldsymbol{\Phi}_2(\boldsymbol{b}|\boldsymbol{\mu}, \Sigma) - \boldsymbol{\Phi}_2(\boldsymbol{a}|\boldsymbol{\mu}, \Sigma)$, $x_1 \in \mathcal{R}_1 = \{x_1 \in \mathbb{R} | a_1 \leq x_1 \leq b_1\}$, and

$$\mu_{2.1} = \mu_2 + \sigma_{21}\sigma_{11}^{-1}(x_1 - \mu_1), \tag{2.55}$$

$$\sigma_{2.1} = \sigma_{22} - \sigma_{21}\sigma_{11}^{-1}\sigma_{12}. \tag{2.56}$$

Obviously, since both terms within the square bracket in the RHS of (3.243) depend on x_1 through $\mu_{2.1}$ given by (2.55), hence $f_{X_1}(x_1|\boldsymbol{\mu}, \Sigma)$ is no longer the PDF of a truncated Gaussian distribution unless $\mu_{2.1}$ is independent of x_1 that occurs only when $\sigma_{12} = \sigma_{21} = 0$. This is while both of the full conditionals $X_1|X_2$ and $X_2|X_1$ correspond to the joint PDF in (3.270) are Gaussian. For example $X_2|X_1 \sim \mathcal{N}(\mu_{2.1}, \sigma_{2.1}, \mathcal{R}_2)$ where $\mathcal{R}_2 = \{x_2 \in \mathbb{R} | a_2 \leq x_2 \leq b_2\}$. Herein, we are willing to apply the two-dimensional single rejection scheme for sampling from distribution (3.270) whose PDF may be rewritten as

$$\begin{aligned} f_{X_1,X_2}(x_1, x_2|\boldsymbol{\mu}, \Sigma) =& \frac{1}{\mathcal{P}_G} \phi\Big(\frac{x_1 - \mu_1}{\sqrt{\sigma_{11}}}\Big) \phi\Big(\frac{x_2 - \mu_{2.1}}{\sqrt{\sigma_{2.1}}}\Big) \times \mathbb{I}_{\mathcal{R}^2}(x_1, x_2) \\ =& c_1 \phi\Big(\frac{x_1 - \mu_1}{\sqrt{\sigma_{11}}}\Big) \phi\Big(\frac{x_2 - \mu_{2.1}}{\sqrt{\sigma_{2.1}}}\Big) \times \mathbb{I}_{\mathcal{R}^2}(x_1, x_2), \end{aligned}$$

where c_1 is a constant independent of $(x_1, x_2)^\top$ and $\mu_{2.1}$ and $\sigma_{2.1}$ are defined in (2.55) and (2.56), respectively. Furthermore, $\mathbb{I}_{\mathcal{R}^2}(x, y)$ is the well-known indicator function defined for the set $\mathcal{R}^2$ defined as

$$\mathbb{I}_{\mathcal{R}^2}(x_1, x_2) = \begin{cases} 1, & \text{if } (x_1, x_2) \in \mathcal{R}^2, \\ 0, & \text{if } (x_1, x_2) \notin \mathcal{R}^2. \end{cases} \tag{2.57}$$

We may consider the distribution of proposal as the product of $\mathcal{N}_{R_1}(\mu_1, \sigma_{11}, \mathcal{R}_1)$ and $\mathcal{U}(a_2, b_2)$ with PDF

$$\begin{aligned} l_{X_1,X_2}(x_1, x_2|\boldsymbol{\mu}, \Sigma) =& \frac{\phi\big(\frac{x_1-\mu_1}{\sqrt{\sigma_{11}}}\big) \times \mathbb{I}_{\mathcal{R}_1}(x_1)}{\Phi(b_1|\mu_1, \sqrt{\sigma_{11}}) - \Phi(a_1|\mu_1, \sqrt{\sigma_{11}})} \times \frac{\mathbb{I}_{\mathcal{R}_2}(x_2)}{b_2 - a_2}, \\ =& c_2 \times \phi\Big(\frac{x_1 - \mu_1}{\sqrt{\sigma_{11}}}\Big) \times \mathbb{I}_{\mathcal{R}_1}(x_1) \times \mathbb{I}_{\mathcal{R}_2}(x_2), \end{aligned}$$

where c_2 is independent of $(x_1, x_2)^\top$. It follows that

$$\begin{aligned} f_{X_1,X_2}(x_1, x_2|\boldsymbol{\mu}, \Sigma) &\leq \frac{c_1}{\sqrt{2\pi\sigma_{2.1}}}\phi\Big(\frac{x_1 - \mu_1}{\sqrt{\sigma_{11}}}\Big) \times \mathbb{I}_{\mathcal{R}^2}(x_1, x_2) \\ &\leq c_1 \frac{b_2 - a_2}{\sqrt{2\pi \times \sigma_{2.1}}} l_{X_1,X_2}(x_1, x_2|\boldsymbol{\mu}, \Sigma) \\ &= c \times l_{X_1,X_2}(x_1, x_2|\boldsymbol{\mu}, \Sigma). \end{aligned}$$

Hence,

$$\frac{f_{X_1,X_2}(x_1, x_2|\boldsymbol{\mu}, \Sigma)}{c \times l_{X_1,X_2}(x_1, x_2|\boldsymbol{\mu}, \Sigma)} = \exp\Big\{\frac{-(x_2 - \mu_{2.1})^2}{2\sigma_{2.1}}\Big\}.$$

In what follows Algorithm 9 describes how to sample from from truncated bivariate Gaussian distribution. In general where A is not necessarily an identical full rank matrix, we suggest the

Algorithm 9 Two-dimensional single rejection scheme for sampling from truncated bivariate Gaussian distribution

1: Read $\boldsymbol{a}$, $\boldsymbol{b}$, $\boldsymbol{\mu}$, and Σ given by (3.270);
2: Generate $u \sim \mathcal{U}(0, 1)$;
3: Generate $x_1 \sim \mathcal{TN}_{R_1}(\mu_1, \sigma_{11})$;
4: Generate $x_2 \sim \mathcal{U}(a_2, b_2)$;
5: Squeezing test: If $-2\sigma_{2.1} \log u > (x_2 - \mu_{2.1})^2$ where $\mu_{2.1}$ and $\sigma_{2.1}$ are given by (2.55) and (2.56), respectively, then accept $(x_1, x_2)^\top$ as a generation from PDF given by (3.270), otherwise return to step 2 and repeat algorithm;

use of method proposed by Geweke [1991] for simulating from truncated bivariate Gaussian distribution that will be discussed in Section 5.7. Here we give a brief discussion of this method. For simulating form PDF given by (2.52), first one needs to simulate from $\boldsymbol{Z} \sim \mathcal{N}_{\boldsymbol{T}}(\boldsymbol{0}, A\Sigma A^\top, \mathcal{R}^p)$ where $\mathcal{R}^p = \{\boldsymbol{z} \in \mathbb{R}^p | \boldsymbol{a} - A\boldsymbol{\mu} \leq \boldsymbol{z} \leq \boldsymbol{b} - A\boldsymbol{\mu}\}$, and then accept $\boldsymbol{X} = \boldsymbol{\mu} + A^{-1}\boldsymbol{Z}$ as a generation from PDF given by (2.52). The `R` function below is provided for generating n sample from truncated bivariate Gaussian distribution through two-dimensional single rejection scheme.

```
R> rtbinorm_rejection <- function(n, Mu, Sigma, a, b, A)
+ {
+ V <- A%*%Sigma%*%t(A)
+ T1 <- matrix( cbind(a - A%*%Mu, b - A%*%Mu), nrow = 2, ncol = 2)
+ X <- Z <- matrix(0, nrow = n, ncol = 2)
+ sigma2.1 <- V[2, 2] - V[1, 2]^2/V[1, 1]
+ sigma1 <- sqrt( V[1, 1])
+ j <- 1
+  while(j < n)
+  {
+   u <- runif(3)
+   cdf.a <- pnorm(T1[1, 1], 0, sigma1)
+   cdf.b <- pnorm(T1[1, 2], 0, sigma1)
+   x <- qnorm( u[1]*(cdf.b - cdf.a) + cdf.a,
```

```
+      mean = 0, sd = sigma1 )
+   mu2.1 <- V[1, 2]/V[1, 1]*x
+   y <- (T1[2, 2] - T1[2, 1])*u[2] + T1[2, 1]
+    if( -2*log(u[3]) > (y - mu2.1)^2/sigma2.1 )
+    {
+     Z[j, ] <- c(x, y)
+     j <- j + 1
+    }
+  }
+ X <- matrix(Mu, nrow = n, ncol = 2, byrow = T) + Z%*%solve(A)
+ return(X)
+ }
```

2.6 Ratio of uniforms method

The Ratio of uniforms method is in fact a rejection method that introduced by Kinderman and Monahan [1977]. Suppose we are interested in generating random variable X with PDF is $f(x)$ that may be known only up to a proportionality constant $k > 0$, that is $f(x) = k \times h(x)$ [2]. If random vector $(U, V)^\top$ is distributed uniformly on $\mathrm{C}_h = \{(u, v) \in \mathbb{R}^2 \mid 0 \leq u \leq \sqrt{h(v/u)}\}$ with PDF $g(u, v)$ given by

$$g(u,v) = \begin{cases} 2k, & \text{if } 0 \leq u \leq \sqrt{h(v/u)}, \\ 0, & otherwise. \end{cases} \tag{2.58}$$

Consider two transformations $X = V/U$ and $Y = U$, it can be shown that the distribution of random X follows a distribution with PDF $f(x)$. To show this claim, we notice that Jacobian of inverted transformations, that is $U = Y$ and $V = XY$, is $-Y$. Therefore

$$f(x,y) = |-y| \times g(v/u, u) = y \times 2k, \quad 0 < y < \sqrt{h(x)}.$$

An expression for the marginal PDF of X is obtained by integrating out y as

$$\int_0^{\sqrt{h(x)}} f(x,y)dy = \int_0^{\sqrt{h(x)}} 2kydy = kh(x) = f(x).$$

For generating X, first we generate two realizations from $\mathcal{U}(0, a)$ and $\mathcal{U}(b, c)$ in which

$$a = \sup_x \sqrt{h(x)}, \quad b = -\sup_x x\sqrt{h(x)} = \inf_x x\sqrt{h(x)}, \quad c = \sup_x x\sqrt{h(x)}.$$

The support of random vector $(U, V)^\top$, that is $\{(u, v) | u \in (0, a) \text{ and } v \in (b, c)\}$, is a bounding rectangle that encloses the region C_h. If simulated vector $(U, V)^\top$ lies inside region C_h, then we accept V/U as a generation of X. Hence, if $h(x)$ and $x^2h(x)$ are finite, then ratio of uniforms method proceeds as follows for simulating X with PDF $f(x)$.

i. Generate $U \sim \mathcal{U}(0, a)$ and $V \sim \mathcal{U}(b, c)$ independently;

ii. If $U^2 < h(V/U)$, then accept $X = V/U$ from PDF $f(x)$, otherwise go back step (i).

Evidently, the ratio of uniforms method is a rejection method. The acceptance rate is

$$\int h(x)dx \times [2a(c-b)]^{-1}.$$

[2] The function $h(x)$ sometimes is called quasi PDF.

As pointed out by [Wakefield et al., 1991, Theorem 2], for all location families of distributions that are unimodal, the acceptance probability for the ratio of uniforms method maximizes when the location parameter is zero. Therefore, for example, to generate $X \sim \mathcal{N}(\mu, \sigma^2)$, first we simulate from $Y = \mathcal{N}(0, \sigma^2)$ and then we have $Y + \mu \sim \mathcal{N}(\mu, \sigma^2)$.

Example 2-9

Suppose we are interested in simulating from $X \sim \mathcal{N}(0, 1)$ through the ratio of uniforms method. Let $h(x) = \exp\{-x^2/2\}$, we can see that

$$\begin{aligned} \mathrm{C}_h &= \left\{(u, v) \in \mathbb{R}^2 \;\middle|\; 0 \le u \le \exp\{-1/2 \times v^2/u^2\}\right\} \\ &= \left\{(u, v) \in \mathbb{R}^2 \;\middle|\; v^2 \le -4u^2 \log u\right\}, \end{aligned}$$

and

$$a = \sup_x \sqrt{h(x)} = 1, \quad b = -\sup_x x\sqrt{h(x)} = -\sqrt{2e}, \quad c = \sup_x x\sqrt{h(x)} = \sqrt{2e},$$

where $e = \exp\{1\} = 2.718281\ldots$. We follow a two-step procedure given below for simulating X.

i. Generate $U \sim \mathcal{U}(0, 1)$ and $V \sim \mathcal{U}(-\sqrt{2/e}, \sqrt{2/e})$ independently;

ii. If $U^2 < \exp\{-1/2 \times V^2/U^2\}(V/U)$, then accept $X = V/U$ from PDF standard Gaussian distribution, otherwise go back to step (i).

The acceptance rate for this method is

$$\int h(x)dx \times [2a(c-b)]^{-1} = \frac{\sqrt{2\pi}}{4\sqrt{2\exp\{-1\}}} \approx 0.731.$$

The pertaining `R` function called `rnorm_ru`, for generating n realizations from $X \sim \mathcal{N}(0, 1)$ based on the ratio of uniforms method is given as follows.

```
R> rnorm_ru <- function(n)
+{
+ X <- rep(NA, n)
+ j <- 1
+ while( j <= n )
+ {
+  U <- runif(1)
+  V <- (2*runif(1) - 1)*sqrt( 2*exp(-1) )
+  if( V^2 < -4*U^2*log( U ) )
+  {
+   X[j] <- V/U
+   j <- j + 1
+  }
+ }
+return(X)
+}
```

■

Example 2-10

Let $X \sim \mathcal{G}(\alpha, \beta)$ with $\alpha > 1$. We know that $X \overset{d}{=} G/\beta$ in which $G \sim \mathcal{G}(\alpha, 1)$, hence without loss of generality, for simulating X through the ratio of uniforms method, we assume that $\beta = 1$. Once we have simulated G, the quantity X is simply produced as $X = G/\beta$. Setting $h(x) = x^{\alpha-1}\exp\{-x\}$, we have

$$\begin{aligned} \mathrm{C}_h &= \Big\{(u,v) \in \mathbb{R}^2 \;\Big|\; u < \sqrt{(v/u)^{\alpha-1}\exp\{-v/u\}}\Big\} \\ &= \Big\{(u,v) \in \mathbb{R}^2 \;\Big|\; 2\log u < (\alpha-1)\log(v/u) - v/u\Big\}, \end{aligned}$$

and

$$a = \sup_x \sqrt{h(x)} = \Big(\frac{\alpha-1}{e}\Big)^{(\alpha-1)/2}, \quad b = -\sup_x x\sqrt{h(x)} = 0, \quad c = \sup_x x\sqrt{h(x)} = \Big(\frac{\alpha+1}{e}\Big)^{(\alpha+1)/2},$$

where $e = \exp\{1\} = 2.718281\ldots$. We follow a three-step procedure given below for simulating X.

i. Generate $U \sim \mathcal{U}(0,1)$ and $V \sim \mathcal{U}(-\sqrt{2/e}, \sqrt{2/e})$ independently;

ii. If $U^2 < (V/U)^{\alpha-1}\exp\{-V/U\}$, then go next step, otherwise go back to step (i).

iii. Accept $X = G/\beta$ as a generation from $\mathcal{G}(\alpha, \beta)$.

```
R> rgamma_ru <- function(n, alpha, beta = 1)
+{
+ if (alpha <= 1) stop( "shape parameter must be greater than one." )
+ a <- ( (alpha - 1)*exp(-1) )^( (alpha - 1)/2 )
+ b <- 0
+ c0 <- ( (alpha + 1)*exp(-1) )^( (alpha + 1)/2 )
+ X <- rep(NA, n)
+ j <- 1
+ while( j <= n )
+ {
+  U <- runif(1, 0, a); V <- runif(1, 0, c0)
+  upper <- 1/2*( (alpha - 1)*log(V/U) - V/U )
+  if( log(U) < upper )
+  {
+   X[j] <- 1/beta*V/U
+   j <- j + 1
+  }
+ }
+return(X)
+}
```

When $\alpha < 1$, then $h(x)$ appears to be unbounded and hence the ratio of uniforms method cannot be applied for simulating from gamma distribution. ■

2.7 Generalized ratio of uniforms method

The ratio of uniforms method discussed earlier has been extended by Wakefield et al. [1991]. Let $g(\cdot) : \mathbb{R}^+ \to \mathbb{R}^+$ denote a strictly increasing function such that $g(0) = 0$. If random vector $(U, V)^\top$ is distributed uniformly on region

$$\mathrm{C}_{h,g} = \Big\{(u,v) \in \mathbb{R}^2 \;\Big|\; 0 < u \le g^{-1}\Big[ch\Big(\frac{v}{g'(u)}\Big)\Big]\Big\}, \tag{2.59}$$

where $c > 0$ is a constant, then $X = V/g'(U)$ follows a distribution with PDF $f(x) = h(x)/\int h(x)$ in which $h(x)$ is the quasi PDF. We note that $g'(\cdot)$ denotes the first derivative

of $g(\cdot)$. The bounding rectangle $(0, a_g) \times (b_g, c_g)$ corresponds to the region $\mathrm{C}_{h,g}$ is constructed by determining

$$\begin{aligned} a_g &= \sup_x g^{-1}[ch(x)] \\ b_g &= \inf_{x \le 0} xg'[g^{-1}(ch(x))] \le 0 \\ c_g &= \sup_{x \ge 0} xg'[g^{-1}(ch(x))] \ge 0. \end{aligned} \tag{2.60}$$

A simple choice for $g(\cdot)$ is then the power function for which $g'(\cdot)$ is also strictly increasing is $g(x) = u^{r+1}/(r+1)$ (for $r \ge 0$) and hence $c = (1+r)^{-1}$. The original method is recovered for $r = 1$ for which the region $\mathrm{C}_{h,g}$ given in (2.59) turns into

$$\mathrm{C}_h(r) = \Big\{(u, v) \in \mathbb{R}^2 \;\Big|\; 0 < u \le \Big[h\Big(\frac{v}{u^r}\Big)\Big]^{1/(r+1)}\Big\}. \tag{2.61}$$

The area of $\mathrm{C}_h(r)$ given in (2.61) becomes $(1+r)^{-1} \int h(x)dx$. If $h(x)$ and $x^{r+1}[h(x)]^r$ are bounded, then the bounds of rectangle (2.61) are

$$\begin{aligned} a(r) &= \sup_x \big[h(x)\big]^{1/(r+1)} \\ b(r) &= \inf_{x \le 0} x\big[h(x)\big]^{r/(r+1)} \le 0 \\ c(r) &= \sup_{x \ge 0} x\big[h(x)\big]^{r/(r+1)} \ge 0. \end{aligned} \tag{2.62}$$

For a pair $(u, v)^\top$ generated from rectangle (2.61), the acceptance probability is

$$p(r) = \frac{\int h(x)}{(r+1) \times a(r) \times [c(r) - b(r)]}. \tag{2.63}$$

Example 2-11

Following the Example 2-9, we would like to simulate from $X \sim \mathcal{N}(0, 1)$ through the generalized ratio of uniforms method. Let $h(x) = \exp\{-x^2/2\}$ for which $\int h(x)dx = \sqrt{2\pi}$. Furthermore, we can see $a(r) = 1$ and $b(r) = -c(r) = -\sqrt{1 + 1/r}\exp\{-1/2\}$. Substituting these quantities into (2.63) yields

$$p(r) = \frac{\sqrt{2\pi r \exp\{1\}}}{2(1+r)^{3/2}}.$$

The maximum value of $p(r)$ is obtained when $r = 0.5$, that is $p(0.5) = 0.795$. Comparing with the acceptance rate of the original method discussed in Example 2-9 that is $p(1) = 0.731$, the generalized ratio of uniforms method is more efficient. ■

Table 2.1 lists some `R` commands for simulating from widely used univariate distributions. For other distributions, user may use the approaches discussed earlier in this chapter or those discussed in Devroye [1986b], Ross [2022].

2.8 Simulating multivariate distributions

This part dealing with simulating from multivariate distributions with widespread use in applications. These include Gaussian, Wishart, and Students t.

Table 2.1: Commands for simulating from some statistical families in R.

Family	command	PDF
$\mathcal{U}(a,b)$	`runif(n,min=a,max=b)`	$\frac{1}{b-a}$
$\mathcal{BET}(a,b)$	`rbeta(n,shape1=a,shape2=b,ncp=0)`	$\frac{1}{B(a,b)}x^{a-1}(1-x)^{b-1}$
$\mathcal{G}(a,b)$	`rgamma(n,shape=a,rate=b)`	$\frac{b^a}{\Gamma(a)}x^{a-1}\exp\{-bx\}$
$\chi_{(\nu)}$	`rchisq(n,df=nu,ncp=0)`	$\frac{2^{-\frac{\nu}{2}}}{\Gamma(\nu/2)}x^{\frac{\nu}{2}-1}\exp\{-\frac{x}{2}\}$
$\mathcal{N}(\mu,\sigma^2)$	`rnorm(n,mean=mu,sd=sigma)`	$\frac{1}{\sqrt{2\pi}\sigma}\exp\{-\frac{(x-\mu)^2}{2\sigma^2}\}$
$\mathcal{W}(a,b)$	`rweibull(n,shape=a,scale=b)`	$\frac{a}{b}\big(\frac{x}{b}\big)^{a-1}\exp\{-\big(\frac{x}{b}\big)^a\}$
$\mathcal{LN}(\mu,\sigma)$	`rlognorm(n,meanlog=mu,sdlog=sigma)`	$\frac{1}{x\sigma\sqrt{2\pi}}\exp\{-\frac{(\log x-\mu)^2}{2\sigma^2}\}$
$\mathcal{T}(\nu)$	`rt(n,df=nu,ncp=0)`	$\frac{\Gamma\left(\frac{\nu+1}{2}\right)}{\sqrt{\nu\pi}\Gamma(\nu/2)}\big[1+\frac{x^2}{\nu}\big]^{-\frac{\nu+1}{2}}$

2.8.1 Gaussian distribution

Let random vector $\boldsymbol{X}=(X_1,\cdots,X_p)^\top$ follows $\mathcal{N}_p(\boldsymbol{\mu},\Sigma)$. Each Gaussian distribution is characterized by its mean (location) vector $\boldsymbol{\mu}=(\mu_1,\cdots,\mu_p)^\top$ and covariance (scale) matrix Σ. The PDF of $\boldsymbol{X}$ is shown by $\mathcal{N}_p(\cdot|\boldsymbol{\mu},\Sigma)$ that is

$$\mathcal{N}_p(\boldsymbol{x}|\boldsymbol{\mu},\Sigma)=\frac{1}{\mathrm{C}_G}\exp\Big\{-\frac{\delta(\boldsymbol{x}-\boldsymbol{\mu},\Sigma)}{2}\Big\}, \tag{2.64}$$

where

$$\mathrm{C}_G=(2\pi)^{\frac{p}{2}}|\Sigma|^{\frac{1}{2}}, \tag{2.65}$$

$$\delta(\boldsymbol{x}-\boldsymbol{\mu},\Sigma)=(\boldsymbol{x}-\boldsymbol{\mu})^\top\Sigma^{-1}(\boldsymbol{x}-\boldsymbol{\mu}). \tag{2.66}$$

The quantity C_G is the normalizing constant and expression (3.29) is the well-known *Mahalanobis* distance. We write $\mathcal{N}_p(\boldsymbol{\mu},\Sigma)$ to show that random vector $\boldsymbol{X}$ follows a p-dimensional Gaussian distribution with location vector $\boldsymbol{\mu}$ and covariance matrix Σ whose PDF is given by (2.64). The structure of Σ may be shown as

$$\Sigma=\begin{pmatrix}\Sigma_{1,1}&\Sigma_{1,2}&\cdots&\Sigma_{1,p}\\ \Sigma_{2,1}&\Sigma_{2,2}&\cdots&\Sigma_{2,p}\\ \vdots&\vdots&\ddots&\vdots\\ \Sigma_{p,1}&\Sigma_{p,2}&\cdots&\Sigma_{p,p}\end{pmatrix}=\begin{pmatrix}\mathrm{var}(X_1)&\mathrm{cov}(X_1,X_2)&\cdots&\mathrm{cov}(X_1,X_p)\\ \mathrm{cov}(X_2,X_1)&\mathrm{var}(X_2)&\cdots&\mathrm{cov}(X_2,X_p)\\ \vdots&\vdots&\ddots&\vdots\\ \mathrm{cov}(X_p,X_1)&\mathrm{cov}(X_p,X_2)&\cdots&\mathrm{var}(X_p)\end{pmatrix}. \tag{2.67}$$

The matrix Σ is a symmetric and positive definite matrix and sometimes is called the variance-covariance matrix. This is because the diagonal elements of Σ are variances, that is $\mathrm{var}(X_i)=\Sigma_{i,i}$, and its (i,j)th off-diagonal element represents the linear association between marginal variables X_i and X_j, that is $\mathrm{cov}(X_i,X_j)=\Sigma_{i,j}$.

2.8.2 Properties of Gaussian distribution

Let $\boldsymbol{X}\sim\mathcal{N}_p(\boldsymbol{\mu},\Sigma)$, herein, we mention three interesting properties of Gaussian distribution.

i. Both of expected value and variance of $\boldsymbol{X}$ are finite and given by $E(\boldsymbol{X})=\boldsymbol{\mu}$ and $\mathrm{var}(\boldsymbol{X})=\Sigma$. The ML estimator of $\boldsymbol{\mu}$ and Σ, based on sample $\{\boldsymbol{x}_1,\cdots,\boldsymbol{x}_n\}$ is obtained readily as

$$\hat{\boldsymbol{\mu}}_{ML}=\bar{\boldsymbol{X}}=\frac{1}{n}\sum_{i=1}^{n}\boldsymbol{x}_i,$$
$$\hat{\Sigma}_{ML}=S=\frac{1}{n}\sum_{i=1}^{n}(\boldsymbol{x}_i-\bar{\boldsymbol{x}})(\boldsymbol{x}_i-\bar{\boldsymbol{x}})^\top, \tag{2.68}$$

respectively.

ii. For a given $d \times p$ constant matrix such as A and constant vector $\boldsymbol{c}$ of length d, we have

$$\boldsymbol{c} + A\boldsymbol{X} \sim \mathcal{N}_d(\boldsymbol{c} + A\boldsymbol{\mu}, A\Sigma A^\top). \tag{2.69}$$

iii. Let for given constant $1 \leq q < p$, the random vector $\boldsymbol{X}$ is represented as

$$\boldsymbol{X} = \begin{pmatrix} \boldsymbol{X}_1 \\ \boldsymbol{X}_2 \end{pmatrix}, \tag{2.70}$$

in which $\boldsymbol{X}_1 = (X_1, \cdots, X_q)^\top$ and $\boldsymbol{X}_2 = (X_{q+1}, \cdots, X_p)^\top$. The mean and scale parameters correspondingly can be represented as

$$\boldsymbol{\mu} = \begin{pmatrix} \boldsymbol{\mu}_1 \\ \boldsymbol{\mu}_2 \end{pmatrix} = \begin{pmatrix} E(\boldsymbol{X}_1) \\ E(\boldsymbol{X}_2) \end{pmatrix}, \tag{2.71}$$

and

$$\left(\begin{array}{c:c} \boldsymbol{\Sigma}_{11} & \boldsymbol{\Sigma}_{12} \\ \hdashline \boldsymbol{\Sigma}_{21} & \boldsymbol{\Sigma}_{22} \end{array}\right) = \left(\begin{array}{ccc:ccc} \Sigma_{1,1} & \cdots & \Sigma_{1,q} & \Sigma_{1,q+1} & \cdots & \Sigma_{1,p} \\ \Sigma_{2,1} & \cdots & \Sigma_{2,q} & \Sigma_{2,q+1} & \cdots & \Sigma_{2,p} \\ \vdots & \vdots & \vdots & \vdots & & \\ \Sigma_{q,1} & \cdots & \Sigma_{q,q} & \Sigma_{q,q+1} & \cdots & \Sigma_{q,p} \\ \hdashline \Sigma_{q+1,1} & \cdots & \Sigma_{q+1,q} & \Sigma_{q+1,q+1} & \cdots & \Sigma_{q+1,p} \\ \vdots & \vdots & \vdots & \vdots & & \\ \Sigma_{p,1} & \cdots & \Sigma_{p,q} & \Sigma_{p,q+1} & \cdots & \Sigma_{p,p} \end{array}\right), \tag{2.72}$$

where $\boldsymbol{\Sigma}_{11} = \text{var}(\boldsymbol{X}_1)$, $\boldsymbol{\Sigma}_{12} = \boldsymbol{\Sigma}_{21} = \text{cov}(\boldsymbol{X}_1, \boldsymbol{X}_2)$ and $\boldsymbol{\Sigma}_{22} = \text{var}(\boldsymbol{X}_2)$. It can be shown that the conditional distribution $\boldsymbol{X}_1$ given $\boldsymbol{X}_2 = \boldsymbol{x}_2$ is $\mathcal{N}_q(\boldsymbol{\mu}_{\boldsymbol{X}_1|\boldsymbol{X}_2}, \Sigma_{\boldsymbol{X}_1|\boldsymbol{X}_2})$ where

$$\begin{aligned} \boldsymbol{\mu}_{\boldsymbol{X}_1|\boldsymbol{X}_2} &= \boldsymbol{\mu}_1 + \boldsymbol{\Sigma}_{12}(\boldsymbol{\Sigma}_{22})^{-1}(\boldsymbol{x}_2 - \boldsymbol{\mu}_1), \\ \Sigma_{\boldsymbol{X}_1|\boldsymbol{X}_2} &= \boldsymbol{\Sigma}_{11} - \boldsymbol{\Sigma}_{12}(\boldsymbol{\Sigma}_{22})^{-1}\boldsymbol{\Sigma}_{12}^\top. \end{aligned} \tag{2.73}$$

iv. Sub-vector $\boldsymbol{X}_1 = (X_1, \cdots, X_q)^\top$, for $q = 1, \cdots, p$, follows $\mathcal{N}_q(\boldsymbol{\mu}_1, \Sigma_{11})$.

v. If $\text{cov}(X_i, X_j) = 0$, for $i \neq j = 1, \cdots, p$, then X_i and X_j are independent.

2.8.3 Generation from Gaussian distribution

Several methods have been introduced for sampling from Gaussian distribution in the literature. Herein, we represent the most three commonly used approaches that are based on decomposition (factorization) of scale matrix Σ.

i. **Cholesky decomposition**: The first approach is based on the Cholesky decomposition of scale matrix Σ. If lower triangular matrix L denotes the Cholesky decomposition of Σ, then we can write $\Sigma = LL^\top$. Furthermore, suppose $\boldsymbol{Z} = (Z_1, \cdots, Z_p)^\top$ is a vector p independent univariate standard Gaussian variates. Using property (2.69) while setting $\boldsymbol{c} = \boldsymbol{\mu}$, $A = L$, and considering the fact that $\boldsymbol{Z} \sim \mathcal{N}_p(\boldsymbol{0}, \boldsymbol{I}_p)$ in which $\boldsymbol{I}_p$ denotes a $p \times p$ identity matrix, we can easily see that

$$\boldsymbol{\mu} + L\boldsymbol{Z} \sim \mathcal{N}_p(\boldsymbol{\mu}, \Sigma). \tag{2.74}$$

ii. **spectral decomposition**: The second way relies on the spectral or eigenvalue decomposition of Σ. Let $\{\lambda_1, \cdots, \lambda_p\}$ is sequence of eigenvalues of Σ and columns of $p \times p$ matrix

V are associated eigenvectors. It is well known that Σ can be represented as

$$\begin{aligned}\Sigma =& V \begin{pmatrix} \lambda_1 & 0 & 0 & \cdots & 0 \\ 0 & \lambda_2 & 0 & \cdots & 0 \\ 0 & 0 & \lambda_3 & & 0 \\ \vdots & \vdots & & \ddots & \\ 0 & 0 & 0 & & \lambda_p \end{pmatrix} V^{-1} \\ =& VDV^\top, \end{aligned} \tag{2.75}$$

where D is a $p \times p$ diagonal matrix whose positive diagonal elements are eigenvalues of Σ. Furthermore, it should be noted that

(a) Since V is orthonormal then we have $V^{-1} = V^\top$.

(b) Usually the software represents the eigenvalues in ascending order meaning that $\lambda_1 > \lambda_2 > \cdots > \lambda_p > 0$.

(c) $\Sigma^n = VD^nV^\top$, for $n \in \mathbb{N}$.

(d) $\Sigma^{-1} = VD^{-1}V^\top$.

(e)

$$D^{-1} = \begin{pmatrix} \frac{1}{\lambda_1} & 0 & 0 & \cdots & 0 \\ 0 & \frac{1}{\lambda_2} & 0 & \cdots & 0 \\ 0 & 0 & \frac{1}{\lambda_3} & & 0 \\ \vdots & \vdots & & \ddots & \\ 0 & 0 & 0 & & \frac{1}{\lambda_p} \end{pmatrix}.$$

Using first property given above that states $V^{-1} = V^\top$, it is easy to check that $\Sigma = UU^\top$ where $U = VD^{1/2}$. So recalling property (ii) given in Sub-section 2.69 while choosing $\boldsymbol{c} = \boldsymbol{\mu}$, $A = U$, and $\boldsymbol{Z} \sim \mathcal{N}_p(\boldsymbol{0}, \boldsymbol{I}_p)$, it turns out that

$$\boldsymbol{\mu} + VD^{\frac{1}{2}}\boldsymbol{Z} \sim \mathcal{N}_p(\boldsymbol{\mu}, \Sigma). \tag{2.76}$$

In what follows,a simple the `R` code is given for simulating from p-dimensional Gaussian distribution through the Cholesky decomposition.

```
R> rmvnorm <- function(n, mean = Mu, scale = Sigma)
+{
+ p <- length(Mu)
+ L <- t( chol(Sigma) )
+ Z <- matrix( rnorm(n*p), nrow = p, ncol = n )
+ L <- t( chol(Sigma) )
+ X <- matrix(Mu, nrow = p, ncol = 1)%*%rep(1, n) + L %*%Z
+ t(X)
+}
```

There are several `R` packages that can be used for simulating from multivariate Gaussian distribution.

2.8.4 Skew Gaussian distribution

Here, we introduce the class of skew Gaussian distributions that is known in the literature as the canonical fundamental unrestricted skew Gaussian distribution, see Arellano-Valle and Azzalini

[2006]. We write $\boldsymbol{X} \sim \mathrm{SG}_{p,q}(\boldsymbol{\mu}, \Sigma, \Lambda)$ to denote that p-dimensional random vector $\boldsymbol{X}$ follows a canonical fundamental unrestricted skew Gaussian distribution with PDF given by

$$f(\boldsymbol{x}|\boldsymbol{\mu}, \Sigma, \Lambda) = 2^q \boldsymbol{\phi}_p(\boldsymbol{x}|\boldsymbol{\mu}, \Omega)\boldsymbol{\Phi}_q(\boldsymbol{m}|\mathbf{0}_q, \Delta), \tag{2.77}$$

where $\Omega = \Sigma + \Lambda\Lambda^\top$, $\Delta = \boldsymbol{I}_q - \Lambda^\top\Omega^{-1}\Lambda$, $\boldsymbol{m} = \Lambda^\top\Omega^{-1}(\boldsymbol{y} - \boldsymbol{\mu})$, and $\boldsymbol{I}_q$ denotes the $q \times q$ identity matrix. Further, $\boldsymbol{\phi}_p(.|\boldsymbol{\mu}, \Omega)$ denotes the PDF of a p-dimensional Gaussian distribution with location vector $\boldsymbol{\mu}$ and dispersion matrix Σ, and $\boldsymbol{\Phi}_q(.|\mathbf{0}_q, \Delta)$ is the CDF of a q-dimensional Gaussian distribution with location vector $\mathbf{0}_q$ (a vector of zeros of length q) and dispersion matrix Δ. The random vector $\boldsymbol{X}$ admits stochastic representation as follows, see [Arellano-Valle and Genton, 2005, Arellano-Valle and Azzalini, 2006, Arellano-Valle et al., 2007].

$$\boldsymbol{X} \overset{d}{=} \boldsymbol{\mu} + \Lambda|\boldsymbol{Z}_0| + \Sigma^{\frac{1}{2}}\boldsymbol{Z}_1, \tag{2.78}$$

where $\overset{d}{=}$ means "distributed as" and

$$\begin{bmatrix}\boldsymbol{Z}_0 \\ \boldsymbol{Z}_1\end{bmatrix} \sim \mathcal{N}_{q+p}\left(\begin{bmatrix}\mathbf{0}_q \\ \mathbf{0}_p\end{bmatrix}, \begin{bmatrix}\boldsymbol{I}_q & \mathbf{0}_{q\times p} \\ \mathbf{0}_{p\times q} & \Sigma\end{bmatrix}\right). \tag{2.79}$$

The Bayesian paradigm for $\mathrm{SG}_{p,q}(\boldsymbol{\mu}, \Sigma, \Lambda)$ distribution with representation (3.306) has been developed by Liseo and Parisi [2013]. It can be shown that [Maleki et al., 2019, Morales et al., 2022]:

$$\begin{bmatrix}\boldsymbol{Z}_0 \\ \boldsymbol{X}\end{bmatrix} \sim \mathcal{N}_{q+p}\left(\begin{bmatrix}\mathbf{0}_q \\ \mathbf{0}_p\end{bmatrix}, \begin{bmatrix}\boldsymbol{I}_q & \boldsymbol{\Lambda}^\top \\ \boldsymbol{\Lambda} & \Sigma + \boldsymbol{\Lambda}\boldsymbol{\Lambda}^\top\end{bmatrix}\right). \tag{2.80}$$

The following `R` code can be used for simulating n realizations from skew Gaussian distribution.

```
R> rsn <- function(n, mean = Mu, scale = Sigma, shape = Lambda)
+ {
+   p <- length(Mu)
+   q <- length(Lambda[1, ])
+   Y <- matrix(NA, nrow = p, ncol = n)
+   Z1 <- rmvnorm( n, mean = rep(0, p), scale = Sigma )
+   Z0 <- abs( rmvnorm(n, mean = rep(0, q), scale = diag(q)) )
+   Y <- matrix(Mu, nrow = p, ncol = 1)%*%rep(1, n) + Lambda%*%t(Z0) + t(Z1)
+   t(Y)
+ }
```

3

Importance Sampling

The Monte Carlo techniques essentially can be seen as a tool for computing single or multiple integrals. This feature of the Monte Carlo techniques that focuses on computing integrals is called *importance* sampling. It is worth noting that the *importance* sampling aims to compute the expected value of an estimator that is typically computing an integral. This concern of this chapter is placed on computing the single and multiple integrals that are widely used in statistical analyses.

3.1 Importance sampling

Let $h(x)$ to be a continuous real function and $f(x)$ is a given PDF. Suppose we would like to compute the expectation

$$E_f\big(h(X)\big) = \int_{\mathbb{R}} h(x)f(x)dx, \tag{3.1}$$

where the generic symbol $E_f(h(X))$ indicates that the expectation of random variable $h(X)$ is computed when X follows the PDF f. Usually, computing (3.1) directly is not a simple task and mus be evaluated numerically. The importance sampling is a tool to get round this difficulty. To this end, notice that the integral (3.1) can be rewritten as

$$E_f\big(h(X)\big) = \int_{\mathbb{R}} h(y)\frac{f(y)}{g(y)}g(y)dy = E_g\Big(h(X)\frac{g(X)}{f(X)}\Big) = E_g\big(h(X)w(X)\big), \tag{3.2}$$

where $w(u) = f(u)/g(u)$. Herein, the PDF $g(\cdot)$ is called the importance or instrumental PDF and its support dominates that of the PDF $f(x)$, that is $\mathcal{S}_f \subset \mathcal{S}_g$. By the law of large numbers, the RHS of (3.2) is estimated as

$$\hat{\mu} = \widehat{E_f\big(h(X)\big)} = \frac{1}{n}\sum_{i=1}^{n} w(x_i)h(x_i), \tag{3.3}$$

where $x_1, \cdots, x_n$ come independently from PDF $g(\cdot)$. Hence, by generating a large number of realizations from PDF $g(\cdot)$ the preceding expectation is approximated well. for a wide ranges of applications the target PDF $f(\cdot)$ is just known up to a normalization constant. In such cases, estimator of $E_f\big(h(X)\big)$ is given by

$$\tilde{\mu} = \widetilde{E_f\big(h(X)\big)} = \frac{\sum_{i=1}^{n} w(x_i)h(x_i)}{\sum_{i=1}^{n} w(x_i)}, \tag{3.4}$$

that is self-normalized. The bias and variance of both estimators $\hat{\mu}$ and $\tilde{\mu}$ can be evaluated [Liu and Liu, 2001]. We have

$$E_g\big(\hat{\mu}\big) = \mu \tag{3.5}$$

$$E_g\big(\tilde{\mu}\big) \approx \mu + \frac{\mu \times \mathrm{var}_g\big(w(X)\big) - \mathrm{cov}_g\big(w(X), w(X)h(X)\big)}{n} \tag{3.6}$$

and

$$\text{var}_g(\hat{\mu}) = \frac{\text{var}_g\big(w(X)h(X)\big)}{n} \tag{3.7}$$

$$\text{var}_g(\tilde{\mu}) \approx \frac{\text{var}_g\big(w(X)h(X)\big) - 2\mu \times \text{cov}_g\big(w(X), w(X)h(X)\big) + \mu^2 \times \text{var}_g\big(w(X)\big)}{n} \tag{3.8}$$

It is clear from (3.5) and (3.6) that $\hat{\mu}$ is unbiased estimator for μ while the estimator $\tilde{\mu}$ is biased. Of course, in some situations the variance (standard error) of the self-normalized estimator $\tilde{\mu}$, as it is seen from RHS of (3.7), can be smaller than that of $\hat{\mu}$ that is given by (3.8). The main feature in each importance sampling problem is to determine the instrumental PDF $g(\cdot)$. Although, theoretically, there are many choices of $g(\cdot)$, but we have to find the best choice that yields as small as possible standard error for $\hat{\mu}$ (or $\tilde{\mu}$). Fortunately, there is a specific rule available for this purpose. It is proved that the optimal choice $g_o(x)$ becomes [Johansen et al., 2010]:

$$g_o(x) = \frac{|h(x)|f(x)}{\int_{\mathbb{R}} |h(u)|f(u)du}. \tag{3.9}$$

Example 3-1

Suppose we are interested in computing $\Omega = E\big(\log(1+X)\big)$ where $X \sim \mathcal{G}(\alpha, \beta)$. We may choose three instrumentals as: 1- exponential distribution with rate parameter β, say $\mathcal{E}(\beta)$, 2- gamma distribution, say $\mathcal{G}(\alpha, \beta)$, and the optimal one with PDF $g_0(x) = h(x)\mathcal{G}(x|\alpha, \beta)$. Under the first choice, we have

$$\hat{\Omega} = \frac{\beta^{\alpha-1}}{\Gamma(\alpha)} \frac{1}{N} \sum_{i=1}^{N} x_i^{\alpha-1} \log(x_i + 1) \tag{3.10}$$

where x_is come independently from $\mathcal{E}(\beta)$. In the case of second choice of instrumental, we have

$$\hat{\Omega} = \frac{1}{N} \sum_{i=1}^{N} x_i, \tag{3.11}$$

where x_is come independently $\mathcal{G}(\alpha, \beta)$. In the case of $g_o(x)$ and $\mathcal{SG}(0, 1, \lambda)$, respectively. and second choices of instrumental, we have

$$\hat{\Omega} = N\Big[\sum_{i=1}^{N} \frac{1}{\log(x_i + 1)}\Big]^{-1}, \tag{3.12}$$

where x_is come independently from PDF given by (3.13).

$$g_o(x) = \frac{\log(x+1)\mathcal{G}(x|\alpha, \beta)}{\int_0^{\infty} \log(u+1)\mathcal{G}(u|\alpha, \beta)du}, \tag{3.13}$$

For sampling from $g_o(x)$ given above, we designate a rejection scheme given by the following steps.

1. Sample realization y from $\mathcal{G}(\alpha + 1, \beta)$ as the proposal distribution;
2. Sample independent realization u from $\mathcal{U}(0, 1)$;
3. if $y < \log(y+1)/y$, then accept y as a generation from PDF $g_o(x)$ given by (3.13); otherwise return to step (1) an repeat this procedure until having acceptance.

The R function `rlgamma`, given below, generates sample from $g_o(x)$ based on above rejection scheme.

```
R> rlgamma <- function(N, alpha, beta)
+ {
+ g <- rep(0, N) # vector of generated realizations
+ j <- 1
+ while(j <= N)
+ {
+  y <- rgamma(1, shape = alpha + 1, rate = beta)
+  u <- runif(1)
+  A <- log(y+1)/(y)
+   if( u < A )
+   {
+   g[j] <- y # y is accepted
+   j <- j + 1
+   }
+ }
+ return(g)
+ }
```

Table 3.1 represents the results of simulation study based on $M = 5000$ runs of samples each of size $N = 5000$. It follows from this table that the second instrumental, that is $\mathcal{SG}(0, 1, \lambda)$, gives the best summary. Not surprisingly, the optimal instrumental does not work as well as or better than others since $\text{var}_{g_o}(W(X))$ is inflated by the small values in the generated data from this instrumental. It is worth noting that the $g_o(\cdot)$ works well if $W(x) = f(x)/g_o(x)$ is bounded. For our case, this condition may fail since $W(x) = f(x)/g_o(x) = 1/|x|$ becomes large in presence of even a few number of small realizations. ■

Table 3.1: Summary statistics for computing $\Omega = E(\log(1 + X))$ where $X \sim \mathcal{G}(\alpha, \beta)$ through three instrumentals.

			summary			
(α, β)	Ω	instrumental	bias	SE	minimum	maximum
(0.5,0.5)	0.6695	$\mathcal{E}(\beta)$	1.1e-04	5.4e-05	0.5047	0.5611
		$\mathcal{G}(\alpha, \beta)$	-4.9e-05	3.4e-06	0.5271	0.5404
		g_o	6.2e-03	1.9e-03	0.0649	0.6249
(0.5,5)	0.0884	$\mathcal{E}(\beta)$	-4.8e-05	-4.9-05	0.0830	0.0948
		$\mathcal{G}(\alpha, \beta)$	-5.9e-07	-5.9e-07	0.0864	0.0902
		g_o	1.4e-03	1.5e-03	0.0066	0.1076
(5,0.5)	2.3152	$\mathcal{E}(\beta)$	1.2e-04	3.4e-05	2.2940	2.3356
		$\mathcal{G}(\alpha, \beta)$	-4.1e-03	1.2e-01	1.4422	6.0980
		g_o	-2.9e-05	-4.1e-05	2.2926	2.3363
(5,5)	0.6695	$\mathcal{E}(\beta)$	-1.8e-05	3.4e-05	0.6586	0.6814
		$\mathcal{G}(\alpha, \beta)$	1.1e-03	1.1e-01	0.4109	1.5190
		g_o	1.2e-05	4.1e-05	0.6562	0.6848

Next example shows that if above conditions are not met, then the importance sampling using the optimal choice $g_o(\cdot)$ may fail to show the best performance.

Example 3-2

Suppose we would like to compute $\Omega = E(|X|)$ where $X \sim \mathcal{SG}(0, 1, \lambda)$ and expression of $\mathcal{SG}(x|0, 1, \lambda)$ is given by (2.43). To this end, we consider three instrumentals including: $\mathcal{N}(0, 1)$, $\mathcal{SG}(0, 1, \lambda)$, and $g_o(x)$. The PDF of the latter is given by

$$g_o(x) = \frac{2|x|\phi(x)\Phi(\lambda x)}{\int_{-\infty}^{\infty} 2|u|\phi(u)\Phi(\lambda u)du}, \tag{3.14}$$

where quantity c plays the role of normalizing constant. But, first, we need to simulate from $g_o(x)$ given by (3.14). In what follows we give two methods for this purpose without proof.

1. Sample independent realizations z_1 and z_2 from $\mathcal{N}(0,1)$;
2. Sample independent realization $u \sim \mathcal{U}(0,1)$;
3. if $z_1 < \lambda z_2$ and $4u < |z_2|$, then accept z_2 as a generation from PDF (3.14); otherwise return to step (1) an repeat this procedure until having acceptance.

The second method is as follows.

1. Sample realization $w \sim \mathcal{W}(2, 1/\sqrt{2})$;
2. Sample independent realization $u \sim \mathcal{U}(0,1)$;
3. If $u < \Phi(\lambda w)$, then accept w as a generation from PDF (3.14); otherwise accept $-w$ as a generation from PDF (3.14).

Evidently, the second algorithm is more efficient than the first one in terms of the time of implementing. For computing $\hat{\Omega}$, we consider under the first choice of instrumental, we have

$$\hat{\Omega} = \frac{2}{N}\sum_{i=1}^{N} |x_i|\Phi(\lambda x_i), \tag{3.15}$$

where x_is come independently from $\mathcal{N}(0,1)$. In the case of second choice of instrumental, we have

$$\hat{\Omega} = \frac{1}{N}\sum_{i=1}^{N} x_i, \tag{3.16}$$

where x_is come independently $\mathcal{SG}(0,1,\lambda)$, respectively. In the case of $g_o(x)$ and $\mathcal{SG}(0,1,\lambda)$, respectively. and second choices of instrumental, we have

$$\hat{\Omega} = N\Big[\sum_{i=1}^{N} \frac{1}{|x_i|}\Big]^{-1}, \tag{3.17}$$

where x_is come independently from PDF given by (3.14). Table represents the results of simulation study based on 5000 runs of samples each of size $N = 5000$. As it may be seen from Table 3.2, overall, the second instrumental, that is $\mathcal{SG}(0,1,\lambda)$, gives the best summary. Not surprisingly, the optimal instrumental does not work as well as or better than others since $\text{var}_{g_o}\big(W(X)\big)$ is inflated by the small values in the generated data from this instrumental. It is worth noting that the $g_o(\cdot)$ works well if $W(x) = f(x)/g_o(x)$ is bounded. For our case, this condition may fail since $W(x) = f(x)/g_o(x) = 1/|x|$ becomes large in presence of even a few number of small realizations. ■

3.2 Variance reduction

Suppose $\hat{\theta}$ is an unbiased candidate (estimator) for estimating unknown parameter θ. This means that $E(\hat{\theta}) = \theta$. For a better unbiased estimator $\hat{\theta}^*$, if there exists, we have $var(\hat{\theta}^*) < var(\hat{\theta})$. Here, we are interested in finding $\hat{\theta}^*$. To this end, we recall the well-known Rao-Blackwellization approach that states the standard error of an given biased (or unbiased) estimator $\hat{\theta}$ can be reduced by conditioning on a *sufficient* statistics (estimator). Overall, we can write

$$var(\hat{\theta}) = E\big[var(\hat{\theta}|T)\big] + var\big[E(\hat{\theta}|T)\big]. \tag{3.18}$$

Table 3.2: Summary statistics for computing $\Omega = E(|X|)$ where $X \sim \mathcal{SG}(0, 1, \lambda)$ through three instrumentals.

		summary			
λ	instrumental	bias	SE	minimum	maximum
$\lambda = 0$	$\mathcal{N}(0, 1)$	5.1e-05	0.0083	0.7688	0.8250
	$\mathcal{SG}(0, 1, \lambda = 0)$	3.8e-05	0.0084	0.7663	0.8259
	$g_o(\cdot)$	1.0e-03	0.0218	0.4925	0.8574
$\lambda = 5$	$\mathcal{N}(0, 1)$	4.3e-04	0.0168	0.7267	0.8560
	$\mathcal{SG}(0, 1, \lambda = 5)$	3.5e-05	0.0084	0.7693	0.8290
	$g_o(\cdot)$	1.1e-03	0.0214	0.5341	0.8573
$\lambda = 10$	$\mathcal{N}(0, 1)$	7.2e-05	0.0165	0.7367	0.8553
	$\mathcal{SG}(0, 1, \lambda = 10)$	2.9e-05	0.0084	0.7683	0.8311
	$g_o(\cdot)$	6.9e-04	0.0221	0.2998	0.8533

This implies

$$var[E(\hat{\theta}|T)] < var(\hat{\theta}). \tag{3.19}$$

Hence, $\hat{\theta}^* = E(\hat{\theta}|T)$ is an unbiased estimator whose standard error is smaller than that of the $\hat{\theta}$.

Example 3-3

Suppose $X_1, X_2 \sim \mathcal{N}(\mu, 1)$ are independent random variables. It is known that $T = X_1 + X_2 \sim \mathcal{N}(2\mu, 2)$ is a sufficient statistic for unknown parameter μ. Moreover, we know that $\hat{\theta} = X_1^2 - 1$ is an unbiased estimator for μ^2 with variance 1. But, an estimator of μ^2 with smaller standard error, called here $\hat{\theta}^*$, can be found using Rao-Blackwellization procedure as follows. First, we can see that the PDF of X_1 given $T = X_1 + X_2$ is

$$\begin{aligned} f_{X_1|T}(x_1|T = t) &= \frac{f_{X_1,X_2}(x_1, t - x_1)}{f_T(t)} \\ &= \frac{f_{X_1}(x_1) f_{X_2}(x_1, t - x_1)}{f_T(t)} \\ &= \frac{\sqrt{2}}{\sqrt{2\pi}} \exp\{-(x_1 - t/2)^2\}. \end{aligned}$$

So, $X_1|X_1 + X_2 = t \sim \mathcal{N}(t/2, 1/2)$. It follows that

$$\begin{aligned} \hat{\theta}^* =& E[\hat{\theta}|T = t] \\ =& E[X_1^2|X_1 + X_2 = t] - 1 \\ =& var[X_1|X_1 + X_2 = t] + \{E[X_1|X_1 + X_2 = t]\}^2 - 1 \\ =& \frac{1}{2} + \frac{t^2}{4} - 1 \\ =& \bar{X}^2 - \frac{1}{2}. \end{aligned}$$

It is evident that both estimators $\hat{\theta}$ and $\hat{\theta}^*$ are unbiased for μ^2, that is $E(\hat{\theta}) = E(\hat{\theta}^*) = \mu^2$, but $var(\hat{\theta}^*) < var(\hat{\theta})$. ■

3.3 Estimators with negative covariance

Suppose both of estimators $\hat{\theta}_1$ and $\hat{\theta}_2$ are unbiased for unknown parameter θ. If $cov(\hat{\theta}_1, \hat{\theta}_2) < 0$, then $\hat{\theta}^* = [\hat{\theta}_1 + \hat{\theta}_2]/2$ is also unbiased estimator for θ with smaller standard error than either $\hat{\theta}_1$ or $\hat{\theta}_2$. This because

$$\begin{aligned} var(\hat{\theta}^*) =& \frac{var(\hat{\theta}_1) + var(\hat{\theta}_2)}{4} + \frac{cov(\hat{\theta}_1, \hat{\theta}_2)}{2} \\ \leq& \frac{var(\hat{\theta}_1) + var(\hat{\theta}_2)}{4} \\ <& \frac{1}{2} \max\{var(\hat{\theta}_1), var(\hat{\theta}_2)\}. \end{aligned}$$

Example 3-4

Let random variable X follows a BS distribution with PDF given by (2.38) in which parameter β is known. It can be shown that Ng et al. [2003]:

$$E(X) = \beta\Big(1 + \frac{\alpha^2}{2}\Big).$$

Since $1/X \sim \mathcal{BS}(\alpha, 1/\beta)$, we can write

$$E\Big(\frac{1}{X}\Big) = \frac{1}{\beta}\Big(1 + \frac{\alpha^2}{2}\Big).$$

Based on a random sample of size n denoted by $x_1, \cdots, x_n$ drawn from $\mathcal{BS}(\alpha, 1)$, one can see that $\hat{\theta}_1 = 2(S-1)$ and $\hat{\theta}_2 = 2(R-1)$ are unbiased estimators for α^2 in which $S = 1/n \sum_{i=1}^n x_i$ and $R = 1/n \sum_{i=1}^n 1/x_i$. Intuitively, the statistics S and R have negative correlation, and hence, we can construct a new estimator represented as

$$\hat{\theta}^* = \frac{\hat{\theta}_1 + \hat{\theta}_2}{2},$$

whose standard error is less that that of $\hat{\theta}_1$ and $\hat{\theta}_2$. We performed a small comparison study in which four estimators including $\hat{\theta}_{MLE}$ (ML estimator of α^2), $\hat{\theta}^*$, $\hat{\theta}_1$, and $\hat{\theta}_2$ take part for estimating α^2 when β is assumed to be known. To this end, aforementioned four competitors have been computed for based on random sample of size $n = 100$ for $N = 1000$ runs. It is worth to note that for simulating for BS distribution we used the `R` function `rbs` given in 2.3.7 and the ML estimator of parameter α^2 is computed using package `bibs` available at `https://cran.r-project.org/web/packages/bibs/index.html`, then the $\hat{\theta}_{MLE}$ is easily obtained as the squared ML estimator of α by the general nice invariance property of the ML estimator. Figure 3.1 shows the average and standard error of these competitors. As it is seen, in terms of average and standard error, both estimators $\hat{\theta}_{MLE}$ and $\hat{\theta}^*$ shows the same performance and simultaneously outperform $\hat{\theta}_1$ and $\hat{\theta}_2$. The corresponding `R` code is given by the following.

```
1 R> library("bibs")
2 R> N <- 1000; n <- 100; alpha <- seq(0.2, 5, length = 10); beta <- 1
3 R> r <- s <- alpha2_mle <- alpha2_star <- alpha2_hat1 <- alpha2_hat2 <- rep(0, N)
4 R> bias <- rmse <- matrix(0, nrow = length(alpha), ncol = 4 )
5 R> for(j in 1:length(alpha) ){
6 + for(i in 1:N){
7 + x <- rbs(n, alpha[j], beta); s[i] <- mean(x); r[i] <- mean(1/x)
```

```
+ alpha2_mle[i] <- mlebs(x, c(alpha[j], beta), method = "Nelder-Mead", CI = 0.95)$Estimates
    [[1]]
+ alpha2_hat1[i] <- ( 2*(r[i]*beta - 1) ); alpha2_hat2[i] <- ( 2*(s[i]/beta - 1) )
+ alpha2_star[i] <- ( alpha2_hat1[i] + alpha2_hat2[i] )/2
+ }
+ hat <- as.matrix( cbind(alpha2_mle^2, alpha2_star, alpha2_hat1, alpha2_hat2), nrow = N,
    ncol = 4 )
+ shifted <- sweep(hat, 2, alpha[j]^2, FUN = '-')
+ bias[j, ] <- apply( shifted, 2, mean )
+ rmse[j, ] <- sqrt( apply( shifted, 2, var ) )
+ }
```

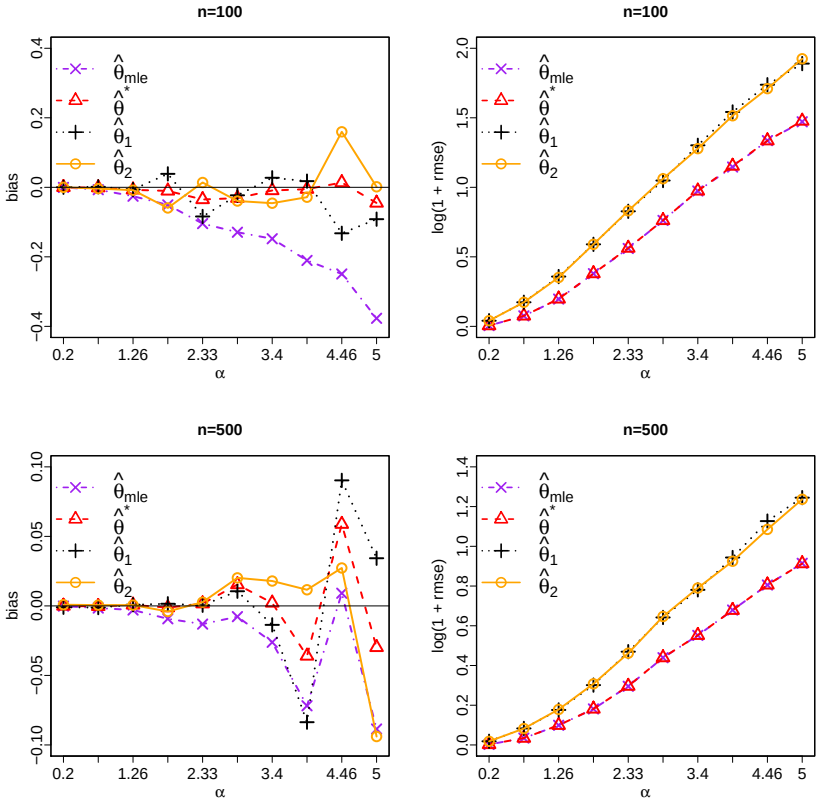


Figure 3.1: The top row shows the bias and log(1+rmse) for $\hat{\theta}_{MLE}$, $\hat{\theta}^*$, $\hat{\theta}_1$, and $\hat{\theta}_2$ computed based on $N = 1000$ independent sample each of size $n = 100$ generated from $\mathcal{BS}(\alpha, \beta = 1)$. The bottom row displays the same graph with the same parameters except $n = 500$.

■

Example 3-5

Let $U, V \sim \mathcal{U}(-1, 1)$ are independent random variables distributed jointly on $(-1, 1) \times (-1, 1)$ (shown by a square in Figure 3.2). It may bee seen that the probability of falling a point inside the circle in Figure 3.2 is $\pi/4$. Hence, the constant π is approximately as

$$\hat{\pi} = 4 \times \frac{\text{number of pairs lie inside the circle}}{\text{number of pairs do not lie inside the circle}}. \tag{3.20}$$

Therefore, the constant π can be estimated by simulating a sufficiently large (n, say) numbers of point distributed uniformly on $(-1, 1) \times (-1, 1)$ and then computing the ratio 3.20. To this end, we define

$$\mathbb{I}_{\mathcal{C}}(u_i, v_i) = \begin{cases} 1, & \text{if } i\text{th pair } (u_i, v_i)^\top \text{ lies inside the circle,} \\ 0, & \text{if } i\text{th pair } (u_i, v_i)^\top \text{ lies outside the circle,} \end{cases} \tag{3.21}$$

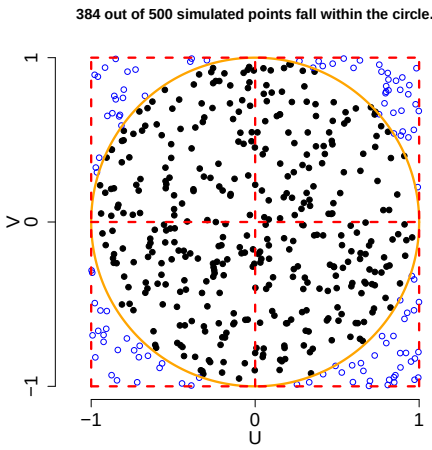


Figure 3.2: Components of random vector $(U, V)^\top$ come independently form $\mathcal{U}(-1, 1)$.

and then, based on the law of large numbers, we have

$$\hat{\pi} = 4 \times \frac{\sum_{i=1}^{n} \mathbb{I}_{\mathcal{C}}(u_i, v_i)}{n}. \tag{3.22}$$

It is easy to check that $E(\hat{\pi}) = \pi$ and $var(\hat{\pi}) = 4\pi(1-\pi/4)/n = 2.69676/n$. Here, we interested in improving estimator $\hat{\pi}$ by conditioning on marginal variable U as follows.

$$\begin{aligned}
\hat{\pi}_1 =& \frac{4}{n} \sum_{i=1}^{n} E[\mathbb{I}_{\mathcal{C}}(U_i, V_i)|U_i = u_i] \\
=& \frac{4}{n} \sum_{i=1}^{n} P[U_i^2 + V_i^2 \leq 1|U_i = u_i] \\
=& \frac{4}{n} \sum_{i=1}^{n} P[V_i^2 \leq 1 - u_i^2|U_i = u_i] \\
=& \frac{4}{n} \sum_{i=1}^{n} P[V_i^2 \leq 1 - u_i^2] \\
=& \frac{4}{n} \sum_{i=1}^{n} \sqrt{1 - u_i^2},
\end{aligned}$$

where $U_i \sim \mathcal{U}(-1, 1)$. This yields

$$\hat{\pi}_1 = \frac{4}{n} \sum_{i=1}^{n} \sqrt{1 - U_i^2}.$$

Therefore, $\hat{\pi}_1$ is obtained, first by generating $u_1, \cdots, u_n \sim \mathcal{U}(-1, 1)$, and then computing $\hat{\pi}_1 = 4/n \sum_{i=1}^{n} \sqrt{1 - u_i^2}$. It is easy to check that

$$\begin{aligned}
E(\hat{\pi}_1) &= \frac{4}{n} \sum_{i=1}^{n} E\Big(\sqrt{1 - U_i^2}\Big) \\
&= \frac{4}{n} \sum_{i=1}^{n} \int_{-1}^{1} \sqrt{1 - y^2} \frac{1}{2} dy \\
&= \frac{4}{n} \sum_{i=1}^{n} E\Big(\sqrt{1 - Z_i^2}\Big) \\
&= \pi,
\end{aligned} \qquad
\begin{aligned}
var(\hat{\pi}_1) &= \frac{16}{n^2} \sum_{i=1}^{n} \Big[E(1 - Z_i^2) - \Big(\frac{\pi}{4}\Big)^2\Big], \\
&= \frac{16}{n} \Big[\frac{2}{3} - \Big(\frac{\pi}{4}\Big)^2\Big] \\
&= \frac{0.79706}{n}.
\end{aligned}$$

where $Z_1, \cdots, Z_n \sim \mathcal{U}(0,1)$. As it is seen, $\hat{\pi}_1$ yields around 70.44% reduction in variance for estimating π compared to $\hat{\pi}$. Yet the performance of estimator $\hat{\pi}_1$ can be improved using the method introduced in Section 3.3. It follows from above that quantity

$$\frac{4}{n}\sum_{i=1}^{n}\sqrt{1-Z_i^2}$$

with variance $0.79706/n$. We know that $Z \sim 1-Z \sim \mathcal{U}(0,1)$ and so the quantity

$$\frac{4}{n}\sum_{i=1}^{n}\sqrt{1-(1-Z)_i^2}$$

is also unbiased estimator for π with the same variance. But, evidently Z and $1-Z$ are correlated negatively and hence

$$\hat{\pi}_2 = \frac{2}{n}\sum_{i=1}^{n}\sqrt{1-Z_i^2} + \frac{2}{n}\sum_{i=1}^{n}\sqrt{1-(1-Z_i)^2},$$

is an unbiased estimator for π whose variance is smaller than that of $\hat{\pi}_1$. As it may seen from

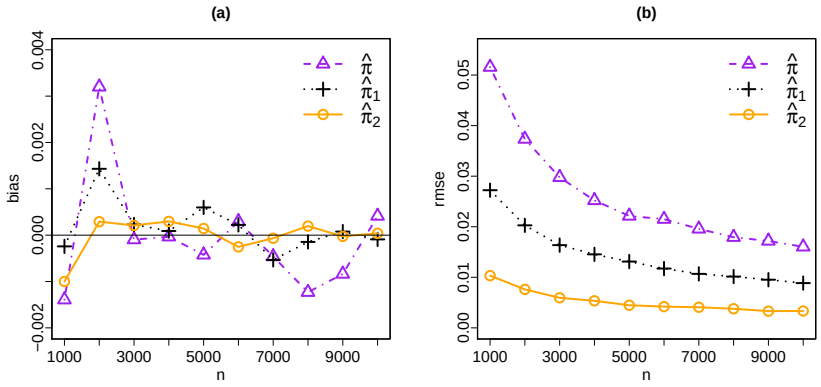


Figure 3.3: The bias (a) and rmse (b) of $\hat{\pi}$, $\hat{\pi}_1$, and $\hat{\pi}_2$ computed based on $N = 1000$ independent samples of size n.

Figure 3.3, the $\hat{\pi}_2$ outperforms $\hat{\pi}_1$ and $\hat{\pi}$, in terms of both bias and rmse criteria. The pertinent R code for computing the bias and rmse of $\hat{\pi}$, $\hat{\pi}_1$, and $\hat{\pi}_2$ is given by the following.

```
R> set.seed(20241012)
R> N <- 1000; n <- 100; sample<- seq(1000, 10000, 1000)
R> pi_hat <- pi_hat_star1 <- pi_hat_star2 <- rep(0, N)
R> bias <- rmse <- matrix(0, nrow = length(sample), ncol = 3 )
R> for(j in 1:length(sample) )
+ {
+  for(i in 1:N)
+ {
+   u <- runif(sample[j], -1, 1); v <- runif(sample[j], -1, 1); Z <- runif(sample[j], 0, 1)
+   pi_hat[i] <- 4*mean( ifelse(u^2 + v^2 <= 1, 1, 0) )
+   pi_hat_star1[i] <- 4*mean( sqrt(1 - u^2) )
+   pi_hat_star2[i] <- ( 4*mean( sqrt(1 - Z^2) ) + 4*mean( sqrt(1 - (1 - Z)^2) ) )/2
+  }
+   hat <- as.matrix( cbind(pi_hat, pi_hat_star1, pi_hat_star2), nrow = N, ncol = 3 )
+   colnames(hat) <- NULL
+   shifted <- sweep(hat, 2, pi, FUN = '-')
+   bias[j, ] <- apply( shifted, 2, mean )
+   rmse[j, ] <- sqrt( apply( shifted, 2, var ) )
+ }
```

3.4 Computing the CDF of Gaussian distribution

In this section, we proceed to provide the compute the Monte Carlo approximation for the CDF of the Gaussian distribution.

3.4.1 Computing the CDF of univariate Gaussian distribution

Indeed computing the CDF of a Gaussian distribution is an importance sampling problem. Following example gives more detailed information.

Example 3-6

Let $Z \sim \mathcal{N}(0,1)$ and we are interested in computing $P(Z < 2)$. Evidently the given probability can be represented as

$$\begin{aligned} P(Z<2) &= \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{2} \exp\Big\{-\frac{z^2}{2}\Big\} dz \\ &= \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} \mathbb{I}_{(-\infty,2)}(z) \times \exp\Big\{-\frac{z^2}{2}\Big\} dz \\ &= E\Big(\mathbb{I}_{(-\infty,2)}(Z)\Big), \end{aligned} \tag{3.23}$$

where $\mathbb{I}_{\mathcal{S}}(x)$ is the well-known indicator function defined as

$$\mathbb{I}_{\mathcal{S}}(x) = \begin{cases} 1, & \text{if } x \in \mathcal{S}, \\ 0, & \text{if } x \notin \mathcal{S}. \end{cases} \tag{3.24}$$

Hence, the quantity can be readily approximated by considering the standard univariate Gaussian as the instrumental distribution. It follows, based on the law of large numbers, that

$$P(Z \leq 2) \approx \frac{1}{N} \sum_{i=1}^{N} \mathbb{I}_{(-\infty,2)}(z_i) = \frac{\text{number of } z_i\text{s less or equal 2 in a sample of size } N}{N}, \tag{3.25}$$

where $z_1, \cdots, z_N$ independently follow $\mathcal{N}(0,1)$ and N is a sufficiently large integer value. This means that the Monte Carlo approximation of $P(Z < 2)$ given in (3.25) is the average of samples that are less than two. The Central limit theorem guarantees that estimator (3.25) is unbiased for $P(Z < 2)$ with variance

$$\frac{\Phi(2|0,1)\big[1 - \Phi(2|0,1)\big]}{N}.$$

■

Of course, the command `integrate(f,lower,upper,...)` (when `f` is the PDF of Gaussian distribution) or `pnorm(q,mean=0,sd=1,...)` can be used in `R` environment for computing the CDF of Gaussian distribution. However, one could use the well known *Gaussian quadrature* rules discussed in next Chapter.

3.5 Computing the CDF of multivariate Gaussian distribution

Due to significant importance of the multivariate Gaussian distribution in a wide range of study fields, computing its CDF has a long history. Several efforts have been made for computing the CDF of multivariate Gaussian distribution. For more detailed information, we refer the reader to Botev [2017] and references therein. One of the pioneers in this context was Genz [1992]

who suggested the separation of variables (SOV) method. Let $\boldsymbol{X} \sim \mathcal{N}_p(\boldsymbol{\mu}, \Sigma)$ and L denote the Cholesky decomposition of Σ, that is $\Sigma = LL^\top$ in which L is a $p \times p$ lower triangular matrix. For computing $\mathcal{P}_G = P(\boldsymbol{a} < \boldsymbol{X} < \boldsymbol{b})$, in which $\boldsymbol{a}$ and $\boldsymbol{b}$ are vectors of real constants, the SOV method can be described as follows. Consider transformation $\boldsymbol{x} = L\boldsymbol{y}$, we have

$$\boldsymbol{x}^\top \Sigma^{-1} \boldsymbol{x} = \boldsymbol{y}^\top L^\top L^{-\top} L^{-1} L \boldsymbol{y} = \boldsymbol{y}^\top \boldsymbol{y} = \sum_{i=1}^{p} y_i^2,$$

and $d\boldsymbol{x} = |L| d\boldsymbol{y} = \sqrt{|\Sigma|} d\boldsymbol{y}$. Hence, $\boldsymbol{a} < L\boldsymbol{y} < \boldsymbol{b}$ or equivalently

$$\begin{aligned}
\frac{a_1}{L_{1,1}} \leq \quad & y_1 \quad \leq \frac{b_1}{L_{1,1}} \\
\frac{a_2 - L_{2,1} y_1}{L_{2,2}} \leq \quad & y_2 \quad \leq \frac{b_2 - L_{2,1} y_1}{L_{2,2}} \\
& \vdots \\
\frac{a_p - \sum_{i=1}^{p-1} L_{p,i} y_i}{L_{p,p}} \leq \quad & y_p \quad \leq \frac{b_p - \sum_{i=1}^{p-1} L_{p,i} y_i}{L_{p,p}}.
\end{aligned}$$

This type of change of variable turns the problem of computing $\mathcal{P}$ into the product of p univariate integrals as

$$\mathcal{P}_G = (2\pi)^{-\frac{p}{2}} \int_{a_1'}^{b_1'} \exp\Big\{-\frac{y_1^2}{2}\Big\} \int_{a_2'}^{b_2'} \exp\Big\{-\frac{y_2^2}{2}\Big\} \cdots \int_{a_p'}^{b_p'} \exp\Big\{-\frac{y_m^2}{2}\Big\} d\boldsymbol{y},$$

where $a_i' = (a_i - \sum_{j=1}^{i-1} L_{i,j} y_j)/L_{i,i}$ and $b_i' = (b_i - \sum_{j=1}^{i-1} L_{i,j} y_j)/L_{i,i}$ in which, by convention, we assume $\sum_{j=1}^{0} L_{i,j} = 0$. Using transformation of the form $z_i = \Phi(y_i)$ (for $i = 1, \cdots, p$), we obtain

$$\mathcal{P}_G = \int_{d_1}^{e_1} \int_{d_2}^{e_2} \cdots \int_{d_p}^{e_p} d\boldsymbol{z}, \tag{3.26}$$

where

$$\begin{aligned}
d_i =& \Phi\Big(\frac{a_i - \sum_{j=1}^{i-1} L_{i,j} \Phi^{-1}(z_j)}{L_{i,i}}\Big), \\
e_i =& \Phi\Big(\frac{b_i - \sum_{j=1}^{i-1} L_{i,j} \Phi^{-1}(z_j)}{L_{i,i}}\Big).
\end{aligned}$$

For computational purposes, we apply an extra transformation $w_i = (z_i - d_i)/(e_i - d_i)$ by which $\mathcal{P}_G$ in (3.26) becomes a $(p-1)$-dimensional integral in unit hypercube $[0,1]^{p-1}$. It follows that

$$\begin{aligned}
\mathcal{P}_G &= \int_0^1 (e_1' - d_1') dw_1 \int_0^1 (e_2' - d_2') dw_2 \cdots \int_0^1 (e_{p-1}' - d_{p-1}') dw_{p-1} \int_0^1 (e_p' - d_p') dw_p, \\
&= (e_1' - d_1') \int_0^1 (e_2' - d_2') dw_1 \int_0^1 (e_3' - d_3') dw_2 \cdots \int_0^1 (e_p' - d_p') dw_{p-1} \int_0^1 dw_p, \\
&= (e_1' - d_1') \int_0^1 (e_2' - d_2') dw_1 \int_0^1 (e_3' - d_3') dw_2 \cdots \int_0^1 (e_p' - d_p') dw_{p-1},
\end{aligned} \tag{3.27}$$

where

$$\begin{aligned}
d_i' =& \Phi\Big(\frac{a_i - \sum_{j=1}^{i-1} L_{i,j} \Phi^{-1}\big(d_j + w_j(e_j - d_j)\big)}{L_{i,i}}\Big), \\
e_i' =& \Phi\Big(\frac{b_i - \sum_{j=1}^{i-1} L_{i,j} \Phi^{-1}\big(d_j + w_j(e_j - d_j)\big)}{L_{i,i}}\Big).
\end{aligned}$$

It is worth to note that in the RHS of (3.27) the quantity $e'_1 - d'_1$ is independent of w. For computing $\mathcal{P}_G$ Genz [1992] suggests a Monte Carlo approach in which w_i (for $i = 1, \cdots, p-1$) is drawn from $\mathcal{U}(0,1)$ as the instrumental distribution. This method as originally developed for zero-mean Gaussian distribution. In general, when $\boldsymbol{\mu}$ of, the constants $\boldsymbol{a}$ and $\boldsymbol{b}$ are replaced with $\boldsymbol{a} - \boldsymbol{\mu}$ and $\boldsymbol{b} - \boldsymbol{\mu}$, respectively. The number of Monte Carlo iterations N, typically is large and hence, the central limit theorem (CLT) guarantees that the standard error (se) for estimator of $\mathcal{P}_G$ is asymptotically

$$\text{se} = \alpha \frac{\sigma}{\sqrt{N}},$$

where σ is the standard deviation of the Monte Carlo approximation. Naturally σ is unknown and therefore is estimated based on N Monte Carlo iterations. The constant α is the Gaussian upper quantile such as $z_{0.995} \approx 2.576$ by which one expects the absolute error in approximating $\mathcal{P}_G$ to be less than given error in 99 percent of the time. Genz [1992] suggests the use of $\alpha = 2.5$. Algorithm 10 given below describes the method of Genz [1992] for computing $\mathcal{P}_G$ in more details.

Algorithm 10 Computing CDF of the multivariate Gaussian distribution

Read $\boldsymbol{a}$, $\boldsymbol{b}$, $\boldsymbol{\mu}$, Σ, ϵ, and N_{max};
Compute the Cholesky decomposition L of Σ;
Set s to be a vector of zeros of length N_{max} and $\alpha = 2.5$;
Set $\boldsymbol{b} = \boldsymbol{b} - \boldsymbol{\mu}$ and $\boldsymbol{a} = \boldsymbol{a} - \boldsymbol{\mu}$;
Set $e_1 = \Phi(b_1/L_{1,1})$, $d_1 = \Phi(a_1/L_{1,1})$, and $f_1 = e_1 - d_1$;
Set $N = 0$ and se $= 10$ (or any value greater than ϵ);
while se $> \epsilon$ and $N < \text{N}_{max}$ **do**
    Generate realizations $w_1, w_2, \cdots, w_{p-1}$ from $\mathcal{U}(0,1)$;
    Set $i = 2$;
    while $i \leq p$ **do**
        Set $j = i - 1$;
        $y_j = \Phi^{-1}\Big(d_j + w_j(e_j - d_j)\Big)$;
        $L_y = \sum_{k=1}^{j} L_{i,k} \times y_k$;
        $d_i = \Phi\Big(\frac{a_i - L_y}{L_{i,i}}\Big)$;
        $e_i = \Phi\Big(\frac{b_i - L_y}{L_{i,i}}\Big)$;
        $f_i = (e_i - d_i) f_j$;
        Set $i = i + 1$;
    end
    $N = N + 1$;
    $s[N] = f_p$ where $s[N]$ denotes the Nth element of vector s;
    $\sigma = \sqrt{\text{Var}(s[1:N])}$;
    se $= \alpha \frac{\sigma}{\sqrt{N}}$;
end
Return $\sum_{r=1}^{N} s[r]/N$ as an estimator for the CDF of multivariate Gaussian distribution with standard error se.

In the following, the `R` function `pmvnorm_Genz` is written for implementing the Genz [1992] algorithm.

```
R> pmvnorm_Genz <- function(a, b, Mu, Sigma, epsilon = 10e-4, N_max = 200)
+{
+ if(any(a >= b) == TRUE) stop(message("vector a must be smaller than vector b."))
```

```
+ p <- length(Mu)
+ L <- t( chol(Sigma) )
+ s <- rep(0, N_max)
+ alpha <- 2.5
+ a0 <- b0 <- d <- e <- f <- w <- y <- rep(0, p)
+ a0 <- a - Mu
+ b0 <- b - Mu
+ k <- 5.32
+ threshold <- k*sqrt( diag(Sigma) )
+  for(i in 1:p)
+  {
+   if( is.infinite( a0[i] ) ) a0[i] <- Mu[i] - threshold[i]
+   if( is.infinite( b0[i] ) ) b0[i] <- Mu[i] + threshold[i]
+   if( is.infinite( a[i] ) & a0[i] >= b0[i] ) a0[i] <- b0[i] - threshold[i]
+   if( is.infinite( b[i] ) & a0[i] >= b0[i] ) b0[i] <- a0[i] + threshold[i]
+  }
+  d[1] <- pnorm( a0[1]/L[1, 1] )
+  e[1] <- pnorm( b0[1]/L[1, 1] )
+  f[1] <- e[1] - d[1]
+  se <- 10
+  N <- 0
+  while( se > epsilon | N < N_max )
+  {
+   w <- runif(p - 1)
+   i <- 2
+    while( i <= p )
+    {
+     j <- i - 1
+     y[j] <- qnorm( d[j] + w[j]*(e[j] - d[j]) )
+     L_y <- sum( L[i, (1:j)]*y[1:j] )
+     d[i] <- pnorm( (a0[i] - L_y )/L[i, i] )
+     e[i] <- pnorm( (b0[i] - L_y )/L[i, i] )
+     f[i] <- (e[i] - d[i]) * f[j]
+     i <- i + 1
+    }
+   N <- N + 1
+   s[N] <- f[p]
+   sigma <- sqrt( var( s[1:N] ) )
+   se <- alpha*sigma/sqrt(N)
+   se <- ifelse( is.na(se), 10, se )
+   }
+ ls <- list("cdf" = mean(s), "se" = se)
+ return(ls)
+}
```

3.5.1 Computing CDF of the multivariate Student's t distribution

Let random vector $\boldsymbol{X} = (X_1, \cdots, X_p)^\top$ follow a p-dimensional Student's t distribution with $\nu > 0$ degrees of freedom. The PDF of $\boldsymbol{X}$ is then given by

$$t_\nu(\boldsymbol{x}|\boldsymbol{\mu}, \Sigma) = \frac{\Gamma\left(\frac{\nu+p}{2}\right)}{(\nu\pi)^{\frac{p}{2}}\Gamma\left(\frac{\nu}{2}\right)|\Sigma|^{\frac{1}{2}}}\left[1 + \frac{\delta(\boldsymbol{x} - \boldsymbol{\mu}, \Sigma)}{\nu}\right]^{-\frac{\nu+p}{2}}, \tag{3.28}$$

where $\delta(\cdot)$ is the ell-known Mahalanobis distance defined as

$$\delta(\boldsymbol{x}, \Sigma) = \boldsymbol{x}^\top \Sigma^{-1} \boldsymbol{x}, \tag{3.29}$$

and $\boldsymbol{\mu}$, Σ, and ν denote the location, scale, and degrees of freedom parameters, respectively. We write $\boldsymbol{X} \sim t_\nu(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ to indicate that $\boldsymbol{X}$ follows a Student's t distribution with PDF (3.28).

In univariate case, the PDF of Student's t distribution with ν degrees of freedom given by

$$t_\nu(x|\mu,\sigma) = \frac{\Gamma\big(\frac{\nu+1}{2}\big)}{\sigma\sqrt{\nu\pi}\Gamma\big(\frac{\nu}{2}\big)}\Big[1+\frac{z^2}{\nu}\Big]^{-\frac{\nu+1}{2}}. \tag{3.30}$$

where $z = (x-\mu)/\sigma$. For a given real constant a, we note that the CDF associated with the PDF (3.30) is represented as

$$T_\nu(x|\mu,\sigma) = \int_{-\infty}^{x} \frac{\Gamma\big(\frac{\nu+1}{2}\big)}{\sigma\sqrt{\nu\pi}\Gamma\big(\frac{\nu}{2}\big)}\Big[1+\frac{z^2}{\nu}\Big]^{-\frac{\nu+1}{2}} du, \tag{3.31}$$

where $z = (u-\mu)/\sigma$. Moreover, the quantile function (inverse of CDF) of the Student's t distribution with ν degrees of freedom is represented as

$$T_\nu^{-1}(u) = \min\{x \mid T_\nu(x|\mu,\sigma) \geq u\}, 0 < u < 1,$$

The focus of this part concerns with computing $\mathcal{P}_t = P(\boldsymbol{a} < \boldsymbol{X} < \boldsymbol{b})$ based on method proposed by Genz and Bretz [1999]. Without loss of generality, we assume that $\boldsymbol{\mu} = \boldsymbol{0}$. If $\boldsymbol{\mu} \neq 0$, the constants $\boldsymbol{a}$ and $\boldsymbol{b}$ are replaced with $\boldsymbol{a}-\boldsymbol{\mu}$ and $\boldsymbol{b}-\boldsymbol{\mu}$, respectively. By definition we have

$$\mathcal{P}_t = \int_{\boldsymbol{a}}^{\boldsymbol{b}} t_\nu(\boldsymbol{x}|\boldsymbol{\mu},\Sigma)d\boldsymbol{x} \tag{3.32}$$

$$= \frac{\Gamma\big(\frac{\nu+p}{2}\big)}{(\nu\pi)^{\frac{p}{2}}\Gamma\big(\frac{\nu}{2}\big)|\Sigma|^{\frac{1}{2}}}\int_{\boldsymbol{a}}^{\boldsymbol{b}}\Big[1+\frac{\delta(\boldsymbol{x},\Sigma)}{\nu}\Big]^{-\frac{\nu+p}{2}} d\boldsymbol{x}, \tag{3.33}$$

where $\boldsymbol{a}$ and $\boldsymbol{b}$ are two real constant vectors of size p such that $\boldsymbol{a} < \boldsymbol{b}$. Based on the SOV method, described earlier in Section 3.5 for computing the CDF of multivariate Gaussian distribution, we have

$$\begin{aligned}\mathcal{P}_t &= \frac{\Gamma\big(\frac{\nu+p}{2}\big)}{(\nu\pi)^{\frac{p}{2}}\Gamma\big(\frac{\nu}{2}\big)}\int_{\boldsymbol{a}'}^{\boldsymbol{b}'}\Big[1+\frac{\sum_{i=1}^p y_i^2}{\nu}\Big]^{-\frac{\nu+p}{2}} d\boldsymbol{y}\\ &= \mathrm{C}_t(\nu,p)\int_{\boldsymbol{a}'}^{\boldsymbol{b}'}\Big[1+\frac{\sum_{i=1}^p y_i^2}{\nu}\Big]^{-\frac{\nu+p}{2}} d\boldsymbol{y},\end{aligned}$$

where $a_i' = (a_i - \sum_{j=1}^{i-1} L_{i,j}y_j)/L_{i,i}$ and $b_i' = (b_i - \sum_{j=1}^{i-1} L_{i,j}y_j)/L_{i,i}$, for $i = 1, \cdots, p$. Taking into account the fact that

$$\Big[1+\frac{\sum_{i=1}^p y_i^2}{\nu}\Big]^{-\frac{\nu+p}{2}} = \Big[1+\frac{y_1^2}{\nu}\Big]^{-\frac{\nu+p}{2}}\Big[1+\frac{y_2^2}{\nu+y_1^2}\Big]^{-\frac{\nu+p}{2}}\cdots\Big[1+\frac{y_p^2}{\nu+\sum_{i=1}^{p-1} y_i^2}\Big]^{-\frac{\nu+p}{2}},$$

it follows that

$$\mathcal{P}_t = \mathrm{C}_t(\nu,p)\int_{\boldsymbol{a}'}^{\boldsymbol{b}'}\Big[1+\frac{y_1^2}{\nu}\Big]^{-\frac{\nu+p}{2}}\Big[1+\frac{y_2^2}{\nu+y_1^2}\Big]^{-\frac{\nu+p}{2}}\cdots\Big[1+\frac{y_p^2}{\nu+\sum_{i=1}^{p-1} y_i^2}\Big]^{-\frac{\nu+p}{2}} d\boldsymbol{y}.$$

Applying, for $i = 1, \cdots, p$, a change of variable of the form

$$u_i = y_i\frac{\sqrt{\nu+i-1}}{\sqrt{v+\sum_{j=1}^{i-1} y_j^2}} \tag{3.34}$$

for which we have

$$dy_i = du_i\frac{\sqrt{v+\sum_{j=1}^{i-1} y_j^2}}{\sqrt{\nu+i-1}}$$

Hence,

$$\begin{aligned}\mathcal{P}_t &= \mathrm{C}_t(\nu,p)\int_{\boldsymbol{a}''}^{\boldsymbol{b}''}\Big[1+\frac{u_1^2}{\nu}\Big]^{-\frac{\nu+p}{2}}\Big[1+\frac{u_2^2}{\nu+1}\Big]^{-\frac{\nu+p}{2}}\cdots\Big[1+\frac{u_p^2}{\nu+p-1}\Big]^{-\frac{\nu+p}{2}}d\boldsymbol{y}\\ &= \mathrm{C}_t(\nu,p)\prod_{i=1}^{p}\int_{a_i''}^{b_i''}\Big[1+\frac{u_i^2}{\nu+i-1}\Big]^{-\frac{\nu+p}{2}}\times\prod_{i=1}^{p}du_i\frac{\sqrt{v+\sum_{j=1}^{i-1}y_j^2}}{\sqrt{\nu+i-1}}\end{aligned}\tag{3.35}$$

where by convention $\sum_{j=1}^{0}x_j^2=0$ and

$$a_i''=a_i'\frac{\sqrt{\nu+i-1}}{\sqrt{v+\sum_{j=1}^{i-1}y_j^2}}=\frac{a_i-\sum_{j=1}^{i-1}L_{i,j}y_j}{L_{i,i}}\frac{\sqrt{\nu+i-1}}{\sqrt{v+\sum_{j=1}^{i-1}y_j^2}}\tag{3.36}$$

$$b_i''=b_i'\frac{\sqrt{\nu+i-1}}{\sqrt{v+\sum_{j=1}^{i-1}y_j^2}}=\frac{b_i-\sum_{j=1}^{i-1}L_{i,j}y_j}{L_{i,i}}\frac{\sqrt{\nu+i-1}}{\sqrt{v+\sum_{j=1}^{i-1}y_j^2}}.\tag{3.37}$$

For a given integer $1\le i\le p$, let $h(i)=v+\sum_{j=1}^{i}y_j^2$. Evidently $h(0)=\nu$ and it is easy to check, based on (3.35), that the recurrence relation

$$h(i-1)=h(i-2)\Big[1+\frac{u_{i-1}^2}{\nu+i-2}\Big],i=2,\cdots,p,$$

holds true and therefore

$$\begin{aligned}\prod_{i=1}^{p}du_i\frac{\sqrt{h(i-1)}}{\sqrt{\nu+i-1}} &= du_1\times\prod_{i=2}^{p}du_i\frac{\sqrt{h(i-1)}}{\sqrt{\nu+i-1}}\\ &= du_1\times\prod_{i=1}^{p-1}du_{i+1}\frac{\sqrt{\nu}}{\sqrt{\nu+i}}\Big[1+\frac{u_i^2}{\nu+i-1}\Big]^{\frac{p-i}{2}}.\end{aligned}\tag{3.38}$$

Substituting the RHS of (3.38) into the RHS of (3.35) yields

$$\begin{aligned}\mathcal{P}_t =&\mathrm{C}_t(\nu,p)\times\prod_{i=1}^{p-1}\frac{\sqrt{\nu}}{\sqrt{\nu+i}}\times\int_{a_1''}^{b_1''}\Big[1+\frac{u_1^2}{\nu}\Big]^{-\frac{\nu+1}{2}}\times\int_{a_2''}^{b_2''}\Big[1+\frac{u_2^2}{\nu+1}\Big]^{-\frac{\nu+2}{2}}\\ &\times\cdots\times\int_{a_p''}^{b_p''}\Big[1+\frac{u_p^2}{\nu+p-1}\Big]^{-\frac{\nu+p}{2}}du_p\cdots du_2du_1\\ =&\mathrm{C}_t(\nu,p)\times\prod_{i=0}^{p-1}\frac{\sqrt{\nu}}{\sqrt{\nu+i}\times\mathrm{C}_t(\nu+i,1)}\times\int_{a_1''}^{b_1''}t_\nu(u_1)\times\int_{a_2''}^{b_2''}t_{\nu+1}(u_2)\\ &\times\cdots\times\int_{a_p''}^{b_p''}t_{\nu+p-1}(u_p)du_p\cdots du_2du_1.\end{aligned}$$

Moreover, it is easy to check that

$$\begin{aligned}\mathrm{C}_t(\nu,p)\times\prod_{i=0}^{p-1}\frac{\sqrt{\nu}}{\sqrt{\nu+i}\times\mathrm{C}_t(\nu+i,1)} &= \frac{\Gamma\big(\frac{\nu+p}{2}\big)}{(\nu\pi)^{\frac{p}{2}}\Gamma\big(\frac{\nu}{2}\big)}\times\prod_{i=0}^{p-1}\frac{\sqrt{\nu}}{\sqrt{\nu+i}}\frac{[(\nu+i)\pi]^{\frac{1}{2}}\Gamma\big(\frac{\nu+i}{2}\big)}{\Gamma\big(\frac{\nu+i+1}{2}\big)}\\ &= \frac{\Gamma\big(\frac{\nu+p}{2}\big)}{\Gamma\big(\frac{\nu}{2}\big)}\times\prod_{i=0}^{p-1}\frac{\Gamma\big(\frac{\nu+i}{2}\big)}{\Gamma\big(\frac{\nu+i+1}{2}\big)}\\ &= 1.\end{aligned}$$

Hence,

$$\mathcal{P}_t=\int_{a_1''}^{b_1''}t_\nu(u_1)\times\int_{a_2''}^{b_2''}t_{\nu+1}(u_2)\times\cdots\times\int_{a_p''}^{b_p''}t_{\nu+p-1}(u_p)du_p\cdots du_2du_1.$$

Using transformation $z_i = T_{\nu+i-1}(u_i)$ in which u_i (for $i = 1, \cdots, p$) is given by (3.34), it follows that

$$\mathcal{P}_t = \int_{d_1}^{e_1} \int_{d_2}^{e_2} \cdots \int_{d_p}^{e_p} dz_p \cdots dz_1, \tag{3.39}$$

where $d_i = T_{\nu+i-1}(a_i'')$ and $e_i = T_{\nu+i-1}(b_i'')$ in which constants a_i'' and b_i'' are given by (3.36) and (3.37), respectively. For computational purposes we may apply another change of variable $w_i = (z_i - d_i) / (e_i - d_i)$ by which $\mathcal{P}_t$ in (3.39) would be computed within a $(p-1)$-dimensional unit hypercube $[0,1]^{p-1}$. We have

$$\begin{aligned} \mathcal{P}_t &= \int_0^1 (e_1 - d_1)\, dw_1 \cdots \int_0^1 (e_{p-1} - d_{p-1})\, dw_{p-1} \int_0^1 (e_p - d_p)\, dw_p \\ &= (e_1 - d_1) \int_0^1 (e_2 - d_2)\, dw_1 \cdots \int_0^1 (e_p - d_p)\, dw_{p-1}. \end{aligned} \tag{3.40}$$

Accomplishing $\mathcal{P}_t$ in (3.40) needs computing quantities $\{d_1, \cdots, d_p\}$ and $\{e_1, \cdots, e_p\}$ at each iteration of the Monte Carlo computation. By (3.36) and (3.37), the initial values d_1 and e_1 are $d_1 = T_\nu(a_1/L_{1,1})$ and $e_1 = T_\nu(b_1/L_{1,1})$. The reminders are recovered using a recurrence relation. Algorithm 11 given below explains how to compute the CDF of a Student's t distribution. In the following, the `R` function `pmvt_Genz` is given for implementing the algorithm proposed by Genz and Bretz [1999].

```
R> pmvt_Genz <- function(a, b, Mu, Sigma, nu, epsilon = 10e-4, N_max = 200)
 + {
 + p <- length(Mu)
 + L <- t( chol(Sigma) )
 + alpha <- 2.5
 + s <- rep(0, N_max)
 + a0 <- b0 <- d <- e <- f <- w <- y <- u <- rep(0, p)
 + a0 <- a- Mu
 + b0 <- b- Mu
 + nu <- ifelse(nu >= 1000, 1000, nu)
 + d[1] <- pt( q = a0[1]/L[1, 1], df = nu )
 + e[1] <- pt( q = b0[1]/L[1, 1], df = nu )
 + f[1] <- e[1]- d[1]
 + se <- 10
 + N <- 0
 + while( se > epsilon | N < N_max )
 + {
 +   w <- runif(p- 1)
 +   u[1] <- qt( p = d[1] + w[1]*(e[1]- d[1]), df = nu )
 +   y[1] <- u[1]
 +   ss_y <- 0;
 +   ss_y <- ss_y + y[1]^2
 +   i <- 2
 +    while( i <= p )
 +    {
 +     j <- i - 1
 +     L_nu <- sqrt( (nu + j)/(nu + ss_y) )
 +     L_y <- sum( L[i, 1:j]*y[1:j] )
 +     a0_second <- (a0[i]- L_y )/L[i, i]*L_nu
 +     b0_second <- (b0[i]- L_y )/L[i, i]*L_nu
 +     d[i] <- pt( q = a0_second, df = nu + j )
 +     e[i] <- pt( q = b0_second, df = nu + j )
 +     d[i] <- ifelse( d[i] == 0, .Machine$double.xmin, d[i] )
 +     e[i] <- ifelse( e[i] == 1, 0.9999999, e[i] )
 +     f[i] <- (e[i] - d[i])*f[j]
 +     u[i] <- qt( p = d[i] + w[i]*(e[i] - d[i]), df = nu + j )
 +     y[i] <- u[i]/L_nu
```

Algorithm 11 Computing CDF of the Multivariate Student's t distribution

Read $\boldsymbol{a}$, $\boldsymbol{b}$, $\boldsymbol{\mu}$, Σ, ν, ϵ, and N_{max};
Compute the Cholesky decomposition L of Σ;
Set s to be a vector of zeros of length N_{max} and $\alpha = 2.5$;
Set $\boldsymbol{b} = \boldsymbol{b} - \boldsymbol{\mu}$ and $\boldsymbol{a} = \boldsymbol{a} - \boldsymbol{\mu}$;
Set $e_1 = T_\nu(b_1/L_{1,1})$, $d_1 = T_\nu(a_1/L_{1,1})$, and $f_1 = e_1 - d_1$;
Set $N = 0$ and $\mathrm{se} = 10$;
while $\mathrm{se} > \epsilon$ and $N < \mathrm{N}_{max}$ **do**
    Generate realizations $w_1, w_2, \cdots, w_{p-1}$ from $\mathcal{U}(0,1)$;
    Set $u_1 = T_\nu^{-1}(d_1 + w_1 f_1)$, $u_1 = y_1$, and $\mathrm{ssy} = 0$;
    Set $\mathrm{ssy} = \mathrm{ssy} + y_1^2$;
    Set $i = 2$;
    while $i \leq p$ **do**
        Set $j = i - 1$;
        $L_\nu = (\nu + j)^{\frac{1}{2}}(\nu + \sum_{k=1}^{j} y_k^2)^{-\frac{1}{2}}$;
        $a_i'' = \frac{a_i - \sum_{k=1}^{j} y_k \times L_{i,k}}{L_{i,i}} \times L_\nu$;
        $b_i'' = \frac{b_i - \sum_{k=1}^{j} y_k \times L_{i,k}}{L_{i,i}} \times L_\nu$;
        $d_i = T_{\nu+j}(a_i'')$;
        $e_i = T_{\nu+j}(b_i'')$;
        $f_i = (e_i - d_i) f_j$;
        $u_i = T_{\nu+j}^{-1}[d_i + w_i(e_i - d_i)]$;
        $y_i = u_i / L_\nu$;
        $\mathrm{ssy} = \mathrm{ssy} + y_i^2$;
        Set $i = i + 1$;
    end
    $N = N + 1$;
    $s[N] = f_p$ where $s[N]$ denotes the Nth element of vector s;
    $\sigma = \sqrt{\mathrm{Var}(s[1:N])}$;
    $\mathrm{se} = \alpha \frac{\sigma}{\sqrt{N}}$;
end
Return $\sum_{r=1}^{N} s[r]/N$ as an estimator for the CDF of multivariate Student's t distribution with standard error se.

```
 +      ss_y <- ss_y + y[i]^2
 +      i <- i + 1
 +    }
+   N <- N + 1
+   s[N] <- f[p]
+   sigma <- sqrt( var( s[1:N] ) )
+   se <- alpha*sigma/sqrt(N)
+   se <- ifelse( is.na(se), 10, se )
+ }
+ ls <- list("cdf" = mean(s), "se" = se)
+ return(ls)
+}
```

3.6 Univariate Gaussian quadrature

The *quadrature rules* involve methods for approximating an integral

$$I = \int_a^b f(x)dx \approx \mathcal{Q}(f) = \sum_{i=1}^{k} \omega_i f(x_i) \tag{3.41}$$

where quantities ω_i and x_i are the ith *weight* and *node* of the quadrature rule, respectively. The idea behind a quadrature rule is to approximate the integrand $f(x)$ through a simple function. Reasonably, an appropriate choice is a polynomial $P(x)$, of degree not greater than $k-1$ that interpolate $f(x)$ at x_i (for $i = 1, \cdot, k$). Suppose, for a given function $f(x)$, we are interested in approximating the integral of the form $f(x)$ over interval (a, b) using a two-point ($k = 2$) quadrature rule. We have

$$\int_a^b f(x)dx \approx \omega_1 f(x_1) + \omega_2 f(x_2), \tag{3.42}$$

where the value of above integral is known when all four unknown constants including ω_1 (first weight), ω_2 (second weight), x_1 (first node), and x_2 (second node) are determined. To this end, we assume $f(x)$ is a polynomial of degree three represented as $f(x) = a_0 + a_1x + a_2x^2 + a_3x^3$. Although we can let for larger values of m, but we set $m = 2$ for simplicity. So, substituting the third-order polynomial $f(x) = a_0 + a_1x + a_2x^2 + a_3x^3$ into the LHS of (3.42), it follows that

$$\begin{aligned}
\int_a^b f(x)\ dx &= \int_a^b a_0 + a_1x + a_2x^2 + a_3x^3 dx \\
&= \Big[a_0x + a_1\tfrac{x^2}{2} + a_2\tfrac{x^3}{3} + a_3\tfrac{x^4}{4}\Big]_a^b \\
&= a_0(b-a) + a_1\Big(\frac{b^2-a^2}{2}\Big) + a_2\Big(\frac{b^3-a^3}{3}\Big) + a_3\Big(\frac{b^4-a^4}{4}\Big).
\end{aligned} \tag{3.43}$$

Representing the RHS of (3.42) in terms of third-order polynomial yields

$$\omega_1 f(x_1) + \omega_2 f(x_2) = \omega_1(a_0 + a_1x_1 + a_2x_1^2 + a_3x_1^3) + \omega_2(a_0 + a_1x_2 + a_2x_2^2 + a_3x_2^3). \tag{3.44}$$

Equating the RHSs of (3.43) and (3.44) yields a set of four equations as follows.

$$\begin{cases}
b - a = \omega_1 + \omega_2, \\
\frac{b^2-a^2}{2} = \omega_1x_1 + \omega_2x_2, \\
\frac{b^3-a^3}{3} = \omega_1{x_1}^2 + \omega_2{x_2}^2, \\
\frac{b^4-a^4}{4} = \omega_1{x_1}^3 + \omega_2{x_2}^3.
\end{cases} \tag{3.45}$$

Solving the set of four equations given in (3.45), we have

$$\begin{aligned}
\omega_1 &= \frac{b-a}{2}, \\
\omega_2 &= \frac{b-a}{2}, \\
x_1 &= -\frac{1}{\sqrt{3}}\Big(\frac{b-a}{2}\Big) + \frac{b+a}{2}, \\
x_2 &= \frac{1}{\sqrt{3}}\Big(\frac{b-a}{2}\Big) + \frac{b+a}{2}.
\end{aligned} \tag{3.46}$$

Substitute the RHS of (3.46) into the RHS of (3.42) to see that

$$\begin{aligned}
\int_a^b f(x)dx &\approx \omega_1 f(x_1) + \omega_2 f(x_2) \\
&= \frac{b-a}{2} f\Big(-\frac{1}{\sqrt{3}}\frac{b-a}{2} + \frac{b+a}{2}\Big) + \frac{b-a}{2} f\Big(\frac{1}{\sqrt{3}}\frac{b-a}{2} + \frac{b+a}{2}\Big).
\end{aligned} \tag{3.47}$$

Following example shows how much a quadrature can be reliable.

Example 3-7

Let $f(x) = x\exp\{-x\}$ and we would like to approximate

$$I = \int_0^3 x\exp\{-x\}dx, \tag{3.48}$$

based on Gaussian quadrature of $m = 2$ points. We compute two weights and nodes through (3.45). It is easy to check that

$$I = \approx 1.5 \times f(0.63397) + 1.5 \times f(2.36602) = 0.83755, \tag{3.49}$$

where the exact value of above integral is 0.80085. Figure 3.4 displays the integrand and the corresponding nodes. ■

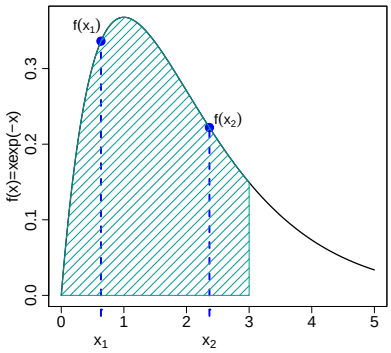


Figure 3.4: Plot of integrand $f(x) = x\exp\{-x\}$ and nodes x_1 and x_2. The area under the integrand is shown by shaded area.

The above reasoning can be extended for $k > 2$ that yields more accurate approximation of integral (3.42). In general, using the quadrature as described above, one can approximate the given definite integral using a quadrature rule with an arbitrary k numbers of nodes and weights. In general, we can write

$$I = \int_a^b f(x)dx = \int_a^b P(x)dx = \int_a^b \sum_{i=1}^k L_i(x)f(x_i)dx = \sum_{i=1}^k \omega_i f(x_i),$$

where

$$\omega_i = \int_a^b L_i(x)dx,$$

in which $L_i(x)$ is the well-known Lagrange interpolating polynomial of order $k-1$. defined as

$$L_i(x) = \prod_{j=1, j\neq i}^{n} \frac{x - x_i}{x_j - x_i}.$$

In what follows, among several quadrature rules introduced in the literature, we briefly describe the *Gaussian quadrature* rules for computing a definite integral.

3.6.1 Gaussian quadrature rules

The quadrature rule presented in the previous sub-section works based on a general polynomial. Herein, we describe the well-known *Gaussian quadrature* that works based on a special class of polynomials known as the *orthogonal polynomials* and can be applied for approximating accurately a wide range of definite integrals [Dahlquist and Björck, 2003]. For presentation and theoretical purposes, the following definitions are useful.

Definition 3.6.1. *[Foupouagnigni and Koepf, 2018] The function $w(\cdot) : (a, b) \to \mathbb{R}^+$ that satisfies the following three conditions is said to be a weight function.*

i. $w(x)$ is measurable.

ii. The kth moment of $w(x)$ must be finite, that is $\int_a^b |x|^k w(x)dx < \infty$ for $k = 0, 1, 2, \cdots$.

iii. For a given polynomial such as $g(x)$ if $\int_a^b g(x)w(x)dx = 0$ then we must have $g(x) = 0$.

Definition 3.6.2. *[Foupouagnigni and Koepf, 2018] The sequence of polynomials $\{f_0(x), f_1(x), f_2(x), \cdots\}$ constitutes a sequence of orthogonal polynomials with respect to weight function $w(\cdot) : (a, b) \to \mathbb{R}^+$ if we have*

$$\langle f_n, f_m \rangle = \int_a^b f_n(x) f_m(x) w(x)dx = C_n \times \mathcal{D}_{m,n},$$

where $\langle ., . \rangle$ represents the inner product operator, C_n is a constant, and $\mathcal{D}_{m,n}$ is the Kronecker delta defined as

$$\mathcal{D}_{m,n} = \begin{cases} 1, & \text{if } m = n, \\ 0, & \text{if } m \neq n. \end{cases} \tag{3.50}$$

There are several types of the orthogonal polynomials such as *Chebyshev of first kind* ($T_{1,n}$), *Chebyshev of second kind* ($T_{2,n}$), *generalized Laguerre* ($\mathcal{GL}_n^{\alpha}$), *Hermite* (H_n), *Jacobi* ($J_n^{\alpha,\beta}$), *Laguerre* ($\mathcal{L}_n$), and *Legendre* (L_n), to name a few. In general, the orthogonal polynomials are solution of differential equation and obtained through a recurrence formula. In what follows, we describe briefly the *Hermite* and *Laguerre* classes of orthogonal polynomials with widespread applications in statistics.

- **Hermite polynomial of first kind**: the Hermite polynomial of degree n, shown by $H_{1,n}(x)$, is orthogonal with respect to weight function $w(x) = \exp\{-x^2\}$ over the range $(-\infty, \infty)$. This means that

$$\langle H_{1,n}, H_{1,m} \rangle = \int_{-\infty}^{\infty} H_{1,n}(x) H_{1,m}(x) \exp\{-x^2\}dx = 2^n \sqrt{\pi}\Gamma(n+1) \times \mathcal{D}_{m,n}. \tag{3.51}$$

The expansion representation for $H_{1,n}(x)$ is

$$H_{1,n}(x) = \Gamma(n+1) \sum_{i=0}^{\lfloor \frac{n}{2} \rfloor} \frac{(-1)^i (2x)^{n-2i}}{\Gamma(i+1)\Gamma(n-2i+1)}, \tag{3.52}$$

where $\lfloor n/2 \rfloor$ denotes the greatest integer less or equal $n/2$ (for $n = 0, 1, 2, \cdots$). This class of orthogonal polynomials satisfies with the recurrence relations

$$\frac{dH_{1,n}(x)}{dx} - 2xH_{1,n}(x) + H_{1,n+1}(x) = 0,$$

for $n \geq 0$ and

$$H_{1,n+1}(x) - 2xH_{1,n}(x) + 2nH_{1,n-1}(x) = 0, \tag{3.53}$$

for $n \geq 1$. Using expression (3.53), the first three members of Hermite polynomials are as follows [Arfken et al., 2011].

$$\begin{aligned} H_{1,0}(x) &= 1, \\ H_{1,1}(x) &= 2x, \\ H_{1,2}(x) &= 4x^2 - 2, \end{aligned}$$

Figure 3.5 displays the graph of Hermite (first kind) and Legendre polynomials of orders

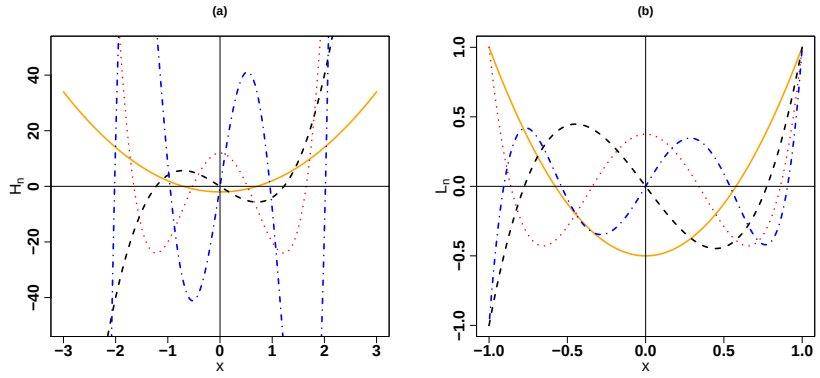


Figure 3.5: Graph of (a): Hermite and (b): Legendre polynomials of orders $n = 2$ (solid orange line), $n = 3$ (dashed black line), $n = 4$ (dotted red line), and $n = 5$ (dotted-dashed blue line).

$n = 2, 3, 4, 5$.

- **Hermite polynomial of second kind**: the Hermite polynomial of second kind, shown by $H_{2,n}(x)$, is orthogonal with respect to weight function $w(x) = \exp\{-x^2\}$ over the range $(0, b)$ where b is a real positive value. This means that

$$\langle H_{2,n}, H_{2,m} \rangle = \int_0^b H_{2,n}(x) H_{2,m}(x) \exp\{-x^2\} dx = \mu_n \times \mathcal{D}_{m,n}, \tag{3.54}$$

where μ_n, for $n \geq 1$, meets the recurrence relation

$$2\mu_n = \mu_{n-1} - \Big[\exp\{-x^2\} H_{2,n}(x) H_{2,n-1}(x) \Big]_0^b, \tag{3.55}$$

with

$$\mu_0 = \frac{\sqrt{\pi}}{2} \text{erf}(b), \tag{3.56}$$

in which $\text{erf}(\cdot)$ is the well-known *error function* that can be expressed in terms of the standard Gaussian CDF as

$$\text{erf}(b) = \frac{2}{\sqrt{\pi}} \int_0^b \exp\{-u^2\} du = 2\Phi(\sqrt{2}b) - 1. \tag{3.57}$$

This class of orthogonal polynomials satisfies, for $n \geq 1$, with the recurrence relation

$$xH_{2,n}(x) - H_{2,n+1}(x) + \alpha_n H_{2,n}(x) + \beta_n H_{2,n-1}(x) = 0, \tag{3.58}$$

where three first polynomials are

$$\begin{aligned} H_{2,0}(x) =& 1, \\ H_{2,1}(x) =& x - \eta(b), \\ H_{2,2}(x) =& \Big[x - \frac{\exp\{-b^2\}\big[b - \eta(b)\big]^2 - \eta^2(b)}{\beta_1 \sqrt{\pi}\ \text{erf}(b)}\Big]\big[x - \eta(b)\big] + \beta_1, \end{aligned} \tag{3.59}$$

where

$$\beta_1 = -\frac{1}{2} + \frac{b}{\sqrt{\pi}\ \text{erf}(b)} - b\eta(b) + \eta^2(b) \tag{3.60}$$

in which

$$\eta(b) = \frac{1 - \exp\{-b^2\}}{\sqrt{\pi}\ \text{erf}(b)}. \tag{3.61}$$

Details for determining coefficients α_n and β_n in (3.58) will be discussed later.

- **Hermite polynomial of third kind**: the Hermite polynomial of third kind $H_{3,n}(x)$, is orthogonal with respect to weight function $w(x) = \exp\{-x^2\}$ over the range $(0, \infty)$. This means that

$$\langle H_{3,n}, H_{3,m} \rangle = \int_0^\infty H_{3,n}(x) H_{3,m}(x) \exp\{-x^2\} dx = \mu_n \times \mathcal{D}_{m,n}, \tag{3.62}$$

where μ_n, for $n \geq 1$, meets the recurrence relation

$$2\mu_n = n\mu_{n-1} + H_{3,n}(0) H_{3,n-1}(0), \tag{3.63}$$

with

$$\mu_0 = \frac{\sqrt{\pi}}{2}. \tag{3.64}$$

The class $H_{3,n}(x)$ satisfies, for $n \geq 1$, with the recurrence relation

$$xH_{3,n}(x) - H_{3,n+1}(x) + \alpha_n H_{3,n}(x) + \beta_n H_{3,n-1}(x) = 0, \tag{3.65}$$

where three first polynomials are

$$\begin{aligned} H_{3,0}(x) =& 1, \\ H_{3,1}(x) =& x - \frac{1}{\sqrt{\pi}}, \\ H_{3,2}(x) =& \Big[x + \frac{2}{\sqrt{\pi}(2 - \pi)}\Big]\big[x - \frac{1}{\sqrt{\pi}}\big] + \frac{1}{\pi} - \frac{1}{2}. \end{aligned} \tag{3.66}$$

Details for determining coefficients α_n and β_n in (3.65) will be discussed later.

- **Laguerre polynomial:** The Laguerre polynomial of degree n, shown by $\mathcal{L}_n(x)$, is orthogonal with respect to weight function $w(x) = \exp\{-x\}$ over the range $(0, \infty)$. This means that

$$\langle \mathcal{L}_n, \mathcal{L}_m \rangle = \int_0^\infty \mathcal{L}_n(x)\mathcal{L}_m(x) \exp\{-x\}dx = \mathcal{D}_{m,n}. \tag{3.67}$$

This class of orthogonal polynomials is solution of the differential equation

$$x\frac{d^2\mathcal{L}_n(x)}{dx^2} + (1-x)\frac{d\mathcal{L}_n(x)}{dx} + n\mathcal{L}_n(x) = 0, \tag{3.68}$$

whose solution becomes

$$\mathcal{L}_n(x) = \Gamma(n+1)\sum_{i=0}^{n} \frac{(-1)^i x^i}{\Gamma(n-i+1)[\Gamma(i+1)]^2}. \tag{3.69}$$

Using expression (3.69), the first three members of Laguerre polynomials are as follows.

$$\begin{aligned} \mathcal{L}_0(x) &= 1, \\ \mathcal{L}_1(x) &= 1 - x, \\ \mathcal{L}_2(x) &= \frac{x^2 - 4x + 2}{2}. \end{aligned}$$

Table 3.3 shows a list of most commonly used orthogonal polynomials in statistics. There

are different types of the Gaussian quadrature rules have been developed in the literature for approximating the integral (3.70). Particular Gauss quadrature rule arises from particular choice of the interval and the form of the weight function $w(x)$. For example, the Gauss-Legendre quadrature rule is applied when $[a, b] = [-1, 1]$ and $w(x) = 1$, and so is employed for approximating integral of the form

$$\int_{-1}^{1} f(x)dx.$$

Moreover, for approximating integral of the form

$$\int_0^\infty \exp\{-x\}f(x)dx,$$

we may use the Gauss-Laguerre quadrature rule since as it is seen the range of integral is $[a, b] = [0, \infty)$ and $w(x) = \exp\{-x\}$, and finally, a k-point Gauss-Chebyshev rule is useful when $[a, b] = [-1, 1]$. The bounds of integral (3.70) can be changed from (a, b) to $[-1, 1]$ through a simple transformation when $a \neq -1$ and $b \neq 1$. For integrand of the form $w(x)f(x)$, in which $w(x)$ is given in Table 3.3, an orthogonal polynomial up to degree $k + 1$ is fitted to integrand $w(x)f(x)$ that intercepts integrand $w(x)f(x)$ at k nodes $\{x_n\}_{n=1}^k$. Once we have found k pairs of nodes and associated weights, shown by $\{(x_n, \omega_n)\}_{n=1}^k$, the integral can be approximated based on a k-point Gaussian quadrature rule as

$$\int_a^b w(x)f(x)dx \approx \sum_{n=1}^{k} \omega_n f(x_n). \tag{3.70}$$

The nodes and weighs for Gauss-Chebyshev quadrature are determined simply. The class $T_{1,n}(\cdot)$ and $T_{2,n}(\cdot)$ of orthogonal polynomials are given by

$$\begin{cases} T_{1,n}(x) = \cos\big(n\theta\big), & \text{for first kind,} \\ T_{2,n}(x) = \frac{\sin\big((n+1)\theta\big)}{\sin(\theta)}, & \text{for second kind,} \end{cases} \tag{3.71}$$

Table 3.3: Most commonly used orthogonal polynomials.

$P_n(\cdot)$	symbol	C_n	$[a,b]$	$w(x)$
Chebyshev (first kind)	$T_{1,n}$	$\begin{cases}\pi & \text{if } n=0,\\ \frac{\pi}{2} & \text{if } n\neq 0.\end{cases}$	$(-1,1)$	$(1-x^2)^{-\frac{1}{2}}$
Chebyshev (second kind)	$T_{2,n}$	$\frac{\pi}{2}$	$[-1,1]$	$(1-x^2)^{\frac{1}{2}}$
Generalized Laguerre	$\mathcal{L}_n^{\alpha}$	$\frac{\Gamma(n+\alpha+1)}{\Gamma(n+1)}$	$[0,\infty)$	$x^{\alpha}\exp\{-x\}$, $\alpha>-1$
Hermite (first kind)	$H_{1,n}$	$2^n\sqrt{\pi}\Gamma(n+1)$	$(-\infty,\infty)$	$\exp\{-x^2\}$
Hermite (second kind)	$H_{2,n}$	no closed form, see (3.54)	$(0,b)$	$\exp\{-x^2\}$
Hermite (third kind)	$H_{3,n}$	no closed form, see (3.62)	$(0,\infty)$	$\exp\{-x^2\}$
Jacobi	$J_n^{\alpha,\beta}$	$\frac{2^{\alpha+\beta+1}}{2n+\alpha+\beta+1}\times\frac{\Gamma(n+\alpha+1)\Gamma(n+\beta+1)}{\Gamma(n+\alpha+\beta+1)\Gamma(n+1)}$	$(-1,1)$	$(1-x)^{\alpha}(1+x)^{\beta}$, $\alpha,\beta>-1$
Laguerre	$\mathcal{L}_n$	1	$[0,\infty)$	$\exp\{-x\}$
Legendre	L_n	$\frac{2}{2n+1}$	$[-1,1]$	1

where $x=\cos(\theta)$ for $\theta\in[0,\pi]$ [Rivlin, 2020]. These two classes of orthogonal polynomials admit the recurrence relation

$$T_{1,n+1}(x)-2xT_{1,n}(x)+T_{1,n-1}(x)=0,\ \text{for } n\geq 1, \tag{3.72}$$

where $T_{1,0}(x)=1$ and $T_{1,1}(x)=x$. The nodes and weights of the Gauss-Chebyshev quadrature, for $n=1,\cdots,k$, are given by

$$\begin{cases} x_n=\cos\big(\frac{2n-1}{2k}\pi\big), & \text{for first kind,}\\ x_n=\cos\big(\frac{n}{k+1}\pi\big), & \text{for second kind,}\end{cases} \tag{3.73}$$

and

$$\begin{cases} \omega_n=\frac{\pi}{2}, & \text{for first kind,}\\ \omega_n=\frac{\pi}{k+1}\sin^2\big(\frac{n}{k+1}\pi\big), & \text{for second kind,}\end{cases} \tag{3.74}$$

respectively.

3.6.2 Golub-Welsch algorithm

The fundamental Gaussian quadrature theorem states that the nodes x_is are the roots of the orthogonal polynomial. It can be shown that each orthogonal polynomial of degree k has distinct real roots on interval $[a,b]$ [Shen et al., 2011, pp. 53]. There are two simple methods for computing the roots of orthogonal polynomial [Shen et al., 2011, pp. 55]. Except for

the Chebyshev case, the roots of other orthogonal polynomials may be obtained trough the *eigenvalue* or *iterative* method. In what follows, we describe briefly the *eigenvalue method* proposed by Golub and Welsch [1969].

- **nodes**: Let $P_n(x)$ denote the orthogonal polynomial. It can be seen that the orthogonal polynomials given on Table 3.3 admits the recurrence formula as [Shen et al., 2011, Theorem 3.1]:

$$xP_n(x) = a_nP_{n+1}(x) + b_nP_n(x) + c_nP_{n-1}(x), \tag{3.75}$$

for $c_0P_{-1}(x) = 0$. The zeros of the orthogonal polynomial $P_n(x)$ are eigenvalues of symmetric threediagonal matrix A given by [Gil et al., 2007]:

$$A = \begin{pmatrix} b_0 & \sqrt{a_0c_1} & & & \\ \sqrt{a_0c_1} & b_1 & \sqrt{a_1c_2} & & \\ & \sqrt{a_1c_2} & b_2 & & \\ & & & \ddots & \sqrt{a_{k-2}c_{k-1}} \\ & & & \sqrt{a_{k-2}c_{k-1}} & b_{k-1} \end{pmatrix}. \tag{3.76}$$

The eigenvalues $e_1, \cdots, e_k$ of matrix A are also nodes x_n, of the Gaussian quadrature rule, that is $e_n = x_n$, for $n = 1, \cdots, k$.

- **weight**: Once we have computed the eigenvalues of A in (3.76), the nth eight ω_n is obtained as

$$\omega_n = \mu_0 \frac{\left(\boldsymbol{v}_n[1]\right)^2}{\|\boldsymbol{v}_n\|}, \tag{3.77}$$

in which

$$\mu_0 = \int_a^b w(x)dx,$$

$\|\cdot\|$ denotes the Euclidean norm, $\{\boldsymbol{v}_1, \cdots, \boldsymbol{v}_k\}$ are the eigenvectors correspond to $\{e_1, \cdots, e_k\}$, and $\boldsymbol{v}_n[1]$ is the first element of eigenvector $\boldsymbol{v}_n$.

In what follows, we compute the Gaussian quadrature rule for orthogonal polynomials presented in Table 3.3.

- **nodes and weighs for generalized Laguerre polynomial**: The class $\mathcal{L}_n^\alpha(\cdot)$ of orthogonal polynomials admits the recurrence relation

$$(n+1)\mathcal{L}_{n+1}^\alpha(x) = (2n+\alpha+1-x)\mathcal{L}_n^\alpha(x) - (n+\alpha)\mathcal{L}_{n-1}^\alpha(x), \tag{3.78}$$

where $\mathcal{L}_0^\alpha(x) = 1$. Comparing the recurrence relations given in (3.75) and (3.78), we obtain $b_n = 0$, for $n = 0, 1, 2, \cdots$, and

$$\begin{cases} a_n = -(n+1) \text{ and } b_n = (2n+\alpha+1), & \text{if } n \geq 0, \\ c_n = -(n+\alpha), & \text{if } n \geq 1. \end{cases} \tag{3.79}$$

In this case, matrix A in (3.76) can be reconstructed by using the constants a_n, b_n, and c_n provided in (3.79) for computing eigenvalues of the reconstructed matrix A as the nodes. For computing weights, it follows from Table 3.3 that

$$\mu_0 = \int_a^b w(x)dx = \int_0^\infty x^\alpha \exp\{-x\}dx = \Gamma(\alpha+1).$$

Hence the weights ω_n (for $n = 1, \cdots, k$) are obtained by considering $\mu_0 = \Gamma(\alpha+1)$ in the RHS of (3.77) in which $\boldsymbol{v}_n$ (for $n = 1, \cdots, k$) is the eigenvector corresponds to ith eigenvalue of matrix A constructed based on information given by (3.79).

- **nodes and weighs for Hermite polynomial of first kind**: The class $H_{1,n}(\cdot)$ of orthogonal polynomials admits the recurrence relation given by (3.53) for $n \geq 1$. Comparing the recurrence relations given in (3.75) and (3.53), we obtain

$$\begin{cases} a_n = \frac{1}{2} \text{ and } b_n = 0, & \text{if } n \geq 0, \\ c_n = n, & \text{if } n \geq 1. \end{cases} \tag{3.80}$$

In this case, matrix A in (3.76) can be reconstructed by using the constants a_n, b_n, and c_n provided in (3.80) for computing eigenvalues of the reconstructed matrix A as the nodes. For computing weights, it follows from Table 3.3 that

$$\mu_0 = \int_a^b w(x)dx = \int_{-\infty}^{\infty} \exp\{-x^2\}dx = \sqrt{2\pi}.$$

Hence the weights ω_n (for $n = 1, \cdots, k$) are obtained by considering $\mu_0 = \sqrt{\pi}$ in the RHS of (3.77) in which $\boldsymbol{v}_n$ (for $n = 1, \cdots, k$) is the eigenvector corresponds to ith eigenvalue of matrix A constructed based on information given by (3.80).

- **nodes and weighs for Hermite polynomial of second kind**: The class $H_{2,n}(\cdot)$ of orthogonal polynomials admits the recurrence relation given by (3.55) for $n \geq 1$. Comparing the recurrence relations given in (3.75) and (3.58), we obtain $a_n = 1$ (for $n = 0, \cdots, k-2$), $b_n = -\alpha_n$ (for $n = 0, \cdots, k-1$), and $c_n = -\beta_n$ (for $n = 1, \cdots, k-1$). Coefficients α_n and β_n are given by

$$\alpha_n = -\frac{1}{\mu_n} \int_0^b x \exp\{-x^2\} H_{2,n}^2(x)dx, \tag{3.81}$$

and

$$\beta_n = -\frac{\mu_n}{\mu_{n-1}}, \tag{3.82}$$

in which

$$\mu_n = \int_0^b \exp\{-x^2\} H_{2,n}^2(x)dx. \tag{3.83}$$

Under the orthogonality assumption of $H_{2,n}(\cdot)$, by multiplying both sides of (3.58) by $\exp\{-x^2\}H_{2,n-1}(x)$ and then integrating, we can see that μ_n in (3.83) can be represented as

$$\mu_n = \int_0^b x \exp\{-x^2\} H_{2,n}(x) H_{2,n-1}(x)dx. \tag{3.84}$$

Using integrating by parts and then applying the well-known Christoffel–Darboux [Darboux, 1878] formula, Steen et al. [1969] showed that μ_n in (3.84) admits the recursion relation (3.55). Using integration by parts, then relation (3.81) equivalently becomes

$$\alpha_n = \frac{1}{2\gamma_n} \Big[\exp\{-x^2\} H_{2,n}^2(x) \Big]_0^b. \tag{3.85}$$

In general, computing coefficients β_n in (3.82) and α_n in (3.85) requires $H_{2,n}$ to be known for $n \geq 2$. To this end, Algorithm 12 describes how to compute coefficients α_n and β_n or which the initial values are given by the following.

$$\begin{cases} H_{2,0}(0) = 1, & H_{2,1}(0) = -\eta(b), \\ H_{2,0}(b) = 1, & H_{2,1}(b) = b - \eta(b), \\ \mu_0 = \frac{\sqrt{\pi}}{2}\text{erf}(b), & \mu_1 = \frac{\sqrt{\pi}}{4}\text{erf}(b) - \frac{1}{2}\Big[\exp\{-b^2\}[b - \eta(b)] + \eta(b)\Big], \\ \alpha_1 = \frac{1}{2\mu_1}\big[\exp\{-b^2\}[b - \eta(b)]^2 - \eta^2(b)\big], & \beta_1 = -\frac{1}{2} + \frac{1}{\sqrt{\pi}\text{erf}(b)}\Big[\exp\{-b^2\}[b - \eta(b)] + \eta(b)\Big]. \end{cases} \tag{3.86}$$

Using computed coefficients α_n and β_n based on Algorithm 12, the elements of matrix A in (3.76) are:

$$\begin{cases} a_n = 1 & \text{for } n = 0, \cdots, k-2, \\ b_n = -\alpha_n & \text{for } n = 0, \cdots, k-1, \text{in which } \alpha_0 = -\eta(b), \\ c_n = -\beta_n & \text{for } n = 1, \cdots, k-1. \end{cases}$$

Finally, taking into account the fact that

$$\mu_0 = \int_a^b w(x)dx = \int_0^b \exp\{-x^2\}dx = \frac{\sqrt{\pi}}{2}\text{erf}(b),$$

the weights ω_n (for $n = 1, \cdots, k$) are obtained using (3.77).

Algorithm 12 Computing first k coefficients of class $H_{2,n}(\cdot)$

Read k and b;
Set $h0$, hb, α, β, and μ to be vectors of length k;
Set α_1, β_1, and μ_1 as given by (3.86);
Set $H00 = H_{2,0}(0)$, and $H10 = H_{2,1}(0)$, $H0b = H_{2,0}(b)$, and $H1b = H_{2,1}(b)$ in which $H_{2,0}(0)$, $H_{2,1}(0)$, $H_{2,0}(b)$, and $H_{2,1}(b)$ are given by (3.86);
Set $n = 1$
while $n \leq k-1$ **do**
    $h0_{n+1} = \alpha_n \times H10 + \beta_n \times H00$;
    $hb_{n+1} = (b + \alpha_n) \times H1b + \beta_n \times H0b$;
    $\mu_{n+1} = \frac{n+1}{2}\mu_n - \frac{1}{2}\Big[\exp\{-b^2\} \times hb_{n+1} \times H1b - h0_{n+1} \times H10\Big]$;
    $\alpha_{n+1} = \frac{1}{2\mu_{n+1}}\Big[\exp\{-b^2\} \times \big(hb_{n+1}\big)^2 - \big(h0_{n+1}\big)^2\Big]$;
    $\beta_{n+1} = -\frac{\mu_{n+1}}{\mu_n}$;
    $H00 \leftarrow H10$ and $H0b \leftarrow H1b$;
    $H10 \leftarrow h0_{n+1}$ and $H1b \leftarrow hb_{n+1}$;
    $n \leftarrow n + 1$;
end
Return $\{\alpha\}_{n=1}^k$ and $\{\beta\}_{n=1}^k$.

- **nodes and weighs for Hermite polynomial of third kind**: If we let $b \to +\infty$ in (3.54), then the class of Hermite polynomials of second kind turns into the class of Hermite polynomials of third kind denoted as $H_{3,n}(\cdot)$. Hence, the coefficients α_n, β_n, and μ_n in class $H_{3,n}(\cdot)$ are those of class $H_{2,n}(\cdot)$ when $b \to +\infty$. We have

$$\alpha_n = -\frac{1}{\mu_n}\int_0^\infty x \exp\{-x^2\}H_{2,n}^2(x)dx, \tag{3.87}$$

and

$$\beta_n = -\frac{\mu_n}{\mu_{n-1}}, \tag{3.88}$$

in which

$$\mu_n = \int_0^\infty \exp\{-x^2\}H_{2,n}^2(x)dx. \tag{3.89}$$

Using integration by parts for (3.87) or taking limit in (3.85) when $b \to +\infty$, one can see

$$\alpha_n = -\frac{H_{2,n}^2(0)}{2\mu_n}. \tag{3.90}$$

Likewise, when $b \to +\infty$ in (3.55), then μ_n in admits the recursion relation

$$\mu_n = \frac{n}{2}\mu_{n-1} + \frac{H_{2,n}(0)H_{2,n-1}(0)}{2}. \tag{3.91}$$

For $n \geq 2$, coefficients α_n in (3.90) and β_n in (3.88) are determined using the recurrence relation (3.55) and the initial values

$$\begin{cases} H_{2,0}(0) = 1, & H_{2,1}(0) = -\frac{1}{\sqrt{\pi}}, \\ \mu_0 = \frac{\sqrt{\pi}}{2}, & \mu_1 = \frac{\mu_0}{2} - \frac{1}{2\sqrt{\pi}}, \\ \alpha_1 = \frac{2}{\sqrt{\pi}(2-\pi)}, & \beta_1 = \frac{2-\pi}{2\pi}, \\ \alpha_0 = -\frac{1}{\sqrt{\pi}}. \end{cases} \tag{3.92}$$

To this end, Algorithm 13 describes how to compute coefficients α_n and β_n for $n = 2, \cdots, k$. Using computed coefficients α_n and β_n, the elements of matrix A in (3.76) are:

Algorithm 13 Computing first k coefficients of class $H_{3,n}(\cdot)$

1: Read k and set h, α, β, and μ to be vectors of length k;
2: Set $\alpha_1 = 2/[\sqrt{\pi}(2-\pi)]$, $\beta_1 = (2-\pi)/(2\pi)$, and $\mu_1 = \sqrt{\pi}/4 - 1/(2\sqrt{\pi})$ using (3.92);
3: Set $n = 1$, $H0 = H_{2,0}(0) = 1$, and $H1 = H_{2,1}(0) = -1/\sqrt{\pi}$ using (3.92);
4: **while** $n \leq k-1$ **do**
5: $\quad h_{n+1} = \alpha_n H1 + \beta_n H0$;
6: $\quad \mu_{n+1} = \frac{n+1}{2}\mu_n + \frac{1}{2}h_{n+1}H1$;
7: $\quad \alpha_{n+1} = -\frac{1}{2\mu_{n+1}}h_{n+1}^2$;
8: $\quad \beta_{n+1} = -\frac{\mu_{n+1}}{\mu_n}$;
9: $\quad H0 \leftarrow H1$;
10: $\quad H1 \leftarrow h_{n+1}$;
11: $\quad$ Set $n = n+1$;
12: **end**
13: Return $\{\alpha\}_{n=1}^{k}$ and $\{\beta\}_{n=1}^{k}$.

$$\begin{cases} a_n = 1 & \text{for } n = 0, \cdots, k-2, \\ b_n = -\alpha_n & \text{for } n = 0, \cdots, k-1, \text{in which } \alpha_0 = -\frac{1}{\sqrt{\pi}}, \\ c_n = -\beta_n & \text{for } n = 1, \cdots, k-1. \end{cases}$$

Finally, taking into account the fact that

$$\mu_0 = \int_a^b w(x)dx = \int_0^\infty \exp\{-x^2\}dx = \frac{\sqrt{\pi}}{2},$$

the weights ω_n (for $n = 1, \cdots, k$) are obtained using (3.77). Since `R` does not have enough binary digits in order to represent results whose digits are more than 22, hence we cannot guarantee the precision of the computed nodes and weights for large k, > 10 say. Of course one can use the `Maple` framework for computing the nodes and weights of the Gauss-Hermite rule. The input parameters of the corresponding procedure, called `GH2`, consist of the number of nodes and b. The output consists of vector of weights, vector of nodes, and EC criterion, respectively.

Table 3.4: The nodes and weights associated with the Gauss-Hermite rule of third kind.

	$k=10$		$k=12$		$k=14$	
n	ω_n	x_n	ω_n	x_n	ω_n	x_n
1	9.85520975191087E-02	3.87385243257289E-02	7.62461468014692E-02	2.98897007730461E-02	6.12109822716413E-02	2.39567896629936E-02
2	2.08678066608185E-01	1.98233304013083E-01	1.66446068894088E-01	1.54204878281815E-01	1.36062060620609E-01	1.24240346144723E-01
3	2.52051688403761E-01	4.65201111814767E-01	2.19394898138567E-01	3.66143963007318E-01	1.88856803527084E-01	2.97338573288085E-01
4	1.98684340038387E-01	8.16861885592273E-01	2.07016508675540E-01	6.50881015894534E-01	1.98577829123488E-01	5.33329221273305E-01
5	9.71984227600620E-02	1.23454132402818E+00	1.37264362783550E-01	9.94366869943082E-01	1.58617337872050E-01	8.21873198117369E-01
6	2.70244164355446E-02	1.70679814968913E+00	6.05056743380072E-02	1.38589120372088E+00	9.28167828948399E-02	1.15406708458062E+00
7	3.80464962249537E-03	2.22994008892494E+00	1.65538019519272E-02	1.81884860850511E+00	3.79316390125047E-02	1.52327480337614E+00
8	2.28886243044656E-04	2.80910374689875E+00	2.58608378742107E-03	2.29084273875259E+00	1.02563910691812E-02	1.92533822320942E+00
9	4.34534479844469E-06	3.46387241949586E+00	2.06237540974292E-04	2.80409679347421E+00	1.72277180701059E-03	2.35860077983781E+00
10	1.24773714817825E-08	4.25536180636608E+00	7.06650986370700E-06	3.36727070424289E+00	1.65956340534487E-04	2.82409376823402E+00
11			7.59131546779026E-08	4.00168347575167E+00	8.19589322531928E-06	3.32626937208736E+00
12			1.18195417081408E-10	4.76821628806517E+00	1.73876608495078E-07	3.87510500420455E+00
13					1.14293978310768E-09	4.49243808452490E+00
14					1.04120010017399E-12	5.23843137515097E+00
	$\text{EC} = 25043 \times 10^{-19}$		$\text{EC} = 36052 \times 10^{-24}$		$\text{EC} = 37125 \times 10^{-29}$	

[1] The CE is a short form of the error coefficient given by (3.99).

```
> restart: with(LinearAlgebra)[`<,>`(Eigenvalues, Eigenvectors)]:
> GH2 := proc (k::integer, b)
    local alpha, A, beta, bn, cn, d, e, EC, H0, H1,
          H2,i, j, l, mu0, mu, n, node, u, V, weight;
    interface(rtablesize = k):
    node := Vector(k): weight := Vector(k): H2 := Vector(k):
    mu := Vector(k): alpha := Vector(k): beta := Vector(k):
    cn := Vector(k-1):  bn := Vector(k):
    A := Matrix(k, k, fill = 0):
    H0 := 1: H1 := x-(1-exp(-b^2))/(sqrt(Pi)*erf(b)):
    mu0 := evalf( (1/2)*sqrt(Pi)*erf(b) ):
    bn[1] := evalf( -(1/2)*(exp(-b^2)-1)/mu0 ):
    mu[1] := evalf( int(exp(-x^2)*H1^2, x=0 .. b) ):
    beta[1] := evalf( -mu[1]/mu0 ):
    alpha[1] := evalf( -(int(x*exp(-x^2)*H1^2, x = 0 .. b))/mu[1] ):
        for n to k-1 do:
            bn[n+1] := -alpha[n]:
            cn[n] := sqrt(-beta[n]):
            H2 := (x + alpha[n])*H1 + beta[n]*H0:
            mu[n+1] := evalf( int(exp(-x^2)*H2^2, x = 0 .. b) ):
            beta[n+1] := evalf( -mu[n+1]/mu[n] ):
            alpha[n+1] := evalf( -(int(x*exp(-x^2)*H2^2, x = 0 .. b))/mu[n+1] ):
            H0 := H1:
            H1 := H2:
        end do:
            d := Vector(bn): u := Vector(cn): l := Vector(cn):
            A := Matrix(k, proc (i, j) options operator, arrow:
            if i = j then d[i] elif i = j-1 then u[j-1]
            elif i = j+1 then l[i-1] else 0 end if end proc):
            e := Eigenvectors(A):
            node := Re(e[1]):
            V := e[2]:
                for n to k do:
                    weight[n] := evalf( mu0*Re(V[1, n])^2/Norm(V[..,n], Euclidean) ):
                end do:
         EC := evalf( mu[k]/GAMMA(2*k+1) ):
         weight, node, EC:
end proc
```

- **nodes and weighs for Jacobi polynomial**: The class $J_i^{\alpha,\beta}(\cdot)$ of orthogonal polynomials admits the recurrence relation

$$J_{n+1}^{\alpha,\beta}(x) = (xa_n^{\alpha,\beta} - b_n^{\alpha,\beta})J_n^{\alpha,\beta}(x) - c_n^{\alpha,\beta}J_{n-1}^{\alpha,\beta}(x), \tag{3.93}$$

with $J_0^{\alpha,\beta}(x) = 1$ and $J_1^{\alpha,\beta}(x) = x(\alpha+\beta+2)/2+(\alpha-\beta)/2$, for $n = 1, 2, \cdots$. Furthermore,

$$\begin{aligned} a_n^{\alpha,\beta} &= \frac{(2n+\alpha+\beta+1)(2n+\alpha+\beta+2)}{2(n+1)(n+\alpha+\beta+1)}, \\ b_n^{\alpha,\beta} &= \frac{(\beta^2-\alpha^2)(2n+\alpha+\beta+1)}{2(n+1)(n+\alpha+\beta+1)(2n+\alpha+\beta)}, \\ c_n^{\alpha,\beta} &= \frac{(n+\alpha)(n+\beta)(2n+\alpha+\beta+2)}{(n+1)(n+\alpha+\beta+1)(2n+\alpha+\beta)}. \end{aligned} \tag{3.94}$$

Rearranging the RHS of (3.93) and comparing it with the recurrence relation given in

(3.75), we obtain

$$\begin{cases} a_0 = \frac{2}{\alpha+\beta+2} \text{ and } b_0 = \frac{\beta-\alpha}{\alpha+\beta+2}, & \text{if } n = 0, \\ a_n = \frac{2(n+1)(n+\alpha+\beta+1)}{(2n+\alpha+\beta+1)(2n+\alpha+\beta+2)}, & \text{if } n \geq 1, \\ b_n = \frac{\beta^2-\alpha^2}{(2n+\alpha+\beta)(2n+\alpha+\beta+2)}, & \text{if } n \geq 1, \\ c_n = \frac{2(n+\alpha)(n+\beta)}{(2n+\alpha+\beta)(2n+\alpha+\beta+1)}, & \text{if } n \geq 1. \end{cases} \tag{3.95}$$

In this case, matrix A in (3.76) can be reconstructed by using the constants a_n, b_n, and c_n provided in (3.95). As before, the nodes are eigenvalues of the reconstructed matrix A and for computing weights, it follows from Table 3.3 that

$$\mu_0 = \int_a^b w(x)dx = \int_{-1}^{1} (1-x)^\alpha (1+x)^\beta dx = \frac{2^{\alpha+\beta+1}\Gamma(\alpha+1)\Gamma(\beta+1)}{\Gamma(\alpha+\beta+2)}.$$

Hence the weights ω_n (for $n = 1, \cdots, k$) are obtained by considering $\mu_0 = 2$ in the RHS of (3.77) in which $\boldsymbol{v}_n$ (for $n = 1, \cdots, k$) is the eigenvector corresponds to nth eigenvalue of matrix A constructed based on information given by (3.95).

- **nodes and weighs for Legendre polynomial**: The class $L_n(\cdot)$ of orthogonal polynomials admits the recurrence relation

$$xL_n(x) = \frac{n+1}{2n+1} L_{n+1}(x) + \frac{n}{2n+1} L_{n-1}(x), \tag{3.96}$$

with $L_0(x) = 1$ and $L_1(x) = x$, for $n = 1, 2, \cdots$. Comparing the recurrence relations given in (3.75) and (3.96), we obtain $b_n = 0$, for $n = 0, 1, 2, \cdots$, and

$$\begin{cases} a_n = \frac{n+1}{2n+1}, & \text{if } n \geq 0, \\ c_n = \frac{n}{2n+1}, & \text{if } n \geq 1. \end{cases} \tag{3.97}$$

In this case, matrix A in (3.76) can be reconstructed by setting $\sqrt{a_{n-1}c_n} = n/\sqrt{4n^2-1}$ (for $n = 1, \cdots, k$) in the RHS of (3.76). The nodes are eigenvalues of the reconstructed matrix A. For computing weights, it follows from Table 3.3 that

$$\mu_0 = \int_a^b w(x)dx = \int_{-1}^{1} dx = 2.$$

Hence the weights ω_n (for $n = 1, \cdots, k$) are obtained by considering $\mu_0 = 2$ in the RHS of (3.77) in which $\boldsymbol{v}_n$ (for $n = 1, \cdots, k$) is the eigenvector corresponds to nth eigenvalue of matrix A constructed based on information given by (3.97).

It should be noted that the accuracy of a Gaussian quadrature depends on the structure of the orthogonal polynomial. In general, if $f(x)$ in (3.70) has continuous derivatives of order $2k$, then the error in approximating integral (3.70) becomes [Kahaner et al., 1989]:

$$\begin{aligned} \text{Error} =& \int_a^b w(x)f(x)dx - \sum_{i=1}^{k} \omega_i f(x_i) \\ =& \frac{1}{\Gamma(2k+1)} \int_a^b w(x) P_k^2(x) dx \times f^{(2k)}(y) \\ =& \frac{\mu_n}{\Gamma(2k+1)} f^{(2k)}(y), \end{aligned} \tag{3.98}$$

where $f^{(2k)}(y)$ denotes the $(2k)$th derivative of $f(x)$ for some $a < y < b$. Regardless of $f^{(2k)}(y)$ in RHS of (3.98), the multiplicative term

$$\text{EC} = \frac{\mu_n}{\Gamma(2k+1)}, \tag{3.99}$$

is called error coefficient (EC). It should be noted that, in special case, when $w(x) = 1$, we have

$$\text{Error} = \frac{(b-a)^{2k+1}\left[\Gamma(k+1)\right]^4}{(2k+1)\left[\Gamma(2k+1)\right]^3} f^{(2k)}(y).$$

If the exact value of integral I is known, then a useful criterion, the goodness of approximation $\hat{I}$ for I is reflected by absolute relative error (ARE) criteria defined as

$$\text{ARE} = \left|\frac{I - \hat{I}}{I}\right|. \tag{3.100}$$

The following `R` function `quad_rule` given below computes the nodes and weights for implementing the Gaussian quadrature rules discussed earlier. We note that the argument `type` includes: `CH1` (for Chebyshev of first kind), `CH2` (for Chebyshev of second kind), `HE1` (for Hermite of first kind), `HE2` (for Hermite of second kind), `HE3` (for Hermite of third kind), `GL` (for generalized Laguerre), `JA` (for Jacobi), and `LE` (for Legendre). The arguments `alpha` and `beta` are parameters of Gaussian-generalized Laguerre (or Gauss-Jacobi) rule and argument b is the parameter of Gauss-Hermite of third kind rule.

```
R> quad_rule <- function(k, type = "HE2", alpha = 0, beta = 0, b = Inf)
+ {
+ if ( any( c(alpha, beta) <= -1 ) )stop("␣alpha␣and␣beta␣must␣be␣greater␣than␣-1.")
+ if ( type == "HE3" & is.infinite(b) == T )stop("␣b␣must␣be␣finite.")
+ A <- diag(0, k)
+ if ( type == "CH1" )
+ {
+ ia <- 1:k
+ node <- cos( (2*ia - 1)/(2*k)*pi )
+ weight <- rep(pi/k, k)
+ }
+ if ( type == "CH2" )
+ {
+ ia <- 1:k
+ node <- cos( ia/(k + 1)*pi )
+ weight <- pi/(k + 1)*( sin( ia/(k + 1)*pi ) )^2
+ }
+ if ( type == "HE1" )
+ {
+ a_nc_n <- 1:(k - 1)/2
+ A[row(A) - col(A) == 1] <- A[row(A) - col(A) == -1] <- sqrt( a_nc_n )
+ A_eigen <- eigen(A)
+ node <- A_eigen$values
+ weight <- sqrt(pi)*( A_eigen$vector[1, ] )^2
+ }
+ if ( type == "HE2" )
+ {
+ alpha <- beta <- mu <- h <- rep(0, k - 1)
+ H0 <- 1
+ H1 <- -1/sqrt(pi)
+ alpha_1 <- 2/( sqrt(pi)*(2 - pi) )
+ alpha[1] <- alpha_1
+ beta[1] <- (2 - pi)/(2*pi)
+ mu_1 <- sqrt(pi)/4 - 1/( 2*sqrt(pi) )
+ mu[1] <- mu_1
+  for(n in 1:(k - 1) )
+  {
+   h[n + 1] <- alpha[n]*H1 + beta[n]*H0
+   mu[n + 1] <- (n + 1)/2*mu[n] + h[n + 1]*H1/2
+   alpha[n + 1] <- -1/(2*mu[n + 1])*( h[n + 1] )^2
+   beta[n + 1] <- -mu[n + 1]/mu[n]
+   H0 <- H1
```

```
+   H1 <- h[n + 1]
+  }
+ b_n <- abs( c( 1/sqrt(pi), -alpha[1:(k - 1)] ) )
+ a_nc_n <- -beta[1:(k - 1)]
+ A[row(A) - col(A) == 1] <- A[row(A) - col(A) == -1] <- sqrt( abs( a_nc_n ) )
+ diag(A) <- b_n
+ A_eigen <- eigen(A)
+ node <- A_eigen$values
+ weight <- sqrt(pi)/2*( A_eigen$vector[1, ] )^2
+ }
+ if ( type == "HE3" )
+ {
+  alpha <- beta <- mu <- h_0 <- h_b <- rep(0, k - 1)
+  erf <- 2*pnorm( sqrt(2)*b ) - 1
+  eta <- ( 1 - exp(-b^2) )/( sqrt(pi)*erf )
+  H0_0 <- H0_b <- 1
+  H1_0 <- -eta
+  H1_b <- b - eta
+  mu_0 <- sqrt(pi)*erf/2
+  mu[1] <- mu_0/2 - 1/2*( exp(-b^2)*H1_b*H0_b - H1_0*H0_0 )
+  beta[1] <- -mu[1]/mu_0
+  alpha[1] <- 1/(2*mu[1])*( exp(-b^2)*(H1_b)^2 - (H1_0)^2 )
+  for(n in 1:(k - 1) )
+  {
+   h_0[n + 1] <- alpha[n]*H1_0 + beta[n]*H0_0
+   h_b[n + 1] <- (b + alpha[n])*H1_b + beta[n]*H0_b
+   mu[n + 1] <- (n + 1)/2*mu[n] - 1/2*( exp(-b^2)*h_b[n + 1]*H1_b - h_0[n + 1]*H1_0 )
+   alpha[n + 1] <- 1/(2*mu[n + 1])*( exp(-b^2)*( h_b[n + 1] )^2 - ( h_0[n + 1] )^2 )
+   beta[n + 1] <- -mu[n + 1]/mu[n]
+   H0_0 <- H1_0
+   H0_b <- H1_b
+   H1_0 <- h_0[n + 1]
+   H1_b <- h_b[n + 1]
+  }
+ b_n <- abs( c( eta, -alpha[1:(k - 1)] ) )
+ a_nc_n <- -beta[1:(k - 1)]
+ A[row(A) - col(A) == 1] <- A[row(A) - col(A) == -1] <- sqrt( abs(a_nc_n) )
+ diag(A) <- b_n
+ A_eigen <- eigen(A)
+ node <- A_eigen$values
+ weight <- mu_0*( A_eigen$vector[1, ] )^2
+ }
+ if ( type == "GL" )
+ {
+ ib <- seq(0, k - 1)
+ b_n <- 2*ib + alpha + 1
+ ia <- seq(0, k - 2)
+ diag(A) <- b_n
+ A[row(A) - col(A) == 1] <- A[row(A) - col(A) == -1] <- sqrt( (ia + 1)*(ia + 1 + alpha) )
+ A_eigen <- eigen(A)
+ node <- A_eigen$values
+ weight <- gamma(alpha + 1)*( A_eigen$vector[1, ] )^2
+ }
+ if ( type == "JA" )
+ {
+ ib <- seq(1, k - 1)
+ b_n <- (beta^2 - alpha^2)/( (2*ib + alpha + beta)*(2*ib + alpha + beta + 2) )
+ ia <- seq(1, k - 2)
+ a_n <- 2*(ia+1)*(ia+alpha+beta+1)/( (2*ia+alpha+beta+1)*(2*ia+ alpha+beta+2) )
+ ic <- ia + 1
+ c_n <- 2*(ic + alpha)*(ic + beta )/( (2*ic + alpha + beta)*(2*ic + alpha + beta + 1) )
+ diag(A) <- c(0, b_n )
+ A[row(A) - col(A) == 1] <- A[row(A) - col(A) == -1] <- c(0, sqrt( a_n*c_n ) )
```

```
+ A[1, 1] <- (beta - alpha)/( alpha + beta + 2 )
+ A[1, 2] <- sqrt( 4*(1 + alpha)*(1 + beta )/( (2 + alpha + beta)^2*(3 + alpha + beta) ) )
+ A[2, 1] <- A[1, 2]
+ A_eigen <- eigen(A)
+ node <- A_eigen$values
+ weight <- 2^(alpha + beta + 1)*gamma(alpha + 1)*gamma(beta + 1)/
+ gamma(alpha + beta + 2)*( A_eigen$vector[1, ] )^2
+ }
+ if ( type == "LE" )
+ {
+ ia <- seq(0, k - 2)
+ a_n <- (ia + 1)/(2*ia + 1)
+ ic <- ia + 1
+ c_n <- ic/(2*ic + 1)
+ A[row(A) - col(A) == 1] <- A[row(A) - col(A) == -1] <- sqrt( a_n*c_n )
+ A_eigen <- eigen(A)
+ node <- A_eigen$values
+ weight <- 2*( A_eigen$vector[1, ] )^2
+ }
+ ls <- list( "node" = node , "weight" = weight)
+ return(ls)
+ }
```

Example 3-8

Let's to approximate integral (3.48) of Example 3-7 through the Gaussian-Legendre rule. To this end, first, we need to convert the range of integration from $[0, 3]$ to $[-1, 1]$. Using a change of variable $z = 2u/3 - 1$ it turns out that

$$\begin{aligned} I = \int_0^3 x \exp\{-x\} dx =& \frac{9}{4} \exp\Big\{-\frac{3}{2}\Big\} \int_{-1}^1 (z+1) \exp\Big\{-\frac{3}{2} z\Big\} dz \\ =& \frac{9}{4} \exp\Big\{-\frac{3}{2}\Big\} \int_{-1}^1 w(z) f(z) dz, \end{aligned} \tag{3.101}$$

where $w(z) = 1$ and $f(z) = (z+1)\exp\{-3z/2\}$. Hence, the sequence of nodes and weights can be computed using function `quad_rule`, when `type="LE"`. Computing the sequence $\{x_i, \omega_i\}_{i=1}^k$, we can write

$$I \approx \frac{9}{4} \exp\Big\{-\frac{3}{2}\Big\} \sum_{i=1}^k \omega_i \times (x_i + 1) \exp\Big\{-\frac{3}{2} x_i\Big\}. \tag{3.102}$$

The pertaining AREs for $k = 2, 3, 5$, and 10 are 4.58e-02, 1.50e-03 1.75e-07, and 3.18e-15, respectively. We remind that the general quadrature rule applied in Example 3-7 yields 0.83755 with ARE = 0.045. Evidently, he Gaussian quadrature shows superior performance with ARE = 0.045 just by adding one more node. ■

Example 3-9

Suppose we are interested in approximating

$$I(p) = \int_0^\infty x^p \log x \exp\{-x\} dx, \tag{3.103}$$

where $p > 0$. Since the range of integrand I is $(0, \infty)$ and involves the term $\exp\{-x\}$, using class $\mathcal{L}_k^p$ of orthogonal polynomials, a Gaussian quadrature rule is applied for approximating $I(p)$ for different settings of k and p. To this end, we compute the sequence of nodes and weights using function `quad_rule`, when `type="GL"` and `alpha=`p. Computing sequence $\{x_i, \omega_i\}_{i=1}^k$, we approximate integral $I(p)$ as follows.

$$I(p) \approx \sum_{i=1}^{k} \omega_i \times (x_i)^p \log x_i. \tag{3.104}$$

Table 3.5 shows the ARE results for computing integral $I(p)$ in (3.103). It turns out from Table 3.5 that for larger p, smaller k provides the accuracy at desired level. Figure 3.6 shows the

Table 3.5: ARE for approximating integral in (3.103).

		ARE			
p	Exact value of $I(p)$	$k = 5$	$k = 10$	$k = 15$	$k = 20$
0	-0.5772156	0.214375	0.108274	7.24e-02	5.44e-02
1	0.42278433	4.50e-01	0.012404	5.70e-03	3.26e-03
2	1.84556867	5.69e-03	9.12e-04	2.96e-04	1.30e-04
3	7.53670601	1.53e-03	1.51e-04	3.54e-05	1.22e-05
4	36.1468240	5.64e-04	3.55e-05	6.14e-06	1.68e-06

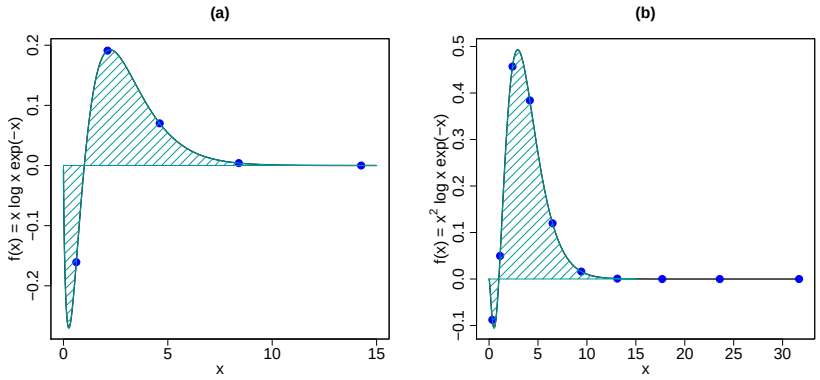


Figure 3.6: Graph of integrand (3.103). (a): $p = 1$ and $k = 5$, (b): $p = 2$ and $k = 10$.

integrand of integral $I(p)$ in (3.103) on which x-coordinates of bullets are the position of the nodes. ■

Example 3-10

Suppose we are interested in approximating

$$I(z, \alpha) = \frac{1}{2\pi} \int_{-\infty}^{\infty} \exp\{-u^\alpha\} \cos(zu) du, \tag{3.105}$$

where $0 < \alpha \leq 2$. In fact, integral $I(z, \alpha)$ is the PDF of symmetric standard α-stable distribution at point z [Nolan, 2020]. For $\alpha = 2$, each standard symmetric α-stable distribution turns into a zero-mean Gaussian distribution with scale $1/\sqrt{2}$. Hence, we have $I(z, 2) = \mathcal{N}(z|0, 1/2)$. Herein, using a Gauss-Hermite rule, we proceed to approximate integral $I(z, \alpha)$. Table 3.6 shows the ARE for different settings of k and z. As it is seen, the ARE for large z decreases and a larger k needs to provide the accuracy of desired level. Figure 3.7 displays the pertaining

integrand on which x-coordinates of bullets are the position of the nodes that are symmetrically distributed around origin. It it worth to note that integrand in (3.105) is an oscillatory function and when z is large the number of oscillations is large [Nolan, 2020, pp. 68]. ■

Table 3.6: ARE for approximating integral in (3.105) for $\alpha = 2$ using Gauss-Hermite rule.

		ARE			
z	Exact value	$k = 5$	$k = 10$	$k = 15$	$k = 20$
0	0.28209479	1.96e-16	1.96e-16	1.96e-16	5.90e-16
1	0.21969564	1.18e-06	2.65e-15	8.84e-16	1.51e-15
2	0.10377687	1.80e-03	2.56e-09	3.61e-15	5.08e-15
3	0.02973257	2.02e-01	1.63e-05	9.84e-11	1.08e-14
4	0.00516674	9.18174	1.27e-02	1.35e-06	2.64e-11
5	0.00054457	284.672	3.568332	3.47e-03	6.29e-07

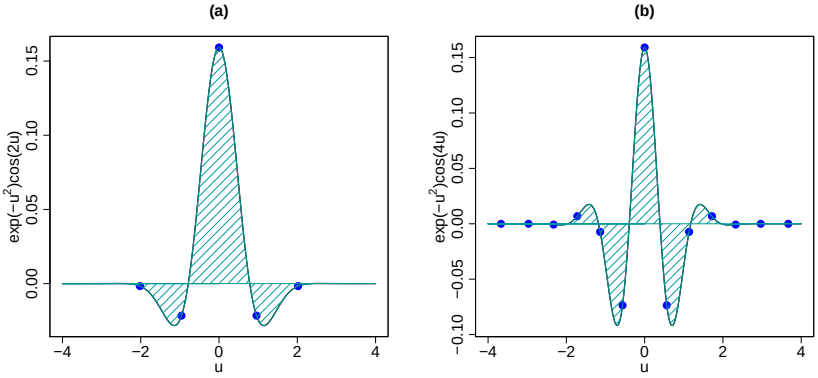


Figure 3.7: Graph of integrand (3.105) for (a): $z = 2$ and $k = 5$, (b):$z = 4$ and $k = 15$.

Example 3-11

The Owen's $T(h, a)$ function is very useful for computing the CDF of skew Gaussian [Azzalini, 2022] and bivariate Gaussian probabilities [Gupta, 1963, Young and Minder, 1974, Thomas, 1986]. This function has an integral representation as [Owen, 1956]:

$$T(h,a) = \frac{1}{2\pi}\int_0^a \frac{\exp\{-\frac{1}{2}h^2(1+z^2)\}}{1+z^2}dz, \tag{3.106}$$

where $h, a \in \mathbb{R}$. Patefield [2000] proposed six methods for evaluating $T(h, a)$. A listing of interesting properties of this function is given by[Owen, 1956]:

$$\begin{cases} T(h,-a) = -T(h,a), & T(-h,a) = T(h,a), \\ T(0,a) = \frac{1}{2\pi}\arctan(a), & T(h,1) = \frac{1}{2}\Phi(h)[1-\Phi(h)], \\ T(h,a) = \frac{1}{2}[\Phi(h)+\Phi(ah)] - \Phi(h)\Phi(ah) - T(ah,\frac{1}{a}), & \text{if } h > 0, \end{cases} \tag{3.107}$$

and

$$T(h,\infty) = \begin{cases} \dfrac{1}{2}\bar{\Phi}(h), & \text{if } h \geq 0, \\ \dfrac{1}{2}\Phi(h), & \text{if } h \leq 0, \end{cases} \tag{3.108}$$

where $\Phi(\cdot)$ is the CDF of a standard univariate Gaussian. For instance, based on the last property in (3.107), computing $T(h,a)$ can be reduced simply for a smaller interval $0 \leq a \leq 1$, rather than the whole half line when $h \geq 0$. However, based on their procedure, each of six approaches has been applied on one or more parts of the region $\mathcal{R} = \{(a,h) \in \mathbb{R}^2 | a > 0, h > 0\}$ consisting of 18 partitions. Among those, their fifth method , $T5$ say, uses a $2k$-point Gauss-Legendre rule for which the nodes and weights can be obtained, for example, form Abramowitz and Stegun [1948] as suggested. Taking into account the fact that integrand in $T(h,a)$ is an even function, they suggest to compute $T(h,a)$ in a simpler form that allows to reduce the number of node (or weight) points by half. This yields

$$\begin{aligned} T(h,a) =& \frac{a}{4\pi} \int_{-1}^{1} \frac{\exp\{-\frac{1}{2}h^2(1+(az)^2)\}}{1+(az)^2} dz \\ \approx& \frac{a}{4\pi} \sum_{i=1}^{k} \omega_i \frac{\exp\{-\frac{1}{2}h^2(1+(ax_i)^2)\}}{1+(ax_i)^2}, \end{aligned} \tag{3.109}$$

where $\{x_i, \omega_i\}_{i=1}^{k}$ in RHS of (3.109) are the set of first k nodes and weights obtained from $2k$-point Gauss-Legendre rule. To this end, we may use command `quad_rule(2k,type="LE")`. We further can approximate the Owen's $T(h,a)$ function using a k-point Gauss-Hermite (second kind) rule. To this end, from (3.106), it may be easily seen that

$$\begin{aligned} T(h,a) =& \frac{\exp\{-\frac{1}{2}h^2\}}{h\sqrt{2\pi}} \int_{0}^{\frac{ha}{\sqrt{2}}} \frac{\exp\{-z^2\}}{1+2h^{-2}z^2} dz \\ \approx& \frac{\exp\{-\frac{1}{2}h^2\}}{h\sqrt{2\pi}} \sum_{i}^{k} \omega_i \Big[1+2h^{-2}x_i^2\Big]^{-1}, \end{aligned} \tag{3.110}$$

where $\{x_i, \omega_i\}_{i=1}^{k}$ in RHS of (3.110) is the set of nodes and weights obtained from k-point Gauss-Hermite (second kind) rule for which $b = ha/\sqrt{2}$. It should be noted that for the pairs $(h,a) = (\sqrt{2},1)$ and $(h,a) = (\sqrt{2},2)$ we have $b = 1$ and $b = 2$, respectively. Hence, we may use function `quad_rule` using commands `quad_rule(k,type="HE2",b=1)` ad `quad_rule(k,type="HE2",b=2)` for computing $\{x_i, \omega_i\}_{i=1}^{k}$ in above two case. In what follows, we compare the performance of Gauss-Hermite (second kind) and Gauss-Legendre rules for computing Owen's $T(\sqrt{2},a)$. We note that the exact value of $T(h,1)$ can be simply computed in terms of $\Phi(\cdot)$ as shown by (3.107). The resultants are given in Table 3.7. As it is seen from Table 3.7, for $a = 1$, the Gauss-Legendre rule outperforms the Gauss-Hermite rule while for $a = 2$, the reverse holds true. ■

3.7 Multivariate Gaussian quadrature

Each univariate k-point Gaussian quadrature rule represented as (3.70) can be straightforwardly extended to the p-dimensional case with k^p points. Let $f(\boldsymbol{x})$ denote real function with domain $\boldsymbol{x} \in \mathbb{R}^p$ and we are willing to approximate the multiple integral of the form

$$\int_{\boldsymbol{a}}^{\boldsymbol{b}} w(\boldsymbol{x}) f(\boldsymbol{x}) d\boldsymbol{x} = \int_{a_1}^{b_1} \cdots \int_{a_p}^{b_p} w(x_1, \cdots, x_p) f(x_1, \cdots, x_p) dx_1 \cdots dx_p, \tag{3.111}$$

in which elements of either $\boldsymbol{a} = (a_1, \cdots, a_p)^\top$ or $\boldsymbol{b} = (b_1, \cdots, b_p)^\top$ may be finite or infinite. It is assumed that the weight function is decomposed as the product of p marginal weight functions as $w(\boldsymbol{x}) = w_1(x_1) w_2(x_2) \cdots w_p(x_p)$. Hence, the integral (3.111) can be represented as

$$\int_{\boldsymbol{a}}^{\boldsymbol{b}} w(\boldsymbol{x}) f(\boldsymbol{x}) d\boldsymbol{x} = \int_{a_1}^{b_1} \cdots \int_{a_p}^{b_p} w_1(x_1) w_2(x_2) \cdots w_p(x_p) f(x_1, \cdots, x_p) dx_1 \cdots dx_p.$$

Table 3.7: Absolute error of the Gaussian quadrature rules in approximating $T(\sqrt{2}, a)$.

			absolute error	
a	k	exact value	Gauss-Hermite	Gauss-Legendre
	2	0.036231921695240241	5.26e-05	1.70e-04
	4		2.18e-07	1.52e-07
$a = 1$	6		4.07e-10	1.32e-10
	8		8.33e-13	1.15e-13
	10		6.30e-14	2.01e-16
	2	0.039282968506234197[1]	5.39e-04	5.86e-03
	4		6.17e-06	1.38e-04
$a = 2$	6		4.02e-08	2.95e-06
	8		2.72e-10	6.30e-08
	10		1.53e-11	1.34e-09

[1] Hint: The exact value of $T(\sqrt{2}, 2)$ has been computed using `Maple`.

The Gaussian quadrature in multivariate case allows to approximate the integral (3.111) as

$$\int_{\boldsymbol{a}}^{\boldsymbol{b}} w(\boldsymbol{x}) f(\boldsymbol{x}) d\boldsymbol{x} \approx \sum_{i_1=1}^{k_1} \sum_{i_2=1}^{k_2} \cdots \sum_{i_p=1}^{k_p} \omega_{1i_1} \omega_{2i_2} \cdots \omega_{pi_p} f(x_{1i_1}, x_{2i_2}, \cdots, x_{pi_p}), \tag{3.112}$$

where $\{(x_{ij}, \omega_{ij})\}_{j=1}^{k_i}$ is the set of k_i nodes and weights associated with the ith coordinate under one of seven Gauss quadrature rules given by Table 3.3, for $i = 1, \cdots, p$. It is worthwhile to note that the Gauss quadrature rule applied to two distinct coordinates may be different. For example, for approximating a double integral, we may compute k_1 pairs of nodes and weights for the first coordinate using the Gauss-Laguerre rule while the set of k_2 nodes and weights for the second coordinate are obtained using the Gauss-Legendre rule.

Example 3-12

Let the PDF of random vector $\boldsymbol{X} = (X_1, X_2)^\top$ is given by

$$g(x_1, x_2|\boldsymbol{\theta}) = \frac{\beta^\alpha x_1^{\alpha+\lambda-1} x_2^{\lambda-1}}{\Gamma(\lambda)\Gamma(\alpha)} \exp\{-\beta x_1 - x_1 x_2\}, \tag{3.113}$$

where $x_1 > 0$, $x_2 > 0$, and $\boldsymbol{\theta} = (\alpha > 0, \beta > 0, \lambda > 0)^\top$. For computing the associated CDF that can be expressed in terms of integral representation as

$$G(q_1, q_2|\boldsymbol{\theta}) = \frac{\beta^\alpha}{\Gamma(\lambda)\Gamma(\alpha)} \int_0^{q_1} \int_0^{q_2} x_1^{\alpha+\lambda-1} x_2^{\lambda-1} \exp\{-\beta x_1 - x_1 x_2\} dx_2 dx_1, \tag{3.114}$$

where $q_1 \in \mathbb{R}^+$ and $q_2 \in \mathbb{R}^+$, we would like to use the Gaussian quadrature rule. To this end, we apply the Gauss-Legendre rule to both coordinates of integral in the RHS of (3.114). Hence, using a simple change of variable, we have

$$\begin{aligned} G(q_1, q_2|\boldsymbol{\theta}) = & \int_{-1}^{1} \int_{-1}^{1} \frac{\beta^\alpha}{\Gamma(\lambda)\Gamma(\alpha)} \Big[\frac{x_1+1}{2} q_1\Big]^{\alpha+\lambda-1} \Big[\frac{x_2+1}{2} q_2\Big]^{\lambda-1} \\ & \times \exp\Big\{-\frac{x_1+1}{2} q_1 \Big[\beta + \frac{x_2+1}{2} q_2\Big]\Big\} dx_2 dx_1 \\ = & \frac{q_1 q_2}{4} \int_{-1}^{1} \int_{-1}^{1} w(\boldsymbol{x}) f(x_1, x_2) dx_2 dx_1, \end{aligned} \tag{3.115}$$

in which

$$w(\boldsymbol{x}) = w(x_1)w(x_2) = 1 \times 1 = 1, \tag{3.116}$$

$$f(u,v) = \frac{\beta^{\alpha}}{\Gamma(\lambda)\Gamma(\alpha)}\Big[\frac{u+1}{2}q_1\Big]^{\alpha+\lambda-1}\Big[\frac{v+1}{2}q_2\Big]^{\lambda-1}\exp\Big\{-\frac{u+1}{2}q_1\Big[\beta+\frac{v+1}{2}q_2\Big]\Big\}. \tag{3.117}$$

It follows form (3.130) that

$$G(q_1,q_2|\boldsymbol{\theta}) \approx \frac{q_1 q_2}{4}\sum_{i=1}^{k_1}\sum_{j=1}^{k_2}\omega_{1i}\omega_{2j}f(x_{1i},x_{2j}), \tag{3.118}$$

where ω_{pi} and x_{pi} denote, accordingly, the ith weight and node associated with the pth coordinate, for $p = 1, 2$, and $f(\cdot,\cdot)$ is given by (3.133). Furthermore, constants k_1 and k_2 are the number of nodes (or weights) corresponds to quadrature rule considered for the first and second coordinates, respectively. Evidently, herein, we should use the Gauss-Legendre rule. In what follows, we give `R` code for approximating $G(q_1,q_2|\boldsymbol{\theta})$.

```
R > G_quad <- function(q1, q2, k, theta)
+ {
+ alpha <- theta[1]; beta <- theta[2]; lambda <- theta[3]
+ out <- quad_rule(k, type = "LE", alpha = 0, beta = 0)
+ weight_grid <- x_grid <- matrix(0, nrow = k^2, ncol = 2)
+ weight_grid <- apply( expand.grid(out$weight, out$weight), 1, prod)
+ x_grid <- expand.grid(out$node, out$node)
+ x1 <- (x_grid[, 1] + 1)*q1/2
+ x2 <- (x_grid[, 2] + 1)*q2/2
+ f <- x1^(alpha + lambda - 1)*x2^(lambda - 1)*exp(-beta*x1 - x1*x2)
+ I1 <- sum( weight_grid*f )
+ q1*q2/4*beta^alpha/(gamma(lambda)*gamma(alpha))*I1
+ }
```

It should be noted that the function `G_quad(...)` depends on `quad_rule(...)` that is given after (3.100). We end this example with a brief investigation on performance of `G_quad(...)` for approximating CDF given by (3.114). Table 3.8 shows the pertinent results.

As it is seen from Table 3.8, when λ is small or both elements of pair (α,λ) are small, then the performance of Gaussian quadrature significantly lessened for $k = 10$ compared to other settings of k. Figure 3.8 (a)-(d) display the PDF (3.131) for two settings of parameter vector $\boldsymbol{\theta}$. Specifications of the Gauss-Legendre rule with $k_1 = k_2 = 5$ points for approximating $G(0.5, 0.5|(0.5, 1, 0.5)^{\top})$ and $G(10, 10|(5, 1, 5)^{\top})$ are shown by Figure 3.8 (e)-(f). ■

Example 3-13

Recall the univariate BS distribution whose CDF is given by (2.38). We write $\boldsymbol{T} \sim \mathcal{BS}(\boldsymbol{\theta})$ to denote random vector $\boldsymbol{T} = (T_1, T_2)^{\top}$ follows a bivariate BS distribution with PDF given by [Kundu et al., 2010]:

$$f(t_1,t_2|\boldsymbol{\theta}) = \boldsymbol{\phi}_2(\boldsymbol{z}|\mathbf{0},\Sigma)\prod_{i=1}^{2}\frac{1}{2\alpha_i\beta_i}\Big[\Big(\frac{\beta_i}{t_i}\Big)^{\frac{1}{2}} + \Big(\frac{\beta_i}{t_i}\Big)^{\frac{3}{2}}\Big], \tag{3.119}$$

where $\boldsymbol{\theta} = (\alpha_1, \beta_1, \alpha_2, \beta_2, \rho)^{\top}$ is the family parameter vector, $\Sigma = [(1,\rho)^{\top}, (\rho,1)^{\top}]$, and $\boldsymbol{z} = (z_1, z_2)^{\top}$ with

$$z_i = \frac{1}{\alpha_i}\Big[\sqrt{\frac{t_i}{\beta_i}} - \sqrt{\frac{\beta_i}{t_i}}\Big],\ i = 1, 2. \tag{3.120}$$

Table 3.8: ARE for approximating CDF (3.114) for $\beta = 1$ using Gauss-Legendre rule.

$q_1 = 0.5$	$q_2 = 0.5$			ARE		
		Exact value[1]	$k = 10$	$k = 40$	$k = 80$	$k = 160$
$\alpha = 0.5$	$\lambda = 0.5$	0.17066603	0.04304235	0.01115619	0.00561266	0.00281506
$\alpha = 0.5$	$\lambda = 5$	3.2578e-07	4.5364e-07	4.53441e-07	4.5341e-07	4.5341e-07
$\alpha = 5$	$\lambda = 0.5$	8.1943e-05	0.04438234	0.01150114	0.00578534	0.00290082
$\alpha = 5$	$\lambda = 5$	5.5782e-10	5.1988e-10	5.1988e-10	5.1988e-10	5.1988e-10
$q_1 = 0.5$	$q_2 = 10$					
$\alpha = 0.5$	$\lambda = 0.5$	0.48776719	0.06751127	0.01745956	0.00878285	0.00440495
$\alpha = 0.5$	$\lambda = 5$	0.05560379	7.0189e-09	5.0508e-10	5.0508e-10	5.0508e-10
$\alpha = 5$	$\lambda = 0.5$	1.7098e-04	0.09553933	0.02465960	0.01240306	0.00622006
$\alpha = 5$	$\lambda = 5$	6.8194e-05	9.2085e-10	9.2843e-10	9.2843e-10	9.2843e-10
$q_1 = 10$	$q_2 = 0.5$					
$\alpha = 0.5$	$\lambda = 0.5$	0.39181881	0.04766594	0.01234977	0.00621302	0.00311616
$\alpha = 0.5$	$\lambda = 5$	0.00118855	5.0360e-06	1.9364e-11	1.9364e-11	1.9364e-11
$\alpha = 5$	$\lambda = 0.5$	0.92144044	0.07497320	0.01938347	0.00975048	0.00489024
$\alpha = 5$	$\lambda = 5$	0.12532847	3.4583e-07	2.5685e-10	2.5685e-10	2.5685e-10
$q_1 = 10$	$q_2 = 10$					
$\alpha = 0.5$	$\lambda = 0.5$	0.80501002	0.10896581	0.02690028	0.01352624	0.00678325
$\alpha = 0.5$	$\lambda = 5$	0.34088538	1.7070e-02	5.3893e-09	2.5295e-10	2.5120e-10
$\alpha = 5$	$\lambda = 0.5$	0.97074572	0.34194629	0.08258106	0.04142828	0.02076378
$\alpha = 5$	$\lambda = 5$	0.97017575	1.3694e-02	2.0986e-11	2.0992e-11	2.0983e-11

[1] The exact value has been computed using `Maple` up to eight decimal places.

We further note that

$$P(T_1 \le t_1, T_2 \le t_2|\boldsymbol{\theta}) = \boldsymbol{\Phi}_2(\boldsymbol{z}|\mathbf{0}, \Sigma). \tag{3.121}$$

where z_i is given by (3.120). Let $\xi_i = \xi_i(\alpha_i, \rho) = 2(1-\rho^2)\alpha_i^2$ for $i = 1, 2$. Then more algebra shows

$$\begin{aligned} E(T_1^m T_2^n) &= \int_0^\infty \int_0^\infty t_1^m t_2^n f(t_1, t_2|\boldsymbol{\theta}) dt_1 dt_2 \\ &\approx \int_{L_1}^{U_1} \int_{L_2}^{U_2} t_1^m t_2^n \boldsymbol{\phi}_2(\boldsymbol{z}|\mathbf{0}, \Sigma) \prod_{i=1}^{2} \frac{1}{2\alpha_i\beta_i} \Big[\Big(\frac{\beta_i}{t_i}\Big)^{\frac{1}{2}} + \Big(\frac{\beta_i}{t_i}\Big)^{\frac{3}{2}} \Big] dt_i, \end{aligned} \tag{3.122}$$

where the quantities L_i and U_i are chosen in such a way as to constitute on a region beyond which integrand in (3.122) is sufficiently small. To this end, we set

$$L_i = \Big\{ t_i \in \mathbb{R}^+ \Big| \frac{1}{\alpha_i} \Big[\sqrt{\frac{t_i}{\beta_i}} - \sqrt{\frac{\beta_i}{t_i}} \Big] = -\eta \Big\}, \tag{3.123}$$

$$U_i = \Big\{ t_i \in \mathbb{R}^+ \Big| \frac{1}{\alpha_i} \Big[\sqrt{\frac{t_i}{\beta_i}} - \sqrt{\frac{\beta_i}{t_i}} \Big] = \eta \Big\}, \tag{3.124}$$

where η is a positive constant. Since z_i is an increasing function of t_i, then by solving non-linear equations in (3.123)-(3.124), we obtain

$$L_i = -\eta\alpha_i\beta_i \Big[\frac{\eta}{2}\alpha_i + \frac{1}{2}\sqrt{\eta^2\alpha_i^2 + 4} \Big] + \beta_i(\eta^2\alpha_i^2 + 1), \tag{3.125}$$

$$U_i = -\eta\alpha_i\beta_i \Big[\frac{\eta}{2}\alpha_i - \frac{1}{2}\sqrt{\eta^2\alpha_i^2 + 4} \Big] + \beta_i(\eta^2\alpha_i^2 + 1). \tag{3.126}$$

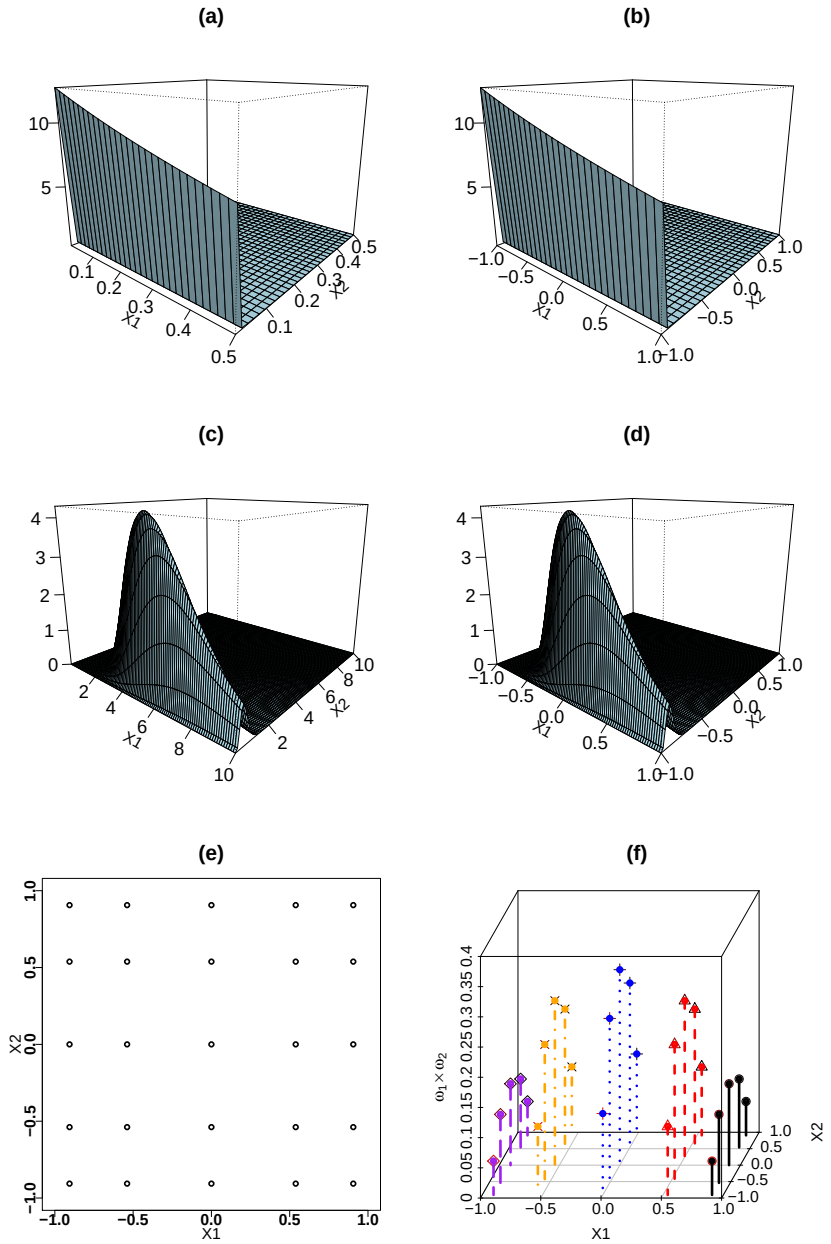


Figure 3.8: (a): Graph of $g(x_1, x_2|\boldsymbol{\theta} = (0.5, 1, 0.5)^\top)$ on region $[0, 0.5] \times [0, 0.5]$, (b): Graph of $g(x_1, x_2|\boldsymbol{\theta} = (0.5, 1, 0.5)^\top)$ transformed to region $[-1, 1] \times [-1, 1]$, (c): Graph of $g(x_1, x_2|\boldsymbol{\theta} = (5, 1, 5)^\top)$ on region $[0, 10] \times [0, 10]$, (b): Graph of $g(x_1, x_2|\boldsymbol{\theta} = (5, 1, 5)^\top)$ transformed to region $[-1, 1] \times [-1, 1]$, (e): Coordinates of nodes computed using Gauss-Legendre rule with $k_1 = k_2 = 5$ points, and (f): Product of weights computed using Gauss-Legendre rule with $k_1 = k_2 = 5$ points.

Herein, we set $\eta = 6$ that guarantees the joint Gaussian PDF, and hence integrand in the RHS of (3.122), is sufficiently small. Using an appropriate transformation of the form $y_i = 2(t_i - L_i)/(U_i - L_i) - 1$, for $i = 1, 2$, then the RHS of (3.122) becomes

$$E(T_1^m T_2^n) \approx \prod_{i=1}^{2} \frac{U_i - L_i}{4\alpha_i \beta_i} \times \int_{-1}^{1} \int_{-1}^{1} \eta_1^m \eta_2^n \phi_2(\boldsymbol{y}|\mathbf{0}, \Sigma) \Big[\Big(\frac{\beta_i}{y_i}\Big)^{\frac{1}{2}} + \Big(\frac{\beta_i}{y_i}\Big)^{\frac{3}{2}} \Big] dy_1 dy_2, \tag{3.127}$$

where $\eta_i = (y_i + 1)(U_i - L_i)/2 + L_i$ for $i = 1, 2$. The RHS of (3.127) can be approximated through a k-point Gauss- Legendre rule. Therefore

$$E(T_1^m T_2^n) \approx \prod_{r=1}^{2} \frac{U_r - L_r}{4\alpha_r\beta_r} \times \sum_{i=1}^{k}\sum_{j=1}^{k} \omega_{1i}\omega_{2j}\Big[\frac{1}{2}(y_{1i}+1)(U_1 - L_1) + L_1\Big]^m \times \Big[\frac{1}{2}(y_{2j}+1)(U_2 - L_2) + L_2\Big]^n \boldsymbol{\phi}_2\Big((y_{1i}, y_{2j})^\top \Big| \mathbf{0}, \Sigma\Big) \times \Big[\Big(\frac{\beta_1}{y_{1i}}\Big)^{\frac{1}{2}} + \Big(\frac{\beta_1}{y_{1i}}\Big)^{\frac{3}{2}}\Big]\Big[\Big(\frac{\beta_2}{y_{2j}}\Big)^{\frac{1}{2}} + \Big(\frac{\beta_2}{y_{2j}}\Big)^{\frac{3}{2}}\Big], \tag{3.128}$$

where ω_{rs} and y_{rs} are, accordingly, the sth (for $s = 1, \cdots, k$) weight and node for the rth coordinate (for $r = 1, 2$) computed based on a k-point Gauss-Legendre rule. A simulation study have been carried out to check the performance of approximation (3.128). The results are shown in Table 3.9. We notice that when $\rho \neq 0$, then the Gauss-Legendre rule works well only when k is large. ■

Table 3.9: Results for approximating $E(T_1^2 T_2^2)$ using k-point Gauss-Legendre rule.

$\rho = 0.99$				Approximation	
		Exact value[1]	$k = 30$	$k = 60$	$k = 120$
$\alpha_1 = 0.2$	$\alpha_2 = 0.2$	1.36981507	2.19135117	1.39966739	1.36981553
$\alpha_1 = 2.0$	$\alpha_2 = 0.2$	74.3634327	74.53853374	74.35577417	74.36313027
$\alpha_1 = 2.0$	$\alpha_2 = 2.0$	17422.2127	17724.84078171	17421.54059427	17421.52544734
$\rho = 0.50$					
$\alpha_1 = 0.2$	$\alpha_2 = 0.2$	1.26793909	1.26793903	1.26793903	1.26793903
$\alpha_1 = 2.0$	$\alpha_2 = 0.2$	52.2184563	52.21790783	52.21827096	52.21835271
$\alpha_1 = 2.0$	$\alpha_2 = 2.0$	5412.76146	5412.62822259	5412.65315429	5412.65234609
$\rho = 0.00$					
$\alpha_1 = 0.2$	$\alpha_2 = 0.2$	1.17158976	1.17158974	1.17158974	1.17158974
$\alpha_1 = 2.0$	$\alpha_2 = 0.2$	35.7191999	35.71896967	35.71904935	35.71917926
$\alpha_1 = 2.0$	$\alpha_2 = 2.0$	1088.99999	1088.98597622	1088.99083451	1088.99875603

[1] The exact value of $E(T_1^2 T_2^2)$ has been computed using `Maple`.

Example 3-14

Let $\mathcal{G}(\alpha, \beta)$ denote the family of gamma distributions with shape parameter α and rate parameter β. The PDF of $\mathcal{G}(\alpha, \beta)$ is then given by

$$f(x|\boldsymbol{\theta}) = \frac{\beta^\alpha}{\Gamma(\alpha)} x^{\alpha-1} \exp\{-\beta x\}, \tag{3.129}$$

where $\boldsymbol{\theta} = (\alpha, \beta)^\top$. Based on a sample of n realizations such as $\boldsymbol{x} = \{x_1, x_2, \cdots, x_n\}$, the likelihood function becomes

$$f(\boldsymbol{x}|\boldsymbol{\theta}) = \left[\frac{\beta^\alpha}{\Gamma(\alpha)}\right]^n \left[\prod_{i=1}^{n} x_i\right]^{\alpha-1} \exp\left\{-\beta \sum_{i=1}^{n} x_i\right\}.$$

It is not hard to check that the Fisher information matrix corresponds to a gamma distribution $\mathcal{G}(\alpha, \beta)$ becomes

$$I(\boldsymbol{\theta}) = \begin{bmatrix} \Psi(1, \alpha) & -\frac{1}{\beta} \\ -\frac{1}{\beta} & \frac{\alpha}{\beta^2} \end{bmatrix}, \tag{3.130}$$

where

$$\Psi(n,x) = \frac{\partial^{n+1}}{\partial x^{n+1}} \log \Gamma(x),$$

is the well-known polygamma function. The most commonly used polygamma functions are digamma and trigamma denoted as $\Psi(0,x) = \Psi(x)$ and $\Psi(1,x)$, respectively. Simple argument shows that

$$\pi_J(\boldsymbol{\theta}) \propto \sqrt{\det I(\boldsymbol{\theta})} = \frac{1}{\beta}\sqrt{1+\alpha\Psi(1,\alpha)}. \tag{3.131}$$

Therefore the posterior PDF is then given by

$$\begin{aligned}\pi(\boldsymbol{\theta}|\boldsymbol{x}) &= \frac{f(\boldsymbol{x}|\boldsymbol{\theta})\pi_J(\boldsymbol{\theta})}{\int_0^\infty \int_0^\infty f(\boldsymbol{x}|\boldsymbol{\theta})\pi_J(\boldsymbol{\theta})d\alpha d\beta} \\ &= \frac{h(\alpha,\beta|\boldsymbol{x})}{\int_0^\infty \int_0^\infty h(\alpha,\beta|\boldsymbol{x})d\alpha d\beta},\end{aligned}$$

where the real function $h(\alpha,\beta \mid \boldsymbol{x}) = f(\boldsymbol{x} \mid \boldsymbol{\theta})\pi_J(\boldsymbol{\theta})$ is given by

$$h(\alpha,\beta|\boldsymbol{x}) = \sqrt{1+\alpha\Psi(1,\alpha)}\frac{\beta^{n\alpha-1}}{[\Gamma(\alpha)]^n}\left[\prod_{i=1}^n x_i\right]^{\alpha-1}\exp\left\{-\beta\sum_{i=1}^n x_i\right\}. \tag{3.132}$$

It is well known that under squared loss function the Bayesian estimator $\hat{\boldsymbol{\theta}}_B$, of $\boldsymbol{\theta}$ is the average of the posterior distribution. This implies

$$\hat{\boldsymbol{\theta}}_B = E_{\pi(\boldsymbol{\theta}|\boldsymbol{x})}(\boldsymbol{\theta}) = \Big(\frac{I_{10}}{I_{00}}, \frac{I_{01}}{I_{00}}\Big)^\top,$$

where

$$I_{rs} = \int_0^\infty \int_0^\infty \alpha^r \beta^s h(\alpha,\beta|\boldsymbol{x})d\alpha d\beta, \tag{3.133}$$

for $r,s = \{0,1\}$. Using a k-point Gauss-Legendre rule, then for approximating I_{10}, I_{01}, and I_{00}, we proceed as follows. Let $\hat{\boldsymbol{\theta}}_{ML} = (\hat{\alpha}_{ML}, \hat{\beta}_{ML})^\top$ denote the ML estimator of the unknown parameter vector $\boldsymbol{\theta} = (\alpha,\beta)^\top$, then based on sample evidence $\boldsymbol{x}$, we define

$$\sigma_{\hat{\alpha}_{ML}} = \frac{3}{\sqrt{n}}\sqrt{I^{-1}_{1,1}(\hat{\boldsymbol{\theta}}_{ML})},$$

where $I^{-1}_{i,j}(\boldsymbol{\theta})$ denotes the (i,j) th entry of the inverse of Fisher information matrix given by (3.130). More algebra shows

$$\sigma_{\hat{\alpha}_{ML}} = \frac{3}{\sqrt{n}}\frac{\sqrt{\hat{\alpha}_{ML}}}{\sqrt{\Psi(1,\hat{\alpha}_{ML})\hat{\alpha}_{ML}-1}}.$$

For our purposes, we set $L_{\hat{\alpha}} = \max\{0, \hat{\alpha}_{ML} - \sigma_{\hat{\alpha}_{ML}}\}$, and $U_{\hat{\alpha}} = \hat{\alpha}_{ML} + \sigma_{\hat{\alpha}_{ML}}$. It is worth some discussion on the posterior distribution of β given α and $\boldsymbol{x}$. It follows from (3.132) that $\pi^* = \pi(\beta \mid \alpha, \boldsymbol{x}) \sim \mathcal{G}(n\alpha, n\bar{x})$, then we let

$$\sigma_{\hat{\beta}_{ML}} = 3 \times \sqrt{\mathrm{var}_{\pi^*}(\beta)} = \frac{3}{\sqrt{n}}\frac{\sqrt{\hat{\alpha}_{ML}}}{\bar{x}}.$$

Using the fact that $E_{\pi^*}(\beta) = \alpha/\bar{x}$, we define $L_{\hat{\beta}} = \max\left\{0, E_{\pi^*}(\beta) - \sigma_{\hat{\beta}_{ML}}\right\}$ and $U_{\hat{\beta}} = E_{\pi^*}(\beta) + \sigma_{\hat{\beta}_{ML}}$. If $n \times \hat{\alpha}_{ML} > 1$, then we set $U_{\hat{\beta}}$ to be the 0.9999th quantile of π^*. Thus, I_{rs} is approximated as

$$\begin{aligned} I_{rs} &= \int_0^\infty \int_0^\infty \alpha^r \beta^s h(\alpha, \beta|\boldsymbol{x}) d\alpha d\beta \\ &\approx \int_{L_{\hat{\beta}}}^{U_{\hat{\beta}}} \int_{L_{\hat{\alpha}}}^{U_{\hat{\alpha}}} \alpha^r \beta^s h(\alpha, \beta|\boldsymbol{x}) d\alpha d\beta. \end{aligned}$$

Define $R_{\hat{\alpha}} = U_{\hat{\alpha}} - L_{\hat{\alpha}}$ and $R_{\hat{\beta}} = U_{\hat{\beta}} - L_{\hat{\beta}}$, then using two suitable change of variables of the form $z_1 = 2(\alpha - L_{\hat{\alpha}})/R_{\hat{\alpha}} - 1$ and $z_2 = 2(\beta - L_{\hat{\beta}})/R_{\hat{\beta}} - 1$, it follows that

$$\begin{aligned} I_{rs} \approx & \frac{R_{\hat{\alpha}} R_{\hat{\beta}}}{4} \int_{-1}^{1} \int_{-1}^{1} \left[\frac{1}{2}(z_1+1)R_{\hat{\alpha}} + L_{\hat{\alpha}}\right]^r \left[\frac{1}{2}(z_2+1)R_{\hat{\beta}} + L_{\hat{\beta}}\right]^s \\ & \times h\left[\frac{1}{2}(z_1+1)R_{\hat{\alpha}} + L_{\hat{\alpha}}, \frac{1}{2}(z_2+1)R_{\hat{\beta}} + L_{\hat{\beta}} \middle| \boldsymbol{x}\right] dz_1 dz_2. \end{aligned} \tag{3.134}$$

Based on a k-point Gauss-Legendre rule, an approximation of I_{rs} in (3.134), is then given by

$$\begin{aligned} I_{rs} \approx & \frac{R_{\hat{\alpha}} R_{\hat{\beta}}}{4} \sum_{i=1}^{k} \sum_{j=1}^{k} \omega_{1i}\omega_{2j} \left[\frac{1}{2}(y_{1i}+1)R_{\hat{\alpha}} + L_{\hat{\alpha}}\right]^i \left[\frac{1}{2}(y_{2j}+1)R_{\hat{\beta}} + L_{\hat{\beta}}\right]^j \\ & \times h\left[\frac{1}{2}(y_{1i}+1)R_{\hat{\alpha}} + L_{\hat{\alpha}}, \frac{1}{2}(y_{2j}+1)R_{\hat{\beta}} + L_{\hat{\beta}} \middle| \boldsymbol{x}\right], \end{aligned} \tag{3.135}$$

where ω_{ij} and y_{ij} for $i = 1, \cdots, k$ and $j = 1, 2$ are, respectively, the weight and node associated with the s th coordinate based on a k-point Gauss-Legendre rule. It is worth to note that when sample size is large, then both of the ML and Bayesian paradigms yield the same estimations. The `R` code below computes $\hat{\boldsymbol{\theta}}_B$ under a squared loss function and based on the Jeffreys prior (3.131) is given below. It should be noted that arguments `alpha0` and `beta0` in function `Bayes_gamma_quad` are the initial (ML) estimators of α and β for constructing $L_{\hat{\alpha}}, U_{\hat{\alpha}}, L_{\hat{\beta}}$, and $U_{\hat{\beta}}$.

```
R> Bayes_gamma_quad <- function(x, k, alpha0, beta0)
+{
+ n <- length(x)
+ y <- matrix(0, nrow = k^2, ncol = 2)
+ x_bar <- mean(x)
+ sigma_alpha <- 3/sqrt(n)*sqrt(alpha0)/sqrt(alpha0*trigamma(alpha0)- 1)
+ L_alpha <- max(0, alpha0- sigma_alpha[1] )
+ U_alpha <- alpha0 + sigma_alpha[1]
+ sigma_beta <- 3/sqrt(n)*sqrt(alpha0)/x_bar
+ L_beta <- max(0, alpha0/x_bar- sigma_beta )
+ if(n*alpha0 > 1)
+ {
+  U_beta <- alpha0/x_bar + sigma_beta
+ }else{
+  U_beta <- qgamma(0.9999, shape = n*alpha0, rate = n*x_bar )
+ }
+ R_alpha <- U_alpha- L_alpha
+ R_beta <- U_beta- L_beta
+ out <- quad_rule(k, type = "LE", alpha = 0, beta = 0)
+ node <- out$node
+ weight <- out$weight
+ node_grid <- as.data.frame( expand.grid(node, node) )
+ weight_grid <- as.data.frame( expand.grid(weight, weight) )
+ omega <- rowSums( weight_grid )
+ y[, 1] <- (node_grid[, 1] + 1)*R_alpha/2 + L_alpha
```

```
+ y[, 2] <- (node_grid[, 2] + 1)*R_beta /2 + L_beta
+ h_theta <- log(omega) + 1/2*log( 1 + y[, 1]*trigamma(y[, 1]) ) +
+ (n*y[, 1]- 1)*log(y[, 2])- n*lgamma(y[, 1])
+ n*y[, 2]*x_bar + (y[, 1]- 1)*sum( log( x ) )
+ I00 <- sum( exp( h_theta- mean( h_theta) ) )
+ if( I00 == 0 | I00 == Inf )
+ {
+  h_theta <- 0
+  I00 <- k^2
+ }
+ I10 <- sum( exp( log(y[, 1]) + h_theta- mean( h_theta) ) )
+ I01 <- sum( exp( log(y[, 2]) + h_theta- mean( h_theta) ) )
+ return( ls = list( "alpha_bayes" = I10/I00, "beta_bayes" = I01/I00 ) )
+}
```

To assess how well a gamma distribution works in modelling a data set, we provide an R function called GOF [1] for computing some goodness-of-fit measures including Anderson Darling (AD) Anderson and Darling [Anderson and Darling, 1954], Akaike information criterion (AIC) [Akaike, 2003], Bayesian information criterion (BIC) [Schwarz et al., 1978], Cramér-von Misses (CVM) [Cramér, 1928], log-likelihood, and the Kolmogorov-Smirnov (KS) statistics with corresponding p-value. Some of these statistics have been used in Sub-section 1.6.3 for testing normality. If $x_1, \cdots, x_n$ independently and identically come from the CDF $F(\cdot|\boldsymbol{\theta})$ whose PDF is $f(\cdot|\boldsymbol{\theta})$, then above statistics are defined as follows.

$$\text{AD} = -n - \sum_{i=1}^{n}\Big(\frac{2i-1}{n}\Big)\Big\{\log\Big[F(x_{(i)}|\boldsymbol{\theta})\Big] + \log\Big[1 - F(x_{(n-i+1)}|\boldsymbol{\theta})\Big]\Big\}, \tag{3.136}$$

$$\text{AIC} = -2\sum_{i=1}^{n}\log\Big[f(x_i|\boldsymbol{\theta})\Big] + 2p, \tag{3.137}$$

$$\text{BIC} = -2\sum_{i=1}^{n}\log\Big[f(x_i|\boldsymbol{\theta})\Big] + p\log(n), \tag{3.138}$$

$$\text{CVM} = \frac{1}{12n} + \sum_{i=1}^{n}\Big[F(x_{(i)}|\boldsymbol{\theta}) - \frac{2i-1}{2n}\Big]^2, \tag{3.139}$$

$$\text{KS} = \sup_{1\le i\le n}\Big|F(x_i|\boldsymbol{\theta}) - F_n(x_i)\Big|, \tag{3.140}$$

where p is the number of family parameters (number of unknown elements of $\boldsymbol{\theta}$), $x_{(i)}$ denotes the ith ordered observation within sample $\boldsymbol{x}$, and $F_n(\cdot)$ is the empirical CDF of sample $\boldsymbol{x}$, that is, $F_n(x) = r/n$ if exactly r scores are less or equal than x (for $r = 0, \cdots, n$). We note that the smaller value for statistics given in (3.136)-(3.140) indicates a better model and, in practice, $\boldsymbol{\theta}$ is substituted with its estimator. [2]

```
R> GOF <- function(x, theta)
+{
+ n <- length(x)
+ index <- 1:n
+ sort_x <- sort(x)
+ n_theta <- length(theta)
+ log_pdf <- dgamma(sort_x, shape = theta[1], rate = theta[2], log = T)
+ cdf <- pgamma(sort_x, shape = theta[1], rate = theta[2])
+ log_cdf <- pgamma(sort_x, shape = theta[1], rate = theta[2], log = T)
```

[1] The function GOF can be modified for computing the GOF statistics when other well-known distributions are fitted to data set. For example, if one would rather to have a Weibull model instead of gamma, it just needs to change expression pgamma in lines 7, 8, 9, 10, and 17 to pweibull.

[2] If a GOF statistic involves a hypothesis testing, such as KS test, replacing $\boldsymbol{\theta}$ with $\hat{\boldsymbol{\theta}}$ would affect the corresponding critical values specially when n is small [Lilliefors, 1967, Shorack and Wellner, 2009]. Although the ML estimator is biased for small n, but in the case of substituting, it is suggested.

```
+ log_survival <- pgamma(sort_x, shape = theta[1], rate = theta[2], log = T, lower.tail = F)
+ log.like <- sum( log_pdf )
+ cdf <- ifelse(cdf == 1, .Machine$double.eps, cdf)
+ AD <- -n - mean( (2*index - 1)*log_cdf + (2*n + 1 - 2*index)*log_survival )
+ AIC <- -2*log.like + 2*n_theta
+ BIC <- -2*log.like + n_theta*log(n)
+ CVM <- sum( ( cdf - (2*index - 1)/(2*n) )^2 ) + 1/(12*n)
+ KS <- suppressWarnings( ks.test(sort_x, "pgamma", theta[1], theta[2]) )
+ out1 <- cbind(AD, AIC, BIC, CVM, log.like)
+ colnames(out1) <- c("AD", "AIC", "BIC", "CVM", "log.likelihood")
+ rownames(out1) <- c("␣")
+ out2 <- cbind(KS$statistic, KS$p.value)
+ colnames(out2) <- c("statistics", "p-value")
+ rownames(out2) <- c("␣")
+ list( "GOF␣statistics" = out1, "Kolmogorov-Smirnov␣test" = out2 )
+}
```

As an illustration, we focus on fatigue life of 10 bearings of a certain type reported by McCool [1974] that is presented in Table 3.10. Different distributions such as BS [Ng et al., 2003] and three-parameter Weibull [Cohen et al., 1984] were fitted to this data set. Herein, we would like to fit a gamma distribution to these observations and then estimate its parameters using the ML method and Bayesian paradigm. To this end, first, we use command `fitdist(x, "gamma", method = "mle")` existing in `R` package `fitdistrplus` to compute the ML estimator $\hat{\boldsymbol{\theta}}_{ML}$, of $\boldsymbol{\theta}$ and then compute $\hat{\boldsymbol{\theta}}_B$ using a 20-point Gauss-Legendre rule for which $\hat{\boldsymbol{\theta}}_{ML}$ is taken as the initial estimate. The results are shown in Table 3.11. Indeed if we accept that observations in Table 3.10 independently follow a gamma distribution, then the $\hat{\boldsymbol{\theta}}_{ML}$ slightly outperforms $\hat{\boldsymbol{\theta}}_B$ in terms of all GOF statistics given in Table 3.11. ■

Table 3.10: Fatigue life of 10 bearings of a certain type (in hours).

152.7	204.7	172.0	216.5	172.5	234.9	173.3	262.6	193.0	422.6

Table 3.11: Estimators and associated GOF statistics for gamma distribution fitted to to fatigue life data given in Table 3.10.

	parameter		GOF statistic					
	α	β	AD	AIC	BIC	CVM	KS	p-value
$\hat{\boldsymbol{\theta}}_{ML}$	11.5619	0.0524	0.6813	115.2275	115.8327	0.0987	0.1853	0.8222
$\hat{\boldsymbol{\theta}}_B$	11.4344	0.0516	0.6967	115.2300	115.8352	0.1041	0.1904	0.7971

Example 3-15

Suppose we are interested in computing double integral

$$I(\boldsymbol{\theta}) = \int_a^b \int_{c(x_1)}^{d(x_1)} f(x_1, x_2|\boldsymbol{\theta}) dx_2 dx_1. \tag{3.141}$$

For approximating $I(\boldsymbol{\theta})$, we may proceed based on a Gauss-Legendre rule as follows. First we apply a k_1-point Gauss-Legendre rule for inner integral as

$$\begin{aligned}\int_{c(x_1)}^{d(x_1)} f(x_1, x_2|\boldsymbol{\theta}) dx_2 &= \int_{-1}^{1} f\Big[x_1, \frac{(x_2+1)\big(d(x_1)-c(x_1)\big)}{2} + c(x_1)\Big|\boldsymbol{\theta}\Big] dx_2 \\ &\approx \frac{d(x_1)-c(x_1)}{2} \sum_{j=1}^{k_2} \omega_{2j} f\Big[x_1, \frac{(x_{2j}+1)\big(d(x_1)-c(x_1)\big)}{2} + c(x_1)\Big|\boldsymbol{\theta}\Big],\end{aligned} \tag{3.142}$$

where ω_{2j} and x_{2j}, for $j = 1, \cdots, k_2$ are, accordingly, the weights and nodes constructed based on a k_2-point Gauss-Legendre rule. Then, we complete computing $I(\boldsymbol{\theta})$ by approximating the outer integral as follows.

$$\begin{aligned} I(\boldsymbol{\theta}) \approx & \frac{b-a}{4} \sum_{i=1}^{k_1} \sum_{j=1}^{k_2} \omega_{1i}\omega_{2j} \big[d(x_{1i}) - c(x_{1i})\big] \\ & \times f\Big[\frac{(x_{1i}+1)(b-a)}{2} + a, \frac{(x_{2j}+1)\big(d(x_{1i}) - c(x_{1i})\big)}{2} + c(x_{1i}) \Big| \boldsymbol{\theta}\Big], \end{aligned} \tag{3.143}$$

where ω_{1i} and x_{1i}, for $i = 1, \cdots, k_1$ are, accordingly, the weights and nodes constructed based on a k_1-point Gauss-Legendre rule. ■

Example 3-16

Indeed, computing the probability of a bivariate Gaussian distribution is simply expressed in terms of $I(\boldsymbol{\theta})$ given in (3.141). Let $\boldsymbol{X} = (X_1, X_2)^\top$ follow a Gaussian distribution with mean $\boldsymbol{\mu} = (\mu_1, \mu_2)^\top$ and covariance matrix $\Sigma = \big[(\Sigma_{1,1}, \Sigma_{1,2})^\top, (\Sigma_{2,1}, \Sigma_{2,2})^\top\big]$. Recall from Subsection 3.5 that discusses computing $\mathcal{P}_G = P(\boldsymbol{a} \leq \boldsymbol{X} \leq \boldsymbol{b})$ using a transformation $\boldsymbol{y} = L(\boldsymbol{x} - \boldsymbol{\mu})$ in which L denotes the Cholesky decomposition of Σ. We have

$$\mathcal{P}_G = \frac{1}{2\pi} \int_{\frac{a_1-\mu_1}{L_{1,1}}}^{\frac{b_1-\mu_1}{L_{1,1}}} \int_{c(y_1)}^{d(y_1)} \exp\Big\{-\frac{y_1^2 + y_2^2}{2}\Big\} dy_2 dy_1, \tag{3.144}$$

where $\boldsymbol{a} = (a_1, a_2)^\top$, $\boldsymbol{b} = (b_1, b_2)^\top$, and

$$\begin{aligned} c(y_1) &= \frac{a_2 - \mu_2 - L_{2,1} y_1}{L_{2,2}}, \\ d(y_1) &= \frac{b_2 - \mu_2 - L_{2,1} y_1}{L_{2,2}}. \end{aligned}$$

Following (3-15) for approximating the RHS of (3.144), it turns out that

$$\begin{aligned} \mathcal{P}_G \approx & \frac{(b_1 - a_1)(b_2 - a_2)}{4 L_{1,1} L_{2,2}} \sum_{i=1}^{k_1} \sum_{j=1}^{k_2} \omega_{1i}\omega_{2j} \exp\Big\{-\frac{z_{1i}^2 + z_{2j}^2}{2}\Big\}, \\ = & \sum_{i=1}^{k_1} \sum_{j=1}^{k_2} \omega_{1i}\omega_{2j} g(z_{1i}, z_{2i}), \end{aligned} \tag{3.145}$$

where

$$\begin{aligned} z_{1i} = & \frac{(x_{1i}+1)(b_1 - a_1)}{2 L_{1,1}} + \frac{a_1 - \mu_1}{L_{1,1}}, \\ z_{2j} = & \frac{(x_{2j}+1)(b_2 - a_2)}{2 L_{2,2}} + \frac{a_2 - \mu_2 - L_{2,1} z_{1i}}{L_{2,2}}. \end{aligned}$$

The `R` code below approximates $\mathcal{P}_G$ given in (3.144) using a k-point Gauss-Legendre rule using Cholesky decomposition of covariance matrix.

```
R> Chol_quad <- function(a, b, Mu, Sigma, k)
+{
+ x <- matrix(0, nrow = k^2, ncol = 2)
+ ch <- t( chol(Sigma) )
+ threshold <- 5*sqrt( diag(Sigma) )
```

```
+ a0 <- a - Mu
+ b0 <- b - Mu
+ a0 <- ifelse( a0 < -threshold, -threshold, a0 )
+ b0 <- ifelse( b0 > threshold, threshold, b0 )
+ f1 <- function(x) ( x + 1 )*(b0[1] - a0[1])/(2*ch[1,1]) + a0[1]/ch[1,1]
+ f2 <- function(x, y) 1/(2*pi)*exp( -x^2/2 - y^2/2 )
+ out <- quad_rule(k, type = "LE", alpha = 0, beta = 0)
+ weight <- apply( expand.grid(out$weight, out$weight), 1, prod )
+ x1 <- f1( out$node )
+ x2 <- out$node
+ x[, 1]<- rep(x1, each = k)
+ x[, 2] <- as.vector( t( sapply(1:k, function(j) (x2[j] + 1)*(b0[2] - a0[2])/
+ (2*ch[2,2]) + (a0[2] - ch[2,1]*x1)/ch[2,2] ) ) )
+ I1 <- (b0[1] - a0[1])*(b0[2] - a0[2])/( 4*ch[1,1]*ch[2,2] )*
+ sum( weight*f2(x[, 1], x[, 2]) )
+ min( abs(I1), 1)
+}
```

3.8 Computing the CDF of multivariate Gaussian distribution using Gaussian Quadrature

Recall form Subsection 3.5 that computes the CDF of Gaussian distribution using SOV technique proposed by Genz [1992]. Herein, we are willing to compute $\mathcal{P}_G$ through the Gaussian quadrature based on spectral decomposition (2.75) of covariance matrix.

3.8.1 Computing the CDF of bivariate Gaussian distribution

Computing bivariate Gaussian probability has a long history. Herein, we describe briefly methods proposed by Drezner and Wesolowsky [1990]. First note that [Sheppard and Pearson, 1900] shows that each bivariate Gaussian probability on region $[x, \infty) \times [y, \infty)$ represented as

$$L(x, y|\rho) = \frac{1}{2\pi\sqrt{1-\rho^2}} \int_x^\infty \int_y^\infty \exp\Big\{-\frac{u^2 + v^2 - 2uv\rho}{2(1-\rho^2)}\Big\} dvdu, \tag{3.146}$$

where $|\rho| \leq 1$ reduces to one-dimensional integral as

$$L(x, y|\rho) = \frac{1}{2\pi} \int_{\cos^{-1}(\rho)}^\pi \exp\Big\{-\frac{x^2 + y^2 - 2xy\cos(z)}{2\sin^2(z)}\Big\} dz. \tag{3.147}$$

Differentiating (3.147) with respect to $\rho > 0$ yields [Plackett, 1954]:

$$\frac{\partial L(x, y|\rho)}{\partial \rho} = \frac{1}{2\pi\sqrt{1-\rho^2}} \exp\Big\{-\frac{x^2 + y^2 - 2xy\rho}{2(1-\rho^2)}\Big\}. \tag{3.148}$$

Based on the Plackett's formula (3.148), a two-step procedure for computing (3.146) has been introduced by Drezner and Wesolowsky [1990] as follows.

- **First step** ($|\rho| < 0.7$): Suppose $\rho > 0$. Using the fact that

$$\int_0^\rho \frac{\partial L(x, y|r)}{\partial r} dr = L(x, y|\rho) - L(x, y|0),$$

or

$$L(x, y|\rho) = L(x, y|0) + \int_0^\rho \frac{\partial L(x, y|r)}{\partial r} dr \tag{3.149}$$

It is immediate from (3.146) that $L(x, y|0) = \Phi(-x)\Phi(-y)$. Substituting $\partial L(x, y|\rho)/\partial\rho$ from (3.148) into the RHS of (3.149) it follows that

$$L(x, y|\rho) = \Phi(-x)\Phi(-y) + \frac{1}{2\pi}\int_0^\rho \frac{1}{\sqrt{1-r^2}} \exp\Big\{-\frac{x^2+y^2-2xyr}{2(1-r^2)}\Big\}dr.$$

If ρ is negative, then a simple change of variable shows

$$L(x, y|\rho) = \Phi(-x)\Phi(-y) + \frac{s}{2\pi}\int_0^{|\rho|} \frac{1}{\sqrt{1-r^2}} \exp\Big\{-\frac{x^2+y^2-2xsyr}{2(1-r^2)}\Big\}dr, \tag{3.150}$$

where $s = \text{sign}(\rho)$. Drezner and Wesolowsky [1990] suggest to approximate (3.150) using a 5-pint Gauss-Legendre rule. They further used k-pint Gauss-Legendre rule (for $k = \{2, 3, 5, 10\}$) and demonstrated that the results are more accurate than that of [Mee and Owen, 1983] for $|\rho| \leq 0.9$.

- **Second step** ($|\rho| \geq 0.7$): Even though the first step works well, but it can be modified when ρ is large ($|\rho| \geq 0.7$, say). For computational purpose, we consider the fact that

$$\int_\rho^s \frac{\partial L(x, y|r)}{\partial r}dr = L(x, y|s) - L(x, y|\rho), \tag{3.151}$$

Rearranging (3.151) and then substituting integrand $\partial L(x, y|r)/\partial r$ from (3.148) we get

$$\begin{aligned} L(x, y|\rho) =& L(x, y|s) - \int_\rho^s \frac{\partial L(x, y|r)}{\partial r}dr \\ =& L(x, y|s) - \frac{1}{2\pi}\int_\rho^s \frac{1}{\sqrt{1-r^2}} \exp\Big\{-\frac{x^2+y^2-2xyr}{2(1-r^2)}\Big\}dr. \end{aligned} \tag{3.152}$$

The RHS of (3.152) can be evidently represented as

$$L(x, y|\rho) = L(x, y|s) - \frac{s}{2\pi}\int_{|\rho|}^1 \frac{1}{\sqrt{1-r^2}} \exp\Big\{-\frac{x^2+y^2-2xsyr}{2(1-r^2)}\Big\}dr, \tag{3.153}$$

and a change of variable $z = \sqrt{1-r^2}$ turns out that

$$L(x, y|\rho) = L(x, y|s) - \frac{s}{2\pi}\int_0^{\sqrt{1-\rho^2}} \frac{1}{\sqrt{1-z^2}} \exp\Big\{-\frac{x^2+y^2-2xsy\sqrt{1-z^2}}{2z^2}\Big\}dz. \tag{3.154}$$

A 5-point Gauss-Legendre rule for computing integral in (3.154) yields error, in most cases, less than 10^{-7} when $|\rho| \geq 0.7$. As pointed out by Drezner and Wesolowsky [1990], more refinements is obtained when $|\rho|$ is large and x is close to sy (but not equal to sy). To this end, we rewrite (3.154) as

$$L(x, y|\rho) = L(x, y|s) - \frac{s}{2\pi}\int_0^\tau \exp\Big\{-\frac{\eta^2}{2z^2}\Big\}\frac{1}{\sqrt{1-z^2}} \exp\Big\{-\frac{xsy}{1+\sqrt{1-z^2}}\Big\}dz, \tag{3.155}$$

where $\eta = |x - sy|$ and $\tau = \sqrt{1-\rho^2}$. If x is close to sy (but not equal to sy) and $|\rho|$ is large (τ is small), then the term $\exp\{-(x-sy)^2/(2z^2)\}$ in RHS of (3.155) is the source of large errors. As it is seen from (3.155) if τ is small, then z is near zero. Using the Taylor expansion

$$\frac{1}{\sqrt{1-z^2}} \exp\Big\{-\frac{xsy}{1+\sqrt{1-z^2}}\Big\} = \exp\Big\{-\frac{xsy}{2}\Big\}[1 + \lambda z^2] + \mathcal{O}(z^4), \tag{3.156}$$

in which $\lambda = (4 - xsy)/8$, the integral in RHS of (3.155) can be written as

$$\begin{aligned}
&\int_0^\tau \exp\Big\{-\frac{\eta^2}{2z^2} - \frac{xsy}{2}\Big\}[1+\lambda z^2]dz + \int_0^\tau \exp\Big\{-\frac{\eta^2}{2z^2}\Big\}\mathcal{O}(z^4)\Big]dz \\
=&\int_0^\tau \exp\Big\{-\frac{\eta^2}{2z^2} - \frac{xsy}{2}\Big\}[1+\lambda z^2]dz \\
&+ \int_0^\tau \frac{1}{\sqrt{1-z^2}} \exp\Big\{-\frac{\eta^2}{2z^2} - \frac{xsy}{1+\sqrt{1-z^2}}\Big\} - \exp\Big\{-\frac{\eta^2}{2z^2} - \frac{xsy}{2}\Big\}(1+\lambda z^2)dz. \quad (3.157)
\end{aligned}$$

For computing the first integral in RHS of (3.157), we take into account the two following identities.

$$\int z^n \exp\Big\{-\frac{\eta^2}{2z^2}\Big\}dz = \frac{z^{n+1}}{n+1}\exp\Big\{-\frac{\eta^2}{2z^2}\Big\} - \frac{\eta^2}{n+1}\int z^{n-2}\exp\Big\{-\frac{\eta^2}{2z^2}\Big\}dz, \quad (3.158)$$

$$\begin{aligned}
\int \exp\Big\{-\frac{\eta^2}{2z^2}\Big\}dz =& z\exp\Big\{-\frac{\eta^2}{2z^2}\Big\} - b\int_{\eta/z}^\infty \exp\Big\{-\frac{u^2}{2}\Big\}du + \mathrm{C} \\
=& z\exp\Big\{-\frac{\eta^2}{2z^2}\Big\}dz - \sqrt{2\pi}\eta\Phi\Big(-\frac{\eta}{z}\Big) + \mathrm{C}, \quad (3.159)
\end{aligned}$$

where C is a real constant. Therefore, for $n = 2$, it follows from (3.158) and (3.159) that

$$\begin{aligned}
I_1 =& \int_0^\tau \exp\Big\{-\frac{xsy}{2} - \frac{\eta^2}{2z^2}\Big\}[1+\lambda z^2]dz \\
=& \exp\Big\{-\frac{xsy}{2}\Big\}\Big[\tau\Big(1-\lambda\frac{\eta^2-\tau^2}{3}\Big)\exp\Big\{-\frac{\eta^2}{2\tau^2}\Big\} - \sqrt{2\pi}\eta\Big[1-\lambda\frac{\eta^2}{3}\Big]\Phi\Big(-\frac{\eta}{\tau}\Big)\Big]. \quad (3.160)
\end{aligned}$$

The second integral in RHS of (3.157) is

$$I_2 = \int_0^\tau \frac{1}{\sqrt{1-z^2}} \exp\Big\{-\frac{\eta^2}{2z^2} - \frac{xsy}{1+\sqrt{1-z^2}}\Big\} - \exp\Big\{-\frac{\eta^2}{2z^2} - \frac{xsy}{2}\Big\}(1+\lambda z^2)dz. \quad (3.161)$$

Hence,

$$L(x,y|\rho) = L(x,y|s) - \frac{s}{2\pi}[I_1 + I_2]. \quad (3.162)$$

Splitting integral in RHS of (3.155) into I_1 and I_2 yields more accurate result since by computing I_1 analytically though (3.160) the rate of error (when $|\rho|$ is large and η is small) is diminished. The integral of reminder term, that is I_2 in (3.161), would be unaffected by error source and computed using a k-point Gauss-Legendre rule. For computing $L(x,y|s)$ in RHS of (3.162), we take into account the fact that

$$\begin{aligned}
L(x,y|\rho) =& \int_x^\infty \int_y^\infty \phi(u)\phi\Big[\frac{v-\rho u}{\sqrt{1-\rho^2}}\Big]dvdu \\
=& \int_x^\infty \phi(u)\bar{\Phi}\Big[\frac{y-\rho u}{\sqrt{1-\rho^2}}\Big]du, \quad (3.163)
\end{aligned}$$

where $\bar{\Phi}(x) = 1 - \Phi(x)$. Taking limit from both sides of (3.163) when $\rho \to 1$, it turns out that

$$L(x,y|1) = \begin{cases}
\displaystyle\int_x^\infty \phi(u)\bar{\Phi}\Big[\frac{y-u}{0^+}\Big]du = \bar{\Phi}(x), & \text{if} \quad y < x, \\
\displaystyle\int_x^y \phi(u)\bar{\Phi}\Big[\frac{y-u}{0^+}\Big]du + \int_y^\infty \phi(u)\bar{\Phi}\Big[\frac{y-u}{0^+}\Big]du & \text{if} \quad y > x. \\
= 0 + \displaystyle\int_y^\infty \phi(u)\bar{\Phi}\Big[\frac{y-u}{0^+}\Big]du & \\
= \bar{\Phi}(y), &
\end{cases} \quad (3.164)$$

From (3.164) we conclude that

$$L(x, y|1) = \bar{\Phi}(\max\{x, y\}) = \Phi(-\max\{x, y\}). \tag{3.165}$$

Similar arguments as above in (3.164) show that

$$\begin{aligned} L(x, y|-1) &= \begin{cases} 0, & \text{if} \quad x + y > 0, \\ \Phi(-y) - \Phi(x), & \text{if} \quad x + y \leq 0. \end{cases} \\ &= \max\Big\{0, \Phi(-y) - \Phi(x)\Big\}. \end{aligned} \tag{3.166}$$

Hence,

$$L(x, y|s) = \begin{cases} \Phi(-\max\{x, y\}), & \text{if} \quad s = 1, \\ \max\Big\{0, \Phi(-y) - \Phi(x)\Big\}, & \text{if} \quad s = -1. \end{cases} \tag{3.167}$$

Implementing the two-step procedure of Drezner and Wesolowsky [1990] described above based on 5-point Gauss-Legendre rule yields a maximum error of 2×10^{-7}. The original work of Drezner and Wesolowsky [1990] incorporates a `FORTRAN` code of 36 lines for computing $L(x, y|\rho)$. In what follows, we give the corresponding `R` code.

```
R> BVN_Drez <- function(a, Mu, Sigma)
+{
+ rho <- Sigma[1, 2]/sqrt( Sigma[1, 1]*Sigma[2, 2] )
+ lb <- c(a - Mu)/sqrt( diag(Sigma) )
+ x <- lb[1]
+ y <- lb[2]
+ s <- sign(rho)
+ w    <- c(0.2369268851, 0.4786286705, 0.5688888889,  0.4786286705,  0.2369268851)
+ node <- c(0.9061798459, 0.5384693101,            0, -0.5384693101, -0.9061798459)
+ x0 <- rho*(1 + node)/(2*s)
+ p1 <- sqrt(1 - x0^2)
+  if( abs(rho) < 0.7)
+  {
+    out <- pnorm(-x)*pnorm(-y) +
+           rho/(4*pi)*sum( w/p1*exp(-(x^2 + y^2 - 2*x*s*y*x0)/( 2*p1^2 ) ) )
+  }else{
+    L_s <- ifelse( rho < 0, max(0, pnorm(-y)-pnorm(x)), pnorm( -max(x, y) ) )
+    y <- s*y
+    tau <- sqrt(1 - rho^2)
+    x0 <- tau*(node + 1)/2
+    p2 <- sqrt(1 - x0^2)
+    lambda <- (4 - x*y)/8
+    eta <- abs(x - y)
+    p3 <- exp(-x*y/2)
+    p4 <- eta/tau
+    p5 <- exp(- p4^2/2 )
+    I1 <- p3*( tau*(1 - lambda*(eta^2 - tau^2)/3)*p5 -
+          sqrt(2*pi)*eta*(1 - lambda*eta^2/3)*pnorm(-p4) )
+    p6 <- exp(-eta^2/(2*x0^2))
+    p7 <- exp(- x*y/( 1 + p2 ) )/p2 - p3*(1 + lambda*x0^2)
+    I2 <- tau/2*sum( w*p6*p7 )
+    out <- L_s - s/(2*pi)*(I1 + I2)
+  }
+return( abs(out) )
+}
```

Genz [2004] pointed out that the hybrid (two-step) method of Drezner and Wesolowsky [1990] described earlier is yet a single precision algorithm[3] so as can yield a maximum absolute error of 2.5×10^{-7} if the threshold for ρ changes to 0.80 rather than 0.70, and further claimed if one uses a three-term Taylor expansion in (3.156) as

$$\frac{1}{\sqrt{1-z^2}} \exp\Big\{-\frac{xsy}{1+\sqrt{1-z^2}}\Big\} = \exp\Big\{-\frac{xsy}{2}\Big\}\Big[1+\lambda z^2 + (\frac{3}{8} - \frac{1}{8}xsy + \frac{1}{128}xy^2)z^4\Big] + \mathcal{O}(z^6), \tag{3.168}$$

then, based on a 20-point Gauss-Legendre rule, method of Drezner and Wesolowsky [1990] would be a double precision algorithm[4] with a maximum error of 5×10^{-16} if the threshold for ρ changes to 0.925 rather than 0.70.

3.8.2 Gaussian mixture distribution

Herein, we introduce two commonly used family of distributions known in the literature as *Gaussian scale (variance) mixture* (GSM) distribution and *Gaussian variance-mean mixture* (GSMM) distribution. These two families incorporate a wide range of well-known distributions that are expressed in terms of the Gaussian distribution.

Definition 3.8.1. *(***univariate GSM***) Let random variable Z follows a standard Gaussian distribution. For a given positive real valued function $\eta(\cdot)$ and positive mixing random variable G with PDF $h(g|\boldsymbol{\theta})$ in which $\boldsymbol{\theta}$ is the family parameter, the random variable X with CDF $F(\boldsymbol{x}|\Psi)$ is said to follow a Gaussian scale mixture model if*

$$X = \mu + \sigma \frac{Z}{\sqrt{\eta(G)}}, \tag{3.169}$$

where $\Psi = (\mu, \sigma, \boldsymbol{\theta})$ in which $\mu \in \mathbb{R}$ and $\sigma \in \mathbb{R}^+$ are the location and scale parameters of $F(x|\Psi)$, respectively.

Likewise, the multivariate version of the Gaussian scale mixture model (representation) is given as follows.

Definition 3.8.2. *(***multivariate GSM***) Let random vector $\boldsymbol{Z}$ follows a p-dimensional standard Gaussian distribution. For a given positive real valued function $\eta(\cdot)$ and a positive mixing random variable G with PDF $h(g|\boldsymbol{\theta})$ in which $\boldsymbol{\theta}$ is the family parameter, the p-dimensional random vector $\boldsymbol{X}$ with CDF $F(\boldsymbol{x}|\Psi)$ is said to follow a Gaussian scale mixture model if*

$$\boldsymbol{X} = \boldsymbol{\mu} + A \frac{\boldsymbol{Z}}{\sqrt{\eta(G)}}, \tag{3.170}$$

where $\Psi = (\boldsymbol{\mu}, \Sigma, \boldsymbol{\theta})$ in which $\boldsymbol{\mu} \in \mathbb{R}^p$ and $p \times p$ positive definite matrix Σ are the location and scale parameters of $F(\boldsymbol{x}|\Psi)$, respectively. The lower triangular matrix A is known as the Cholesky decomposition of scale matrix, that is $\Sigma = AA^\top = (\sigma_{i,j})$. Obviously identity (3.170) admits the following hierarchy

$$\begin{aligned} \boldsymbol{X}\Big|G = g &\sim \mathcal{N}_p\Big(\boldsymbol{\mu}, \frac{\Sigma}{\eta(g)}\Big), \\ G &\sim h(g|\boldsymbol{\theta}), \end{aligned} \tag{3.171}$$

[3] In single precision algorithm, of a total of 32 bits, machine assigns one bit for sign, 8 bits for exponent, and 23 bits for fraction part of a real number. Therefore, a single precision format approximately provides a $\log_{10}(2^{24}) = 7.22471\cdots \approx 7$ digits of precision.

[4] In double precision algorithm, of a total of 64 bits, machine assigns one bit for sign, 11 bits for exponent, and 52 bits for fraction part of a real number. Therefore, a double precision format approximately provides a $\log_{10}(2^{53}) = 15.9545\cdots \approx 16$ digits of precision.

or equivalently

$$F(\boldsymbol{x}|\Psi) = \int_0^\infty \Phi_p\Big[\boldsymbol{x} - \boldsymbol{\mu}\Big|\mathbf{0}, \frac{\Sigma}{\eta(g)}\Big] h(g|\boldsymbol{\theta})dg, \tag{3.172}$$

where $\Phi_p(\cdot|\boldsymbol{\mu}, \Sigma)$ *is the CDF of a p-dimensional random vector* $\boldsymbol{X} \sim \mathcal{N}_p(\boldsymbol{\mu}, \Sigma)$.

Corollary 3.8.1. *The ith, for* $i = 1, \cdots, p$*, marginal of each GSM distribution in Definition 3.8.2 with* $\Sigma = (\sigma_{i,j})$ *follows a univariate GSM distribution as given in Definition 3.8.1 for which* $\sigma = \sqrt{\sigma_{i,i}}$.

Definition 3.8.3. *(***univariate GSSM***) Let random variable Z follows a standard Gaussian distribution. For a given positive real valued function* $\eta(\cdot)$ *and positive mixing random variable G with PDF* $h(g|\boldsymbol{\theta})$ *in which* $\boldsymbol{\theta}$ *is the family parameter, the random variable X with CDF* $F(\boldsymbol{x}|\Psi)$ *is said to follow a Gaussian scale mixture model if*

$$X = \mu + \frac{\alpha}{\sqrt{\eta(G)}} + \sigma\frac{Z}{\sqrt{\eta(G)}}, \tag{3.173}$$

where $\Psi = (\mu, \alpha, \sigma, \boldsymbol{\theta})$ *in which* $\mu \in \mathbb{R}$*,* $\alpha \in \mathbb{R}$*, and* $\sigma \in \mathbb{R}^+$ *are, accordingly, the location, skewness, and scale parameters of* $F(x|\Psi)$*, respectively.*

Likewise, the multivariate version of the GSMM distribution is given as follows.

Definition 3.8.4. *(***multivariate GSSM***) Let random vector* $\boldsymbol{Z}$ *follows a p-dimensional standard Gaussian distribution. For a given positive real valued function* $\eta(\cdot)$ *and a positive mixing random variable G with PDF* $h(g|\boldsymbol{\theta})$ *in which* $\boldsymbol{\theta}$ *is the family parameter, the p-dimensional random vector* $\boldsymbol{X}$ *with CDF* $F(\boldsymbol{x}|\Psi)$ *is said to follow a Gaussian scale mixture model if*

$$\boldsymbol{X} = \boldsymbol{\mu} + \frac{\boldsymbol{\alpha}}{\eta(G)} + A\frac{\boldsymbol{Z}}{\sqrt{\eta(G)}}, \tag{3.174}$$

where $\Psi = (\boldsymbol{\mu}, \boldsymbol{\alpha}, \Sigma, \boldsymbol{\theta})$ *in which* $\boldsymbol{\mu} \in \mathbb{R}^p$*,* $\boldsymbol{\alpha} \in \mathbb{R}^p$*, and* $p \times p$ *positive definite matrix* Σ *are, respectively, the location, skewness, and scale parameters of* $F(\boldsymbol{x}|\Psi)$*, respectively. The lower triangular matrix A is known as the Cholesky decomposition of scale matrix, that is* $\Sigma = AA^\top = (\sigma_{i,j})$*. Obviously identity (3.170) admits the following hierarchy*

$$\begin{aligned} \boldsymbol{X}\Big|G = g &\sim \mathcal{N}_p\Big(\boldsymbol{\mu} + \frac{\boldsymbol{\alpha}}{\eta(g)}, \frac{\Sigma}{\eta(g)}\Big), \\ G &\sim h(g|\boldsymbol{\theta}), \end{aligned} \tag{3.175}$$

or equivalently the PDF of $\boldsymbol{X}$ *becomes*

$$f(\boldsymbol{x}|\Psi) = \int_0^\infty \Phi_p\Big[\boldsymbol{x} - \boldsymbol{\mu} - \frac{\boldsymbol{\alpha}}{\eta(g)}\Big|\mathbf{0}, \frac{\Sigma}{\eta(g)}\Big] h(g|\boldsymbol{\theta})dg, \tag{3.176}$$

where $\Phi_p(\cdot|\boldsymbol{\mu}, \Sigma)$ *is the CDF of a p-dimensional Gaussian distribution with mean* $\boldsymbol{\mu}$ *and covariance matrix* Σ.

Example 3-17

Set $\eta(g) = 1/g$ in Definition 3.8.4 in which $G \sim \mathcal{GIG}(\psi, \chi, \lambda)$ with PDF given in (2.48), Then the established GSMM distribution becomes generalized hyperbolic (GH) distribution with PDF given by [McNeil et al., 2015]:

$$\begin{aligned} f(\boldsymbol{x}|\Psi) =& \frac{\exp\{(\boldsymbol{x} - \boldsymbol{\mu})^\top \Sigma^{-1}\boldsymbol{\alpha}\}}{(2\pi)^{p/2}|\Sigma|^{1/2}} \Big[\frac{\chi + \delta(\boldsymbol{x} - \boldsymbol{\mu}, \Sigma)}{\psi + \delta(\boldsymbol{\alpha}, \Sigma)}\Big]^{\frac{\lambda - p/2}{2}} \Big(\frac{\psi}{\chi}\Big)^{\frac{\lambda}{2}} \\ &\times \frac{\mathcal{K}_{\lambda - p/2}\big(\sqrt{[\psi + \delta(\boldsymbol{\alpha}, \Sigma)][\chi + \delta(\boldsymbol{x} - \boldsymbol{\mu}, \Sigma)]}\big)}{\mathcal{K}_\lambda(\sqrt{\psi\chi})}, \end{aligned} \tag{3.177}$$

where $\Psi = (\boldsymbol{\mu}, \boldsymbol{\alpha}, \Sigma, \boldsymbol{\theta})$ in which $\boldsymbol{\theta} = (\psi, \chi, \lambda)^\top$ and $\delta(\cdot, \cdot)$ is defined as (3.29). The GH distribution with PDF in (3.177) has been introduced briefly by Barndorff-Nielsen [1977a] and then its properties discussed in several works Barndorff-Nielsen [1977b, 1978]. As an application, this distribution was used for modelling the distribution of the Australian personal income [Barndorff-Nielsen, 1978]. For useful accounts of GH distribution, the reader is referred to Weibel et al. [2020] and referenced therein. One issue that may arise with the GH distribution is its non-identifiability [Murray et al., 2017]. For any $m > 0$, let $\Psi = (\boldsymbol{\mu}, \boldsymbol{\alpha}, \Sigma, \boldsymbol{\theta})$ and $\Psi^\star = (\boldsymbol{\mu}, m\boldsymbol{\alpha}, m\Sigma, \boldsymbol{\theta})$ in which $\boldsymbol{\theta} = (m\psi, \chi/m, \lambda)^\top$ denote two different choices of the family parameter vector. It is easy to check from (3.177) that $f(\boldsymbol{x}|\Psi) = f(\boldsymbol{x}|\Psi^\star)$. Hence, the GH distribution with PDF in (3.177) is not identifiable. In order to ensure the identifiablility, we may set $\psi = \chi = a$, for $a > 0$, and $\boldsymbol{\alpha} = \mathbf{0}$. The result is the symmetric hyperbolic (SH) distribution with PDF

$$f(\boldsymbol{x}|\Psi) = \Big[\frac{a + \delta(\boldsymbol{x} - \boldsymbol{\mu}, \Sigma)}{a}\Big]^{\frac{\lambda - p/2}{2}} \frac{\mathcal{K}_{\lambda - p/2}\big(\sqrt{a[a + \delta(\boldsymbol{x} - \boldsymbol{\mu}, \Sigma)]}\big)}{(2\pi)^{p/2}|\Sigma|^{1/2}\mathcal{K}_\lambda(a)}, \tag{3.178}$$

where $\Psi = (\boldsymbol{\mu}, \Sigma, \boldsymbol{\theta})$ in which $\boldsymbol{\theta} = (a, \lambda)^\top$. Evidently, each SH distribution is a GSM distribution that follows identity (3.170) in which $\eta(g) = 1/g$, $A = \Sigma^{1/2}$, and $G \sim \mathcal{GIG}(a, a, \lambda)$. The SH distribution is a heavy tailed distribution and is commonly used in finance for modelling the distribution of return's rate Eberlein and Keller [1995] and also for computing the value-at-risk parameter for a certain portfolio Bauer [2000]. ■

Example 3-18

Based on Definition 3.8.1, for $\Psi = (0, \sigma, \boldsymbol{\theta})^\top$, the CDF of GSM model can be represented as

$$F(x|\Psi) = \begin{cases} \dfrac{1}{2} + \dfrac{\text{sign}(x)}{\sqrt{\pi}} \displaystyle\int_0^\infty H\Big(\frac{x^2}{2\sigma^2 z^2}\Big|\boldsymbol{\theta}\Big) \exp\{-z^2\} dz, & \text{if} \quad \eta(g) = g, \\ \dfrac{1}{2} + \dfrac{\text{sign}(x)}{\sqrt{\pi}} \displaystyle\int_0^\infty \bar{H}\Big(\frac{2\sigma^2 z^2}{x^2}\Big|\boldsymbol{\theta}\Big) \exp\{-z^2\} dz, & \text{if} \quad \eta(g) = \frac{1}{g}, \end{cases} \tag{3.179}$$

where $H(u|\boldsymbol{\theta})$ is the CDF of G in Definition 3.8.1 and $\bar{H}(u|\boldsymbol{\theta}) = 1 - H(u|\boldsymbol{\theta})$. Hence, $F(x|\Psi)$ in (3.179) can be approximated using the Gauss-Hermite rule of third (or first) kind. As the second method, the CDF $F(x|\Psi)$ can be obtained directly by integration the corresponding PDF. For example, suppose in Definition 3.8.1 we have $G \sim \mathcal{G}(\nu/2, \nu/2)$ and $\eta(g) = 1/g$. Then, clearly $F(\cdot|\Psi)$ in (3.179) becomes the CDF of Student's t distribution with ν degrees of freedom.

$$F(x|\Psi) = \frac{1}{2} + \frac{\text{sign}(x)}{2\sqrt{\pi}} \frac{\Gamma\big(\frac{\nu+1}{2}\big)}{\Gamma\big(\frac{\nu}{2}\big)\sqrt{\nu\pi}\sigma} \int_0^{|x|} \Big(1 + \frac{z^2}{\nu\sigma^2}\Big)^{-\frac{\nu+1}{2}} dz, \tag{3.180}$$

The `R` function `pt(x, df, ...)` computes the CDF of Student's t distribution with degrees of freedom `df`. Table 3.12 shows the results for approximating $F(\cdot|\Psi)$ in which $\Psi = (0, 1, \nu)^\top$ using a k-point Gauss-Hermite rule of third kind and a k-point Gaussian-Legendre rule applied, accordingly, on RHS of (3.179), when $\eta(g) = 1/g$, and (3.180). Our investigation reveals that using a hybrid process, described as follow, the maximum ARE would be less or equal than 8×10^{-10}. Based on the hybrid process, a 10-point Gauss-Hermite rule is applied when $\nu \leq 8$ and $x \geq 3.5\sigma$ and a 10-point Gaussian-Legendre rule otherwise. ■

3.8.3 Computing the CDF of bivariate Student's t distribution

Herein, we describe the method of Genz [2004] for computing the CDF of a bivariate Student's t distribution. The CDF of a p-dimensional Student's t distribution with ν degrees of freedom

Table 3.12: Computed ARF for approximating the CDF of Student's t using k-point Gaussian-Legendre and Gauss-Hermite rules.

		ARE	
k	ν	Gauss-Hermite	Gauss-Legendre
2	2	0.02424	0.00013
2	10	0.03742	0.00012
2	40	0.02748	0.00014
5	2	0.00018	1.81e-08
5	10	0.01700	5.73e-10
5	40	0.02720	6.11e-11
10	2	6.31e-08	2.81e-16
10	10	9.22e-04	1.33e-16
10	40	0.01706	3.97e-16

is

$$T_\nu(\boldsymbol{b}|\boldsymbol{\mu}, R) = \frac{(\nu\pi)^{-\frac{p}{2}}}{\sqrt{|R|}} \frac{\Gamma(\frac{\nu+p}{2})}{\Gamma(\frac{\nu}{2})} \int_{-\infty}^{b_1} \cdots \int_{-\infty}^{b_p} \Big[1 + \frac{(\boldsymbol{x}-\boldsymbol{\mu})^\top R^{-1}(\boldsymbol{x}-\boldsymbol{\mu})}{\nu}\Big]^{-\frac{\nu+p}{2}} dx_p \cdots dx_1, \quad (3.181)$$

where $R\{\rho_{i,j}\}$ denotes the correlation matrix. In bivariate case, the CDF of Student's t is given by

$$T_\nu(\boldsymbol{b}|\boldsymbol{0}, \rho) = \frac{1}{2\pi\sqrt{1-\rho^2}} \int_{-\infty}^{b_1} \int_{-\infty}^{b_2} \Big[1 + \frac{x_1^2 + x_2^2 - 2\rho x_1 x_2}{\nu(1-\rho^2)}\Big]^{-\frac{\nu+2}{2}} dx_2 dx_1, \quad (3.182)$$

where $\boldsymbol{b} = (b_1, b_2)^\top$ denotes a vector of real constants. It should be noted that each Student's t is a Gaussian scale mixture model. Hence, setting $\boldsymbol{\mu} = \boldsymbol{0}$, $\Sigma = [(1,\rho)^\top, (\rho,1)^\top]$, $\eta(g) = g$, and $G \sim \mathcal{G}(\nu/2, \nu/2)$, then it follows from definition 3.8.2 that $\boldsymbol{X}$ has a standard bivariate Student's t distribution with CDF $F(\boldsymbol{b}|\Psi = (\rho,\nu)^\top) = T_\nu(\boldsymbol{b}|\rho)$. So using relation (3.176), the bivariate Student's t distribution given in (3.182) can be represented as

$$T_\nu(\boldsymbol{b}|\boldsymbol{0}, \rho) = \int_0^\infty \Phi_2(\sqrt{g}\boldsymbol{b}|\boldsymbol{0}, \rho)\mathcal{G}\Big(g\Big|\frac{\nu}{2}, \frac{\nu}{2}\Big)dg.$$

For computational purpose, using transformation $g\nu = u^2$, we obtain another representation of (3.182) given by [Cornish, 1954] as

$$T_\nu(\boldsymbol{b}|\boldsymbol{0}, \rho) = \frac{2^{1-\nu/2}}{\Gamma(\frac{\nu}{2})} \int_0^\infty \Phi_2\Big(\frac{u\boldsymbol{b}}{\sqrt{\nu}}\Big|\boldsymbol{0}, \rho\Big) u^{\nu-1} \exp\Big\{-\frac{u^2}{2}\Big\} du, \quad (3.183)$$

By applying the Plackett's formula (3.148) on the term $\Phi_2(\cdot|\boldsymbol{0}, \rho)$ in RHS of (3.183), we obtain

$$\frac{\partial}{\partial\rho} T_\nu(\boldsymbol{b}|\boldsymbol{0}, \rho) = \frac{2^{1-\nu/2}}{\Gamma(\frac{\nu}{2})} \int_0^\infty \frac{u^{\nu-1}}{2\pi\sqrt{1-\rho^2}} \exp\Big\{-\frac{u^2}{2}\Big[1 + \frac{\delta(\boldsymbol{b}, \rho)}{\nu}\Big]\Big\} du, \quad (3.184)$$

where $\delta(\boldsymbol{b}, \rho)$ is given by

$$\begin{aligned} \delta(\boldsymbol{b}, \Sigma) =& \boldsymbol{b}^\top \Sigma^{-1} \boldsymbol{b} \\ =& \frac{b_1^2 + b_2^2 - 2\rho b_1 b_2}{(1-\rho^2)} \\ =& \delta(\boldsymbol{b}, \rho). \end{aligned} \quad (3.185)$$

Applying the change of variable $z = u\sqrt{1+\delta(\boldsymbol{b},\rho)/\nu}$ on RHS of (3.184), it turns out that

$$\begin{aligned}\frac{\partial}{\partial\rho}T_\nu(\boldsymbol{b}|\boldsymbol{0},\rho) =& \frac{2^{1-\nu/2}}{\Gamma(\frac{\nu}{2})2\pi\sqrt{1-\rho^2}}\Big[1+\frac{\delta(\boldsymbol{b},\rho)}{\nu}\Big]^{-\frac{\nu}{2}}\int_0^\infty z^{\nu-1}\exp\Big\{-\frac{z^2}{2}\Big\}dz,\\ =& \frac{1}{2\pi\sqrt{1-\rho^2}}\Big[1+\frac{\delta(\boldsymbol{b},\rho)}{\nu}\Big]^{-\frac{\nu}{2}}.\end{aligned} \tag{3.186}$$

Integrating both sides of (3.186) over the range (s,ρ) shows

$$T_\nu(\boldsymbol{b}|\boldsymbol{0},\rho) = T_\nu(\boldsymbol{b}|\boldsymbol{0},s) + \frac{1}{2\pi}\int_s^\rho \frac{\Big[1+\frac{\delta(\boldsymbol{b},r)}{\nu}\Big]^{-\frac{\nu}{2}}}{\sqrt{1-r^2}}dr. \tag{3.187}$$

where $s = \text{sign}(\rho)$ and

$$T_\nu(\boldsymbol{b}|\boldsymbol{0},s) = \begin{cases} T_\nu(\min\{b_1,b_2\}), & \text{if} \quad s=+1,\\ \max\Big\{0, T_\nu(b_1)-T_\nu(-b_2)\Big\}, & \text{if} \quad s=-1.\end{cases}$$

Applying a change of variable $r = \sin(\theta)$ on (3.187), it turns out that

$$T_\nu(\boldsymbol{b}|\boldsymbol{0},\rho) = T_\nu(\boldsymbol{b}|\boldsymbol{0},s) + \frac{1}{2\pi}\int_{s\frac{\pi}{2}}^{\sin^{-1}(\rho)}\Big[1+\frac{b_1^2+b_2^2-2\sin(\theta)b_1b_2}{\nu\cos(\theta)^2}\Big]^{-\frac{\nu}{2}}d\theta. \tag{3.188}$$

It is worthwhile to mention that if $\rho = 0$, then the marginals of multivariate Student's t are not independent and hence, $T_\nu(\boldsymbol{b}|\boldsymbol{0},0)$ cannot be expressed as the product of $T_\nu(b_1|0,1)$ and $T_\nu(b_2|0,1)$. Furthermore, for any positive small constant such as ϵ, if $|\rho| \le \epsilon$, then based on the continuity property of $T_\nu(\boldsymbol{b}|\boldsymbol{0},\rho)$, formula (3.188) can be applied for computing $T_\nu(\boldsymbol{b}|\boldsymbol{0},\rho)$ if ρ and s are, accordingly, replaced with ϵ and $-\epsilon$ where ϵ is sufficiently small, $\epsilon = 10e-8$ say. Genz [2004] uses a k-point Gauss-Legendre rule for approximating bivariate Student's t probabilities using (3.188). The maximum recorded of absolute errors are $10^{-3}, 10^{-4}, 10^{-6}$, and 6×10^{-11} when $k = 6, 12, 24$, and 48, respectively. In what follows, we give the pertinent `R` function `BVT_quad` for computing $T_\nu(\boldsymbol{b}|\boldsymbol{0},R)$.

Example 3-19

```
R> BVT_Genz <- function(b, rho, nu, k)
+{
+ T1 <- max(0, pt( q = b[1], df = nu ) - pt( q = -b[2], df = nu ))
+ T2 <- pt( q = min(b), df = nu )
+ rho <- ifelse( abs(rho) < 10e-8, 10e-8, rho)
+ s <- sign(rho)
+ p1 <- ifelse(s < 0, T1, T2)
+ out <- quad_rule(k, type = "LE", alpha = 0, beta = 0)
+ node <- out$node; w <- out$weight;
+ x <- (node + 1)*( asin(rho) - s*pi/2 )/2 + s*pi/2
+ delta <- ( b[1]^2 + b[2]^2 - 2*sin(x)*b[1]*b[2] )/cos(x)^2
+ p2 <- sum( w*( 1 + delta/nu )^(-nu/2) )
+ CDF <- p1 + ( asin(rho) - s*pi/2 )/(4*pi)*p2
+ return(CDF)
+}
```

It is worth to note that the built-in function `pt(.,.)` in the fourth line of function `BVT_Genz` can be replaced with that of has been just suggested in Example 3-18. ■

3.8.4 Computing the CDF of bivariate Gaussian scale mixture distribution

Herein, Theorem 3.8.1 gives a general framework for computing the CDF of bivariate Gaussian scale mixture distribution.

Theorem 3.8.1. *Let $\mathcal{M}_G(t|\boldsymbol{\theta}) = E\big(\exp\{tG\}\big)$ denote the MGF of the mixing variable G in Definition 3.8.2. Set $\eta(g) = g$ and $\Psi = (\mathbf{0}, R, \boldsymbol{\theta})$ in which $R = [(1,\rho)^\top, (\rho,1)^\top]$. Then*

$$F(\boldsymbol{x}|\Psi) = \begin{cases} F\big(\min\{x_1,x_2\}\big) + \dfrac{1}{2\pi}\displaystyle\int_{\frac{\pi}{2}}^{\sin^{-1}(\rho)} \mathcal{M}_G\Big[-\frac{\delta(\boldsymbol{x},\sin(u))}{2}\Big]du, & \text{if } \rho > 0, \\ \max\Big\{0, F(x_2) - F(-x_1)\Big\} + \dfrac{1}{2\pi}\displaystyle\int_{-\frac{\pi}{2}}^{\sin^{-1}(\rho)} \mathcal{M}_G\Big[-\frac{\delta(\boldsymbol{x},\sin(u))}{2}\Big]du, & \text{if } \rho < 0, \end{cases} \tag{3.189}$$

where $|\rho| < 1$ and $\delta(\boldsymbol{x}, u)$, for $|u| < 1$, is defined as (3.185).

Proof: Suppose in Definition 3.170 that $\boldsymbol{\mu} = \mathbf{0}$, $\eta(g) = g$, $p = 2$, and $A = [(1,0)^\top, (\rho, \sqrt{1-\rho^2})^\top]$ for which the scale (correlation) matrix is $\Sigma = R = [(1,\rho)^\top, (\rho,1)^\top]$. Then $\Psi = (\mathbf{0}, R, \boldsymbol{\theta})$ and the CDF of $\boldsymbol{X}$ becomes

$$F(\boldsymbol{x}|\Psi) = \int_0^\infty \Phi_2(\sqrt{g}\boldsymbol{x}|\mathbf{0},\rho)h(g|\boldsymbol{\theta})dg, \tag{3.190}$$

where $\boldsymbol{x} = (x_1, x_2)^\top$. Taking into account the fact

$$\Phi_2(\sqrt{g}\boldsymbol{x}|\mathbf{0},\rho) = L(-\sqrt{g}x_1, -\sqrt{g}x_2|\rho), \tag{3.191}$$

where $L(x,y|\rho)$ is given in (3.146) and then applying the Plackett's formula (3.148) on the both sides of (3.190), we have

$$\begin{aligned} \frac{\partial}{\partial\rho}F\big[\boldsymbol{x}\big|\Psi = (\mathbf{0},\rho,\boldsymbol{\theta})\big] &= \int_0^\infty \frac{\partial}{\partial\rho}\Phi_2(\sqrt{g}\boldsymbol{x}|\mathbf{0},\rho)h(g|\boldsymbol{\theta})dg \\ &= \int_0^\infty \frac{1}{2\pi\sqrt{1-\rho^2}}\exp\Big\{-g\frac{x_1^2+x_2^2-2x_1x_2\rho}{2(1-\rho^2)}\Big\}h(g|\boldsymbol{\theta})dg \\ &= \frac{1}{2\pi\sqrt{1-\rho^2}}\int_0^\infty \exp\Big\{-g\frac{\delta(\boldsymbol{x},\rho)}{2}\Big\}h(g|\boldsymbol{\theta})dg \\ &= \frac{1}{2\pi\sqrt{1-\rho^2}}\mathcal{M}_G\Big(-\frac{1}{2}\delta(\boldsymbol{x},\rho)\Big|\boldsymbol{\theta}\Big), \end{aligned} \tag{3.192}$$

where $\delta(\boldsymbol{b},\rho)$ is given in (3.185) and $\mathcal{M}_G(\cdot)$ denotes the moment generating function (MGF) of random variable G. Integrating both sides of (3.192) from $s = \text{sign}(\rho)$ to ρ yields

$$F\big[\boldsymbol{x}\big|\Psi = (\mathbf{0},\rho,\boldsymbol{\theta})\big] = F\big[\boldsymbol{x}\big|\Psi = (\mathbf{0},s,\boldsymbol{\theta})\big] + \frac{1}{2\pi}\int_s^\rho \frac{1}{\sqrt{1-r^2}}\mathcal{M}_G\Big(-\frac{1}{2}\delta(\boldsymbol{x},r)\Big|\boldsymbol{\theta}\Big)dr.$$

Using a change of variable of the form $r = \sin(u)$, we obtain

$$F\big[\boldsymbol{x}\big|\Psi = (\mathbf{0},\rho,\boldsymbol{\theta})\big] = F\big[\boldsymbol{x}\big|\Psi = (\mathbf{0},s,\boldsymbol{\theta})\big] + \frac{1}{2\pi}\int_{s\frac{\pi}{2}}^{\sin^{-1}(\rho)} \mathcal{M}_G\Big(-\frac{1}{2}\delta(\boldsymbol{x},\sin(u))\Big|\boldsymbol{\theta}\Big)du. \tag{3.193}$$

For computing $F(\boldsymbol{x}|\Psi)$ in RHS of (3.193), it follows from (3.190) that

$$F\big[\boldsymbol{x}\big|\Psi = (\mathbf{0},s,\boldsymbol{\theta})\big] = \int_0^\infty \Phi_2(\sqrt{g}\boldsymbol{x}|\mathbf{0},s)h(g|\boldsymbol{\theta})dg. \tag{3.194}$$

From (3.191) it turns out that

$$F\big[\boldsymbol{x}|\Psi=(\mathbf{0},s,\boldsymbol{\theta})\big]=\int_0^{\infty}L(-\sqrt{g}x_1,-\sqrt{g}x_2|s)h(g|\boldsymbol{\theta})dg, \tag{3.195}$$

and based on (3.167), we can write

$$\begin{aligned}
F\big[\boldsymbol{x}|\Psi=(\mathbf{0},s,\boldsymbol{\theta})\big]&=\begin{cases}\int_0^{\infty}\Phi(-\max\{-\sqrt{g}x_1,-\sqrt{g}x_2\})h(g|\boldsymbol{\theta})dg, & \text{if } s=+1,\\ \int_0^{\infty}\max\Big\{0,\Phi(\sqrt{g}x_2)-\Phi(-\sqrt{g}x_1)\Big\}h(g|\boldsymbol{\theta})dg, & \text{if } s=-1.\end{cases}\\
&=\begin{cases}\int_0^{\infty}\Phi(\sqrt{g}\min\{x_1,x_2\})h(g|\boldsymbol{\theta})dg, & \text{if } s=+1,\\ \max\Big\{0,\int_0^{\infty}\Phi(\sqrt{g}x_2)h(g|\boldsymbol{\theta})dg-\int_0^{\infty}\Phi(-\sqrt{g}x_1)h(g|\boldsymbol{\theta})dg\Big\}, & \text{if } s=-1.\end{cases}\\
&=\begin{cases}F(\min\{x_1,x_2\}), & \text{if } s=+1,\\ \max\Big\{0,F(x_2)-F(-x_1)\Big\}, & \text{if } s=-1.\end{cases}
\end{aligned} \tag{3.196}$$

Proof is complete by substituting RHS of (3.196) into (3.193).

We note that when $\rho=0$, then Theorem 3.8.1 can be used if $\Psi=(\mathbf{0},R=\{0\},\boldsymbol{\theta})$ is replaced with $\Psi'=(\mathbf{0},R=\{\epsilon\},\boldsymbol{\theta})$ in which where ϵ is sufficiently small positive constant, $\epsilon=10e-8$ say. Corollary 3.8.2 given below is useful when the MGF of mixing distribution has not closed form (or does not exist), but its characteristic function (CF) does exist.

Corollary 3.8.2. *Let $\varphi_G(t|\boldsymbol{\theta})=E(\exp\{itG\})$ denote the CF of the mixing variable G in Definition 3.8.2. Set $\eta(g)=g$ and $\Psi=(\mathbf{0},R,\boldsymbol{\theta})$ in which $R=[(1,\rho)^{\top},(\rho,1)^{\top}]$. Then*

$$F(\boldsymbol{x}|\Psi)=\begin{cases}F\big(\min\{x_1,x_2\}\big)+\dfrac{1}{2\pi}\displaystyle\int_{\frac{\pi}{2}}^{\sin^{-1}(\rho)}\varphi_G\Big[i\frac{\delta\big(\boldsymbol{x},\sin(u)\big)}{2}\Big]du, & \text{if } \rho>0,\\ \max\Big\{0,F(x_2)-F(-x_1)\Big\}+\dfrac{1}{2\pi}\displaystyle\int_{-\frac{\pi}{2}}^{\sin^{-1}(\rho)}\varphi_G\Big[i\frac{\delta\big(\boldsymbol{x},\sin(u)\big)}{2}\Big]du, & \text{if } \rho<0,\end{cases}$$

where $i^2=-1$, $|\rho|<1$ and $\delta(\boldsymbol{x},u)$ is defined as (3.185).

Corollary 3.8.3. *Let $\mathcal{M}_{G^{-1}}(t|\boldsymbol{\theta})=E(\exp\{tG^{-1}\})$ denote the MGF of the mixing variable G^{-1} in Definition 3.8.2. Set $\eta(g)=g^{-1}$ and $\Psi=(\mathbf{0},R,\boldsymbol{\theta})$ in which $R=[(1,\rho)^{\top},(\rho,1)^{\top}]$. Then*

$$F(\boldsymbol{x}|\Psi)=\begin{cases}F\big(\min\{x_1,x_2\}\big)+\dfrac{1}{2\pi}\displaystyle\int_{\frac{\pi}{2}}^{\sin^{-1}(\rho)}\mathcal{M}_{G^{-1}}\Big(-\frac{\delta\big(\boldsymbol{x},\sin(u)\big)}{2}\Big)du, & \text{if } \rho>0,\\ \max\Big\{0,F(x_2)-F(-x_1)\Big\}+\dfrac{1}{2\pi}\displaystyle\int_{-\frac{\pi}{2}}^{\sin^{-1}(\rho)}\mathcal{M}_{G^{-1}}\Big(-\frac{\delta\big(\boldsymbol{x},\sin(u)\big)}{2}\Big)du, & \text{if } \rho<0,\end{cases}$$

where $i^2=-1$, $|\rho|<1$ and $\delta(\boldsymbol{x},u)$, for $|u|<1$, is defined as (3.185).

Theorem 3.8.1, Corollary 3.8.2, and Corollary 3.8.3 are also applicable for computing the CDF of a GSM distribution when the parameter vector Ψ involves the scale matrix $\Sigma=\big[(\sigma_{1,1},\sigma_{1,2})^{\top},(\sigma_{2,1},\sigma_{2,2})^{\top}\big]$ rather than the correlation one. Corollary (3.8.4) given by the following is useful in this case.

Corollary 3.8.4. *Let $\Psi'=(\boldsymbol{\mu},\Sigma,\boldsymbol{\theta})$ and $\Psi=(\mathbf{0},R,\boldsymbol{\theta})$. Then*

$$F(\boldsymbol{x}|\Psi')=F\Big(\frac{x_1-\mu_1}{\sqrt{\sigma_{1,1}}},\frac{x_2-\mu_2}{\sqrt{\sigma_{2,2}}}\Big|\Psi\Big),$$

where $R=\{\rho_{i,j}\}$ and $\Sigma=\{\sigma_{i,j}\}$, for $i,j=1,2$, are the correlation and covariance matrices, respectively.

Theorem 3.8.1 and Corollary 3.8.4 are useful for computing the CDF of bivariate Student's t [Kotz and Nadarajah, 2004] and SH [McNeil et al., 2015] distributions, respectively. To this end, we consider two cases given by the following.

- **Student's t:** Theorem 3.8.1 specializes to the method of Genz [2004] given in (3.188) for computing the CDF of bivariate Student's t distribution if $G \sim \mathcal{G}(\nu/2, \nu/2)$, for which $\mathcal{M}_G(t) = (1 - 2t/\nu)^{\nu/2}$, $\eta(g) = g$, and $\Psi = (\mathbf{0}, R = \{\rho\}, \boldsymbol{\theta} = \nu)^\top$. Here, $F(\cdot)$ is the CDF of univariate Student's t with ν degrees of freedom.

- **SH:** For computing the CDF $F[\boldsymbol{x}|\Psi = (\mathbf{0}, R = \{\rho\}, \boldsymbol{\theta} = (a, \lambda)^\top)]$ whose PDF is given in (3.178), we may use Corollary 3.8.3 when $F(\cdot)$ is the CDF of univariate SH distribution and $\mathcal{M}_{G^{-1}}[t|\boldsymbol{\theta} = (a, a, \lambda)^\top]$ where

$$\mathcal{M}_{G^{-1}}[t|\boldsymbol{\theta} = (\psi, \chi, \lambda)^\top] = \mathcal{M}_G[t|\boldsymbol{\theta} = (\psi, \chi, -\lambda)^\top], \tag{3.197}$$

  in which $G \sim \mathcal{GIG}(\psi, \chi, \lambda)$ with PDF (2.48) and CF

$$\mathcal{M}_G[t|\boldsymbol{\theta} = (\psi, \chi, \lambda)^\top] = \Big(\frac{\psi}{\psi - 2t}\Big)^{\frac{\lambda}{2}} \frac{\mathcal{K}_\lambda(\sqrt{\chi(\psi - 2t)})}{\mathcal{K}_\lambda(\sqrt{\chi\psi})}. \tag{3.198}$$

3.8.5 Computing the CDF of trivariate Gaussian distribution

- **Owen's method**: The efforts for numerical evaluation in three-dimensional case dates back to [Owen, 1956] that suggests the use of

$$\Phi_3(\boldsymbol{x}|\mathbf{0}, R) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x_1} \exp\Big\{-\frac{y^2}{2}\Big\} \Phi_2(\boldsymbol{z}(y)|\mathbf{0}, R^\star = \{\rho^\star\}) dy, \tag{3.199}$$

  where $\boldsymbol{x} = (x_1, x_2, x_3)^\top$, $R = \{\rho_{i,j}\}$ denotes the associated 3×3 correlation matrix, $\boldsymbol{z}(y) = [z_1(y), z_2(y)]^\top$ in which

$$z_i(y) = \frac{x_{i+1} - \rho_{i+1,1} \times y}{\sqrt{1 - \rho_{i+1,1}^2}}, \ i = 1, 2, \tag{3.200}$$

  and

$$\rho^\star = \frac{\rho_{2,3} - \rho_{2,1}\rho_{1,3}}{\prod_{i=1}^2 (1 - \rho_{i+1,1}^2)^{\frac{1}{2}}}. \tag{3.201}$$

  Genz [2004] tested two methods for implementing the Owen's idea in (3.199) as follows. As the first method, using transformation $y = \Phi^{-1}(u)$, it follows that

$$\Phi_3(\boldsymbol{x}|\mathbf{0}, R) = \int_0^{\Phi(x_1)} \Phi_2\Big\{\boldsymbol{z}[\Phi^{-1}(u)]\Big|\mathbf{0}, \rho^\star\Big\} du, \tag{3.202}$$

  where $z_i(\cdot)$ and $\rho^\star$, accordingly, are given by (3.200) and (3.201). The second method, that relies on the method of Drezner [1992], uses the Gauss-Hermite rule of third kind. To this end, applying the transformation $u = y - x_1$, then more algebra shows that

$$\Phi_3(\boldsymbol{x}|\mathbf{0}, R) = \frac{\exp\{-\frac{x_1^2}{2}\}}{\sqrt{2\pi}} \int_0^\infty \exp\Big\{-\frac{u^2}{2} + ux_1\Big\} \Phi_2(\boldsymbol{z}(u)|\mathbf{0}, \rho^\star) du, \tag{3.203}$$

  where $z_i(u) = x_1 - u$, for $i = 1, 2$, is given in (3.200). Since the shortest integration interval typically yields more accurate result, hence one can choose the smallest x_i, among the components of $\boldsymbol{x}$, as the upper bound of integration in RHS of (3.199) .

- **Plackett's method**: Despite its elegance, as pointed out by [Gassmann, 2003], the Plackett's approach seems to have been largely forgotten. Plackett [1954] derives the partial differential equation

$$\frac{\partial \boldsymbol{\phi}_p(x_1, x_2, \cdots, x_p|\mathbf{0}, R)}{\partial \rho_{i,j}} = \frac{\partial^2 \boldsymbol{\phi}_p(x_1, x_2, \cdots, x_p|\mathbf{0}, R)}{\partial x_i \partial x_j}, \tag{3.204}$$

and employed it for computing $\partial L_p(\boldsymbol{a}|R)/\partial \rho_{1,2}$ as follows.

$$\begin{aligned}\frac{\partial L_p(\boldsymbol{a}|R)}{\partial \rho_{1,2}} &= \int_{a_1}^{\infty}\int_{a_2}^{\infty}\cdots\int_{a_p}^{\infty}\frac{\partial \boldsymbol{\phi}_p(\boldsymbol{x}|\mathbf{0}, R)}{\partial \rho_{1,2}}dx_1dx_2\cdots dx_p \\ &= \int_{a_3}^{\infty}\int_{a_4}^{\infty}\cdots\int_{a_p}^{\infty}\Big[\int_{a_1}^{\infty}\int_{a_2}^{\infty}\frac{\partial \boldsymbol{\phi}_p(\boldsymbol{x}|\mathbf{0}, R)}{\partial x_1 x_2}dx_1dx_2\Big]dx_3dx_4\cdots dx_p. \end{aligned}\tag{3.205}$$

Using property (3.204), the RHS of (3.205) becomes

$$\frac{\partial L_p(\boldsymbol{a}|R)}{\partial \rho_{1,2}} = \int_{a_3}^{\infty}\int_{a_4}^{\infty}\cdots\int_{a_p}^{\infty}\boldsymbol{\phi}_p\Big[(a_1, a_2, x_3, x_4, \cdots, x_p)^\top\Big|\mathbf{0}, R\Big]dx_1dx_2dx_3dx_4\cdots dx_p. \tag{3.206}$$

Let

$$R = \left(\begin{array}{c:c} R_{11} & R_{12} \\ \hdashline R_{21} & R_{22} \end{array}\right), \tag{3.207}$$

be a decomposition of matrix R in which

$$R_{11} = [(1, \rho_{2,1})^\top, (\rho_{1,2}, 1)^\top]. \tag{3.208}$$

Furthermore, set

$$\tilde{R} = R_{22} - R_{21}R_{11}^{-1}R_{12}, \tag{3.209}$$

and $\boldsymbol{x}_{[-1,-2]}(\rho_{1,2})$ is vector of length $p-2$ defined as

$$\boldsymbol{x}_{[-1,-2]}(\rho_{1,2}) = (x_3, x_4, \cdots, x_p)^\top - R_{21}R_{11}^{-1} \times (a_1, a_2)^\top, \tag{3.210}$$

whose elements, for $k = 3, 4, \cdots, p$, are

$$x_k - \frac{(\rho_{1,k} - \rho_{1,2}\rho_{2,k})a_1 + (\rho_{2,k} - \rho_{2,1}\rho_{1,k})a_2}{1 - \rho_{1,2}^2}. \tag{3.211}$$

Based on decomposition (3.207), the integrand in the RHS of (3.206) can be expressed as the product of a 2-dimensional unconditional and a $(p-2)$-dimensional conditional Gaussian distributions. It follows that

$$\begin{aligned}\frac{\partial L_p(\boldsymbol{a}|R)}{\partial \rho_{1,2}} &= \boldsymbol{\phi}_2[(a_1, a_2)^\top|\mathbf{0}, R_{11}]\int_{a_3}^{\infty}\int_{a_4}^{\infty}\cdots\int_{a_p}^{\infty}\boldsymbol{\phi}_{p-2}\Big[\boldsymbol{x}_{[-1,-2]}(\rho_{1,2})\Big|\mathbf{0}, \tilde{R}\Big]dx_3dx_4\cdots dx_p, \\ &= \boldsymbol{\phi}_2[(a_1, a_2)^\top|\mathbf{0}, R_{11}] \times L_{p-2}\Big[\boldsymbol{a}_{[-1,-2]}(\rho_{1,2})\Big|\tilde{R}\Big], \end{aligned}\tag{3.212}$$

where R_{11}, $\tilde{R} = \{\tilde{\rho}_{i,j}\}$, and $\boldsymbol{a}_{[-1,-2]}(\rho_{1,2})$ are given, accordingly, in (3.208), (3.209), and (3.210). For any choice of R such as $R^\star = \{\rho^\star_{i,j}\}$ for which computing $L_p(\boldsymbol{a}|R^\star)$ is feasible easily, consider the positive definite matrix

$$\Omega(t) = \{\omega_{i,j}(t)\} = tR + (1-t)R^\star, \tag{3.213}$$

for $t \in (0,1]$. It can be shown that

$$L_p(\boldsymbol{a}|R) = L_p(\boldsymbol{a}|R^\star) + \sum_{i<j}^{p}\sum \int_0^1 (\rho_{i,j} - \rho^\star_{i,j}) \frac{\partial L_p(\boldsymbol{a}|\Omega(t))}{\partial \rho_{i,j}} dt. \tag{3.214}$$

The quantities $L_p(\boldsymbol{a}|R)$ and $L_p(\boldsymbol{a}|R^\star)$ are known in the literature as the *target* and *reference* probabilities, respectively [Gassmann, 2003]. The Plackett's recursion formula is then, by combining (3.212) and (3.214), given by

$$\begin{aligned} L_p(\boldsymbol{a}|R) =& L_p(\boldsymbol{a}|R^\star) + \sum_{i<j}^{p}\sum \int_0^1 (\rho_{i,j} - \rho^\star_{i,j}) \\ & \times \boldsymbol{\phi}_2\Big[(a_i, a_j)^\top \Big| \mathbf{0}, R_{11}(t)\Big] \times L_{p-2}\Big[\boldsymbol{a}_{[-i,-j]}(\omega_{i,j}(t)) \Big| \tilde{\Omega}(t)\Big] dt, \end{aligned} \tag{3.215}$$

where $R_{11}(t) = \big[(1, \omega_{i,j}(t))^\top, (\omega_{j,i}(t), 1)^\top\big]$, $\tilde{\Omega}(t) = t\tilde{R} + (1-t)R^\star$ in which $\tilde{R}$ is given in (3.209), and elements of $\boldsymbol{a}_{[-i,-j]}(\omega_{i,j}(t))$ are given, for $k \neq \{i,j\} = 1, \cdots, p$, by

$$a_k - \frac{[\omega_{i,k}(t) - \omega_{i,j}(t)\omega_{j,k}(t)]a_i + [\omega_{j,k}(t) - \omega_{j,i}(t)\omega_{i,k}(t)]a_j}{1 - \omega^2_{i,j}(t)}. \tag{3.216}$$

It is immediate from (3.215) that, if $\rho_{i,j} \geq \rho^\star_{i,j}$ for all $i < j$, then $L_p(\boldsymbol{a}|R) \geq L_p(\boldsymbol{a}|R^\star)$ Slepian [1962].

Drezner [1994] introduces two approaches for computing $L_3(\boldsymbol{a}|R)$ based on the Plackett's method as follows.

I. **First method**: it follows from (3.212) that

$$\frac{\partial L_3(\boldsymbol{a}|R)}{\partial \rho_{1,2}} = \boldsymbol{\phi}_2\big[(a_1, a_2)^\top | \mathbf{0}, R_{11}\big] \times L_1(a_{[-1,-2]}(\rho_{1,2})|1), \tag{3.217}$$

where $\boldsymbol{a} = (a_1, a_2, a_3)^\top$, $R_{11} = \big[(1, \rho_{1,2})^\top, (\rho_{2,1}, 1)^\top\big]$, and

$$\begin{aligned} a_{[-1,-2]}(\rho_{1,2}) =& a_3 - R_{21}R_{11}^{-1} \times (a_1, a_2)^\top \\ =& a_3 - \frac{a_2[\rho_{2,3} - \rho_{2,1}\rho_{1,3}] + a_1[\rho_{1,3} - \rho_{1,2}\rho_{2,3}]}{(1 - \rho^2_{1,2})}. \end{aligned}$$

Using the correlation matrix

$$R_1^\star = \boldsymbol{I}_3 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{3.218}$$

for the reference probability it follows that

$$L_3(\boldsymbol{a}|R_1^\star) = \prod_{m=1}^{3} L_1(a_m|1) = \prod_{m=1}^{3} \bar{\Phi}(a_m), \tag{3.219}$$

$$\Omega(t) = \begin{bmatrix} 1 & \omega_{1,2}(t) & \omega_{1,3}(t) \\ \omega_{2,1}(t) & 1 & \omega_{2,3}(t) \\ \omega_{3,1}(t) & \omega_{3,2}(t) & 1 \end{bmatrix} = \begin{bmatrix} 1 & t\rho_{1,2} & t\rho_{1,3} \\ t\rho_{2,1} & 1 & t\rho_{2,3} \\ t\rho_{3,1} & t\rho_{3,2} & 1 \end{bmatrix}, \tag{3.220}$$

$$R_{11} = R_{11}(t) = \big[(1, t\rho_{i,j})^\top, (t\rho_{j,i}, 1)^\top\big], \tag{3.221}$$

$$\tilde{\Omega}(t) = 1 - \frac{t^2\rho^2_{i,k} + t^2\rho^2_{j,k} - 2t^3\rho_{i,j}\rho_{i,k}\rho_{j,k}}{1 - t^2\rho^2_{i,j}}, \tag{3.222}$$

$$a_{[-i,-j]}(\rho_{i,j}(t)) = a_k - \frac{[t\rho_{i,k} - t^2\rho_{i,j}\rho_{j,k}]a_i + [t\rho_{j,k} - t^2\rho_{j,i}\rho_{i,k}]a_j}{1 - t^2\rho^2_{i,j}}, \tag{3.223}$$

where $i < j$ and $k \neq \{i, j\} = 1, \cdots, 3$. Applying information given in (3.219)-(3.223) on the RHS of (3.215) for $p = 3$, then more algebra shows

$$L_3(\boldsymbol{a}|R) = \prod_{i=1}^{3} L_1(a_i|1) + \sum_{i<j}^{3}\sum \int_0^1 \rho_{i,j}\boldsymbol{\phi}_2\Big[(a_i, a_j)^\top \Big| \mathbf{0}, R_{11}(t)\Big]\bar{\Phi}\Big[\frac{a_{[-i,-j]}(\rho_{i,j}(t))}{\sqrt{\tilde{\Omega}(t)}}\Big]dt,$$

where $\bar{\Phi}(\cdot) = 1 - \Phi(\cdot)$.

II. **Second method**: under the second approach, the correlation matrix of the reference probability is

$$R_2^\star = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & \rho_{2,3} \\ 0 & \rho_{3,2} & 1 \end{bmatrix}. \tag{3.224}$$

It follows that

$$L_3(\boldsymbol{a}|R_2^\star) = L_1(a_1|1) \times L_2\big[(a_2, a_3)^\top | R = \{\rho_{2,3}\}\big], \tag{3.225}$$

$$\Omega(t) = \begin{bmatrix} 1 & \omega_{1,2}(t) & \omega_{1,3}(t) \\ \omega_{2,1}(t) & 1 & \omega_{2,3}(t) \\ \omega_{3,1}(t) & \omega_{3,2}(t) & 1 \end{bmatrix} = \begin{bmatrix} 1 & t\rho_{1,2} & t\rho_{1,3} \\ t\rho_{2,1} & 1 & \rho_{2,3} \\ t\rho_{3,1} & \rho_{3,2} & 1 \end{bmatrix}, \tag{3.226}$$

$$R_{11}(t) = \big[(1, t\rho_{1,j})^\top, (t\rho_{j,1}, 1)^\top\big], \tag{3.227}$$

$$\tilde{\Omega}(t) = 1 - \frac{t^2\rho_{1,k}^2 + \rho_{k,j}^2 - 2t^2\rho_{1,j}\rho_{j,k}\rho_{1,k}}{1 - t^2\rho_{1,j}^2}, \tag{3.228}$$

$$a_{[-1,-j]}(\rho_{i,j}(t)) = a_k - \frac{[t\rho_{1,k} - t\rho_{1,j}\rho_{j,k}]a_1 + [\rho_{j,k} - t^2\rho_{1,j}\rho_{1,k}]a_j}{1 - t^2\rho_{1,j}^2}, \tag{3.229}$$

for $k \neq j = 2, 3$. Applying information given in (3.225)-(3.229) on the RHS of (3.215) for $p = 3$, then more algebra shows

$$\begin{aligned} L_3(\boldsymbol{a}|R) =& L_1(a_1|1) \times L_2\big[(a_2, a_3)^\top | R = \{\rho_{2,3}\}\big] \\ &+ \sum_{j=2}^{3} \int_0^1 \rho_{1,j}\boldsymbol{\phi}_2\Big[(a_1, a_j)^\top \Big| \mathbf{0}, R_{11}(t) = \{t\rho_{1,j}\}\Big]\bar{\Phi}\Big[\frac{a_{[-1,-j]}(\rho_{i,j}(t))}{\sqrt{\tilde{\Omega}(t)}}\Big]dt, \end{aligned}$$

where $\bar{\Phi}(\cdot) = 1 - \Phi(\cdot)$.

Based on a 5-point Gauss-Legendre rule, as pointed out by Drezner [1994], both methods described above work well with maximum error less than 10^{-7} when $\rho_{\max} \leq 0.5$. If $\rho_{\max} = 0.9$, then in most cases the maximum error is limited by 10^{-7}, but attains 10^{-5} in very few cases. Moreover, it is shown that the error is more dependent on the determinant of R than it is on the magnitude of $\rho_{\max}$. It is shown, by simulation, that if $|R| > 0.15$, then the error is limited by 10^{-7}.

In what follows, the `R` functions `TVN1_Drez` and `TVN2_Drez` are given for implementing, accordingly, the first and second methods of Drezner [1994].

```
R> TVN1_Drez <- function(a, Mu, R)
+{
+ p <- length(a); threshold <- rep(5, p); a <- a - Mu
+ a <- ifelse( a < -threshold, -threshold, a )
+ w <- c(0.2369268851, 0.4786286705, 0.5688888889, 0.4786286705, 0.2369268851)
+ node <- c(0.9061798459, 0.5384693101, 0, -0.5384693101, -0.9061798459)
```

```
+ Sum <- 0
+ for( i in 1:(p - 1) )
+ {
+  for( j in (i + 1):p )
+  {
+   index <- c( R[ c(-i, -j), c(i, j) ] )
+   r <- c(1:p)[ c(-i, -j) ]; t <- (node + 1)/2
+   R11 <- matrix( c(1, R[i, j], R[i, j], 1), nrow = 2, ncol = 2)
+   p1 <- sqrt( 1 - t^2*R[i, j]^2 )
+   p2 <- exp( -1/2*( a[i]^2 + a[j]^2 - 2*t*a[i]*a[j]*R[i,j] )/p1^2 )/( 2*pi*p1 )
+   Omega <- 1 - ( (t*R[r,i])^2 + (t*R[r,j])^2 - 2*t^3*R[r,i]*R[r,j]*R[i,j] )/p1^2
+   a_tilde <- a[r] - ( ( t*R[r,i] - t^2*R[r,j]*R[i,j] )*a[i] +
+              ( t*R[r,j] - t^2*R[r,i]*R[i,j] )*a[j] )/p1^2
+   p3 <- pnorm( a_tilde/sqrt(Omega), lower.tail = F )
+   Sum <- Sum + R[i, j]*sum( w*p2*p3 )/2
+  }
+ }
+ prod( pnorm(a, lower.tail = F) ) + Sum
+}
```

```
R> TVN2_Drez <- function(a, Mu, R)
+{
+ p <- length(a); threshold <- rep(5, p); a <- a - Mu
+ a <- ifelse( a < -threshold, -threshold, a )
+ w <- c(0.2369268851, 0.4786286705, 0.5688888889, 0.4786286705, 0.2369268851)
+ node <- c(0.9061798459, 0.5384693101, 0, -0.5384693101, -0.9061798459)
+ Sum <- 0
+ for( j in 2:3 )
+ {
+  i <- 1
+  index <- c( R[ c(-i, -j), c(i, j) ] )
+  r <- c(1:p)[ c(-i, -j) ]
+  t <- (node + 1)/2
+  R11 <- matrix( c(1, R[i, j], R[i, j], 1), nrow = 2, ncol = 2)
+  p1 <- sqrt( 1 - t^2*R[i, j]^2 )
+  p2 <- exp( -1/2*( a[i]^2 + a[j]^2 - 2*t*a[i]*a[j]*R[i,j] )/p1^2 )/( 2*pi*p1 )
+  Omega <- 1 - ( (t*R[r,i])^2 + (R[r,j])^2 - 2*t^2*R[r,i]*R[r,j]*R[i,j] )/p1^2
+  a_tilde <- a[r] - ( (t*R[r,i]- t*R[r,j]*R[i,j])*a[i]+
+             ( R[r,j]- t^2*R[r,i]*R[i,j])*a[j] )/p1^2
+  p3 <- pnorm( a_tilde/sqrt(Omega), lower.tail = F )
+  Sum <- Sum + R[i, j]*sum( w*p2*p3 )/2
+ }
+ R23 <- matrix( c(1, R[2,3], R[3,2], 1), nrow = 2, ncol = 2)
+ pnorm(a[1], lower.tail = F)*BVN_Drez(a[-1], c(0, 0), R23) + Sum
+}
```

Example 3-20

We are willing to compute $\Phi_3(\boldsymbol{a}|\boldsymbol{0}, R) = L_3(-\boldsymbol{a}|R)$ using the first method of Drezner [1994] for which $\boldsymbol{a} = (-1.2, -1.0, 0.5)^\top$ and $R = [(1, 0.7, -0.2)^\top, (0.7, 1, -0.4)^\top, (-0.2, -0.4, 1)^\top]$. Hence, command `TVN1_Drez(-c(-1.2,-1.0,0.5), rep(0,3), R)` gives

$$\Phi_3(\boldsymbol{a}|\boldsymbol{0}, R) \approx 0.208019949666041.$$

■

3.8.6 Computing the CDF of trivariate Student's t distribution

Herein, we describe four methods for computing the CDF of Student's t distribution with ν degrees of freedom proposed by Genz [2004]. The first three methods are based on the Plackett's

formula for the multivariate Gaussian distribution. The last one relies on the method of [Owen, 1956] for computing the CDF of trivariate Gaussian distribution given in (3.199). First, we introduce those methods that are based on the Plackett's formula. To this end, we consider the CDF of the Student's t distribution, proposed by Cornish [1954], as follows.

$$T_\nu(\boldsymbol{b}|\mathbf{0},R)=\frac{2^{1-\nu/2}}{\Gamma(\nu/2)}\int_0^\infty u^{\nu-1}\exp\Big\{-\frac{u^2}{2}\Big\}\Phi_3\Big(\frac{u\boldsymbol{b}}{\sqrt{\nu}}\Big|\mathbf{0},R\Big)du. \tag{3.230}$$

In order to generalize the Plackett's formula to the Student's t distribution, one needs to compute $\partial T_\nu(\boldsymbol{b}|\mathbf{0},R)\partial\rho_{1,2}$. Using (3.230), it turns out that

$$\frac{\partial T_\nu(\boldsymbol{b}|\mathbf{0},R)}{\partial\rho_{1,2}}=\frac{2^{1-\nu/2}}{\Gamma(\nu/2)}\int_0^\infty u^{\nu-1}\exp\Big\{-\frac{u^2}{2}\Big\}\frac{\partial}{\partial\rho_{1,2}}\Phi_3\Big(\frac{u\boldsymbol{b}}{\sqrt{\nu}}\Big|\mathbf{0},R\Big)du. \tag{3.231}$$

We notice that the Plackett's formula (3.212) can be represented as

$$\frac{\partial\Phi_3(\boldsymbol{b}|\mathbf{0},R)}{\partial\rho_{i,j}}=\frac{\exp\Big\{-\frac{1}{2}\delta\big[(b_i,b_j)^\top,\rho_{i,j}\big]\Big\}}{2\pi\sqrt{1-\rho_{i,j}^2}}\Phi\big[h(\rho_{i,j})\big], \tag{3.232}$$

where $\boldsymbol{b}=(b_1,b_2,b_3)^\top$, $\delta(\cdot,\cdot)$ is given in (3.185), and

$$h(\rho_{i,j},\rho_{i,k})=\frac{b_k(1-\rho_{i,j}^2)-b_j(\rho_{j,k}-\rho_{i,j}\rho_{i,k})-b_i(\rho_{i,k}-\rho_{i,j}\rho_{j,k})}{\big[(1-\rho_{i,j}^2)(1-\rho_{i,j}^2-\rho_{i,k}^2-\rho_{j,k}^2+2\rho_{i,j}\rho_{i,k}\rho_{j,k})\big]^{\frac{1}{2}}}, \tag{3.233}$$

with $k\neq i\neq j=\{1,2,3\}$. Substituting (3.232) into the RHS of (3.231) gives

$$\frac{\partial T_\nu(\boldsymbol{b}|\mathbf{0},R)}{\partial\rho_{i,j}}=\frac{2^{1-\frac{\nu}{2}}}{\Gamma(\frac{\nu}{2})}\int_0^\infty\frac{u^{\nu-1}\exp\Big\{-\frac{u^2}{2}\big(1+\frac{1}{\nu}\delta\big[(b_i,b_j)^\top,\rho_{i,j}\big]\big)\Big\}}{(2\pi)^{\frac{3}{2}}\sqrt{1-\rho_{i,j}^2}}\int_{-\infty}^{\frac{uh(\rho_{i,j},\rho_{i,k})}{\sqrt{\nu}}}\exp\Big\{-\frac{x^2}{2}\Big\}dxdu.$$

Applying change of variable $y=\sqrt{\nu}x/u$ yields

$$\frac{\partial T_\nu(\boldsymbol{b}|\mathbf{0},R)}{\partial\rho_{i,j}}=\frac{2^{1-\frac{\nu}{2}}}{\sqrt{\nu}\Gamma(\frac{\nu}{2})}\int_{-\infty}^{h(\rho_{i,j},\rho_{i,k})}\int_0^\infty\frac{u^\nu\exp\Big\{-\frac{u^2}{2}\Big[1+\frac{\delta\left[(b_i,b_j)^\top,\rho_{i,j}\right]+y^2}{\nu}\Big]\Big\}}{(2\pi)^{\frac{3}{2}}\sqrt{1-\rho_{i,j}^2}}dudy.$$

Using another transformation of the form

$$r=u\Big[1+\frac{\delta\big[(b_i,b_j)^\top,\rho_{i,j}\big]+y^2}{\nu}\Big]^{\frac{1}{2}},$$

giving

$$\begin{aligned}\frac{\partial T_\nu(\boldsymbol{b}|\mathbf{0},R)}{\partial\rho_{i,j}}=&\frac{2^{1-\frac{\nu}{2}}}{\sqrt{\nu}\Gamma(\frac{\nu}{2})}\int_{-\infty}^{h(\rho_{i,j},\rho_{i,k})}\frac{\Big[1+\frac{\delta\left[(b_i,b_j)^\top,\rho_{i,j}\right]+y^2}{\nu}\Big]^{-\frac{\nu+1}{2}}}{(2\pi)^{\frac{3}{2}}\sqrt{1-\rho_{i,j}^2}}\int_0^\infty r^\nu\exp\Big\{-\frac{r^2}{2}\Big\}drdy\\=&\frac{\Gamma(\frac{\nu+1}{2})}{\sqrt{\nu\pi}\Gamma(\frac{\nu}{2})}\int_{-\infty}^{h(\rho_{i,j},\rho_{i,k})}\frac{\Big[1+\frac{\delta\left[(b_i,b_j)^\top,\rho_{i,j}\right]+y^2}{\nu}\Big]^{-\frac{\nu+1}{2}}}{2\pi\sqrt{1-\rho_{i,j}^2}}dy.\end{aligned}$$

Taking into account the fact that

$$\Big[1+\frac{\delta\big[(b_i,b_j)^\top,\rho_{i,j}\big]+y^2}{\nu}\Big]=\Big[1+\frac{\delta\big[(b_i,b_j)^\top,\rho_{i,j}\big]}{\nu}\Big]\times\Big[1+\frac{y^2}{\nu+\delta\big[(b_i,b_j)^\top,\rho_{i,j}\big]}\Big],$$

it follows that

$$\frac{\partial T_\nu(\boldsymbol{b}|\boldsymbol{0},R)}{\partial \rho_{i,j}} = \frac{\Gamma\big(\frac{\nu+1}{2}\big)}{\sqrt{\nu\pi}\Gamma\big(\frac{\nu}{2}\big)} \frac{\Big[1+\frac{\delta[(b_i,b_j)^\top,\rho_{i,j}]}{\nu}\Big]^{-\frac{\nu+1}{2}}}{2\pi\sqrt{1-\rho_{i,j}^2}} \int_{-\infty}^{h(\rho_{i,j},\rho_{i,k})} \Big[1+\frac{y^2}{\nu+\delta\big[(b_i,b_j)^\top,\rho_{i,j}\big]}\Big]^{-\frac{\nu+1}{2}} dy.$$

We need the final transformation $z = y(1+\nu^{-1}\delta\big[(b_i,b_j)^\top,\rho_{i,j}\big])^{-1/2}$ to see that

$$\frac{\partial T_\nu(\boldsymbol{b}|\boldsymbol{0},R)}{\partial \rho_{i,j}} = \frac{\Big[1+\nu^{-1}\delta[(b_i,b_j)^\top,\rho_{i,j}]\Big]^{-\frac{\nu}{2}}}{2\pi\sqrt{1-\rho_{i,j}^2}} \frac{\Gamma\big(\frac{\nu+1}{2}\big)}{\sqrt{\nu\pi}\Gamma\big(\frac{\nu}{2}\big)} \int_{-\infty}^{\eta} \Big[1+\frac{z^2}{\nu}\Big]^{-\frac{\nu+1}{2}} dz,$$

where $\eta = h(\rho_{i,j},\rho_{i,k})(1+\nu^{-1}\delta[(b_i,b_j)^\top,\rho_{i,j}])^{-1/2}$ or equivalently

$$\frac{\partial T_\nu(\boldsymbol{b}|\boldsymbol{0},R)}{\partial \rho_{i,j}} = \frac{\Big[1+\nu^{-1}\delta[(b_i,b_j)^\top,\rho_{i,j}]\Big]^{-\frac{\nu}{2}}}{2\pi\sqrt{1-\rho_{i,j}^2}} T_\nu\left[\frac{h(\rho_{i,j},\rho_{i,k})}{\sqrt{1+\nu^{-1}\delta[(b_i,b_j)^\top,\rho_{i,j}]}}\right]. \tag{3.234}$$

Relation in (3.234) can regarded as the Plackett's formula for the bivariate Student's t distribution. The first and second methods of Let $T_\nu(\boldsymbol{b}|\boldsymbol{0},R_1^\star)$ and $T_\nu(\boldsymbol{b}|\boldsymbol{0},R_2^\star)$ denote two reference probabilities in which

$$R_1^\star = \begin{bmatrix} 1 & \rho_{1,2}^\star & \rho_{1,3} \\ \rho_{2,1}^\star & 1 & \rho_{2,3} \\ \rho_{3,1} & \rho_{3,2} & 1 \end{bmatrix} \quad \text{and} \quad R_2^\star = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & \rho_{2,3} \\ 0 & \rho_{3,2} & 1 \end{bmatrix}. \tag{3.235}$$

Regarding $T_\nu(\boldsymbol{b}|\boldsymbol{0},R_1^\star)$ and $T_\nu(\boldsymbol{b}|\boldsymbol{0},R_2^\star)$ as the two reference probabilities, by integrating both sides of (3.234), the first and second method of Genz [2004] are given by the following.

I. **First method**: using the reference probability $T_\nu(\boldsymbol{b}|\boldsymbol{0},R_1^\star)$ for which $R_1^\star$ is given in (3.235), the first method suggests to compute $T_\nu(\boldsymbol{b}|\boldsymbol{0},R)$ as follows.

$$\begin{aligned} T_\nu(\boldsymbol{b}|\boldsymbol{0},R) =& T_\nu(\boldsymbol{b}|\boldsymbol{0},R_1^\star) \\ &+ \frac{1}{2\pi}\int_{\rho_{1,2}^\star}^{\rho_{1,2}} \frac{\Big(1+\nu^{-1}\delta[(b_1,b_2)^\top,r]\Big)^{-\frac{\nu}{2}}}{\sqrt{1-r^2}} T_\nu\left[\frac{h(r\rho_{1,2},r\rho_{1,3})}{\sqrt{1+\nu^{-1}\delta[(b_1,b_2)^\top,r]}}\right] dr, \end{aligned} \tag{3.236}$$

where $h(\cdot,\cdot)$ is given in (3.233).

II. **Second method**: using the reference probability $T_\nu(\boldsymbol{b}|\boldsymbol{0},R_1^\star)$ for which $R_2^\star$ is given in (3.235), the second method suggests to compute $T_\nu(\boldsymbol{b}|\boldsymbol{0},R)$ as follows.

$$\begin{aligned} T_\nu(\boldsymbol{b}|\boldsymbol{0},R) =& T_\nu(\boldsymbol{b}|\boldsymbol{0},R_2^\star) \\ &+ \int_0^1 \frac{\Big(1+\nu^{-1}\delta[(b_1,b_2)^\top,t\rho_{1,2}]\Big)^{-\frac{\nu}{2}}}{2\pi\rho_{1,2}^{-1}\sqrt{1-t^2\rho_{1,2}^2}} T_\nu\left[\frac{h(t\rho_{1,2},t\rho_{1,3})}{\sqrt{1+\nu^{-1}\delta[(b_1,b_2)^\top,t\rho_{1,2}]}}\right] dt \\ &+ \int_0^1 \frac{\Big(1+\nu^{-1}\delta[(b_1,b_3)^\top,t\rho_{1,3}]\Big)^{-\frac{\nu}{2}}}{2\pi\rho_{1,3}^{-1}\sqrt{1-t^2\rho_{1,3}^2}} T_\nu\left[\frac{h(t\rho_{1,3},t\rho_{1,2})}{\sqrt{1+\nu^{-1}\delta[(b_1,b_3)^\top,t\rho_{1,3}]}}\right] dt. \end{aligned} \tag{3.237}$$

We note that the second method in (3.237) is computationally superior to the first one in (3.236).

III. **Third method**: efficient computing of (3.237) is possible if the reference probability $T_\nu(\boldsymbol{b}|\mathbf{0}, R_2^\star)$ be replaced with

$$\begin{aligned} T_\nu(\boldsymbol{b}|\mathbf{0}, R_2^\star) =& T_\nu(\boldsymbol{b}|\mathbf{0}, R_3^\star) \\ &+ \frac{1}{2\pi}\int_s^{\rho_{2,3}} \frac{\Big(1+\nu^{-1}\delta[(b_2,b_3)^\top, r]\Big)^{-\frac{\nu}{2}}}{\sqrt{1-r^2}} T_\nu\left[\frac{b_1}{\sqrt{1+\nu^{-1}\delta[(b_2,b_3)^\top, r]}}\right] dr, \end{aligned} \tag{3.238}$$

where

$$R_3^\star = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & s \\ 0 & s & 1 \end{bmatrix},$$

with $s = \text{sign}(\rho_{2,3})$. Hence, by combining (3.237) and (3.238), then the third method suggests to compute $T_\nu(\boldsymbol{b}|\mathbf{0}, R)$ as follows.

$$\begin{aligned} T_\nu(\boldsymbol{b}|\mathbf{0}, R) =& T_\nu(\boldsymbol{b}|\mathbf{0}, R_3^\star) \\ &+ \frac{1}{2\pi}\int_s^{\rho_{2,3}} \frac{\Big(1+\nu^{-1}\delta[(b_2,b_3)^\top, r]\Big)^{-\frac{\nu}{2}}}{\sqrt{1-r^2}} T_\nu\left[\frac{b_1}{\sqrt{1+\nu^{-1}\delta[(b_2,b_3)^\top, r]}}\right] dr \\ &+ \int_0^1 \frac{\Big(1+\nu^{-1}\delta[(b_1,b_2)^\top, t\rho_{1,2}]\Big)^{-\frac{\nu}{2}}}{2\pi\rho_{1,2}^{-1}\sqrt{1-t^2\rho_{1,2}^2}} T_\nu\left[\frac{h(t\rho_{1,2}, t\rho_{1,3})}{\sqrt{1+\nu^{-1}\delta[(b_1,b_2)^\top, t\rho_{1,2}]}}\right] dt \\ &+ \int_0^1 \frac{\Big(1+\nu^{-1}\delta[(b_1,b_3)^\top, t\rho_{1,3}]\Big)^{-\frac{\nu}{2}}}{2\pi\rho_{1,3}^{-1}\sqrt{1-t^2\rho_{1,3}^2}} T_\nu\left[\frac{h(t\rho_{1,3}, t\rho_{1,2})}{\sqrt{1+\nu^{-1}\delta[(b_1,b_3)^\top, t\rho_{1,3}]}}\right] dt, \end{aligned} \tag{3.239}$$

where

$$T_\nu(\boldsymbol{b}|\mathbf{0}, R_3^\star) = \begin{cases} T_\nu\Big[(b_1, \min\{b_2, b_3\})\Big|\mathbf{0}, R=0\Big], & \text{if } s=+1, \\ \max\Big\{0, T_\nu\big[(b_1,b_2)|\mathbf{0}, R=0\big] - T_\nu\big[(b_1,-b_3)|\mathbf{0}, R=0\big]\Big\}, & \text{if } s=-1. \end{cases}$$

It should be noted that for computing the second term in RHS of (3.239) the use of transformation $r = \sin(\theta)$ may be useful.

IV. **Fourth method**: based on representation (3.199), proposed by [Owen, 1956] for the CDF of trivariate Gaussian distribution, it can be shown that the CDF of Student's t distribution with ν degrees of freedom is

$$T_\nu(\boldsymbol{b}|\mathbf{0}, R) = \frac{\Gamma\big(\frac{\nu+1}{2}\big)}{\sqrt{\nu\pi}\,\Gamma\big(\frac{\nu}{2}\big)} \int_{-\infty}^{b_1} \Big[1+\frac{y^2}{\nu}\Big]^{-\frac{\nu+1}{2}} T_\nu(\boldsymbol{z}(y)|\mathbf{0}, R_4^\star = \{\rho^\star\}) dy, \tag{3.240}$$

where $\boldsymbol{b} = (b_1, b_2, b_3)^\top$, $R = \{\rho_{i,j}\}$ denotes the associated 3×3 correlation matrix, $\boldsymbol{z}(y) = \big[z_1(y), z_2(y)\big]^\top$ in which

$$z_i(y) = \frac{(\nu+1)^{\frac{1}{2}} \times (b_{i+1} - \rho_{i+1,1} \times y)}{\sqrt{(\nu+y^2)(1-\rho_{i+1,1}^2)}},\ i = 1, 2,$$

and

$$\rho^{\star} = \frac{\rho_{2,3} - \rho_{1,2}\rho_{1,3}}{\prod_{i=1}^{2}(1-\rho_{i+1,1}^{2})^{\frac{1}{2}}}.$$

Applying transformation $y = T_{\nu}^{-1}(u)$ on the RHS of (3.240) giving

$$T_{\nu}(\boldsymbol{b}|\boldsymbol{0}, R) = \int_{0}^{T_{\nu}(b_1)} T_{\nu}\Big[\boldsymbol{z}(T_{\nu}^{-1}(t))\Big|\boldsymbol{0}, R^{\star} = \{\rho^{\star}\}\Big]dt, \tag{3.241}$$

where $T_{\nu}(\cdot)$ and $T_{\nu}^{-1}(\cdot)$ denote, accordingly, the CDF and quantile function of a standard Student's t distribution with ν degrees of freedom.

Genz [2004] shows by simulation that the third method in (3.239) outperforms, in almost all of cases, the other three methods both in terms of maximum (average) error and taken time for implementation [Genz, 2004, Tables 3-5, pp. 258-259]. For instance, when $\nu = 1$, using 6-point Gauss-Legendre and adaptive algorithm (with $\epsilon = 0.1$) the maximum (average) errors becomes $10^{-5}(10^{-7})$ and $10^{-13}(10^{-17})$, respectively. In what follows, the `R` functions `TVT_Genz` is given for implementing (3.239) when computing the CDF of 3-dimensional Student's t distribution.

```
R> TVT_Genz <- function(b, R, nu, k)
+{
+ out <- quad_rule(k, type = "LE", alpha = 0, beta = 0)
+ node <- out$node; w <- out$weight; eps <- 10e-8
+ R[2, 3] <- ifelse( abs(R[2, 3]) < eps, eps, R[2, 3])
+ s <- sign(R[2, 3])
+ T1 <- BVT_Genz( c( b[1], min(b[-1]) ), eps, nu, k )
+ T2 <- BVT_Genz( c( b[1], b[2]), eps, nu, k )
+ T3 <- BVT_Genz( c( b[1], -b[3]), eps, nu, k )
+ T_star <- ifelse(s < 0, max(0, T2 - T3 ), T1)
+ L <- ( asin(R[2, 3]) - s*pi/2 )/2
+ x <- (node + 1)*L + s*pi/2
+ delta <- ( b[2]^2 + b[3]^2 - 2*sin(x)*b[2]*b[3] )/cos(x)^2
+ p1 <- pt( b[1]/sqrt( 1 + delta/nu ), df = nu )
+ CDF_23 <- T_star + L/(2*pi)*sum( w*( 1 + delta/nu )^(-nu/2)*p1 )
+ x <- (node + 1)/2
+ J <- c(2, 3)
+ CDF <- 0
+ for( i in 1:2 )
+ {
+  r <- x*R[J[i], 1]
+  p2 <- sqrt( 1 - r^2 )
+  delta <- ( b[1]^2 + b[J[i]]^2 - 2*r*b[1]*b[J[i]] )/p2^2
+  p3 <- ( 1 + delta/nu )^(-nu/2)/p2
+  p4 <- 1 - R[1, 2]^2*x^2 - R[1, 3]^2*x^2 - R[2, 3]^2 +
+        2*x^2*R[1, 2]*R[1, 3]*R[2, 3]
+  p5 <- b[J[-i]]*( 1 - r^2 ) - b[1]*x*( R[J[-i], 1] - R[J[i], 1]*R[2, 3] ) -
+        b[J[i]]*( R[3, 2] - x^2*R[3, 1]*R[2, 1] )
+  p6 <- p2*sqrt(p4)
+  h <- p5/p6
+  p5 <- pt( h/sqrt(1 + delta/nu), df = nu)
+  CDF <- CDF + R[J[i], 1]/(4*pi)*sum( w*p3*p5 )
+ }
+return( CDF_23 + CDF )
+}
```

3.8.7 Rectangular Gaussian probability with positive correlation matrix

The rectangular Gaussian probability may be represented as

$$\boldsymbol{\Phi}_p(\mathcal{R}^p|\mathbf{0},R)=\int_{a_1}^{b_1}\int_{a_2}^{b_2}\cdots\int_{a_p}^{b_p}\boldsymbol{\phi}_p(\boldsymbol{x}|\mathbf{0},R)dx_1\cdots dx_p, \tag{3.242}$$

where region $\mathcal{R}^p$ is a hypercube defined as (3.270). Let $R^+=\{\rho^+_{i,j}\}$ denote the absolute value of elements of correlation matrix $R=\{\rho_{i,j}\}$, herein we are willing to show how the Gaussian probability in (3.242) can be expressed in terms of $\boldsymbol{\Phi}_p(\mathcal{R}^p_\star|\mathbf{0},R^+)$ where

$$\mathcal{R}^p_\star=\{\boldsymbol{x}\in\mathbb{R}^p|\boldsymbol{a}^\star\leq\boldsymbol{x}\leq\boldsymbol{b}^\star\}, \tag{3.243}$$

where $\boldsymbol{a}^\star=(a^\star_1,\cdots,a^\star_p)^\top$ and $\boldsymbol{b}^\star=(b^\star_1,\cdots,b^\star_p)^\top$. Suppose $s_{i,j}=\text{sign}(\rho_{i,j})$, $i,j=1,\cdots,p$, in which $\text{sign}(u)$ is the well-known sign function defined as $\text{sign}(u)=+1$ if $u\geq 0$ and $\text{sign}(u)=-1$ otherwise. Each normal distribution benefit from the well-known property of changeability in the sense that sign of each coordinate and corresponding elements of correlation matrix R can be interchanged. For example, taking $p=2$, we can write

$$\boldsymbol{\phi}_2(s_{1,2}\times x_1,x_2|\mathbf{0},\rho^+_{1,2})=\boldsymbol{\phi}_2(x_1,s_{1,2}\times x_2|\mathbf{0},\rho^+_{1,2}). \tag{3.244}$$

Evidently, for each set of three arbitrary random variables with second finite moment such as Y_1, Y_2, and Y_3, if we have $\rho_{2,1}=\text{corr}(Y_2,Y_1)\lesseqgtr 0$ and $\rho_{1,3}=\text{corr}(Y_1,Y_3)\lesseqgtr 0$, then $\rho_{2,3}=\text{corr}(Y_2,Y_3)>0$. In general, for $i,j=1,2,\cdots,p$, we have $s_{i,j}=s_{i,1}\times s_{1,j}$ that implies $s_{i,j}<0$ if and only if $s_{1,i}=s_{i,1}<0$ or $s_{1,j}=s_{j,1}<0$. Hence, if the jth element, $j=1,\cdots,p$, of vector $\boldsymbol{s}_1=(s_{1,1},\cdots,s_{1,p})^\top$ is negative, then the jth coordinate of $\boldsymbol{x}$ within $\boldsymbol{\phi}_p(\boldsymbol{x}|\mathbf{0},R)$ is multiplied by $s_{1,j}$ while the corresponding column in correlation matrix, that is $R[,j]=\boldsymbol{\rho}_j=\{\rho_{1,j},\rho_{2,j},\cdots,\rho_{p,j}\}$, is replaced with its absolute value $R^+[,j]=\boldsymbol{\rho}^+_j$. Taking into account the above fact, we start with a Lemma that is important for establishing the main result of this work.

Lemma 3.8.1. *Let $\boldsymbol{\phi}_p(x_1,x_2,\cdots,x_p|\mathbf{0},R)$ be the PDF of a p-variate zero-mean Gaussian distribution with correlation matrix R and $s_{i,j}=sign(\rho_{i,j})$, $i,j=1,2,\cdots,p$. Then*

$$\boldsymbol{\phi}_p(x_1,x_2,\cdots,x_p|\mathbf{0},R)=\boldsymbol{\phi}_p(s_{1,1}\times x_1,s_{1,2}\times x_2,\cdots,s_{1,p}\times x_p|\mathbf{0},R^+), \tag{3.245}$$

where R^+ is the absolute value of correlation matrix R.

Using Lemma 3.8.1 while applying a change of variable $x_i=s_{1,i}\times z_i$, $i=1,2,\cdots,p$, on the RHS of (3.242), then Lemma 3.8.2 given below, shows how the CDF of two Gaussian distributions one with correlation matrix R and the other with R^+ are connected to each other.

Lemma 3.8.2. *Let $\boldsymbol{\Phi}_p(\mathcal{R}^p|\mathbf{0},R)$ be the CDF of a p-variate zero-mean normal distribution with correlation matrix $R=\{\rho_{i,j}\}$ computed on hypercube defined in (3.244) and further assume $s_{i,j}=sign(\rho_{i,j})$, $i,j=1,2,\cdots,p$. Then*

$$\boldsymbol{\Phi}_p(\mathcal{R}^p|\mathbf{0},R)=\boldsymbol{\Phi}_p(\mathcal{R}^p_\star|\mathbf{0},R^+), \tag{3.246}$$

where $\mathcal{R}^p_\star$ is defined as (3.243) for which elements of $\boldsymbol{a}^\star$ and $\boldsymbol{b}^\star$ are as

$$\begin{cases} a^\star_j=a_j \quad \textit{and } b^\star_j=b_j, & \textit{if } s_{1,j}=+1,\\ a^\star_j=-b_j \ \textit{and } b^\star_j=-a_j, & \textit{if } s_{1,j}=-1, \end{cases}$$

for $j=1,2,\cdots,p$.

Example 3-21

Recall from Example 3-20, suppose $\boldsymbol{a} = (-1, -1, -\infty)^\top$ and $\boldsymbol{b} = (\infty, \infty, 2)^\top$. Based on Lemma 3.8.2, it is not hard to check that

$$\boldsymbol{\Phi}_3(\mathcal{R}^3|\boldsymbol{0}, R) = \boldsymbol{\Phi}_3(\mathcal{R}^3_\star|\boldsymbol{0}, R^+) \approx 0.755968249358648,$$

where $\boldsymbol{a}^\star = (-1, -1, -2)$ and $\boldsymbol{b}^\star = (\infty, \infty, \infty)$. Not surprisingly, it can be seen easily that this probability is equal to $L_3(\boldsymbol{a}^\star|R^+)$. ■

3.8.8 Computing the CDF of multivariate Gaussian distribution: General case

[Forbes and Wraith, 2014] show that the PDF of each p-dimensional Gaussian distribution can be represented as the product of p univariate Gaussian PDFs as

$$\boldsymbol{\phi}_p(\boldsymbol{x}|\boldsymbol{\mu}, \Sigma) = \boldsymbol{\phi}_p(\boldsymbol{x}|\boldsymbol{\mu}, VDV^\top) = \prod_{i=1}^{p} \phi\Big(\boldsymbol{x}\Big|\big[V^\top \boldsymbol{z}\big]_i, d_{i,i}\Big) = \prod_{i=1}^{p} \phi\Big(\big[V^\top \boldsymbol{x}\big]_i\Big|\big[V^\top \boldsymbol{\mu}\big]_i, d_{i,i}\Big), \quad (3.247)$$

where $\boldsymbol{z} = \boldsymbol{x} - \boldsymbol{\mu}$, $\big[V^\top \boldsymbol{z}\big]_i$ denotes the ith element of vector $V^\top \boldsymbol{z}$, and $d_{i,i}$ is the ith diagonal entry of the diagonal matrix $D = (d_{i,j})$ whose main diagonal elements are eigen values of Σ. Let p-dimensional random vector $\boldsymbol{X}$ follows $\mathcal{N}(\boldsymbol{\mu}, \Sigma)$. For computing $\mathcal{P}_G = P(\boldsymbol{a} \leq \boldsymbol{X} \leq \boldsymbol{b})$, based on spectral decomposition, we have

$$\begin{aligned}
\mathcal{P}_G &= \int_{a_1}^{b_1} \cdots \int_{a_p}^{b_p} \prod_{i=1}^{p} \phi\Big(\big[V^\top \boldsymbol{x}\big]_i\Big|\big[V^\top \boldsymbol{\mu}\big]_i, d_{i,i}\Big) dx_1 \cdots dx_p \\
&= \int_{a_1-\mu_1}^{b_1-\mu_1} \cdots \int_{a_p-\mu_p}^{b_p-\mu_p} \prod_{i=1}^{p} \phi\Big(\big[V^\top \boldsymbol{x}\big]_i\Big|0, d_{i,i}\Big) dx_1 \cdots dx_p \\
&= \int_{-1}^{1} \cdots \int_{-1}^{1} \prod_{i=1}^{p} \Big[\frac{b_i - a_i}{2}\Big] \phi\Big(\Big[V^\top \boldsymbol{y}\Big]_i\Big|0, d_{i,i}\Big) dx_1 \cdots dx_p,
\end{aligned} \quad (3.248)$$

where $\boldsymbol{y} = (\boldsymbol{x} + 1)(\boldsymbol{b} - \boldsymbol{a})/2 + \boldsymbol{a} - \boldsymbol{\mu}$. Using the Gauss-Legendre rule the quantity $\mathcal{P}_G$ can be approximated as

$$\mathcal{P}_G \approx \prod_{i=1}^{p} \Big[\frac{b_i - a_i}{2\sqrt{2\pi d_{i,i}}}\Big] \times \sum_{i_1=1}^{k_1} \cdots \sum_{i_p=1}^{k_p} \omega_{1i_1} \cdots \omega_{pi_p} \exp\Big\{-\frac{1}{2} \sum_{r=1}^{p} \frac{\big(\sum_{c=1}^{p} v_{r,c} \times z_{ci_c}\big)^2}{d_{r,r}}\Big\}, \quad (3.249)$$

where $z_{ci_c} = (x_{ci_c} + 1)(b_c - a_c)/2 + a_c - \mu_c$ in which x_{ci_c} is the i_cth node computed based on a k_c-point Gauss-Legendre rule for the cth coordinate and furthermore $v_{r,c}$ denotes the (r, c)th entry of matrix $V^\top$.

Example 3-22

Herein, we want to compute $\mathcal{P}_G$ given in Example 3-16 based on spectral decomposition discussed earlier. For $p = 2$ and $k_1 = k_2 = k$, it follows from (3.249) that

$$\begin{aligned}
\mathcal{P}_G &\approx \prod_{i=1}^{2} \Big[\frac{b_i - a_i}{2\sqrt{2\pi d_{i,i}}}\Big] \times \sum_{i=1}^{k} \sum_{j=1}^{k} \omega_{1i}\omega_{2j} \exp\Big\{-\frac{\big(v_{1,1}z_{1i} + v_{1,2}z_{2j}\big)^2}{2d_{1,1}} - \frac{\big(v_{2,1}z_{1i} + v_{2,2}z_{2j}\big)^2}{2d_{2,2}}\Big\}, \\
&= \sum_{i=1}^{k} \sum_{j=1}^{k} \omega_{1i}\omega_{2j} g(z_{1i}, z_{2i}),
\end{aligned} \quad (3.250)$$

where $z_{ij} = (x_{ij}+1)(b_i - a_i)/2 + a_i - \mu_i$ in which x_{ij} (or ω_{ij}) for $i = 1, 2$ and $j = 1, \cdots, k$ is the jth node (or weight) computed based on a k-point Gauss-Laguerre rule for the ith coordinate. Table 3.13 shows the performance of the k-point Gauss-Legendre rule for approximating $\mathcal{P}_G$ when $\boldsymbol{\mu} = (0,0)^\top$ and covariance matrix $\Sigma = [(1,\rho)^\top, (\rho,1)^\top]$. It is worth to note that both of the Cholesky and spectral decompositions show the same results in cases discussed in Table 3.13.

Table 3.13: ARE for approximating $\mathcal{P} = P(\boldsymbol{a} \leq \boldsymbol{X} \leq \boldsymbol{b})$ using k-point Gauss-Legendre rule based on Cholesky and spectral decompositions.

$\rho = 0.99$			ARE		
		Exact value[1]	$k = 10$	$k = 20$	$k = 40$
$a = (-\infty, -\infty)^\top$	$b = (2,2)^\top$	0.97421138	1.245136942	0.20878929	5.673800e-04
$a = (2,2)^\top$	$b = (\infty, \infty)^\top$	0.01971164	0.003664231	0.00198487	1.984895e-03
$a = (-2,-2)^\top$	$b = (2,2)^\top$	0.94842275	0.520591497	0.01685183	7.154119e-08
$\rho = 0.50$					
			$k = 10$	$k = 20$	$k = 40$
$a = (-\infty, -\infty)^\top$	$b = (2,2)^\top$	0.95855268	6.519226e-05	6.557075e-05	6.557075e-05
$a = (2,2)^\top$	$b = (\infty, \infty)^\top$	0.00405294	8.493075e-03	8.493075e-03	8.493075e-03
$a = (-2,-2)^\top$	$b = (2,2)^\top$	0.91711185	2.453014e-09	2.856401e-09	2.856405e-09
$\rho = 0.00$					
			$k = 10$	$k = 20$	$k = 40$
$a = (-\infty, -\infty)^\top$	$b = (2,2)^\top$	0.95501730	6.438875e-05	6.481121e-05	6.481121e-05
$a = (2,2)^\top$	$b = (\infty, \infty)^\top$	0.00051756	2.765946e-03	2.765946e-03	2.765946e-03
$a = (-2,-2)^\top$	$b = (2,2)^\top$	0.91106974	6.285136e-09	6.829245e-09	6.829248e-09

[1] The exact value has been computed using `Maple` software up to eight decimal places.

Figure 3.9 displays the address of nodes and product of weights using the Gauss-Legendre rule with $k_1 = k_2 = 5$ points for approximating $\mathcal{P}_G$ when $\boldsymbol{\mu} = (0,0)^\top$ and $\Sigma = [(1,\rho)^\top, (\rho,1)^\top]$. More investigations reveal that a k-point Gauss-Legendre rule based on spectral decomposition shows superior performance than the Cholesky decomposition since in the latter case the integration region is rectangular. In general, for approximating a p-dimensional integral one needs to compute k^p nodes and weights that computationally is expensive when p becomes large. In such case, the Gaussian quadrature rules are not applicable. Alternatively, we suggest the use of `R` package `TruncatedNormal` that has been developed by [Botev, 2017] and is available at CRAN. Our proposed function `spec_quad` works well for small p ($p \leq 5$, say).

```
R> Spec_quad <- function(a, b, Mu, Sigma, k)
+ {
+ p <- length(Mu)
+ n <- k^p
+ threshold <- 5*sqrt( diag(Sigma) )
+ a0 <- a - Mu
+ b0 <- b - Mu
+ a0 <- ifelse( a0 < -threshold, -threshold, a0 )
+ b0 <- ifelse( b0 > threshold, threshold, b0 )
+ e <- eigen(Sigma)
+ ev <- e$vectors
+ out <- quad_rule(k, type = "LE", alpha = 0, beta = 0)
+ weight <- out$weight
+ node <- out$node
+ weight_matrix <- as.data.frame( matrix(rep(weight, p), nrow = k, ncol = p) )
+ node_matrix <- as.data.frame( matrix(rep(node , p), nrow = k, ncol = p) )
```

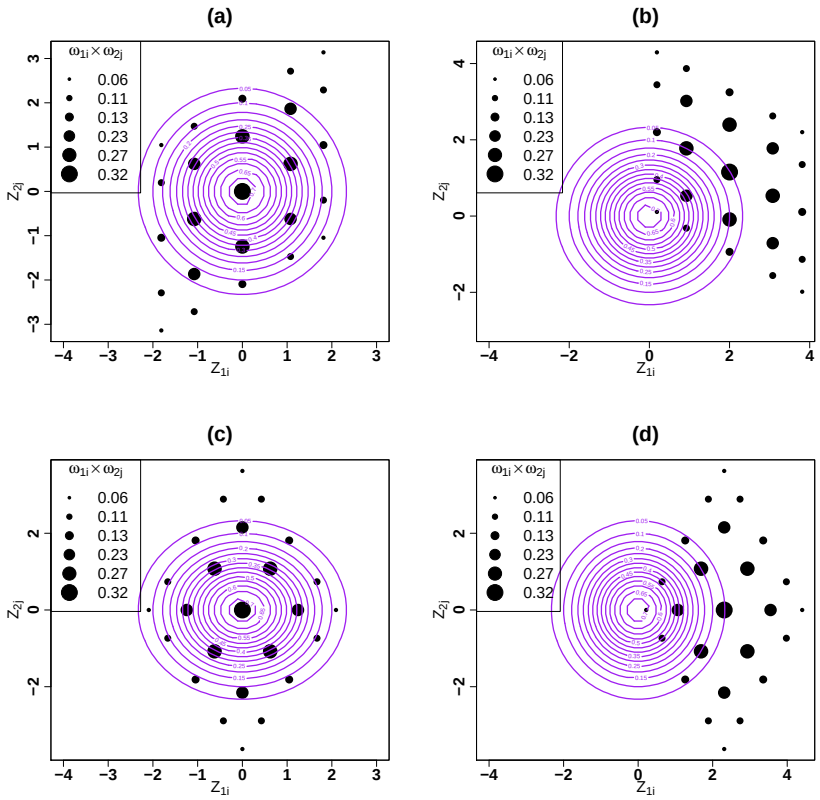


Figure 3.9: First row: contour plot of $g(z_{1i}, z_{2j})$ given in the RHS of (3.145). The filled black points show the address of nodes based on the Cholesky decomposition for (a): $\boldsymbol{a} = (-2, -2)^\top$, $\boldsymbol{b} = (2, 2)^\top$, and $\rho = -0.50$ and (b): $\boldsymbol{a} = (0, 0)^\top$, $\boldsymbol{b} = (\infty, \infty)^\top$, and $\rho = 0.50$. Second row: contour plot of $g(z_{1i}, z_{2j})$ given in the RHS of (3.250). The filled black points show the address of nodes based on the spectral decomposition for (c): $\boldsymbol{a} = (-2, -2)^\top$, $\boldsymbol{b} = (2, 2)^\top$, and $\rho = -0.50$ and (d): $\boldsymbol{a} = (0, 0)^\top$, $\boldsymbol{b} = (\infty, \infty)^\top$, and $\rho = 0.50$.

```
+ weight_grid <- as.matrix( do.call( expand.grid, weight_matrix ), nrow = n, ncol = p)
+ omega <- sapply( 1:n, function(i) prod( weight_grid[i, ] ) )
+ node_grid <- as.matrix( do.call( expand.grid, node_matrix ), nrow = n, ncol = p)
+ zij <- node_grid %*% diag((b0 - a0)/2) + matrix( (b0 + a0)/2 , n, p, byrow = T )
+ p1 <- apply( ( t( zij%*%sweep(ev, 2, sqrt(e$values), "/") ) )^2, 2, sum)
+ I1 <- 1/sqrt( (2*pi)^p*prod(e$values) )*prod( (b0 - a0)/2 )*sum( omega*exp( -p1/2 ) )
+ min( abs(I1), 1)
+ }
```

Problems

1. The Simpson's 1/3 rule rule for numerically computation of an integral over interval $[a, b]$ works by computing the integrand $f(\cdot)$ at two end points and the midpoint of interval as

$$\int_a^b f(x)dx \approx \frac{h}{3}\Big[f(a) + f(b) + 4f(a + h)\Big],$$

where $h = (b - a)/2$. The error in approximating above integral becomes $-f^{(4)} \times (b - a)^5/2880$ in which $f^{(\xi)}$, for $a < \xi < b$, is the fourth order derivative of $f(\cdot)$ over $[a, b]$. Approximate the CDF of a standard Gaussian distribution at $z = \{-3, 0, 3\}$ using the Simpson's rule and compare the results with those of the `R` built-in function `pnorm(z)`.

2. When the correlation matrix $R = \{\rho_{i,j}\}$ of a zero-mean p-variate Gaussian distribution is tridiagonal so that $\rho_{i,i-1} = \rho$ for $2 \le i \le p$ and $\rho_{i,j} = 0$, for $|i-j| > 1$, then the exact value of orthant probability, that is $L(\mathbf{0}|R)$, would be known for some special cases [Miwa et al., 2003, pp. 225-226]. For instance, if $\rho = -1/2$, then $L(\mathbf{0}|R) = 1/(1+p)!$. Check this fact numerically for $p = 2$ and $p = 3$ using functions `BVN_Drez` and `TVN1_Drez`, respectively.

3. Let $F(x|\Psi)$ is the CDF of a GSM distribution. Show that (3.179) can be approximated as follows.

$$F(x|\Psi) \approx \begin{cases} \frac{1}{2} + \frac{(b-a)\text{sign}(x)}{2\sqrt{\pi}} \int_{-1}^{1} H\Big(\frac{x^2}{2\sigma^2 y^2}\Big|\boldsymbol{\theta}\Big) \exp\{-y^2\} dz, & \text{if} \quad \eta(g) = g, \\ \frac{1}{2} + \frac{(b-a)\text{sign}(x)}{2\sqrt{\pi}} \int_{-1}^{1} \bar{H}\Big(\frac{2\sigma^2 y^2}{x^2}\Big|\boldsymbol{\theta}\Big) \exp\{-y^2\} dz, & \text{if} \quad \eta(g) = \frac{1}{g}, \end{cases}$$

   where $H(\cdot|\boldsymbol{\theta})$ is the CDF of mixing distribution and $\bar{H}(\cdot|\boldsymbol{\theta}) = 1 - H(\cdot|\boldsymbol{\theta})$.

4. Explain that how $F(x|\Psi)$ in (3.179) can be approximated by a k-point Gauss-Hermite rule of the first kind.

5. Let $f(x,y) = \max\Big\{0, F(x|\nu) - F(-y|\nu)\Big\}$ in which $F(x|\nu)$ is the CDF of the Student's t distribution with ν degrees of freedom. Prove that $f(x,y) = f(y,x)$ for any paired $(x,y) \in \mathbb{R}^2$. Is this result valid for any symmetric distribution around origin ?

6. Prove the continuity property of $F(\boldsymbol{x}|\Psi)$ in Theorem 3.8.1 at $\rho = 0$. Hint: integrate both sides of (3.192) over interval $(-\epsilon, \epsilon)$ and then take limit when $\epsilon \downarrow 0$ to show that

$$\lim_{\epsilon \to 0} |F(\boldsymbol{x}|\Psi) - F(\boldsymbol{x}|\Psi')| \approx \lim_{\epsilon \to 0} \frac{|\epsilon|}{2\pi} \mathcal{M}_G\Big[-\frac{1}{2}\|\boldsymbol{x}\|^2\Big] = 0,$$

   where $\Psi = (\mathbf{0}, R = \{\epsilon\}, \boldsymbol{\theta})$, $\Psi' = (\mathbf{0}, R = \{-\epsilon\}, \boldsymbol{\theta})$, and $\|\cdot\|$ denotes the Euclidean norm.

7. Let $\boldsymbol{X} \sim \mathcal{N}_p(\boldsymbol{\mu}, R = \{\rho\})$ and P_i denote the ith set among all $p!$ permutations of the set $\boldsymbol{x} = \{x_1, \cdots, x_p\}$.

   (a) Prove that $\Phi_p(P_i|\mathbf{0}, R = \{\rho\}) = \Phi_p(P_j|\mathbf{0}, R = \{\rho\})$ for $i = j = 1, \cdots, p!$. In special case, when $p = 2$, we have

$$\Phi_2((x_1, x_2)^\top|\mathbf{0}, R = \{\rho\}) = \Phi_2((x_2, x_1)^\top|\mathbf{0}, R = \{\rho\}). \tag{3.251}$$

   (b) Does identity (3.251) hold true if we have $\boldsymbol{\mu} = (\mu_1, \mu_2)^\top = (\mu, \mu)^\top$?

8. Prove property (3.108) of Owen's T function.

9. Write $B(x, y|\rho)$ to denote the function proposed by Owen [1956] as

$$B(x, y|\rho) = \frac{\Phi(x)}{2} + \frac{\Phi(y)}{2} - T(x, a_x) - T(y, a_y) - \begin{cases} 0, & \text{if} \quad xy > 0, \\ \frac{1}{2}, & \text{if} \quad xy = 0 \text{ and } x + y \ge 0 \end{cases}, \tag{3.252}$$

   for computing $\Phi_2(\boldsymbol{x}|\mathbf{0}, \rho)$ where $\boldsymbol{x} \in \mathbb{R}^2$. Use the following `R` code for approximating the Owen's $B(x, y|\rho)$ function in (3.252) and compare its performance with functions `BVN_Drez` and `Spec_quad` given in this section for this purpose.

```
B_owen <- function(x, y, rho, k)
{
if( x*y > 0 | (x*y == 0 & x + y >= 0) ){ p1 <- 0 }else{ p1 <- 1/2 }
a_x <- y/( x*sqrt(1 - rho^2) ) - rho/sqrt(1 - rho^2)
a_y <- x/( y*sqrt(1 - rho^2) ) - rho/sqrt(1 - rho^2)
out <- quad_rule(k, type = "LE"); x0 <- out$node; w <- out$weight;
T_x <- a_x/(4*pi)*sum( w*exp(-x^2/2*(1 + (a_x*x0)^2))/(1 + (a_x*x0)^2) )
T_y <- a_y/(4*pi)*sum( w*exp(-y^2/2*(1 + (a_y*x0)^2))/(1 + (a_y*x0)^2) )
pnorm(x)/2 + pnorm(y)/2 - T_x - T_y - p1
}
```

10. Let $\boldsymbol{X} \sim \mathcal{N}_p(\boldsymbol{0}, \Sigma)$ with PDF $\phi_p(\boldsymbol{x}|\boldsymbol{0}, \Sigma)$ that can be represented, using the well-known *inversion formula*, as

$$\phi_p(\boldsymbol{x}|\boldsymbol{0}, \Sigma) = \frac{1}{(2\pi)^p} \int \cdots \int \exp\{-i\boldsymbol{t}^\top \boldsymbol{x} - \frac{1}{2}\boldsymbol{t}^\top \Sigma \boldsymbol{t}\} dt_1 \cdots dt_p, \tag{3.253}$$

where $i^2 = -1$ and $\boldsymbol{t} \in \mathbb{R}^p$. Verify identity (3.204) by taking derivatives, first with respect to $\rho_{r,c}$ and, second with respect to x_r and x_c from RHS of (3.253) for $r < c = 1, \cdots, p$.

11. The random vector $\boldsymbol{Y} = (Y_1, \ldots, Y_p)^T$ with CF

$$\psi(\boldsymbol{t}|\boldsymbol{\Sigma}, \boldsymbol{\alpha}, \lambda) = \Big(1 + \frac{\boldsymbol{t}^\top \Sigma \boldsymbol{t}}{2} - i\boldsymbol{\alpha}^T \boldsymbol{t}\Big)^{-\lambda},$$

where $i^2 = -1$, $\boldsymbol{t} \in \mathbb{R}^p$, $\boldsymbol{\alpha} \in \mathbb{R}^p$, $a > 0$, and Σ is $p \times p$ non-negative definite symmetric matrix is called generalized asymmetric Laplace (GAL) distribution Kotz et al. [2012]. In fact each GAL distribution is a GSMM model with representation

$$\boldsymbol{Y} = \boldsymbol{\mu} + \boldsymbol{\alpha} G + \sqrt{G}\boldsymbol{X}, \tag{3.254}$$

where $\boldsymbol{X} \sim \mathcal{N}_p(\boldsymbol{0}, \Sigma)$ and $G \sim \mathcal{G}(\lambda, 1)$ are independent. Prove that the PDF of $\boldsymbol{Y}$ becomes

$$f(\boldsymbol{y}|\Psi) = \frac{2\exp\Big\{(\boldsymbol{y} - \boldsymbol{\mu})^\top \Sigma^{-1} \boldsymbol{\alpha}\Big\}}{(2\pi)^{p/2}|\Sigma|^{1/2}\Gamma(\lambda)} \Big(\frac{d_1(\boldsymbol{y})}{d_2(\boldsymbol{y})}\Big)^{\frac{p}{4} - \frac{\lambda}{2}} K_{\lambda - \frac{p}{2}}\big(\sqrt{d_1(\boldsymbol{y}) d_2(\boldsymbol{y})}\big), \tag{3.255}$$

where $\Psi = (\boldsymbol{\mu}, \boldsymbol{\alpha}, \Sigma, \lambda)$, $d_1(\boldsymbol{y}) = 2 + \delta(\boldsymbol{\alpha}, \Sigma)$, $d_2(\boldsymbol{y}) = \delta(\boldsymbol{x} - \boldsymbol{\mu}, \Sigma)$. It may be interesting if in (3.177) we let $\lambda = 1$, $\psi = 2$, and $\chi \to 0$, then PDFs in (3.255) and (3.177) are equal.

12. The PDF of a univariate SH distribution is

$$f(x|\Psi) = \Big[\frac{a + \sigma^{-2}(x - \mu)^2}{a}\Big]^{\frac{\lambda - 1/2}{2}} \frac{\mathcal{K}_{\lambda - 1/2}\big(\sqrt{a[a + \sigma^{-2}(x - \mu)^2]}\big)}{\sqrt{2\pi}\sigma \mathcal{K}_\lambda(a)}, \tag{3.256}$$

where $\Psi = (\mu, \sigma, \boldsymbol{\theta})$ in which $\boldsymbol{\theta} = (a, \lambda)^\top$. Describe a method for computing its CDF numerically using the Gaussian quadrature rule and write down the pertinent `R` code.

13. If $G \sim \mathcal{GIG}(a, a, \lambda)$, then it follows from (3.197), (3.198), and the fact $\mathcal{K}_\lambda(x) = \mathcal{K}_{-\lambda}(x)$ that

$$\mathcal{M}_{G^{-1}}(t|\boldsymbol{\theta}) = \Big(\frac{a}{a - 2t}\Big)^{-\frac{\lambda}{2}} \frac{\mathcal{K}_\lambda(\sqrt{a(a - 2t)})}{\mathcal{K}_\lambda(a)},$$

where $\boldsymbol{\theta} = (a, a, \lambda)^\top$. Based on the above result, use the Corollary 3.8.3 for computing the CDF of bivariate SH distribution with PDF is given in (3.178) numerically using the Gaussian quadrature rule. Write the `R` code down similar to Example 3-20 for bivariate Student's t distribution.

14. It is well known that if $(X,Y)^\top \sim \mathcal{N}_2(\mathbf{0}, R=\{\rho\})$, then $Y|X=h$ is distributed normally. But, $Y_h = Y|X \le h$, for $h \in \mathbb{R}$, does not follow a Gaussian distribution.

    (a) Prove that the PDF of Y_h is [Johnson et al., 1972, pp. 112-114]:

    $$f(t) = \frac{\phi(t)}{\Phi(h)}\Phi\Big[\frac{h-\rho t}{\sqrt{1-\rho^2}}\Big],$$

    with mean $\mu_h = -\rho\phi(h)/\Phi(h)$ and variance $\sigma_h^2 = 1+\rho h\mu - \mu^2$ in which $-1 \le \rho \le 1$.

    (b) Taking account into the fact that when ρ is small, $|\rho| \le 0.5$ say, then the PDF's of Y_h and $Z \sim \mathcal{N}(\mu_h, \sigma_h^2)$ are virtually indistinguishable. Prove that

    $$\Phi((x,y)^\top|\mathbf{0}, R=\{\rho\}) \approx \Phi(x)\Phi\Big[\frac{y-\mu_h}{\sigma_h}\Big]. \tag{3.257}$$

    (c) Mee and Owen [1983] proposes the following three-step algorithm with maximum error less than 0.0008 for approximating $\Phi\big((c,d)|\mathbf{0}, R=\{r\}\big)$ when $|r| \le 0.5$.

    i. Choose quantities c and d so that $|c| \ge |d|$. Such choices always exist since identity (3.251) states $\Phi\big((c,d)|\mathbf{0}, R=\{r\}\big)$=$\Phi\big((d,c)|\mathbf{0}, R=\{r\}\big)$;

    ii. If $c \le 0$, then set $x=c$, $y=d$, $\rho=r$, and approximate $\Phi\big((c,d)|\mathbf{0}, R=\{r\}\big)$ by (3.257);

    iii. If $c > 0$, then set $x=-c$, $y=d$, $\rho=-r$, and approximate $\Phi\big((c,d)^\top|\mathbf{0}, R=\{r\}\big) = \Phi(d) - \Phi\big((-c,d)^\top|\mathbf{0}, R=\{-r\}\big)$ by $\Phi(y) - \Phi\big((x,y)^\top|\mathbf{0}, R=\{\rho\}\big)$.

    Apply the above algorithm for approximating $\Phi\big((0.2,1)^\top|\mathbf{0}, R=\{0.7\}\big) = 0.558416\cdots$ and check that its absolute error is around 0.000233.

15. Apply the first method of Drezner [1994] to show that the orthant probability $L_3(\mathbf{0}|R)$, becomes explicitly

    $$L_3(\mathbf{0}|R) = \frac{1}{8} + \frac{1}{4\pi}\sum\sum_{i<j}\sin^{-1}(\rho_{i,j}).$$

16. Let $R=\{\rho_{i,j}\}$ denote the correlation matrix of a 3-dimensional Gaussian distribution.

    (a) Show that $|R| = 1-\rho_{1,2}^2-\rho_{1,3}^2-\rho_{2,3}^2+2\rho_{1,2}\rho_{1,3}\rho_{2,3}$.

    (b) Show that $|R| \le 1$.

    (c) For reference matrix $R_2^\star = \{\rho_{i,j}^\star\}$ given in (3.224), show that $|\Omega(t)| = 1-t^2\rho_{1,2}^2 - t^2\rho_{1,3}^2 - \rho_{2,3}^2 + 2t^2\rho_{1,2}\times\rho_{1,3}\times\rho_{2,3}$ where $\Omega(t)$ is given in (3.213).

    (d) Show that if $|\Omega(t)|$ is positive for $t=1$, then it is positive for $0 \le t \le 1$. Hint: show that $\rho_{1,2}^2+\rho_{1,3}^2-2\rho_{1,2}\times\rho_{1,3}\times\rho_{2,3} \ge 0$.

17. Prove relation (3.230).

3.9 Moments of truncated distributions

Truncated distribution is widely used in different fields of statistics such as, survival analysis, regression analysis, and model-based clustering. The truncated random variable may take any value on region defined by (2.52). If both the lower and upper bounds of region $\mathcal{R}$ are finite, then we have a doubly truncated of random variable. If the lower (upper) bound of region $\mathcal{R}$ is not finite, then we have a right (left) truncated random variable. In other word, the support of doubly, right, and left truncated random variable are, accordingly, (a,b), $(-\infty,b)$, and (a,∞).

Write $X_{\mathcal{R}}$ to denote random variable X truncated on region $\mathcal{R} = \{x \in \mathbb{R} | a \leq x \leq b\}$. The PDF of random variable $X_{\mathcal{R}}$ truncated on (a, b) is simply given by

$$f_{X_{\mathcal{R}}}(x) = \frac{f(x)}{\int_{\mathcal{R}} f(u)du}, \tag{3.258}$$

where $x \in \mathcal{R}$. Each truncated random variable $X_{\mathcal{R}}$ is in fact a conditional version of random variable X. In other words

$$X_{\mathcal{R}} = X | X \in \mathcal{R}.$$

Let $F(\cdot)$ denotes the CDF of random variable X. The CDF of $\mathcal{X}$ can be represented in terms of $F(\cdot)$ as follows.

$$F_{X_{\mathcal{R}}}(x) = \frac{F(x) - F(a)}{F(b) - F(a)},$$

where $x \in \mathcal{R}$. Hence, in univariate case, the rth moment of truncated random variable X on (a, b), for $r = 1, 2, 3, \cdots$ is

$$\Omega^r = E(X^r | a < X < b) = \frac{\int_a^b u^r f(u)du}{F(b) - F(a)}, \ -\infty < a < x < b < \infty. \tag{3.259}$$

In general, let $\boldsymbol{X}_{\mathcal{R}^p}$ denote the truncated version of p-dimensional random vector $\boldsymbol{X}$ on region (hyper-cube) $\mathcal{R}^p$ given in (2.52). Then the PDF of $\boldsymbol{X}_{\mathcal{R}^p}$ is

$$\frac{f(\boldsymbol{x})}{\int_{\boldsymbol{a}}^{\boldsymbol{b}} f(\boldsymbol{u})d\boldsymbol{u}} = \frac{f(\boldsymbol{x})}{\mathcal{P}_F}, \tag{3.260}$$

where $\boldsymbol{x} \in \mathcal{R}^p$ and $F(\cdot)$ is the CDF of $\boldsymbol{X}$. Hence, the first and second moments of truncated random vector $\boldsymbol{X}$, respectively, are

$$\begin{aligned}
\boldsymbol{\Omega} =& E(\boldsymbol{X}_{\mathcal{R}^p}) = E(\boldsymbol{X} | a_1 \leq X_1 \leq b_1, \cdots, a_p \leq X_p \leq b_p), \\
\boldsymbol{\Omega}^2 =& E(\boldsymbol{X}_{\mathcal{R}^p} \boldsymbol{X}_{\mathcal{R}^p}^\top) = E(\boldsymbol{X}\boldsymbol{X}^\top | a_1 \leq X_1 \leq b_1, \cdots, a_p \leq X_p \leq b_p).
\end{aligned}$$

Herein, the concern of study mainly is devoted to computing the first and second moments given by since these quantities have received more attention than other moments in practice. If $\boldsymbol{X} \sim f(\cdot)$, by definition, the first moment of $\boldsymbol{X}$ is computed as

$$\boldsymbol{\Omega} = \frac{1}{\mathcal{P}_F} \int_{\boldsymbol{a}}^{\boldsymbol{b}} \boldsymbol{u} f(\boldsymbol{u})d\boldsymbol{u}. \tag{3.261}$$

The Monte Carlo approximation of $\boldsymbol{\Omega}$ can be obtained by considering the instrumental distribution with PDFs $f(\cdot)/[F(a) - F(b)]$. Here, we can use the rejection approach for simulating from truncated random variable as follows. Suppose, $x_1, \cdots, x_N$, for sufficiently large N, are realizations from instrumental with PDF f_1, the Monte Carlo approximation of $\boldsymbol{\Omega}$ is obtained as

$$\hat{\boldsymbol{\Omega}} = \frac{1}{N[F(a) - F(b)]} \sum_{i=1}^{N} f(x_i). \tag{3.262}$$

The pseudo code for sampling from $f(\cdot)/[F(a) - F(b)]$ is given as follows.

1. Read a, b, and N;
2. Set $j = 0$ and $y = 0$;

3. Simulate $x \sim f(\cdot)$

4. If $a < x < b$, then accept x as a generation from $f(\cdot)/\big[F(a) - F(b)\big]$, set $y = y + x$ and $j = j + 1$;

5. If $j = N$, then go to the next step, otherwise return to step 2;

6. Approximate Ω as $\hat{\Omega} = y/N$;

7. End.

The focus of this chapter is placed on compute the the first and second moments of truncated Gaussian distribution in univariate and multivariate cases. Herein, we proceed to compute the exact values of first and second moments of the truncated univariate and multivariate Gaussian distributions.

3.10 Moments of truncated Gaussian distribution in univariate case

Recall from Example 2-8 that deals with simulating from truncated Gaussian distribution on region given by (2.52), herein we are willing to compute the moments of this family.

3.10.1 First two moments of truncated Gaussian distribution

Let $X \sim \mathcal{N}(\mu, \sigma^2, \mathcal{R})$. By definition, from (3.259) for $r = 1$, it follows that

$$\Omega_1 = E(X) = \frac{1}{\mathcal{P}_G} \int_{\mathcal{R}} x\phi(x|\mu, \sigma^2)dx, \tag{3.263}$$

where $\mathcal{P}_G = \Phi(b|\mu, \sigma^2) - \Phi(a|\mu, \sigma^2)$ and $\mathcal{R} = \{x \in \mathbb{R}|a < x < b\}$. It is easy to check that

$$\frac{\partial}{\partial \mu}\phi(x|\mu, \sigma^2) = \Big(\frac{x - \mu}{\sigma^2}\Big)\phi(x|\mu, \sigma^2). \tag{3.264}$$

Integrating both sides of (3.264) and dividing by $\mathcal{P}_G$ yields

$$\frac{1}{\mathcal{P}_G} \int_a^b \frac{\partial}{\partial \mu}\phi(x|\mu, \sigma^2)dx = \frac{\Omega}{\sigma^2} - \frac{\mu}{\sigma^2}.$$

It follows that

$$\begin{aligned}\Omega &= \mu + \frac{\sigma^2}{\mathcal{P}_G}\frac{\partial}{\partial \mu}\int_{a-\mu}^{b-\mu} \phi(u|0, \sigma^2)du \\ &= \mu + \frac{\sigma^2}{\mathcal{P}_G}\Big[\phi(a|\mu, \sigma^2) - \phi(b|\mu, \sigma^2)\Big].\end{aligned}$$

Hence,

$$\Omega = \mu + \sigma^2 \frac{\phi(a|\mu, \sigma^2) - \phi(b|\mu, \sigma^2)}{\Phi(b|\mu, \sigma^2) - \Phi(a|\mu, \sigma^2)}. \tag{3.265}$$

By definition, from (3.259) for $m = 2$, it follows that

$$\Omega^2 = E(X^2) = \frac{1}{\mathcal{P}_G} \int_a^b x^2\phi(x|\mu, \sigma^2)dx,$$

Taking the second order derivative from (3.264) with respect to μ, we have

$$\frac{\partial^2}{\partial \mu^2}\phi(x|\mu,\sigma^2) = \Big(\frac{x-\mu}{\sigma^2}\Big)^2 \phi(x|\mu,\sigma^2) - \frac{\phi(x|\mu,\sigma^2)}{\sigma^2}. \tag{3.266}$$

Integrating both sides of (3.266), dividing by $\mathcal{P}_G$, and then rearranging yields

$$\frac{1}{\mathcal{P}_G}\frac{\partial^2}{\partial \mu^2}\int_{a-\mu}^{b-\mu}\phi(x|0,\sigma^2)dx = \frac{1}{\mathcal{P}_G}\int_a^b \Big(\frac{x-\mu}{\sigma^2}\Big)^2\phi(x|\mu,\sigma^2)dx - \frac{1}{\sigma^2}. \tag{3.267}$$

It is easy to check that the LHS of (3.267) becomes

$$\frac{1}{\mathcal{P}_G}\frac{\partial^2}{\partial \mu^2}\int_{a-\mu}^{b-\mu}\phi(x|0,\sigma^2)dx = \frac{1}{\mathcal{P}_G}\Big[\Big(\frac{a-\mu}{\sigma^2}\Big)\phi(a|\mu,\sigma^2) - \Big(\frac{b-\mu}{\sigma^2}\Big)\phi(b|\mu,\sigma^2)\Big], \tag{3.268}$$

and further

$$\begin{aligned}\frac{1}{\mathcal{P}_G}\int_a^b \Big(\frac{x-\mu}{\sigma^2}\Big)^2\phi(x|\mu,\sigma^2)dx =& \frac{1}{\sigma^2}E\Big[\Big(\frac{X-\mu}{\sigma}\Big)^2\Big|a<X<b\Big] \\ =& \frac{1}{\sigma^4}E(X^2|a<X<b) - 2\frac{\mu}{\sigma^4}E(X|a<X<b)\Big] + \frac{\mu^2}{\sigma^4}.\end{aligned} \tag{3.269}$$

Hence, substituting the RHS of (3.268) and (3.269) into (3.267) and rearranging, we obtain

$$\Omega^2 = \sigma^2\frac{(a-\mu)\phi(a|\mu,\sigma^2) - (b-\mu)\phi(b|\mu,\sigma^2)}{\Phi(b|\mu,\sigma^2) - \Phi(a|\mu,\sigma^2)} + \sigma^2 - \mu^2 + 2\mu\Omega.$$

In what follows `R` function `Ex` is given for computing the first and second moments of truncated univariate Gaussian distribution.

```
R> Ex <- function(mu, sigma, a, b)
+ {
+   pb <- pnorm(b, mu, sigma)
+   pa <- pnorm(a, mu, sigma)
+   da <- dnorm(a, mu, sigma)
+   db <- dnorm(b, mu, sigma)
+   Omega <- mu + sigma^2*( da - db )/( pb - pa )
+   P1 <- pnorm(b, mu, sigma) - pnorm(a, mu, sigma)
+   ex <- mu + sigma^2*( da - db )/( pb - pa )
+   Omega2 <- sigma^2*( (a-mu)*da - (b-mu)*db )/( pb - pa ) + sigma^2 -mu^2 + 2*mu*ex
+   out2 <- list("Ex" = Omega, "Exx" = Omega2)
+   return( out2 )
+ }
```

3.11 First moment of truncated multivariate Gaussian distribution

Recall that $\mathcal{N}_p(\boldsymbol{\mu}, \Sigma, \mathcal{R}^p)$ is a truncated Gaussian distribution truncated on region

$$\mathcal{R}^p = \{\boldsymbol{x} \in \mathbb{R}^p | \boldsymbol{a} \le \boldsymbol{x} \le \boldsymbol{b}\}, \tag{3.270}$$

where $\boldsymbol{a} = (a_1, \cdots, a_p)^\top$ and $\boldsymbol{b} = (b_1, \cdots, b_p)^\top$. Moreover, define

$$\boldsymbol{v}_i[x] = (v_1, \cdots, v_{i-1}, x, v_{i+1}, \cdots, v_p), \tag{3.271}$$

$$\boldsymbol{v}[-i] = (v_1, \cdots, v_{i-1}, v_{i+1}, \cdots, v_p), \tag{3.272}$$

in which $\boldsymbol{v} = \big(v_1, v_2, \cdots, v_p\big)^\top$ is a vector of real values. By definition, we can write

$$\begin{aligned}
\boldsymbol{\Omega} =& \frac{1}{\mathcal{P}_G} \int_{\boldsymbol{a}}^{\boldsymbol{b}} \boldsymbol{x} \times \boldsymbol{\phi}_p(\boldsymbol{x}|\boldsymbol{\mu}, \Sigma) d\boldsymbol{x} \\
=& \frac{1}{\mathbf{C}_G \mathcal{P}_G} \int_{\boldsymbol{a}}^{\boldsymbol{b}} \boldsymbol{x} \exp\Big\{-\frac{\delta(\boldsymbol{x}-\boldsymbol{\mu}, \Sigma)}{2}\Big\} d\boldsymbol{x} \\
=& \frac{1}{\mathbf{C}_G \mathcal{P}_G} \int_{\boldsymbol{a}}^{\boldsymbol{b}} (\boldsymbol{x}-\boldsymbol{\mu}) \exp\Big\{-\frac{\delta(\boldsymbol{x}-\boldsymbol{\mu}, \Sigma)}{2}\Big\} d\boldsymbol{x} \\
&+ \frac{\boldsymbol{\mu}}{\mathbf{C}_G \mathcal{P}_G} \int_{\boldsymbol{a}}^{\boldsymbol{b}} \exp\Big\{-\frac{\delta(\boldsymbol{x}-\boldsymbol{\mu}, \Sigma)}{2}\Big\} d\boldsymbol{x} \\
=& \frac{1}{\mathbf{C}_G \mathcal{P}_G} \int_{\boldsymbol{a}}^{\boldsymbol{b}} (\boldsymbol{x}-\boldsymbol{\mu}) \exp\Big\{-\frac{\delta(\boldsymbol{x}-\boldsymbol{\mu}, \Sigma)}{2}\Big\} d\boldsymbol{x} + \boldsymbol{\mu},
\end{aligned} \tag{3.273}$$

where

$$\mathbf{C}_G = (2\pi)^{\frac{p}{2}} |\Sigma|^{\frac{1}{2}}, \tag{3.274}$$

$$\mathcal{P}_G = P(\boldsymbol{X} \in \mathcal{R}^p) = \boldsymbol{\Phi}_p(\mathcal{R}^p|\boldsymbol{\mu}, \Sigma) = \frac{1}{\mathbf{C}_G} \int_{\boldsymbol{a}}^{\boldsymbol{b}} \exp\Big\{-\frac{\delta(\boldsymbol{x}-\boldsymbol{\mu}, \Sigma)}{2}\Big\} d\boldsymbol{x}, \tag{3.275}$$

and $\delta(\cdot, \cdot)$ is the Mahalanobis distance defined as (3.29). Taking into account the fact that

$$\frac{\partial}{\partial \boldsymbol{\mu}} \frac{\delta(\boldsymbol{x}-\boldsymbol{\mu}, \Sigma)}{2} = -\Sigma^{-1}(\boldsymbol{x}-\boldsymbol{\mu}),$$

then $\boldsymbol{\Omega}$ in RHS of (3.273) can be represented as

$$\begin{aligned}
\boldsymbol{\Omega} =& \frac{1}{\mathbf{C}_G \mathcal{P}_G} \int_{\boldsymbol{a}}^{\boldsymbol{b}} \Sigma \frac{\partial}{\partial \boldsymbol{\mu}} \exp\Big\{-\frac{\delta(\boldsymbol{x}-\boldsymbol{\mu}, \Sigma)}{2}\Big\} d\boldsymbol{x} + \boldsymbol{\mu} \\
=& \frac{\Sigma}{\mathbf{C}_G \mathcal{P}_G} \frac{\partial}{\partial \boldsymbol{\mu}} \int_{\boldsymbol{a}-\boldsymbol{\mu}}^{\boldsymbol{b}-\boldsymbol{\mu}} \exp\Big\{-\frac{\delta(\boldsymbol{u}, \Sigma)}{2}\Big\} d\boldsymbol{u} + \boldsymbol{\mu} \\
=& \Sigma \times \boldsymbol{I} + \boldsymbol{\mu},
\end{aligned}$$

where the ith element of vector $\boldsymbol{I} = \big(I_1, I_2, \cdots, I_p\big)^\top$, for $i = 1, \cdots, p$, becomes

$$I_i = \frac{1}{\mathbf{C}_G \mathcal{P}_G} \int_{\boldsymbol{a}[-i]-\boldsymbol{\mu}[-i]}^{\boldsymbol{b}[-i]-\boldsymbol{\mu}[-i]} \Big[\exp\Big\{-\frac{\delta\big(\boldsymbol{u}_i[a_i-\mu_i], \Sigma\big)}{2}\Big\} - \exp\Big\{-\frac{\delta\big(\boldsymbol{u}_i[b_i-\mu_i], \Sigma\big)}{2}\Big\}\Big] d\boldsymbol{u}[-i]. \tag{3.276}$$

We further note that the quadratic form $\delta\big(\boldsymbol{u}_i[x], \Sigma\big)$ in RHS of (3.276) can be represented as

$$\delta\big(\boldsymbol{u}_i[x], \Sigma\big) = x^2 \times \big(\Sigma_{i,i}\big)^{-1} + \delta\Big(\boldsymbol{u}[-i] - \boldsymbol{\xi}(x), \Delta_{ii}\Big), \tag{3.277}$$

where

$$\boldsymbol{\xi}(x) = \Sigma_{i,-i} \times \big(\Sigma_{i,i}\big)^{-1} \times x, \tag{3.278}$$

$$\Delta_{ii} = \Sigma_{-i,-i} - \big(\Sigma_{i,i}\big)^{-1} \times \big(\Sigma_{i,-i}\big)^\top \Sigma_{i,-i}, \tag{3.279}$$

in which $\Sigma_{i,j}$ denotes the (i,j)th element of Σ, $\Sigma_{-i,-j}$ denotes the matrix Σ when its ith row and jth column is eliminated, and $\Sigma_{i,-j}$ is the ith row of matrix Σ when its jth column is eliminated, for $i, j = 1, \cdots, p$. Hence, using expression (3.277), the integrand in RHS of (3.276) can be rewritten as

$$\begin{aligned}
I_i =& \frac{1}{\mathbf{C}_G \mathcal{P}_G} \exp\Big\{-\frac{(a_i-\mu_i)^2}{2\Sigma_{i,i}}\Big\} \int_{\boldsymbol{a}[-i]-\boldsymbol{\mu}[-i]}^{\boldsymbol{b}[-i]-\boldsymbol{\mu}[-i]} \exp\Big\{-\frac{1}{2}\delta\Big(\boldsymbol{u}[-i] - \boldsymbol{\xi}(a_i-\mu_i), \Delta_{ii}\Big)\Big\} d\boldsymbol{u}[-i] \\
&- \frac{1}{\mathbf{C}_G \mathcal{P}_G} \exp\Big\{-\frac{(b_i-\mu_i)^2}{2\Sigma_{i,i}}\Big\} \int_{\boldsymbol{a}[-i]-\boldsymbol{\mu}[-i]}^{\boldsymbol{b}[-i]-\boldsymbol{\mu}[-i]} \exp\Big\{-\frac{1}{2}\delta\Big(\boldsymbol{u}[-i] - \boldsymbol{\xi}(b_i-\mu_i), \Delta_{ii}\Big)\Big\} d\boldsymbol{u}[-i],
\end{aligned} \tag{3.280}$$

where quantity $\mathbf{C}_G$ in (3.274) can be rewritten as

$$\mathbf{C}_G = (2\pi)^{\frac{1}{2}} |\Sigma_{i,i}|^{\frac{1}{2}} (2\pi)^{\frac{p-1}{2}} |\Delta_{ii}|^{\frac{1}{2}}. \tag{3.281}$$

Once we have computed I_i in (3.280), for $i = 1, \cdots, p$, the first moment of truncated Gaussian distribution is then obtained by substituting the constructed vector $\boldsymbol{I}$ into the RHS of (3.273).

Here, we carry out a small simulation study for computing the first moment of vector $\boldsymbol{X}_{\mathcal{R}^2} \sim \mathcal{N}_2(\mathbf{0}, \Sigma, \mathcal{R}^2)$ under two scenarios. Under the first scenario, we assume that $\sigma_1^2 = 2, \sigma_2^2 = 1, \rho = 0.3535$, $\boldsymbol{a} = (-1, -1)^\top$, and $\boldsymbol{b} = (2, 2)^\top$. Under the second scenario it is assumed that $\sigma_1^2 = 2, \sigma_2^2 = 2, \rho = 0.25$, $\boldsymbol{a} = (0, 0)^\top$, and $\boldsymbol{b} = (+\infty, +\infty)^\top$. For both scenarios, we compute $\boldsymbol{\Omega}$ by generating samples of sizes $N = 2000, 5000$, and 20000 based on 5000 runs. The instrumental distribution is set to be $\mathcal{N}_2(\boldsymbol{\mu}, \Sigma)$. For computing $\boldsymbol{\Omega}$ through the $\mathcal{N}_2(\boldsymbol{\mu}, \Sigma)$, one can follow two ways. In the first way the quantity $\boldsymbol{\Omega}$ is estimated as

$$\hat{\boldsymbol{\Omega}} = \frac{1}{N} \sum_{i=1}^{N} |\boldsymbol{x}_i| \tag{3.282}$$

where $|\boldsymbol{x}_i| = \left(|x_{1i}|, |x_{2i}|\right)^\top$ where $\boldsymbol{x}_i$s come independently from $\mathcal{N}_2(\boldsymbol{\mu}, \Sigma)$, for sufficiently large N. The second method computes $\hat{\boldsymbol{\Omega}}$ based on sample of size N when the lower and upper truncation bounds are $\boldsymbol{a} = \mathbf{0}$ and $\boldsymbol{b} = +\boldsymbol{\infty}$, respectively. Details for implementing the second method are given as follows.

1. Read $\boldsymbol{a} = \mathbf{0}, \boldsymbol{b} = +\boldsymbol{\infty}, \boldsymbol{\mu}$, Σ, and N;
2. Set $j = 0$ and $\boldsymbol{y} = (0, 0)^\top$;
3. Simulate $\boldsymbol{x} = (x_1, x_2)^\top \sim \mathcal{N}_2(\mathbf{0}, \Sigma)$
4. If $a_1 \leq x_1 \leq b_1$ and $a_2 \leq x_2 \leq b_2$, then accept $\boldsymbol{x}$ as a generation from $\mathcal{N}_2(\mathbf{0}, \Sigma, \mathcal{R}^2)$, set $\boldsymbol{y} = \boldsymbol{y} + (x_1, x_2)^\top$ and $j = j + 1$;
5. If $j = N$, then go to the next step, otherwise return to step 2;
6. Compute an approximation of $\boldsymbol{\Omega} = (\Omega_1, \Omega_2)^\top$ as $\hat{\boldsymbol{\Omega}} = (y_1/N, y_2/N)^\top$;
7. End.

While the exact value of $\boldsymbol{\Omega}$ under both scenarios is computed using (3.273), Table 3.14 shows the results of simulation study for approximating $\boldsymbol{\Omega}$ under both scenarios through the instrumental $\mathcal{N}_2(\boldsymbol{\mu}, \Sigma)$. The PDF of bivariate Gaussian distribution and the corresponding truncated version under two scenarios mentioned above is depicted in Figure 3.10. The corresponding `R` function for computing $\boldsymbol{\Omega}$ is given by the following.

```
R> library(mvtnorm) # for computing multivariate Gaussian CDF
R> a <- c(-1, -1); b <- c(2, 2); Mu <- c(0, 0);
R> Sigma <- matrix( c(2, 0.5, 0.5, 2), nrow = length(Mu), ncol = length(Mu), byrow = TRUE)
R> EX <- function(Mu, Sigma, a, b)
+ {
+ if( any(a >= b) == TRUE )
+ stop(message("lower bound must be smaller than the upper bound."))
+ if( length(a) != length(Mu) | | length(b) != length(Mu) )
+ stop(message("truncation bounds and location parameter must be of the same size."))
+ p <- length(Mu)
+ threshold <- 6*sqrt( diag(Sigma) )
+ a0 <- a - Mu
+ b0 <- b - Mu
```

Table 3.14: Summary statistics for computing $\boldsymbol{\Omega}$ through the exact and importance sampling with $\mathcal{N}_2(\mathbf{0}, \Sigma)$ as the instrumental.

Scenario	$\boldsymbol{a}^\top$	$\boldsymbol{b}^\top$	N	$\boldsymbol{\Omega}$	$\hat{\boldsymbol{\Omega}}$	summary			
						bias	SE	min.	max.
1	(-1,-1)	(2,2)	2000	0.3691	$\hat{\Omega}_1$	-1.7e-04	0.0174	0.3021	0.4440
				0.2603	$\hat{\Omega}_2$	5.9e-05	0.0158	0.2072	0.3158
			5000	0.3691	$\hat{\Omega}_1$	7.5e-06	0.0113	0.3268	0.4071
				0.2603	$\hat{\Omega}_2$	-1.8e-04	0.0101	0.2255	0.2971
2	(0,0)	(∞, ∞)	2000	1.2150	$\hat{\Omega}_1$	-7.8e-05	0.0197	1.1287	1.2802
				1.2150	$\hat{\Omega}_2$	-8.3e-04	0.0196	1.1462	1.2827
			5000	1.2150	$\hat{\Omega}_1$	-7.4e-04	0.0124	1.1690	1.2603
				1.2150	$\hat{\Omega}_2$	-4.7e-04	0.0125	1.1701	1.2774

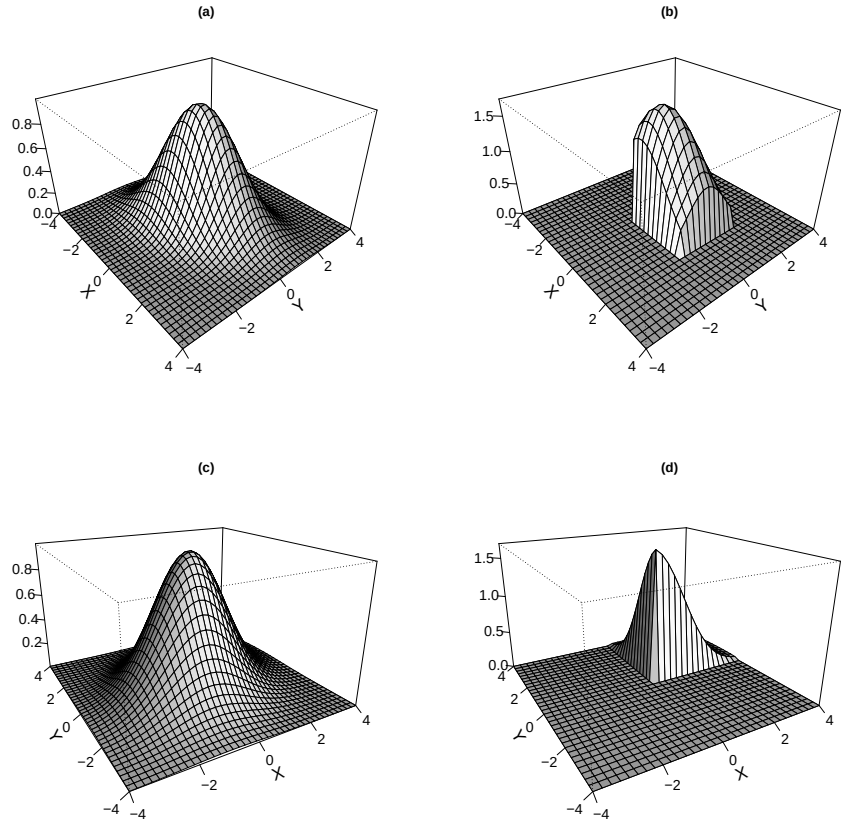


Figure 3.10: (a): The PDF of bivariate Gaussian distribution with parameters $\mu_1 = \mu_2 = 0$, $\sigma_1^2 = 1$, $\sigma_2^2 = 2$, and $\rho = 0.3535$. (b): The PDF of the corresponding truncated bivariate Gaussian distribution on area with lower limits $\boldsymbol{a} = (-1, -1)^\top$ and upper limits $\boldsymbol{b} = (2, 2)^\top$. (c) The PDF of bivariate Gaussian distribution with parameters $\mu_1 = \mu_2 = 0$, $\sigma_1^2 = 2$, $\sigma_2^2 = 2$, and $\rho = 0.25$. (b): The PDF of the corresponding truncated bivariate Gaussian distribution on area with lower limits $\boldsymbol{a} = (0, 0)^\top$ and upper limits $\boldsymbol{b} = (+\infty, +\infty)^\top$.

```
14 +   for(i in 1:p)
15 +   {
16 +     if( is.infinite( a0[i] ) ) a0[i] <- Mu[i] - threshold[i]
```

```
+     if( is.infinite( b0[i] ) ) b0[i] <- Mu[i] + threshold[i]
+   if( is.infinite( a[i] ) & a0[i] >= b0[i] ) a0[i] <- b0[i] - threshold[i]
+  if( is.infinite( b[i] ) & a0[i] >= b0[i] ) b0[i] <- a0[i] + threshold[i]
+   }
+ Omega <- numeric(p)
+ V <- numeric(p)
+ if(p == 2)
+ {
+   for(i in 1:p)
+   {
+     Mu_ia <- Sigma[i, -i]*a0[i]/Sigma[i, i]
+     Mu_ib <- Sigma[i, -i]*b0[i]/Sigma[i, i]
+     Delta_i <- sqrt( Sigma[-i, -i] - Sigma[i, -i]^2/Sigma[i, i] )
+     pai  <- pnorm(a0[-i], mean = Mu_ia, sd = Delta_i)
+     p_ai <- pnorm(a0[-i], mean = Mu_ib, sd = Delta_i)
+     pbi  <- pnorm(b0[-i], mean = Mu_ia, sd = Delta_i)
+     p_bi <- pnorm(b0[-i], mean = Mu_ib, sd = Delta_i)
+     V[i] <- dnorm(a0[i], 0, sd = sqrt(Sigma[i, i]) )*( pbi - pai   )-
+             dnorm(b0[i], 0, sd = sqrt(Sigma[i, i]) )*( p_bi - p_ai )
+   }
+ }else{
+  for(i in 1:p)
+   {
+     Mu_ia <- Sigma[i, -i]*a0[i]/Sigma[i, i]
+     Mu_ib <- Sigma[i, -i]*b0[i]/Sigma[i, i]
+     Delta_i <- Sigma[-i, -i] - c( Sigma[i, -i]%*%t(Sigma[i, -i])/Sigma[i, i] )
+      p1 <- pmvnorm(a0[-i], b0[-i], mean = Mu_ia, sigma = Delta_i)[1]
+      p2 <- pmvnorm(a0[-i], b0[-i], mean = Mu_ib, sigma = Delta_i)[1]
+     V[i] <- dnorm(a0[i], 0, sqrt(Sigma[i, i]) )*p1 -
+             dnorm(b0[i], 0, sqrt(Sigma[i, i]) )*p2
+   }
+ }
+ Omega <- Sigma%*%V/pmvnorm(a, b, mean = Mu, sigma = Sigma)[1] + Mu
+ return(Omega)
+ }
R> EX(Mu, Sigma, a, b)
```

3.12 Second moment of truncated Gaussian distribution

Let $\boldsymbol{v} = (v_1 \cdots, v_p)^\top$ represent a real-valued vector of length p. Based on vector $\boldsymbol{v}$, for $i < j$, we define the following notations.

$$\boldsymbol{v}_{i,j}[x, y] = (v_1, \cdots, v_{i-1}, x, v_{i+1}, \cdots, v_{j-1}, y, v_{j+1}, \cdots, v_p), \tag{3.283}$$

$$\boldsymbol{v}[-i, -j] = (v_1, \cdots, v_{i-1}, v_{i+1}, \cdots, v_{j-1}, v_{j+1}, \cdots, v_p), \tag{3.284}$$

$$\boldsymbol{v}[i, j] = (v_i, v_j). \tag{3.285}$$

Let $\boldsymbol{X} \sim \mathcal{N}_p(\boldsymbol{\mu}, \Sigma, \mathcal{R}^p)$. By definition, the second (raw) moment of truncated Gaussian distribution is given by

$$\boldsymbol{\Omega}^2 = E(\boldsymbol{X}\boldsymbol{X}^\top) = \frac{1}{\mathbf{C}_G \mathcal{P}_G} \int_{\boldsymbol{a}}^{\boldsymbol{b}} \boldsymbol{x}\boldsymbol{x}^\top \exp\Big\{-\frac{\delta(\boldsymbol{x} - \boldsymbol{\mu}, \Sigma)}{2}\Big\} d\boldsymbol{x},$$

where $\mathbf{C}_G$ and $\mathcal{P}_G$ are given by (3.274) and (3.275), respectively. More matrix algebra shows

$$\frac{\partial^2}{\partial \boldsymbol{\mu} \partial \boldsymbol{\mu}^\top} \exp\Big\{-\frac{\delta(\boldsymbol{x} - \boldsymbol{\mu}, \Sigma)}{2}\Big\} = \Big[\Sigma^{-1}(\boldsymbol{x} - \boldsymbol{\mu})(\boldsymbol{x} - \boldsymbol{\mu})^\top \Sigma^{-1} - \Sigma^{-1}\Big] \times \exp\Big\{-\frac{\delta(\boldsymbol{x} - \boldsymbol{\mu}, \Sigma)}{2}\Big\}. \tag{3.286}$$

Integrating both sides of (3.286), then multiplying from left and right by Σ, and finally rearranging, we have

$$\begin{aligned}\boldsymbol{\Omega}^2 =&\boldsymbol{\Omega}\boldsymbol{\mu}^\top + \boldsymbol{\mu}\boldsymbol{\Omega}^\top - \boldsymbol{\mu}\boldsymbol{\mu}^\top + \Sigma + \frac{\Sigma}{\mathbf{C}_G\mathcal{P}_G}\int_{\boldsymbol{a}}^{\boldsymbol{b}} \frac{\partial^2}{\partial\boldsymbol{\mu}\partial\boldsymbol{\mu}^\top}\exp\Big\{-\frac{\delta(\boldsymbol{x}-\boldsymbol{\mu},\Sigma)}{2}\Big\}d\boldsymbol{x}\Sigma\\ =&\boldsymbol{\Omega}\boldsymbol{\mu}^\top + \boldsymbol{\mu}\boldsymbol{\Omega}^\top - \boldsymbol{\mu}\boldsymbol{\mu}^\top + \Sigma + \frac{\Sigma}{\mathbf{C}_G\mathcal{P}_G}\frac{\partial^2}{\partial\boldsymbol{\mu}\partial\boldsymbol{\mu}^\top}\int_{\boldsymbol{a}-\boldsymbol{\mu}}^{\boldsymbol{b}-\boldsymbol{\mu}}\exp\Big\{-\frac{\delta(\boldsymbol{u},\Sigma)}{2}\Big\}d\boldsymbol{u}\Sigma\\ =&\boldsymbol{\Omega}\boldsymbol{\mu}^\top + \boldsymbol{\mu}\boldsymbol{\Omega}^\top - \boldsymbol{\mu}\boldsymbol{\mu}^\top + \Sigma + \Sigma\boldsymbol{I}\Sigma, \end{aligned} \tag{3.287}$$

where

$$\boldsymbol{I} = \frac{1}{\mathbf{C}_G\mathcal{P}_G}\frac{\partial^2}{\partial\boldsymbol{\mu}\partial\boldsymbol{\mu}^\top}\int_{\boldsymbol{a}-\boldsymbol{\mu}}^{\boldsymbol{b}-\boldsymbol{\mu}}\exp\Big\{-\frac{\delta(\boldsymbol{u},\Sigma)}{2}\Big\}d\boldsymbol{u}. \tag{3.288}$$

As it may seen, $\boldsymbol{I}$ is a $p\times p$ square matrix whose (i,j)th element is shown by $I_{i,j}$. In order to compute $I_{i,j}$, we consider three scenarios given by the following.

- **Scenario 1**: $i\neq j=1,\cdots,p$ and $p=2$
- **Scenario 2**: $i\neq j=1,\cdots,p$ and $p>2$
- **Scenario 3**: $i=j=1,\cdots,p$

In what follows, we proceed to compute $I_{i,j}$ under three scenarios mentioned above.

- **Scenario 1** ($i\neq j=1,\cdots,p$, and $p=2$): It follows from (3.288) that

$$I_{1,2} = \frac{1}{\mathbf{C}_G\mathcal{P}_G}\frac{\partial^2}{\partial\mu_1\partial\mu_2}\int_{a_1-\mu_1}^{b_1-\mu_1}\int_{a_2-\mu_2}^{b_2-\mu_2}\exp\Big\{-\frac{\delta(\boldsymbol{u},\Sigma)}{2}\Big\}du_1du_2. \tag{3.289}$$

  Applying the Leibniz integral rule for the RHS of (3.289), we have

$$I_{1,2} = \frac{1}{\mathcal{P}_G}\Big[\boldsymbol{\phi}_2(\boldsymbol{b}|\boldsymbol{\mu},\Sigma) - \boldsymbol{\phi}_2((b_1,a_2)^\top|\boldsymbol{\mu},\Sigma) + \boldsymbol{\phi}_2(\boldsymbol{a}|\boldsymbol{\mu},\Sigma) - \boldsymbol{\phi}_2((a_1,b_2)^\top|\boldsymbol{\mu},\Sigma)\Big].$$

  Obviously $I_{2,1}=I_{1,2}$.

- **Scenario 2** ($i\neq j=1,\cdots,p$, and $p>2$): For a given vector $(x,y)^\top$, it can be seen that the quadratic form $\delta(\boldsymbol{u}_{i,j}[x,y],\Sigma)$, for which $\boldsymbol{u}_{i,j}[x,y]$ is defined in (3.283), can be decomposed as

$$\delta(\boldsymbol{u}_{i,j}[x,y],\Sigma) = \delta_{ij}(x,y) + \delta\Big(\boldsymbol{u}[-i,-j] - \boldsymbol{\xi}(x,y),\Delta_{ij}\Big), \tag{3.290}$$

  where $\delta_{ij}(x,y) = (x,y)^\top\times\big(\Sigma_{\mathcal{I},\mathcal{I}}\big)^{-1}\times(x,y)$ for $\mathcal{I}=\{i,j\}$ and

$$\boldsymbol{\xi}(x,y) = \Sigma_{-\mathcal{I},\mathcal{I}}\times\big(\Sigma_{\mathcal{I},\mathcal{I}}\big)^{-1}\times(x,y), \tag{3.291}$$

$$\Delta_{ij} = \Sigma_{-\mathcal{I},-\mathcal{I}} - \big(\Sigma_{-\mathcal{I},\mathcal{I}}\big)\times\big(\Sigma_{\mathcal{I},\mathcal{I}}\big)^{-1}\times\big(\Sigma_{\mathcal{I},-\mathcal{I}}\big). \tag{3.292}$$

  Herein, $\boldsymbol{\xi}(x,y)$ is a vector of length $p-2$, Δ_{ij} is a $(p-2)\times(p-2)$ positive definite scale matrix, and 2×2 positive definite scale matrix matrix $\Sigma_{\mathcal{I},\mathcal{I}}$ is given by

$$\Sigma_{\mathcal{I},\mathcal{I}} = \begin{bmatrix}\Sigma_{i,i} & \Sigma_{i,j}\\ \Sigma_{j,i} & \Sigma_{j,j}\end{bmatrix}.$$

  For constructing $(p-2)\times 2$ matrix $\Sigma_{-\mathcal{I},\mathcal{I}}$, first construct a two-column matrix based on the ith and jth columns of matrix Σ and then remove ith and jth rows of the constructed

two-column matrix. We apply the Leibniz integral rule twice to the RHS of (3.288), and simultaneously utilize the property (3.290), to see that

$$\begin{aligned} I_{i,j} =& \frac{1}{\mathbf{C}_G \mathcal{P}_G} \exp\Big\{-\frac{\delta_{ij}(\boldsymbol{b}[i,j] - \boldsymbol{\mu}[i,j])}{2}\Big\} \times I_1(\boldsymbol{b}[i,j]) \\ & - \frac{1}{\mathbf{C}_G \mathcal{P}_G} \exp\Big\{-\frac{\delta_{ij}((\boldsymbol{a}[i], \boldsymbol{b}[j]) - \boldsymbol{\mu}[i,j])}{2}\Big\} \times I_1(\boldsymbol{a}[i], \boldsymbol{b}[j]) \\ & - \frac{1}{\mathbf{C}_G \mathcal{P}_G} \exp\Big\{-\frac{\delta_{ij}((\boldsymbol{b}[i], \boldsymbol{a}[j]) - \boldsymbol{\mu}[i,j])}{2}\Big\} \times I_1(\boldsymbol{b}[i], \boldsymbol{a}[j]) \\ & + \frac{1}{\mathbf{C}_G \mathcal{P}_G} \exp\Big\{-\frac{\delta_{ij}(\boldsymbol{a}[i,j] - \boldsymbol{\mu}[i,j])}{2}\Big\} \times I_1(\boldsymbol{a}[i,j]), \end{aligned} \tag{3.293}$$

where

$$I_1(\boldsymbol{z}) = \int_{\boldsymbol{a}[-i,-j]-\boldsymbol{\mu}[-i,-j]}^{\boldsymbol{b}[-i,-j]-\boldsymbol{\mu}[-i,-j]} \exp\Big\{-\frac{1}{2}\delta\Big(\boldsymbol{u}[-i,-j] - \boldsymbol{\xi}(\boldsymbol{z}), \Delta_{ij}\Big)\Big\} d\boldsymbol{u}[-i,-j]. \tag{3.294}$$

We further notice that the quantity $\mathbf{C}_G$ can be represented as

$$\mathbf{C}_G = (2\pi)^{\frac{p}{2}} |\Sigma|^{\frac{1}{2}} = 2\pi |\Sigma_{\mathcal{I},\mathcal{I}}|^{\frac{1}{2}} (2\pi)^{\frac{p-2}{2}} |\Delta_{ij}|^{\frac{1}{2}}. \tag{3.295}$$

Substituting (3.295) into the RHS of (3.293), it follows that

$$\begin{aligned} I_{i,j} =& \frac{1}{\mathcal{P}_G} \boldsymbol{\phi}_2\Big(\boldsymbol{b}[i,j] \Big| \boldsymbol{\mu}[i,j], \Sigma_{\mathcal{I},\mathcal{I}}\Big) \times I_1(\boldsymbol{b}[i,j]) \\ & - \frac{1}{\mathcal{P}_G} \boldsymbol{\phi}_2\Big((\boldsymbol{a}[i], \boldsymbol{b}[j]) \Big| \boldsymbol{\mu}[i,j], \Sigma_{\mathcal{I},\mathcal{I}}\Big) \times I_1(\boldsymbol{a}[i], \boldsymbol{b}[j]) \\ & - \frac{1}{\mathcal{P}_G} \boldsymbol{\phi}_2\Big((\boldsymbol{b}[i], \boldsymbol{a}[j]) \Big| \boldsymbol{\mu}[i,j], \Sigma_{\mathcal{I},\mathcal{I}}\Big) \times I_1(\boldsymbol{b}[i], \boldsymbol{a}[j]) \\ & + \frac{1}{\mathcal{P}_G} \boldsymbol{\phi}_2\Big(\boldsymbol{b}[i,j] \Big| \boldsymbol{\mu}[i,j], \Sigma_{\mathcal{I},\mathcal{I}}\Big) \times I_1(\boldsymbol{a}[i,j]). \end{aligned} \tag{3.296}$$

The matrix $\boldsymbol{I}$ is constructed by computing $I_{i,j}$ in (3.296) for all $i \neq j = 1, \cdots, p$. The second moment of truncated Gaussian distribution is then obtained by substituting $\boldsymbol{I}$ into the RHS of (3.287).

- **Scenario 3** ($i = j = 1, \cdots, p$): Using the Leibniz integral rule, the first order derivative of the RHS of (3.288) is

$$\begin{aligned} \frac{\partial}{\partial \mu_i} \boldsymbol{I} =& \frac{1}{\mathbf{C}_G \mathcal{P}_G} \frac{\partial}{\partial \mu_i} \int_{\boldsymbol{a}-\boldsymbol{\mu}}^{\boldsymbol{b}-\boldsymbol{\mu}} \exp\Big\{-\frac{\delta(\boldsymbol{u}, \Sigma)}{2}\Big\} d\boldsymbol{u} \\ =& \frac{1}{\mathbf{C}_G \mathcal{P}_G} \int_{\boldsymbol{a}[-i]-\boldsymbol{\mu}[-i]}^{\boldsymbol{b}[-i]-\boldsymbol{\mu}[-i]} \Big[\exp\Big\{-\frac{\delta(\boldsymbol{u}_i[a_i - \mu_i], \Sigma)}{2}\Big\} \\ & - \exp\Big\{-\frac{\delta(\boldsymbol{u}_i[b_i - \mu_i], \Sigma)}{2}\Big\}\Big] d\boldsymbol{u}[-i], \end{aligned} \tag{3.297}$$

for $i = 1, \cdots, p$. We note that the quadratic form $\delta(\boldsymbol{u}_i[x], \Sigma)$, for which $\boldsymbol{u}_i[x]$ is defined in (3.283), can be rewritten as

$$\delta(\boldsymbol{u}_i[x], \Sigma) = \delta_{ii}(x, x) + \delta\Big(\boldsymbol{u}[-i] - \boldsymbol{\xi}(x), \Delta_{ii}\Big), \tag{3.298}$$

where $\delta_{ii}(x, x) = (x_i - \mu_i)^2 \times (\Sigma_{i,i})^{-1}$ is a scalar and quantities $\boldsymbol{\xi}(\cdot)$ and Δ_{ii} are given in (3.278) and (3.279), respectively. We note that $\boldsymbol{\xi}(x)$ is a vector of length $p-1$ and Δ_{ii} is

a $(p-1)\times(p-1)$ positive definite scale matrix. We further notice that the quantity $\mathbf{C}_G$ is given by 3.281). Using information given in (3.298), (3.278), (3.279), and (3.274), the RHS of (3.297) can be represented as

$$\begin{aligned}\frac{\partial}{\partial\mu_i}\boldsymbol{I} =&\frac{1}{\mathcal{P}_G}\phi\Big(a_i\Big|\mu_i,\Sigma_{i,i}\Big)\times\boldsymbol{\Phi}_{p-1}\Big(\mathcal{R}^{p-1}\Big|\boldsymbol{\xi}(a_i-\mu_i),\Delta_{ii}\Big)\\ &-\frac{1}{\mathcal{P}_G}\phi\Big(b_i\Big|\mu_i,\Sigma_{i,i}\Big)\times\boldsymbol{\Phi}_{p-1}\Big(\mathcal{R}^{p-1}\Big|\boldsymbol{\xi}(b_i-\mu_i),\Delta_{ii}\Big)\\ =&I_{ia}-I_{ib},\end{aligned}\tag{3.299}$$

where

$$\mathcal{R}^{p-1}=\{\boldsymbol{x}[-i]\in\mathbb{R}^{p-1}|\boldsymbol{a}[-i]-\boldsymbol{\mu}[-i]\leq\boldsymbol{x}[-i]\leq\boldsymbol{b}[-i]-\boldsymbol{\mu}[-i]\}\tag{3.300}$$

Now, taking the second partial derivative with respect to μ_i from RHS of (3.299), we obtain

$$\begin{aligned}\frac{\partial I_{ia}}{\partial\mu_i} =&\frac{(a_i-\mu_i)}{\Sigma_{i,i}\times\mathcal{P}_G}\phi\Big(a_i\Big|\mu_i,\Sigma_{i,i}\Big)\times\boldsymbol{\Phi}_{p-1}\Big(\mathcal{R}^{p-1}\Big|\boldsymbol{\xi}(a_i-\mu_i),\Delta_{ii}\Big)\\ &-\frac{\phi(a_i|\mu_i,\Sigma_{i,i})}{\mathcal{P}_G}\times\frac{\Sigma_{-i,i}}{\Sigma_{i,i}}\times\int_{\boldsymbol{a}[-i]-\boldsymbol{\mu}[-i]}^{\boldsymbol{b}[-i]-\boldsymbol{\mu}[-i]}\Delta_{ii}^{-1}\Big[\boldsymbol{u}[-i]-\boldsymbol{\xi}(a_i-\mu_i)\Big]\\ &\times\exp\Big\{-\frac{\delta(\boldsymbol{u}[-i]-\boldsymbol{\xi}(a_i-\mu_i),\Delta_{ii})}{2}\Big\}d\boldsymbol{u}[-i].\end{aligned}\tag{3.301}$$

Clearly, the RHS of (3.301) can be represented in terms of the first moment of a truncated Gaussian distribution. That means

$$\begin{aligned}\frac{\partial I_{ia}}{\partial\mu_i} =&\frac{1}{\Sigma_{i,i}\times\mathcal{P}_G}\phi\Big(a_i\Big|\mu_i,\Sigma_{i,i}\Big)\times\boldsymbol{\Phi}_{p-1}\Big(\mathcal{R}^{p-1}\Big|\boldsymbol{\xi}(a_i-\mu_i),\Delta_{ii}\Big)\\ &\times\Big[a_i-\mu_i-\Sigma_{-i,i}(\Delta_{ii})^{-1}\boldsymbol{\Omega}_a\Big],\end{aligned}\tag{3.302}$$

where $\mathcal{R}^{p-1}$ is given by (3.300) and $\boldsymbol{\Omega}_a=E(\boldsymbol{Y}_a)$ where $\boldsymbol{Y}_a\sim\mathcal{N}_{p-1}(\mathbf{0},\Delta_{ii},\mathcal{R}_a^{p-1})$ in which

$$\mathcal{R}_a^{p-1}=\{\boldsymbol{y}\in\mathbb{R}^{p-1}|\boldsymbol{a}[-i]-\boldsymbol{\mu}[-i]-\boldsymbol{\xi}(a_i-\mu_i)\leq\boldsymbol{y}\leq\boldsymbol{b}[-i]-\boldsymbol{\mu}[-i]-\boldsymbol{\xi}(a_i-\mu_i)\}.$$

For computing $\partial I_{ib}/\partial\mu_i$, we likewise have

$$\begin{aligned}\frac{\partial I_{ib}}{\partial\mu_i} =&\frac{1}{\Sigma_{i,i}\times\mathcal{P}_G}\phi\Big(b_i\Big|\mu_i,\Sigma_{i,i}\Big)\times\boldsymbol{\Phi}_{p-1}\Big(\mathcal{R}^{p-1}\Big|\boldsymbol{\xi}(b_i-\mu_i),\Delta_{ii}\Big)\\ &\times\Big[b_i-\mu_i-\Sigma_{-i,i}(\Delta_{ii})^{-1}\boldsymbol{\Omega}_b\Big],\end{aligned}\tag{3.303}$$

where $\mathcal{R}^{p-1}$ is given by (3.300) and $\boldsymbol{\Omega}_b=E(\boldsymbol{Y}_b)$ where $\boldsymbol{Y}_b\sim\mathcal{N}_{p-1}(\mathbf{0},\Delta_{ii},\mathcal{R}_b^{p-1})$ in which

$$\mathcal{R}_b^{p-1}=\{\boldsymbol{y}\in\mathbb{R}^{p-1}|\boldsymbol{a}[-i]-\boldsymbol{\mu}[-i]-\boldsymbol{\xi}(b_i-\mu_i)\leq\boldsymbol{y}\leq\boldsymbol{b}[-i]-\boldsymbol{\mu}[-i]-\boldsymbol{\xi}(b_i-\mu_i)\}.$$

Once we have computed $\partial I_{ia}/\partial\mu_i$ in (3.302) and $\partial I_{ib}/\partial\mu_i$ in (3.303), for $i=1,\cdots,p$, then the diagonal elements of $\boldsymbol{I}$ given by (3.287) are available.

Once we have construct matrix $\boldsymbol{I}$, we can compute $\boldsymbol{\Omega}^2$ by substituting computed $\boldsymbol{I}$ at the RHS of (3.287).

Herein, we carry out a small simulation study for computing the first moment of vector $\boldsymbol{X}=(X_1,X_2)^\top$ under two scenarios given by the following.

i. we suppose that $\boldsymbol{X} \sim \mathcal{N}_2(\boldsymbol{\mu}, \Sigma, \mathcal{R}^2)$ in which $\boldsymbol{\mu} = \mathbf{0}$, $\Sigma = [(2, 0.5)^\top, (0.5, 1)^\top]$ truncated on region $\mathcal{R}^2$ whose lower and upper bounds are $\boldsymbol{a} = (-1, -1)^\top$ and $\boldsymbol{b} = (2, 2)^\top$, respectively.

ii. we suppose that $(X, Y)^\top$ follows truncated bivariate Gaussian distribution with parameters $\boldsymbol{\mu} = \mathbf{0}$, $\Sigma = [(2, 0.5)^\top, (0.5, 2)^\top]$ truncated on region $\mathcal{R}^2$ whose lower and upper bounds are $\boldsymbol{a} = (0, 0)^\top$ and $\boldsymbol{b} = (+\infty, +\infty)^\top$, respectively.

We note that the concern of the second scenario is placed on computing the absolute moment, that is $\boldsymbol{\Omega} = E(\boldsymbol{X}_{\mathcal{R}^2}) = E(|\boldsymbol{X}|)$. For both scenarios, we would compute $\boldsymbol{\Omega}$ by generating samples of sizes $N = 2000, 5000$, and 20000 based on 5000 runs. The instrumental distribution is set to be $\mathcal{N}_2(\boldsymbol{\mu}, \Sigma)$. To this end, one may follow on of two following ways. First, the quantity $\boldsymbol{\Omega}$ is estimated as

$$\hat{\boldsymbol{\Omega}} = \frac{1}{N} \sum_{i=1}^{N} |\boldsymbol{x}_i| \tag{3.304}$$

where $|\boldsymbol{x}_i| = \left(|x_{1i}|, |x_{2i}|\right)^\top$ where $\boldsymbol{x}_i$s come independently from $\mathcal{N}_2(\boldsymbol{\mu}, \Sigma)$, for sufficiently large N. Second, $\boldsymbol{\Omega}$ is estimated based on sample of size N when the lower and upper truncation bounds are $\boldsymbol{a} = \mathbf{0}$ and $\boldsymbol{b} = +\boldsymbol{\infty}$, respectively. Details for implementing the second method are given as follows. The corresponding `R` function `EXX` for computing $\boldsymbol{\Omega}$ is given by the following.

```
R> library(mvtnorm) # for computing multivariate Gaussian CDF
R> EXX <- function(Mu, Sigma, a, b)
+{
+ if(any(a >= b) == TRUE) stop(message("vector␣a␣must␣be␣smaller␣than␣vector␣b."))
+ p <- length(Mu)
+ PG <- pmvnorm(lower = a, upper = b, mean = Mu, sigma = Sigma)[1]
+ I0 <- matrix(0, nrow = p, ncol = p)
+ threshold <- 6*sqrt( diag(Sigma) )
+ a0 <- a - Mu
+ b0 <- b - Mu
+ for(i in 1:p)
+ {
+   if( is.infinite( a0[i] ) ) a0[i] <- Mu[i] - threshold[i]
+   if( is.infinite( b0[i] ) ) b0[i] <- Mu[i] + threshold[i]
+   if( is.infinite( a[i] ) & a0[i] >= b0[i] ) a0[i] <- b0[i] - threshold[i]
+  if( is.infinite( b[i] ) & a0[i] >= b0[i] ) b0[i] <- a0[i] + threshold[i]
+ }
+ if( p == 2 )
+ {
+  for(i in 1:p)
+  {
+   Eta_ia <- Sigma[i, -i]*a0[i]/Sigma[i, i]
+   Eta_ib <- Sigma[i, -i]*b0[i]/Sigma[i, i]
+   Delta_i <- Sigma[-i, -i] - Sigma[i, -i]*Sigma[i, -i]/Sigma[i, i]
+   La_i <- a0[-i] - Eta_ia
+   Ua_i <- b0[-i] - Eta_ia
+   Lb_i <- a0[-i] - Eta_ib
+   Ub_i <- b0[-i] - Eta_ib
+   PG_ia <- pnorm( Ua_i, mean = rep(0, p-1), sd = sqrt( Delta_i) )-
+  pnorm( La_i, mean = rep(0, p-1), sd = sqrt( Delta_i) )
+   PG_ib <- pnorm( Ub_i, mean = rep(0, p-1), sd = sqrt( Delta_i) )-
+  pnorm( Lb_i, mean = rep(0, p-1), sd = sqrt( Delta_i) )
+  Ex_ia <- Delta_i*( dnorm(La_i, 0, sqrt(Delta_i)) -
+        dnorm(Ua_i, 0, sqrt(Delta_i)) )/PG_ia
+  Ex_ib <- Delta_i*( dnorm(Lb_i, 0, sqrt(Delta_i)) -
+        dnorm(Ub_i, 0, sqrt(Delta_i)) )/PG_ib
+   p1 <- a0[i]*dnorm(a0[i],0,sqrt(Sigma[i, i]) )*PG_ia-
+   PG_ia*dnorm(a0[i], 0, sqrt(Sigma[i, i]) )*Sigma[i,-i]*Ex_ia/Delta_i
```

```
+   p2 <- b0[i]*dnorm(b0[i],0,sqrt(Sigma[i, i]))*PG_ib -
+ PG_ib*dnorm(b0[i], 0, sqrt(Sigma[i, i]) )*Sigma[i,-i]*Ex_ib/Delta_i
+ p1 <- ifelse( is.na(p1) | is.nan(p1) | is.infinite(p1), 0, p1)
+ p2 <- ifelse( is.na(p2) | is.nan(p2) | is.infinite(p2), 0, p2)
+ I0[i, i] <- (p1 - p2)/(PG*Sigma[i, i])
+ }
+   I0[1, 2] <-( dmvnorm(b0, mean = rep(0, p), sigma = Sigma )-
+   dmvnorm(c(a0[1], b0[2]), mean = rep(0, p), sigma = Sigma )-
+      dmvnorm(c(b0[1], a0[2]), mean = rep(0, p), sigma = Sigma )+
+    dmvnorm(a0, mean = rep(0, p), sigma = Sigma ) )/PG
+   I0[2, 1] <- I0[1, 2]
+ }
+ if( p == 3 )
+ {
+   for(i in 1:(p - 1))
+ {
+ for(j in (i + 1):p)
+ {
+    I_ij <- c(i, j)
+    Sigma_ij <- Sigma[I_ij, I_ij]
+    V0 <- c( Sigma[I_ij, -I_ij]%*%solve(Sigma_ij) )
+    Mu_aa <- (V0%*%c(a0[i], a0[j]))[1]
+    Mu_ab <- (V0%*%c(a0[i], b0[j]))[1]
+    Mu_ba <- (V0%*%c(b0[i], a0[j]))[1]
+    Mu_bb <- (V0%*%c(b0[i], b0[j]))[1]
+    Delta_ij <- Sigma[-I_ij, -I_ij] - ( V0%*%Sigma[-I_ij, I_ij] )[1]
+    p1 <- dmvnorm(c(b0[i], b0[j]), mean = rep(0, 2), sigma = Sigma_ij)*(
+ pnorm(b[-I_ij] - Mu[-I_ij], mean = Mu_bb, sd = sqrt( Delta_ij ) )-
+ pnorm(a[-I_ij] - Mu[-I_ij], mean = Mu_bb, sd = sqrt( Delta_ij ) ) )
+    p2 <- dmvnorm(c(a0[i], b0[j]), mean = rep(0, 2), sigma = Sigma_ij)*(
+    pnorm(b[-I_ij] - Mu[-I_ij], mean = Mu_ab, sd = sqrt( Delta_ij ) )-
+  pnorm(a[-I_ij] - Mu[-I_ij], mean = Mu_ab, sd = sqrt( Delta_ij ) ) )
+ p3 <- dmvnorm(c(b0[i], a0[j]), mean = rep(0, 2), sigma = Sigma_ij)*(
+  pnorm(b[-I_ij] - Mu[-I_ij], mean = Mu_ba, sd = sqrt( Delta_ij ) )-
+  pnorm(a[-I_ij] - Mu[-I_ij], mean = Mu_ba, sd = sqrt( Delta_ij ) ) )
+ p4 <- dmvnorm(c(a0[i], a0[j]), mean = rep(0, 2), sigma = Sigma_ij)*(
+  pnorm(b[-I_ij] - Mu[-I_ij], mean = Mu_aa, sd = sqrt( Delta_ij ) )-
+  pnorm(a[-I_ij] - Mu[-I_ij], mean = Mu_aa, sd = sqrt( Delta_ij ) ) )
+ I0[i, j] <- (p1 - p2 - p3 + p4)/PG
+ I0[j, i] <- I0[i, j]
+ }
+ }
+ for(i in 1:p)
+ {
+  Mu_ia <- Sigma[i, -i]*a0[i]/Sigma[i, i]
+  Mu_ib <- Sigma[i, -i]*b0[i]/Sigma[i, i]
+  Delta_i <- Sigma[-i, -i] - Sigma[i, -i]%*%t(Sigma[i, -i])/Sigma[i, i]
+  La_i <- a0[-i] - Mu_ia
+  Ua_i <- b0[-i] - Mu_ia
+  Lb_i <- a0[-i] - Mu_ib
+  Ub_i <- b0[-i] - Mu_ib
+ PG_ia <- pmvnorm(lower=La_i, upper=Ua_i, mean=rep(0, p-1), sigma=Delta_i)[1]
+ PG_ib <- pmvnorm(lower=Lb_i, upper=Ub_i, mean=rep(0, p-1), sigma=Delta_i)[1]
+ p5 <- a0[i]*dnorm(a0[i],0,sqrt(Sigma[i, i]) )*
+ pmvnorm(a0[-i], b0[-i], mean = Mu_ia, sigma = Delta_i)[1]-
+  PG_ia*dnorm(a0[i],0,sqrt(Sigma[i, i]) )*(t(Sigma[i, -i])%*%
+ solve(Delta_i)%*%EX(rep(0,p-1), Delta_i, La_i, Ua_i))[1]
+ p6 <- b0[i]*dnorm(b[i]-Mu[i],0,sqrt(Sigma[i, i]) )*
+ pmvnorm(a0[-i], b0[-i], mean = Mu_ib, sigma = Delta_i)[1]-
+  PG_ib*dnorm(b0[i],0, sqrt(Sigma[i, i]) )*(t(Sigma[i, -i])%*%
+ solve(Delta_i)%*%EX(rep(0,p-1), Delta_i, Lb_i, Ub_i))[1]
+ p5 <- ifelse( is.nan(p5) | is.na(p5), 0, p5 )
+ p6 <- ifelse( is.nan(p6) | is.na(p6), 0, p6 )
```

```
+ I0[i, i] <- (p5 - p6)/(PG*Sigma[i, i])
+ }
+ }
+ if ( p > 3 )
+ {
+  for(i in 1:(p - 1))
+ {
+   for(j in (i + 1):p)
+ {
+ I_ij <- c(i, j)
+ Sigma_ij <- Sigma[I_ij, I_ij]
+ V0 <- Sigma[-I_ij, I_ij]%*%solve(Sigma_ij)
+ Mu_aa <- c( V0%*%c(a0[i], a0[j]) )
+ Mu_ab <- c( V0%*%c(a0[i], b0[j]) )
+ Mu_ba <- c( V0%*%c(b0[i], a0[j]) )
+ Mu_bb <- c( V0%*%c(b0[i], b0[j]) )
+ Delta_ij <- Sigma[-I_ij, -I_ij] - V0%*%Sigma[I_ij, -I_ij]
+ p1 <- dmvnorm(c(b0[i], b0[j]), mean = rep(0, 2), sigma = Sigma_ij)*
+ pmvnorm(a0[-I_ij], b0[-I_ij], mean = Mu_bb, sigma = Delta_ij )
+ p2 <- dmvnorm(c(a0[i], b0[j]), mean = rep(0, 2), sigma = Sigma_ij)*
+ pmvnorm(a0[-I_ij], b0[-I_ij], mean = Mu_ab, sigma = Delta_ij )
+ p3 <- dmvnorm(c(b0[i], a0[j]), mean = rep(0, 2), sigma = Sigma_ij)*
+ pmvnorm(a0[-I_ij], b0[-I_ij], mean = Mu_ba, sigma = Delta_ij )
+ p4 <- dmvnorm(c(a0[i], a0[j]), mean = rep(0, 2), sigma = Sigma_ij)*
+ pmvnorm(a0[-I_ij], b0[-I_ij], mean = Mu_aa, sigma = Delta_ij )
+ I0[i, j] <- (p1[1] - p2[1] - p3[1] + p4[1])/PG
+ I0[j, i] <- I0[i, j]
+ }
+ }
+ for(i in 1:p)
+ {
+  Mu_ia <- Sigma[i, -i]*a0[i]/Sigma[i, i]
+  Mu_ib <- Sigma[i, -i]*b0[i]/Sigma[i, i]
+  Delta_i <- Sigma[-i, -i] - Sigma[i, -i]%*%t(Sigma[i, -i])/Sigma[i, i]
+  La_i <- a0[-i] - Mu_ia
+  Ua_i <- b0[-i] - Mu_ia
+  Lb_i <- a0[-i] - Mu_ib
+  Ub_i <- b0[-i] - Mu_ib
+ PG_ia <- pmvnorm(lower=La_i, upper=Ua_i, mean=rep(0, p-1), sigma=Delta_i)[1]
+ PG_ib <- pmvnorm(lower=Lb_i, upper=Ub_i, mean=rep(0, p-1), sigma=Delta_i)[1]
+ p51 <- a0[i]*dnorm(a0[i], 0, sqrt(Sigma[i, i]))*
+ pmvnorm(a0[-i], b0[-i], mean=Mu_ia, sigma=Delta_i)[1]
+ p52 <- PG_ia*dnorm(a0[i], 0, sqrt(Sigma[i, i]) )*( t(Sigma[i, -i])%*%
+ solve(Delta_i)%*%EX( rep(0, p - 1), Delta_i, La_i, Ua_i) )[1]
+ p61 <- b0[i]*dnorm(b0[i],0,sqrt(Sigma[i, i]) )*
+ pmvnorm(a0[-i], b0[-i], mean=Mu_ib, sigma=Delta_i)[1]
+ p62 <- PG_ib*dnorm(b0[i], 0, sqrt(Sigma[i, i]))*( t(Sigma[i, -i])%*%
+ solve(Delta_i)%*%EX(rep(0, p - 1), Delta_i, Lb_i, Ub_i) )[1]
+ p51 <- ifelse( is.nan(p51) | is.na(p51), 0, p51 )
+ p52 <- ifelse( is.nan(p52) | is.na(p52), 0, p52 )
+ p61 <- ifelse( is.nan(p61) | is.na(p61), 0, p61 )
+ p62 <- ifelse( is.nan(p62) | is.na(p62), 0, p62 )
+ I0[i, i] <- (p51 - p52 - p61 + p62)/(PG*Sigma[i, i])
+ }
+ }
+ Omega <- EX(Mu, Sigma, a, b)
+ EXX <- Omega%*%Mu + t(Omega%*%Mu) - Mu%*%t(Mu) + Sigma + Sigma%*%I0%*%Sigma
+ return(EXX)
+ }
```

3.13 Truncated skew Gaussian distribution

Here, we introduce the class of skew Gaussian distributions that is known in the literature as the canonical fundamental unrestricted skew Gaussian distribution, see Arellano-Valle and Azzalini [2006]. We write $\boldsymbol{X} \sim \mathrm{SG}_{p,q}(\boldsymbol{\mu}, \Sigma, \Lambda)$ to denote that p-dimensional random vector $\boldsymbol{X}$ follows a canonical fundamental unrestricted skew Gaussian distribution with PDF given by

$$f(\boldsymbol{x}|\boldsymbol{\mu}, \Sigma, \Lambda) = 2^q \boldsymbol{\phi}_p(\boldsymbol{x}|\boldsymbol{\mu}, \Omega)\boldsymbol{\Phi}_q(\boldsymbol{m}|\mathbf{0}_q, \Delta), \tag{3.305}$$

where $\Omega = \Sigma + \Lambda\Lambda^\top$, $\Delta = \boldsymbol{I}_q - \Lambda^\top\Omega^{-1}\Lambda$, $\boldsymbol{m} = \Lambda^\top\Omega^{-1}(\boldsymbol{y} - \boldsymbol{\mu})$, and $\boldsymbol{I}_q$ denotes the $q \times q$ identity matrix. Further, $\boldsymbol{\phi}_p(.|\boldsymbol{\mu}, \Omega)$ denotes the PDF of a p-dimensional Gaussian distribution with location vector $\boldsymbol{\mu}$ and dispersion matrix Σ, and $\boldsymbol{\Phi}_q(.|\mathbf{0}_q, \Delta)$ is the CDF of a q-dimensional Gaussian distribution with location vector $\mathbf{0}_q$ (a vector of zeros of length q) and dispersion matrix Δ. The random vector $\boldsymbol{X}$ admits stochastic representation as follows, see [Arellano-Valle and Genton, 2005, Arellano-Valle and Azzalini, 2006, Arellano-Valle et al., 2007].

$$\boldsymbol{X} \overset{d}{=} \boldsymbol{\mu} + \Lambda|\boldsymbol{Z}_0| + \Sigma^{\frac{1}{2}}\boldsymbol{Z}_1, \tag{3.306}$$

where $\overset{d}{=}$ means "distributed as" and random vectors $\boldsymbol{Z}_0 \sim \mathcal{N}_q(\mathbf{0}, \boldsymbol{I}_q)$ and $\boldsymbol{Z}_1 \sim \mathcal{N}_q(\mathbf{0}, \boldsymbol{I}_p)$ are independent. Furthermore, it can be seen that [Lee and McLachlan, 2016, Maleki et al., 2019, Morales et al., 2022]

$$\boldsymbol{X} \overset{d}{=} \boldsymbol{\mu} + (\boldsymbol{W}|\boldsymbol{W}_0 > \mathbf{0}), \tag{3.307}$$

in which

$$\boldsymbol{Y} = \begin{bmatrix}\boldsymbol{W}_0 \\ \boldsymbol{W}_1\end{bmatrix} \sim \mathcal{N}_{q+p}\left(\begin{bmatrix}\mathbf{0}_q \\ \mathbf{0}_p\end{bmatrix}, \begin{bmatrix}\boldsymbol{I}_q & \boldsymbol{\Lambda}^\top \\ \boldsymbol{\Lambda} & \Sigma + \boldsymbol{\Lambda}\boldsymbol{\Lambda}^\top\end{bmatrix}\right). \tag{3.308}$$

By the property (3.308), we can construct a method for simulating from

$$\boldsymbol{X}_{\mathcal{R}^p} \overset{d}{=} \boldsymbol{X}|\boldsymbol{a} \leq \boldsymbol{X} \leq \boldsymbol{b},$$

where $\boldsymbol{X}$ admits relation (3.306). To this end, we need to simulate from $(p+q)$-dimensional Gaussian random vector $\boldsymbol{Y}$ given by (3.308). Some algebra show

$$\boldsymbol{X}_{\mathcal{R}^p} \overset{d}{=} \boldsymbol{\mu}^* + \boldsymbol{Y}|(\boldsymbol{a}^* - \boldsymbol{\mu}^* \leq \boldsymbol{Y} \leq \boldsymbol{b}^* - \boldsymbol{\mu}^*),$$

where $a^* = (\mathbf{0}_q^\top, \boldsymbol{a}^\top)^\top$, $b^* = (\boldsymbol{\infty}_q^\top, \boldsymbol{b}^\top)^\top$, and $\boldsymbol{\mu}^* = (\mathbf{0}_q^\top, \boldsymbol{\mu}^\top)^\top$ in which $\boldsymbol{\infty}_q$ is a vector of length q of infinities. Hence, using the methodology given in Sections 3.11 and 3.12 for computing the first and second moments of multivariate Gaussian distribution, we can compute the first and second moments of truncated skew Gaussian distribution by computing $E(\boldsymbol{Y}|\boldsymbol{a}^* \leq \boldsymbol{Y} \leq \boldsymbol{b}^*)$ and $E(\boldsymbol{Y}\boldsymbol{Y}^\top|\boldsymbol{a}^* \leq \boldsymbol{Y} \leq \boldsymbol{b}^*)$, respectively. We note that an `R` package called `MomTrunc` uploaded at `https://cran.r-project.org/web/packages/MomTrunc/index.html` developed for this aim [Morales et al., 2022]. The following `R` code can be used for computing first two moments of the truncated random vector $\boldsymbol{X}$ where $\boldsymbol{X} \sim \mathrm{SG}_{2,2}(\boldsymbol{\mu}, \Sigma, \Lambda)$ truncated on $(\boldsymbol{a}, \boldsymbol{b})$ in which $\boldsymbol{\mu} = (0, 0)^\top$, $\boldsymbol{a} = (-3, -3)^\top$, $\boldsymbol{b} = (2, 2)^\top$,

$$\Sigma = \begin{bmatrix}2 & 0.5 \\ 0.5 & 2\end{bmatrix}, \Lambda = \begin{bmatrix}4 & 2 \\ 2 & -5\end{bmatrix}.$$

```r
library("MomTrunc")
a <- c(-3, -3); b <- c(2, 2); Mu <- c(0, 0); p <- 2; q<- 2
a_star <- c(0, 0, a); b_star <- c(Inf, Inf, b); Mu_star <- c(0, 0, Mu)
Sigma <- matrix( c(2, 0.5, 0.5, 2), nrow = p , ncol = p)
ch <- t(chol(Sigma)); Lambda <- matrix( c(4, 2 ,2, -5 ), nrow = p , ncol = q)
p1 <- diag(q); p2 <- t(Lambda); p3 <- t(p2); p4 <- Sigma + p3%*%t(p3)
Sigma_star <- matrix(
     c(p1[1,1],p1[1,2], p2[1,1], p2[1,2],
       p1[2,1],p1[2,2], p2[2,1], p2[2,2],
       p3[1,1],p3[1,2], p4[1,1], p4[1,2],
       p3[2,1],p3[2,2], p4[2,1], p4[2,2]), nrow = p + q, ncol = p + q)
out <- MCmeanvarTMD(a_star, b_star, rep(0, p + q), Sigma_star, dist = "normal")
out$mean[(p+1):(p+q)]
out$EYY[(p+1):(p+q), (p+1):(p+q)]
```

3.14 Moment of truncated Student's t distribution

Let random vector $\boldsymbol{X} = (X_1, \cdots, X_p)^\top$ follows a p-dimensional Student's t distribution with $\nu > 0$ degrees of freedom whose PDF given by (3.28). For simplicity we use generic symbols $\boldsymbol{t}_\nu(\cdot|\boldsymbol{\mu}, \Sigma)$ and $\boldsymbol{T}_\nu(\cdot|\boldsymbol{\mu}, \Sigma)$ to denote, accordingly, the PDF and CDF of a p-dimensional Student's t distribution. Recall from Definition 3.8.2 that the Student's t distribution is a Gaussian scale mixture model if $h(G) = \sqrt{G}$ where $G \sim \mathcal{G}(\nu/2, \nu/2)$. By definition

$$E(\boldsymbol{X}) = \int_{-\infty}^{\infty} \frac{\boldsymbol{x}\Gamma\left(\frac{\nu+p}{2}\right)}{(\nu\pi)^{\frac{p}{2}}\Gamma\left(\frac{\nu}{2}\right)|\Sigma|^{\frac{1}{2}}}\left[1 + \frac{\delta(\boldsymbol{x}-\boldsymbol{\mu}, \Sigma)}{\nu}\right]^{-\frac{\nu+p}{2}} d\boldsymbol{x}, \tag{3.309}$$

for $\nu > 1$. Suppose $\boldsymbol{t}_\nu(\boldsymbol{\mu}, \Sigma, \mathcal{R}^p)$ accounts for the family of the truncated Student's t distributions on region $\mathcal{R}^p$. The first moment of $\boldsymbol{t}_\nu(\boldsymbol{\mu}, \Sigma, \mathcal{R}^p)$ can be evaluated through relation (3.261) in which $f(\boldsymbol{u})$ is replaced with $\boldsymbol{t}_\nu(\boldsymbol{u}|\boldsymbol{\mu}, \Sigma)$ given in (3.28) and $P(\boldsymbol{a} \leq \boldsymbol{X} \leq \boldsymbol{b}) = \int_{\boldsymbol{a}}^{\boldsymbol{b}} \boldsymbol{t}_\nu(\boldsymbol{x}|\boldsymbol{\mu}, \Sigma)d\boldsymbol{x}$. However, herein, we proceed to compute $\boldsymbol{\Omega} = E(\boldsymbol{X}_{\mathcal{R}^p})$ based on the property of *Gaussian scale mixture model* introduced in Section 3.8.4.

3.14.1 First two moments of truncated Student's t distribution distribution in univariate case

Herein, we proceed to compute the first and second moments of an univariate Student's t distribution. The PDF and CDF of univariate Student's t distribution distribution are given by (3.30) and (3.31), respectively. |Furthermore, we write $X \sim t_\nu(\mu, \sigma)$ to show that random variable X follows a Student's t distribution with location parameter μ, scale parameter σ, and ν degrees of freedom. The truncated version on interval $\mathcal{R} = \{x \in \mathbb{R}|a \leq x \leq b\}$ is then represented by $t_\nu(\mu, \sigma, \mathcal{R})$.

If $X \sim t_\nu(\mu, \sigma, \mathcal{R})$, it can be seen from (3.259) for $r = 1$ that

$$\begin{aligned}\Omega =& \frac{1}{\mathcal{P}_t}\int_a^b x \times t(x|\mu, \sigma^2, \nu)dx,\\ =& \frac{1}{\mathcal{P}_t}\int_a^b \frac{x\Gamma\left(\frac{\nu+1}{2}\right)}{\sigma\sqrt{\nu\pi}\Gamma\left(\frac{\nu}{2}\right)}\left[1 + \frac{(x-\mu)^2}{\nu\sigma^2}\right]^{-\frac{\nu+1}{2}} dx,\end{aligned} \tag{3.310}$$

where $\mathcal{P}_t = T_\nu(b|\mu, \sigma) - T_\nu(a|\mu, \sigma)$. Rather computing Ω through (3.310), we prefer to use Definition 3.8.1 that states each Student's t distribution is a Gaussian scale mixture model. In

fact X admits the hierarchy given by

$$X|G \sim \mathcal{N}\Big(\mu, \frac{\sigma^2}{G}\Big),$$
$$G \sim \mathcal{G}\Big(\frac{\nu}{2}, \frac{\nu}{2}\Big).$$

Using above hierarchy, the quantity Ω can be computed as follows.

$$\begin{aligned}
\Omega &= \frac{1}{\mathcal{P}_t}\int_a^b x \int_0^\infty \phi\Big(x\Big|\boldsymbol{\mu}, \frac{\sigma^2}{g}\Big)\mathcal{G}\Big(g\Big|\frac{\nu}{2}, \frac{\nu}{2}\Big)dgdx, \\
&= \frac{1}{C_G\mathcal{P}_t}\int_0^\infty g^{\frac{1}{2}}\int_a^b x\exp\Big\{-\frac{g(x-\mu)^2}{2\sigma^2}\Big\}dx\mathcal{G}\Big(g\Big|\frac{\nu}{2}, \frac{\nu}{2}\Big)dg, \\
&= \frac{\sigma^2}{C_G\mathcal{P}_t}\int_0^\infty g^{-\frac{1}{2}}\int_a^b \frac{g(x-\mu)}{\sigma^2}\exp\Big\{-\frac{g(x-\mu)^2}{2\sigma^2}\Big\}dx\mathcal{G}\Big(g\Big|\frac{\nu}{2}, \frac{\nu}{2}\Big)dg + \mu,
\end{aligned}$$

where $C_G = \sqrt{2\pi}\sigma$. On the other hand, since

$$\frac{\partial}{\partial\mu}\exp\Big\{-\frac{g(x-\mu)^2}{2\sigma^2}\Big\} = g \times \Big(\frac{x-\mu}{\sigma^2}\Big)\exp\Big\{-\frac{g(x-\mu)^2}{2\sigma^2}\Big\}, \tag{3.311}$$

it turns out that

$$\begin{aligned}
\Omega =& \frac{\sigma^2}{C_G\mathcal{P}_t}\int_0^\infty g^{-\frac{1}{2}}\frac{\partial}{\partial\mu}\int_a^b \exp\Big\{-\frac{g(x-\mu)^2}{2\sigma^2}\Big\}dx\mathcal{G}\Big(g\Big|\frac{\nu}{2}, \frac{\nu}{2}\Big)dg + \mu, \\
=& \frac{\sigma^2}{C_G\mathcal{P}_t}\int_0^\infty g^{-\frac{1}{2}}\frac{\partial}{\partial\mu}\int_{a-\mu}^{b-\mu} \exp\Big\{-\frac{gu^2}{2\sigma^2}\Big\}du\mathcal{G}\Big(g\Big|\frac{\nu}{2}, \frac{\nu}{2}\Big)dg + \mu, \\
=& \frac{\sigma^2}{C_G\mathcal{P}_t}\int_0^\infty g^{-\frac{1}{2}}\Big[\exp\Big\{-\frac{g(a-\mu)^2}{2\sigma^2}\Big\} - \exp\Big\{-\frac{g(b-\mu)^2}{2\sigma^2}\Big\}\Big]\mathcal{G}\Big(g\Big|\frac{\nu}{2}, \frac{\nu}{2}\Big)dg + \mu, \\
=& \frac{\sigma^2}{C_G\mathcal{P}_t\Gamma\big(\frac{\nu}{2}\big)}\Big(\frac{\nu}{2}\Big)^{\frac{\nu}{2}}\int_0^\infty g^{\frac{\nu-1}{2}-1}\exp\Big\{-g\Big[\frac{(a-\mu)^2}{2\sigma^2} + \frac{\nu}{2}\Big]\Big\}dg \\
& - \frac{\sigma^2}{C_G\mathcal{P}_t\Gamma\big(\frac{\nu}{2}\big)}\Big(\frac{\nu}{2}\Big)^{\frac{\nu}{2}}\int_0^\infty g^{\frac{\nu-1}{2}-1}\exp\Big\{-g\Big[\frac{(b-\mu)^2}{2\sigma^2} + \frac{\nu}{2}\Big]\Big\}dg + \mu \\
=& \frac{\sigma^2\Gamma\big(\frac{\nu-1}{2}\big)}{C_G\mathcal{P}_t\Gamma\big(\frac{\nu}{2}\big)}\Big(\frac{\nu}{2}\Big)^{\frac{\nu}{2}}\Big[\Big[\frac{(a-\mu)^2}{2\sigma^2} + \frac{\nu}{2}\Big]^{-\frac{\nu-1}{2}} + \Big[\frac{(b-\mu)^2}{2\sigma^2} + \frac{\nu}{2}\Big]^{-\frac{\nu-1}{2}}\Big] + \mu.
\end{aligned} \tag{3.312}$$

Rearranging the RHS of (3.312) in terms of Student's t PDF, we obtain

$$\Omega = \frac{\sigma(\nu)}{\mathcal{P}_t}\Big[t_{\nu-2}\Big(\frac{a-\mu}{\sigma(\nu)}\Big|0,1\Big) - t_{\nu-2}\Big(\frac{b-\mu}{\sigma(\nu)}\Big|0,1\Big)\Big] + \mu, \tag{3.313}$$

where $\sigma(\nu) = \sigma\sqrt{\nu}/\sqrt{\nu-2}$ for $\nu > 2$. Applying the above argument to computing Ω, we have

$$\Omega^2 = \frac{1}{\mathcal{P}_t}\int_a^b x^2\int_0^\infty \phi\Big(x\Big|\boldsymbol{\mu}, \frac{\sigma^2}{g}\Big)\mathcal{G}\Big(g\Big|\frac{\nu}{2}, \frac{\nu}{2}\Big)dgdx.$$

For computing Ω^2, we take into account the fact that

$$\frac{\partial^2}{\partial\mu^2}\exp\Big\{-g\frac{(x-\mu)^2}{2\sigma^2}\Big\} = \Big[-\frac{g}{\sigma^2} + g^2 \times \Big(\frac{x-\mu}{\sigma^2}\Big)^2\Big]\exp\Big\{-g\frac{(x-\mu)^2}{2\sigma^2}\Big\}. \tag{3.314}$$

Rearranging equation (3.314) yields

$$\begin{aligned}
\frac{g^2x^2}{\sigma^4}\exp\Big\{-g\frac{(x-\mu)^2}{2\sigma^2}\Big\} =& \frac{\partial^2}{\partial\mu^2}\exp\Big\{-g\frac{(x-\mu)^2}{2\sigma^2}\Big\} \\
& + \Big[\Big(\frac{g}{\sigma^2} - \frac{\mu^2}{\sigma^4}g^2 + 2\frac{\mu}{\sigma^4}g^2x\Big)\exp\Big\{-g\frac{(x-\mu)^2}{2\sigma^2}\Big\}\Big].
\end{aligned} \tag{3.315}$$

Multiplying both sides of (3.315) by $\sigma^4 g^{-3/2}\mathcal{G}\big(g\big|\frac{\nu}{2},\frac{\nu}{2}\big)$, then integrating over x and g, and finally dividing by $C_G\mathcal{P}_t$, we have

$$\begin{aligned}\Omega^2 =&\frac{\sigma^4}{C_G\mathcal{P}_t}\int_0^\infty g^{-\frac{3}{2}}\frac{\partial^2}{\partial\mu^2}\int_a^b \exp\Big\{-g\frac{(x-\mu)^2}{2\sigma^2}\Big\}\mathcal{G}\Big(g\Big|\frac{\nu}{2},\frac{\nu}{2}\Big)dgdx\\ &+\Big[\frac{\sigma^2}{C_G\mathcal{P}_t}\mathcal{B}_{-\frac{1}{2},0}-\frac{\mu^2}{C_G\mathcal{P}_t}\mathcal{B}_{\frac{1}{2},0}+2\frac{\mu}{C_G\mathcal{P}_t}\mathcal{B}_{\frac{1}{2},1}\Big],\end{aligned}\tag{3.316}$$

where

$$\mathcal{B}_{i,j}=\int_a^b\int_0^\infty g^i x^j\exp\Big\{-g\frac{(x-\mu)^2}{2\sigma^2}\Big\}\mathcal{G}\Big(g\Big|\frac{\nu}{2},\frac{\nu}{2}\Big)dgdx.\tag{3.317}$$

For integral in the RHS of (3.316), it is not hard to check that

$$\begin{aligned}&\frac{\sigma^4}{C_G\mathcal{P}_t}\int_0^\infty g^{-\frac{3}{2}}\frac{\partial^2}{\partial\mu^2}\int_a^b \exp\Big\{-g\frac{(x-\mu)^2}{2\sigma^2}\Big\}\mathcal{G}\Big(g\Big|\frac{\nu}{2},\frac{\nu}{2}\Big)dgdx\\ &=\frac{\sigma(\nu)}{\mathcal{P}_t}(a-\mu)t_{\nu-2}\Big(\frac{a-\mu}{\sigma(\nu)}\Big|0,1\Big)-\frac{\sigma(\nu)}{\mathcal{P}_t}(b-\mu)t_{\nu-2}\Big(\frac{b-\mu}{\sigma(\nu)}\Big|0,1\Big),\end{aligned}\tag{3.318}$$

where $\sigma(\nu)=\sigma\sqrt{\nu}/\sqrt{\nu-2}$ for $\nu>2$. Hence, replacing the RHS of (3.318) with integral in RHS of (3.316), we have

$$\begin{aligned}\Omega^2 =&\frac{\sigma(\nu)}{\mathcal{P}_t}(a-\mu)t_{\nu-2}\Big(\frac{a-\mu}{\sigma(\nu)}\Big|0,1\Big)-\frac{\sigma(\nu)}{\mathcal{P}_t}(b-\mu)t_{\nu-2}\Big(\frac{b-\mu}{\sigma(\nu)}\Big|0,1\Big)\\ &+\frac{\sigma^2}{C_G\mathcal{P}_t}\mathcal{B}_{-\frac{1}{2},0}-\frac{\mu^2}{C_G\mathcal{P}_t}\mathcal{B}_{\frac{1}{2},0}+2\frac{\mu}{C_G\mathcal{P}_t}\mathcal{B}_{\frac{1}{2},1}.\end{aligned}\tag{3.319}$$

Moreover, it may seen, for $i>(1-\nu)/2$ and $j=\{0,1\}$, that

$$\mathcal{B}_{i,j}=\begin{cases}\dfrac{2^i\Gamma\big(\frac{\nu(i)}{2}\big)\sigma(i)\nu^{-i}}{\big[\pi\nu(i)\big]^{-\frac{1}{2}}\Gamma\big(\frac{\nu}{2}\big)}\Big[T_{\nu(i)}\Big(\dfrac{b-\mu}{\sigma(i)}\Big|0,1\Big)-T_{\nu(i)}\Big(\dfrac{a-\mu}{\sigma_i}\Big|0,1\Big)\Big], & \text{if } j=0,\\ \dfrac{2^i\Gamma\big(\frac{\nu(i)}{2}\big)\sigma(i)\nu^{-i}}{\big[\pi\nu(i)\big]^{-\frac{1}{2}}\Gamma\big(\frac{\nu}{2}\big)}\Big[T_{\nu(i)}\Big(\dfrac{b-\mu}{\sigma(i)}\Big|0,1\Big)-T_{\nu(i)}\Big(\dfrac{a-\mu}{\sigma_i}\Big|0,1\Big)\Big]\Omega_*^j, & \text{if } j>0,\end{cases}\tag{3.320}$$

where $\sigma(i)=\sigma\sqrt{\nu}/\sqrt{\nu+2i-1}$ (for $i>(1-\nu)/2$), $\nu(i)=\nu+2i-1$, and $\Omega_*^j=E(Y^j|a\le Y\le b)$ is the jth moment of $Y\sim t_{\nu(i)}(\mu,\sigma_i,\mathcal{R})$ in which $\mathcal{R}=\{x\in\mathbb{R}|a\le x\le b\}$. Computing the quantity $\mathcal{B}_{i,j}$ represented in (3.319) using (3.320), for $i=\{-1/2,1/2\}$ and $j=\{0,1\}$, and then replacing the results in the RHS of (3.316), one can compute the quantity Ω^2. The `R` function given by the following is developed for computing first two moments of the Student's t distribution truncated on $\mathcal{R}$.

```
R> Ex <- function(mu, sigma, nu, a, b)
+ {
+  CG <- sqrt(2*pi*sigma^2)
+  za <- (a - mu)/sigma; zb <- (b - mu)/sigma
+  Pt <- pt(zb, df = nu ) - pt(za, df = nu )
+    ex <- function(m, s, nu, a, b)
+    {
+     za <- (a - m)/s; zb <- (b - m)/s
+     Pt <- pt(zb, df = nu ) - pt(za, df = nu )
+     s_nu <- s*sqrt( nu/(nu - 2) )
+     za_nu <- (a - m)/s_nu
+     zb_nu <- (b - m)/s_nu
+     p1 <- s_nu/Pt*dt(za_nu, df = nu - 2)
+     p2 <- s_nu/Pt*dt(zb_nu, df = nu - 2)
```

```
+     return( list("p1" = p1, "p2" = p2, "Omega" = p1 - p2 + m) )
+     }
+   out1 <- ex(mu, sigma, nu, a, b)
+   p3 <- (a - mu)*out1$p1 - (b - mu)*out1$p2
+    Rij <- function(i, j)
+    {
+     nu_i <- nu + 2*i -1
+     sigma_i <- sigma*sqrt( nu/nu_i )
+     zb_i <- (b - mu)/sigma_i
+     za_i <- (a - mu)/sigma_i
+     C_ij <- (2/nu)^(i)*sqrt( pi*nu_i )*gamma(nu_i/2)/gamma(nu/2)*sigma_i
+     Pt_i <- pt(zb_i, df = nu_i ) - pt(za_i, df = nu_i )
+      if(j == 0)
+      {
+      R_ij <- C_ij*Pt_i
+      }else{
+      R_ij <- C_ij*Pt_i*ex(mu, sigma_i, nu_i, a, b)$Omega
+      }
+     R_ij
+     }
+ Omega2 <- p3 + sigma^2/(CG*Pt)*Rij(-1/2, 0) - mu^2/(CG*Pt)*Rij(1/2, 0) +
+ 2*mu/(CG*Pt)*Rij(1/2, 1)
+ out2 <- list("Ex" = out1$Omega, "Exx" = Omega2)
+ return( out2 )
+ }
```

3.15 First moment of truncated Student's t distribution in multivariate case

Recalling from Definition 3.8.2 that each Student's t distribution is a Gaussian scale mixture model. In fact if $\boldsymbol{X} \sim \boldsymbol{t}_\nu(\boldsymbol{\mu}, \Sigma)$, then it admits the hierarchy given by

$$\begin{aligned} \boldsymbol{X}|G &\sim \mathcal{N}_p\Big(\boldsymbol{\mu}, \frac{\Sigma}{G}\Big), \\ G &\sim \mathcal{G}\Big(\frac{\nu}{2}, \frac{\nu}{2}\Big). \end{aligned} \tag{3.321}$$

Using above hierarchy, the quantity $E(\boldsymbol{X})$ in (3.309) can be represented as

$$\begin{aligned} E(\boldsymbol{X}) &= \int_{-\infty}^{\infty} \boldsymbol{x} \int_0^{\infty} \boldsymbol{\phi}_p\Big(\boldsymbol{x}\Big|\boldsymbol{\mu}, \frac{\Sigma}{g}\Big)\mathcal{G}\Big(g\Big|\frac{\nu}{2}, \frac{\nu}{2}\Big)dgd\boldsymbol{x}. \\ &= \frac{1}{\mathbf{C}_G}\int_0^{\infty} g^{\frac{p}{2}} \int_{-\infty}^{\infty} \boldsymbol{x} \exp\Big\{-\frac{g}{2}\delta(\boldsymbol{x}-\boldsymbol{\mu}, \Sigma)\Big\}d\boldsymbol{x}\mathcal{G}\Big(g\Big|\frac{\nu}{2}, \frac{\nu}{2}\Big)dg, \end{aligned}$$

where $\mathbf{C}_G = (2\pi)^{p/2}|\Sigma|^{1/2}$. The first moment of truncated Student's t distribution $\boldsymbol{t}_\nu(\boldsymbol{\mu}, \Sigma, \mathcal{R}^p)$, is

$$\boldsymbol{\Omega} = E(\boldsymbol{X}_{\mathcal{R}^p}) = \frac{1}{\mathbf{C}_G\mathcal{P}_t}\int_0^{\infty}\int_{\boldsymbol{a}}^{\boldsymbol{b}} g^{\frac{p}{2}}\boldsymbol{x}\exp\Big\{-\frac{g}{2}\delta(\boldsymbol{x}-\boldsymbol{\mu}, \Sigma)\Big\}d\boldsymbol{x}\mathcal{G}\Big(g\Big|\frac{\nu}{2}, \frac{\nu}{2}\Big)dg,$$

where $\mathcal{P}_t = P(\boldsymbol{a} \leq \boldsymbol{X} \leq \boldsymbol{b})$. We can write

$$\begin{aligned} \boldsymbol{\Omega} =& \frac{1}{\mathbf{C}_G\mathcal{P}_t}\int_0^{\infty}\int_{\boldsymbol{a}}^{\boldsymbol{b}} g^{\frac{p}{2}}(\boldsymbol{x}-\boldsymbol{\mu})\exp\Big\{-\frac{g}{2}\delta(\boldsymbol{x}-\boldsymbol{\mu}, \Sigma)\Big\}d\boldsymbol{x}\mathcal{G}\Big(g\Big|\frac{\nu}{2}, \frac{\nu}{2}\Big)dg \\ &+ \frac{\boldsymbol{\mu}}{\mathbf{C}_G\mathcal{P}_t}\int_0^{\infty}\int_{\boldsymbol{a}}^{\boldsymbol{b}} g^{\frac{p}{2}}\exp\Big\{-\frac{g}{2}\delta(\boldsymbol{x}-\boldsymbol{\mu}, \Sigma)\Big\}d\boldsymbol{x}\mathcal{G}\Big(g\Big|\frac{\nu}{2}, \frac{\nu}{2}\Big)dg \\ =& \frac{1}{\mathbf{C}_G\mathcal{P}_t}\int_0^{\infty}\int_{\boldsymbol{a}}^{\boldsymbol{b}} g^{\frac{p}{2}}(\boldsymbol{x}-\boldsymbol{\mu})\exp\Big\{-\frac{g}{2}\delta(\boldsymbol{x}-\boldsymbol{\mu}, \Sigma)\Big\}d\boldsymbol{x}\mathcal{G}\Big(g\Big|\frac{\nu}{2}, \frac{\nu}{2}\Big)dg + \boldsymbol{\mu} \\ =& \boldsymbol{I} + \boldsymbol{\mu}. \end{aligned} \tag{3.322}$$

Since

$$\frac{\partial}{\partial \boldsymbol{\mu}} \frac{g}{2} \delta(\boldsymbol{x}-\boldsymbol{\mu}, \Sigma) = -g\Sigma^{-1}(\boldsymbol{x}-\boldsymbol{\mu}),$$

we have

$$\boldsymbol{I} = \frac{\Sigma}{\mathcal{P}_t} \int_0^\infty \frac{g^{\frac{p}{2}-1}}{\mathbf{C}_G} \frac{\partial}{\partial \boldsymbol{\mu}} \int_{\boldsymbol{a}}^{\boldsymbol{b}} \exp\Big\{-\frac{g}{2}\delta(\boldsymbol{x}-\boldsymbol{\mu}, \Sigma)\Big\} d\boldsymbol{x} \mathcal{G}\Big(g\Big|\frac{\nu}{2}, \frac{\nu}{2}\Big) dg.$$

Applying the change of variable $\boldsymbol{u} = \boldsymbol{x} - \boldsymbol{\mu}$, it follows that

$$\begin{aligned}\boldsymbol{I} =& \frac{\Sigma}{\mathcal{P}_t} \int_0^\infty \frac{g^{\frac{p}{2}-1}}{\mathbf{C}_G} \frac{\partial}{\partial \boldsymbol{\mu}} \int_{\boldsymbol{a}-\boldsymbol{\mu}}^{\boldsymbol{b}-\boldsymbol{\mu}} \exp\Big\{-\frac{g}{2}\delta(\boldsymbol{u}, \Sigma)\Big\} d\boldsymbol{u} \mathcal{G}\Big(g\Big|\frac{\nu}{2}, \frac{\nu}{2}\Big) dg \\ =& (I_1, I_2, \cdots, I_p)^\top,\end{aligned}$$

where

$$\begin{aligned}I_i =& \frac{\Sigma}{\mathcal{P}_t} \int_0^\infty \frac{g^{\frac{p}{2}-1}}{\mathbf{C}_G} \int_{\boldsymbol{a}[-i]-\boldsymbol{\mu}[-i]}^{\boldsymbol{b}[-i]-\boldsymbol{\mu}[-i]} \Bigg[\exp\Big\{-\frac{g\delta\big(\boldsymbol{u}_i[a_i-\mu_i], \Sigma\big)}{2}\Big\} \\ & - \exp\Big\{-\frac{g\delta\big(\boldsymbol{u}_i[b_i-\mu_i], \Sigma\big)}{2}\Big\}\Bigg] d\boldsymbol{u}[-i] \mathcal{G}\Big(g\Big|\frac{\nu}{2}, \frac{\nu}{2}\Big) dg, \end{aligned} \tag{3.323}$$

for $i = 1, \cdots, p$. Using identity (3.277) and the fact that

$$\frac{g^{\frac{p}{2}-1}}{\mathbf{C}_G} = \Big[\frac{1}{\sqrt{2\pi\Sigma_{i,i}}} \frac{1}{(2\pi)^{\frac{p-1}{2}} \big|\Delta_{ii}\big|^{\frac{1}{2}}}\Big] \times g^{\frac{p}{2}-1} = \frac{g^{-1}}{\sqrt{2\pi}} \Big[\frac{\Sigma_{i,i}}{g}\Big]^{-\frac{1}{2}} \frac{1}{(2\pi)^{\frac{p-1}{2}}} \Big|\frac{\Delta_{ii}}{g}\Big|^{-\frac{1}{2}},$$

the integrand in the RHS of (3.323) can be rewritten as

$$\begin{aligned}I_i =& \frac{\Sigma}{\mathcal{P}_t} \int_0^\infty \frac{g^{-1}}{\sqrt{2\pi}} \Big[\frac{\Sigma_{i,i}}{g}\Big]^{-\frac{1}{2}} \exp\Big\{-\frac{g(a_i-\mu_i)^2}{2\Sigma_{i,i}}\Big\} \mathcal{G}\Big(g\Big|\frac{\nu}{2}, \frac{\nu}{2}\Big) \\ & \times \int_{\boldsymbol{a}[-i]-\boldsymbol{\mu}[-i]}^{\boldsymbol{b}[-i]-\boldsymbol{\mu}[-i]} \frac{1}{(2\pi)^{\frac{p-1}{2}}} \Big|\frac{\Delta_{ii}}{g}\Big|^{-\frac{1}{2}} \exp\Big\{-\frac{g}{2}\delta\Big(\boldsymbol{u}[-i] - \boldsymbol{\xi}(a_i-\mu_i), \Delta_{ii}\Big)\Big\} d\boldsymbol{u}[-i] dg \\ & - \frac{\Sigma}{\mathcal{P}_t} \int_0^\infty \frac{g^{-1}}{\sqrt{2\pi}} \Big[\frac{\Sigma_{i,i}}{g}\Big]^{-\frac{1}{2}} \exp\Big\{-\frac{g(b_i-\mu_i)^2}{2\Sigma_{i,i}}\Big\} \mathcal{G}\Big(g\Big|\frac{\nu}{2}, \frac{\nu}{2}\Big) \\ & \times \int_{\boldsymbol{a}[-i]-\boldsymbol{\mu}[-i]}^{\boldsymbol{b}[-i]-\boldsymbol{\mu}[-i]} \frac{1}{(2\pi)^{\frac{p-1}{2}}} \Big|\frac{\Delta_{ii}}{g}\Big|^{-\frac{1}{2}} \exp\Big\{-\frac{g}{2}\delta\Big(\boldsymbol{u}[-i] - \boldsymbol{\xi}(b_i-\mu_i), \Delta_{ii}\Big)\Big\} d\boldsymbol{u}[-i] dg, \end{aligned} \tag{3.324}$$

where $\boldsymbol{\xi}(x)$ and Δ_{ii} are defined in (3.278) and (3.279), respectively. Moreover, it is not hard to check that

$$\begin{aligned}& \frac{g^{-1}}{\sqrt{2\pi}} \Big[\frac{\Sigma_{i,i}}{g}\Big]^{-\frac{1}{2}} \exp\Big\{-\frac{g(a_i-\mu_i)^2}{2\Sigma_{i,i}}\Big\} \mathcal{G}\Big(g\Big|\frac{\nu}{2}, \frac{\nu}{2}\Big) \\ =& \sqrt{\frac{\nu_2}{\Sigma_{i,i}}} t_{\nu-2}\Big(a_i\Big|\mu_i, \nu_2\Sigma_{i,i}\Big) \mathcal{G}\Big(g\Big|\frac{\nu-1}{2}, \lambda_a\Big), \end{aligned} \tag{3.325}$$

where $\nu_2 = \nu/(\nu-2)$, $\lambda_a = (a_i-\mu_i)^2/(2\Sigma_{i,i}) + \nu/2$. Further, for given vectors $\boldsymbol{c}_1$ and $\boldsymbol{c}_2$ of length $p-1$ such that $\boldsymbol{c}_1 < \boldsymbol{c}_2$, we have

$$\begin{aligned}& \int_0^\infty \int_{\boldsymbol{c}_1}^{\boldsymbol{c}_2} \frac{1}{(2\pi)^{\frac{p-1}{2}}} \Big|\frac{\Delta_{ii}}{g}\Big|^{-\frac{1}{2}} \exp\Big\{-\frac{g}{2}\delta(\boldsymbol{y}-\boldsymbol{m}, \Delta_{ii})\Big\} d\boldsymbol{y} \mathcal{G}\Big(g\Big|\frac{\nu-1}{2}, \lambda_a\Big) dg \\ =& \int_{\boldsymbol{c}_1}^{\boldsymbol{c}_2} \frac{\lambda_a^{\frac{\nu-1}{2}}}{(2\pi)^{\frac{p-1}{2}} \big|\Delta_{ii}\big|^{\frac{1}{2}}} \frac{\Gamma\big(\frac{\nu+p-2}{2}\big)}{\Gamma\big(\frac{\nu-1}{2}\big)} \Big[\frac{\delta(\boldsymbol{y}-\boldsymbol{m}, \Delta_{ii})}{2} + \lambda_a\Big]^{-\frac{\nu+p-2}{2}} d\boldsymbol{y} \\ =& \boldsymbol{T}_{\nu-1}\Big(\boldsymbol{c}_2\Big|\boldsymbol{m}, \frac{2\lambda_a}{\nu-1}\Delta_{ii}\Big) - \boldsymbol{T}_{\nu-1}\Big(\boldsymbol{c}_1\Big|\boldsymbol{m}, \frac{2\lambda_a}{\nu-1}\Delta_{ii}\Big), \end{aligned} \tag{3.326}$$

where $\nu > 1$. Using the facts given in (3.325) and (3.326), the RHS of (3.324) becomes

$$
\begin{aligned}
I_i = \sqrt{\frac{\nu_2}{\Sigma_{i,i}}}\frac{\Sigma}{\mathcal{P}_t} t_{\nu-2}\Big(a_i\Big|\mu_i, \nu_2\Sigma_{i,i}\Big)\Big[\boldsymbol{T}_{\nu-1}\Big(\boldsymbol{b}[-i]-\boldsymbol{\mu}[-i]\Big|\boldsymbol{\xi}(a_i-\mu_i), \frac{2\lambda_a}{\nu-1}\Delta_{ii}\Big) \\
- \boldsymbol{T}_{\nu-1}\Big(\boldsymbol{a}[-i]-\boldsymbol{\mu}[-i]\Big|\boldsymbol{\xi}(a_i-\mu_i), \frac{2\lambda_a}{\nu-1}\Delta_{ii}\Big)\Big] - \\
\sqrt{\frac{\nu_2}{\Sigma_{i,i}}}\frac{\Sigma}{\mathcal{P}_t} t_{\nu-2}\Big(b_i\Big|\mu_i, \nu_2\Sigma_{i,i}\Big)\Big[\boldsymbol{T}_{\nu-1}\Big(\boldsymbol{b}[-i]-\boldsymbol{\mu}[-i]\Big|\boldsymbol{\xi}(b_i-\mu_i), \frac{2\lambda_b}{\nu-1}\Delta_{ii}\Big) \\
- \boldsymbol{T}_{\nu-1}\Big(\boldsymbol{a}[-i]-\boldsymbol{\mu}[-i]\Big|\boldsymbol{\xi}(b_i-\mu_i), \frac{2\lambda_b}{\nu-1}\Delta_{ii}\Big)\Big], \qquad (3.327)
\end{aligned}
$$

where $\nu > 2$ and $\lambda_b = (b_i-\mu_i)^2/(2\Sigma_{i,i})+\nu/2$. Computing I_i in (3.327), for $i = 1, \cdots, p$, the first moment of truncated Student's t distribution is then obtained by substituting the constructed vector $\boldsymbol{I}$ into the RHS of (3.322). In what follows, we perform a small simulation study for computing $\boldsymbol{\Omega}$ when $\boldsymbol{X} = (X_1, X_2, X_3)^\top$ follows three-dimensional Student's t distribution with parameters $\nu = 3$, $\boldsymbol{\mu} = (1, 1, 1)^\top$ and

$$
\Sigma = \begin{bmatrix} 2 & 0.5 & 0.4 \\ 0.5 & 1.5 & 0.3 \\ 0.4 & 0.3 & 1 \end{bmatrix}. \qquad (3.328)
$$

The truncation bounds are $\boldsymbol{a} = (0, 0, 0)^\top$ and $\boldsymbol{b} = (10, 10, 10)^\top$. To this end, we compute $\boldsymbol{\Omega}$ using the exact method described above and the importance sampling in which the instrumental distribution is $\boldsymbol{t}_\nu(\boldsymbol{\mu}, \Sigma)$. For computing $\boldsymbol{\Omega}$ through the importance sampling, we generate a sample of size N through the rejection method. Table 3.15 shows the results of simulation study for approximating $\boldsymbol{\Omega}$ under both of exact and importance sampling scenarios.

Table 3.15: Summary statistics for computing $\boldsymbol{\Omega} = E(\boldsymbol{X}_{\mathcal{R}^p})$ through the exact and importance sampling methods.

						summary			
	$\boldsymbol{a}^\top$	$\boldsymbol{b}^\top$	N	$\boldsymbol{\Omega}$	$\hat{\boldsymbol{\Omega}}$	bias	SE	min.	max.
1	(-1,-1)	(2,2)	2000	0.3691	$\widehat{\Omega_1}$	-1.7e-04	0.0174	0.3021	0.4440
				0.2603	$\widehat{\Omega_2}$	5.9e-05	0.0158	0.2072	0.3158
			5000	0.3691	$\widehat{\Omega_1}$	7.5e-06	0.0113	0.3268	0.4071
				0.2603	$\widehat{\Omega_2}$	-1.8e-04	0.0101	0.2255	0.2971
2	(0,0)	(∞, ∞)	2000	1.2150	$\widehat{\Omega_1}$	-7.8e-05	0.0197	1.1287	1.2802
				1.2150	$\widehat{\Omega_2}$	-8.3e-04	0.0196	1.1462	1.2827
			5000	1.2150	$\widehat{\Omega_1}$	-7.4e-04	0.0124	1.1690	1.2603
				1.2150	$\widehat{\Omega_2}$	-4.7e-04	0.0125	1.1701	1.2774

```
R> library(MomTrunc) #for computing the CDF of multivariate Student's t
R> a <- rep(0, p); Mu <- rep(1, p); b <- rep(100, p); nu <- 3;
R> Sigma <- matrix( c(2,0.5,0.4,0.5,1.5,0.3,0.4,0.3,1), nrow = 3, ncol = 3)
R> EX <- function(Mu, Sigma, nu, a, b)
+ {
+ p <- length(Mu)
+ V <- numeric(p)
+ for(i in 1:p){
+ p1 <- (a[i] - Mu[i])/sqrt(nu/(nu-2)*Sigma[i, i])
+ p2 <- (b[i] - Mu[i])/sqrt(nu/(nu-2)*Sigma[i, i])
+ lambda_a <- (a[i] - Mu[i])^2/(2*Sigma[i, i])+ nu/2
+ lambda_b <- (b[i] - Mu[i])^2/(2*Sigma[i, i])+ nu/2
+ da <- sqrt(nu)/sqrt( (nu-2)*Sigma[i, i] )*dt(p1, df = nu - 2)
+ db <- sqrt(nu)/sqrt( (nu-2)*Sigma[i, i] )*dt(p2, df = nu - 2)
```

```
+ xi <- function(x) Sigma[i, -i]*x/Sigma[i, i]
+  if(p == 2)
+  {
+  Delta_i <- Sigma[-i, -i] - Sigma[-i, i]^2/Sigma[i, i]
+  }else{
+  Delta_i <- Sigma[-i, -i] - Sigma[-i, i]%*%t(Sigma[-i, i])/Sigma[i, i]
+  }
+ Delta_ia <- 2*lambda_a/(nu - 1)*Delta_i
+ Delta_ib <- 2*lambda_b/(nu - 1)*Delta_i
+ p3 <- da*pmvEST( a[-i] - Mu[-i], b[-i] - Mu[-i], mu = xi(a[i] - Mu[i]),
+       Sigma = Delta_ia, lambda = rep(0, p-1), tau = 0, nu = nu - 1 )
+ p4 <- db*pmvEST( a[-i] - Mu[-i], b[-i] - Mu[-i], mu = xi(b[i] - Mu[i]),
+       Sigma = Delta_ib, lambda = rep(0, p-1), tau = 0, nu = nu - 1 )
+ V[i] <- p3 - p4
+ }
+ EX <- Sigma%*%V/pmvEST(a, b, Mu, Sigma, rep(0, p), tau = 0, nu = nu) + Mu
+ return(EX)
+ }
```

4

Bayesian inference

A Bayesian prior is an assumption of an infinite amount of past relevant experience. But you cannot forget that you have just made up a whole bunch of data [1]

– Bradley Efron, Stanford University

In the framework of classical (or frequentist) statistics, the unknown parameter is assumed to be fixed and there is not any guess or belief about its true value. The only source to estimate the unknown parameter is the sample evidence (information). Contrary to this school of thinking, the Bayesian statistics permits to involve our *prior* knowledge or belief in process of estimating of unknown parameter. For this reason, the classical and Bayesian statistics are sometimes called *objective* and *subjective*[2] statistics, respectively. In other words, the subjectivity of belief about unknown parameter leads us to assume that unknown parameter is a random variable with a known distribution, called in the literature *prior* distribution. The Bayesian statistics would make a revision on the experimenter belief about the unknown parameter using the sample evidence. Let the PDF corresponds to the prior distribution of unknown parameter θ is shown by $\pi(\theta)$. Furthermore suppose the revised version of $\pi(\theta)$ given sample evidence $\boldsymbol{x} = \{x_1, x_2, \cdots, x_n\}$, is represented as $\pi(\theta|\boldsymbol{x})$ that is known in the literature as the *posterior* PDF. The elementary Bayesian formula links the *prior* with *posterior* as

$$\pi(\theta|\boldsymbol{x}) = \frac{f(\boldsymbol{x}|\theta)}{\int_\theta f(\boldsymbol{x}|\theta)\pi(\theta)d\theta}\pi(\theta) = \frac{f(\boldsymbol{x}|\theta)}{m(\boldsymbol{x})}\pi(\theta) \propto f(\boldsymbol{x}|\theta)\pi(\theta) = L(\theta|\boldsymbol{x})\pi(\theta), \tag{4.1}$$

where $L(\theta|\boldsymbol{x}) = f(\boldsymbol{x}|\theta) = f(x_1, x_2, \cdots, x_n|\theta) = \Pi_{i=1}^n f(x_i|\theta)$ is the likelihood function. Hence,

$$\pi(\theta|\boldsymbol{x}) \propto \Pi_{i=1}^n f(x_i|\theta)\pi(\theta), \tag{4.2}$$

Notice that joint PDF of sample evidence in (4.2), is represented as the product of marginal PDFs since the sample evidence $x_1, x_2, \cdots, x_n$ are assumed to be independent. Notice that in (5.30) the *prior* may be either *proper* or *improper*[3]. If the choosen *prior* $\pi(\theta)$ is *improper*, then the formal Bayes formula (5.30) is valid if we have $\pi(\theta|\boldsymbol{x})$ is *proper* or equivalently $m(\boldsymbol{x}) = \int_\theta f(\boldsymbol{x}|\theta)\pi(\theta)d\theta < \infty$. Furthermore, we note that the parameter space may be d-dimensional (multi-parameter case), represented as $\boldsymbol{\theta} = (\theta_1, \cdots, \theta_d)^\top$.

4.1 Prior selection

Although the prior reflects experimenter's belief or *prior* knowledge on θ, but the term *subjectivity* does not mean that experimenter is free to choose any candidate for distribution of *prior*.

[1] `https://rss.onlinelibrary.wiley.com/doi/pdf/10.1111/j.1740-9713.2010.00460.x`

[2] The belief about unknown parameter differs from individual to another one, so it is *subjective*.

[3] The *prior* $\pi(\theta)$ is called improper if $\int_\theta \pi(\theta)d\theta$ diverges; otherwise $\pi(\theta)$ is called *proper*.

In practice, there are a veriety of rules for this mean that makes the process of *prior* selection a quite *objective*[4] process that may need comprehensive study or knowledge about the unknown parameter. The structure of *prior* may also depend on some extra parameter(s) that is known in the literature as *hyperparameter*. In what follows, we cite some known classes of *objective prior*.

4.1.1 Uniform prior

The first class is the uniform prior $\pi_U(\theta)$, that is useful when the parameter space is compact interval and experimenter believes that all points in parameter space are equally likely outcomes. The famous example of this type may happen when one is interested in Bayesian inference for the success probability in a Bernoulli process for which we may have $\pi_U(\theta) \propto 1$. Considering a beta distribution for the law of *prior*, then $\pi_U(\theta|a, b)$, for which a and b play the role of *hyperparameters*, is no longer uniform. Doob [1949]

4.1.2 Reference prior

The first class of *priors* is known as the reference *prior* $\pi_R(\theta)$, proposed by Bernardo [1979]. For convenience let us to give some useful definitions as follows.

Definition 4.1.1. *Let $F(x|\boldsymbol{\theta})$ and $f(x|\boldsymbol{\theta})$ denote, respectively, the CDF and PDF of the random variable X with unknown parameter $\theta \in \boldsymbol{\theta}$.*

(i) **location family***: The family of random variable X is a location family if $F(x-\theta|\boldsymbol{\theta})$ or $f(x-\theta|\boldsymbol{\theta})$ for $\theta \in \mathbb{R}$ no longer depends on θ. Statistically speaking, distribution of transformation $X-\theta$ does not depend on θ. Herein, θ is known in the literature as the location parameter.*

(ii) **scale family***: The family of random variable X is a scale family if $F(x/\theta|\boldsymbol{\theta})$ or $1/\theta \times f(x/\theta|\boldsymbol{\theta})$ for $\theta \in \mathbb{R}^+$ no longer depends on θ. Statistically speaking, distribution of transformation X/θ does not depend on θ. Herein, θ is known in the literature as the scale parameter.*

(iii) **sufficient statistic** *Let $\boldsymbol{x} = \{x_1, x_2, \cdots, x_n\}$ denote a sequence of observations of random variables $X_1, X_2, \cdots, X_n$ that follow independently a family of distributions with PDF $f(x|\boldsymbol{\theta})$. A function of random variables $\boldsymbol{X} = \{X_1, X_2, \cdots, X_n\}$ such as $S(\boldsymbol{X})$ is called sufficient statistic if $S(\boldsymbol{X})$ provides information enough about unknown parameter $\theta \in \boldsymbol{\theta}$. Theoretically speaking, the conditional distribution of $\boldsymbol{X}$ given $S(\boldsymbol{X})$ does not depend on θ.*

The reference *prior* is obtained by maximizing the expected value of the Kullback-Leibler (or logarithmic) divergence between the *prior* and the posterior. The logarithmic divergence of PDF $\pi(\theta)$ from PDF $\tilde{\pi}(\theta)$ is represented as

$$\mathcal{K}(\pi||\tilde{\pi}) = \int_{\theta} \pi(\theta) \log \frac{\pi(\theta)}{\tilde{\pi}(\theta)} d\theta, \tag{4.3}$$

provided that integral (4.3) is finite. The reference *prior* must justify two convincing rationales including *permissibility* and *expected logarithmically convergence* [Berger et al., 2009, Definition 5]. There follows a listing of nice properties of the reference prior including:

(i) Consistency under reparameterization of the unknown parameter θ

[4]The term *objective* means that experimenter has some plausible guideline for selecting the distribution of *prior*.

(ii) Independence of sample size when observations are independent

(iii) Compatibility with sufficient statistic

(iiii) For a distribution for which nothing is known about unknown parameter θ, the reference prior follows the uniform distribution

For a proof of properties noted above, we refer the reader to Berger et al. [2009]. The property (i) states, for a location family, that the reference *prior* for location parameter is flat (equally likely over $\mathbb{R}$) and within a scale family, the reference *prior* of the scale parameter θ is proportional to $1/\theta$ that is the same of Jeffreys *prior*, that is $\pi_R(\theta) = \pi_J(\theta) \propto 1/\theta$ [Bernardo, 2005]. Although the reference prior is an objective prior and inherits nice properties among them some noted above, but it suffers from some drawbacks as follows.

(i) The reference prior must be computed point-by-point and then approximated through interpolation techniques.

(ii) The burden of computations grows when sample size increases and it may be critical if $f(x|\theta)$ lacks closed form such as α-stable distribution.

(iii) Although the conditions for existence of reference prior for multi-parameter case is guaranteed, but obviously demands for more computational complexity.

In general, a reference prior is obtained by following the steps of the algorithm given below [Berger et al., 2009].

Algorithm 14 Computing the reference prior

1: Choose a moderate integer value for k;
2: Choose an arbitrary positive function $\pi^*(\theta)$ such as $\pi^*(\theta) = 1$;
3: Consider the sequence $\theta_1, \cdots, \theta_m$ to be m arbitrary distinct points that θ can take on;
4: Set $j = 1$;
5: **while** $j \leq m$ **do**
6: Generate realizations $x_1, x_2, \cdots, x_k$ from PDF $f(\cdot|\theta_j)$;
7: Compute the integral $c_j = \int_{\mathbb{R}} \prod_{i=1}^{k} f(x_i|u)\pi^*(u)du$;
8: Compute $r_j = \log\Big(\prod_{i=1}^{k} f(x_j|\theta_j)\pi^*(\theta_j)/c_j\Big)$;
9: Compute $\pi(\theta_j) = \exp\{m^{-1}\sum_{j=1}^{m} r_j\}$ and store the pair $(\theta_j, \pi(\theta_j))$
10: Set $j \leftarrow j + 1$;
11: **end**
12: The set $\{(\theta_j, \pi(\theta_j))\}_{j=1}^{m}$ is the sequence of m paired from which $\pi(\theta)$ can be obtained through interpolation.

4.1.3 Jeffreys prior

The second class of objective *prior* is known as the Jeffreys *prior* $\pi_J(\boldsymbol{\theta})$ that was proposed by Jeffreys [1946]. We have

$$\pi_J(\boldsymbol{\theta}) \propto \sqrt{\det I(\boldsymbol{\theta})}, \tag{4.4}$$

where $I(\boldsymbol{\theta}) = -E\big[\partial^2/\partial\theta_i\partial\theta_j \log L(\boldsymbol{\theta}|\boldsymbol{x})\big]$, for $i, j = 1, \cdots, d$ is the Fisher information matrix.

4.1.4 Conjugate prior

The last class of priors is the conjugate prior $\pi_C(\theta)$. If prior and posterior follow the same family of distributions, then the corresponding *prior* is called a conjugate *prior*. For example,

let $x_1, x_2, \cdots, x_n$ independently follow $\mathcal{N}(\mu, \sigma^2)$. Considering a *prior* with PDF $\mathcal{N}(\cdot|\mu_0, \sigma_0^2)$, it is easy to see that *posterior* of μ is

$$\pi(\mu|\boldsymbol{x}) \sim \mathcal{N}\Big(\frac{\frac{\sigma^2}{n}\mu_0 + \sigma_0^2\bar{x}}{\frac{\sigma^2}{n} + \sigma_0^2}, \Big(\frac{n}{\sigma^2} + \frac{1}{\sigma_0^2}\Big)^{-1}\Big), \tag{4.5}$$

where $\bar{x}$ is the sample mean and $\boldsymbol{\theta}_0 = (\mu_0, \sigma_0)^\top$ is vector of *hyperparameters*. As it is seen from (4.5), the *posterior* follows a Gaussian distribution too. We note that the conjugate prior is an *subjective* prior since it is chosen by experimenter without rationale and just for the sake of simplicity in form and specially sampling from. The main advantage of the conjugate *prior* is that sampling from the corresponding posterior is an easy task.

If the PDF of family under study has not closed form, finding $\pi_J(\theta)$, $\pi_R(\theta)$, or even $\pi_C(\theta)$ may be difficult. As we will see in the next section, uniform and conjugate priors will be proposed for α and $\delta = \sigma^2$, respectively. These types were chosen by [Buckle, 1995, Lombardi, 2007, Godsill and Kuruoglu, 1999] for Bayesian estimation of tail thickness and scale parameters of α-*stable* distribution. However, some drawbacks of IG conjugate *prior* were addressed by [Gelman, 2006, Teimouri et al., 2024]. Indeed, investigating advantages and disadvantages of above-noted *priors* in more details needs more space that is out of scope of this work. We refer the reader to [Jeffreys, 1946, Jaynes, 1968, Gelman et al., 1995, Berger et al., 2009, Box and Tiao, 2011] for further specific information.

4.2 Empirical Bayes

As noted earlier, the prior itself may depend on hyperparameter(s). Hence, sometimes we use the notation $\pi(\theta|\boldsymbol{\theta}_0)$ rather than $\pi(\theta)$ to emphasize the fact that prior depends on hyperparameter $\boldsymbol{\theta}_0$. Evidently, for characterizing distribution of the unknown parameter θ through the posterior in (5.30), it may be seen

$$\pi(\theta|\boldsymbol{x}, \boldsymbol{\theta}_0) = \frac{f(\boldsymbol{x}|\theta)}{m(\boldsymbol{x}|\boldsymbol{\theta}_0)}\pi(\theta|\boldsymbol{\theta}_0), \tag{4.6}$$

where the unknown *hyperparameter* $\boldsymbol{\theta}_0$ plays the role of a *nuisance* parameter. The empirical Bayes is essentially an inferential approach, for estimating $\boldsymbol{\theta}_0$, that was introduced by Robbins [1956]. It takes two *non-parametric* and *parametric* types [Casella, 1985]. The empirical Bayes proposed by Robbins [1956] is a *non-parametric* approach under which experimenter has no any assumption about specification(s) of the prior's distribution while within a *parametric* framework the family of prior is fully specified by experimenter [Morris, 1983, Casella, 1985]. More precisely, the *parametric* empirical Bayes takes into account information provided by sample evidence $\boldsymbol{x}$ for estimating *hyperparameter* $\boldsymbol{\theta}_0$. Herein, we assume that $\boldsymbol{\theta}_0$ is fixed, but unknown. If $\boldsymbol{\theta}_0$ itself follows some distribution, then the procedure of estimating θ within Bayesian framework is called *hierarchical Bayesian inference*. For a single observation x_i, we recall from (4.6) that

$$\pi(\theta|x_i, \boldsymbol{\theta}_0) = \frac{f(x_i|\theta)}{m(x_i|\boldsymbol{\theta}_0)}\pi(\theta|\boldsymbol{\theta}_0). \tag{4.7}$$

If $\pi(\theta|x_i, \boldsymbol{\theta}_0)$ is proper, then $m(x_i|\boldsymbol{\theta}_0)$ in denominator of (4.7) itself is a PDF since $0 < m(x_i|\boldsymbol{\theta}_0) < \infty$ and $\int_{\mathbb{R}} m(x_i|\boldsymbol{\theta}_0)d\boldsymbol{x} = 1$. In fact, $m(x_i|\boldsymbol{\theta}_0)$ plays the role of *marginal* PDF (likelihood function) when $\boldsymbol{\theta}$ is integrated out. The unknown hyperparameter $\boldsymbol{\theta}_0$ is then estimated through the maximum likelihood (ML) by maximizing $\prod_{i=1}^n m(x_i|\boldsymbol{\theta}_0)$ with respect to $\boldsymbol{\theta}_0$ or using the method of moments [Teimouri et al., 2024]. In what follows, we proceed to compute the *parametric* empirical Bayes through an example.

Example 4-1

Let $x_1, x_2, \cdots, x_n$ independently follow $\mathcal{N}(\mu, \sigma^2)$ in which scale parameter $\sigma > 0$ is assumed to be known. Furthermore, a Gaussian prior $\pi(\mu) = \mathcal{N}(\mu|\mu_0, \sigma_0^2)$, is assumed for μ. For estimating $\boldsymbol{\theta}_0 = (\mu_0, \sigma_0)^\top$, we can write

$$m(x_i|\boldsymbol{\theta}_0) = \int_{-\infty}^{\infty} f(x_i|\mu)\pi(\mu|\boldsymbol{\theta}_0)d\mu = \int_{-\infty}^{\infty} f(x_i|\mu)\pi(\mu|\boldsymbol{\theta}_0)d\mu,$$

where $f(x_i|\mu) = \mathcal{N}(x_i|\mu, \sigma^2)$. We have

$$m(x_i|\boldsymbol{\theta}_0) = \int_{-\infty}^{\infty} \frac{1}{\sqrt{2\pi}\sigma} \exp\Big\{-\frac{(x_i-\mu)^2}{2\sigma^2}\Big\} \frac{1}{\sqrt{2\pi}\sigma_0} \exp\Big\{-\frac{(\mu-\mu_0)^2}{2\sigma_0^2}\Big\} d\mu.$$

Simplifying and some algebra leads to

$$m(x_i|\boldsymbol{\theta}_0) = \mathcal{N}(x_i|\mu_0, \sigma^2 + \sigma_0^2).$$

It is well known that the ML estimator of mean and variance of each Gaussian distribution are the sample counterparts. So, the empirical Bayesian estimator of μ_0 and σ_0 are $\widehat{\mu_0}_{EB} = \bar{x}$ and $\widehat{\sigma_0}_{EB} = \sqrt{|S^2 - \sigma^2|}$ where $\bar{x}$ and S^2 are the sample mean and variance, respectively. It is noteworthy that the quantity $S^2 - \sigma^2$ may be negative especially when sample size is small. Hence, the absolute sign is used to avoid negative value for $\widehat{\sigma_0}_{EB}$. ■

It is immediate from Example 4-1 above that the Bayesian estimator of μ, that is $\hat{\mu}_B$, becomes the average of the posterior (4.5). Hence, substituting $\widehat{\mu_0}_{EB}$ and $\widehat{\sigma_0}_{EB}$ obtained above in RHS of (4.5) yields $\hat{\mu}_B = \bar{x}$ with standard error

$$SE(\hat{\mu}_B) = \Big(\frac{n}{\sigma^2} + \frac{1}{|S^2 - \sigma^2|}\Big)^{-\frac{1}{2}}.$$

Evidently, $\hat{\mu}_B$ is more efficient than the ML counterpart $\hat{\mu}_{ML} = \bar{x}$ since

$$SE(\hat{\mu}_B) < SE(\hat{\mu}_{ML}) = \frac{\sigma}{\sqrt{n}}.$$

A different version of Example 4-1 in which x_is independently, for $i = 1, \cdots, n$, follow $\mathcal{N}(\mu_i, \sigma^2)$ is discussed by Casella [1985].

Example 4-2

Suppose $x_1, x_2, \cdots, x_n$ follow independently a Bernoulli distribution with success probability y, that is $X_i \sim \mathcal{BER}(y)$ for $i = 1, \cdots, n$ and $0 < y < 1$. For computing the Bayes estimation of y, that is $\hat{y}_B$, we may consider a Beta distribution with PDF $\pi(y|\boldsymbol{\theta}_0) = 1/\mathcal{B}(a,b)y^{a-1}(1-y)^{b-1}$ for the prior of y where $\boldsymbol{\theta}_0 = (a,b)^\top$ is the vector of hyperparameters. We can write

$$\begin{aligned}\pi(y|\boldsymbol{x}, \boldsymbol{\theta}_0) &= y^{\sum_{i=1}^n x_i}(1-y)^{n-\sum_{i=1}^n x_i}\pi(y|a,b) \\ &\propto y^{\sum_{i=1}^n x_i + a - 1}(1-y)^{n-\sum_{i=1} x_i + b - 1}, \end{aligned} \tag{4.8}$$

where $\boldsymbol{x} = \{x_1, x_2, \cdots, x_n\}$. Obviously, $y|\boldsymbol{x}$ follows a beta distribution. So, we can consider the mean of posterior, that is $(\sum_{i=1}^n x_i + a - 1)/(n + a + b - 2)$, as the Bayes estimator for the success probability y. Recalling from Subsection 4.2, for computing the vector of hyperparameters $\boldsymbol{\theta}_0 = (a,b)^\top$, we can write

$$\begin{aligned} m(x_i|\boldsymbol{\theta}_0) = \int_0^1 f(x_i|y)\pi(y|\boldsymbol{\theta}_0)dy &= \int_0^1 \frac{y^{x_i+a-1}(1-y)^{b-x_i}}{\mathcal{B}(a,b)}dy \\ &= \frac{\mathcal{B}(x_i + a, b - x_i + 1)}{\mathcal{B}(a,b)} \\ &= \frac{\Gamma(x_i+a)\Gamma(b-x_i+1)}{(a+b)\Gamma(a)\Gamma(b)}. \end{aligned}$$

Hence the likelihood function becomes

$$m(\boldsymbol{x}|\boldsymbol{\theta}_0) = \frac{\Pi_{i=1}^{n}\Gamma(x_i+a)\Gamma(b-x_i+1)}{(a+b)^n[\Gamma(a)\Gamma(b)]^n}. \tag{4.9}$$

Differentiating the RHS of (4.9) with respect to $\boldsymbol{\theta}_0$ and some algebra leads to the empirical Bayes estimator of $\boldsymbol{\theta}_0$ as $\widehat{\boldsymbol{\theta}_{0EB}} = (\sum_{i=1}^{n} x_i/n, (n-\sum_{i=1}^{n} x_i)/n)^\top$. ■

If in Example 4-2, the Bayesian estimator $\hat{y}_B$, of success probability y is desired, one could assume $\hat{y}_B$ is equal to the expected value of posterior as usual. To obtain $\hat{y}_B$, we recall from (4.8) that

$$\pi(y|\boldsymbol{x}) = \frac{y^{\sum_{i=1}^{n} x_i+a-1}(1-y)^{n-\sum_{i=1} x_i+b-1}}{\mathcal{B}(\sum_{i=1}^{n} x_i+a-1, n-\sum_{i=1} x_i+b-1)}. \tag{4.10}$$

It turns out that $y|\boldsymbol{x}$ follows a Beta distribution with parameters $\sum_{i=1}^{n} x_i+a$ and $n-\sum_{i=1} x_i+b$. Hence,

$$\hat{y}_B = E(y|\boldsymbol{x}) = \int_0^1 y\pi(y|\boldsymbol{x})dy = \frac{\sum_{i=1}^{n} x_i+a}{n+a+b}. \tag{4.11}$$

Substituting $\boldsymbol{\theta}_0$ computed above into the RHS of (4.11) yields

$$\hat{y}_B = \frac{\sum_{i=1}^{n} x_i + \frac{\sum_{i=1}^{n} x_i}{n}}{n+1}.$$

As it is see, the Bayesian estimator $\hat{y}_B$ is asymptotically (that is, $n \to \infty$) equal to the maximum likelihood (ML) estimator $\hat{y}_{ML} = \sum_{i=1}^{n} x_i/n$ of success probability y.

Example 4-3

Let $x_1, x_2, \cdots, x_n$ come independently from $\mathcal{G}(\alpha, \beta)$ with unknown α and known β. For computing the Bayesian estimator of α, we suppose that $\pi(\alpha|\boldsymbol{\theta}_0) = \mathcal{G}(\alpha|\alpha_0, \beta_0)$. But, first we need to determine hyperparameter $\boldsymbol{\theta}_0 = (\alpha_0, \beta_0)^\top$. To this end, we proceed through the empirical Bayes as follows.

$$\pi(\alpha|\boldsymbol{x}, \boldsymbol{\theta}_0) = \frac{f(\boldsymbol{x}|\alpha)\pi(\alpha|\boldsymbol{\theta}_0)}{m(\boldsymbol{x}|\boldsymbol{\theta}_0)},$$

where

$$\begin{aligned} f(\boldsymbol{x}|\alpha)\pi(\alpha|\boldsymbol{\theta}_0) =& \Big[\prod_{i=1}^{n} \frac{\beta^\alpha x_i^{\alpha-1}}{\Gamma(\alpha)} \exp\{-\beta x_i\}\Big] \frac{\beta_0^{\alpha_0}\alpha^{\alpha_0-1}}{\Gamma(\alpha_0)} \exp\{-\beta_0\alpha\}, \\ =& \frac{\beta_0^{\alpha_0}\beta^{n\alpha}\alpha^{\alpha_0-1}}{\Gamma(\alpha_0)\left[\Gamma(\alpha)\right]^n} \Big[\prod_{i=1}^{n} x_i\Big]^{\alpha-1} \exp\{-\beta\sum_{i=1}^{n} x_i - \beta_0\alpha\}, \end{aligned}$$

and

$$\begin{aligned} m(x_i|\boldsymbol{\theta}_0) &= \int_0^\infty f(x_i|\alpha)\pi(\alpha|\boldsymbol{\theta}_0)d\alpha \\ &= \int_0^\infty \beta^\alpha \frac{x_i^{\alpha-1}}{\Gamma(\alpha)} \exp\{-\beta x_i\}\beta_0^{\alpha_0} \frac{\alpha^{\alpha_0-1}}{\Gamma(\alpha_0)} \exp\{-\beta_0\alpha\}d\alpha. \end{aligned} \tag{4.12}$$

Obviously, the integral in the RHS of (4.1) has no closed form and so computing $\widehat{\boldsymbol{\theta}_{0EB}}$ using the ML approach is computationally cumbersome. Herein, we proceed to find the moment-based estimator of $\boldsymbol{\theta}_0$. First note that

$$\begin{aligned} E(X^s|\boldsymbol{\theta}_0) &= \int_0^\infty x^s \int_0^\infty f(x|\alpha)\pi(\alpha|\boldsymbol{\theta}_0)d\alpha dx \\ &= \int_0^\infty x^s \int_0^\infty \beta^\alpha \frac{x^{\alpha-1}}{\Gamma(\alpha)} \exp\{-\beta x\} \beta_0^{\alpha_0} \frac{\alpha^{\alpha_0-1}}{\Gamma(\alpha_0)} \exp\{-\beta_0\alpha\} d\alpha dx \\ &= \int_0^\infty \frac{\beta^\alpha \beta_0^{\alpha_0} \alpha^{\alpha_0-1} \exp\{-\beta_0\alpha\}}{\Gamma(\alpha)\Gamma(\alpha_0)} \int_0^\infty x^{s+\alpha-1} \exp\{-\beta x\} dx d\alpha \\ &= \int_0^\infty \frac{\beta^\alpha \beta_0^{\alpha_0} \alpha^{\alpha_0-1} \Gamma(s+\alpha) \exp\{-\beta_0\alpha\}}{\Gamma(\alpha)\Gamma(\alpha_0)\beta^{s+\alpha}} d\alpha, \end{aligned} \tag{4.13}$$

where $s \in \mathbb{N}$. It is easy to check that

$$E(X|\boldsymbol{\theta}_0) = \frac{\alpha_0}{\beta_0\beta}, \tag{4.14}$$

and

$$E(X^2|\boldsymbol{\theta}_0) = \frac{\alpha_0}{\beta_0\beta^2} + \frac{\alpha_0(\alpha_0+1)}{\beta_0^2\beta^2}. \tag{4.15}$$

Using (4.14) and (4.15), and some simplifications yields

$$\alpha_0 = \frac{\beta\big[E(X|\boldsymbol{\theta}_0)\big]^2}{\beta \text{var}(X|\boldsymbol{\theta}_0) - E(X|\boldsymbol{\theta}_0)}. \tag{4.16}$$

So, based on sample evidence $\{x_1, x_2, \cdots, x_n\}$, the moment-based estimator $\widehat{\alpha_{0EB}}$ is given by

$$\widehat{\alpha_{0EB}} = \frac{\beta\bar{x}^2}{|\beta S^2 - \bar{x}|}. \tag{4.17}$$

Consequently, the moment-based estimator $\widehat{\beta_{0EB}}$ is then given by

$$\widehat{\beta_{0EB}} = \frac{\widehat{\alpha_{0EB}}}{\beta\bar{x}}. \tag{4.18}$$

As it is seen, theoretically, denominator in the RHS of (4.16) is zero. This means that $\widehat{\alpha_{0EB}}$ must be large. Moreover, the absolute sign is used in denominator of (4.17) to avoid negative value for $\widehat{\alpha_{0EB}}$. Figure 4.1 shows the visual comparison between prior and posterior distributions when sample evidence are generated from $\mathcal{G}(\alpha = 2, \beta = 2)$ as underlying distribution for two data sets of sizes $n = 50$ and $n = 100$. ■

4.3 Credible interval

One common type of inference in frequentist statistics is *confidence interval*. The Bayesian analogue of confidence interval is known as *credible interval* and defined as follows.

Definition 4.3.1. *Given the observed sample evidence $\boldsymbol{x} = \{x_1, \cdots, x_n\}$, the interval (L, U) is a $100(1-\gamma)\%$ credible interval for unknown parameter θ if*

$$P(L < \theta < U|\boldsymbol{x}) = \int_L^U \pi(\theta|\boldsymbol{x})d\theta \geq 1 - \gamma, \tag{4.19}$$

where $0 < \gamma < 1$.

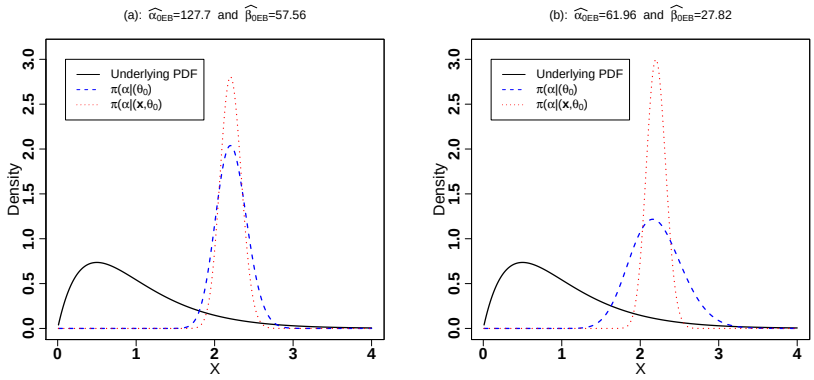


Figure 4.1: Schematic diagram for PDF of underlying, prior, and posterior when (a): $n = 50$ and (b): $n = 100$.

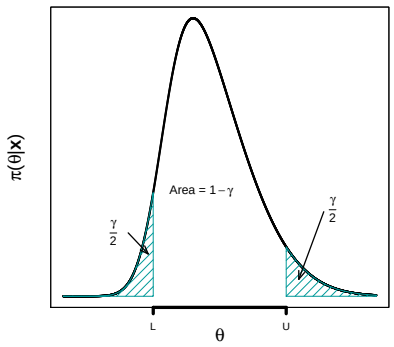


Figure 4.2: Schematic diagram for credible interval.

Indeed, Definition 4.2 states that there is at least $100(1-\gamma)\%$ confidence that interval (L, U) includes unknown parameter θ. Figure 4.2 provides a schematic diagram for credible interval.

Example 4-4

Let $x_1, x_2, \cdots, x_n$ come independently from $\mathcal{N}(\mu, \sigma^2)$ with known variance. By considering a conjugate prior $\mathcal{N}(\cdot|\mu_0, \sigma_0^2)$ for μ, we recall from (4.5) that $\mu|\boldsymbol{x}$ follows a Gaussian distribution. Hence

$$\sqrt{\frac{n}{\sigma^2}+\frac{1}{\sigma_0^2}}\left(\mu-\frac{n^{-1}\sigma^2\mu_0+\sigma_0^2\bar{x}}{n^{-1}\sigma^2+\sigma_0^2}\right)\sim\mathcal{N}(0,1). \tag{4.20}$$

Therefore, $100(1-\gamma)\%$ credible interval for μ is given by

$$\frac{n^{-1}\sigma^2\mu_0+\sigma_0^2\bar{x}}{n^{-1}\sigma^2+\sigma_0^2}\pm z_{\gamma/2}\sqrt{\frac{n}{\sigma^2}+\frac{1}{\sigma_0^2}}, \tag{4.21}$$

where $z_{\gamma/2}$ is the upper $\gamma/2$-quantile of the standard Gaussian distribution, that is, $P(Z > z_{\gamma/2}) = \gamma/2$ in which $Z \sim \mathcal{N}(0,1)$. ■

As it is seen from Example 4-4, for constructing credible interval, the upper and lower $\gamma/2$ of the posterior must be known. If the posterior gets complicated from the latter quantities cannot be found easily. In such cases, credible interval is constructed based on upper $\gamma/2$-quantile of

the samples generated from the posterior. Let $\pi_{\gamma/2}$ denote the posterior's upper $\gamma/2$-quantile, that is

$$\int_{\pi_{\gamma/2}}^{\infty} \pi(\theta|\boldsymbol{x})d\theta = \frac{\gamma}{2}. \tag{4.22}$$

Furthermore, suppose $\lceil u \rceil$ denotes the smallest integer value equal or greater than real value u and $\theta_{(r)}$ is rth order statistic of sample $\{\theta_1, \theta_2, \cdots, \theta_N\}$ generated from posterior with PDF $\pi(\theta|\boldsymbol{x})$. When $N \to \infty$, then strong law of large numbers states that $\pi_{\gamma/2} \approx \theta_{(v)}$ where $v = \lceil (1-\gamma/2)N \rceil$. The definition below gives the empirical credible interval.

Definition 4.3.2. *Let $\{\theta_1, \theta_2, \cdots, \theta_N\}$, for large N, denote generated samples form posterior with PDF $\pi(\theta|\boldsymbol{x})$. A $100(1-\gamma)\%$ empirical credible interval for unknown parameter θ is then give by*

$$(L_e, U_e) = (\theta_{(v)}, \theta_{(N-v)}), \tag{4.23}$$

where $v = \lceil (1-\gamma/2)N \rceil$.

Example 4-5 (continued)

Recalling from Example 4-3, suppose we observed $n = 200$ observations, represented as $\boldsymbol{x} = \{x_1, x_2, \cdots, x_{200}\}$ from $\mathcal{G}(\alpha = 2, \beta = 2)$. For known β, using the moment-based method described earlier in Example 4-3, we obtain $\boldsymbol{\theta}_0 = (\widehat{\alpha_0}_{EB}, \widehat{\beta_0}_{EB})^\top = (62.179, 30.083)^\top$. The corresponding posterior is given by

$$\begin{aligned}\pi(\alpha|\boldsymbol{x}, \boldsymbol{\theta}_0) &\propto \Big[\prod_{i=1}^{n} \frac{\beta^\alpha x_i^{\alpha-1}}{\Gamma(\alpha)} \exp\{-\beta x_i\}\Big] \frac{\beta_0^{\alpha_0} \alpha^{\alpha_0-1}}{\Gamma(\alpha_0)} \exp\{-\beta_0 \alpha\}, \\ &\propto \frac{\beta^{n\alpha} \alpha^{\alpha_0-1}}{[\Gamma(\alpha)]^n} \Big[\prod_{i=1}^{n} x_i\Big]^{\alpha-1} \exp\{-\beta_0 \alpha\}. \end{aligned} \tag{4.24}$$

In order to construct a 95% empirical credible interval for α, we need to sample from posterior (4.24) using methods such as MH or ARS. Taking into account into the fact that $\widehat{\alpha_0}_{EB} = 62.179$ and it is simple to see that the posterior (4.24) is log-concave for $\alpha_0 \geq 1$. So, we proceed for simulating from this posterior using the ARS approach. Based on a sample of size 5000 generated from posterior (4.24), we obtain quantities $L_e = 1.778$, $\hat{\alpha}_B = 1.945$, and $U_e = 2.121$. The scatterplot of generated samples as well as the latter quantities are shown in Figure 4.3. The R code for constructing the 95% empirical credible interval for α is given below.

```
R> f.a <- function(x, beta, alpha0, beta0, y)
+  {
+  n=length(y); n*x*log(beta)-n*lgamma(x)+(alpha0-1)*log(x)+
+   (x-1)*sum(y)-beta0*x
+  }
R> fprim.a <- function(x, beta, alpha0, beta0, y)
+  {
+    n=length(y); n*log(beta)-n*digamma(x)+(alpha0-1)/x+sum(y)-beta0
+  }
R> library(ars)
R> set.seed(20240608)
R> n <- 200; alpha <-2; beta <- 2
R> y <- rgamma(n, shape = alpha, rate = beta)
R> alpha0 <- beta*mean(y)^2/abs( beta*var(y) - mean(y) )
R> beta0 <- alpha0/(beta*mean(y))
R> N <- 5000 # number of generations
R> M <- 0 # size of burn-in period
R> theta <- rep(0, N)
```

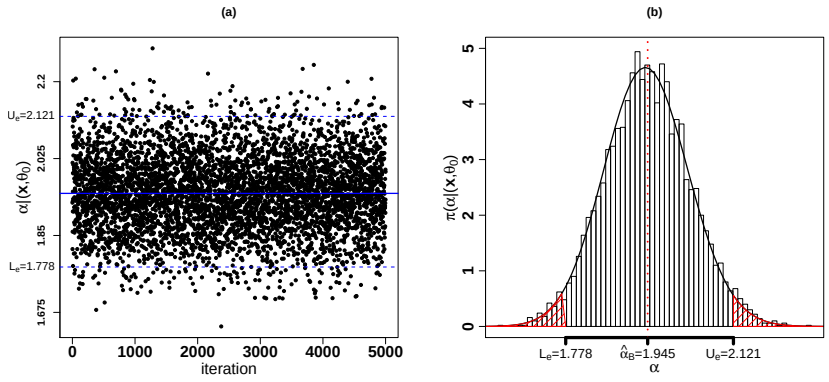


Figure 4.3: (a): the 95% empirical credible interval for parameter α based on a sample of $n = 200$ realizations generated from $\mathcal{G}(\alpha = 2, \beta = 2)$ distribution and (b): histogram of the generated samples with fitted posterior $\pi(\alpha|\boldsymbol{x}, \boldsymbol{\theta}_0)$.

```
R> for(j in 1:N)
+ {
+   theta[j] <- ars(1, f.a, fprim.a, x = c(0.1, 4, 10, 30), m = 4,
+    lb = TRUE, xlb = 0, ub = FALSE, beta = beta, alpha0 = alpha0,
+    beta0 = beta0, y = log(y) )
+  }
R> Le <- quantile(theta[(M+1):N], 0.025)[[1]]
R> Ue <- quantile(theta[(M+1):N], 0.975)[[1]]
R> alphahat_B <- mean(theta)
R> list("Lower_bound"=Le, "Bayes estimator"=alphahat_B, "Upper_bound"=Ue)
```

5

Gibbs sampling

"Statistics is the science of information gathering, especially when the information arrives in little pieces instead of big ones."

– Bradley Efron, Stanford University

This section has three parts. First, we describe the concept of Gibbs sampling as a general framework for generating independent samples from the given multivariate target distribution. Second, we discuss the situation that the Gibbs sampling is utilized to estimate the model's unknown parameter(s) through the Bayesian paradigm.

5.1 Gibbs sampling in broad sense

The Gibbs sampling (or sampler) is an iterative Markov chain Monte Carlo (MCMC) sampling approach was developed by Geman and Geman [1984] for image reconstruction, but its application in statistical analyses were shown by Gelfand and Smith [1990]. It is noteworthy that sampling from a p-dimensional distribution using the methods such as MH or AMH discussed earlier may involve undesired computational burden. In such cases, the Gibbs sampling technique that works efficiently is becoming increasingly popular in a wide range of applications. Assuming that we are at tth iteration of the Gibbs sampling for generating $\boldsymbol{x} = (x_1, \cdots, x_p)^\top$ from a p-dimensional target distribution with PDF $\pi(\cdot|\boldsymbol{\theta})$ [1]. For presentation purposes, we consider a set of generic symbols as follows.

$$\begin{aligned}
&\boldsymbol{x}^{(t)} = (x_1^{(t)}, x_2^{(t)}, \cdots, x_p^{(t)}),\\
&\boldsymbol{x}_i^{(t)}[y] = (x_1^{(t)}, \cdots, x_{i-1}^{(t)}, y, x_{i+1}^{(t)}, \cdots, x_p^{(t)}),\\
&\boldsymbol{x}^{(t)}[-i] = (x_1^{(t)}, \cdots, x_{i-1}^{(t)}, x_{i+1}^{(t)}, \cdots, x_p^{(t)}).
\end{aligned} \tag{5.1}$$

Indeed, each Gibbs sampling scheme works by generating sample from conditional distribution of vector $\boldsymbol{x} = (x_1, \cdots, x_p)^\top$ that is known in the literature as the *full conditional*. Hence, for generating $\boldsymbol{x}$ form target distribution with PDF $\pi(\cdot|\boldsymbol{\theta})$, we need to extract sample from p *full conditionals* each with PDF $\pi(x_i|\boldsymbol{x}[-i], \boldsymbol{\theta})$. It is proved [Geman and Geman, 1984] that, after *burn-in* period [Roberts and Smith, 1994], the generated sample comes from the target distribution whose PDF is $\pi(\cdot|\boldsymbol{\theta})$. The Gibbs sampling technique for simulating from a p-dimensional random vector with target distribution hose PDF is $\pi(\cdot|\boldsymbol{\theta})$ is summarized in Algorithm 15 algorithm.

Example 5-1

[1] Herein it is assumed that the family parameter $\boldsymbol{\theta}$ is a vector of known constants. If elements of $\boldsymbol{\theta}$ are not known, then they can be regarded as a variable of interest to sample from within the Gibbs sampling scheme.

Algorithm 15 Gibbs sampling for simulating from multivariate Gaussian distribution

Set $t = 0$, read M (*burn-in* period), N (total number of iterations), and $\boldsymbol{\theta}$;
Set $\boldsymbol{x}^{(t)} = \big(x_1^{(t)}, \cdots, x_p^{(t)}\big)^\top$ as the vector of initial sample;
while $t \leq N$ **do**
    Set $i = 1$;
    while $i \leq p$ **do**
        Generate sample y_i from full conditional with PDF $\pi(\cdot|\boldsymbol{x}^{(t)}[-i], \boldsymbol{\theta})$;
        Set $\boldsymbol{x}^{(t)}[i] \leftarrow y_i$;
        Set $i \leftarrow i + 1$;
    end
    Set $\boldsymbol{x}^{(t)} \leftarrow \boldsymbol{x}^{(t)}$;
    Set $t \leftarrow t + 1$;
end
Sequence $\big\{\boldsymbol{x}^{(M)}, \boldsymbol{x}^{(M+1)}, \cdots, \boldsymbol{x}^{(N)}\big\}$ is a sample of size $N - M + 1$ from target distribution with PDF $\pi(\cdot|\boldsymbol{\theta})$.

Let $\boldsymbol{X} = \big(X_1, X_2\big)^\top$ follows a bivariate Gaussian distribution with PDF

$$\pi(\boldsymbol{x}|\rho) = \frac{1}{2\pi\sqrt{1-\rho^2}} \exp\Big\{-\frac{x_1^2 + x_2^2 - 2\rho x_1 x_2}{2(1-\rho^2)}\Big\}. \tag{5.2}$$

where $\boldsymbol{x} = \big(x_1, x_2\big)^\top$ and $-1 < \rho < 1$. Since we have just two variables X_1 and X_2, so for simulating from $\boldsymbol{X}$ using the Gibbs sampling technique, we need to simulate from full conditionals $X_1|(X_2, \rho)$ and $X_2|(X_1, \rho)$. It can be easily seen that

$$\pi(x_1|x_2, \rho) \propto \exp\Big\{-\frac{x_1^2 - 2\rho x_1 x_2}{2(1-\rho^2)}\Big\}. \tag{5.3}$$

Since $x_1^2 - 2\rho x_1 x_2 = (x_1 - \rho x_2)^2 - (\rho x_2)^2$, then it follows that

$$\pi(x_1|x_2, \rho) \propto \exp\Big\{-\frac{(x_1 - \rho x_2)^2}{2(1-\rho^2)}\Big\}. \tag{5.4}$$

This means that $X_1|(X_2, \rho) \sim \mathcal{N}(\rho x_2, 1 - \rho^2)$. Likewise, we can write $X_2|(X_1, \rho) \sim \mathcal{N}(\rho x_1, 1 - \rho^2)$. Hence, based on the Algorithm 15, for simulating a sample of $n = 3000$ observations we may proceed as follows.

1. Set $t = 0$ and generate $\boldsymbol{x}^{(t)} = \big(x_1^{(t)}, x_2^{(t)}\big)^\top$ in which $x_1^{(t)} \sim \mathcal{N}(0, 1 - \rho^2)$ and $x_2^{(t)} \sim \mathcal{N}(0, 1 - \rho^2)$;

2. Simulate y_1 from the full conditional $\pi(x_1|x_2^{(t)}, \rho) \sim \mathcal{N}(\rho x_2^{(t)}, 1 - \rho^2)$;

3. Having $x_1^{(t)} = y_1$, simulate y_2 from the full conditional $\pi(x_2|y, \rho) \sim \mathcal{N}(\rho y_1, 1 - \rho^2)$;

4. $\boldsymbol{x}^{(t+1)} \leftarrow \big(y_1, y_2\big)^\top$ and $t \leftarrow t + 1$;

5. If $t = 5000$, then go to the next step, otherwise return to step 2;

6. Sequence $\big\{\boldsymbol{x}^{(2001)}, \boldsymbol{x}^{(2002)}, \cdots, \boldsymbol{x}^{(5000)}\big\}$ constitutes a sample of size 3000 from target distribution with PDF given in (5.2).

The associated `R` code for generating 3000 realizations from $\mathcal{N}_2(\mu_1 = 0, \mu_2 = 0, \sigma_1^2 = 1, \sigma_2^2 = 1, \rho = 0.50)$ is given as follows.

```
R > set.seed(20240520)
R > N <- 5000 # number of generations
R > M <- 2000 # size of burn-in period
R > rho <- 0.5
R > X <- matrix(0, nrow = N, ncol = 2)
R > sigma1 <- sigma2 <- 1
R > Sigma <- matrix( c(1, rho*sigma1*sigma2, rho*sigma1*sigma2, 1), 2, 2)
R > X[1, ] <- rnorm (2, mean = 0, sd = 1 - rho^2) # intitial generation
R > j <- 1
R > while(j < N)
+ {
+    X[j + 1, 1] <- rnorm (1, mean = rho*sqrt(Sigma[1,1]/Sigma[2,2])*X[j , 2],
+    sd = 1 - rho^2)
+    X[j + 1, 2] <- rnorm (1, mean = rho*sqrt(Sigma[2,2]/Sigma[1,1])*X[j + 1, 1],
+    sd= 1 - rho^2)
+    j <- j + 1
+   }
R > plot(X[(N-M+1):N,])
```

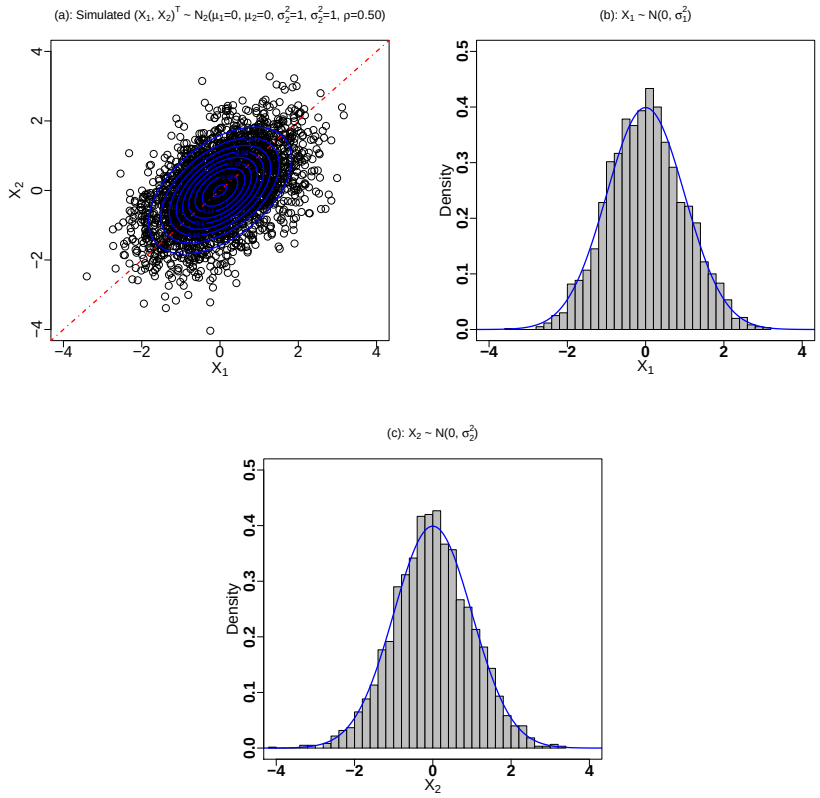


Figure 5.1: (a): Scatterplot with fitted contours based on 3000 samples generated from $\mathcal{N}_2(\mathbf{0}, \Sigma)$ in which $\Sigma = [(1, 0.5)^\top, (0.5, 1)^\top]$, (b): histogram of generated samples from marginal X_1, and (c): histogram of generated samples from marginal X_2.

5.2 Moments of truncated Gaussian distribution: Gibbs sampling scheme

The first two moments of the truncated multivariate Gaussian distribution, that is $\mathcal{N}_p(\boldsymbol{\mu}, \Sigma, \mathcal{R}^p)$, on region (3.270) has been discussed in Subsection 3.11. Herein, we are interested in dealing

with computing these quantities as well as the corresponding rectangular probability through the method of Gibbs sampling.

Recall from Example 2-8 that the marginals of $\mathcal{N}_p(\boldsymbol{\mu}, \Sigma, \mathcal{R}^p)$ are not necessarily truncated univariate Gaussian. In contrast, the full conditionals are truncated univariate Gaussian. Let $\boldsymbol{X} = (X_1, \cdots, X_p)^\top \sim \mathcal{N}_p(\boldsymbol{\mu}, \Sigma, \mathcal{R}^p)$. It is not hard to check that

$$\pi(x_i|\boldsymbol{x}[-i], \boldsymbol{\mu}, \Sigma) \sim \mathcal{N}\Big[\mu_{i|(-i)}(\boldsymbol{x})(\boldsymbol{x}[-i] - \boldsymbol{\mu}[-i]), \sigma^2_{i|(-i)}, \mathcal{R}_i\Big], \tag{5.5}$$

where $\boldsymbol{x}[-i]$ is given in (3.272) and

$$\mu_{i|(-i)}(\boldsymbol{x}) = \Sigma^\top_{i,-i} \times \Sigma^{-1}_{-i,-i} \times \boldsymbol{x}, \tag{5.6}$$

$$\sigma^2_{i|(-i)} = \Sigma_{i,i} - \Sigma^\top_{i,-i} \times \Sigma^{-1}_{-i,-i} \times \Sigma_{i,-i}, \tag{5.7}$$

$$\mathcal{R}_i = \{x \in \mathbb{R} | a_i \leq x \leq b_i\}, \tag{5.8}$$

in which a_i and b_i are the ith elements of vectors $\boldsymbol{a} = (a_1, \cdots, a_p)^\top$ and $\boldsymbol{b} = (b_1, \cdots, b_p)^\top$, respectively. In what follows, Algorithm 16 describes how to simulate from $\mathcal{N}(\mu, \Sigma, \mathcal{R}^p)$.

Algorithm 16 Computing the moments of truncated multivariate Gaussian distribution

Set $t = 0$, read M (*burn-in* period), N (total number of iterations), $\boldsymbol{\mu}$, Σ, $\boldsymbol{a}$, and $\boldsymbol{b}$;
Set $\boldsymbol{x}^{(t)} = (x_1^{(t)}, \cdots, x_p^{(t)})^\top$ as the vector of initial values in which $x_j^{(t)} = (a_j + b_j)/2$, for $j = 1, \cdots, p$;
while $t \leq N$ **do**
    Set $i = 1$;
    while $i \leq p$ **do**
        Generate sample y_i from full conditional with PDF

$$\pi(x_i|\boldsymbol{x}^{(t)}[-i], \boldsymbol{\mu}, \Sigma) \sim \mathcal{N}\Big[\mu_{i|(-i)}(\boldsymbol{x}[-i] - \boldsymbol{\mu}[-i]), \sigma^2_{i|(-i)}, \mathcal{R}_i\Big],$$

        in which $\mu_{i|(-i)}(\boldsymbol{x})$, $\sigma^2_{i|(-i)}$, and $\mathcal{R}_i$ are given in (5.6), (5.7), and (5.8), respectively.
        Set $\boldsymbol{x}^{(t)}[i] \leftarrow y_i$;
        Set $i \leftarrow i + 1$;
    end
    Set $t \leftarrow t + 1$;
end
Sequence $\{\boldsymbol{x}^{(M)}, \boldsymbol{x}^{(M+1)}, \cdots, \boldsymbol{x}^{(N)}\}$ is a sample of size $N - M + 1$ from $\mathcal{N}(\mu, \Sigma, \mathcal{R}^p)$.

Example 5-2

Recall that Example 3-20 computes the CDF of trivariate Gaussian distribution. Herein, we are interested in computing $\Phi_3(\boldsymbol{a}|\boldsymbol{0}, R)$ where $R = [(1, 0.7, -0.2)^\top, (0.7, 1, -0.4)^\top, (-0.2, -0.4, 1)^\top]$ and $\boldsymbol{a} = (-1.2, -1, 0.5)^\top$. Using `R` function `rtnorm_gibbs(N,M,Mu,Sigma,a,b)` given as follows.

```
R>rtnorm_gibbs <- function(N, M, Mu, Sigma, a, b)
+{
+ p <- length(Mu)
+ k <- 5.32
+ Z <- matrix(0, nrow = N, ncol = p)
+ V <- matrix(0, nrow = p, ncol = p-1)
+ threshold <- k*sqrt( diag(Sigma) )
```

```
 for(i in 1:p)
 {
  if( is.infinite( a[i] ) ) a[i] <- Mu[i] - threshold[i]
  if( is.infinite( b[i] ) ) b[i] <- Mu[i] + threshold[i]
  if( is.infinite( a[i] ) & a[i] >= b[i] ) a[i] <- b[i] - threshold[i]
  if( is.infinite( b[i] ) & a[i] >= b[i] ) b[i] <- a[i] + threshold[i]
  V[i, ] <- as.numeric( Sigma[i, -i]%*%solve( Sigma[-i, -i] ) )
 }
U <- matrix( cbind(a - Mu, b - Mu), nrow = p, ncol = 2)
Z[1, ] <- (a + b)/2
j <- 1
while(j < N)
{
 u <- runif(p)
 for(i in 1:p)
 {
  mu_i_minus_i <- V[i, ]%*%Z[j, -i]
  sigma_i_minus_i <- sqrt( (Sigma[i, i] - V[i, ]%*%Sigma[i, -i])[[1]] )
  Phi_a <- pnorm(U[i, 1], mu_i_minus_i, sigma_i_minus_i)
  Phi_b <- pnorm(U[i, 2], mu_i_minus_i, sigma_i_minus_i)
  Z[j, i] <- qnorm( u[i]*(Phi_b - Phi_a) + Phi_a,
    mu_i_minus_i, sigma_i_minus_i )
  Z[j+1, i] <- Z[j, i]
 }
 j <- j + 1
}
Z <- matrix(Mu, nrow = N, ncol = p, byrow = T) + Z
return( Z[((M+1):N), ] )
}
```

Setting $R = \Sigma$ and $\boldsymbol{b} = (+\infty, +\infty, +\infty)^\top$, we can easily compute the first moment of truncated Gaussian distribution by running `R` command as

```
R> set.seed(20251020)
R> N <- 10000; M <- 1500; a <- c(-1.2, -1, 0.5); b <- rep(Inf, 3)
R> v <- c(1, 0.7, -0.2, 0.7, 1, -0.4, -0.2, -0.4, 1)
R> Mu <- rep(0, 3); Sigma <- matrix(v, nrow = 3, ncol = 3)
R> x <- rtnorm_gibbs(N, M, Mu, Sigma, a, b) # dim x is  8500 * 3
R> apply(x, 2, mean)
R> EX(Mu, Sigma, a, b)
```

The result, based on 8500 realizations, would be `(0.20568623,0.02342306,1.08851900)` that is close enough to the exact value `(0.20067270,0.02071517,1.08395808)` obtained through function `EX(Mu,Sigma,a,b)` given earlier. It is worth to mention that in the last line of function `EX(Mu,Sigma,a,b)`, we prefer to use function `Spec_quad(a,b,Mu,Sigma,30)` rather than function `pmvnorm(a,b,mean=Mu,sigma=Sigma)` to avoid calling package `mvtnorm` that cannot yield an exact evaluation for $\mathcal{P}_G$ given in (3.275). ■

5.3 Gibbs sampling in Bayesian paradigm

Herein, we show how a Gibbs sampling technique described in the previous section can be used for computing the Bayesian estimator of the unknown parameter $\boldsymbol{\theta} = (\theta_1, \cdots, \theta_p)^\top$. Notice that, based on random sample $\{x_1, \cdots, x_n\}$ following a distribution with PDF $f(\cdot|\boldsymbol{\theta})$, the Bayes

estimator of $\boldsymbol{\theta}$ usually becomes the mean[2] of the posterior defined as

$$E[\boldsymbol{\theta}|\boldsymbol{x}] = \int_{\mathbb{R}^p} \boldsymbol{\theta}\pi(\boldsymbol{\theta}|\boldsymbol{x})d\boldsymbol{\theta}. \tag{5.9}$$

Computing integration (5.9) may involve huge computational burden especially when number of parameters p, is large. The law of large numbers guarantees that the sample mean converges with probability one to the population mean when sample size is sufficiently large, that is

$$\hat{\boldsymbol{\theta}} = \lim_{N\to\infty} \frac{\boldsymbol{\theta}^{(1)} + \cdots + \boldsymbol{\theta}^{(N)}}{N} = E[\boldsymbol{\theta}|\boldsymbol{x}].$$

Hence, if we could generate sample enough from the posterior $\pi(\boldsymbol{\theta}|\boldsymbol{x})$, then $\boldsymbol{\theta}$ can be estimated via $\hat{\boldsymbol{\theta}}$ in which $\{\boldsymbol{\theta}^{(1)}, \cdots, \boldsymbol{\theta}^{(N)}\}$ is generated from posterior $\pi(\boldsymbol{\theta}|\boldsymbol{x})$ when N is sufficiently large (here $\boldsymbol{\theta}^{(i)}$ denotes the ith sample generated from posterior $\pi(\boldsymbol{\theta}|\boldsymbol{x})$). Generating sample from the target distribution (that is here $\pi(\boldsymbol{\theta}|\boldsymbol{x})$) using the MCMC techniques described in the previous section appears to be challenging when p is large. In such cases, the Gibbs sampling is guaranteed to work well. It should be noted that since in each Bayesian framework the unknown parameter $\boldsymbol{\theta} = (\theta_1, \cdots, \theta_p)^\top$ itself is a random variable, in addition of the model's variables, herein we must consider an extra number of p additional full conditionals within a Gibbs sampler schemes. For instance, recalling from the Example (3), we must consider three full conditionals given by variables $X_1|X_2, \rho$, $X_2|X_1, \rho$, and $\rho|X_1, X_2$. In what follows, we consider the foregoing example when Bayesian inference on ρ is desired.

Example 5-3

Let $\underline{\boldsymbol{x}} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_n\}$ account for a sequence of n independent observations in which $\boldsymbol{x}_i = (x_{i1}, x_{i2})^\top$. Furthermore, assume $\boldsymbol{x}_i$ (for $i = 1, 2, \cdots, n$) follows a bivariate Gaussian distribution with PDF given by

$$\pi(\boldsymbol{x}_i|\boldsymbol{\theta}) = \frac{1}{2\pi\sigma_1\sigma_2\sqrt{1-\rho^2}} \exp\Big\{-\frac{1}{2(1-\rho^2)}\Big[\Big(\frac{x_{i1}}{\sigma_1}\Big)^2 + \Big(\frac{x_{i2}}{\sigma_2}\Big)^2 - 2\rho\frac{x_{i1}x_{i2}}{\sigma_1\sigma_2}\Big]\Big\}, \tag{5.10}$$

where $\boldsymbol{\theta} = (\sigma_1, \sigma_2, \rho)^\top$ is the unknown parameter vector. Using a reparameterization $\boldsymbol{\theta} = (\theta_1, \theta_2, \theta_3)^\top$ in which $\theta_1 = \sigma_1/\sigma_2$, $\theta_2 = \sigma_1\sigma_2\sqrt{1-\rho^2}$, and $\theta_3 = \rho$, the bivariate Gaussian PDF is rewritten as by

$$\pi(\boldsymbol{x}_i|\boldsymbol{\theta}) = \frac{1}{2\pi\theta_2} \exp\Big\{-\frac{1}{2(1-\theta_3^2)^{1/2}\theta_2}\Big[\frac{x_{i1}^2}{\theta_1} + \theta_1 x_{i2}^2 - 2\theta_3 x_{i1}x_{i2}\Big]\Big\}. \tag{5.11}$$

For computing the Bayesian estimator of $\boldsymbol{\theta}$, we first note that

$$\pi(\rho|\underline{\boldsymbol{x}}) \propto \prod_{i=1}^{n} \pi(\boldsymbol{x}_i|\boldsymbol{\theta})\pi(\boldsymbol{\theta}). \tag{5.12}$$

The marginals of joint prior $\pi(\boldsymbol{\theta})$ are assumed to be independent, that is, $\pi(\boldsymbol{\theta}) = \pi(\theta_1)\pi(\theta_2)\pi(\theta_3)$. Also, we consider gamma prior for both of θ_1, inverse gamma prior for θ_2, and an uniform on (-1,1) prior for θ represented, respectively, as follows.

$$\begin{aligned}
\pi(\theta_1|a_1, b_1) &= \frac{b_1^{a_1}\theta_1^{a_1-1}}{\Gamma(a_1)} \exp\{-b_1\theta_1\}, \quad \theta_1 > 0, \\
\pi(\theta_2|a_2, b_2) &= \frac{b_2^{a_2}\theta_2^{-a_2-1}}{\Gamma(a_2)} \exp\{-\frac{b_2}{\theta_2}\}, \quad \theta_2 > 0, \\
\pi(\theta_3) &= \frac{1}{2}, \quad -1 < \theta_3 < 1.
\end{aligned} \tag{5.13}$$

[2]Statistically speaking, the form of Bayesian estimator depends on the type of the loss function. For example, if the loss function takes either *square* or *absolute* form, then the Bayesian estimator is *mean* or *median* of posterior, respectively. But, the most commonly used summary representing the central tendency such as mean, median, or mode of the posterior would be considered as the Bayesian estimator.

We note that, here we consider above priors just for the simplicity of computations. It follows that

$$\pi(\boldsymbol{\theta}|\underline{\boldsymbol{x}}) \propto \frac{1}{\theta_2^n} \exp\Big\{-\frac{1}{2\theta_2\sqrt{1-\theta_3^2}}\Big[\frac{\sum_{i=1}^n x_{i1}^2}{\theta_1} + \theta_1 \sum_{i=1}^n x_{i2}^2 - 2\theta_3 \sum_{i=1}^n x_{i1}x_{i2}\Big]\Big\}$$
$$\times \frac{b_1^{a_1}\theta_1^{a_1-1}}{\Gamma(a_1)} \exp\{-b_1\theta_1\} \times \frac{b_2^{a_2}\theta_2^{-a_2-1}}{\Gamma(a_2)} \exp\{-\frac{b_2}{\theta_2}\} \times \frac{1}{2} \tag{5.14}$$

We note that, herein, the model's variables are θ_1, θ_2, and θ_3. So, by dropping out all terms that do not depend on θ_1 from the posterior (5.14), then the full conditional $\theta_1|(\theta_2, \theta_3, \underline{\boldsymbol{x}})$, after some algebra, is shown to be[3]

$$\pi(\theta_1|\theta_2, \theta_3, \underline{\boldsymbol{x}}) \propto \theta_1^{-a_1-1} \exp\Big\{-\frac{\theta_1}{2}\Big[2b_1 + \frac{\sum_{i=1}^n x_{i2}^2}{\theta_2(1-\theta_3)^{1/2}}\Big] - \frac{1}{2\theta_1}\Big[\frac{\sum_{i=1}^n x_{i1}^2}{\theta_2(1-\theta_3)^{1/2}}\Big]\Big\}. \tag{5.15}$$

Comparing the RHSs of (5.15) and (2.48), it turns out that (5.15) is the pseudo PDF corresponds to a $\mathcal{GIG}(a, b, c)$ distribution where $a = a_1$, $b = 2b_1 + \sum_{i=1}^n x_{i2}^2/[\theta_2(1-\theta_3)^{1/2}]$, $c = \sum_{i=1}^n x_{i1}^2/[\theta_2(1-\theta_3)^{1/2}]$. So,

$$\theta_1|(\theta_2, \theta_3, \underline{\boldsymbol{x}}) \sim \mathcal{GIG}\Big(a_1, 2b_1 + \sum_{i=1}^n x_{i2}^2/[\theta_2(1-\theta_3)^{1/2}], \sum_{i=1}^n x_{i1}^2/[\theta_2(1-\theta_3)^{1/2}]\Big). \tag{5.16}$$

Likewise, we can see that the full conditional $\theta_2|(\theta_1, \theta_3, \underline{\boldsymbol{x}})$ follows an inverse gamma distribution. We have

$$\theta_2|(\theta_1, \theta_3, \underline{\boldsymbol{x}}) \sim \mathcal{IG}\Big(n + a_2, b_2 + \frac{1}{2\theta_2(1-\theta_3)^{1/2}}$$
$$\times \Big[\theta_1^{-1}\sum_{i=1}^n x_{i1}^2 + \theta_1 \sum_{i=1}^n x_{i2}^2 - 2\theta_3 \sum_{i=1}^n x_{i1}x_{i2}\Big]\Big), \tag{5.17}$$

and finally

$$\theta_3|(\theta_1, \theta_2, \underline{\boldsymbol{x}}) \propto \exp\Big\{-\frac{1}{2\theta_2\sqrt{1-\theta_3^2}}\Big[\theta_1^{-1}\sum_{i=1}^n x_{i1}^2 + \theta_1 \sum_{i=1}^n x_{i2}^2 - 2\theta_3 \sum_{i=1}^n x_{i1}x_{i2}\Big]\Big\}. \tag{5.18}$$

As it is seen, the full conditional $\theta_3|(\theta_1, \theta_2, \underline{\boldsymbol{x}})$ has no closed form. Hence, we can use the MH within Gibbs sampling technique. Recall from Subsection **??**, considering a symmetric uniform proposal on (-1,1), the transition probability $p(\theta_{3(n)} \to \theta_{3(n+1)}) = \min\{1, \exp\{R_1 - R_2\}\}$, is

$$R_1 = -\frac{1}{2\theta_2\sqrt{1-\theta_{3(n+1)}^2}}\Big[\theta_1^{-1}\sum_{i=1}^n x_{i1}^2 + \theta_1 \sum_{i=1}^n x_{i2}^2 - 2\theta_{3(n+1)} \sum_{i=1}^n x_{i1}x_{i2}\Big]$$

$$R_2 = -\frac{1}{2\theta_2\sqrt{1-\theta_{3(n)}^2}}\Big[\theta_1^{-1}\sum_{i=1}^n x_{i1}^2 + \theta_1 \sum_{i=1}^n x_{i2}^2 - 2\theta_{3(n)} \sum_{i=1}^n x_{i1}x_{i2}\Big] \tag{5.19}$$

In what follows, we give the steps for computing Bayesian estimation of $\boldsymbol{\theta}$.

1. Read N, M, a_1, b_1, a_2, b_2, and suggest $\boldsymbol{\theta}^{(0)} = (\theta_1^{(0)}, \theta_2^{(0)}, \theta_3^{(0)})^\top$ arbitrarily;
2. Set $t = 0$;

[3]It is noteworthy that $\underline{\boldsymbol{x}}$ has been observed and we no longer look at it as variable. So, the model's variables are θ_1, θ_2, and θ_3.

3. Simulate $\theta_1^{(t+1)}$ from the full conditional $\theta_1^{(t+1)}|(\theta_2^{(t)}, \theta_3^{(t)}, \underline{\boldsymbol{x}})$ that follows

$$\mathcal{GIG}\Big(a_1, 2b_1 + \sum_{i=1}^{n} x_{2i}^2/[\theta_2^{(t)}(1-\theta_3^{(t)})^{1/2}], \sum_{i=1}^{n} x_{1i}^2/[\theta_2^{(t)}(1-\theta_3^{(t)})^{1/2}]\Big);$$

4. Having $\theta_1^{(t+1)}$, simulate $\theta_2^{(t+1)}$ from the full conditional $\theta_2^{(t)}|(\theta_1^{(t+1)}, \theta_3^{(t)}, \underline{\boldsymbol{x}})$ that follows

$$\mathcal{GIG}\Bigg(n + a_2, 2b_2, \Big[\frac{1}{\theta_1^{(t+1)}}\sum_{i=1}^{n} x_{i1}^2 + \theta_1^{(t+1)}\sum_{i=1}^{n} x_{i2}^2 - 2\theta_3^{(t)}\sum_{i=1}^{n} x_{i1}x_{i2}\Big]$$
$$\Big[\theta_2^{(t)}(1-\theta_3^{(t)})^{1/2}\Big]^{-1}\Bigg);$$

5. Having $\theta_1^{(t+1)}$ and $\theta_2^{(t+1)}$, simulate $\theta_3^{(t+1)}$ from the full conditional $\theta_3^{(t)}|(\theta_2^{(t+1)}, \theta_3^{(t)}, \underline{\boldsymbol{x}})$ using MH-within-Gibbs sampling technique as follows.
   (a) Set $k = 0$, $K = 100$, and choose the initial state as $y_0 \sim \mathcal{U}(-1, 1)$;
   (b) Generate $y^{(k+1)} \sim \mathcal{U}(-1, 1)$
   (c) Compute $p(y^{(k)} \to y^{(k+1)}) = \min\{1, \exp\{R_1 - R_2\}\}$ where

$$R_1 = -\frac{1}{2\theta_2^{(t+1)}\sqrt{1-(y^{(k+1)})^2}}\Big[\frac{1}{\theta_1^{(t+1)}}\sum_{i=1}^{n} x_{i1}^2 + \theta_1\sum_{i=1}^{n} x_{i1}^2 - 2y^{(k+1)}\sum_{i=1}^{n} x_{i1}x_{i2}\Big]$$

   and

$$R_2 = -\frac{1}{2\theta_2^{(t+1)}\sqrt{1-(y^{(k)})^2}}\Big[\frac{1}{\theta_1^{(t+1)}}\sum_{i=1}^{n} x_{i1}^2 + \theta_1^{(t+1)}\sum_{i=1}^{n} x_{i2}^2 - 2y^{(k)}\sum_{i=1}^{n} x_{i1}x_{i2}\Big]$$

   (d) Generate $u \sim \mathrm{U}(0, 1)$;
   (e) If $u < p(y^{(k)}, y^{(k+1)})$ then $y^{(k)} \to y^{(k+1)}$ and $k \to k+1$;
   (f) Repeat algorithm from step (b);
   (g) If $k = K$ then $y^{(k+1)} \to \theta_3^{(t+1)}$ and stop the algorithm.
6. If $t = N$, then go to the next step, otherwise $t+1 \to t$ and return step 3;
7. The Bayesian estimator $\hat{\boldsymbol{\theta}}_B$, is the average of $N-M+1$ samples, that is, $\boldsymbol{\theta}^{(M+1)}, \boldsymbol{\theta}^{(M+2)}, \cdots, \boldsymbol{\theta}^{(M+N)}$ generated from target PDF $\pi(\boldsymbol{\theta}|\underline{\boldsymbol{x}})$.

We note that, in practice, hyperparameters a_1, b_1, a_2 and b_2 are determined based on experimenter's belief or the empirical Bayes. If sample size is small, then the improper Jeffreys *prior* is suggested. It is noteworthy that for simulating from GIG distribution, one can use the method proposed by Hörmann and Leydold [2014]. ■

The pertaining `R` code for implementing example above is given as follows. To this end, a sample of $N = 5000$ realizations have been generated from a zero-mean bivariate Gaussian when $\sigma_1 = 0.5$, $\sigma_2 = 1$, and $\rho = -0.75$. The output of the Gibbs sampler is displayed in Figure 5.2.

```
R > set.seed(20240522)
R > library(MASS)
R > library(GIGrvg)
R > n <- 5000 # sample size
R > Mu <- c(0, 0)
```

```
R > X <- matrix(0, nrow = n, ncol = 2)
R > sigma1 <- 1; sigma2 <- .5; rho <- -0.75
R > Sigma <- matrix( c(sigma1^2,rho*sigma1*sigma2,rho*sigma1*sigma2,sigma2^2), 2, 2)
R > X <- mvrnorm(n, Mu, Sigma)
R > n.sim <- 5000  # number of Gibbs sampling iterations
R > n.burn <- 2000 # length of burn-in period
R > K <- 100       # number of MH algorithm iterations
R > theta3i <- rep(0, K)
R > a1 <- 0.5; b1<- 0.5; a2 <- 0.5; b2 <- 0.5;
R > theta <- matrix(0, nrow = n.sim, ncol = 3)
R > theta[1, ] <- c(1, 1, 0.75)
R > sx1 <- sum(X[, 1]^2); sx2 <- sum(X[, 2]^2); sx12 <- sum(X[, 1]*X[, 2])
R > j <- 1
R >  while(j < n.sim)
+   {
+   c0 <- ( theta[j, 2]*sqrt( 1 - theta[j, 3]^2 ) )
+   rate1 <- function(x) (sx1/theta[j, 1] + theta[j, 1]*sx2 - 2*x*sx12)/sqrt(1-x^2)
+   theta[j + 1, 1] <- rgig(n = 1, lambda = a1, chi = sx1/c0, psi = 2*b1 + sx2/c0 )
+   theta[j + 1, 2] <- 1/rgamma(n=1, shape=n + a2, rate=rate1(theta[j, 3])/2 + b2 )
#   start of MH-within-Gibbs sampling for sampling
#     from full conditional of theta3
+   theta3i[1] <- runif(1, -1, 1)
+    for(k in 2:K)
+    {
+     theta3.n <- runif(1, -1, 1)
+     dif <- -( rate1(theta3.n) - rate1(theta3i[k - 1]) )/(2*theta[j + 1, 2])
+      if( runif(1) < exp( dif ) )
+      {
+       theta3i[k] <- theta3.n # theta3.n is accepted
+      }else{
+       theta3i[k] <- theta3i[ k - 1] # theta3.n is rejected
+      }
+    }
#   start of MH-within-Gibbs sampling for sampling
#     from full conditional of theta3
+   theta[j + 1, 3] <- theta3i[k]
+   j <- j + 1
+   }
R > theta.hat <- apply(theta[(n.sim - n.burn + 1):n.sim, ], 2, mean)
R > rho <- theta.hat[3] #estimator of rho
R > sigma1 <- sqrt(theta.hat[1]*theta.hat[2]/sqrt(1-rho.hat^2)) #estimator of sigma1
R > sigma2 <- sigma1.hat/theta.hat[1] #estimator of sigma2
```

Example 5-4

Suppose response variable Y depends linearly on independent variable (covariate) X through a simple linear model as

$$y_i = \beta_0 + \beta_1 x_i + \epsilon_i; \quad i = 1, 2, 3, \cdots, n. \tag{5.20}$$

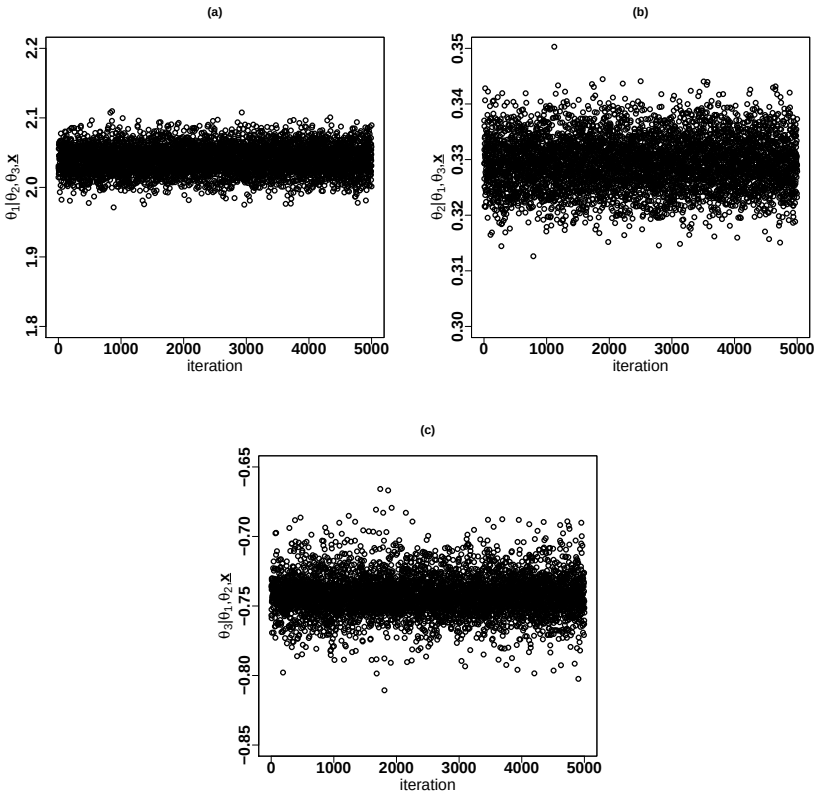


Figure 5.2: Scatterplot of 5000 samples generated from full conditionals: (a) $\theta_1|(\theta_2, \theta_3, \boldsymbol{x})$, (b): $\theta_2|(\theta_1, \theta_3, \boldsymbol{x})$, and (c): $\theta_3|(\theta_1, \theta_2, \boldsymbol{x})$.

Recording n measurements on Y represented as $y_1, y_2, y_3, \cdots, y_n$ at the corresponding levels of covariate X shown by $x_1, x_2, \cdots, x_n$, the focus is placed on estimating the regression coefficients β_0 and β_1 through the Bayesian approach. Herein, ϵ_is are the simple linear model's theoretical errors, independently (that is, $\text{cov}(\epsilon_i, \epsilon_j) = 0$ for $i \neq j = 1, 2, 3, \cdots, n$) assumed to follow $\mathcal{N}(0, \sigma^2)$. Of course the skewed Student's t [Teimouri et al., 2020] and α-stable [Nolan and Ojeda-Revah, 2013] may be reasonable candidates for distribution of theoretical error in presence of outliers that yield more robust estimation of $\boldsymbol{\beta} = (\beta_0, \beta_1)^\top$ than the Gaussian one. It is known that the least-square (LS) estimator $\hat{\boldsymbol{\beta}}_{ls}$ and ML estimator $\hat{\boldsymbol{\beta}}_{ml}$, of the regression coefficients $\boldsymbol{\beta}$ are the same and given by

$$\begin{aligned}
\hat{\beta}_{1ML} &= \hat{\beta}_{1LS} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2}, \\
\hat{\beta}_{0ML} &= \hat{\beta}_{0LS} = \bar{y} - \hat{\beta}_{0ML} \times \bar{x},
\end{aligned} \tag{5.21}$$

where $\bar{x} = (x_1 + x_2 + \cdots + x_n)/n$. Assuming independent Gaussian priors for elements of $\boldsymbol{\beta}$ and a mutually independent conjugate inverse gamma prior for $\delta = \sigma^2$ as

$$\begin{aligned}
\pi(\beta_0) =& \mathcal{N}(\beta_0|\mu_0, \sigma_0^2), \\
\pi(\beta_1) =& \mathcal{N}(\beta_1|\mu_1, \sigma_1^2), \\
\pi(\delta) =& \mathcal{IG}\Big(\delta\Big|\frac{a_0}{2}, \frac{b_0}{2}\Big).
\end{aligned}$$

It follows that

$$\begin{aligned}\pi(\boldsymbol{\theta}|\boldsymbol{y},\boldsymbol{x}) =& L(\boldsymbol{\theta}|\boldsymbol{y},\boldsymbol{x})\pi(\beta_0)\pi(\beta_1)\pi(\delta)\\ \propto& \frac{1}{\delta^{\frac{n}{2}}}\exp\Big\{-\frac{1}{2\delta}\sum_{i=1}^{n}\big(y_i-\beta_0-\beta_1x_i\big)^2\Big\}\frac{1}{\sqrt{2\pi\sigma_0^2}}\exp\Big\{-\frac{\big(\beta_0-\mu_0\big)^2}{2\sigma_0^2}\Big\}\\ &\times\frac{1}{\sqrt{2\pi\sigma_1^2}}\exp\Big\{-\frac{\big(\beta_1-\mu_1\big)^2}{2\sigma_1^2}\Big\}\Big(\frac{b_0}{2}\Big)^{\frac{a_0}{2}}\frac{\delta^{-\frac{a_0}{2}-1}}{\Gamma\big(\frac{a_0}{2}\big)}\exp\Big\{-\frac{b_0}{2\delta}\Big\},\end{aligned}\tag{5.22}$$

where $\delta=\sigma^2$ and $\boldsymbol{\theta}=(\beta_0,\beta_1,\delta)^\top$. Then more algebra shows

$$\pi(\beta_0|\boldsymbol{y},\boldsymbol{x},\boldsymbol{\theta}_{(-\beta_0)})=\mathcal{N}\bigg(\beta_0\Big|D_0\Big[\frac{\mu_0}{\sigma_0^2}+\frac{\sum_{i=1}^{n}\big(y_i-\beta_1x_i\big)}{\sigma^2}\Big],D_0\bigg),\tag{5.23}$$

and

$$\pi(\beta_1|\boldsymbol{y},\boldsymbol{x},\boldsymbol{\theta}_{(-\beta_1)})=\mathcal{N}\bigg(\beta_1\Big|D_1\Big[\frac{\mu_1}{\sigma_1^2}+\frac{\sum_{i=1}^{n}x_i\big(y_i-\beta_0\big)}{\sigma^2}\Big],D_1\bigg),\tag{5.24}$$

where $D_0=\sigma_0^2\delta/(n\sigma_0^2+\delta)$ and $D_1=\sigma_1^2\delta/(\sigma_1^2\sum_{i=1}^{n}x_i^2+\delta)$. Furthermore, by considering transformation $\delta=\sigma^2$, it can be seen easily that

$$\pi(\delta|\boldsymbol{y},\boldsymbol{x},\boldsymbol{\theta}_{(-\boldsymbol{\beta})})=\mathcal{IG}\bigg(\delta\Big|\frac{a_0+n}{2},\frac{a_0+\sum_{i=1}^{n}\big(y_i-x_i\beta\big)^2}{2}\Big]\bigg).\tag{5.25}$$

Applying a monotone transformation such used as above is a common trick in each Bayesian paradigm to yield a conjugate prior. Finally, the simple reverse transformation gives back the Bayesian estimator of σ. There follows the steps for computing Bayesian estimation of $\boldsymbol{\theta}$.

1. Read N, M, determine hyperparameters $\mu_0,\mu_1,\sigma_0,\sigma_1,a_0,b_0$, and propose $\boldsymbol{\theta}^{(0)}=(\beta_0^{(0)},\beta_1^{(0)},\delta^{(0)})^\top$;
2. Set $t=0$;
3. Simulate $\beta_0^{(t+1)}$ from the full conditional $\beta_0|(\boldsymbol{y},\boldsymbol{x},\boldsymbol{\theta}_{(-\beta_0)})$ that follows

$$\mathcal{N}\bigg(D_0^{(t)}\Big[\frac{\mu_0}{\sigma_0^2}+\frac{\sum_{i=1}^{n}\big(y_i-\beta_1^{(t)}x_i\big)}{\delta^{(t)}}\Big],D_0^{(t)}\bigg),$$

   where $D_0^{(t)}=\sigma_0^2\delta^{(t)}/(n\sigma_0^2+\delta^{(t)})$.

4. Having $\beta_0^{(t+1)}$, simulate $\beta_1^{(t+1)}$ from the full conditional $\beta_1|(\boldsymbol{y},\boldsymbol{x},\boldsymbol{\theta}_{(-\beta_1)})$ that follows

$$\mathcal{N}\bigg(D_1^{(t)}\Big[\frac{\mu_1}{\sigma_1^2}+\frac{\sum_{i=1}^{n}x_i\big(y_i-\beta_0^{(t+1)}\big)}{\delta^{(t)}}\Big],D_1^{(t)}\bigg),$$

   where $D_1^{(t)}=\sigma_1^2\delta^{(t)}/(\sigma_1^2\sum_{i=1}^{n}x_i^2+\delta^{(t)})$.

5. Having $\boldsymbol{\beta}^{(t+1)}$, simulate $\delta^{(t+1)}$ from the full conditional $\delta|(\boldsymbol{y},\boldsymbol{x},\boldsymbol{\theta}_{(-\delta)})$ that follows

$$\mathcal{IG}\bigg(\delta\Big|\frac{a_0+n}{2},\frac{b_0+\sum_{i=1}^{n}\big(y_i-\beta_0^{(t+1)}-x_i\beta_1^{(t+1)}\big)^2}{2}\Big]\bigg).$$

6. If $t=N$, then go to the next step. Otherwise set $t+1\to t$ and return to step 3;

7. The average of sequence $\{\boldsymbol{\theta}^{(M+1)}, \boldsymbol{\theta}^{(M+2)}, \cdots, \boldsymbol{\theta}^{(M+N)}\}$ sampled from target distribution with PDF $\pi(\boldsymbol{\theta}|\boldsymbol{y}, \boldsymbol{x})$ is the Bayesian estimator $\hat{\boldsymbol{\theta}}_B = (\hat{\beta}_{0B}, \hat{\beta}_{1B}, \hat{\sigma}_B)^\top$.

Herein, we set the hyperparameters $\{\mu_0, \mu_1\}$ are supposed to be $\widehat{\boldsymbol{\beta}_{ml}}$ and also, we set $\sigma_0 = 1$, $\sigma_1 = 1$, and $\sigma_2 = 1$. Furthermore, a_0 and b_0 are determined through the empirical Bayes as follows.

$$
\begin{aligned}
m(\boldsymbol{y}|a_0, b_0) =& \Pi_{i=1}^n m(y_i|a_0, b_0) \\
=& \Pi_{i=1}^n \int_0^\infty f(y_i|\delta)\pi(\delta|a_0, b_0)d\delta \\
=& \Pi_{i=1}^n \int_0^\infty \frac{\delta^{-\frac{1}{2}}}{\sqrt{2\pi}} \exp\Big\{-\frac{1}{2\delta}(y_i - \beta_0 - \beta_1 x_i)^2\Big\} \\
& \times \big(\frac{b_0}{2}\big)^{\frac{a_0}{2}} \delta^{-\frac{a_0}{2}-1} \Gamma^{-1}\big(\frac{a_0}{2}\big) \exp\Big\{-\frac{b_0}{2\delta}\Big\} d\delta \\
=& \Pi_{i=1}^n \big(\frac{b_0}{2}\big)^{\frac{a_0}{2}} \int_0^\infty \frac{\delta^{-\frac{a_0+1}{2}-1}}{\sqrt{2\pi}\Gamma\big(\frac{a_0}{2}\big)} \exp\Big\{-\frac{1}{2\delta}\big[(y_i - \beta_0 - \beta_1 x_i)^2 + b_0\big]\Big\} d\delta \\
=& \Pi_{i=1}^n \frac{\Gamma\big(\frac{a_0+1}{2}\big)}{\sqrt{b_0\pi}\Gamma\big(\frac{a_0}{2}\big)} \Big[1 + \frac{(y_i - \beta_0 - \beta_1 x_i)^2}{b_0}\Big]^{-\frac{a_0+1}{2}},
\end{aligned} \tag{5.26}
$$

where $f(y_i|\delta) = \mathcal{N}(y_i|\beta_0 + \beta_1 x_i, \delta = \sigma^2)$. Further, taking into account the fact that if $a_0 = b_0$, the RHS of (5.26) becomes $\Pi_{i=1}^n t(y_i|\beta_0 + \beta_1 x_i, b_0, a_0)$. Assuming $\boldsymbol{\beta}$ is known, the ML estimator of a_0 and b_0 is obtained by maximizing the RHS of (5.26) based on the sequence of n observed data $\{y_i - \beta_0 - \beta_1 x_i\}_{i=1}^n$. Moreover, the hyperparameters can be estimated through the the moment-based (MO) approach. While the likelihood function in the RHS of (5.26) is complicated in terms of a_0, a simple argument shows that this function gets its maximum when b_0 is large. We consider this fact when finding the ML estimators of a_0 and b_0. To illustrate how one can estimate the simple regression coefficients within a Bayesian framework, we generate $n = 200$ realizations $x_1, x_2, \cdots, x_n$ form uniform distribution on (-1,1) and then responses $y_1, y_2, \cdots, y_n$ are obtained using the linear relationship $y_i = \beta_0 + \beta_1 x_i + \epsilon_i = 5 + 2x_i + \epsilon_i$ where $\epsilon_i \sim \mathcal{N}(0, 1)$ for $i = 1, 2, \cdots, n$. ■

The corresponding `R` code for implementing this example is given as follows. The output of the Gibbs sampler is displayed in Figure 5.3.

```
R> set.seed(20240530)
R> n <- 200
R> x <- runif( n, -4, 4)
R> y <- 5 + 2*x + rnorm(n, mean = 0, sd = 2)
R> out <- lm(y ~ x)
R> ML <- as.vector( coefficients(out) )
R> error <- as.vector( y - ML[1] - ML[2]*x )
R> param0 <- var(error)
R> obj <- function(par) -sum( -log(par[2])/2 + lgamma( (par[1]+1)/2 ) -
+         lgamma( par[1]/2 ) - (par[1]+1)/2*log( 1 + error^2/par[2] ))
R> hyper0 <- optim( rep(param0, 2), fn = obj, method = "L-BFGS-B",
+            lower = rep(0.5, 2), upper = rep(n, 2) )$par
R> N <- 5000 # number of generations
R> M <- 2000 # size of burn-in period
R> theta <-matrix(0, nrow = N, ncol = 3 )
R> sigma0 <- 1; sigma1 <- 1;
R> Sigma0 <- c(sigma0, sigma1)
R> mu0 <- ML
R> delta <- 1
R> theta[1, ] <- c(ML, delta )
R> s.x2 <- c(n, sum(x^2) )
```

```
R> j <- 1
R> while(j < N)
+  {
+    Di <- Sigma0^2*theta[j, 3]/(Sigma0^2*s.x2 + theta[j, 3])
+    A0 <- sum( y - x*theta[j, 2] )
+    Mu0 <- Di[1]*(mu0[1]/Sigma0[1]^2 + sum(A0)/delta)
+    hat0 <- rnorm(1, Mu0, sqrt(Di[1]) )
+    theta[j + 1, 1] <- hat0
+    A1 <- sum( x*(y - theta[j + 1, 1]) )
+    Mu1 <- Di[2]*(mu0[2]/Sigma0[2]^2 + sum(A1)/delta)
+    hat1 <- rnorm(1, Mu1, sqrt(Di[2]) )
+    theta[j + 1, 2] <- hat1
+    SSE <- sum(error^2)
+    delta <- 1/sqrt( rgamma( 1, shape = (theta0[1] + n)/2,
+             rate = (theta0[2] + SSE)/2 ) )
+    theta[j + 1, 3] <- delta
+    j <- j + 1
+  }
R> Bayes <- apply(theta[(N - M + 1):N, ], 2, mean)
R> list(beta0 = Bayes[1], beta1 = Bayes[2], sigma = Bayes[3])
```

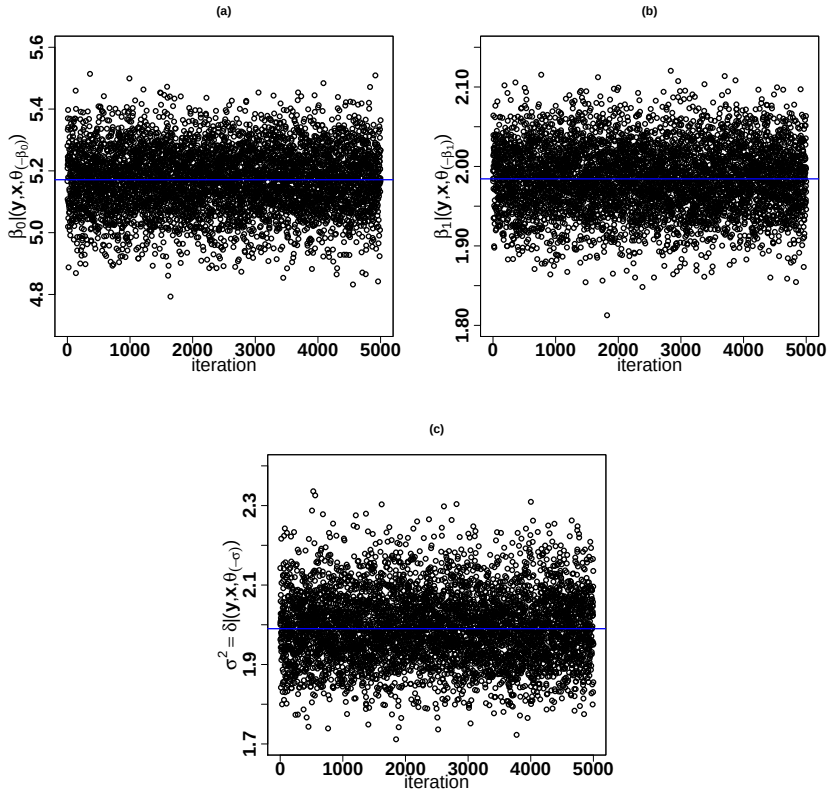


Figure 5.3: Scatterplot of 5000 samples generated from full conditionals: (a) $\beta_0|(\boldsymbol{y}, \boldsymbol{x}, \boldsymbol{\theta}_{(-\beta_0)})$, (b): $\beta_1|(\boldsymbol{y}, \boldsymbol{x}, \boldsymbol{\theta}_{(-\beta_1)})$, and (c): $\sigma|(\boldsymbol{y}, \boldsymbol{x}, \boldsymbol{\theta}_{(-\sigma)})$. The blue line, in each subfigure, shows the Bayesian estimator.

Example 5-5

Suppose response variable Y depends linearly on independent variable (covariate) X through a simple linear model as

$$\begin{aligned} y_i =& \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_K x_{iK} + \epsilon_i \\ =& \boldsymbol{x}_i^\top \boldsymbol{\beta} + \epsilon_i; \quad i = 1, 2, 3, \cdots, n, \end{aligned}$$

where $\boldsymbol{x}_i = (1, x_{i1}, \cdots, x_{iK})^\top$, $\boldsymbol{\beta} = (\beta_0, \beta_1, \cdots, \beta_K)^\top$ is the vector of unknown regression coefficients, and ϵ_is are the model's theoretical error assumed to follow $\mathcal{N}(0, \sigma^2)$ independently (that is, $\text{cov}(\epsilon_i, \epsilon_j) = 0$ for $i \neq j = 1, 2, 3, \cdots, n$). Of course the skewed Student's t [Teimouri et al., 2020] and α-stable [Nolan and Ojeda-Revah, 2013] may be reasonable candidates for distribution of theoretical error in presence of outliers that yield more robust estimation of $\boldsymbol{\beta}$ than the Gaussian one. It is known that the least-square (LS) estimator $\hat{\boldsymbol{\beta}}_{ls}$ and ML estimator $\hat{\boldsymbol{\beta}}_{ml}$, of the regression coefficients $\boldsymbol{\beta}$ are the same and given by

$$\hat{\boldsymbol{\beta}}_{ML} = \hat{\boldsymbol{\beta}}_{LS} = (\boldsymbol{X}^\top \boldsymbol{X})^{-1} \boldsymbol{X}^\top \boldsymbol{y}, \tag{5.27}$$

where $\boldsymbol{y}$ (vector of responses) and $\boldsymbol{X}$ is $n \times (K+1)$ design matrix are

$$\boldsymbol{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}, \ \boldsymbol{X} = \begin{bmatrix} \boldsymbol{x}_1 \\ \boldsymbol{x}_2 \\ \vdots \\ \boldsymbol{x}_n \end{bmatrix} = \begin{bmatrix} 1 & x_{11} & x_{12} & \cdots & x_{1K} \\ 1 & x_{21} & x_{22} & \cdots & x_{2K} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & x_{n1} & x_{n2} & \cdots & x_{nK} \end{bmatrix}. \tag{5.28}$$

Although it is possible to consider a Gaussian conjugate prior for each regression coefficient separately, but we prefer to consider a $(K+1)$-dimensional Gaussian prior for vector of regression coefficients $\boldsymbol{\beta}$ and a mutually independent inverse gamma conjugate prior for $\delta = \sigma^2$ as

$$\begin{aligned} \pi(\boldsymbol{\beta}) =& \mathcal{N}_{K+1}(\boldsymbol{\beta}|\boldsymbol{\mu}_0, \Sigma_0), \\ \pi(\delta) =& \mathcal{IG}\Big(\delta\Big|\frac{a_0}{2}, \frac{b_0}{2}\Big). \end{aligned} \tag{5.29}$$

It follows that

$$\begin{aligned} \pi(\boldsymbol{\Psi}|\boldsymbol{y}, \boldsymbol{x}) =& L(\boldsymbol{\Psi}|\boldsymbol{y}, \boldsymbol{x})\pi(\boldsymbol{\beta}) \\ \propto& \delta^{-\frac{n}{2}} \exp\Big\{-\frac{1}{2\delta}\sum_{i=1}^{n}(y_i - \boldsymbol{x}_i^\top \boldsymbol{\beta})^2\Big\} \exp\Big\{-\frac{1}{2}(\boldsymbol{\beta} - \boldsymbol{\mu}_0)^\top \Sigma_0^{-1}(\boldsymbol{\beta} - \boldsymbol{\mu}_0)\Big\} \\ & \times \Big(\frac{a_0}{2}\Big)^{\frac{a_0}{2}} \frac{\delta^{-a_0/2-1}}{\Gamma\big(\frac{a_0}{2}\big)} \exp\Big\{-\frac{a_0}{2\delta}\Big\}, \end{aligned} \tag{5.30}$$

where $\boldsymbol{\Psi} = (\boldsymbol{\beta}^\top, \delta)^\top$. For computing the full conditional $\boldsymbol{\beta}|(\boldsymbol{y}, \boldsymbol{x}, \boldsymbol{\Psi}_{(-\boldsymbol{\beta})})$, it is worth taking into account to the fact that

$$\sum_{i=1}^{n}(y_i - \boldsymbol{x}_i^\top \boldsymbol{\beta})^2 = \sum_{i=1}^{n} y_i^2 - 2\boldsymbol{\beta}^\top \boldsymbol{X}^\top \boldsymbol{y} + \boldsymbol{\beta}^\top \boldsymbol{X}^\top \boldsymbol{X}\boldsymbol{\beta}.$$

Then more algebra shows

$$\pi(\boldsymbol{\beta}|\boldsymbol{y}, \boldsymbol{x}, \boldsymbol{\Psi}_{(-\boldsymbol{\beta})}) \propto \exp\Big\{-\frac{1}{2}(\boldsymbol{\beta} - \boldsymbol{\mu}_\bullet)^\top \Sigma_\bullet^{-1}(\boldsymbol{\beta} - \boldsymbol{\mu}_\bullet)\Big\},$$

where $\boldsymbol{\mu}_\bullet = \Sigma_\bullet[\Sigma_0^{-1}\boldsymbol{\mu}_0 + \boldsymbol{X}^\top \boldsymbol{y}]$ and $\Sigma_\bullet = [\Sigma_0^{-1} + \boldsymbol{X}^\top \boldsymbol{X}]^{-1}$. So,

$$\pi(\boldsymbol{\beta}|\boldsymbol{y}, \boldsymbol{x}, \boldsymbol{\Psi}_{(-\boldsymbol{\beta})}) = \mathcal{N}_{K+1}\Big(\boldsymbol{\beta}\Big|\Sigma_\bullet\Big[\Sigma_0^{-1}\boldsymbol{\mu}_0 + \boldsymbol{X}^\top \boldsymbol{y}\Big], \Sigma_\bullet\Big).$$

Furthermore, it can be seen easily that

$$\pi(\delta|\boldsymbol{y}, \boldsymbol{x}, \boldsymbol{\Psi}_{(-\delta)}) = \mathcal{IG}\Big(\delta\Big|\frac{a_0 + n}{2}, \frac{b_0 + \sum_{i=1}^{n}(y_i - \boldsymbol{x}_i^\top \boldsymbol{\beta})^2}{2}\Big]\Big).$$

The hyperparameters a_0 and b_0 can be estimated in the same way as in Example 5-4, $\boldsymbol{\mu}_0$ is assumed to be (5.27), and $\Sigma_0 = \text{cov}(\hat{\boldsymbol{\beta}}_{ML}) = \sigma^2(\boldsymbol{X}^\top \boldsymbol{X})^{-1}$. In what follows, we give the steps for computing Bayesian estimation of $\boldsymbol{\Psi}$ in Example 5-5.

1. Read N, M, and determine hyperparameters a_0, b_0, $\boldsymbol{\mu}_0$, and Σ_0;
2. Set $t = 0$;
3. Simulate $\boldsymbol{\beta}^{(t+1)}$ from the full conditional $\boldsymbol{\beta}|(\boldsymbol{y}, \boldsymbol{x}, \boldsymbol{\Psi}_{(-\boldsymbol{\beta})})$ with PDF

$$\pi(\boldsymbol{\beta}|\boldsymbol{y}, \boldsymbol{x}, \boldsymbol{\Psi}_{(-\boldsymbol{\beta})}) = \mathcal{N}_{K+1}\Big(\boldsymbol{\beta}\Big|\boldsymbol{\mu}_\bullet, \Sigma_\bullet\Big),$$

   where $\boldsymbol{\mu}_\bullet = \Sigma_\bullet[\Sigma_0^{-1}\boldsymbol{\mu}_0 + \boldsymbol{X}^\top \boldsymbol{y}]$ and $\Sigma_\bullet = [\Sigma_0^{-1} + \boldsymbol{X}^\top \boldsymbol{X}]^{-1}$. So,
4. Having $\boldsymbol{\beta}^{(t+1)}$, set $\delta^{(t+1)} = \sqrt{z}$ where z follows $\mathcal{IG}$ distribution with PDF

$$\mathcal{IG}\left(z\left|\frac{a_0+n}{2}, \frac{a_0 + \sum_{i=1}^n (y_i - \boldsymbol{x}_i^\top \boldsymbol{\beta}^{(t+1)})^2}{2}\right.\right).$$

5. If $t = N$, then go to the next step. Otherwise set $t + 1 \to t$ and return to step 3;
6. The average of sequence $\{\boldsymbol{\Psi}^{(M+1)}, \boldsymbol{\Psi}^{(M+2)}, \cdots, \boldsymbol{\Psi}^{(M+N)}\}$ sampled from target distribution with PDF $\pi(\boldsymbol{\Psi}|\boldsymbol{y}, \boldsymbol{x})$ is the Bayesian estimator $\hat{\boldsymbol{\Psi}}_B = (\hat{\boldsymbol{\beta}}_B, \hat{\sigma}_B)^\top$.

■

Herein, Example 5-5 will be illustrated with computing the Bayesian estimator of regression plane coefficients fitted to `trees` data. This data set is available in `R` environment. The corresponding `R` code for implementing this example is given as follows. The output of the Gibbs sampler is displayed in Figure 5.4.

```
R> set.seed(20240601)
R> data(trees)
R> y <- trees$Volume;
R> n <- length(y)
R> x <- cbind(trees$Girth, trees$Height)
R> K <- 3 # number of regression coefficients
R> X <- as.matrix( cbind(1, x), nrow = n, ncol = K)
R> out <- lm(y ~ X[, 2] + X[, 3])
R> ML <- as.vector( coefficients(out) )
R> error <- as.vector(y - X%*%ML)
R> param0 <- var(error)
R> obj <- function(par) -sum( -log(par[2])/2 + lgamma( (par[1]+1)/2 ) -
+           lgamma( par[1]/2 ) - (par[1]+1)/2*log( 1 + error^2/par[2] ))
R> theta0 <- optim( rep(param0, 2), fn = obj, method = "L-BFGS-B",
+           lower = rep(0.5, 2), upper = rep(n, 2) )$par
R> N <- 5000 # number of generations
R> M <- 2000 # size of burn-in period
R> Psi <- matrix(0, nrow = N, ncol = K + 1)
R> delta <- 1
R> Psi[1, ] <- c(ML, delta )
R> hat <- Psi[1, ]
R> Mu0 <- ML
R> sigma2_hat <- theta0[2]/abs(theta0[1] - 1)
R> Sigma0_inv <- solve( sigma2_hat*solve(t(X)%*%X) )
R> j <- 1
R> while(j < N)
+  {
+  Sigma_bullet <- solve( Sigma0_inv + t(X)%*%X )
+  Mu_bullet <- Sigma_bullet%*%( Sigma0_inv%*%Mu0 + t(X)%*%y )
+  Psi[j+1, 1:K] <- mvrnorm(1, mu = Mu_bullet, Sigma = Sigma_bullet )
+  error <- as.vector( y - X%*%Psi[j+1, 1:K] )
+  SSE <- sum(error^2)
+  delta <- 1/sqrt(rgamma( 1, shape = (theta0[1] + n)/2,
```

```
+            rate = (theta0[2] + SSE)/2 ) )
+  Psi[j + 1, 4] <- delta
+  j <- j + 1
+  }
R> Bayes <- apply(Psi[(N - M + 1):N, ], 2, mean)
R> list(beta0=Bayes[1],beta1=Bayes[2],beta2=Bayes[3],sigma=sqrt(Bayes[4]))
```

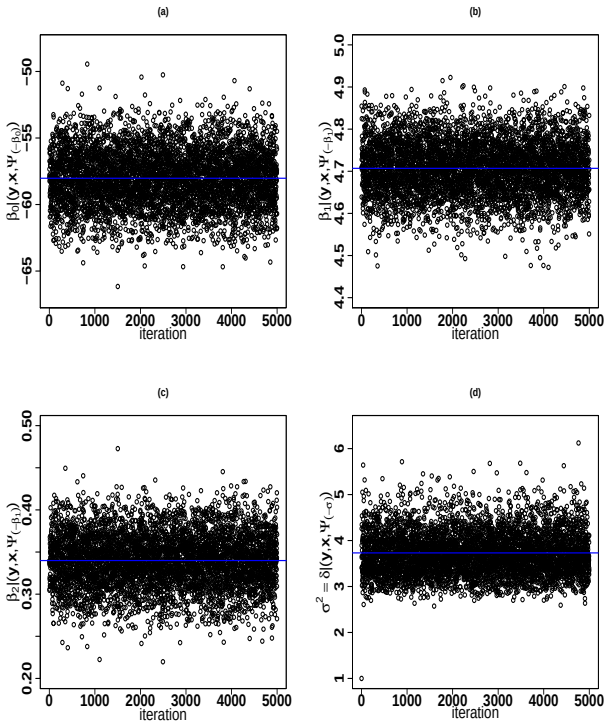


Figure 5.4: Scatterplot of 5000 samples generated from full conditionals: (a) $\beta_0|(\boldsymbol{y}, \boldsymbol{x}, \boldsymbol{\Psi}_{(-\beta_0)})$, (b): $\beta_1|(\boldsymbol{y}, \boldsymbol{x}, \boldsymbol{\Psi}_{(-\beta_1)})$, (c): $\beta_2|(\boldsymbol{y}, \boldsymbol{x}, \boldsymbol{\Psi}_{(-\beta_2)})$, and (d): $\sigma^2 = \delta|(\boldsymbol{y}, \boldsymbol{x}, \boldsymbol{\Psi}_{(-\delta)})$. The blue line, in each subfigure, shows the Bayesian estimator.

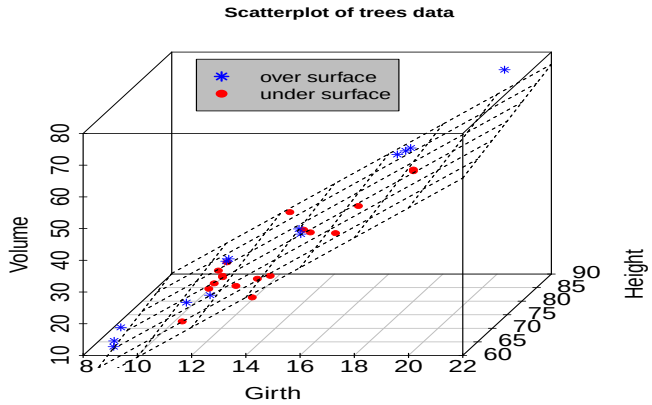


Figure 5.5: Scatterplot of trees data. Superimposed is the fitted regression plane whose coefficients are estimated through the Bayesian paradigm.

5.4 Gibbs sampling in Bayesian paradigm with one latent variable

Sometimes implementing the MCMC techniques such as MH or Gibbs sampling for drawing the Bayesian inference is computationally inefficient due to the complex form of the posterior. In such cases, inference about the unknown parameter(s) may be drawn by involving some latent (*unobservable* or *augmented*) variable(s). The rationale behind using the term "latent" traced back to this fact that the experimenter cannot observe some extra variable(s) existing in the model at quick glance, but there exists. For example, let random vector $\boldsymbol{X} = (X_1, \cdots, X_p)^\top$ follows a p-dimensional Student's t distribution with $\nu > 0$ degrees of freedom with PDF given by (3.28). Evidently implementing the MH or Gibbs sampling in which the full conditionals are constructed based on PDF (3.28) is quite difficult. On the other hand, it is easy to see that $\boldsymbol{X}$ admits the stochastic representation $\boldsymbol{X} \overset{d}{=} \boldsymbol{\mu} + \Sigma^{1/2}\boldsymbol{Z}/\sqrt{G}$ where $\boldsymbol{Z} \sim \mathcal{N}_p(\boldsymbol{0}_p, \boldsymbol{I}_p)$ and $G \sim \mathcal{G}(\nu/2, \nu/2)$ ($\mathcal{G}(a, b)$ accounts for a gamma distribution with shape and rate parameters, a and b, respectively.). Herein, random variable G plays the role of a latent variable. We also can consider more than one latent variable in each statistical model. Nevertheless, the Gibbs sampling is a very useful approach in presence of latent variable in the case of either sampling or Bayesian inference. In what follows, we give a definition that plays a main role in understanding Bayesian inference for the parameters of a statistical model that includes latent variable(s).

Definition 5.4.1. *Let $x_1, \cdots, x_n$ are a sample of n independent observed values from a model with PDF $f(x|\boldsymbol{\theta})$ where $\boldsymbol{\theta}$ is the unknown parameter vector. If model includes latent variable g whose PDF is $f(g|\boldsymbol{\theta}_g)$ in which $\boldsymbol{\theta}_g$ is the unknown parameter vector of distribution of g. Furthermore, let $f(x, g|\boldsymbol{\Psi})$ is the joint PDF of vector $(x, g)^\top$ where $\boldsymbol{\Psi} = \boldsymbol{\theta} \cup \boldsymbol{\theta}_g$. Then we have:*

- *The sequence $\{x_1, \cdots, x_n\}$ and $\{g_1, \cdots, g_n\}$, are called the observed and latent data, respectively.*
- *The sequence of paired-data $\{(x_1, g_1), \cdots, (x_n, g_n)\}$ is called complete data that evidently is the union of observed and latent data.*
- *The complete data likelihood function becomes*

$$L_c(\boldsymbol{\Psi}|\boldsymbol{x}, \boldsymbol{g}) = f(\boldsymbol{x}, \boldsymbol{g}|\boldsymbol{\Psi}) = \prod_{i=1}^{n} f(x_i|g_i, \boldsymbol{\Psi}) = \prod_{i=1}^{n} f(x_i|g_i, \boldsymbol{\theta}) f(g_i|\boldsymbol{\theta}_g). \tag{5.31}$$

Definition 5.4.1 is based on just one latent variable, but can be easily extended and is valid for two or more latent variables. It is noteworthy that for a Bayesian paradigm in which we have latent variable(s), we rewrite the posterior (5.30) in terms of the *complete data* likelihood function as follows.

$$\pi(\boldsymbol{\Psi}|\boldsymbol{x}, \boldsymbol{g}) \propto L_c(\boldsymbol{\Psi}|\boldsymbol{x}, \boldsymbol{g})\pi(\boldsymbol{\Psi}), \tag{5.32}$$

where $L_c(\boldsymbol{\Psi}|\boldsymbol{x}, \boldsymbol{g})$ is given by Definition 5.4.1. To implement the Bayesian paradigm, notice that the conditional PDF $f(x_i|g_i, \boldsymbol{\theta})$ and marginal PDF $f(g_i|\boldsymbol{\theta}_g)$ given in RHS of (5.31) must be known and possibly in closed form. In what follows, we give an example in which Bayesian inference on the parameters of a Student's t distribution is desired.

Example 5-6

He et al. [2021] Suppose $\underline{\boldsymbol{x}} = (\boldsymbol{x}_1, \cdots, \boldsymbol{x}_n)$ is vector of n independent observations from a p-dimensional Student's t distribution. Recall that for each Student's t random vector $\boldsymbol{X}$ we can write $\boldsymbol{X} \overset{d}{=} \boldsymbol{\mu} + \Sigma^{1/2}\boldsymbol{Z}/\sqrt{G}$. Evidently $\boldsymbol{X}$ admits the hierarchy given by

$$\begin{aligned} \boldsymbol{X}|G \sim f(\boldsymbol{x}_i|g_i, \boldsymbol{\theta} = (\boldsymbol{\mu}, \Sigma)) &= \mathcal{N}_p\Big(\boldsymbol{x}_i\Big|\boldsymbol{\mu}, \frac{\Sigma}{G}\Big), \\ G \sim f(g_i|\boldsymbol{\theta}_g = \nu) &= \mathcal{G}(g_i|\nu/2, \nu/2), \end{aligned} \tag{5.33}$$

where $\mathcal{N}_p(.|\boldsymbol{\mu}, \Sigma)$ denotes the PDF of a p-dimensional Gaussian distribution with location vector $\boldsymbol{\mu}$ and scale matrix Σ and $\mathcal{G}(\cdot|a, b)$ accounts for the PDF of a gamma distribution with shape parameter a and rate parameter b. Hence, the complete data likelihood is

$$\begin{aligned} L_c(\boldsymbol{\Psi}|\underline{\boldsymbol{x}}, \boldsymbol{g}) &= \prod_{i=1}^{n}\Big[\mathcal{N}_p(\boldsymbol{x}_i|\boldsymbol{\mu}, [g_i]^{-1}\Sigma) f(g_i|\boldsymbol{\theta}_g)\Big] \\ &= \Big[\Gamma\Big(\frac{\nu}{2}\Big)\Big]^{-n}\Big(\frac{\nu}{2}\Big)^{\frac{n\nu}{2}} \\ &\quad \times \prod_{i=1}^{n}\Big\{[g_i]^{\frac{\nu+p}{2}-1}\exp\Big\{-\frac{g_i}{2}\Big[\nu + (\boldsymbol{x}_i - \boldsymbol{\mu})^\top\Sigma^{-1}(\boldsymbol{x}_i - \boldsymbol{\mu})\Big]\Big\}\Big\}, \end{aligned} \tag{5.34}$$

where $\boldsymbol{\Psi} = \boldsymbol{\theta} \cup \boldsymbol{\theta}_g = (\boldsymbol{\mu}, \Sigma, \nu)$ is the model's parameter vector. For the prior, we proceed to assume independence, that is $\pi(\boldsymbol{\Psi}) = \pi(\boldsymbol{\mu})\pi(\Sigma)\pi(\nu)$ in which $\pi(\boldsymbol{\mu})$ and $\pi(\Sigma)$ are conjugate Gaussian and inverse Wishart priors. Furthermore, we let $\pi(\nu)$ to be the PDF of a gamma distribution. We have

$$\begin{aligned} \pi(\boldsymbol{\mu}) &\sim \mathcal{N}_p(\mathcal{M}_0, \mathcal{S}_0), \\ \pi(\Sigma) &\sim \mathcal{IW}(\mathcal{D}_0, \nu_0), \\ \pi(\nu) &\sim \mathcal{G}(a_0, b_0). \end{aligned}$$

We note that in above an objective prior such as Jeffreys' prior can be chosen, however, the most concern is placed on $\pi(\nu)$ that was proposed by Fernández and Steel [1999] to be $\pi(\nu) = 10\exp\{-10\nu\}$. It can be seen that the jeffreys' prior becomes $\pi(\boldsymbol{\mu}, \Sigma, \nu) = |\Sigma|^{-(p+1)/2}$ [Fernández and Steel, 1999], and so considering a flat prior for both of $\boldsymbol{\mu}$ and ν is acceptable. We note that for drawing inference about $\boldsymbol{\Psi}$, one needs to generate from full conditional of the latent variable too. So, by considering $\boldsymbol{\mu}$, Σ, ν, and g (latent variable) as the model's variables, within a Gibbs sampling framework, we proceed to sample from full conditionals $\boldsymbol{\mu}|(\Sigma, \nu)$, $\Sigma|(\boldsymbol{\mu}, \nu)$, $\nu|(\boldsymbol{\mu}, \Sigma)$. But, keep in mind that for each observed data $\boldsymbol{y}_i$, g_i (for $i = 1, \cdots, n$) is also latent unobservable). In order to g_i being known, we proceed to simulate g_i (for $i = 1 \cdots, n$) for each $\boldsymbol{y}_i$. To do this, in each iteration of the Gibbs sampling, we simulate (for each $\boldsymbol{y}_i$) from full conditional $g_i\big|(\boldsymbol{\mu}, \Sigma, \nu, \boldsymbol{y}_i)$ as follows. From RHS of (5.34), for a single observed value $\boldsymbol{y}_i$, it follows that

$$\pi(g_i|\boldsymbol{\Psi}, \boldsymbol{y}_i) \propto [g_i]^{\frac{\nu+p}{2}-1}\exp\Big\{-\frac{g_i}{2}\Big[\nu + (\boldsymbol{y}_i - \boldsymbol{\mu})^\top\Sigma^{-1}(\boldsymbol{y}_i - \boldsymbol{\mu})\Big]\Big\}.$$

This means that

$$\pi(g_i|\boldsymbol{\Psi}, \boldsymbol{y}_i) \sim \mathcal{G}\Big((\nu + p)/2, \nu/2 + (\boldsymbol{y}_i - \boldsymbol{\mu})^\top\Sigma^{-1}(\boldsymbol{y}_i - \boldsymbol{\mu})/2\Big).$$

Let $\boldsymbol{\Psi}_{(-\theta)}$ denote the whole parameter vector excluding its element θ, once we have generated a whole vector of latent variable as $\boldsymbol{g} = (g_1, \cdots, g_n)^\top$, the complete data is known and we proceed to simulate from other full conditionals as follows.

- **full conditional of $\boldsymbol{\mu}|(\underline{\boldsymbol{y}}, \boldsymbol{g}, \Psi_{(-\boldsymbol{\mu})})$:** By considering a conjugate prior, shown earlier by $\pi(\boldsymbol{\mu}) \sim \mathcal{N}_p(\mathcal{M}_0, \mathcal{S}_0)$, we have

$$\pi(\boldsymbol{\mu}|\underline{\boldsymbol{y}}, \boldsymbol{g}, \Psi_{(-\boldsymbol{\mu})}) \propto L_c(\boldsymbol{\Psi}|\underline{\boldsymbol{x}}, \boldsymbol{g})\mathcal{N}_p(\boldsymbol{\mu}|\mathcal{M}_0, \mathcal{S}_0).$$

  More algebra shows that

$$\boldsymbol{\mu}|(\underline{\boldsymbol{y}}, \boldsymbol{g}, \Psi_{(-\boldsymbol{\mu})}) \sim \mathcal{N}_p\Big(\mathcal{Q}_0\Big[\mathcal{S}_0^{-1}\mathcal{M}_0 + \Sigma^{-1}\sum_{i=1}^{n} g_i\boldsymbol{y}_i\Big], \mathcal{Q}_0\Big),$$

  where

$$\mathcal{Q}_0 = \Big[\mathcal{S}_0^{-1} + \Sigma^{-1}\sum_{i=1}^{n} g_i\Big]^{-1}.$$

- **full conditional of $\Sigma|(\underline{\boldsymbol{y}}, \boldsymbol{g}, \Psi_{(-\Sigma)})$:** By considering a conjugate prior, that is $\Sigma \sim \mathcal{IW}(\mathcal{D}_0, \nu_0)$, we can write

$$\pi(\Sigma|\underline{\boldsymbol{y}}, \boldsymbol{g}, \Psi_{(-\Sigma)}) \propto L_c(\boldsymbol{\Psi}|\underline{\boldsymbol{x}}, \boldsymbol{g})\mathcal{IW}(\Sigma|\mathcal{D}_0, \nu_0),$$

  where $\mathcal{IW}(\Sigma|\mathcal{D}_0, \nu_0)$ denotes the PDF of an inverse Wishart distribution. More algebra shows that

$$\Sigma|(\underline{\boldsymbol{y}}, \boldsymbol{g}, \Psi_{(-\Sigma)}) \sim \mathcal{IW}(\mathcal{R}_0, \nu_0 + n),$$

  where $\mathcal{R}_0 = \mathcal{D}_0 + \sum_{i=1}^{n} g_i(\boldsymbol{y}_i - \boldsymbol{\mu})(\boldsymbol{y}_i - \boldsymbol{\mu})^\top$.

- **full conditional of $\nu|(\underline{\boldsymbol{y}}, \boldsymbol{g}, \Psi_{(-\nu)})$:** By considering a truncated exponential distribution with PDF $\pi(\nu) = \mathcal{G}(\nu - 2|1, b_0)$ for $\nu \geq 2$, we can write

$$\begin{aligned}\pi(\nu|\underline{\boldsymbol{y}}, \boldsymbol{g}, \Psi_{(-\nu)}) &\propto L_c(\boldsymbol{\Psi}|\underline{\boldsymbol{x}}, \boldsymbol{g})\pi(\nu) \\ &\propto \Big[\Gamma\Big(\frac{\nu}{2}\Big)\Big]^{-n}\Big(\frac{\nu}{2}\Big)^{\frac{n\nu}{2}} b_0 \exp\Big\{-\frac{\nu}{2}[2b_0 + \sum_{i=1}^{n} g_i]\Big\}\prod_{i=1}^{n}[g_i]^{\frac{\nu}{2}}. \qquad (5.35)\end{aligned}$$

  As it is seen, the full conditional (5.35) has not closed form. The MH algorithm can be suggested to draw sample from this full conditional. It is easy to check that $\pi(\nu|\underline{\boldsymbol{y}}, \boldsymbol{g}, \Psi_{(-\nu)})$ is log-concave and hence the ARS-within-Gibbs sampling algorithm can be suggested for simulating from this full conditional. For computing the vector of hyperparameter $\boldsymbol{\theta}_0 = b_0$, it follows from hierarchy (5.33) that

$$\begin{aligned}m(\underline{\boldsymbol{x}}|b_0) &= \Pi_{i=1}^{n}\int_2^\infty f(\boldsymbol{x}_i|\nu)\pi(\nu|b_0)d\nu \\ &= \Pi_{i=1}^{n}\int_2^\infty \frac{\Gamma(\frac{\nu+p}{2})\pi(\nu|b_0)d\nu}{(\nu\pi)^{\frac{p}{2}}\Gamma(\frac{\nu}{2})|\Sigma|^{\frac{1}{2}}\big[1 + \frac{(\boldsymbol{x}-\boldsymbol{\mu})^\top\Sigma^{-1}(\boldsymbol{x}-\boldsymbol{\mu})}{\nu}\big]^{\frac{\nu+p}{2}}}, \qquad (5.36)\end{aligned}$$

where $\underline{\boldsymbol{x}} = (\boldsymbol{x}_1, \cdots, \boldsymbol{x}_n)$. Maximizing the RHS of (5.36) with respect to b_0 is not a simple task, an so we proceed to compute the moment-type estimator of b_0 as follows. First notice that

$$\begin{aligned} E\big[(\boldsymbol{X}-\boldsymbol{\mu})(\boldsymbol{X}-\boldsymbol{\mu})^\top | b_0\big] &= \int_2^\infty \int_{\mathbb{R}^p} (\boldsymbol{x}-\boldsymbol{\mu})(\boldsymbol{x}-\boldsymbol{\mu})^\top f(\boldsymbol{x}|\nu)\pi(\nu|b_0) d\boldsymbol{x} d\nu \\ &= \int_2^\infty \int_{\mathbb{R}^p} \frac{(\boldsymbol{x}-\boldsymbol{\mu})(\boldsymbol{x}-\boldsymbol{\mu})^\top \Gamma\big(\frac{\nu+p}{2}\big)}{(\nu\pi)^{\frac{p}{2}}\Gamma\big(\frac{\nu}{2}\big)|\Sigma|^{\frac{1}{2}}\big[1+\frac{(\boldsymbol{x}-\boldsymbol{\mu})^\top\Sigma^{-1}(\boldsymbol{x}-\boldsymbol{\mu})}{\nu}\big]^{\frac{\nu+p}{2}}} \\ &\quad \times \frac{b_0^{b_0-1}}{\Gamma(b_0)}(\nu-2)^{b_0-1}\exp\{-b_0(\nu-2)\} d\boldsymbol{x} d\nu \\ &= \int_2^\infty \Big(\frac{\nu}{\nu-2}\Sigma\Big)\frac{b_0^{b_0-1}}{\Gamma(b_0)}(\nu-2)^{b_0-1}\exp\{-b_0(\nu-2)\} d\nu \\ &= 3 - \frac{1}{b_0}. \end{aligned}$$

Let AD denote the average of diagonal elements of matrix $\big[\sum_{i=1}^n (\boldsymbol{x}_i^2-\boldsymbol{\mu})(\boldsymbol{x}_i^2-\boldsymbol{\mu})^\top/n\big]\Sigma^{-1}$, then b_0 can be exploited as $b_0 = 1/(3-AD)$.

Furthermore, one can employ the MH-within-Gibbs sampling technique for drawing sample from full conditional (5.35). To this end, we assume gamma distribution for the prior with PDF $\pi(\nu) = \mathcal{G}(.|b_0, b_0)$. Herein there is just one hyperparameter that we prefer to compute it as $b_0 = 1/(3-AD)$. We further choose $gamma(b_0, b_0)$ as the candidate. For other hyperparameters including $\mathcal{M}_0, \mathcal{S}_0, \mathcal{D}_0$, and ν_0, in example above, we set $\mathcal{M}_0$ to be the mean of $\underline{\boldsymbol{y}}$ and $\nu_0 = 1$. For $\mathcal{S}_0$ and $\mathcal{D}_0$ we suggest to consider $\boldsymbol{I}_p$. The steps for computing the Bayesian estimator of $\boldsymbol{\Psi}$ are given by Algorithm 17. ■

The pertaining `R` code for implementing example above is given as follows. The `R` code is applied to a sample of $n = 500$ realizations have been generated from a bivariate Student's t distribution with parameters $\boldsymbol{\mu} = (0, 2)^\top$, $\sigma_{11} = 1$, $\sigma_{12} = -0.50$, $\sigma_{22} = 0.75$, and $\nu = 5$. The output of the Gibbs sampler is displayed in Figure 5.6.

Algorithm 17 Bayesian inference for Student's t distribution

Read N, M, K, $\mathcal{M}_0$, $\mathcal{S}_0$, $\mathcal{D}_0$, and ν_0;
Set $t \leftarrow 0$;
Set $\boldsymbol{\mu}^{(0)} \leftarrow \mathcal{M}_0$, $\Sigma^{(0)} \leftarrow 1/n \sum_{i=1}^{n} (\boldsymbol{y}_i - \boldsymbol{\mu}^{(0)})(\boldsymbol{y}_i - \boldsymbol{\mu}^{(0)})^\top$, $\nu^{(0)} = 2$,
and set $\boldsymbol{\Psi}^{(0)} = (\boldsymbol{\mu}^{(0)}, \Sigma^{(0)}, \nu^{(0)})$;
while $t \leq N$ **do**
    Set $i = 1$;
    while $i \leq n$ **do**
        Simulate g_i from $\mathcal{G}(a, b)$ where $a = (\nu^{(t)} + p + 1)/2$ and $b =$
        $\Big[\nu^{(t)}/2 + (\boldsymbol{y}_i - \boldsymbol{\mu}^{(t)})^\top \big[\Sigma^{(t)}\big]^{-1} (\boldsymbol{y}_i - \boldsymbol{\mu}^{(t)})\Big]/2$;
        Set $i \leftarrow i + 1$;
    end
    Generate $\boldsymbol{\mu}^{(t+1)}$ from $\mathcal{N}_p\Big(\mathcal{Q}_0\Big[\mathcal{S}_0^{-1}\mathcal{M}_0 + \big[\Sigma^{(t)}\big]^{-1} \sum_{i=1}^{n} g_i \boldsymbol{y}_i\Big], \mathcal{Q}_0\Big)$ where
    $\mathcal{Q}_0 = \Big[\mathcal{S}_0^{-1} + \big[\Sigma^{(t)}\big]^{-1} \sum_{i=1}^{n} g_i\Big]^{-1}$;
    Generate $\Sigma^{(t+1)}$ from $\mathcal{IW}(\mathcal{R}_0, \nu_0 + n)$ where $\mathcal{R}_0 = \mathcal{D}_0 +$
    $\sum_{i=1}^{n} g_i (\boldsymbol{y}_i - \boldsymbol{\mu}^{(t+1)})(\boldsymbol{y}_i - \boldsymbol{\mu}^{(t+1)})^\top$;
    Use the MH-within-Gibbs sampling technique for simulating $\nu^{(t+1)}$
    as follows;
    Set $k = 0$ and $x_0 \leftarrow \nu_0$
    while $k \leq K$ **do**
        $x_0 \sim \mathcal{G}(a_0, b_0)$;
        Generate $x^{(k+1)} \sim \mathcal{G}(a_0, b_0)$;
        Compute $p(x^{(k)} \to x^{(k+1)}) = \min\{1, \exp\{R_1 - R_2\}\}$ where
        $R_1 = -n \log \Gamma\big(\frac{x^{(k+1)}}{2}\big) + \frac{n x^{(k+1)}}{2} \log\big(\frac{x^{(k+1)}}{2}\big) + (b_0 - 1) \log x^{(k+1)}$
        $-\frac{x^{(k+1)}}{2}\big[2b_0 + \sum_{i=1}^{n} g_i\big] + \frac{x^{(k+1)}}{2} \sum_{i=1}^{n} \log g_i$
        and
        $R_2 = -n \log \Gamma\big(\frac{x^{(k)}}{2}\big) + \frac{n x^{(k)}}{2} \log\big(\frac{x^{(k)}}{2}\big) + (b_0 - 1) \log x^{(k)}$
        $-\frac{x^{(k)}}{2}\big[2b_0 + \sum_{i=1}^{n} g_i\big] + \frac{x^{(k)}}{2} \sum_{i=1}^{n} \log g_i$;
        Generate $u \sim \mathcal{U}(0, 1)$;
        If $u < p\big(x^{(k)}, x^{(k+1)}\big)$ then $x^{(k)} \to x^{(k+1)}$ and $k \to k + 1$;
        Repeat algorithm from line (21);
        If $k = K$ then $\nu^{(t+1)} \leftarrow x^{(k+1)}$ and stop the algorithm;
    end
Set $\boldsymbol{\Psi}^{(t+1)} \leftarrow \big(\boldsymbol{\mu}^{(t+1)}, \Sigma^{(t+1)}, \nu^{(t+1)}\big)$ and $t \leftarrow t + 1$;
end
Sequence $\big\{\boldsymbol{\Psi}^{(M)}, \boldsymbol{\Psi}^{(M+1)}, \cdots, \boldsymbol{\Psi}^{(M)}\big\}$ is a sample of size $N - M + 1$ from
posterior with PDF $\pi(\boldsymbol{\Psi}|\underline{\boldsymbol{x}})$;
The Bayesian estimator of $\hat{\boldsymbol{\Psi}}_B$ is given by $\frac{1}{N-M+1} \sum_{t=1}^{N-M+1} \boldsymbol{\Psi}^{(t)}$.

```
R > rssg<-function(n, nu, Mu, Sigma)
+  {
+   Dim <- length(Mu)
+   Y <- matrix(NA, nrow = n, ncol = Dim)
+   for (i in 1:n)
+   {
+    Z <- rgamma(1, shape = nu/2, rate = nu/2)
+    X <- mvrnorm(n = 1, mu = rep(0, Dim), Sigma = Sigma )
+    Y[i, ] <- Mu + X/sqrt(Z)
+   }
+    Y
+  }
R > f.a <- function(x, b0, y)
+  {
+   n <- length(y)
+   -n*lgamma(x/2) + n*(x/2)*log(x/2) - x/2*( 2*b0 + sum(y) ) + x/2*sum( log(y) ) + (b0-1)*
    log(x)
+  }
R > fprim.a <- function(x, b0, y)
+  {
+   n <- length(y)
+   -n*digamma(x/2)/2 + n/2 + n/2*log(x/2) - ( 2*b0 + sum(y) )/2 + sum( log(y) )/2 + (b0-1)/
    x
+  }
R > set.seed(20240529)
R > library(MASS); library(ars)
R > n <- 500; nu <- 5; Mu <- c(0, 2)
R > Sigma <- matrix( c(1,-0.5,-0.5,0.75), 2, 2)
R > Y <- rssg(n, nu, Mu, Sigma)
R > N <- 5000; M <- 2000; p <- length(Mu)
R > Psi <- matrix(0, nrow = N, ncol = p + 1 + p^2)
R > g <- rate <- rep(0, n)
R > SOMu <- diag(2); MOMu <- rep(0, 2); nu0 <- 2
R > D0 <- matrix(diag(2), 2, 2)
R > for(j in 1:N)
+   {
+    rate <- mahalanobis(x = Y, center = Mu, cov = Sigma )
+    for(i in 1:n) g[i] <- rgamma( 1, shape = (nu + p)/2, rate = rate[i]/2 + nu/2 )
+    Sigmainv <- solve( Sigma )
+    dy <- mahalanobis( Y, center = Mu, cov = Sigma )
+    Sum.g <- sum(g)
+    M_sum <- rowSums( sapply(1:n, function(i) g[i]*c( Y[i, ] ) ) )
+    Q <- solve( solve( SOMu ) + Sigmainv*Sum.g )
+    Mu <- c( mvrnorm(1, mu = Q%*%( solve( SOMu )%*%MOMu +
+           Sigmainv%*%M_sum ) , Sigma = Q) )
+    Psi[j, 1:p] <- Mu
+    R0 <- matrix(0, nrow = p, ncol = p)
+    for(i in 1:n){R0 <- R0 + g[i]*c( Y[i, ] - Mu )%*%t(c( Y[i, ] - Mu ))}
+    Sigma <- solve(rWishart(1, df = nu0 + n, Sigma = solve(D0 + R0) )[,,1])
+    Psi[j, (p+1):(p + p^2)] <- as.numeric(Sigma)
+    b0 <- 1/(3-(cov(Y)%*%solve(Sigma))[1,1])
+    nu <- ars(1, f.a, fprim.a, x = c(2, 5, 10, 50), m = 4,
+          lb = TRUE, xlb = 2, b0 = b0, y = g)
+    Psi[j, p + 1 + p^2] <- nu
+   }
R > Psi.hat <- apply(Psi[(N - M + 1):N, ], 2, mean)
R > Mu <- Psi.hat[1:p]
R > Sigma <- Psi.hat[(p+1):(p + p^2)]
R > nu <- Psi.hat[p + 1 + p^2]
```

5.5 Gibbs sampling in Bayesian paradigm with two latent variables

5.5.1 Heavy-tailed skew models

The Bayesian paradigm for $\mathrm{SG}_{p,q}(\boldsymbol{\mu}, \Sigma, \Lambda)$ distribution with representation (3.306) has been developed by Liseo and Parisi [2013]. The skew Gaussian model introduced in Section 2.8.4 can be developed for modelling the heavy tailed skewed data. This can be accomplished by involving an extra latent variable G. Let

$$\boldsymbol{Y} \overset{d}{=} \boldsymbol{\mu} + \frac{\boldsymbol{X}}{\sqrt{h(G)}}, \tag{5.37}$$

in which $\boldsymbol{X} \sim \mathrm{SG}_{p,q}(\mathbf{0}_p, \Sigma, \Lambda)$ and $h(\cdot)$ is a continuous monotone real function. Equivalently,

$$\boldsymbol{Y} \overset{d}{=} \boldsymbol{\mu} + \frac{\Lambda|\boldsymbol{Z}_0|}{\sqrt{h(G)}} + \Sigma^{\frac{1}{2}} \frac{\boldsymbol{Z}_1}{\sqrt{h(G)}}. \tag{5.38}$$

It is easy to see that random vector $\boldsymbol{Y}$ in (5.38) admits the hierarchy given by

$$\begin{aligned} \boldsymbol{Y}|\boldsymbol{U}, G &\sim \mathcal{N}_p\Big(\boldsymbol{\mu} + \Lambda\boldsymbol{U}, \frac{\Sigma}{h(G)}\Big), \\ \boldsymbol{U}|G &\sim \mathcal{HN}_q\Big(\mathbf{0}_q, \frac{\boldsymbol{I}_q}{h(G)}\Big), \\ G &\sim f(g|\boldsymbol{\theta}), \end{aligned} \tag{5.39}$$

where the short form $\mathcal{HN}_q(\mathbf{0}_q, \Sigma)$ accounts for the half-normal distribution truncated on $\mathbb{R}^{q+}$ with a zero-location vector of length q and scale matrix Σ. Regarding $\{\boldsymbol{y}_1, \cdots, \boldsymbol{y}_n\}$ as the sequence of observed data and two sequences of latent variables are represented as $\{\boldsymbol{u}_1, \cdots, \boldsymbol{u}_n\}$ and $\{g_1, \cdots, g_n\}$. Based on hierarchy (5.39) the complete data log-likelihood is

$$\begin{aligned} L_c(\boldsymbol{\Psi}|\underline{\boldsymbol{y}}, \underline{\boldsymbol{u}}, \boldsymbol{g}) &= \prod_{i=1}^{n} \Big[\boldsymbol{\phi}_p(\boldsymbol{y}_i|\boldsymbol{\mu} + \boldsymbol{\Lambda}\boldsymbol{u}_i, [h(g_i)]^{-1}\Sigma)\boldsymbol{\phi}_q(\boldsymbol{u}_i|\mathbf{0}_q, [h(g_i)]^{-1}\boldsymbol{I}_q) f(g_i|\boldsymbol{\theta})\Big] \\ &\propto \prod_{i=1}^{n} [h(g_i)]^{\frac{p+q}{2}} \exp\Big\{-\frac{h(g_i)}{2}\Big[(\boldsymbol{y}_i - \boldsymbol{\mu} - \boldsymbol{\Lambda}\boldsymbol{u}_i)^\top \Sigma^{-1}(\boldsymbol{y}_i - \boldsymbol{\mu} - \boldsymbol{\Lambda}\boldsymbol{u}_i) \\ &\quad + \boldsymbol{u}_i^\top \boldsymbol{I}_q \boldsymbol{u}_i\Big]\Big\} f(g_i|\boldsymbol{\theta}), \end{aligned} \tag{5.40}$$

where $\boldsymbol{\Psi} = (\boldsymbol{\mu}, \Sigma, \boldsymbol{\Lambda}, \boldsymbol{\theta})$ is the model's parameter vector. We write $\boldsymbol{Y} \sim \mathrm{SGX}_{p,q}(\boldsymbol{\mu}, \Sigma, \boldsymbol{\Lambda}, \boldsymbol{\theta})$ (the term SG in SGX is a short form for skewed Gaussian and X refers to distribution of latent variable G), to denote the family of distributions that follow representation (5.38). For example, if in (5.38) G follows a generalized Lindley distribution, then $\mathrm{SGGL}_{p,q}(\boldsymbol{\mu}, \Sigma, \boldsymbol{\Lambda}, \boldsymbol{\theta})$ refers to the class of skew Gaussian generalized Lindely distribution in which $\boldsymbol{\theta} = (\omega, \beta, \gamma)^\top$.

Theorem 5.5.1. *Let $\boldsymbol{Y} \sim SGGL_{p,q}(\boldsymbol{\mu}, \Sigma, \Lambda, \boldsymbol{\theta})$. The PDF of $\boldsymbol{Y}$ is given by*

$$\begin{aligned} f(\boldsymbol{y}|\boldsymbol{\Psi}) =& \frac{2^q \beta^{1-\frac{p}{2}} \omega^{\frac{p}{2}}}{(\beta+\gamma)} \boldsymbol{t}_p\Big(\sqrt{\frac{\omega}{\beta}}(\boldsymbol{y} - \boldsymbol{\mu})\Big|\mathbf{0}, \Omega, 2\omega\Big) \boldsymbol{T}_q\Big(\boldsymbol{m}\sqrt{\frac{p+2\omega}{2\beta + d(\boldsymbol{y})}}\Big|\mathbf{0}, \Delta, p+2\omega\Big) \\ &+ \frac{2^q \gamma \beta^{-\frac{p}{2}} (\omega+1)^{\frac{p}{2}}}{(\beta+\gamma)} \boldsymbol{t}_p\Big(\sqrt{\frac{\omega+1}{\beta}}(\boldsymbol{y} - \boldsymbol{\mu})\Big|\mathbf{0}, \Omega, 2\omega+2\Big) \\ &\times \boldsymbol{T}_q\Big(\boldsymbol{m}\sqrt{\frac{p+2\omega+2}{2\beta + d(\boldsymbol{y})}}\Big|\mathbf{0}, \Delta, p+2\omega+2\Big), \end{aligned}$$

where $\boldsymbol{\Psi} = (\boldsymbol{\mu}, \Sigma, \Lambda, \boldsymbol{\theta})$ *,* $\boldsymbol{\theta} = (\omega, \beta, \gamma)^\top$*,* $d(\boldsymbol{y}) = (\boldsymbol{y}-\boldsymbol{\mu})^\top \Omega^{-1}(\boldsymbol{y}-\boldsymbol{\mu})$*,* $\boldsymbol{m} = \Lambda^\top \Omega^{-1}(\boldsymbol{y}-\boldsymbol{\mu})$*,* $\Omega = \Sigma + \Lambda\Lambda^\top$*,* $\Delta = \boldsymbol{I}_q - \Lambda^\top \Omega^{-1}\Lambda$*,* $\boldsymbol{t}_p(.|\boldsymbol{0}, \Omega, 2\omega)$ *is the PDF of a p-dimensional Student's t with location parameter* $\boldsymbol{0}$*, dispersion matrix* Ω *and* 2ω *degrees of freedom, and* $\boldsymbol{T}_q(.|\boldsymbol{0}, \Delta, p+2\omega+2)$ *is the CDF of q-dimensional Student's t distribution with location parameter* $\boldsymbol{0}$*, dispersion matrix* Δ*, and* $p + 2\omega + 2$ *degrees of freedom.*

Proof: By definition, based on (5.39), we have

$$\begin{aligned}
f_{\boldsymbol{Y}}(\boldsymbol{y}|\Psi) =& \int_{\mathbb{R}^{q+}} \int_0^\infty f(\boldsymbol{y}|\boldsymbol{t}, g, \Psi) f(\boldsymbol{t}|g) f(g|\boldsymbol{\theta}) dg d\boldsymbol{t}, \\
=& \frac{2^q |\Sigma|^{-\frac{1}{2}}}{(2\pi)^{\frac{p+q}{2}}} \int_{\mathbb{R}^{q+}} \int_0^\infty g^{\frac{p+q}{2}} \exp\Big\{-\frac{g}{2}(\boldsymbol{y}-\boldsymbol{\mu}-\Lambda\boldsymbol{t})^\top \Sigma^{-1}(\boldsymbol{y}-\boldsymbol{\mu}-\Lambda\boldsymbol{t})\Big\} \\
& \times \exp\Big\{-\frac{g}{2}\boldsymbol{t}^\top \boldsymbol{I}_q \boldsymbol{t}\Big\} f(g|\boldsymbol{\theta}) dg d\boldsymbol{t}
\end{aligned}$$

$$\begin{aligned}
=& \frac{2^q}{(2\pi)^{\frac{p+q}{2}} |\Sigma|^{\frac{1}{2}}} \int_{\mathbb{R}^{q+}} \int_0^\infty g^{\frac{p+1}{2}} \exp\Big\{-\frac{g}{2}\Big[d(\boldsymbol{y}) + (\boldsymbol{t}-\boldsymbol{m})^\top \Delta^{-1}(\boldsymbol{t}-\boldsymbol{m})\Big]\Big\} \\
& \times f(g|\boldsymbol{\theta}) dg d\boldsymbol{t} \\
=& \frac{2^q \beta^{\alpha+1}}{(\beta+\gamma)\Gamma(\alpha)(2\pi)^{\frac{p+q}{2}} |\Sigma|^{\frac{1}{2}}} \int_{\mathbb{R}^{q+}} \int_0^\infty g^{\frac{p+q+2\alpha}{2}-1} \\
& \times \exp\Big\{-\frac{g}{2}\Big[d(\boldsymbol{y}) + (\boldsymbol{t}-\boldsymbol{m})^\top \Delta^{-1}(\boldsymbol{t}-\boldsymbol{m}) + 2\beta\Big]\Big\} dg d\boldsymbol{t} \\
& + \frac{2^q \gamma \beta^{\alpha+1}}{\alpha(\beta+\gamma)\Gamma(\alpha)(2\pi)^{\frac{p+q}{2}} |\Sigma|^{\frac{1}{2}}} \int_{\mathbb{R}^{q+}} \int_0^\infty g^{\frac{p+q+2\alpha+2}{2}-1} \\
& \times \exp\Big\{-\frac{g}{2}\Big[d(\boldsymbol{y}) + (\boldsymbol{t}-\boldsymbol{m})^\top \Delta^{-1}(\boldsymbol{t}-\boldsymbol{m}) + 2\beta\Big]\Big\} dg d\boldsymbol{t}.
\end{aligned}$$

It follows that

$$\begin{aligned}
f_{\boldsymbol{Y}}(\boldsymbol{y}|\Psi) =& \frac{2^q \beta^{\alpha+1} \Gamma\big(\frac{p+q+2\alpha}{2}\big) |\Sigma|^{-\frac{1}{2}}}{(\beta+\gamma)\Gamma(\alpha)(2\pi)^{\frac{p+q}{2}}} \int_{\mathbb{R}^{q+}} \frac{d\boldsymbol{t}}{\Big[\frac{2\beta + d(\boldsymbol{y}) + (\boldsymbol{t}-\boldsymbol{m})^\top \Delta^{-1}(\boldsymbol{t}-\boldsymbol{m})}{2}\Big]^{\frac{p+q+2\alpha}{2}}} \\
& + \frac{2^q \gamma \beta^{\alpha+1} \Gamma\big(\frac{p+q+2\alpha+2}{2}\big) |\Sigma|^{-\frac{1}{2}}}{\alpha(\beta+\gamma)\Gamma(\alpha)(2\pi)^{\frac{p+q}{2}}} \int_{\mathbb{R}^{q+}} \frac{d\boldsymbol{t}}{\Big[\frac{2\beta + d(\boldsymbol{y}) + (\boldsymbol{t}-\boldsymbol{m})^\top \Delta^{-1}(\boldsymbol{t}-\boldsymbol{m})}{2}\Big]^{\frac{p+q+2\alpha+2}{2}}} \\
=& I_1 + I_2. \qquad (5.41)
\end{aligned}$$

Using a change of variable $\boldsymbol{v} = (\boldsymbol{t}-\boldsymbol{m})\sqrt{p+2\alpha}/\sqrt{2\beta + d(\boldsymbol{y})}$, we can write

$$\begin{aligned}
I_1 =& \frac{2^q \beta^{\alpha+1} \Gamma\big(\frac{p+q+2\alpha}{2}\big) |\Sigma|^{-\frac{1}{2}}}{(\beta+\gamma)\Gamma(\alpha)(2\pi)^{\frac{p+q}{2}}} \frac{2^{\frac{p+q+2\alpha}{2}} (p+2\alpha)^{-\frac{q}{2}}}{\Big[2\beta + d(\boldsymbol{y})\Big]^{\frac{p+2\alpha}{2}}} \int_{-\boldsymbol{\infty}}^{\boldsymbol{m}_0} \frac{d\boldsymbol{v}}{\Big[1 + \frac{\boldsymbol{v}^\top \Delta^{-1} \boldsymbol{v}}{p+2\alpha}\Big]^{\frac{p+q+2\alpha}{2}}}, \\
=& \frac{2^q \beta^{\alpha+1} \Gamma\big(\frac{p+2\alpha}{2}\big) |\Sigma|^{-\frac{1}{2}}}{(\beta+\gamma)\Gamma(\alpha)(2\pi)^{\frac{p}{2}}} \frac{2^{\frac{p+2\alpha}{2}} |\Delta|^{\frac{1}{2}}}{\Big[2\beta + d(\boldsymbol{y})\Big]^{\frac{p+2\alpha}{2}}} \boldsymbol{T}_q\Big(\boldsymbol{m}\sqrt{\frac{p+2\alpha}{2\beta + d(\boldsymbol{y})}}\Big|\boldsymbol{0}, \Delta, p+2\alpha\Big),
\end{aligned}$$

where $\boldsymbol{m}_0 = \boldsymbol{m}\sqrt{p+2\alpha}/\sqrt{2\beta+d(\boldsymbol{y})}$. It follows that

$$\begin{aligned}
I_1 =& \frac{2^q\beta^{\alpha+1}2^{\frac{p+2\alpha}{2}}(2\alpha)^{\frac{p}{2}}|\Omega|^{\frac{1}{2}}|\Delta|^{\frac{1}{2}}}{(\beta+\gamma)2^{\frac{p}{2}}|\Sigma|^{\frac{1}{2}}}\frac{\Gamma\big(\frac{p+2\alpha}{2}\big)}{\pi^{\frac{p}{2}}(2\alpha)^{\frac{p}{2}}\Gamma(\alpha)|\Omega|^{\frac{1}{2}}\Big[2\beta+d(\boldsymbol{y})\Big]^{\frac{p+2\alpha}{2}}} \\
&\times \boldsymbol{T}_q\Big(\boldsymbol{m}\sqrt{\frac{p+2\alpha}{2\beta+d(\boldsymbol{y})}}\Big|\boldsymbol{0},\Delta,p+2\alpha\Big) \\
=& \frac{2^q\beta^{1-\frac{p}{2}}\alpha^{\frac{p}{2}}}{(\beta+\gamma)}\boldsymbol{t}_p\Big(\sqrt{\frac{\alpha}{\beta}}(\boldsymbol{y}-\boldsymbol{\mu})\Big|\boldsymbol{0},\Omega,2\alpha\Big)\boldsymbol{T}_q\Big(\boldsymbol{m}\sqrt{\frac{p+2\alpha}{2\beta+d(\boldsymbol{y})}}\Big|\boldsymbol{0},\Delta,p+2\alpha\Big). \qquad (5.42)
\end{aligned}$$

It should be noted that for obtaining the term in the RHS of (5.42), we use this fact that $|\Delta|^{\frac{1}{2}}\Omega|^{\frac{1}{2}}/|\Sigma|^{\frac{1}{2}} = 1$. Likewise, we can obtain I_2 as

$$\begin{aligned}
I_2 =& \frac{2^q\gamma\beta^{\alpha+1}\Gamma\big(\frac{p+2\alpha+2}{2}\big)}{\alpha(\beta+\gamma)\Gamma(\alpha)(2\pi)^{\frac{p}{2}}|\Sigma|^{\frac{1}{2}}}\frac{2^{\frac{p+2\alpha+2}{2}}|\Delta|^{\frac{1}{2}}}{\Big[2\beta+d(\boldsymbol{y})\Big]^{\frac{p+2\alpha+2}{2}}} \\
&\times \boldsymbol{T}_q\Big(\boldsymbol{m}\sqrt{\frac{p+2\alpha+2}{2\beta+d(\boldsymbol{y})}}\Big|\boldsymbol{0},\Delta,p+2\alpha\Big), \\
=& \frac{2^q\gamma\beta^{\alpha+1}2^{\frac{p+2\alpha+2}{2}}(2\alpha+2)^{\frac{p}{2}}|\Omega|^{\frac{1}{2}}|\Delta|^{\frac{1}{2}}}{(\beta+\gamma)2^{\frac{p}{2}}|\Sigma|^{\frac{1}{2}}}\frac{\Gamma\big(\frac{p+2\alpha+2}{2}\big)\Big[2\beta+d(\boldsymbol{y})\Big]^{-\frac{p+2\alpha+2}{2}}}{\pi^{\frac{p}{2}}(2\alpha+2)^{\frac{p}{2}}\alpha\Gamma(\alpha)|\Omega|^{\frac{1}{2}}} \\
&\times \boldsymbol{T}_q\Big(\boldsymbol{m}\sqrt{\frac{p+2\alpha+2}{2\beta+d(\boldsymbol{y})}}\Big|\boldsymbol{0},\Delta,p+2\alpha+2\Big) \\
=& \frac{2^q\gamma\beta^{-\frac{p}{2}}(\alpha+1)^{\frac{p}{2}}}{(\beta+\gamma)}\boldsymbol{t}_p\Big(\sqrt{\frac{\alpha+1}{\beta}}(\boldsymbol{y}-\boldsymbol{\mu})\Big|\boldsymbol{0},\Omega,2\alpha+2\Big) \\
&\times \boldsymbol{T}_q\Big(\boldsymbol{m}\sqrt{\frac{p+2\alpha+2}{2\beta+d(\boldsymbol{y})}}\Big|\boldsymbol{0},\Delta,p+2\alpha+2\Big).
\end{aligned}$$

The proof is complete.

Corollary 5.5.1. *If in (5.39) $G \sim \mathcal{G}(\nu/2,\nu/2)$, then $\boldsymbol{Y} \sim SNG_{p,q}(\nu,\boldsymbol{\mu},\Sigma,\Lambda)$ follows skew Student's t distribution with PDF given by*

$$f(\boldsymbol{y}|\boldsymbol{\Psi}) = 2^q\boldsymbol{t}_p(\boldsymbol{y}|\boldsymbol{\mu},\Omega,\nu)\boldsymbol{T}_q\Big(\boldsymbol{m}\sqrt{\frac{\nu+p}{\nu+d(\boldsymbol{y})}}\Big|\boldsymbol{0},\Delta,p+\nu\Big),$$

where $\boldsymbol{\Psi} = (\nu,\boldsymbol{\mu},\Sigma,\Lambda)$.

Theorem 5.5.2. *Let $\boldsymbol{Y} \sim SNGIG_{p,q}(\boldsymbol{\mu},\Sigma,\Lambda,\boldsymbol{\theta})$. The PDF of $\boldsymbol{Y}$ is given by*

$$f(\boldsymbol{y}|\boldsymbol{\Psi}) = 2^q\boldsymbol{h}_p(\boldsymbol{y}|\boldsymbol{\mu},\Omega,\tau,\chi,\chi)\boldsymbol{H}_q\Big(\boldsymbol{m}\Big[\frac{\chi}{\chi+d(\boldsymbol{y})}\Big]^{\frac{1}{4}}\Big|\boldsymbol{0},\Delta,\tau-\frac{p}{2},\kappa,\kappa\Big),$$

where $\boldsymbol{\Psi} = (\boldsymbol{\mu},\Sigma,\boldsymbol{\Lambda},\boldsymbol{\theta})$, $\boldsymbol{\theta} = (\tau,\chi)^\top$, $\kappa = \chi\sqrt{1+d(\boldsymbol{y})/\chi}$, $\boldsymbol{h}_p(.|\boldsymbol{\mu},\Omega,\tau,\chi,\chi)$ and $\boldsymbol{H}_q(\cdot|\boldsymbol{0},\Delta,\tau-p/2,\kappa,\kappa)$ are the PDF and CDF of a p- and q-dimensional symmetric hyperbolic distribution, respectively.

Proof: See Murray et al. [2017].

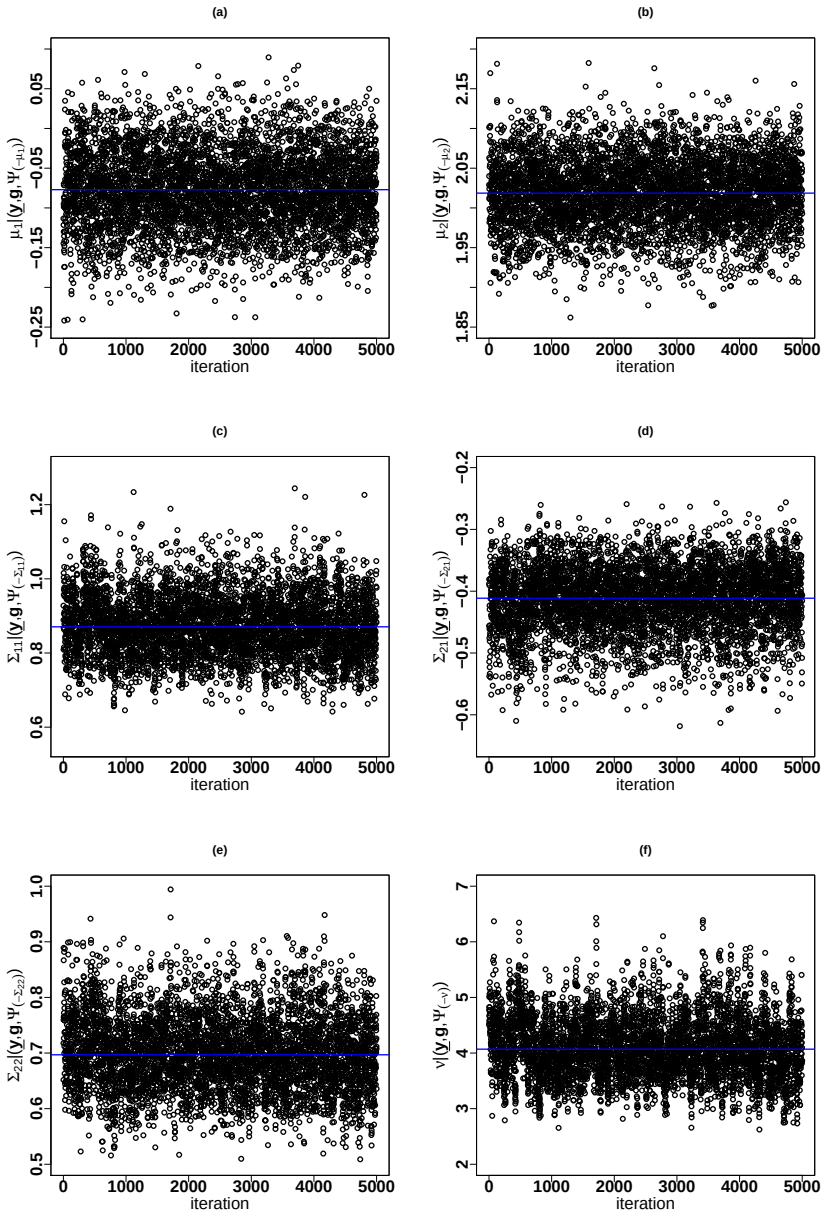


Figure 5.6: Scatterplot of 5000 samples generated from full conditionals: (a) $\mu_1|(\boldsymbol{y}, \boldsymbol{g}, \Psi_{(-\mu_1)})$, (b): $\mu_2|(\boldsymbol{y}, \boldsymbol{g}, \Psi_{(-\mu_2)})$, (c): $\Sigma_{11}|(\boldsymbol{y}, \boldsymbol{g}, \Psi_{(-\Sigma_{11})})$, (d) $\Sigma_{12}|(\boldsymbol{y}, \boldsymbol{g}, \Psi_{(-\Sigma_{12})})$, (e) $\Sigma_{22}|(\boldsymbol{y}, \boldsymbol{g}, \Psi_{(-\Sigma_{22})})$, and (f) $\nu|(\boldsymbol{y}, \boldsymbol{g}, \Psi_{(-\nu)})$. The blue line, in each subfigure, shows the Bayesian estimator.

Table 5.1: Family of heavy-tailed skew models

Characteristics	$\text{SGGL}_{p,q}(\boldsymbol{\mu}, \Sigma, \Lambda, \boldsymbol{\theta})$	$\text{SGG}_{p,q}(\boldsymbol{\mu}, \Sigma, \Lambda, \boldsymbol{\theta})$	$\text{SGGIG}_{p,q}(\boldsymbol{\mu}, \Sigma, \Lambda, \boldsymbol{\theta})$
family of G	Generalized Lindley ($\mathcal{GL}(\omega, \beta, \gamma)$)	$\mathcal{G}(\eta, \zeta)$	generalized inverse Gaussian ($\mathcal{GIG}(\tau, \chi, \chi)$)
parameter	$\boldsymbol{\theta} = (\omega, \beta, \gamma)^\top$	$\boldsymbol{\theta} = (\eta, \zeta)^\top$	$\boldsymbol{\theta} = (\tau, \chi)^\top$
PDF of G	$f(g\|\boldsymbol{\theta}) = \frac{\beta^{\omega+1} g^{\omega-1}(\omega+\gamma g)}{(\beta+\gamma)\Gamma(\omega+1)} \exp\{-\beta g\}$	$f(g\|\boldsymbol{\theta}) = \frac{\zeta^\eta}{\Gamma(\eta)} g^{\eta-1} \exp\{-\zeta g\}$	$f(g\|\boldsymbol{\theta}) = \frac{g^{\tau-1}}{2\mathcal{K}_\tau(\chi)} \exp\{-\frac{\chi g}{2} - \frac{\chi}{2g}\}$
PDF	$f(\boldsymbol{y}\|\boldsymbol{\Psi}) = \frac{2^q \beta^{1-\frac{p}{2}} \omega^{\frac{p}{2}}}{(\beta+\gamma)} \boldsymbol{t}_p\Big(\sqrt{\frac{\omega}{\beta}}(\boldsymbol{y}-\boldsymbol{\mu})\Big\|\boldsymbol{0}, \Omega, 2\omega\Big)$	$f(\boldsymbol{y}\|\boldsymbol{\Psi}) = 2^q \big(\frac{\eta}{\zeta}\big)^{\frac{p}{2}}$	$f(\boldsymbol{y}\|\boldsymbol{\Psi}) = 2^q \boldsymbol{h}_p\Big(\boldsymbol{y}\big\|\boldsymbol{\mu}, \Omega, \tau, \chi, \chi\Big)$
	$\times \boldsymbol{T}_q\Big(\boldsymbol{m}\sqrt{\frac{p+2\omega}{2\beta+d(\boldsymbol{y})}}\Big\|\boldsymbol{0}, \Delta, p+2\omega\Big)$	$\times \boldsymbol{t}_p\Big(\sqrt{\frac{\eta}{\zeta}}(\boldsymbol{y}-\boldsymbol{\mu})\Big\|\boldsymbol{0}, \Omega, 2\eta\Big)$	$\times \boldsymbol{H}_q\Big(\boldsymbol{m}\Big[\frac{\chi}{\chi+d(\boldsymbol{y})}\Big]^{\frac{1}{4}}\Big\|\boldsymbol{0}, \Delta, \tau-\frac{p}{2}, \chi\sqrt{1+\frac{d(\boldsymbol{y})}{\chi}},$
	$+\frac{2^q \gamma \beta^{-\frac{p}{2}} (\omega+1)^{\frac{p}{2}}}{(\beta+\gamma)} \boldsymbol{t}_p\Big(\sqrt{\frac{\omega+1}{\beta}}(\boldsymbol{y}-\boldsymbol{\mu})\Big\|\boldsymbol{0}, \Omega, 2\omega+2\Big)$	$\times \boldsymbol{T}_q\Big(\boldsymbol{m}\sqrt{\frac{p+2\eta}{2\zeta+d(\boldsymbol{y})}}\Big\|\boldsymbol{0}, \Delta, p+2\eta\Big)$	$\chi\sqrt{1+\frac{d(\boldsymbol{y})}{\chi}}\Big)$
	$\times \boldsymbol{T}_q\Big(\boldsymbol{m}\sqrt{\frac{p+2\omega+2}{2\beta+d(\boldsymbol{y})}}\Big\|\boldsymbol{0}, \Delta, p+2\omega+2\Big)$		
Representation	$\boldsymbol{Y} \overset{d}{=} \boldsymbol{\mu} + \frac{\Lambda\|\boldsymbol{Z}_0\|}{\sqrt{GL}} + \Sigma^{\frac{1}{2}} \frac{\boldsymbol{Z}_1}{\sqrt{GL}}$	$\boldsymbol{Y} \overset{d}{=} \boldsymbol{\mu} + \frac{\Lambda\|\boldsymbol{Z}_0\|}{\sqrt{G}} + \Sigma^{\frac{1}{2}} \frac{\boldsymbol{Z}_1}{\sqrt{G}}$	$\boldsymbol{Y} \overset{d}{=} \boldsymbol{\mu} + \frac{\Lambda}{\sqrt{GIG}}\big\|\boldsymbol{Z}_0\big\| + \Sigma^{\frac{1}{2}} \frac{\boldsymbol{Z}_1}{\sqrt{GIG}}$

Full conditional $G|(\boldsymbol{Y}=\boldsymbol{y}_i,\cdots)$

Implementing the Gibbs sampling for making Bayesian inference about parameters of the heavy-tailed skew models represented earlier in (5.39) involves sampling from the full conditionals of variables G and $\boldsymbol{U}$. Herein, we represent the full conditionals $G|(\boldsymbol{Y}=\boldsymbol{y}_i,\boldsymbol{U}=\boldsymbol{u}_i,\boldsymbol{\Psi})$ and $\boldsymbol{U}|(\boldsymbol{Y}=\boldsymbol{y}_i,G=g_i,\boldsymbol{\Psi})$, and describe how to sample from it under three scenarios shown in Table 5.1.

i. **Scenario 1 (mixing random variable is $\mathcal{GL}(\omega,\beta,\gamma)$):** By definition we have

$$\begin{aligned}\pi(g_i|\boldsymbol{y}_i,\boldsymbol{u}_i,\boldsymbol{\Psi}) =&\frac{f(\boldsymbol{y}_i,\boldsymbol{u}_i,g_i|\boldsymbol{\Psi})}{f(\boldsymbol{y}_i,\boldsymbol{u}_i|\boldsymbol{\Psi})}\\ \propto& f(\boldsymbol{y}_i,\boldsymbol{u}_i,g_i|\boldsymbol{\Psi})\\ =& f(\boldsymbol{y}_i|\boldsymbol{u}_i,g_i|\boldsymbol{\Psi})f(\boldsymbol{u}_i|g_i)f(g_i|\boldsymbol{\theta})\\ \propto&\frac{\beta^{\omega+1}}{(\beta+\gamma)\Gamma(\omega+1)}\Big[h(g_i)\Big]^{\frac{p+q}{2}}\Big[g_i^{\omega-1}(\omega+\gamma g_i)\Big]\exp\Big\{-\beta g_i\\ &-\frac{h(g_i)}{2}\Big[(\boldsymbol{y}_i-\boldsymbol{\mu}-\boldsymbol{\Lambda}\boldsymbol{u}_i)^\top\Sigma^{-1}(\boldsymbol{y}_i-\boldsymbol{\mu}-\boldsymbol{\Lambda}\boldsymbol{u}_i)+\boldsymbol{u}_i^\top\boldsymbol{u}_i\Big]\Big\}.\end{aligned}\tag{5.43}$$

Setting $h(g_i)=g_i$ in the RHS of (5.43), it turns out that

$$\begin{aligned}\pi(g_i|\boldsymbol{y}_i,\boldsymbol{u}_i,\boldsymbol{\Psi})\propto& g_i^{\frac{p+q}{2}+\omega-1}(\omega+\gamma g_i)\exp\Big\{-g_i\Big[\beta+\frac{\boldsymbol{u}_i^\top\boldsymbol{u}_i}{2}+(\boldsymbol{y}_i-\boldsymbol{\mu}-\Lambda\boldsymbol{u}_i)^\top\\ &\Sigma^{-1}(\boldsymbol{y}_i-\boldsymbol{\mu}-\Lambda\boldsymbol{u}_i)/2\Big]\Big\}.\end{aligned}\tag{5.44}$$

Comparing the RHS of (5.44) with the PDF $f(g|\boldsymbol{\theta})$ given in the first column of Table 5.1. Then more algebra shows that

$$G|(\boldsymbol{Y}=\boldsymbol{y}_i,\boldsymbol{U}=\boldsymbol{u}_i,\boldsymbol{\Psi})\sim\mathcal{GL}(\omega^*,\beta^*,\gamma^*),\tag{5.45}$$

where $\omega^*=(p+q)/2+\omega$, $\beta^*=\beta+(\boldsymbol{y}_i-\boldsymbol{\mu}-\Lambda\boldsymbol{u}_i)^\top\Sigma^{-1}(\boldsymbol{y}_i-\boldsymbol{\mu}-\boldsymbol{\Lambda}\boldsymbol{u}_i)/2+\boldsymbol{u}_i^\top\boldsymbol{u}_i/2$, and $\gamma^*=\gamma/\omega$.

ii. **Scenario 2 (mixing random variable is $\mathcal{G}(\eta,\zeta)$):** Following the same argument as in Scenario 1 when $\gamma=0$, $\omega=\eta$, and $\beta=\zeta$, then it turns out from RHS of (5.45) that

$$G|(\boldsymbol{Y}=\boldsymbol{y}_i,\boldsymbol{U}=\boldsymbol{u}_i,\boldsymbol{\Psi})\sim\mathcal{G}(\eta^*,\zeta^*),\tag{5.46}$$

where $\eta^*=(p+q)/2+\eta$ and $\zeta^*=2\zeta+(\boldsymbol{y}_i-\boldsymbol{\mu}-\Lambda\boldsymbol{u}_i)^\top\Sigma^{-1}(\boldsymbol{y}_i-\boldsymbol{\mu}-\Lambda\boldsymbol{u}_i)/2+\boldsymbol{u}_i^\top\boldsymbol{u}_i/2$.

iii. **Scenario 3 (mixing random variable is $\mathcal{GIG}(\tau,\chi,\chi)$):** If we let $f(g_i|,\boldsymbol{\theta})$ in the RHS of (5.40) to be PDF of GIG distribution whose form is given in third column of Table 5.1, we have

$$\begin{aligned}\pi(g_i|\boldsymbol{y}_i,\boldsymbol{u}_i,\boldsymbol{\Psi})\propto&\Big[\frac{g_i^{\tau-1}}{2\mathcal{K}_\tau(\chi)}\exp\Big\{-\frac{\chi g_i}{2}-\frac{\chi}{2g_i}\Big\}\Big]\big[h(g_i)\big]^{\frac{p+q}{2}}\exp\Big\{-\frac{h(g_i)}{2}\\ &\times\Big[(\boldsymbol{y}_i-\boldsymbol{\mu}-\boldsymbol{\Lambda}\boldsymbol{u}_i)^\top\Sigma^{-1}(\boldsymbol{y}_i-\boldsymbol{\mu}-\boldsymbol{\Lambda}\boldsymbol{u}_i)+\boldsymbol{u}_i^\top\boldsymbol{I}_q\boldsymbol{u}_i\Big]\Big\},\end{aligned}\tag{5.47}$$

If in the RHS of (5.47) we let $h(g_i)=g_i$, then it turns out that

$$\pi(g_i|\boldsymbol{y}_i,\boldsymbol{u}_i,\boldsymbol{\Psi})\propto g_i^{\frac{p+q}{2}+\tau-1}\exp\Big\{-\epsilon\frac{g_i}{2}-\frac{\chi}{2g_i}\Big\}.\tag{5.48}$$

Evidently,

$$G|(\boldsymbol{Y}=\boldsymbol{y}_i,\boldsymbol{U}=\boldsymbol{u}_i,\boldsymbol{\Psi})\sim\mathcal{GIG}\Big(\frac{p}{2}+\frac{q}{2}+\tau,\epsilon,\chi\Big),$$

where $\epsilon=[(\boldsymbol{y}_i-\boldsymbol{\mu}-\Lambda\boldsymbol{u}_i)^\top\Sigma^{-1}(\boldsymbol{y}_i-\boldsymbol{\mu}-\Lambda\boldsymbol{u}_i)+\boldsymbol{u}_i^\top\boldsymbol{u}_i+\chi]$.

Full conditional $\boldsymbol{U}|(\boldsymbol{Y}=\boldsymbol{y}_i,\cdots)$

By definition we have

$$\begin{aligned}\pi(\boldsymbol{u}_i|\boldsymbol{y}_i,g_i,\boldsymbol{\Psi}) =&\frac{f(\boldsymbol{y}_i,\boldsymbol{u}_i,g_i|\boldsymbol{\Psi})}{f(\boldsymbol{y}_i,g_i|\boldsymbol{\Psi})}\times\mathbb{I}_{\mathbb{R}^{q+}}(\boldsymbol{u}_i)\\ \propto& f(\boldsymbol{y}_i,\boldsymbol{u}_i,g_i|\boldsymbol{\Psi})\times\mathbb{I}_{\mathbb{R}^{q+}}(\boldsymbol{u}_i)\\ =& f(\boldsymbol{y}_i|\boldsymbol{u}_i,g_i|\boldsymbol{\Psi})f(\boldsymbol{u}_i|g_i)f(g_i|\boldsymbol{\theta})\times\mathbb{I}_{\mathbb{R}^{q+}}(\boldsymbol{u}_i)\\ \propto& f(\boldsymbol{y}_i|\boldsymbol{u}_i,g_i|\boldsymbol{\Psi})f(\boldsymbol{u}_i|g_i)\times\mathbb{I}_{\mathbb{R}^{q+}}(\boldsymbol{u}_i)\\ \propto&-\exp\Big\{\frac{h(g_i)}{2}\Big[(\boldsymbol{y}_i-\boldsymbol{\mu}-\boldsymbol{\Lambda}\boldsymbol{u}_i)^\top\Sigma^{-1}(\boldsymbol{y}_i-\boldsymbol{\mu}-\boldsymbol{\Lambda}\boldsymbol{u}_i)\\ &+\boldsymbol{u}_i^\top\boldsymbol{u}_i\Big]\Big\}\times\mathbb{I}_{\mathbb{R}^{q+}}(\boldsymbol{u}_i),\end{aligned}$$

where $\mathbb{I}_{\mathcal{S}}(x)$ is defined in (3.24). We use the fact that

$$\begin{aligned}&(\boldsymbol{y}_i-\boldsymbol{\mu}-\boldsymbol{\Lambda}\boldsymbol{u}_i)^\top\Sigma^{-1}(\boldsymbol{y}_i-\boldsymbol{\mu}-\boldsymbol{\Lambda}\boldsymbol{u}_i)+\boldsymbol{u}_i^\top\boldsymbol{u}_i\\ &=d(\boldsymbol{y}_i)+(\boldsymbol{u}_i-\boldsymbol{m})^\top\Delta^{-1}(\boldsymbol{u}_i-\boldsymbol{m}),\end{aligned}\tag{5.49}$$

where $d(\boldsymbol{y}_i)$, $\boldsymbol{m}$, and Δ are defined in Theorem 5.5.1. Hence,

$$\pi(\boldsymbol{u}_i|\boldsymbol{y}_i,g_i,\boldsymbol{\Psi})\propto\exp\Big\{-\frac{h(g_i)}{2}(\boldsymbol{u}_i-\boldsymbol{m})^\top\Delta^{-1}(\boldsymbol{u}_i-\boldsymbol{m})\Big\}\times\mathbb{I}_{\mathbb{R}^{q+}}(\boldsymbol{u}_i).\tag{5.50}$$

This means that

$$\boldsymbol{U}|(\boldsymbol{Y}=\boldsymbol{y}_i,G=g_i,\boldsymbol{\Psi})\sim\mathcal{TN}_q\Big(\boldsymbol{m},\frac{\Delta}{h(g_i)}\mathbf{0},\boldsymbol{\infty}\Big),\tag{5.51}$$

where $\mathcal{TN}_q(\cdot,\cdot,\cdot,\cdot)$ refers to the family of truncated q-dimensional Gaussian distributions introduced in Sub-section 3.11.

Bayesian inference for skew Student's t distribution

As it is seen from second column of Table 5.1, the skew Gaussian gamma distribution in which $G\sim\mathcal{G}(\alpha,\beta)$ specializes, for $\alpha=\beta=\nu/2$, to skew Student's t distribution with ν degrees of freedom for which a Bayesian paradigm has been developed by [Parisi and Liseo, 2018]. Herein, we compute the Bayesian estimators for parameters of $\text{SGG}_{p,q}(\boldsymbol{\mu},\Sigma,\boldsymbol{\Lambda},\boldsymbol{\theta})$ in which $\boldsymbol{\theta}=(\alpha,\beta)^\top$. To this end, first, we simulate from full conditionals $g_i|(\boldsymbol{\mu},\Sigma,\nu,\boldsymbol{u}_i,\boldsymbol{y}_i)$ and $\boldsymbol{u}_i|(\boldsymbol{\mu},\Sigma,\nu,g_i,\boldsymbol{y}_i)$ (for $i=1\cdots,n$) in order to construct the complete data $\{(\boldsymbol{y}_1,\boldsymbol{u}_1,g_1),\cdots,(\boldsymbol{y}_n,\boldsymbol{u}_n,g_n)\}$. It follows from the RHS of (5.46) that

$$\pi(g_i|\boldsymbol{y}_i,\boldsymbol{u}_i,\boldsymbol{\Psi})=\mathcal{G}(\alpha^*,\beta^*),\tag{5.52}$$

where $\alpha^*=(p+q)/2+\alpha$ and $\beta^*=2\beta+(\boldsymbol{y}_i-\boldsymbol{\mu}-\boldsymbol{\Lambda}\boldsymbol{u}_i)^\top\Sigma^{-1}(\boldsymbol{y}_i-\boldsymbol{\mu}-\boldsymbol{\Lambda}\boldsymbol{u}_i)/2+\boldsymbol{u}_i^\top\boldsymbol{u}_i/2$. Moreover, using $h(g_i)=g_i$, it turns out from (5.51) that

$$\pi(\boldsymbol{u}_i\Big|\boldsymbol{y}_i,g_i,\boldsymbol{\Psi})=\mathcal{TN}_q\Big(\boldsymbol{m}_i,\frac{\Delta}{g_i},\mathbf{0},\boldsymbol{\infty}\Big),\tag{5.53}$$

where $\boldsymbol{m}_i=\Lambda^\top\Omega^{-1}(\boldsymbol{y}_i-\boldsymbol{\mu})$, $\Delta=\boldsymbol{I}_q-\Lambda^\top\Omega^{-1}\Lambda$, and $\Omega=\Sigma+\Lambda\Lambda^\top$. Constructing the complete data, we proceed to generate from the remainder full conditionals $\boldsymbol{\mu}|(\underline{\boldsymbol{y}},\underline{\boldsymbol{u}},\boldsymbol{g},\boldsymbol{\Psi}_{(-\boldsymbol{\mu})})$, $\Sigma|(\underline{\boldsymbol{y}},\underline{\boldsymbol{u}},\boldsymbol{g},\boldsymbol{\Psi}_{(-\Sigma)})$, $\Lambda|(\underline{\boldsymbol{y}},\underline{\boldsymbol{u}},\boldsymbol{g},\boldsymbol{\Psi}_{(-\Lambda)})$, $\alpha|(\underline{\boldsymbol{y}},\underline{\boldsymbol{u}},\boldsymbol{g},\boldsymbol{\Psi}_{(-\alpha)})$, and $\beta|(\underline{\boldsymbol{y}},\underline{\boldsymbol{u}},\boldsymbol{g},\boldsymbol{\Psi}_{(-\beta)})$ in which $\underline{\boldsymbol{y}}=(\boldsymbol{y}_1^\top,\cdots,\boldsymbol{y}_n^\top)^\top$, $\underline{\boldsymbol{u}}=(\boldsymbol{u}_1^\top,\cdots,\boldsymbol{u}_n^\top)^\top$, and $\boldsymbol{y}=(g_1,\cdots,g_n)^\top$. First, we notice that

$$\begin{aligned}\pi(\boldsymbol{\Psi}|\underline{\boldsymbol{y}},\underline{\boldsymbol{u}},\boldsymbol{g})&\propto L_c(\boldsymbol{\Psi}|\underline{\boldsymbol{y}},\underline{\boldsymbol{u}},\boldsymbol{g})\pi(\boldsymbol{\theta})\\ &=L_c(\boldsymbol{\Psi}|\underline{\boldsymbol{y}},\underline{\boldsymbol{u}},\boldsymbol{g})\pi(\boldsymbol{\mu})\pi(\Sigma)\pi(\Lambda)\pi(\alpha)\pi(\beta),\end{aligned}\tag{5.54}$$

where, as usual, we assumed that priors are independent and furthermore

$$\begin{aligned}
&\pi(\boldsymbol{\mu}) \sim \mathcal{N}_p(\mathcal{M}_0, \mathcal{S}_0),\\
&\pi(\Sigma) \sim \mathcal{IW}(\mathcal{D}_0, \nu_0),\\
&\pi(\Lambda) \sim \mathcal{N}_q(\mathcal{L}_0, \mathcal{P}_0),\\
&\pi(\alpha) \sim \mathcal{G}(a_0, a_0),\\
&\pi(\beta) \sim \mathcal{G}(b_0, b_0).
\end{aligned}$$

As it is seen, for priors of α and β, we assumed a gamma distribution with common shape and rate parameters. The results below give the full conditionals needed for implementing the Bayesian paradigm.

- **full conditional of $\boldsymbol{\mu}\big|(\underline{\boldsymbol{y}}, \underline{\boldsymbol{u}}, \boldsymbol{g}, \boldsymbol{\Psi}_{(-\boldsymbol{\mu})})$:** By considering a conjugate prior, that is $\pi(\boldsymbol{\mu}) \sim \mathcal{N}_p(\mathcal{M}_0, \mathcal{S}_0)$, more algebra shows that

$$\pi(\boldsymbol{\mu}|\underline{\boldsymbol{y}}, \underline{\boldsymbol{u}}, \boldsymbol{g}, \boldsymbol{\Psi}_{(-\boldsymbol{\mu})}) = \mathcal{N}_p\Big(\boldsymbol{\mu}\Big|\mathcal{Q}_0\Big[\mathcal{S}_0^{-1}\mathcal{M}_0 + \Sigma^{-1}\sum_{i=1}^{n} g_i(\boldsymbol{y}_i - \Lambda\boldsymbol{u}_i)\Big], \mathcal{Q}_0\Big),$$

  where

$$\mathcal{Q}_0 = \Big[\mathcal{S}_0^{-1} + \Sigma^{-1}\sum_{i=1}^{n} g_i\Big]^{-1}.$$

- **full conditional of $\Sigma\big|(\underline{\boldsymbol{y}}, \underline{\boldsymbol{u}}, \boldsymbol{g}, \boldsymbol{\Psi}_{(-\Sigma)})$:** By considering a conjugate prior, that is $\Sigma \sim \mathcal{IW}(\mathcal{D}_0, \nu_0)$, we can write

$$\pi(\Sigma|\underline{\boldsymbol{y}}, \underline{\boldsymbol{u}}, \boldsymbol{g}, \boldsymbol{\Psi}_{(-\Sigma)}) = \mathcal{IW}(\cdot|\mathcal{R}_0, \nu_0 + n),$$

  where $\mathcal{R}_0 = \mathcal{D}_0 + \sum_{i=1}^{n} g_i(\boldsymbol{y}_i - \boldsymbol{\mu} - \Lambda\boldsymbol{u}_i)(\boldsymbol{y}_i - \boldsymbol{\mu} - \Lambda\boldsymbol{u}_i)^\top$.

- **full conditional of $\Lambda\big|(\underline{\boldsymbol{y}}, \underline{\boldsymbol{u}}, \boldsymbol{g}, \boldsymbol{\Psi}_{(-\Lambda)})$:** Although a possible choice for prior of Λ is a matrix normal normal distribution, but we prefer rather to sample from this full conditional in a slightly different method. We perform this by sampling from each column of Λ separately. Let $\Lambda_{(c)}$ denote the c-th column (for $c = 1, \ldots, q$) of matrix Λ and $\Lambda_{(-c)}$ is the matrix Λ when its c-th column is removed. Likewise, assume $u_{i(c)}$ denote the c-th element of vector $\boldsymbol{u}_i$ and $\boldsymbol{u}_{i(-c)}$ is vector $\boldsymbol{u}_i$ when its c-th element is removed. Suppose $\boldsymbol{\xi}_i = \boldsymbol{y}_i - \boldsymbol{\mu} - \Lambda_{(-c)}\boldsymbol{u}_{i(-c)}$, then it can be seen that

$$\begin{aligned}
&(\boldsymbol{y}_i - \boldsymbol{\mu} - \Lambda\boldsymbol{u}_i)^\top\Sigma^{-1}(\boldsymbol{y}_i - \boldsymbol{\mu} - \Lambda\boldsymbol{u}_i)\\
&= (\Lambda_{(c)}u_{i(c)} - \boldsymbol{\xi}_i)^\top\Sigma^{-1}(\Lambda_{(c)}u_{i(c)} - \boldsymbol{\xi}_i).
\end{aligned}$$

  Replacing above identity in the RHS of (5.40) and simultaneously assuming a multivariate Gaussian conjugate prior for $\Lambda_{(c)}$, that is $\Lambda_{(c)} \sim \mathcal{N}_p(\mathcal{L}_{0(c)}, \mathcal{P}_{0(c)})$, it turns out that

$$\Lambda_{(c)}\big|(\underline{\boldsymbol{y}}, \underline{\boldsymbol{u}}, \boldsymbol{g}, \Psi_{(-\Lambda_{0(c)})}) \sim \mathcal{N}_p(\mathcal{V}, \mathcal{W}),$$

  where, for $c = 1, \ldots, q$, we have

$$\begin{aligned}
\mathcal{V} &= \mathcal{W}\Big[\mathcal{P}_{0(c)}^{-1}\mathcal{L}_{0(c)} + \Sigma^{-1}\sum_{i=1}^{n} g_i u_{i(c)}\boldsymbol{\xi}_i\Big],\\
\mathcal{W} &= \Big[\mathcal{P}_{0(c)}^{-1} + \Sigma^{-1}\sum_{i=1}^{n} g_i u_{i(c)}^2\Big]^{-1}.
\end{aligned}$$

  After simulating sequence $\{\Lambda_{(1)}, \Lambda_{(2)}, \ldots, \Lambda_{(q)}\}$, the generated skewness matrix Λ is reconstructed as $\Lambda = [\Lambda_{(1)}|\Lambda_{(2)}|\ldots|\Lambda_{(q)}]$.

- **full conditionals $\alpha|(\underline{\boldsymbol{y}}, \underline{\boldsymbol{u}}, \boldsymbol{g}, \boldsymbol{\Psi}_{(-\alpha)})$ and $\beta|(\underline{\boldsymbol{y}}, \underline{\boldsymbol{u}}, \boldsymbol{g}, \boldsymbol{\Psi}_{(-\alpha)})$:** We have

$$\boldsymbol{\theta}|(\underline{\boldsymbol{y}}, \underline{\boldsymbol{u}}, \boldsymbol{g}, \boldsymbol{\Psi}_{(-\boldsymbol{\theta})}) \propto L_c(\boldsymbol{\Psi}|\underline{\boldsymbol{y}}, \underline{\boldsymbol{u}}, \boldsymbol{g})\pi(\boldsymbol{\theta}), \tag{5.55}$$

By assuming independence between marginals of $\pi(\boldsymbol{\theta})$, it follows from RHS of in (5.55) that

$$\begin{aligned}\pi(\alpha|\underline{\boldsymbol{y}}, \underline{\boldsymbol{u}}, \boldsymbol{g}, \boldsymbol{\Psi}_{(-\alpha)}) &\propto \prod_{i=1}^{n} f(g_i|\boldsymbol{\theta})\pi(\alpha) \\ &= \frac{\beta^{n\alpha}\alpha_0^{\alpha_0}}{[\Gamma(\alpha)]^n\Gamma(\alpha_0)}\Big(\prod_{i=1}^{n} g_i\Big)^{\alpha-1}\alpha^{\alpha_0-1}\exp\{-\alpha_0\alpha\}. \end{aligned}\tag{5.56}$$

It can be easily seen that $\pi(\alpha|\underline{\boldsymbol{y}}, \underline{\boldsymbol{u}}, \boldsymbol{g}, \boldsymbol{\Psi}_{(-\alpha)})$ is log-concave for $\alpha_0 \geq 1$, and hence the ARS approach is suggested for sampling from this full conditional. If one would like to employ the MH approach, one choice for candidate may be $\mathcal{G}(\alpha_0, \alpha_0)$. This yields the the acceptance ratio as

$$\begin{aligned}p(\alpha_k, \alpha_{k+1}) = \min\Bigg\{1,\ &\Big(\frac{\alpha_{k+1}}{\alpha_k}\Big)^{\alpha_0-1}\Big[\frac{\Gamma(\alpha_k)}{\Gamma(\alpha_{k+1})}\Big]^n\beta^{n(\alpha_{k+1}-\alpha_k)} \\ &\times \exp\Big\{\alpha_0(\alpha_k-\alpha_{k+1})\Big\}\Big(\prod_{i=1}^{n} g_i\Big)^{\alpha_{k+1}-\alpha_k}\Bigg\}. \end{aligned}\tag{5.57}$$

Similarly, assuming $\pi(\beta) \sim \mathcal{G}(\beta_0, \beta_0)$. We have

$$\begin{aligned}\pi(\beta|\underline{\boldsymbol{y}}, \underline{\boldsymbol{u}}, \boldsymbol{g}, \boldsymbol{\Psi}_{(-\beta)}) &\propto \prod_{i=1}^{n} f(g_i|\boldsymbol{\theta})\pi(\beta) \\ &\propto \beta^{n\alpha+\beta_0-1}\exp\Big\{-\beta\big(\beta_0+\sum_{i=1}^{n} g_i\big)\Big\}. \end{aligned}\tag{5.58}$$

From the RHS of (5.58), it turns out that $\beta|(\underline{\boldsymbol{y}}, \underline{\boldsymbol{u}}, \boldsymbol{g}, \boldsymbol{\Psi}_{(-\beta)}) \sim \mathcal{G}(\beta_0 + n\alpha, \beta_0 + \sum_{i=1}^{n} g_i)$.

In what follows, Algorithm 18 describes how to compute the Bayesian estimator for parameters of $\text{SGG}_{p,q}(\boldsymbol{\mu}, \Sigma, \Lambda, \boldsymbol{\theta})$.

Example 5-7

Herein, we suppose a sample of $n = 500$ observations are drawn from $\boldsymbol{Y} \sim \text{SGG}_{2,2}(\boldsymbol{\mu}, \Sigma, \boldsymbol{\Lambda}, \boldsymbol{\theta})$ with parameters

$$\boldsymbol{\mu} = \begin{bmatrix} 2 \\ -2 \end{bmatrix},\ \Sigma = \begin{bmatrix} 0.4 & 0 \\ 0 & 0.4 \end{bmatrix},\ \boldsymbol{\Lambda} = \begin{bmatrix} -2 & 2 \\ 3 & 0 \end{bmatrix},\ \boldsymbol{\theta} = \begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \begin{bmatrix} 5 \\ 5 \end{bmatrix}.$$

We follow the steps of Algorithm 7 to obtain Bayesian estimator of $\boldsymbol{\Psi}$. ■

The pertaining `R` code for implementing example above is given as follows. As it may seen from `R` code, we hyperparameters as $\mathcal{M}_0 = (0, 0)^\top$, $\mathcal{S}_0 = \mathcal{D}_0 = \mathcal{P}_0 = \boldsymbol{I}_p$, $\nu_0 = 2$, $\mathcal{L}_0 = (0, 0)^\top$, $\beta_0 = 2$, and $\alpha_0 = 2$. The output of the Gibbs sampler is displayed in Figure 5.6.

```
R > f.a <- function(x, beta, alpha0, y)
+    {
+    n = length(y)
+    n*x*log(beta) - n*lgamma(x) + (x-1)*sum(log(y)) +
+      (alpha0-1)*log(x) - alpha0*x
+    }
```

Algorithm 18 Bayesian inference for SGG

Read N, M, and determine quantities $\mathcal{M}_0$, $\mathcal{S}_0$, $\mathcal{D}_0$, ν_0, $\mathcal{L}_0$, $\mathcal{P}_0$, a_0, and
$\boldsymbol{\theta}_0 = (a_0, b_0)^\top$;
Set $t \leftarrow 0$;
Set $\boldsymbol{\mu}^{(0)} \leftarrow \mathcal{M}^0$, $\Sigma^{(0)} \leftarrow 1/n \sum_{i=1}^{n} (\boldsymbol{y}_i - \boldsymbol{\mu}^{(0)})(\boldsymbol{y}_i - \boldsymbol{\mu}^{(0)})^\top$, $\nu^{(0)} = 2$,
and set $\boldsymbol{\Psi}^{(0)} = (\boldsymbol{\mu}^{(0)}, \Sigma^{(0)}, \Lambda^{(0)}, \boldsymbol{\theta}^{(0)})$;
while $t \leq N$ **do**
    Set $i = 1$;
    while $i \leq n$ **do**
        Simulate g_i from $\mathcal{G}(a, b)$ where $a = \alpha^{(t)} + (p+q)/2$ and $b =$
        $\Big[2\beta^{(t)} + (\boldsymbol{y}_i - \boldsymbol{\mu}^{(t)} - \Lambda^{(t)}\boldsymbol{u}_i)^\top [\Sigma^{(t)}]^{-1} (\boldsymbol{y}_i - \boldsymbol{\mu}^{(t)} - \Lambda^{(t)}\boldsymbol{u}_i)\Big]/2 + \boldsymbol{u}_i^\top \boldsymbol{u}_i/2$;
        Simulate $\boldsymbol{u}_i$ from $\mathcal{HN}_q(\boldsymbol{m}_i^{(t)}, \frac{\Delta^{(t)}}{g_i})$ where $\boldsymbol{m}_i^{(t)} = [\Lambda^{(t)}]^\top [\Omega^{(t)}]^{-1} (\boldsymbol{y}_i -$
        $\boldsymbol{\mu})$ with $\Omega^{(t)} = \Sigma^{(t)} + \Lambda^{(t)}[\Lambda^{(t)}]^\top$ and $\Delta^{(t)} = \boldsymbol{I}_q - [\Lambda^{(t)}]^\top [\Omega^{(t)}]^{-1} \Lambda^{(t)}$;
        Set $i \leftarrow i + 1$;
    end
    Generate $\boldsymbol{\mu}^{(t+1)} \sim \mathcal{N}_p\Big(\mathcal{Q}_0 \Big[\mathcal{S}_0^{-1}\mathcal{M}_0 + [\Sigma^{(t)}]^{-1} \sum_{i=1}^{n} g_i (\boldsymbol{y}_i - \Lambda^{(t)}\boldsymbol{u}_i)\Big], \mathcal{Q}_0\Big)$
    where $\mathcal{Q}_0 = \Big[\mathcal{S}_0^{-1} + [\Sigma^{(t)}]^{-1} \sum_{i=1}^{n} g_i\Big]^{-1}$;
    Generate $\Sigma^{(t+1)}$ from $\mathcal{IW}(\mathcal{R}_0, \nu_0 + n)$ where $\mathcal{R}_0 = \mathcal{D}_0 + \sum_{i=1}^{n} g_i (\boldsymbol{y}_i -$
    $\boldsymbol{\mu}^{(t+1)})(\boldsymbol{y}_i - \boldsymbol{\mu}^{(t+1)})^\top$;
    Set $j = 1$;
    while $j \leq q$ **do**
        Set $\mathcal{W}^{-1} = [\mathcal{P}_{0(j)}^{-1} + [\Sigma^{-1}]^{(t+1)} \sum_{i=1}^{n} g_i u_{i(j)}^2]$, $\mathcal{V} = \mathcal{W}[\mathcal{P}_{0(j)}^{-1}\mathcal{L}_{0(j)} +$
        $[\Sigma^{(t+1)}]^{-1} \sum_{i=1}^{n} g_i u_{i(j)} \boldsymbol{\xi}_i]$, and $\boldsymbol{\xi}_i = \boldsymbol{y}_i - \boldsymbol{\mu}^{(t+1)} - \Lambda_{(-j)}^{(t)} \boldsymbol{u}_{i(-j)}$;
        Generate $\Lambda_{(j)} \sim \mathcal{N}_p(\mathcal{V}, \mathcal{W})$;
    end
    Construct $\Lambda^{(t+1)} = [\Lambda_{(1)} | \Lambda_{(2)} | \ldots | \Lambda_{(q)}]$;
    Use the ARS-within-Gibbs sampling technique for simulating $\alpha^{(t+1)}$ while
    $\log \pi(\alpha | \underline{\boldsymbol{y}}, \underline{\boldsymbol{u}}, \boldsymbol{g}, \boldsymbol{\Psi}_{(-\alpha)}) \propto n\alpha \log \beta - n \log \Gamma(\alpha) + (\alpha - 1) \sum_{i=1}^{n} \log g_i +$
    $(\alpha_0 - 1) \log \alpha - \alpha_0 \alpha$
    Simulate $\beta^{(t+1)}$ from full conditional $\beta | (\underline{\boldsymbol{y}}, \underline{\boldsymbol{u}}, \boldsymbol{g}, \boldsymbol{\Psi}_{(-\beta)}) \sim \mathcal{G}(\beta_0 +$
    $n\alpha^{(t+1)}, \beta_0 + \sum_{i=1}^{n} g_i)$;
end
Set $\boldsymbol{\Psi}^{(t+1)} \leftarrow (\boldsymbol{\mu}^{(t+1)}, \Sigma^{(t+1)}, \Lambda^{(t+1)}, \boldsymbol{\theta}^{(t+1)})$ and $t \leftarrow t + 1$;
end
Sequence $\{\boldsymbol{\Psi}^{(M)}, \boldsymbol{\Psi}^{(M+1)}, \cdots, \boldsymbol{\Psi}^{(N)}\}$ is a sample of size $N - M + 1$ from
posterior $\pi(\boldsymbol{\Psi} | \underline{\boldsymbol{y}}, \underline{\boldsymbol{u}}, \boldsymbol{g})$;
The Bayesian estimator of $\boldsymbol{\Psi}$ is given by $\frac{1}{N-M+1} \sum_{t=1}^{N-M+1} \boldsymbol{\Psi}^{(t)}$.

```
R > fprim.a <- function(x, beta, alpha0, y)
+    {
+    n = length(y)
+    n*log(beta) - n*digamma(x) + sum(log(y)) + (alpha0-1)/x - alpha0
+    }
R> rssg <- function(n, alpha, beta, Mu, Sigma, Lambda)
+  {
+   Dim <- length(Mu)
+   Q <- length(Lambda[1,])
+   Y <- matrix(NA, nrow = n, ncol = Dim)
+    for (i in 1:n)
+    {
+     Z <- rgamma(1, shape = alpha, rate = beta)
+     X <- mvrnorm(n= 1, mu = rep(0, Dim), Sigma = Sigma )
+     u <- abs( rnorm(Q) )
+     Y[i, ] <- Mu + Lambda%*%u/sqrt(Z) + X/sqrt(Z)
+    }
+   Y
+  }
R> set.seed(20240520)
R> library(MASS); library(tmvtnorm); library(ars)
R> n <- 500; alpha <- 5; beta <- 5; Sigma <- matrix( c(.4,0,0,.4), 2, 2);
R> Mu <- c(2, -2); Lambda <- matrix( c(-2,3,2,0), 2, 2)
R> Y <- rssg(n, alpha, beta, Mu, Sigma, Lambda );
R> g <- rgamma(n, shape = alpha, rate = beta);
R> Q <- dim(Lambda)[2]
R> p <- length(Mu)
R> T <- matrix( abs(rnorm(n*Q, mean = 0, sd = 1)), nrow = n, ncol = Q)
R> N <-10000; M <-5000; Mu0c <-rep(0, p); beta0 <-2; alpha0 <-2; nu0 <-2;
R> S0Mu <- diag(p); M0Mu <- rep(0, p);
R> D0 <- S0c <- P0c <-diag(p);
R> Psi <-matrix(0, nrow = N, ncol = p + p^2 + p*Q + 2 )
R> for(k in 1:N)
+ {
+ Sigmainv <- solve(Sigma)
+ Omega <- Sigma + tcrossprod( Lambda )
+ Omegainv <- solve(Omega)
+ Delta <- diag(Q) - (t(Lambda)%*%Omegainv%*%Lambda)
+   for(i in 1:n)
+   {
+    rate <- mahalanobis(x = Y[i, ], center = Mu + Lambda%*%T[i, ], cov = Sigma )/2 + (T[i,]
    %*%T[i,])[1]/2 + beta
+    g[i] <- rgamma(1, shape = alpha + (p + Q)/2, rate = rate )
+    T[i,] <- rtmvnorm(1, mean = c(t(Lambda)%*%Omegainv%*%c( Y[i, ] - Mu )), sigma = Delta/g
    [i],
+    lower = rep(0, length = Q), upper = rep( Inf, length = Q), algorithm = c("rejection") )
+   }
+ Sum_g <- sum(g)
+ M_sum <- rowSums( sapply(1:n, function(i) g[i]*c( Y[i, ] -
+          Lambda%*%T[i, ]) ) )
+ Q0 <- solve( solve( S0Mu ) + Sigmainv*Sum_g )
+ Mu <- c( mvrnorm(1, mu = Q0%*%( solve( S0Mu )%*%M0Mu + Sigmainv%*%M_sum ),
+          Sigma = Q0) );
+ Psi[k, 1:p] <- Mu
+ R <- matrix(0, nrow = p, ncol = p)
+   for(i in 1:n)
+   {
+    R <- R + g[i]*c( Y[i, ] - Mu - Lambda%*%T[i, ])%*%t( c( Y[i, ] - Mu -
+          Lambda%*%T[i, ])
+   )}
+   Sigma <- solve( rWishart(1, df = n+nu0+1, Sigma = solve(D0 + R) )[,,1] )
+   Psi[k, (p + 1):(p + p^2)] <- as.numeric(Sigma)
+     for(j in 1:Q)
```

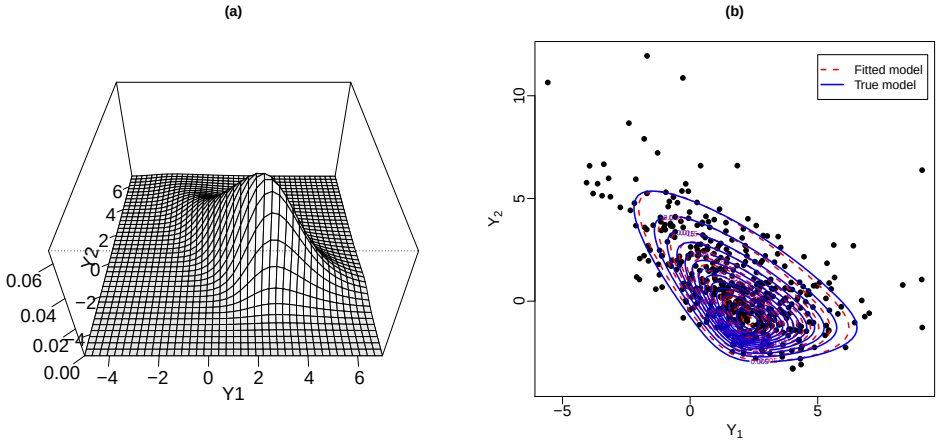


Figure 5.7: The PDF of $\text{SGG}_{2,2}(\boldsymbol{\mu}, \Sigma, \Lambda, \boldsymbol{\theta})$. (a): true model and (b): true and fitted contour plots.

```
+     {
+      Sumgt <- sum( g*T[, j]^2 )
+      Sc <- solve( solve( S0c ) + Sigmainv*Sumgt )
+       if(Q >=3)
+       {
+        Mc_sum <- rowSums( sapply(1:n,function(i) g[i]*T[i, j]*c( Y[i, ] -
+                Mu -  Lambda[,-j]%*%T[i, -j]) ) )
+        }else{
+        Mc_sum <- rowSums( sapply(1:n,
+        function(i) g[i]*T[i, j]*c( Y[i, ] - Mu - Lambda[,-j]*T[i, -j]) ) )
+       }
+      Mc <- Sc%*%( solve( S0c )%*%Mu0c + Sigmainv%*%Mc_sum )
+      Lambda[, j] <- mvrnorm(n= 1, mu = Mc , Sigma = Sc )
+     }
+ Psi[k, (p + p^2 + 1):(p + p^2 + p*Q)] <- as.numeric(Lambda)
+ alpha <- ars(1, f.a, fprim.a, x=c(0.5, 4, 10, 30), m=4, lb=TRUE, xlb=0,
+           ub=FALSE, beta=beta, alpha0=alpha0, y=g )
+ beta <- rgamma( 1, n*alpha + beta0, beta0 + sum(g) )
# b0 <- 1/(3-(cov(Y)%*%solve(Sigma))[1,1])
+ Psi[k, (p + p^2 + p*Q + 1)] <- alpha
+ Psi[k, (p + p^2 + p*Q + 2)] <- beta
+ }
R > Psi.hat <- apply(Psi[(N-M+1):N,], 2, mean)
R > Mu <- Psi.hat[1:p]
R > Sigma <- Psi.hat[(p+1):(p + p^2)]
R > Lambda <- Psi.hat[(p + p^2 + 1):(p + p^2 + p*Q)]
R > alpha <- Psi.hat[p + p^2 + p*Q + 1]
R > beta <- Psi.hat[p + p^2 + p*Q + 2]
```

5.6 Gibbs sampling in Bayesian paradigm with more than two latent variables: Model-based Clustering

The aim of clustering technique is to divide a non-homogeneous population into K numbers of homogeneous sub-populations. Obviously, members of a non-homogeneous population have dissimilarities while the clustered or partitioned sub-populations have more similarities. Herein, the term "similarity" may refer to some measurable attribute or criterion such as Euclidean distance. If two observations are similar, then they are close together (high degree of similarity)

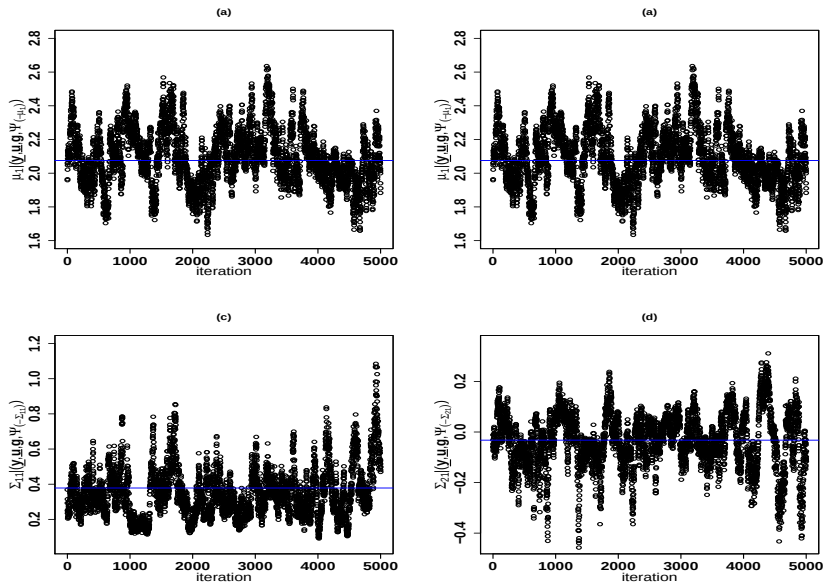


Figure 5.8: Scatterplot of 5000 samples generated from full conditionals involved in Bayesian paradigm of $\text{SGG}_{2,2}(\boldsymbol{\mu}, \Sigma, \Lambda, \boldsymbol{\theta})$.

and otherwise, two observations are dissimilar (low degree of similarity).

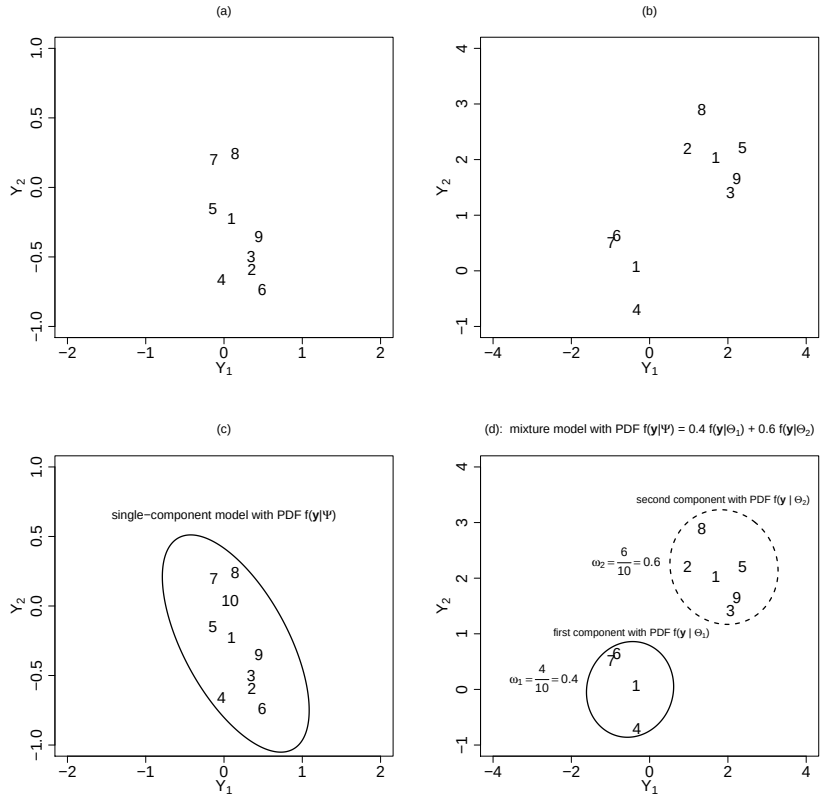


Figure 5.9: (a): Ten data points that can constitute a cluster, (b): Ten data points that should be clustered into two groups, (c): Ten data points in part (a) that can be modelled through a single-component (homogeneous) model with PDF $f(\boldsymbol{y}|\boldsymbol{\Theta})$, and (d): Ten data points in part (b) that can be modelled through a two-component (non-homogeneous) mixture model with PDF $0.4f(\boldsymbol{y}|\boldsymbol{\Theta}_1) + 0.6f(\boldsymbol{y}|\boldsymbol{\Theta}_2)$.

Figure 5.9(a) displays a set of ten observations, labeled with numbers $1, 2, \cdots, 10$, that are similar and so can constitute a cluster. On the other hand, consider the situation in which ten observations are non-homogeneous or dissimilar as shown by Figure 5.9(b). Herein, ten observations with labels $\{7, 6, 1, 4\}$ are similar together, but quite dissimilar with observations whose label are $\{2, 8, 1, 3, 9, 5\}$. So, observations in Figure 5.9(b) should be groped into two clusters. We emphasize that the objective of clustering technique is to determine the label of each observation. Furthermore, sometimes the yardstick for measuring the similarity between observations may not be a quantitative tool such as the Euclidean distance. The clustering technique proceeds to divide observations into K groups such that the observations within each group have the maximum degree of similarity while the observations between groups have the maximum degree of dissimilarity. Sometimes the clustering technique is known as semi-supervised or unsupervised machine learning since label of observations are not known and the only available information are observed data McNicholas [2016].

The model-based clustering (or non-hierarchical clustering) uses a finite mixture of multivariate statistical distributions of the same family for the purpose of clustering. Hence, the model-based clustering and finite mixture models are often used interchangeably in the literature. As mentioned above, the aim of clustering is to determine the label or the origin of an observation is coming from. It is worth to note that the concepts "origin", "cluster", "label", and "component" may be used interchangeably. For more clarifications, Figure 5.9(c) shows a set of ten observations with perfect similarity and, within the model-based clustering framework, we assume that these ten observations follow independently the same multivariate distribution. On the other hand, Figure 5.9(d) shows another set of ten observations with two clusters, and hence within the model-based clustering framework, we prefer to assume these data points are coming from two different multivariate statistical distribution of the same family. In other words, for the first and second clusters we consider two multivariate statistical distributions of the same family whose PDFs are $f(\boldsymbol{y}|\boldsymbol{\Theta}_1)$ and $f(\boldsymbol{y}|\boldsymbol{\Theta}_2)$, respectively. The corresponding weights are 0.4 and 0.6 and hence the distribution of ten observations is a mixture of two distributions fitted to two clusters whose PDF is

$$g(\boldsymbol{y}|\boldsymbol{\Psi}) = 0.4f(\boldsymbol{y}|\boldsymbol{\Theta}_1) + 0.6f(\boldsymbol{y}|\boldsymbol{\Theta}_2). \tag{5.59}$$

In (5.59), the vector $\boldsymbol{\omega} = (0.40, 0.60)^\top$, is known as the weight vector that means 40% of observations in Figure 5.9(d) are coming from the first component and the remaining 60% are coming from the second component. Overall, for p-dimensional random vector $\boldsymbol{Y}$, the PDF of a K-component mixture model is given by

$$g(\boldsymbol{y}|\boldsymbol{\Psi}) = \sum_{k=1}^{\mathrm{K}} \omega_k f(\boldsymbol{y}|\boldsymbol{\Theta}_k), \tag{5.60}$$

where $\boldsymbol{\Psi} = (\boldsymbol{\omega}^\top, \boldsymbol{\Theta}_1, \cdots, \boldsymbol{\Theta}_{\mathrm{K}})$ is the parameter space in which $\boldsymbol{\omega} = (\omega_1, \ldots, \omega_{\mathrm{K}})^\top$ is vector of mixing parameters such that $\sum_{k=1}^{\mathrm{K}} \omega_k = 1$. In (5.60), $f_{\boldsymbol{Y}}(\boldsymbol{y}|\boldsymbol{\Theta}_k)$ is the PDF of k-th component and hence the PDF $g(\boldsymbol{y}|\boldsymbol{\Psi})$ is proper since

$$\int_{\mathbb{R}^p} g(\boldsymbol{y}|\boldsymbol{\Psi}) d\boldsymbol{y} = \sum_{k=1}^{\mathrm{K}} \omega_k \int_{\mathbb{R}^p} f(\boldsymbol{y}|\boldsymbol{\Theta}_k) d\boldsymbol{y} = \sum_{k=1}^{\mathrm{K}} \omega_k = 1.$$

The weight parameter $\boldsymbol{\omega}$ provides the chance of observing sample from components. Sometimes, representation (5.60) is called the *finite mixture model* since the quantity K or number of clusters is assumed to be finite. In next two subsections, we proceed to discuss about generating realization from a finite mixture model and how to determine quantity K.

5.6.1 Generating from Gaussian finite mixture model

For generating n realizations from a K-component mixture model, we may follow one of the two following approaches. In the first method, one can generate independently $n_k = \lceil n \times \omega_k \rceil$ (for $k = 1, \cdots, \mathrm{K}-1$) realizations from PDF $f(\boldsymbol{y}|\boldsymbol{\Theta}_k)$ in which $\lceil n \times \omega_k \rceil$ denotes the smallest integer value equal or greater than $n \times \omega_k$, and finally $n_{\mathrm{K}} = n - \sum_{k=1}^{\mathrm{K}-1} n_k$ realizations are drawn independently from PDF $f(\boldsymbol{y}|\boldsymbol{\Theta}_{\mathrm{K}})$. The following pseudo code describes how to draw a sample of size n from a K-component Gaussian mixture model under the second method.

Algorithm 19 Simulating from K-component Gaussian mixture model.

Read n, K (number of components), and $\boldsymbol{\Psi} = (\boldsymbol{\omega}, \boldsymbol{\Theta}_1, \cdots, \boldsymbol{\Theta}_{\mathrm{K}})$;
Set $\boldsymbol{Y}$ a vector of length n;
Set $i = 1$;
while $i \leq n$ **do**
    Sample $\boldsymbol{u} \sim \mathcal{MUL}(1, \boldsymbol{\omega})$;
    Set $k_{max} = \{k|\boldsymbol{u}[k] = 1\}$ where $\boldsymbol{u}[k]$ is the kth element of vector $\boldsymbol{u}$ for
    $k = 1, \cdots, \mathrm{K}$;
    Sample $\boldsymbol{y}$ from PDF $f(\cdot|\boldsymbol{\Theta}_{k_{max}})$
    Set $\boldsymbol{Y}[i] \leftarrow \boldsymbol{y}$;
    Set $i \leftarrow i + 1$;
end
Accept $\boldsymbol{Y}$ as sample of size n from K-component mixture model.

Example 5-8

Suppose we are interested in generating a sample of $n = 300$ observations from observations are coming from a three-component bivariate Gaussian mixture model with PDF given by

$$g(\boldsymbol{y}|\boldsymbol{\Psi}) = \frac{1}{3}\mathcal{N}_2(\boldsymbol{\mu}_1, \Sigma_1) + \frac{1}{3}\mathcal{N}_2(\boldsymbol{\mu}_2, \Sigma_2) + \frac{1}{3}\mathcal{N}_2(\boldsymbol{\mu}_3, \Sigma_3), \tag{5.61}$$

where $\boldsymbol{\Psi} = (\omega_1, \omega_2, \omega_3, \Theta_1, \Theta_2, \Theta_3)$ is whole parameter vector whose elements are

$$\begin{aligned}
\Theta_1 = \omega_1 = \frac{1}{3}, \boldsymbol{\mu}_1 = (-3, -3)^\top, \Sigma_1 = \begin{bmatrix} 1 & 0.5 \\ 0.5 & 1 \end{bmatrix}, \\
\Theta_2 = \omega_2 = \frac{1}{3}, \boldsymbol{\mu}_2 = (0, 0)^\top, \quad \Sigma_2 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \\
\Theta_3 = \omega_3 = \frac{1}{3}, \boldsymbol{\mu}_3 = (3, 3)^\top, \quad \Sigma_3 = \begin{bmatrix} 1 & -0.25 \\ -0.25 & 1 \end{bmatrix}.
\end{aligned} \tag{5.62}$$

Based on Algorithm 19, we use the following `R` code to generate from three-component bivariate Gaussian mixture model with whole parameter given as above. Figure 5.10(a) displays the corresponding scatterplot. ■

```
R> library(MASS) # for generating realization from multivariate Gaussian
R> set.seed(20240713)
R> K <- 3
R> omega <- rep(1/3, K)
R> Mu1 <- rep(-3, p); Mu2 <- rep(0, p); Mu3 <- c(3, -3)
R> Sigma1 <- diag(0.5, p) + matrix(.5, p, p)
R> Sigma2 <- diag(1, p)
R> Sigma3 <- diag(1.5, p) + matrix(-0.5, p, p)
R> Sigma2 <-matrix(c(1, 0, 0, 1), 2, 2);
R> Sigma3 <-matrix(c(1,-0.25,-0.25,1), 2, 2);
R> Mu <- list(Mu1, Mu2, Mu3)
```

```
R> Sigma <- list(Sigma1, Sigma2, Sigma3)
R> rmixnorm <- function(n, Mu, Sigma, omega)
+ {
+ p <- length( Mu[[1]] )
+ label <- rep(0, n)
+ Y <- matrix(0, nrow = n, ncol = p)
+  for(i in 1:n)
+  {
+  r_MUL <- rmultinom(n = 1, size = 1, omega)
+  k <- apply(r_MUL, 2, which.max)
+  label[i] <- k
+  Y[i, ] <- mvrnorm(n = 1, mu = Mu[[k]], Sigma = Sigma[[k]])
+  }
+ ls <- list( "sample" = Y, "label" = label )
+ return(ls)
+ }
R> Y <- rmixnorm(n, Mu, Sigma, omega)
R> plot( Y$sample, col = Y$label )
```

5.6.2 Model selection

The true value of K is not known in practice. This quantity can be determined through the Bayesian information criterion (BIC) [Schwarz et al., 1978]. The process of finding best K is known as the *model selection* in the literature. Based on n realizations $\boldsymbol{y}_1, \cdots, \boldsymbol{y}_n$ that are assumed to follow independently a distribution with PDF $g(\boldsymbol{y}|\boldsymbol{\Psi})$, the BIC is defined as

$$\text{BIC} = 2\sum_{i=1}^{n} \log \sum_{k=1}^{\text{K}} \widehat{\omega_k} f\left(\boldsymbol{y}_i | \widehat{\boldsymbol{\Theta}_k}\right) - N \log n, \tag{5.63}$$

where $\widehat{\omega_k}$ and $\widehat{\boldsymbol{\Theta}_k}$ (for $k = 1, \cdots, \text{K}$) account for the ML estimators of ω_k and $\boldsymbol{\Theta}_k$, respectively. Furthermore, constant N denotes number of model's free parameters that must be estimated. The BIC is an effective criterion for model selection in Gaussian mixture models, see Fraley and Raftery [2002], Mcnicholas and Murphy [2008], and has been used frequently for other model-based clustering, seeKeribin [2000], Fraley and Raftery [1998, 2002], Mcnicholas and Murphy [2008], McNicholas and Murphy [2010]. In practice, for determining K, we compute BIC defined by (5.63) when data points are clustered through other clustering approaches such as k-mean [Hartigan and Wong, 1979] or other approaches of clustering, for $k = 1, 2, \cdots$. We choose the mixture model for which BIC is the largest. For instance, suppose that we do not know the true value of K when 300 observations are coming from a three-component bivariate Gaussian mixture model with true parameters given by (5.62). As expected, when K = 3 the BIC gets its maximum value, see Figure 5.10(b). The `R` function `BIC` has been developed for determining the true value of K for a K-component Gaussian mixture model.

```
R> dmvnorm <- function(x, mean, sigma)
+ {
+ p <- length(mean)
+ 1/sqrt((2*pi)^p*det(sigma))*exp(-mahalanobis(x, mean, sigma)/2)
+ }
R> BIC <- function(Y, param = NULL, k_max = 6)
+ {
+ dim_y <- dim(Y)
+ n <- dim_y[1]
+ p <- dim_y[2]
+ if( is.null(param) )
+ {
+  bic <- rep(0, k_max)
+  for(k in 1:k_max)
+  {
```

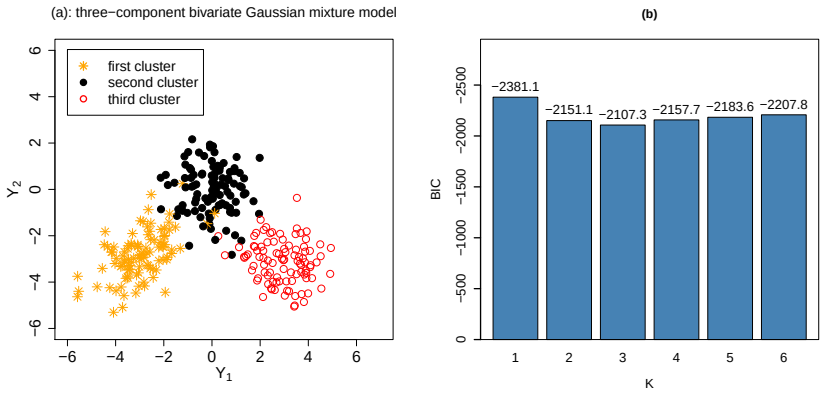


Figure 5.10: (a): Scatterplot of 300 data points drawn from $\boldsymbol{Y} = (Y_1, Y_2)^\top$ following a three-component Gaussian mixture model and (b): Computed BIC versus number of clusters.

```
+  Mu_hat <- matrix(0, nrow = k, ncol = p)
+  Sigma_hat <- array(0, dim = c(p, p, k) )
+  N <- (k - 1) + k*( p + p*(p + 1)/2 )
+  cluster <- kmeans(Y, k)
+  omega <- cluster$size/n
+  Sum_bic <- matrix(0, nrow = n, ncol = k)
+   for(i in 1:k)
+   {
+   clust <- Y[cluster$cluster == i, ]
+   Mu_hat[i,] <- apply(clust, 2, mean)
+   Sigma_hat[,,i] <- cov(clust)
+   Sum_bic[, i] <- omega[i]*dmvnorm(Y, mean = Mu_hat[i,],
+                    sigma = Sigma_hat[,,i])
+   }
+  bic[k] <- 2*sum( log( apply(Sum_bic, 1, sum) ) ) - N*log(n)
+  }
+  out <- data.frame(k = 1:k, bic = bic)
+  ls <- list( "model" = out, "K" = which.max(bic) )
+ }else{
+  omega <- param[[1]]
+  Mu <- param[[2]]
+  Sigma <- param[[3]]
+  K <- length( Mu )
+  Sum_bic <- matrix(0, nrow = n, ncol = K)
+  N <- (K - 1) + K*( p + p*(p + 1)/2 )
+   for(k in 1:K)
+   {
+   Sum_bic[, k] <- omega[k]*dmvnorm(Y, mean = Mu[[k]],
+                    sigma = Sigma[[k]])
+   }
+  out <- 2*sum( log( apply(Sum_bic, 1, sum) ) ) - N*log(n)
+  ls <- list( "bic" = out, "K" = K )
+ }
+ return(ls)
+ }
```

As noted earlier, determining the label of each observation is the main task of each clustering technique. To this end, we may use the Bayesian (Gibbs sampling) or the expectation-maximization (EM) [Dempster et al., 1977] algorithm. If the former applied, depending on the model structure, the Gibbs sampling involves one, two, or more latent variables.

5.6.3 Finite mixture of Gaussian distributions

Evidently the most commonly used statistical distribution for the purpose of model-based clustering is Gaussian mixture model. Suppose p-dimensional random vector $\boldsymbol{Y}$ follows a K-component Gaussian mixture model. It is clear that

$$\begin{aligned}\boldsymbol{Y}|H_{ik}=1 &\sim \mathcal{N}_p(\boldsymbol{\mu}_k,\Sigma_k),\\ \boldsymbol{H}_i &\sim \mathcal{MUL}(\boldsymbol{\omega}|1).\end{aligned}$$

Since $\boldsymbol{H}_i$ is latent variable, hence we encounter with a incomplete data problem. Here $\{\boldsymbol{y}_1,\cdots,\boldsymbol{y}_n\}$ is the sequence of observed data and $\boldsymbol{H}_i=(H_{i1},\cdots,H_{i\mathrm{K}})^\top$ is the missing component's label. Herein $H_{ik}=1$ takes on values zero and one. If $H_{ik}=1$, then $\boldsymbol{y}_i$ comes from k-th component, otherwise $H_{i1}=0$. Hence, the complete data PDF can be represented as

$$g_c(\underline{\boldsymbol{y}}|\boldsymbol{\Psi})=\prod_{i=1}^{n} g(\boldsymbol{y}_i,\boldsymbol{H}_i|\boldsymbol{\Psi}), \tag{5.64}$$

where $\underline{\boldsymbol{y}}=\{\boldsymbol{y}_1,\cdots,\boldsymbol{y}_n\}$ and

$$\begin{aligned}g(\boldsymbol{y}_i,\boldsymbol{H}_i|\boldsymbol{\Psi}) &= g(\boldsymbol{y}_i|\boldsymbol{H}_i,\boldsymbol{\Psi})\times\mathcal{MUL}(\boldsymbol{\omega}|1)\\ &=\Big[\mathcal{N}_p\Big(\boldsymbol{y}_i\Big|\boldsymbol{\mu}_k,\Sigma_k\Big)\Big]^{H_{ik}}\times\omega_1^{H_{i1}}\times\cdots\times\omega_{\mathrm{K}}^{H_{i\mathrm{K}}}\\ &=\prod_{k=1}^{\mathrm{K}}\Big[\omega_k\mathcal{N}_p\Big(\boldsymbol{y}_i\Big|\boldsymbol{\mu}_k,\Sigma_k\Big)\Big]^{H_{ik}}.\end{aligned} \tag{5.65}$$

Replacing the RHS of (5.65) into the RHS of (5.64), the PDF of complete-data is

$$g_c(\underline{\boldsymbol{y}}|\boldsymbol{\Psi})=\prod_{i=1}^{n}\prod_{k=1}^{\mathrm{K}}\Big[\omega_k\mathcal{N}_p\Big(\boldsymbol{y}_i\Big|\boldsymbol{\mu}_k,\Sigma_k\Big)\Big]^{H_{ik}}. \tag{5.66}$$

Thus, the complete-data likelihood function is

$$\begin{aligned}L_c(\boldsymbol{\Psi}|\underline{\boldsymbol{y}},\underline{\boldsymbol{H}}) =&(2\pi)^{-\frac{np}{2}}\times\omega_k^{\sum_{i=1}^{n}H_{ik}}\times\cdots\times\omega_{\mathrm{K}}^{\sum_{i=1}^{n}H_{i\mathrm{K}}}\times\big|\Sigma_k\big|^{-\frac{\sum_{i=1}^{n}\sum_{k=1}^{\mathrm{K}}H_{ik}}{2}}\\ &\times\exp\Big\{-\frac{1}{2}\sum_{i=1}^{n}\sum_{k=1}^{\mathrm{K}}H_{ik}(\boldsymbol{y}_i-\boldsymbol{\mu}_k)^\top\Sigma_k^{-1}(\boldsymbol{y}_i-\boldsymbol{\mu}_k)\Big\},\\ \propto&\omega_1^{n_1}\times\cdots\times\omega_{\mathrm{K}}^{n_{\mathrm{K}}}\times\big|\Sigma_1\big|^{-\frac{n_1}{2}}\times\cdots\times\big|\Sigma_{\mathrm{K}}\big|^{-\frac{n_{\mathrm{K}}}{2}}\\ &\times\exp\Big\{-\frac{1}{2}\sum_{i=1}^{n}\sum_{k=1}^{\mathrm{K}}H_{ik}(\boldsymbol{y}_i-\boldsymbol{\mu}_k)^\top\Sigma_k^{-1}(\boldsymbol{y}_i-\boldsymbol{\mu}_k)\Big\},\end{aligned} \tag{5.67}$$

where $\underline{\boldsymbol{H}}=\{\boldsymbol{H}_1,\cdots,\boldsymbol{H}_n\}$ in which $\boldsymbol{H}_i=(H_{i1},\cdots,H_{i\mathrm{K}})^\top$ and $n_k=\sum_{i=1}^{n}H_{ik}$ provided that $n=\sum_{k=1}^{\mathrm{K}}n_k$. Given ith observed data $\boldsymbol{y}_i$, for computing PDF of $\boldsymbol{H}_i|\boldsymbol{Y}=\boldsymbol{y}_i$, it follows from (5.65) that

$$L_c(\boldsymbol{\Psi}|\boldsymbol{y}_i,\boldsymbol{H}_i)\propto g(\boldsymbol{y}_i|\boldsymbol{H}_i,\boldsymbol{\Psi})\times\mathcal{MUL}(\boldsymbol{\omega}|1).$$

Obviously,

$$\boldsymbol{H}_i|\boldsymbol{\Psi},\boldsymbol{Y}=\boldsymbol{y}_i\sim\mathcal{MUL}(\boldsymbol{\tau}_i|1), \tag{5.68}$$

where $\boldsymbol{\tau}_i=(\tau_{i1},\cdots,\tau_{i\mathrm{K}})^\top$ denotes the ith row of $n\times\mathrm{K}$ matrix $\boldsymbol{\tau}$ whose (i,k)th element is

$$\tau_{i,k}=\frac{\omega_k\mathcal{N}_p(\boldsymbol{y}_i|\boldsymbol{\mu}_k,\Sigma_k)}{\sum_{k=1}^{\mathrm{K}}\omega_k\mathcal{N}_p(\boldsymbol{y}_i|\boldsymbol{\mu}_k,\Sigma_k)}. \tag{5.69}$$

Once we have generated the entire sample (for $i = 1, \cdots, n$) from full conditional $\boldsymbol{H}_i|(\boldsymbol{\Psi}, \boldsymbol{y}_i)$, we proceed to generate from full conditionals $\boldsymbol{\mu}_k|(\underline{\boldsymbol{y}}, \underline{\boldsymbol{H}}, \boldsymbol{\Psi}_{(-\boldsymbol{\mu}_k)})$ and $\Sigma_k|(\underline{\boldsymbol{y}}, \underline{\boldsymbol{H}}, \boldsymbol{\Psi}_{(-\Sigma_k)})$ (for $k = 1, \cdots, \mathrm{K}$). To this end, first we have

$$\begin{aligned}\pi(\boldsymbol{\Psi}|\underline{\boldsymbol{y}}, \underline{\boldsymbol{H}}) &\propto L_c(\boldsymbol{\Psi}|\underline{\boldsymbol{y}}, \underline{\boldsymbol{H}}) \times \pi(\boldsymbol{\Psi}_1) \times \cdots \times \pi(\boldsymbol{\Psi}_{\mathrm{K}}) \\ &= L_c(\boldsymbol{\Psi}|\underline{\boldsymbol{y}}, \underline{\boldsymbol{H}}) \times \pi(\boldsymbol{\omega}) \times \pi(\boldsymbol{\mu}_1) \times \cdots \times \pi(\boldsymbol{\mu}_{\mathrm{K}}) \times \pi(\Sigma_1) \times \cdots \times \pi(\Sigma_{\mathrm{K}}). \qquad (5.70)\end{aligned}$$

Substituting $L_c(\boldsymbol{\Psi}|\underline{\boldsymbol{y}}, \underline{\boldsymbol{H}})$ given by (5.67) into the RHSS of (5.70) yields

$$\begin{aligned}\pi(\boldsymbol{\Psi}|\underline{\boldsymbol{y}}, \underline{\boldsymbol{H}}) \propto \times \pi(\boldsymbol{\omega}) \times \pi(\boldsymbol{\mu}_1) \times \cdots \times \pi(\boldsymbol{\mu}_{\mathrm{K}}) \times \pi(\Sigma_1) \times \cdots \times \pi(\Sigma_{\mathrm{K}}) \times \\ \omega_1^{n_1} \times \cdots \times \omega_{\mathrm{K}}^{n_{\mathrm{K}}} \times |\Sigma_1|^{-\frac{n_1}{2}} \times \cdots \times |\Sigma_{\mathrm{K}}|^{-\frac{n_{\mathrm{K}}}{2}} \\ \times \exp\Big\{-\frac{1}{2}\sum_{i=1}^{n}\sum_{k=1}^{\mathrm{K}} H_{ik}(\boldsymbol{y}_i - \boldsymbol{\mu}_k)^\top \Sigma_k^{-1}(\boldsymbol{y}_i - \boldsymbol{\mu}_k)\Big\}. \qquad (5.71)\end{aligned}$$

We consider, for $k = 1, \cdots, \mathrm{K}$, independent conjugate priors as follows.

$$\pi(\boldsymbol{\omega}) \sim \mathcal{DIR}(\boldsymbol{\rho}), \qquad (5.72)$$
$$\pi(\boldsymbol{\mu}_k) \sim \mathcal{N}_p(\mathcal{M}_k, \mathcal{S}_k), \qquad (5.73)$$
$$\pi(\Sigma_k) \sim \mathcal{IW}(\mathcal{D}_k, \nu_k), \qquad (5.74)$$

where $\mathcal{DIR}(\cdot)$ is defined in (**??**). In what follows, we give the full conditionals needed for implementing the Gibbs sampling.

- **full conditional of $\boldsymbol{\omega}|(\underline{\boldsymbol{y}}, \underline{\boldsymbol{H}}, \boldsymbol{\Psi}_{(-\boldsymbol{\omega})})$:** Substituting prior $\pi(\boldsymbol{\omega})$ given by (5.71) into the RHS of (5.70) yields

  $$\pi(\boldsymbol{\omega}|\underline{\boldsymbol{y}}, \underline{\boldsymbol{H}}, \boldsymbol{\Psi}_{(-\boldsymbol{\mu})}) \propto \mathcal{DIR}(\boldsymbol{\omega}|\boldsymbol{\rho}) \times \omega_1^{n_1} \times \cdots \times \omega_{\mathrm{K}}^{n_{\mathrm{K}}} = \mathcal{DIR}(\boldsymbol{\omega}|\boldsymbol{\rho}_*),$$

  where

  $$\boldsymbol{\rho}_* = \Big(\rho_1 + \sum_{i=1}^{n} H_{i1}, \cdots, \rho_{\mathrm{K}} + \sum_{i=1}^{n} H_{i\mathrm{K}}\Big)^\top.$$

- **full conditional of $\boldsymbol{\mu}_k|(\underline{\boldsymbol{y}}, \underline{\boldsymbol{H}}, \boldsymbol{\Psi}_{(-\boldsymbol{\mu}_k)})$:** Substitute the prior $\pi(\boldsymbol{\mu}_k)$ given by (5.72) into the RHS of (5.70). Then more algebra shows

  $$\pi(\boldsymbol{\mu}|\underline{\boldsymbol{y}}, \underline{\boldsymbol{H}}, \boldsymbol{\Psi}_{(-\boldsymbol{\mu}_k)}) = \mathcal{N}_p\Big(\boldsymbol{\mu}\Big|\mathcal{Q}_k\Big[\mathcal{S}_k^{-1}\mathcal{M}_k + \Sigma_k^{-1}\sum_{i=1}^{n} H_{ik}\boldsymbol{y}_i\Big], \mathcal{Q}_k\Big),$$

  where

  $$\mathcal{Q}_k = \Big[\mathcal{S}_k^{-1} + \Sigma_k^{-1} \times \sum_{i=1}^{n} H_{ik}\Big]^{-1}.$$

- **full conditional of $\Sigma_k|(\underline{\boldsymbol{y}}, \underline{\boldsymbol{H}}, \boldsymbol{\Psi}_{(-\Sigma_k)})$:** Substitute the prior $\pi(\Sigma_k)$ given by (5.73) into the RHS of (5.70). Then more algebra shows

  $$\pi(\Sigma|\underline{\boldsymbol{y}}, \underline{\boldsymbol{H}}, \boldsymbol{\Psi}_{(-\Sigma_k)}) \sim \mathcal{IW}(\mathcal{R}_k, \nu_k + \sum_{i=1}^{n} H_{ik}),$$

  where $\mathcal{R}_k = \mathcal{D}_k + \sum_{i=1}^{n} H_{ik}(\boldsymbol{y}_i - \boldsymbol{\mu}_k)(\boldsymbol{y}_i - \boldsymbol{\mu}_k)^\top$.

Algorithm 20 Bayesian inference for finite mixture of Gaussian distributions

Read N, M, and determine quantities K, ω_k, $\mathcal{M}_k$, $\mathcal{S}_k$, $\mathcal{D}_k$, ν_k, for $k = 1 \cdots, \mathrm{K}$
Set $t = 0$;
Determine initial values $\boldsymbol{\omega}^{(0)}$, $\boldsymbol{\mu}^{(0)}$, $\Sigma^{(0)}$, and set $\boldsymbol{\Psi}^{(0)} = (\boldsymbol{\omega}^{(0)}, \boldsymbol{\mu}^{(0)}, \Sigma^{(0)})$;
while $t \leq N$ **do** ▷ start Gibbs sampling
    Set $i = 1$;
    while $i \leq n$ **do** ▷ start for sampling form missing labels
        Simulate the i row of $n \times \mathrm{K}$ matrix $\boldsymbol{H}$ as $\boldsymbol{H}_i \sim \mathcal{MUL}(\cdot|\boldsymbol{\tau}_i)$ where
        $\boldsymbol{\tau}_i = (\tau_{i1}, \ldots, \tau_{i\mathrm{K}})^\top$ is the ith row of matrix $\boldsymbol{\tau}$ is given by (5.69);
        Set $i = i + 1$;
    end
    Set, for $k = 1, \cdots, \mathrm{K}$, $n_k = \sum_{i=1}^{n} H_{i,k}$ where $H_{i,k}$ denotes the (i,k)th
    element of $\boldsymbol{H}$;
    Simulate $\boldsymbol{\omega} \sim \mathcal{DIR}(1|\rho_*)$ where $\rho_* = (\rho_1 + n_1, \cdots, \rho_{\mathrm{K}} + n_{\mathrm{K}})^\top$;
    Set $k = 1$;
    while $k \leq K$ **do** ▷ start sampling from full conditionals of $\boldsymbol{\mu}_k$, and Σ_k
        Generate $\boldsymbol{\mu}_k^{(t+1)} \sim \mathcal{N}_p\Big(\mathcal{Q}_k\Big[\mathcal{S}_k^{-1}\mathcal{M}_k + \big[\Sigma_k^{(t)}\big]^{-1} \sum_{i=1}^{n} H_{i,k} \times \boldsymbol{y}_i\Big], \mathcal{Q}_k\Big)$
        where $\mathcal{Q}_0 = \Big[\mathcal{S}_0^{-1} + \big[\Sigma_k^{(t)}\big]^{-1} \times n_k\Big]^{-1}$;
        Generate $\Sigma_k^{(t+1)}$ from $\mathcal{IW}(\mathcal{R}_k, \nu_k + n_k)$ where $\mathcal{R}_k = \mathcal{D}_k + \sum_{i=1}^{n} H_{i,k} \times$
        $(\boldsymbol{y}_i - \boldsymbol{\mu}_k^{(t+1)})(\boldsymbol{y}_i - \boldsymbol{\mu}_k^{(t+1)})^\top$;
        Set $k = k + 1$;
    end
    Set $\boldsymbol{\Psi}^{(t+1)} = (\boldsymbol{\omega}^{(t+1)}, \boldsymbol{\mu}_1^{(t+1)} \cdots, \boldsymbol{\mu}_{\mathrm{K}}^{(t+1)}, \Sigma_1^{(t+1)}, \cdots, \Sigma_{\mathrm{K}}^{(t+1)})$ and $t = t + 1$;
end
Sequence $\{\boldsymbol{\Psi}^{(M+1)}, \cdots, \boldsymbol{\Psi}^{(N)}\}$ is a sample of size $N - M$ from
posterior $\pi(\boldsymbol{\Psi}|\underline{\boldsymbol{y}}, \underline{\boldsymbol{H}})$;
The Bayesian estimator of $\boldsymbol{\Psi}$ is given by $\frac{1}{N-M} \sum_{t=1}^{N-M} \boldsymbol{\Psi}^{(t)}$.

The following pseudo code gives the details for implementing the model-based clustering for finite mixture of Gaussian distributions.

While Algorithm 20 works reasonably well, its performance is subject to outliers. If clusters have large variances, then the Gibbs sampling may fail to allocate true label to each cluster. This mainly because this fact that Gaussian distribution has light tails and the PDF decays exponentially for observation are far from the center of distribution. For instance, consider a two-component bivariate mixture model with equal weights $\omega_1 = \omega_2 = 0.5$ while the second cluster has large locations and variances. It is easy to check that the elements of second column of matrix $\boldsymbol{\tau}$ in (5.69) are very small and hence, $P(H_{i,2} = 1) \to 1$ for $i = 1, \cdots, n$. Hence, all observations are allocated to the first cluster. One way to tackle this issue is standardization of observations before clustering [Tortora et al., 2021]. The following example gives more detailed information about this case.

Example 5-9

Herein, we are interested in clustering the famous Old Faithful Geyser Data (known as faithful data). This faithful data consists of 272 observations on two variables *duration of the eruption* in minutes and *waiting time between eruptions* for the Old Faithful geyser in Yellowstone National Park, Wyoming, USA, [Azzalini and Bowman, 1990]. Figure 5.11(a) displays the scatterplot of this dataset. Our investigation shows that the BIC of -833.7747 suggests that K = 2 for which the k-mean algorithm pre-clustering results shows that the variances of second variable in both clusters are significantly large as 31.66 and 34.75. Using the `R` code after standardization given by

```
R> data(faithful)
R> Y <- matrix( cbind(faithful$eruptions, faithful$waiting),
+       nrow = 272, ncol = 2)
R> Z <- apply( Y, 2, function(x)( x - mean(x) )/sd(x) )
R> bic <- BIC(Z)
R> out_kmeans <- kmeans(Z, bic$K)
R> label <- out_kmeans$cluster
R> sapply(1:K, function(x) var( Z[label == x, ] ))
```

then, it can be seen that variances of second variable in both clusters are 0.1876 and 0.1874, correspondingly. The R package `mclust` [Scrucca et al., 2016] applies the EM algorithm for model-based clustering based on the finite mixture of Gaussian distributions. We notice that the clustering results based on both the Gibbs sampling proposed here and EM algorithm proposed in [Scrucca et al., 2016] are the same. The most commonly used criterion for measuring the degree of agreement between clustering approaches is the adjusted Rand index (ARI) that varies from 0 (for zero agreement in clustering) to 1 (for perfect agreement in clustering) [Hubert and Arabie, 1985]. Herein, we use the ARI to compare the Bayesian paradigm and EM algorithm for clustering the faithful data. To this end, among several `R` packages that can compute the ARI, we call the package `MixGHD`. Fortunately, the agreement between two above approaches in clustering faithful data is perfect, that is ARI=1. Figure 5.11(b) and (c) display the contour and scatterplot fitted to standardized faithful data. The `R` function `FGM` given by the following can be used for applying the model-based clustering to the Gaussian finite mixture model.

```
1 R> FMG <- function(Y, K, n_burn, n_sim, param, hyper = NULL, scale = FALSE )
2 + {
3 + if( scale == TRUE ) Y <- apply( Y, 2, function(x)( x - mean(x) )/sd(x) )
4 + dim_y <- dim(Y)
5 + n <- dim_y[1]
6 + p <- dim_y[2]
```

```
+ dim_Psi <- K + K*p + K*p^2
+ Psi <- matrix(0, nrow = n_sim, ncol = dim_Psi)
+  if( is.null(hyper) )
+  {
+  rho <- rep(1, K)
+  M <- rep( list( rep(0, p) ), K )
+  S <- rep( list( diag(1, p) ), K )
+  nu <- rep(1, K)
+  D0 <- rep( list( diag(10, p) ), K )
+  }else{
+  rho <- hyper[[1]]
+  M <- hyper[[2]]
+  S <- hyper[[3]]
+  D0 <- hyper[[4]]
+  nu <- hyper[[5]]
+  }
+  omega <- param[[1]]
+  Mu <- param[[2]]
+  Sigma <- param[[3]]
+  index_Mu <- (K + 1):(K*p + K)
+  index_Sigma <- (K*p + K + 1):(K*(p^2+p+1))
+  Psi[1, 1:K ] <- omega
+  Psi[1, index_Mu] <- as.numeric( unlist( Mu ) )
+  Psi[1, index_Sigma] <- as.numeric( unlist(Sigma) )
+  A <- tau <- H <- matrix(0, nrow = n, ncol = K)
+    for(j in 2:n_sim)
+    {
+    for(k in 1:K) A[, k] <- omega[k]*dmvnorm(Y, mean = Mu[[k]], sigma =
+                  Sigma[[k]])
+    tau <- A/apply(A, 1, sum)
+    for(i in 1:n) H[i, ] <- rmultinom(n = 1, size = 1, tau[i, ])
+    n_k <- apply(H, 2, sum)
+    omega <- rDIR( rho + n_k )
+    for(k in 1:K){
+    index_H <- which(H[, k] == 1)
+    Y_k <- Y[index_H, ]
+    Q_k <- solve( solve(S[[k]]) + solve(Sigma[[k]])*n_k[k] )
+    Sum_Yk <- apply(Y_k, 2, sum)
+    Mu_k <- Q_k%*%( solve(S[[k]])%*%M[[k]] + solve(Sigma[[k]])%*%Sum_Yk )
+    Mu[[k]] <- mvrnorm(n = 1, mu = Mu_k, Sigma = Q_k )
+    R <- matrix(0, nrow = p, ncol = p)
+    for(i in index_H){R <- R + c( Y[i, ] - Mu[[k]] )%o%c( Y[i, ] - Mu[[k]] )}
+    Sigma[[k]] <- solve( rWishart(1, df = n_k[k] + nu[k], Sigma =
+                  solve(D0[[k]] + R) )[,,1] )
+    #print(Sigma[[k]])
+    }
+    Psi[j, 1:K ] <- omega
+    Psi[j, index_Mu] <- as.numeric( unlist( Mu ) )
+    Psi[j, index_Sigma] <- as.numeric( unlist(Sigma) )
+    }
+   omega_hat <- rep(1/K, K)
+   Mu_hat <- rep( list( rep(0, p) ), K )
+   Sigma_hat <- rep( list( diag(p) ), K )
+   out <- apply(Psi[n_burn:n_sim, ], 2, mean)
+   for(k in 1:K)
+   {
+   omega_hat[k] <- out[k]
+   Mu_hat[[k]] <- out[index_Mu[((k-1)*p + 1):(k*p)]]
+   Sigma_hat[[k]] <- matrix( out[index_Sigma[((k-1)*p^2 + 1):(k*p^2)]],
+                     nrow = p, ncol = p )
+   A[, k] <- omega_hat[k]*dmvnorm(Y, mean = Mu_hat[[k]], sigma =
+             Sigma_hat[[k]])
+   }
```

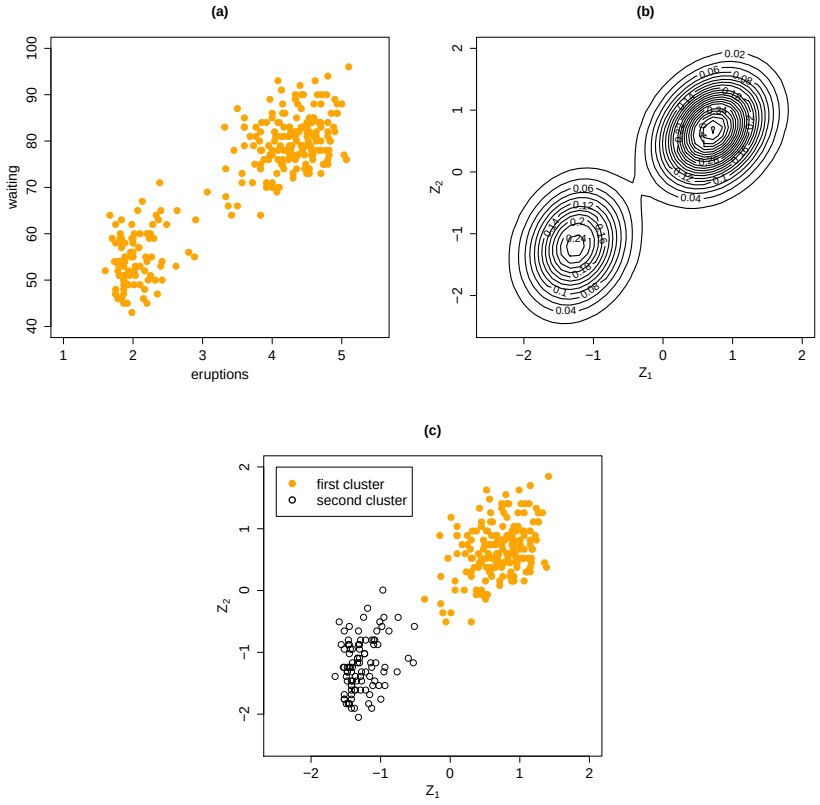


Figure 5.11: (a): Scatterplot of faithful data, (b): contour plot fitted to standardized faithful data, and (c): scatterplot of clustered standardized faithful data.

```
+ label <- apply(A, 1, which.max)
+ ls <- list( "omega" = omega, "Mu" = Mu_hat, "Sigma" = Sigma_hat,
+              "label" = label )
+ return(ls)
+ }
```

5.7 Simulating from truncated Gaussian distribution

A mentioned in Section 3.9, the truncated distribution is widely used in practice. Generating from truncated multivariate Gaussian distribution may arise in various contexts sch as probit regression [Albert and Chib, 1993], likelihood estimation for max-stable processes [Genton et al., 2011], fitting mixed effects models with censored data [Grün and Hornik, 2012], model-based clustering [Maleki et al., 2019], regression analysis [Rodriguez-Yam et al., 2004], logistic regression with covariates subject to measurement error and left censoring [Teimouri and Sinha, 2024]. For more detailed information on application of truncated multivariate Gaussian distribution, we the refer to Botev [2017] and references therein. Section 3.9 gives the first and second moments of a Gaussian distribution under rectangular constraint. Suppose p-dimensional random vector $\boldsymbol{X} = (X_1, \cdots, X_p)^\top$ follows a truncated Gaussian distribution on region with positive Lebesgue measure $\boldsymbol{R}$ denoted as

$$\mathcal{TN}_{\boldsymbol{R}}(\boldsymbol{\mu}, \Sigma), \quad \boldsymbol{R} = \{\boldsymbol{x} \in \mathbb{R}^p | \boldsymbol{a} \leq A\boldsymbol{x} \leq \boldsymbol{b}\}, \tag{5.75}$$

where $\boldsymbol{a} = (a_1, \cdots, a_p)^\top$ and $\boldsymbol{b} = (b_1, \cdots, b_p)^\top$ are vectors of finite or infinite constants. Moreover, matrix A is a $p \times p$ full rank matrix that constructs a set of p independent linear constraints.

One can investigate easily, as pointed out by [Geweke, 1991], that the marginals of $\boldsymbol{X}$ with distribution given by (5.75) are not truncated Gaussian, but the full conditionals, that is distribution of X_i (for $i = 1, \cdots, p$) *conditional* on all of other members of $\boldsymbol{X}$ are truncated Gaussian. To verify this claim, we shall confine ourselves here to suppose $p = 2$ and $A = \boldsymbol{I}_p$ in (5.75). This means that

$$\mathcal{TN}_{\boldsymbol{R}}(\boldsymbol{\mu}, \Sigma), \quad \boldsymbol{R} = \{\boldsymbol{x} = (x_1, x_2)^\top \in \mathbb{R}^2 | a_1 \leq x_1 \leq b_1, a_2 \leq x_2 \leq b_2\}, \tag{5.76}$$

where $\boldsymbol{\mu} = (\mu_1, \mu_2)^\top$ and $\Sigma = [(\sigma_{11}, \sigma_{12})^\top, (\sigma_{21}, \sigma_{22})^\top]$. For computing the marginal distribution of (5.76), e.g. X_1, whose PDF is denoted by $\mathcal{TN}_{R_1}(x_1|\boldsymbol{\mu}, \Sigma)$, we have

$$\begin{aligned}
\mathcal{TN}_{R_1}(x_1|\boldsymbol{\mu}, \Sigma) =& \frac{1}{\mathcal{P}_G} \int_{a_2}^{b_2} \boldsymbol{\phi}_2(\boldsymbol{x}|\boldsymbol{\mu}, \Sigma) dx_2, \\
=& \frac{1}{\mathcal{P}_G} \phi(x_1|\mu_1, \sigma_{11}) \int_{a_2}^{b_2} \phi(x_2|\mu_2^*, \sigma_{22}^*) dx_2, \\
=& \frac{1}{\mathcal{P}_G} \phi(x_1|\mu_1, \sigma_{11}) \Big[\Phi(b_2|\mu_2^*, \sigma_{22}^*) - \Phi(a_2|\mu_2^*, \sigma_{22}^*)\Big],
\end{aligned} \tag{5.77}$$

where $x_1 \in R_1$, $\mu_2^* = \mu_2 + \sigma_{12}\sigma_{11}^{-1}(x_1 - \mu_1)$, $\sigma_{22}^* = \sigma_{22} - \sigma_{12}\sigma_{11}^{-1}\sigma_{21}$, and $\mathcal{P}_G = \boldsymbol{\Phi}_2(\boldsymbol{b}|\boldsymbol{\mu}, \Sigma) - \boldsymbol{\Phi}_2(\boldsymbol{a}|\boldsymbol{\mu}, \Sigma)$. Obviously, since both terms within the square bracket in the RHS of (5.77) depend on x_1 through μ_2^*, hence $\mathcal{TN}_{R_1}(x_1|\boldsymbol{\mu}, \Sigma)$ is no longer the PDF of a Gaussian distribution unless μ_2^* is independent of x_1 that occurs only when $\sigma_{12} = \sigma_{21} = 0$. This is while both of the full conditionals $X_1|X_2$ and $X_2|X_1$ correspond to the joint PDF in (5.76) are Gaussian. For example, $X_1|X_2 \sim \mathcal{TN}_{R_1}(\mu_1^*, \sigma_{11}^*)$ where $R_1 = \{x_1 \in \mathbb{R} | a_1 \leq x_1 \leq b_1\}$, $\mu_1^* = \mu_1 + \sigma_{12}\sigma_{22}^{-1}(x_2 - \mu_2)$, $\sigma_{11}^* = \sigma_{11} - \sigma_{12}\sigma_{22}^{-1}\sigma_{21}$, and $a_2 \leq x_2 \leq b_2$. The following proposition generalizes the fact described above for higher dimensions.

Proposition 5.7.1. *Rodriguez-Yam et al. [2004] Suppose $\boldsymbol{X} \sim \mathcal{TN}_{\boldsymbol{R}}(\boldsymbol{\mu}, \Sigma)$ as defined in (5.75). Furthermore $\boldsymbol{Y} = \boldsymbol{c} + B\boldsymbol{X}$ where $\boldsymbol{c} = (c_1, \cdots, c_p)^\top$ is a vector of finite constants and B is a $p \times p$ full rank matrix. Then,*

i. $\boldsymbol{Y} \sim \mathcal{TN}_{\boldsymbol{T}}(\boldsymbol{c} + B\boldsymbol{\mu}, B\Sigma B^\top)$, $\boldsymbol{T} = \{\boldsymbol{y} \in \mathbb{R}^p | \boldsymbol{c} + BA^{-1}\boldsymbol{a} \leq \boldsymbol{y} \leq \boldsymbol{c} + BA^{-1}\boldsymbol{b}\}$.

ii. For a given partition of $\boldsymbol{X}$, $\boldsymbol{\mu}$, and Σ given by

$$\boldsymbol{X} = \begin{bmatrix} \boldsymbol{X}_1 \\ X_p \end{bmatrix}, \boldsymbol{\mu} = \begin{bmatrix} \boldsymbol{\mu}_1 \\ \mu_p \end{bmatrix}, \Sigma = \begin{bmatrix} \Sigma_{11} & \Sigma_1 \\ \Sigma_1^\top & \sigma_{pp} \end{bmatrix},$$

we have

$$X_p|\boldsymbol{X}_1 = \boldsymbol{x}_1 \sim \mathcal{TN}_{R_p}(\mu_p^*, \sigma_{pp}^*), \tag{5.78}$$

where

$$\mu_p^* = \mu_p + \Sigma_1^\top \Sigma_{11}^{-1}(\boldsymbol{x}_1 - \boldsymbol{\mu}_1), \tag{5.79}$$

$$\sigma_{pp}^* = \sigma_{pp} - \Sigma_1^\top \Sigma_{11}^{-1}\Sigma_1, \tag{5.80}$$

$$R_p = \{x_p \in \mathbb{R} | (\boldsymbol{x}_1, x_p) \in \boldsymbol{R}\}.$$

In what follows, we review two algorithms for generating from truncated multivariate Gaussian distribution on a set of linearly independent constraints using the Gibbs sampling. The first algorithm is due to Geweke [1991]. For a given covariance matrix Σ and a $p \times p$ full rank matrix A, suppose $\boldsymbol{Z} \sim \mathcal{TN}_{\boldsymbol{T}}(\boldsymbol{0}, A\Sigma A^\top)$ follows a truncated Gaussian distribution on region $\boldsymbol{T} = \{\boldsymbol{z} \in \mathbb{R}^p | \boldsymbol{\alpha} \leq \boldsymbol{z} \leq \boldsymbol{\beta}\}$ with positive Lebesgue measure in which

$$\boldsymbol{\alpha} = \boldsymbol{a} - A\boldsymbol{\mu}, \tag{5.81}$$

$$\boldsymbol{\beta} = \boldsymbol{b} - A\boldsymbol{\mu}, \tag{5.82}$$

are vectors of finite or infinite constants. It follows from Proposition (5.7.1) (i) that $\boldsymbol{X} = \boldsymbol{\mu} + A^{-1}\boldsymbol{Z} \sim \mathcal{TN}_{\boldsymbol{R}}(\boldsymbol{\mu}, \Sigma)$ where $\boldsymbol{R} = \{\boldsymbol{x} \in \mathbb{R}^p | \boldsymbol{a} \leq A\boldsymbol{x} \leq \boldsymbol{b}\}$ as (5.75). Let $\boldsymbol{z}_{-i}$ denote the realization associated with $\boldsymbol{Z}_{-i} = (Z_1, \cdots, Z_{i-1}, Z_{i+1}, \cdots, Z_p)^\top$. It follows from Proposition (5.7.1) (ii) that

$$Z_i | (\boldsymbol{Z}_{-i} = \boldsymbol{z}_{-i}) \sim \mathcal{TN}_{T_i}(\mu_i^*, \sigma_{ii}^*), \quad T_i = \{z_i \in \mathbb{R} | c_i \leq z_i \leq d_i\}, \tag{5.83}$$

where elements of vectors $\boldsymbol{c} = (c_1, \cdots, c_p)^\top$ and $\boldsymbol{d} = (d_1, \cdots, d_p)^\top$ are given by (5.81) and (5.82), respectively. Furthermore, based on (5.79) and (5.80), for $i = 1, \cdots, p$, we have

$$\mu_i^* = V_1^\top V_{11}^{-1} \boldsymbol{z}_{-i}, \tag{5.84}$$

$$\sigma_{ii}^* = V_{ii} - V_1^\top V_{11}^{-1} V_1, \tag{5.85}$$

in which $V = A\Sigma A^\top$. Algorithm 21 describes the method proposed by Geweke [1991] for simulating from truncated Gaussian distribution. The second algorithm is due to Rodriguez-

Algorithm 21 Simulating truncated Gaussian distribution: Geweke's method

1: Read $\boldsymbol{a}$, $\boldsymbol{b}$, $\boldsymbol{\mu}, \Sigma, A$ given by (5.75), and N for number of Gibbs sampler runs;
2: Set $t = 0$ and suggest the initial vector $\boldsymbol{z}^{(t)} = (z_1^{(t)}, \cdots, z_p^{(t)})^\top \in \boldsymbol{T}$ where T_i (for $i = 1, \cdots, p$) is given by (5.83);
3: **while** $t < N$ **do**
4: set $i = 1$;
5: **while** $i \leq p$ **do**
6: Draw $z_i^{(t+1)}$ form full conditional $Z_i^{(t+1)} | (Z_1^{(t+1)} = z_1^{(t+1)}, \cdots, Z_{i-1}^{(t+1)} = z_{i-1}^{(t+1)}, Z_{i+1}^{(t)} = z_{i+1}^{(t)}, \cdots, Z_p^{(t)} = z_p^{(t)})$ that follows a truncated Gaussian distribution with PDF given by (5.83);
7: set $i = i + 1$;
8: **end**
9: $\boldsymbol{z}^{(t+1)} \leftarrow \boldsymbol{z}^{(t)}$;
10: Accept $\boldsymbol{x}^{(t+1)} = \boldsymbol{\mu} + A^{-1}\boldsymbol{z}^{(t+1)}$ as a sample of from distribution in (5.75);
11: set $t = t + 1$;
12: **end**

Yam et al. [2004]. Let p-dimensional random vector $\boldsymbol{X} = (X_1, \cdots, X_p)^\top$ follows a truncated Gaussian distribution on region $\boldsymbol{R}$ denoted as

$$\mathcal{TN}_{\boldsymbol{R}}(\boldsymbol{\mu}, \Sigma), \quad \boldsymbol{R} = \{\boldsymbol{x} \in \mathbb{R}^p | C\boldsymbol{x} \leq \boldsymbol{b}\}, \tag{5.86}$$

where rows of matrix B are not restricted to be linearly independent. If D is a square matrix of full rank for which we have $D\Sigma D^\top = \boldsymbol{I}_p$, then for transformation $\boldsymbol{Z} = D\boldsymbol{X}$ it follows from Proposition 5.7.1 (i) that

$$\boldsymbol{Z} \sim \mathcal{TN}_{\boldsymbol{T}}(D\boldsymbol{\mu}, \boldsymbol{I}_p), \quad \boldsymbol{T} = \{\boldsymbol{z} = D\boldsymbol{x} \in \mathbb{R}^p | E\boldsymbol{z} \leq \boldsymbol{b}\}, \tag{5.87}$$

where

$$E = CD^{-1}. \tag{5.88}$$

It is not hard to see that the only choice for D is inverse of the lower-triangular Cholesky decomposition of Σ. Using the transformation $\boldsymbol{Z} = D\boldsymbol{X}$ avoids computing μ_i^* and σ_{ii}^* given in (5.83) when applying Algorithm 21 proposed by Geweke [1991]. Suppose $\boldsymbol{\alpha} = D\boldsymbol{\mu}$, $\boldsymbol{Z}_{-i} =$

$(Z_1, \cdots, Z_{i-1}, Z_{i+1}, \cdots, Z_p)^\top$, and $\boldsymbol{z}_{-i}$ denotes the realization of $\boldsymbol{Z}_{-i}$. Based on Proposition 5.7.1 (ii), we have

$$Z_i \big| (\boldsymbol{Z}_{-i} = \boldsymbol{z}_{-i}) \sim \mathcal{TN}_{T_i}(\alpha_i, 1), \quad T_i = \{z_i \in \mathbb{R} \big| E\boldsymbol{z} \leq \boldsymbol{b}\}, \tag{5.89}$$

where α_i is the ith element of vector $\boldsymbol{\alpha}$. Let E_{-i} denote the matrix E given in (5.88) when its i-th column is removed and $\boldsymbol{e}_i$ is the i-th column of matrix E. Then, the region T_i given in (5.89) can be represented as

$$T_i = \{z_i \in \mathbb{R} \big| \boldsymbol{e}_i z_i \leq \boldsymbol{b} - E_{-i}\boldsymbol{z}_{-i}\}. \tag{5.90}$$

As pointed out by Rodriguez-Yam et al. [2004], since constraints on $\boldsymbol{X}$ construct a convex region in $\mathbb{R}^p$, hence the region T_i given by (5.90) takes the forms $l_i \leq z_i \leq u_i$, $-\infty \leq z_i \leq u_i$, or $l_i \leq u_i \leq \infty$ in which l_i and u_i are, accordingly, the lower and upper bounds of inequalities obtained from (5.90). Algorithm 22 describes how to implement the method proposed by Rodriguez-Yam et al. [2004] for simulating from a truncated Gaussian distribution. In what follows, we are willing to review an example given by [Rodriguez-Yam et al., 2004] for simulating from a bivariate truncated Gaussian distribution. It is worthwhile to note that the methods proposed by Geweke [1991] and [Rodriguez-Yam et al., 2004] hereafter are denoted as Method 1 and Method 2, respectively.

Algorithm 22 Simulating truncated Gaussian distribution: Rodriguez-Yam et al. method

1: Read $\boldsymbol{b}$, $\boldsymbol{\mu}$, Σ, C given by (5.86), and N for number of Gibbs sampler runs;
2: Compute $\boldsymbol{\alpha} = D\boldsymbol{\mu}$ and $E = CD^{-1}$ in which D is the Cholesky decomposition of Σ;
3: Set $t = 0$ and suggest the initial value $\boldsymbol{z}^{(t)} = (z_1^{(t)}, \cdots, z_p^{(t)})^\top \in \boldsymbol{T}$ where $\boldsymbol{T}$ is given by (5.90);
4: **while** $t \leq N$ **do**
5: set $i = 1$;
6: **while** $i < p$ **do**
7: Draw $z_i^{(t+1)}$ form full conditional $Z_i^{(t+1)} \big| (Z_1^{(t+1)} = z_1^{(t+1)}, \cdots, Z_{i-1}^{(t+1)} = z_{i-1}^{(t+1)}, Z_{i+1}^{(t)} = z_{i+1}^{(t)}, \cdots, Z_p^{(t)} = z_p^{(t)})$ that follows a truncated Gaussian distribution with PDF given by (5.89);
8: set $i = i + 1$;
9: **end**
10: $\boldsymbol{Z}^{(t+1)} \leftarrow \boldsymbol{Z}^{(t)}$;
11: $\boldsymbol{X}^{(t+1)} = D^{-1}\boldsymbol{Z}^{(t+1)}$
12: set $t = t + 1$;
13: **end**

Example 5-10

Suppose we are interested in simulating from $\boldsymbol{X} = (X_1, X_2)^\top \sim \mathcal{TN}_{\boldsymbol{R}}(\boldsymbol{\mu}, \Sigma)$ where $\boldsymbol{R} = \{\boldsymbol{x} \in \mathbb{R}^2 \big| \boldsymbol{x} \geq \boldsymbol{0}\}$, $\boldsymbol{\mu} = (\mu_1, \mu_2)^\top$ and $\Sigma = [(1, 0.8)^\top, (0.8, 1)^\top]$. We discuss the key features for implementing Algorithm 21 and Algorithm 22 as follows.

- **Method 1**: For implementing the first step of Algorithm 21, we set $A = [(1, 0)^\top, (0, 1)^\top]$, $\boldsymbol{a} = (0, 0)^\top$, $\boldsymbol{b} = (+\infty, +\infty)^\top$, and $N = 2000$. The vector of initial values in the second step can be $\boldsymbol{z}^{(0)} = (1, 1)^\top$ and further, using $\boldsymbol{\mu} = (0, 0)^\top$, it follows from (5.81) and (5.82) that $T_1 = T_2 = [0, \infty)$. Based on (5.79) and (5.80), the full conditionals in the sixth step of Algorithm 21 are given as follows.

$$Z_1 \big| (\boldsymbol{Z}_{-1} = \boldsymbol{z}_{-1}) \sim \mathcal{TN}_{T_1}\Big(\frac{4}{5}z_2, \frac{9}{25}\Big),$$
$$Z_2 \big| (\boldsymbol{Z}_{-2} = \boldsymbol{z}_{-2}) \sim \mathcal{TN}_{T_2}\Big(\frac{4}{5}z_1, \frac{9}{25}\Big).$$

The R function called `rtrunnorm_gibbs(\cdot)` for implementing Method 1 is given as follows.

```
R> rtrunnorm_gibbs <- function(N, Mu, Sigma, a, b, A)
 +{
 + p <- length(Mu)
 + V <- A%*%Sigma%*%t(A)
 + T1 <- matrix( cbind(a[1:p] - A%*%Mu[1:p], b[1:p] - A%*%Mu[1:p]), nrow = p, ncol = 2)
 + Z <- matrix(0, nrow = N, ncol = p)
 + Z[1, ] <- (a + b)/2
 + Vinv <- array( 0, dim = c(p-1, p-1, p) )
 + V1 <- matrix(0, nrow = p, ncol = p-1)
 + for(i in 1:p)
 + {
 +  Vinv[, , i] <- solve( V[-i, -i] )
 +  V1[i, ] <- as.numeric( V[i, -i]%*%Vinv[, , i] )
 + }
 + j <- 1
 + while(j < N)
 + {
 +  for(i in p:1)
 +  {
 +   mu.star <- V1[i, ]%*%Z[j, -i]
 +   sigma.star <- sqrt( V[i, i] - V1[i, ]%*%V[i, -i] )
 +   u <- runif(1)
 +   cdf.a <- pnorm(T1[i, 1], mu.star, sigma.star)
 +   cdf.b <- pnorm(T1[i, 2], mu.star, sigma.star)
 +   Z[j, i] <- qnorm( u*(cdf.b - cdf.a) + cdf.a,
 +               mean = mu.star, sd = sigma.star )
 +   Z[j+1, i] <- Z[j, i]
 +  }
 + j <- j + 1
 + }
 + Z <- matrix(Mu, nrow = N, ncol = p, byrow = T) + Z%*%solve(A)
 + return(Z)
 +}
```

Hence, for simulating truncated multivariate Gaussian using Method 1, we proceed by the following.

```
R > N <- 20000 # number of Gibbs sampler iterations
R > X <- Z <- matrix(0, nrow = N, ncol = 2)
R > Mu <- c(0, 0)
R > Sigma <- matrix( c(1, 0.8, 0.8, 1), nrow = 2, ncol = 2)
R > A <- diag(2)
R > a <- rep(0, 2) # truncation lower bounds
R > b <- rep(Inf, 2) # truncation upper bounds
R > rtrunnorm_gibbs(N, Mu, Sigma, a, b, A)
```

- **Method 2**: For implementing the first step of Algorithm 22, we set $\boldsymbol{b} = \mathbf{0}$, $C = -\boldsymbol{I}_2$, and $N = 2000$. For completing the second step, the Cholesky decomposition D, of Σ and its inverse, are $D = [(1, 4/5)^\top, (0, 3/5)^\top]$ and $D^{-1} = [(1, -4/3)^\top, (0, 5/3)^\top]$, respectively. Hence, for $\boldsymbol{\mu} = (0, 0)^\top$, we have $\boldsymbol{\alpha} = D^{-1}\boldsymbol{\mu} = (0, 0)^\top$ and $E = CD = [(-1, -4/5)^\top, (0, -3/5)^\top]$ where $E_{-1} = (0, -3/5)^\top$ and $E_{-2} = (-1, -4/5)^\top$. Moreover,

$$\begin{aligned} T_1 &= \left\{ z_1 \in \mathbb{R} \middle| \begin{bmatrix} -1 \\ -\frac{4}{5} \end{bmatrix} z_1 \leq - \begin{bmatrix} 0 \\ -\frac{3}{5} \end{bmatrix} z_2 \right\}, \\ &= \left\{ z_1 \in \mathbb{R} \middle| z_1 \geq 0, z_1 \geq -\tfrac{3}{4} z_2 \right\}, \\ &= \left[\max\left\{ 0, -\tfrac{3}{4} z_2 \right\}, \infty \right), \end{aligned} \tag{5.91}$$

and

$$\begin{aligned} T_2 &= \left\{ z_2 \in \mathbb{R} \middle| \begin{bmatrix} 0 \\ -\frac{3}{5} \end{bmatrix} z_2 \leq - \begin{bmatrix} -1 \\ -\frac{4}{5} \end{bmatrix} z_1 \right\}, \\ &= \left\{ z_2 \in \mathbb{R} \middle| z_2 \geq -\tfrac{4}{3} z_1 \right\}, \\ &= [-\tfrac{4}{3} z_1, \infty). \end{aligned} \tag{5.92}$$

Using (5.91), (5.92), and $\boldsymbol{z}^{(0)} = (1, 1)^\top$ the third step of the Algorithm 22 is complete. Finally, the required full conditionals in the seventh step of this algorithm are given as follows.

$$\begin{aligned} Z_1 | (\boldsymbol{Z}_{-1} = \boldsymbol{z}_{-1}) &\sim \mathcal{TN}_{T_1}(\mu_1, 1), \\ Z_2 | (\boldsymbol{Z}_{-2} = \boldsymbol{z}_{-2}) &\sim \mathcal{TN}_{T_2}\Big(-\frac{4}{3}\mu_1 + \frac{5}{3}\mu_2, 1 \Big). \end{aligned}$$

The `R` code for implementing Method 2, when $\boldsymbol{\mu} = (0, 0)^\top$, is given as follows.

```
N <- 20000 # number of Gibbs sampler iterations
Z <- matrix(0, nrow = N, ncol = 2)
Mu <- c(0, 0) # mean vector can be changed arbitrarily
Sigma <- matrix( c(1, 0.8, 0.8, 1), nrow = 2, ncol = 2)
C <- -diag(2)
D0 <- t( chol(Sigma) )
D0inv <- solve( D0 )
E <- C%*%D0
alpha <- D0inv%*%Mu
Z[1, ] <- rep(1, 2) # vector of initial values
j <- 1
 while(j < N)
 {
  Z[j + 1, 1] <- rtnorm(1, alpha[1], 1, max(0, -3/4*Z[j, 2]), Inf)
  Z[j + 1, 2] <- rtnorm(1, alpha[2], 1, -4/3*Z[j + 1, 1], Inf)
  j <- j + 1
 }
X <- Z%*%t(D0) # X is vector of realizations following truncated
               # multivariate Gaussian distribution
```

■

Bibliography

Milton Abramowitz and Irene A Stegun. Handbook of mathematical functions with formulas, graphs, and mathematical tables, volume 55. US Government printing office, 1948.

Hirotugu Akaike. A new look at the statistical model identification. IEEE Transactions on Automatic Control, 19(6):716–723, 2003.

James H Albert and Siddhartha Chib. Bayesian analysis of binary and polychotomous response data. Journal of the American Statistical Association, 88(422):669–679, 1993.

Theodore W Anderson and Donald A Darling. A test of goodness of fit. Journal of the American Statistical Association, 49(268):765–769, 1954.

RB Arellano-Valle, H Bolfarine, and VH Lachos. Bayesian inference for skew-normal linear mixed models. Journal of Applied Statistics, 34(6):663–682, 2007.

Reinaldo B Arellano-Valle and Adelchi Azzalini. On the unification of families of skew-normal distributions. Scandinavian Journal of Statistics, 33(3):561–574, 2006.

Reinaldo B Arellano-Valle and Marc G Genton. On fundamental skew distributions. Journal of Multivariate Analysis, 96(1):93–116, 2005.

George B Arfken, Hans J Weber, and Frank E Harris. Mathematical Methods for Physicists: A Comprehensive Guide. Academic Press, 2011.

Adelchi Azzalini. A class of distributions which includes the normal ones. Scandinavian Journal of Statistics, pages 171–178, 1985.

Adelchi Azzalini. sn: The skew-normal and related distributions such as the skew-t and the sun. R package version, 2(0), 2022.

Adelchi Azzalini and Adrian W Bowman. A look at some data on the old faithful geyser. Journal of the Royal Statistical Society: Series C (Applied Statistics), 39(3):357–365, 1990.

Ole Barndorff-Nielsen. Exponentially decreasing distributions for the logarithm of particle size. Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences, 353 (1674):401–419, 1977a.

Ole Barndorff-Nielsen. Hyperbolic distributions and distributions on hyperbolae. Scandinavian Journal of Statistics, 5:151–157, 1978.

Ole Eiler Barndorff-Nielsen. Contribution to the discussion of dr cox: The role of significance tests. Scandinavian Journal of Statistics, 4:67–69, 1977b.

Christian Bauer. Value at risk using hyperbolic distributions. Journal of Economics and Business, 52(5):455–467, 2000.

Richard A. Becker, John M. Chambers, and Allan R. Wilks. The New S Language. Chapman & Hall, London, 1988.

James O Berger, José M Bernardo, and Dongchu Sun. The formal definition of reference priors. The Annals of Statistics, 37(2):905–938, 2009.

Jose M Bernardo. Reference posterior distributions for bayesian inference. Journal of the Royal Statistical Society Series B: Statistical Methodology, 41(2):113–128, 1979.

José M Bernardo. Reference analysis. Handbook of Statistics, 25:17–90, 2005.

Marta Blangiardo and Michela Cameletti. Spatial and Spatio-temporal Bayesian Models with R-INLA. Wiley, Chichester, West Sussex, United Kingdom, 1st edition, 2015. ISBN 978-1-118-32655-8. URL `https://eu.wiley.com/WileyCDA/WileyTitle/productCd-1118326555.html`.

Victor A. Bloomfield. Using R for Numerical Analysis in Science and Engineering. Chapman & Hall/CRC, 2014. ISBN 978-1439884485. URL `http://www.crcpress.com/product/isbn/9781439884485`.

Zdravko I Botev. The normal law under linear restrictions: simulation and estimation via minimax tilting. Journal of the Royal Statistical Society Series B: Statistical Methodology, 79(1):125–148, 2017.

George EP Box and Mervin E Muller. A note on the generation of random normal deviates. The Annals of Mmathematical Statistics, 29(2):610–611, 1958.

George EP Box and George C Tiao. Bayesian Inference in Statistical Analysis. John Wiley & Sons, 2011.

DJ Buckle. Bayesian inference for stable distributions. Journal of the American Statistical Association, 90(430):605–613, 1995.

George Casella. An introduction to empirical bayes data analysis. The American Statistician, 39(2):83–87, 1985.

John M. Chambers. Programming with Data. Springer, New York, 1998. ISBN 978-0-387-98503-9.

John M Chambers. Graphical Methods for Data Analysis. Chapman and Hall/CRC, 2018.

John M. Chambers. S, R, and Data Science. The R Journal, 12(1):462–476, 2020. doi: 10.32614/RJ-2020-028. URL `https://doi.org/10.32614/RJ-2020-028`.

John M. Chambers and Trevor J. Hastie. Statistical Models in S. Chapman & Hall, London, 1992. ISBN 9780412830402.

John M Chambers, Colin L Mallows, and BW4159820341 Stuck. A method for simulating stable random variables. Journal of the American Statistical Association, 71(354):340–344, 1976.

KE Chernin and IA Ibragimov. On the unimodality of stable laws. Teor. Veroyatnost. i Primenen, 4:453–456, 1959.

A Clifford Cohen, Betty Jones Whitten, and Yihua Ding. Modified moment estimation for the three-parameter weibull distribution. Journal of Quality Technology, 16(3):159–167, 1984.

Edmund A Cornish. The multivariate t-distribution associated with a set of normal sample deviates. Australian Journal of Physics, vol. 7, p. 531, 7:531–542, 1954.

Harald Cramér. On the composition of elementary errors: First paper: Mathematical deductions. Scandinavian Actuarial Journal, 1928(1):13–74, 1928.

Germund Dahlquist and Åke Björck. Numerical Methods. Courier Corporation, 2003.

Gaston Darboux. Mémoire sur l'approximation des fonctions de très-grands nombres, et sur une classe étendue de développements en série. Journal de Mathématiques pures et appliquées, 4:5–56, 1878.

Gergely Daróczi. Mastering Data Analysis with R. Packt Publishing, 9 2015. ISBN 9781783982028. URL https://www.packtpub.com/product/mastering-data-analysis-with-r/9781783982028.

Vikram Dayal. An Introduction to R for Quantitative Economics: Graphing, Simulating and Computing. Springer, 2015. ISBN 978-81-322-2340-5. URL https://www.springer.com/978-81-322-2340-5.

Arthur P Dempster, Nan M Laird, and Donald B Rubin. Maximum likelihood from incomplete data via the em algorithm. Journal of the Royal Statistical Society. Series B (Methodological), pages 1–38, 1977.

Luc Devroye. Non-Uniform Random Variate Generation. Springer, New York, 1986a.

Luc Devroye. Non-Uniform Random Variate Generation. Springer-Verlag, Berlin, Germany, 1986b.

Luc Devroye. Random variate generation for exponentially and polynomially tilted stable distributions. ACM Transactions on Modeling and Computer Simulation (TOMACS), 19(4): 1–20, 2009.

Joseph L Doob. Application of the theory of martingales. Le calcul des probabilites et ses applications, pages 23–27, 1949.

Zvi Drezner. Computation of the multivariate normal integral. ACM Transactions on Mathematical Software (TOMS), 18(4):470–480, 1992.

Zvi Drezner. Computation of the trivariate normal integral. Mathematics of Computation, 62 (205):289–294, 1994.

Zvi Drezner and George O Wesolowsky. On the computation of the bivariate normal integral. Journal of Statistical Computation and Simulation, 35(1-2):101–107, 1990.

Ernst Eberlein and Ulrich Keller. Hyperbolic distributions in finance. Bernoulli, pages 281–299, 1995.

Carmen Fernández and Mark FJ Steel. Multivariate student-t regression models: Pitfalls and inference. Biometrika, 86(1):153–167, 1999.

Ronald A Fisher. The use of multiple measurements in taxonomic problems. Annals of Eugenics, 7.2:179–188, 1936.

Florence Forbes and Darren Wraith. A new family of multivariate heavy-tailed distributions with variable marginal amounts of tailweight: application to robust clustering. Statistics and Computing, 24(6):971–984, 2014.

Mama Foupouagnigni and Wolfram Koepf. Orthogonal Polynomials. Springer, 2018.

John Fox. An R and S-Plus Companion to Applied Regression. Sage, 2002.

John Fox and Sanford Weisberg. An R Companion to Applied Regression. Sage publications, 2018.

Chris Fraley and Adrian E Raftery. How many clusters? which clustering method? answers via model-based cluster analysis. The computer journal, 41(8):578–588, 1998.

Chris Fraley and Adrian E Raftery. Model-based clustering, discriminant analysis, and density estimation. Journal of the American statistical Association, 97(458):611–631, 2002.

James Gareth, Witten Daniela, Hastie Trevor, and Tibshirani Robert. An Introduction to Statistical Learning: with Applications in R. Spinger, 2013.

HI Gassmann. Multivariate normal probabilities: implementing an old idea of plackett's. Journal of Computational and Graphical Statistics, 12(3):731–752, 2003.

Alan E Gelfand and Adrian FM Smith. Sampling-based approaches to calculating marginal densities. Journal of the American Statistical Association, 85(410):398–409, 1990.

Andrew Gelman. Prior distributions for variance parameters in hierarchical models. Bayesian Analysis, 1(3):551–533, 2006.

Andrew Gelman, John B Carlin, Hal S Stern, and Donald B Rubin. Bayesian Data Analysis. Chapman and Hall/CRC, 1995.

Stuart Geman and Donald Geman. Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. IEEE Transactions on Pattern Analysis and Machine Intelligence, PAMI-6(6):721–741, 1984.

Marc G Genton, Yanyuan Ma, and Huiyan Sang. On the likelihood function of gaussian max-stable processes. Biometrika, 98(2):481–488, 2011.

Alan Genz. Numerical computation of multivariate normal probabilities. Journal of Computational and Graphical Statistics, 1(2):141–149, 1992.

Alan Genz. Numerical computation of rectangular bivariate and trivariate normal and t probabilities. Statistics and Computing, 14(3):251–260, 2004.

Alan Genz and Frank Bretz. Numerical computation of multivariate t-probabilities with application to power calculation of multiple contrasts. Journal of Statistical Computation and Simulation, 63(4):103–117, 1999.

John Geweke. Efficient simulation from the multivariate normal and student-t distributions subject to linear constraints and the evaluation of constraint probabilities. In Computing science and statistics: Proceedings of the 23rd symposium on the interface, volume 571, page 578. Fairfax, Virginia: Interface Foundation of North America, Inc, 1991.

Amparo Gil, Javier Segura, and Nico M Temme. Numerical Methods for Special Functions. SIAM, 2007.

Walter R Gilks and Pascal Wild. Adaptive rejection sampling for gibbs sampling. Journal of the Royal Statistical Society: Series C (Applied Statistics), 41(2):337–348, 1992.

Simon Godsill and Ercan E Kuruoglu. Bayesian inference for time series with heavy-tailed symmetric α-stable noise processes. Proc. Applications of heavy tailed distributions in economics, engineering and statistics, 1999.

Gene H Golub and John H Welsch. Calculation of gauss quadrature rules. Mathematics of Computation, 23(106):221–230, 1969.

Bettina Grün and Kurt Hornik. Modelling human immunodeficiency virus ribonucleic acid levels with finite mixtures for censored longitudinal data. Journal of the Royal Statistical Society Series C: Applied Statistics, 61(2):201–218, 2012.

Shanti S Gupta. Probability integrals of multivariate normal and multivariate t1. The Annals of Mathematical Statistics, pages 792–828, 1963.

Frank E. Harrell. Regression Modeling Strategies, with Applications to Linear Models, Survival Analysis and Logistic Regression. Springer, 2001. ISBN 978-3-319-19425-7. URL `https://hbiostat.org/doc/rms/`.

John A Hartigan and Manchek A Wong. Algorithm as 136: A k-means clustering algorithm. Journal of the Royal Statistical Society. Series c (Applied Statistics), 28(1):100–108, 1979.

Trevor Hastie, Robert Tibshirani, and Jerome Friedman. The elements of statistical learning: Data mining, inference, and prediction, 2017.

W Keith Hastings. Monte carlo sampling methods using markov chains and their applications. Biometrika, 57(1):97–109, 1970.

Daojiang He, Dongchu Sun, and Lei He. Objective Bayesian Analysis for the Student-t Linear Regression. Bayesian Analysis, 16(1):129 – 145, 2021.

Norbert Henze. A probabilistic representation of the'skew-normal'distribution. Scandinavian Journal of Statistics, pages 271–275, 1986.

James P Hobert and Christian P Robert. A mixture representation of π with applications in markov chain monte carlo and perfect sampling. Annals of Applied Probability, pages 1295–1305, 2004.

Marius Hofert. Sampling exponentially tilted stable distributions. ACM Transactions on Modeling and Computer Simulation (TOMACS), 22(1):1–11, 2011.

Wolfgang Hörmann and Josef Leydold. Generating generalized inverse gaussian random variates. Statistics and Computing, 24:547–557, 2014.

Torsten Hothorn and Brian S. Everitt. A Handbook of Statistical Analyses Using R. Chapman & Hall/CRC Press, Boca Raton, Florida, USA, 3rd edition, 2014. ISBN 978-1-4822-0458-2. URL `http://www.crcpress.com/product/isbn/9781482204582`.

Lawrence Hubert and Phipps Arabie. Comparing partitions. Journal of Classification, 2:193–218, 1985.

Ross Ihaka and Robert Gentleman. R: A language for data analysis and graphics. Journal of Computational and Graphical Statistics, 5(3):299–314, 1996.

Edwin T Jaynes. Prior probabilities. IEEE Transactions on Systems Science and Cybernetics, 4(3):227–241, 1968.

Harold Jeffreys. An invariant form for the prior probability in estimation problems. Proceedings of the Royal Society of London. Series A. Mathematical and Physical Sciences, 186(1007): 453–461, 1946.

Adam M Johansen, Ludger Evers, and N Whiteley. Monte carlo methods. International Encyclopedia of Education, pages 296–303, 2010.

Norman L Johnson, Samuel Kotz, and Narayanaswamy Balakrishnan. Continuous Univariate Distributions, volume 1. John Wiley & Sons, 1995.

Norman Lloyd Johnson, Samuel Kotz, and Narayanaswamy Balakrishnan. Continuous Multivariate Distributions. John Wiley & Sons New York, 1972.

Bent Jorgensen. Statistical properties of the generalized inverse Gaussian distribution, volume 9. Springer Science & Business Media, 2012.

David Kahaner, Cleve Moler, and Stephen Nash. Numerical Methods and Software. Prentice-Hall, Inc., 1989.

M. Kanter. Stable densities under change of scale and total variation inequalities. Annals of Probability, 3:4, 1975.

Dan E. Kelley. Oceanographic Analysis with R. Springer-Verlag, New York, 2018. ISBN 978-1-4939-8842-6. URL `https://www.springer.com/us/book/9781493988426`.

Christine Keribin. Consistent estimation of the order of mixture models. Sankhyā: The Indian Journal of Statistics, Series A, pages 49–66, 2000.

Albert J Kinderman and John F Monahan. Computer generation of random variables using the ratio of uniform deviates. ACM Transactions on Mathematical Software (TOMS), 3(3): 257–260, 1977.

Matthias Kohl. Introduction to statistical data analysis with R. bookboon.com, London, 2015. ISBN 978-87-403-1123-5.

Samuel Kotz and Saralees Nadarajah. Multivariate t-Distributions and Their Applications. Cambridge university press, 2004.

Samuel Kotz, Tomasz Kozubowski, and Krzystof Podgorski. The Laplace Distribution and Generalizations: A Revisit With Applications To Communications, Economics, Engineering, and Finance. Springer Science & Business Media, 2012.

Debasis Kundu, Narayanaswamy Balakrishnan, and Ahad Jamalizadeh. Bivariate birnbaum–saunders distribution and associated inference. Journal of Multivariate Analysis, 101(1): 113–125, 2010.

Sharon Lee and Geoffrey J McLachlan. Finite mixtures of canonical fundamental skew t distributions: The unification of the restricted and unrestricted skew t-mixture models. Statistics and Computing, 26(3):573–589, 2016.

Lawrence Leemis. Learning Base R. Lightning Source, 2016. ISBN 978-0-9829174-8-0. URL `https://www.amazon.com/Learning-Base-Lawrence-Mark-Leemis/dp/0982917481`.

Hubert W Lilliefors. On the kolmogorov-smirnov test for normality with mean and variance unknown. Journal of the American Statistical Association, 62(318):399–402, 1967.

Brunero Liseo and Antonio Parisi. Bayesian inference for the multivariate skew-normal model: A population monte carlo approach. Computational Statistics & Data Analysis, 63:125–138, 2013.

Jun S Liu and Jun S Liu. Monte Carlo Strategies in Scientific Computing, volume 10. Springer, 2001.

Marco J Lombardi. Bayesian inference for α-stable distributions: A random walk mcmc approach. Computational Statistics & Data Analysis, 51(5):2688–2700, 2007.

Mohsen Maleki, Darren Wraith, and Reinaldo B Arellano-Valle. Robust finite mixture modeling of multivariate unrestricted skew-normal generalized hyperbolic distributions. Statistics and Computing, 29:415–428, 2019.

George Marsaglia and Thomas A Bray. A convenient method for generating normal variables. SIAM Review, 6(3):260–264, 1964.

JI McCool. Inferential techniques for weibull populations. aerospace research laboratories report. Technical report, ARL TR 74-0180, Wright-Patterson AFB, Ohio, 1974.

Alexander J McNeil, Rüdiger Frey, and Paul Embrechts. Quantitative risk management: concepts, techniques and tools-revised edition. Princeton University Press, 2015.

Paul D McNicholas. Mixture Model-Based Classification. Chapman and Hall/CRC, 2016.

Paul D McNicholas and T Brendan Murphy. Model-based clustering of longitudinal data. Canadian Journal of Statistics, 38(1):153–168, 2010.

Paul David Mcnicholas and Thomas Brendan Murphy. Parsimonious gaussian mixture models. Statistics and Computing, 18(3):285–296, 2008.

Robert W Mee and DB Owen. A simple approximation for bivariate normal probabilities. Journal of Quality Technology, 15(2):72–75, 1983.

Nicholas Metropolis, Arianna W Rosenbluth, Marshall N Rosenbluth, Augusta H Teller, and Edward Teller. Equation of state calculations by fast computing machines. The Journal of Chemical Physics, 21(6):1087–1092, 1953.

John R Michael, William R Schucany, and Roy W Haas. Generating random variates using transformations with multiple roots. The American Statistician, 30(2):88–90, 1976.

Tetsuhisa Miwa, AJ Hayter, and Satoshi Kuriki. The evaluation of general non-centred orthant probabilities. Journal of the Royal Statistical Society Series B: Statistical Methodology, 65 (1):223–234, 2003.

Christian E Galarza Morales, Larissa A Matos, Dipak K Dey, and Victor H Lachos. On moments of folded and doubly truncated multivariate extended skew-normal distributions. Journal of Computational and Graphical Statistics, 31(2):455–465, 2022.

Carl N Morris. Parametric empirical bayes inference: theory and applications. Journal of the American Statistical Association, 78(381):47–65, 1983.

Paula M Murray, Ryan P Browne, and Paul D McNicholas. Hidden truncation hyperbolic distributions, finite mixtures thereof, and their application for clustering. Journal of Multivariate Analysis, 161:141–156, 2017.

Steven Murray. Apprendre R en un Jour. SJ Murray, 2017. URL `https://www.amazon.com/dp/B071W6ZJCV/ref=sr_1_1?s=digital-text&ie=UTF8&qid=1496261881&sr=1-1`. Ebook.

HKT Ng, Debasis Kundu, and N Balakrishnan. Modified moment estimation for the two-parameter birnbaum–saunders distribution. Computational Statistics & Data Analysis, 43 (3):283–298, 2003.

Deborah Nolan and Terry Speed. Stat Labs: Mathematical Statistics Through Applications. Springer Texts in Statistics. Springer, 2000. ISBN 978-0-387-22743-6. URL `https://www.stat.berkeley.edu/users/statlabs/`.

John Nolan. Stable distributions: Models for Heavy Tailed Data. Birkhauser New York, 2003.

John P Nolan. Parameterizations and modes of stable distributions. Statistics & probability letters, 38(2):187–195, 1998.

John P Nolan. Univariate stable distributions: Models for heavy tailed data. Springer Series in Operations Research and Financial Engineering, 2020.

John P Nolan and Diana Ojeda-Revah. Linear and nonlinear regression with stable errors. Journal of Econometrics, 172(2):186–194, 2013.

Donald B Owen. Tables for computing bivariate normal probabilities. The Annals of Mathematical Statistics, 27(4):1075–1090, 1956.

Antonio Parisi and Brunero Liseo. Objective bayesian analysis for the multivariate skew-t model. Statistical Methods & Applications, 27(2):277–295, 2018.

Mike Patefield. Fast and accurate calculation of owen's t function. Journal of Statistical Software, 5(5):1–25, 2000. doi: 10.18637/jss.v005.i05. URL `https://www.jstatsoft.org/index.php/jss/article/view/v005i05`.

Jose C. Pinheiro and Douglas M. Bates. Mixed-Effects Models in S and S-Plus. Springer, 2000. ISBN 978-0-387-22747-4.

Robin L Plackett. A reduction formula for normal multivariate integrals. Biometrika, 41(3/4): 351–360, 1954.

Yan Qu, Angelos Dassios, and Hongbiao Zhao. Random variate generation for exponential and gamma tilted stable distributions. ACM Transactions on Modeling and Computer Simulation (TOMACS), 31(4):1–21, 2021.

Thomas Rahlf. Data Visualisation with R. Springer International Publishing, New York, 2017. ISBN 978-3-319-49750-1. URL `http://www.datavisualisation-r.com`.

Theodore J Rivlin. Chebyshev Polynomials. Courier Dover Publications, 2020.

Herbert Robbins. An empirical bayes approach to statistics. In Proceedings of the Third Berkeley Symposium on Mathematical Statistics and Probability, volume 1, pages 157–163, 1956.

Christian P Robert, George Casella, Christian P Robert, and George Casella. The metropolis-hastings algorithm. Monte Carlo Statistical Methods, pages 267–320, 2004.

Christian P Robert, George Casella, and George Casella. Introducing Monte Carlo Methods with R, volume 18. Springer, 2010.

Gareth O Roberts and Jeffrey S Rosenthal. Examples of adaptive mcmc. Journal of Computational and Graphical Statistics, 18(2):349–367, 2009.

Gareth O Roberts and Adrian FM Smith. Simple conditions for the convergence of the gibbs sampler and metropolis-hastings algorithms. Stochastic processes and their applications, 49 (2):207–216, 1994.

Gabriel Rodriguez-Yam, Richard A Davis, and Louis L Scharf. Efficient gibbs sampling of truncated multivariate normal with application to constrained linear regression. Unpublished manuscript, 2004.

Sheldon M Ross. Simulation. academic press, 2022.

G. Samorodnitsky and Murad. S. Taqqu. Stable Non-Gaussian Random Processes: Stochastic Models with Infinite Variance. Chapman & Hall, New York, 1994.

Gideon Schwarz et al. Estimating the dimension of a model. The Annals of Statistics, 6(2): 461–464, 1978.

Luca Scrucca, Michael Fop, Thomas Brendan Murphy, and Adrian E. Raftery. mclust 5: clustering, classification and density estimation using gaussian finite mixture models. The R Journal, 8(1):205–233, 2016. URL https://journal.r-project.org/archive/2016-1/scrucca-fop-murphy-etal.pdf.

Jie Shen, Tao Tang, and Li-Lian Wang. Spectral Methods: Algorithms, Analysis and Applications, volume 41. Springer Science & Business Media, 2011.

William Fleetwood Sheppard and Karl Pearson. On the calculation of the double integral expressing normal correlation. Transactions of the Cambridge Philosophical Society, 19:23–69, 1900.

Galen R Shorack and Jon A Wellner. Empirical Processes with Applications to Statistics. SIAM, 2009.

Jonathan Shuster. On the inverse gaussian distribution function. Journal of the American Statistical Association, 63(324):1514–1516, 1968.

David Slepian. The one-sided barrier problem for gaussian noise. Bell System Technical Journal, 41(2):463–501, 1962.

NM Steen, GD Byrne, and EM Gelbard. Gaussian quadratures for the integrals?? 0 exp (-x 2) f (x) dx and? b 0 exp (-x 2) f (x) dx. Mathematics of Computation, pages 661–671, 1969.

Michael A Stephens. Tests based on edf statistics. In Goodness-of-fit-techniques, pages 97–194. Routledge, 2017.

C. Sun. Empirical Research in Economics: Growing up with R. Pine Square, Starkville, Mississippi, USA, 1st edition, 2015. ISBN 978-0-9965854-0-8. URL https://www.amazon.com/Empirical-Research-Economics-Changyou-Sun/dp/0996585400/ref=aag_m_pw_dp?ie=UTF8&m=A1TZL30UWYSSR8.

Mahdi Teimouri. Fast bayesian inference for birnbaum-saunders distribution. Computational Statistics, 38(2):569–601, 2023.

Mahdi Teimouri and Sanjoy K Sinha. Inference for logistic regression with covariates subject to limit of detection and measurement error. METRON, 82(2):161–182, 2024.

Mahdi Teimouri, Jeffrey W Doser, and Andrew O Finley. Forestfit: An r package for modeling plant size distributions. Environmental Modelling & Software, 131:104668, 2020.

Mahdi Teimouri, Seyed Mehdi Hoseini, and Maria Sabrina Greco. Bayesian inference for amplitude distribution with application to radar clutter. Digital Signal Processing, 148:104443, 2024.

Terry M. Therneau and Patricia M. Grambsch. Modeling Survival Data: Extending the Cox Model. Statistics for Biology and Health. Springer, 2000. ISBN 978-1-4757-3294-8.

HC Thode. Testing for normality, 2002.

GE Thomas. Remark asr 65: A remark on algorithm as76: An integral useful in calculating non-central t and bivariate normal probabilities. Journal of the Royal Statistical Society. Series C (Applied Statistics), 35(3):310–312, 1986.

Cristina Tortora, Ryan P Browne, Aisha ElSherbiny, Brian C Franczak, and Paul D McNicholas. Model-based clustering, classification, and discriminant analysis using the generalized hyperbolic distribution: Mixghd r package. Journal of Statistical Software, 98(1):1–24, 2021.

William N. Venables and Brian D. Ripley. S Programming. Springer, New York, 2000. ISBN 978-0-387-21856-4. URL `http://www.stats.ox.ac.uk/pub/MASS3/Sprog/`.

Jonathan C Wakefield, Alan E Gelfand, and Adrian FM Smith. Efficient generation of random variates via the ratio-of-uniforms method. Statistics and Computing, 1:129–133, 1991.

Marc Weibel, Wolfgang Breymann, and David Lüthi. ghyp: A package on generalized hyperbolic distributions. Manual for the R Package GHYP, 1:1–27, 2020.

R Weron. On the chambers-mallows-stuck method for simulating skewed stable random variables. Statistics and Probability Letters, 28:165–171, 1996.

Richard C Yang, Antel Kozak, and J Harry G Smith. The potential of weibull-type functions as flexible growth curves. Canadian Journal of Forest Research, 8(4):424–431, 1978.

JC Young and Ch E Minder. As 76: An integral useful in calculating non-central t and bivariate normal probabilities. Journal of the Royal Statistical Society Series C: Applied Statistics, 23 (3):455–457, 1974.

Vladimir M Zolotarev. One-dimensional stable distributions, volume 65. American Mathematical Soc., The American Mathematical Society, 1986.