3

INTRODUCTION TO R PACKAGES

An R package can be viewed as a set of functions organized around a topic. For example, finance, econometrics, graphs, machine learning, portfolio optimization, and statistics are topics. Thus, knowledge of R packages could speed up our understanding of the R language. This is especially true for new learners since they don't have to reinvent the wheel. In this chapter, the following topics will be covered:

- Introduction to R packages
- Auto-loaded packages
- Three ways to install R packages
- Three ways to load an R package
- How to find help for a given package
- Loaded vs. preinstalled packages
- How to find a list of all R packages
- Searching for R packages by using our R data sets
- Checking whether an R package is loaded
- How to find the manual for a given R package
- Yahoo!Finance: `yfR` package
- What is a Task View?
- How do you install a hundred R packages included in a Task View?

3.1 WHAT ARE PACKAGES?

R packages are not just collections of functions, datasets, and other resources. They are collaborative tools bundled together for specific purposes, making it easier for users to extend R's functionality. These packages serve as modular units of code that can be easily shared, installed, and used in R projects. They play a crucial role in enhancing R's capabilities by providing specialized tools for tasks such as data manipulation, statistical

analysis, data visualization, machine learning, and more. Notable examples of R packages include "dplyr" and "tidyr" for data wrangling, "ggplot2" for data visualization, "caret" for machine learning, and "forecast" for time series analysis. The rich ecosystem of R packages fosters a collaborative environment within the R community, empowering analysts and data scientists to leverage diverse tools and accelerate development.
R packages not only extend functionality but also play a crucial role in ensuring the reproducibility of analyses. Users can confidently share their code and the specific package versions, knowing others can reproduce the study using the same tools. This modularity and openness make R packages a fundamental aspect of the R programming language, enabling users to tap into vast resources and efficiently address the varied needs of data analysis and statistical modeling.

3.2 DEFAULT PACKAGES

Before doing anything, when launching R, several R packages will be uploaded automatically. The search function can be used to find the names of those R packages.

```
> search()
[1] ".GlobalEnv"        "package:stats"     "package:graphics"
[4] "package:grDevices" "package:utils"     "package:datasets"
[7] "package:methods"   "Autoloads"         "package:base"
>
```

The stats R package includes many statistical functions, such as max, min, mean, and sd (standard deviation). The R package datasets contains many datasets. Later in the chapter, we will show how to find the contents of a given R package. Currently, students learn that some R packages are automatically loaded when they launch R, and the function used is search().

3.3 FIRST WAY TO INSTALL R PACKAGES

We can use install.packages() install R packages. Note that the word packages is plural. For example, we are interested in an R package called qrcode.

```
> install.packages("qrcode")    # Method I
```

After pressing Enter, we will see the following image.

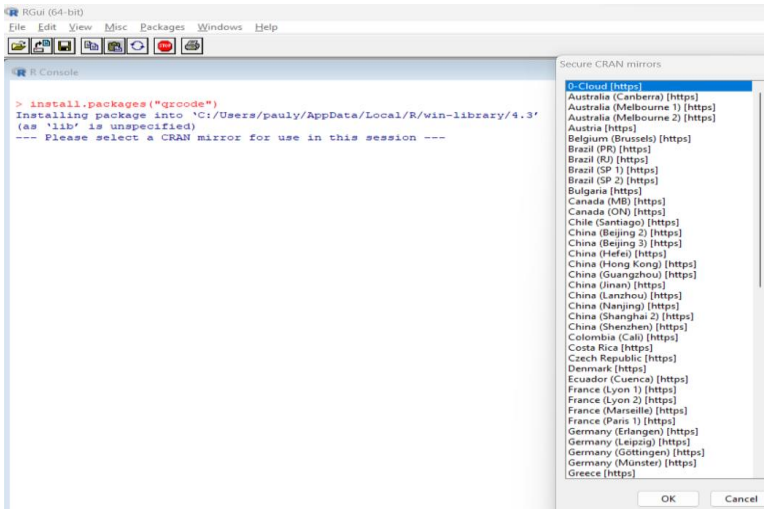


Figure 3-1 A list of potential locations (mirrors)

The right panel shows us a list of mirrors (servers). Just choose one near your location. For example, we can select `USA (TN) [https]`, as shown below. Note that the mirror (location) must be chosen once when installing several packages simultaneously.

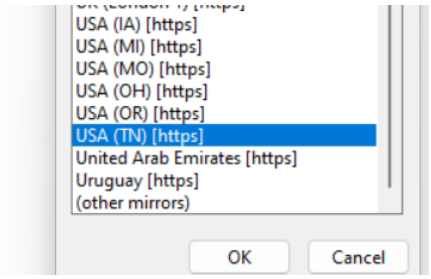


Figure 3-2 Choosing a nearby location

After hitting [OK], several R packages will be installed successfully, as shown below.

```
> install.packages("qrcode")
Installing package into 'C:/Users/pauly/AppData/Local/R/win-library/4.3'
(as 'lib' is unspecified)
--- Please select a CRAN mirror for use in this session ---
also installing the dependency 'assertthat'

trying URL 'https://mirrors.nics.utk.edu/cran/bin/windows/contrib/4.3/assertthat_0.2.1.zip'
Content type 'application/zip' length 54857 bytes (53 KB)
downloaded 53 KB

trying URL 'https://mirrors.nics.utk.edu/cran/bin/windows/contrib/4.3/qrcode_0.2.2.zip'
Content type 'application/zip' length 125940 bytes (122 KB)
downloaded 122 KB

package 'assertthat' successfully unpacked and MD5 sums checked
package 'qrcode' successfully unpacked and MD5 sums checked

The downloaded binary packages are in
        C:\Users\pauly\AppData\Local\Temp\RtmpcjquY8\downloaded_packages
>
```

Figure 3-3 The result of installing the "`qrcode`" R package

When installing one specific package, other packages might be installed simultaneously. For example, the above image shows that two packages were downloaded and installed. In a later section, we will explain the concept of package dependency. Later in the chapter, we will discuss the second and third methods of installing R packages.

3.4 SECOND WAY TO INSTALL AN R PACKAGE

The second way involves using a menu bar.

Step 1: Click "Packages" on the menu bar
Step 2: Choose a mirror location
Step 3: Find a desired package to install it

For example, if we are interested in the `fImport` R package, we can click it, as shown below.

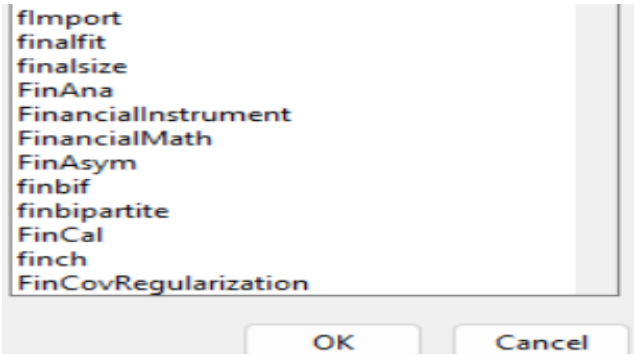


Figure 3-4 How to install an R package called "`fImport`"

Ordinary users use the third method to install R packages. We will touch on it briefly at the end of this chapter to ensure completeness.

3.5 RSTUDIO PACKAGE INSTALLATION

In RStudio, we click "Packages" in the menu bar on the right-hand side, as shown below.

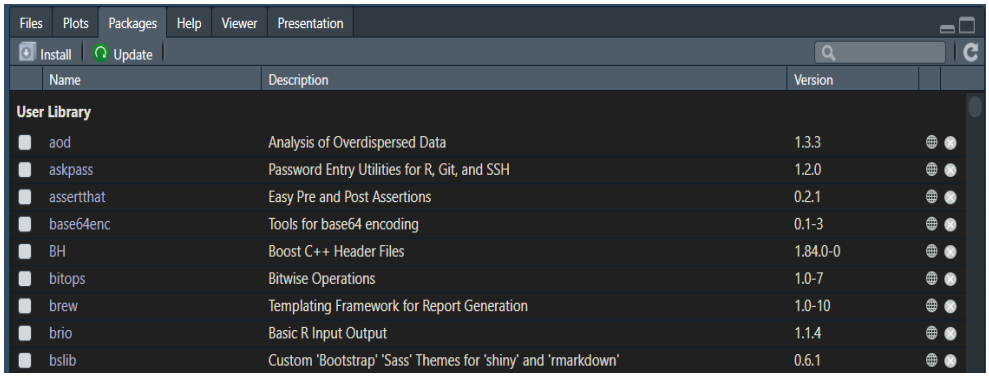


Figure 3-5 RStudio: How to install R packages

To install a package, we select it from the menu above and click "Install" in the menu bar (on the left-hand side).

3.6 LISTS OF ALL R PACKAGES

Finding a list of all available R packages is a good idea. To do so, we go to the R home website at https://r-project.org. Then, choose CRAN on the left-hand side under "Download", at https://cran.r-project.org/mirrors.html. Choose a mirror near your location. Using Duke University as an example; its corresponding website is at https://archive.linux.duke.edu/cran/. After clicking "`Packages`", we will see two lists of all R packages. The two entries are shown below.

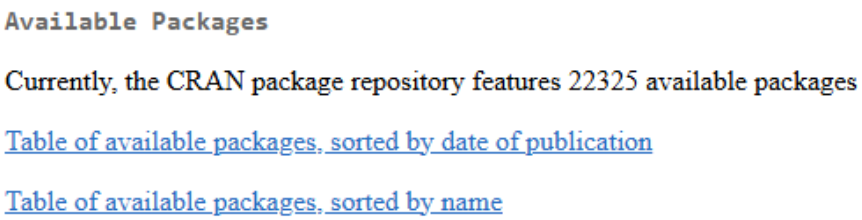


Figure 3-6: The two lists of R packages are available

As of 4/16/2025, there are 22,325 R packages available. The first list is sorted by date, while the second list is sorted by name. The top part of the first list is shown below.

Available CRAN Packages By Date of Publication

Date	Package	Title
2025-04-16	adespatial	Multivariate Multiscale Spatial Analysis
2025-04-16	budgetIVr	Partial Identification of Causal Effects with Mostly Invalid Instruments
2025-04-16	CptNonPar	Nonparametric Change Point Detection for Multivariate Time Series
2025-04-16	datana	Datasets and Functions to Accompany Analisis De Datos Con R

Figure 3-7 The top part of the list of R packages sorted by date

Later in the chapter, we will explain another way to search for R packages. The delay is due to students having to issue specific R codes to activate our related functions and data sets.

3.7 SECOND WAY TO FIND ALL R PACKAGES

The second way to find all R packages needs just one line of R code.

```
x<-available.packages(repos ="http://cran.us.r-project.org")
df<-data.frame(x)
dim(df)
   [1] 22194    17
>
```

From the above code, we know there are 20,755 R packages (as of 5/27/2024). The dataset has 17 columns, as shown below.

```
> colnames(df)
 [1] "Package"         "Version"               "Priority"
 [4] "Depends"         "Imports"               "LinkingTo"
 [7] "Suggests"        "Enhances"              "License"
[10] "License_is_FOSS" "License_restricts_use" "OS_type"
[13] "Archs"            "MD5sum"            "NeedsCompilation"
[16] "File"           "Repository"
>
```

3.8 THREE WAYS TO LOAD R PACKAGES

There are three ways to load a specific package.

```
> library("qrcode")                      # Method I
> require(qrcode)                        # Method II
# Click "Packages"[on menu],Load packages# Method III
```

Assume that we want to generate a QR code.

```
> library(qrcode)
> a<-qr_code("http://wsj.com")
> plot(a)
```

The related image is shown below.

Figure 3-8 The image of a QR code

However, receiving the following error message is common after issuing the first command to import `fImport`.

```
> library(fImport)
Error in library(fImport) : there is no package called
'fImport'
```

The error message tells us that the `fImport` package has not been preinstalled. The first way to install an R package is to have the following one-line code.

```
> install.packages("fImport")
```

3.9 THE `yfR` PACKAGE

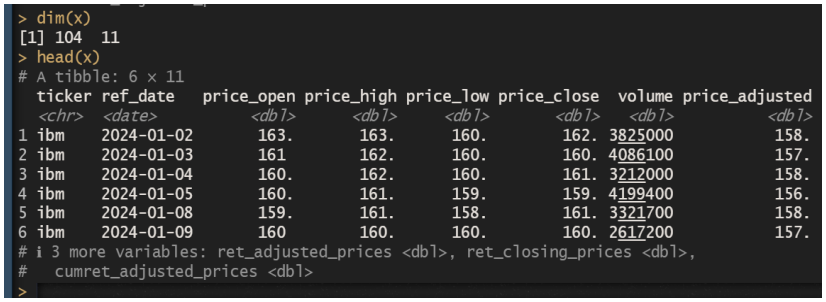
Let's look at the power of one package called `yfR` (Yahoo!Finance using R). To install R, we can use the following code.

```
> install.packages("yfR")
```

Assume that we want to download IBM's daily historical data. The code is given below.

```
library(yfR)
ticker="ibm"
begdate="2024-1-1"
enddate="2024-5-31"
x=yf_get(ticker,begdate,enddate)
head(x)
```

The output is shown below.



```
> dim(x)
[1] 104  11
> head(x)
# A tibble: 6 × 11
  ticker ref_date   price_open price_high price_low price_close  volume price_adjusted
  <chr>  <date>          <dbl>      <dbl>     <dbl>       <dbl>   <dbl>          <dbl>
1 ibm    2024-01-02       163.       163.      160.        162. 3825000           158.
2 ibm    2024-01-03       161        162.      160.        160. 4086100           157.
3 ibm    2024-01-04       160.       162.      160.        161. 3212000           158.
4 ibm    2024-01-05       160.       161.      159.        159. 4199400           156.
5 ibm    2024-01-08       159.       161.      158.        161. 3321700           158.
6 ibm    2024-01-09       160        160.      160.        160. 2617200           157.
# i 3 more variables: ret_adjusted_prices <dbl>, ret_closing_prices <dbl>,
#   cumret_adjusted_prices <dbl>
>
```

Figure 3-9 The dimensions and the first several lines of a data frame

3.10 FINDING INFO ABOUT AN R PACKAGE

There are several ways to find information about a specific R package. The easiest way is to use the `help()` function. Assume we are interested in the qrcode R package. Then, we issue the following R code.

```
>library(qrcode)
>help(package=qrcode)
```

The underlying program launches a related website; see the top part below.

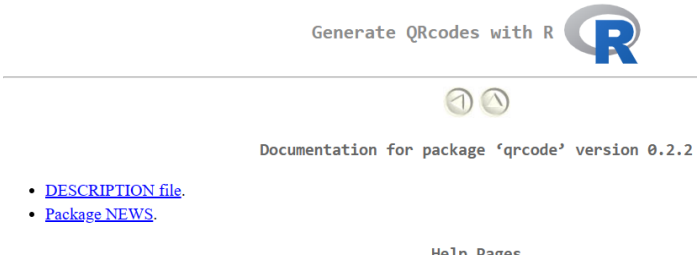


Figure 3-10 The top part of the help for the `QRcode` package

3.11 FINDING THE MANUAL FOR R PACKAGES

For a new R package, many learners prefer a PDF manual. We have the following steps to find and download the “`fImport`” manual.

Step 1: Go to http://www.r-project.org/
Step 2: Click “CRAN”
Step 3: Choose a “mirror”
Step 4: Click “packages” on the left-hand side (under software)
Step 5: Find a specific package using Ctrl-F to search
Step 6: Download the manual in PDF format to your computer

The following image shows the view of the last step.

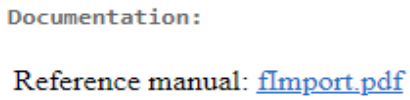
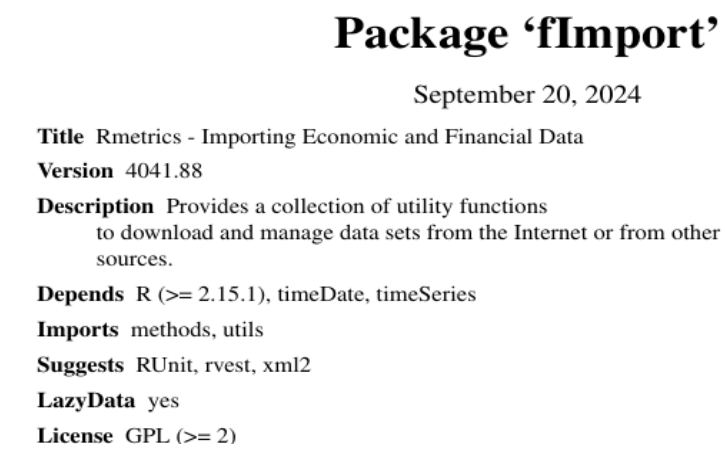


Figure 3-11 The image of the PDF file for the `fImport` package

The top part of the file `fImport.pdf` is shown below.



Package 'fImport'

September 20, 2024

Title Rmetrics - Importing Economic and Financial Data

Version 4041.88

Description Provides a collection of utility functions to download and manage data sets from the Internet or from other sources.

Depends R (>= 2.15.1), timeDate, timeSeries

Imports methods, utils

Suggests RUnit, rvest, xml2

LazyData yes

License GPL (>= 2)

Figure 3-12 The top part of the PDF of the `fImport` R package

3.12 LOADED VS. PREINSTALLED PACKAGES

To find all the loaded packages, we use the `search()` function.

```
> search()
[1] ".GlobalEnv"        "package:stats"     "package:graphics"
[4] "package:grDevices" "package:utils"     "package:datasets"
[7] "package:methods"   "Autoloads"         "package:base"
```

Those seven packages are automatically uploaded when we launch R. We use the `library()` function to find all preinstalled packages.

```
> library()  # show all installed packages
```

The top part is shown below.

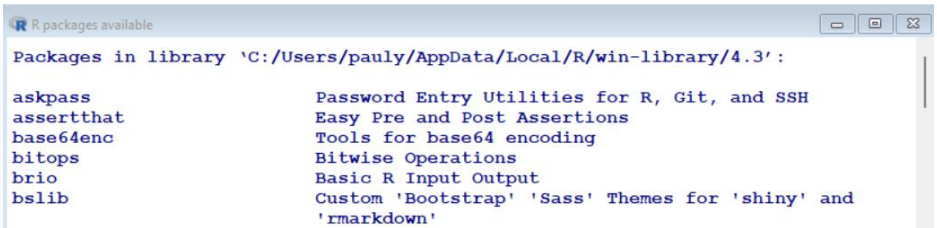


Figure 3-13 The top part of the output after issuing the `library()`

The difference between a loaded package and a pre-installed package is that you can use its related functions if it is loaded. If a package is preinstalled but not loaded, we can use the `library()` function to load it. If a package is not preinstalled, we must install it before loading it.

3.13 PACKAGE DEPENDENCY

In R, package dependencies refer to the relationships between packages, mainly when one package relies on the functions, objects, or capabilities provided by another. When package A depends on or requires another package B, it means that to use the functions or features of package A, you must have package B installed and loaded in your R environment.

There are two main types of dependencies in R packages:

> Imports: If a package imports another package, the functions or objects from the imported package are used within the importing package's code. The importing package will typically include the imported package in its DESCRIPTION file under the "Imports" field. When you install or load the importing package, R will automatically ensure that the imported package is also installed and loaded.
>
> Suggested dependencies are not required for the package's basic functionality but are recommended for additional features. These dependencies are listed in the "Suggests" field in the DESCRIPTION file. Users can choose to install these suggested packages to use the optional features they provide.

Managing package dependencies is crucial for ensuring that your R code runs smoothly and that others can quickly reproduce your analyses. The `install.packages()` function in R automatically installs dependencies, but it's essential to watch for potential conflicts or version issues among packages. The `library()` function is then used to load the required packages into the R session.

When developing R packages, accurately specifying dependencies helps users and other developers understand the code's requirements, facilitating a more seamless, reproducible workflow. The `devtools` package and its `install_deps()` function can be helpful during package development, as they automatically install and load dependencies. If you're doing anything, several R packages will be uploaded when launching R.

3.14 WHAT ARE TASK VIEWS?

CRAN task views aim to guide which CRAN packages are relevant for tasks related to a specific topic. They briefly overview the included packages, which can also be automatically installed using the `ctv` package. The views are intended to be sharply focused

so that it is sufficiently clear which packages should be included (or excluded), and they are not meant to endorse the "best" packages for a given task.

Step 1: Go to http://www.r-project.org/
Step 2: Click “CRAN”
Step 3: Choose a “mirror” (location)
Step 4: Click “Task Views” on the left-hand side

The following image shows the top part of the various Task Views.

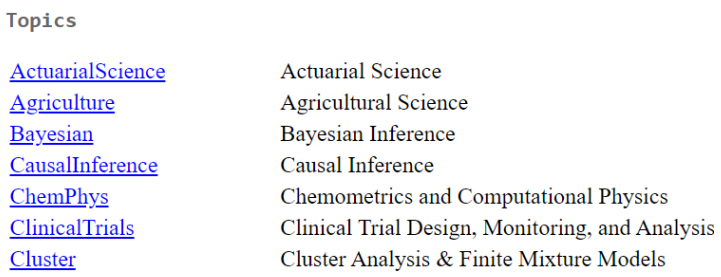


Figure 3-14 The top part of the list of Task-Views

There are 44 Task Views (topics) so far (as of 6/3/2024). The above image shows just the first seven to save space.

3.15 INSTALLING MANY PACKAGES AT ONCE

One Task View might include several dozen or even a hundred packages. For example, the Task View for Finance has several categories: standard regression models, time series, finance, risk management, books, data, and data management. For standard regression, we have the following image.

*

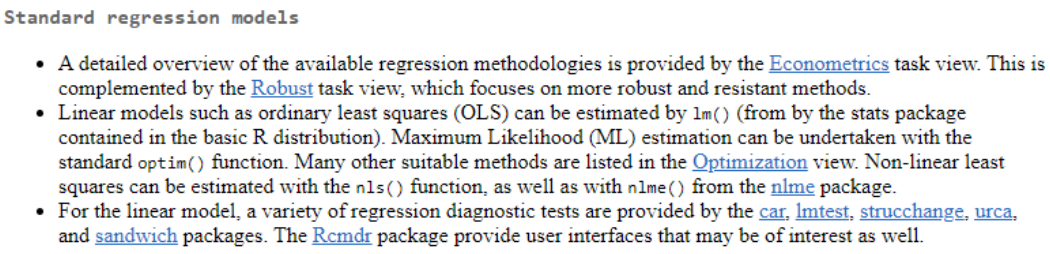


Figure 3-15 The contents of the Task Views

The image above shows 10 related R packages. Now, the question is how to install those R packages included in the Finance Task view. The following three lines of R code can be used to install those R packages simultaneously.

```
>install.packages("ctv")
>library("ctv")
>ctv::install.views("Finance")
```

3.16 COMMON COMMANDS FOR PACKAGES

R packages are the most important 'tools' for using R. We usually list all available packages and then pick those related to finance or those we need. However, the following approach might be more appropriate for a new user: First, find all loaded packages. Then, find out the contents of a loaded package. Finally, search for more information on a specific function's usage and examples. The following table presents the most used commands related to packages.

Table 3.1: Commonly used commands related to R packages

Description	R command
Find out all loaded packages	`>search()`
Find out whether a specific package is loaded or not	`> "package:XML" %in% search()` `[1] TRUE`
Find out all pre-installed packages	`>library()`
	`>.packages(all.available=T)`
Find out whether a specific package is preinstalled	`>"XML" %in% .packages(all.available=T)` `[1] TRUE`
Load a preinstalled package	`>library("fImport")`
	`> require(financial)`
Unload a package	`> detach(package:fImport)` `>detach("package:fImport",unload=TRUE)`
Get information about a package	`>help(package="timeDate")` `>library(help="timeDate")`
Check all available packages.	Step1: http://www.r-project.org/ Step 2: Click "CRAN" Step 3: Choose a location Step 4: Click "packages"

3.17 INFO ABOUT A PACKAGE

We can quickly find the version of a package.

```
loc<-"http://cran.us.r-project.org"
x<-available.packages(repos=loc)
colnames(x)
[1] "Package"         "Version"              "Priority"
[4] "Depends"         "Imports"              "LinkingTo"
```

```
[7] "Suggests"        "Enhances"              "License"
[10]"License_is_FOSS" "License_restricts_use" "OS_type"
[13]"Archs"           "MD5sum"                "NeedsCompilation"
[16]"File"            "Repository"
```

The above lines show the 17 column names.

```
p<-"fImport"
grep(toupper(p),toupper(df$Package))
  [1] 5682
df[grep(toupper(p),toupper(df$Package)),]
```

The related output is shown below.

```
> df[grep(toupper(p),toupper(df$Package)),]
        Package Version Priority                                  Depends        Imports
fImport fImport 4032.87     <NA> R (>= 2.15.1), timeDate, timeSeries methods, utils
        LinkingTo Suggests Enhances    License License_is_FOSS License_restricts_use
fImport      <NA>    RUnit     <NA> GPL (>= 2)            <NA>                  <NA>
        OS_type Archs                          MD5sum NeedsCompilation File
fImport    <NA>  <NA> c2db3b61ba9536d358169150841a27fe               no <NA>
                                  Repository
fImport http://cran.us.r-project.org/src/contrib
```

Figure 3-16 The packages associated with `fImport`

Our current version of `fImport` is 4032.87, and its compatible R version should be more significant than 2.15.1.

3.18 DATA SETS INCLUDED IN A PACKAGE

There are many embedded finance datasets we can use for various exercises. The significant advantage is that we can retrieve those datasets easily without any trouble. For example, the `fImport` package includes the following data sets.

```
library(fImport)
data(package= "fImport ")
```

The following data sets will appear.

```
Data sets in package ‘fImport’:
amexListing   Provider Listing of Symbols and Descriptions
h15Listing    Provider Listing of Symbols and Descriptions
nasdaqListing Provider Listing of Symbols and Descriptions
nyseListing   Provider Listing of Symbols and Descriptions
oandaListing  Provider Listing of Symbols and Descriptions
stoxxListing  Provider Listing of Symbols and Descriptions
swxListing    Provider Listing of Symbols and Descriptions
```

3.19 DEFUNCT FUNCTIONS

Over the years, many functions included in various R packages have been modified or even deprecated. Unfortunately, we still find outdated explanations or examples when searching

online or reading their manuals. For instance, `getFinancials("GE")` was designed to retrieve General Motors' (GM) financial statements. However, we get the following error message after issuing the related code.

```
> library(quantmod)
> getFinancials('GE')
Error: 'getFinancials.google' is defunct.
Google Finance stopped providing data in March 2018.
You could try some of the data sources via Quandl instead.
See help("Defunct") and help("quantmod-defunct")
```

3.20 PACKAGE VERSION ISSUE

This section is optional.

Based on the previous explanation, many packages depend on the functions contained in other packages. This is called a package dependency. In normal circumstances, users would not consider its impact when installing or updating a new package. Occasionally, this kind of dependency causes trouble. Let's look at one example. When installing an R package called 'rattle', use the `install command.packages('rattle')`, you might get the following message:

```
>library(rattle)
  Loading required package: tibble
  Loading required package: bitops
  Rattle: A free graphical interface for data science with R.
  Version 5.5.1 Copyright (c) 2006-2021 Togaware Pty Ltd.
  Type 'rattle()' to shake, rattle, and roll your data.
> rattle()
    Error in rattle() :
    The RGtk2 package is not available but is required.
    Please install the package using, for example:
    Install.packages("RGtk2")
```

This indicates that 'rattle' depends on 'RGtk2'. When installing this package, we get the following warning message.

```
> install.packages('RGtk2')
Installing package into 'C:/Users/Paul Yan/AppData/Local/R/win-library/4.3'
(as 'lib' is unspecified)
Warning message:
package 'RGtk2' is not available for this version of R

A version of this package for your version of R might be available elsewhere,
see the ideas at
https://cran.r-project.org/doc/manuals/r-patched/R-admin.html#Installing-packages
>
```

Figure 3-17 Installing RGtk2

Now, the question is how to solve this problem. There are two issues here.

1) What should we do if this package's version is not comparable with our R?
2) What should we do if this package's version is not comparable with `rattle`?

```
loc<-"http://cran.us.r-project.org"
x<-available.packages(repos=loc)
p<-'RGtk2'
colnames(x)
[1] "Package"          "Version"                "Priority"
[4] "Depends"          "Imports"                "LinkingTo"
[7] "Suggests"         "Enhances"               "License"
[10] "License_is_FOSS" "License_restricts_use" "OS_type"
[13] "Archs"           "MD5sum"              "NeedsCompilation"
[16] "File"            "Repository"
```

Below, let's search for a specific package.

```
df<-data.frame(x)
grep(toupper(p),toupper(df$Package))
integer(0)  # no such a package!!!!
```

The conclusion is that there is no such package as `PGtk2`.

3.21 THE THIRD WAY TO INSTALL R PACKAGES

This section is optional.
The third way is to install a package from a locally saved zip file, so we need to save a zip file on our computers. Below are several steps for installing the "`quantmod`" package from a locally saved zip file.

Step 1: Go to http://www.r-project.org/
Step 2: Choose a mirror server
Step 3: Click the package on the left-hand side
Step 4: Use Ctrl-f to search "quantmod"
Step 5: Download it to our PC
Step 6: Click "Packages" on the menu bar
Step 7: Install a package from our local files (recently downloaded)

3.22 UNABLE TO INSTALL R PACKAGES' ISSUES

This section is optional.
If we observe the following error message, the person trying to install specific R packages does not have the right to write to the library.

```
> utils:::menuInstallPkgs()
Warning in install.packages(NULL, .libPaths()[1L], dependencies = NA,
type = type) :
  'lib = "C:/Program Files/R/R-2.15.2/library"' is not writable
```

```
Error in install.packages(NULL, .libPaths()[1L], dependencies = NA,
type = type) :
  unable to install packages
```

To correct this problem, we change the permission.

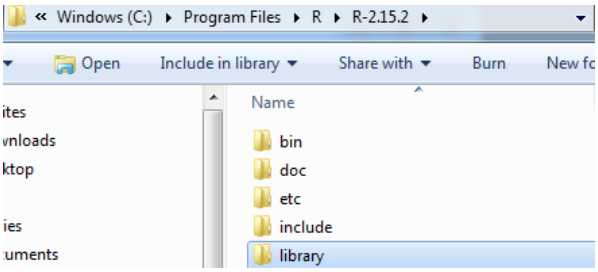


Figure 3-18 The location of the library

Choose “library”, then right-click the mouse and choose Properties.

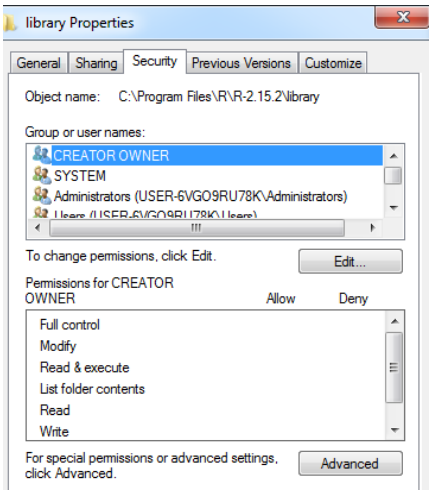


Figure 3-19 The properties of the library

Then we change the permission; see below.

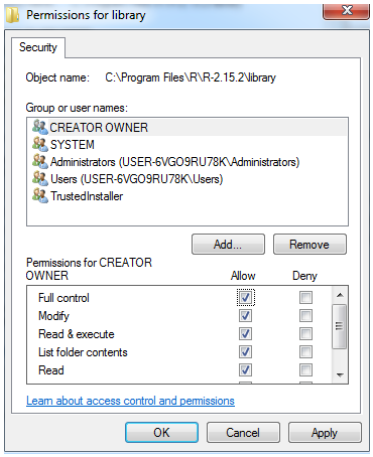


Figure 3-20 The setting of the permissions

3.23 SAME NAME FUNCTIONS

As mentioned earlier, using meaningful function names is a good practice. However, when designing an R package, producers could use many publicly available R packages to save time and avoid reinventing the wheel. One potential issue is that functions with the same names from different packages might exist. In addition, studying functions in others' packages is time-consuming. Functions with the same name will be masked to avoid unnecessary program failures when we import an R package. One example is shown below.

```
> library(fImport)
Loading required package: timeDate
Loading required package: timeSeries
Attaching package: 'timeSeries'
The following objects are masked from 'package:graphics':
    lines, points
```

In the above example, the `lines` and `points` functions exist in `fImport` and `graphics`. When loading the `fImport` package, the `lines` and `points` functions from `fImport` are unavailable. After issuing `help(lines)`, we will get the following information.

Help on topic 'lines' was found in the following packages:

Plot 'timeSeries' objects
(in package timeSeries in library C:/Users/Patron/AppData/Local/Programs/R/R-4.3.3/library)
Add Connected Line Segments to a Plot
(in package graphics in library C:/Users/Patron/AppData/Local/Programs/R/R-4.3.3/library)

Figure 3-21 The result from the help function

3.24 USING THE .libPaths() FUNCTION

This section is optional.
One way to avoid the above issue of administration privileges is to use the `.libPaths()` function. Note that since there is a dot before `libPath()`. To introduce a personal anecdote, we couldn't install any R packages when teaching "Financial Analysis with R" because I didn't have permission. Assume that my memory stick is "e:\". Then, after issuing the following R commands, we can install any R packages we chose.

```
>.libPaths("e:/")
>install.packages("XML")
```

REFERENCES

Holden and Jacobsen, 2022, "Liquidity Measurement Problems in Fast, Competitive Markets: Expensive and Cheap Solutions, " The Journal of Finance, http://datayyy.com/doc_pdf/longVariableNames.pdf.

Task View for Finance on the http://r-project.org platform, https://CRAN.R-project.org/view=Finance

EDGAR Filer Manual - Volume 2 (sec.gov)
https://www.sec.gov/files/edgar/filermanual/efmvol2-c6.pdf

https://stackoverflow.com/questions/44046055/error-in-installing-packages-rgtk2-and-rattle-in-r

Appendix A: After launching R, issue one of the following lines of R code.

```
source("http://datayyy.com/fmr/week2.txt")
```

```
*-------------------------------------------------------*
* Financial Modeling using R (2nd edition)  2026 Yan  *
*-------------------------------------------------------*
*  .c1  R Basics                                        *
*  .c2  R functions                                     *
*  .c3  Introduction to R packages                      *
*  .c4  Data Frame, list, and date                      *
*-------------------------------------------------------*
* >.c4       # go to chapter 3 (a dot in front of c4)  *
* >.uu       # go to utility submenu                    *
* >.fm       # back to this menu                        *
*-------------------------------------------------------*
```

Figure 3-22 The menu for Week 2

Appendix B: For Chapter 3, type `.chapter3` or .c3, as shown below.

```
> .chapter3
function(i=0){
" i  Chapter 3: Introduction to R packages   i  Explanation
  -  ------------------------------------  --  ------------------------------
  1  What is an R package?                  21  R program to get all R packages
  2  Advantages of using R packages         22  link to a list
  3  How to find a list of all R packages   23  location of allRpackages.RData
  4  Manual for a specific package          24       code
  5  Loaded vs. preinstalled packages
  6  Load and unload an R package
  7  Find out all loaded R packages
  8  Find out all preinstalled  R packages
  9  Find out all preinstalled R packages
 10  Method   I to install an R package
 11  Method  II to install an R package
 12  Method III to install an R package
 13  A var has all installed R packages
 14  Is an R package loaded?
 15  Is an R package installed already?
 16  Package description
 17  Functions included in a package
 18  Most useful R commands related to R packages
 19  Videos
 20  Links

 Example #1:>.c3    # see the above list
 Example #2:>.c3()  # see the above list
 Example #3:>.c3(1) # see the first explanation
```

Figure 3-23 The contents of Chapter 3

QUESTIONS

3.1 What is an R package, and why is it useful in R programming?
3.2 Describe the steps to install an R package from CRAN.
3.3 How can you load an installed R package into your R session?
3.4 Explain the purpose of the `library()` function in R.
3.5 What is the difference between the `library()` and `require()` functions in R?
3.6 How can you check which packages are currently loaded in your R session?
3.7 Describe how to update an R package to its latest version.
3.8 Explain how to uninstall an R package.
3.9 What are the essential components of an R package?
3.10 How can you access a specific R package's documentation and help files?
3.11 What is the purpose of the DESCRIPTION file in an R package?
3.12 Give an example of how to list all the functions available in a specific R package.
3.13 How many R packages are available right now?
3.14 Ask ChatGPT how to find help related to a specific package, such as `fImport`.
3.15 Ask ChatGPT how to find the data sets within a given R package, such as `fImport`.
3.16 Ask ChatGPT how to check whether a given R package is preinstalled.
3.17 What is the difference between a loaded package and a preinstalled one?
3.18 How do you find all loaded R packages?
3.19 How do you find all preinstalled R packages?
3.20 How to install an R package?
3.21 How do you find the manual for a specific R package?

3.22 How many R packages are available today?
3.23 How many ways are there to install an R package?
3.24 How many functions are available for "`quantmod`"?
3.25 Download analyst recommendations for IBM, DELL, and MSFT.
3.26 Analyze the results from the above question.
3.27 What was the exchange rate between the US dollar and the Russian Ruble on January 1, 2024?
3.28 Write an R program to put four exchange rates side by side. [e.g., you can use USD/GBP, USD/DEM, USD/FRF, and SD/CNY.] Estimate how strong they are correlated.
3.29 Download the prime rate and USD/GBP. What is the correlation between them?
3.30 What is the correlation coefficient between the prime rate and S&P500 (using the ticker `^GSPC` to retrieve daily data from Yahoo Finance for S&P500)?
3.31 What is wrong with the following code?

```
> require("ctv")
Loading required package: ctv
> install.views("finance")
Error in split.default(X, group) : first argument must be a vector
In addition, a warning message:
In .get_pkgs_from_ctv_or_repos(views = views, coreOnly =
coreOnly,  :
  CRAN task view finance is not available
```

3.32 Find more information about the R package called `yfR`.