

Chapter 2 R/Rstudio basics

2.1 What is R / Rstudio



R은 통계나 생물통계, 유전학을 연구하는 사람들 사이에서 널리 사용되는 오픈소스 프로그래밍 언어입니다. Bell Lab에서 개발한 S 언어에서 유래했으며 엄청나게 많은 라이브러리 (다른 사람들이 만들어 놓은 코드)가 있어서 쉽게 가져다 사용할 수 있습니다. R은 복잡한 수식이나 통계 알고리즘을 간단히 구현하고 사용할 수 있으며 C, C++, Python 등 다른 언어들과의 병행 사용도 가능합니다. 2019년 top five language에 랭크되었으며 이는 빅데이터 증가에 따라 인기가 높아진 것으로 볼 수 있습니다 (참고로 2018년에는 7위).

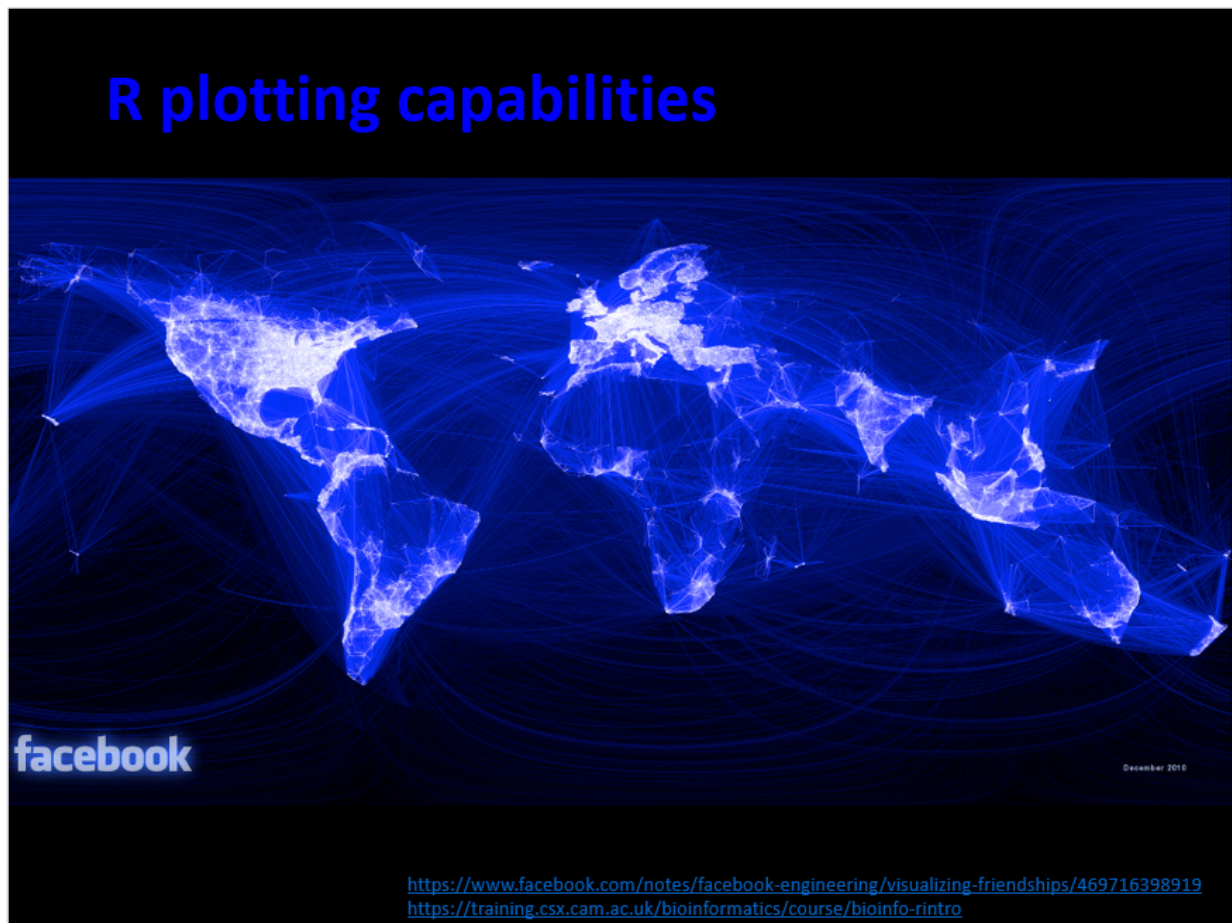
Rank	Language	Type	Score
1	Python	🌐 🖥️ ⚙️	100.0
2	Java	🌐 📱 🖥️	96.3
3	C	📱 🖥️ ⚙️	94.4
4	C++	📱 🖥️ ⚙️	87.5
5	R	🖥️	81.5
6	JavaScript	🌐	79.4
7	C#	🌐 📱 🖥️ ⚙️	74.5
8	Matlab	🖥️	70.6
9	Swift	📱 🖥️	69.1
10	Go	🌐 🖥️	68.0

Despite being a much more specialized language than the others, it's maintained its popularity in recent years due to the world being awash in an ever-growing pile of big data.

<https://spectrum.ieee.org/computing/software/the-top-programming-languages-2019>

R은 통계분석에 널리 사용되는데 이는 데이터 가시화를 위한 그래픽 기능이나 벡터 연산 등의 편리함 때문에 점점 더 많은 사람들이 사용하고 있습니다. 기존에는 느린 속도나 부족한 확장성이 다른 언어들에 비해 단점으로 지적되었으나 R 언어의 지속적인 개발과 업데이트로 이러한 단점들이 빠르게 극복되고 있습니다.

R 사용을 위해서는 R 언어의 코어 프로그램을 먼저 설치하고 그 다음 R 언어용 IDE인 RStudio 설치가 필요합니다.



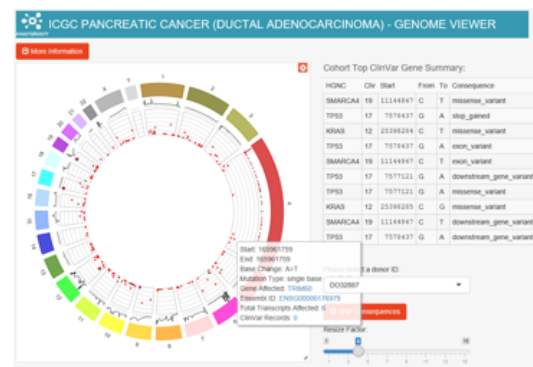
Interactive web applications

Shiny
from R Studio

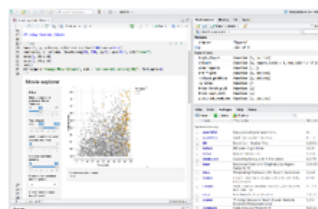
Get Started Gallery Articles Reference Deploy Help Contribute

Interact. Analyze. Communicate.

Take a fresh, interactive approach to telling your data story with Shiny. Let users interact with your data and your analysis. And do it all with R.



Shiny is an R package that makes it easy to build interactive web apps straight from R. You can host standalone apps on a webpage or embed them in R Markdown documents or build dashboards. You can also extend your Shiny apps with CSS themes, htmlwidgets, and JavaScript actions.



2.2 R / Rstudio installation

2.2.1 R 설치

- R 사이트에 접속 후 (<https://www.r-project.org/>) 좌측 메뉴 상단에 위치한 CRAN 클릭.
- 미리 사이트 목록에서 Korea의 아무 사이트나 들어감
- Download R for Windows를 클릭 후 base 링크 들어가서
- Download R x.x.x for Windows 링크 클릭으로 실행 프로그램 다운로드 - 로컬 컴퓨터에 Download 된 R-x.x.x-win.exe 를 실행
- 설치 가이드에 따라 R 언어 소프트웨어 설치 완료

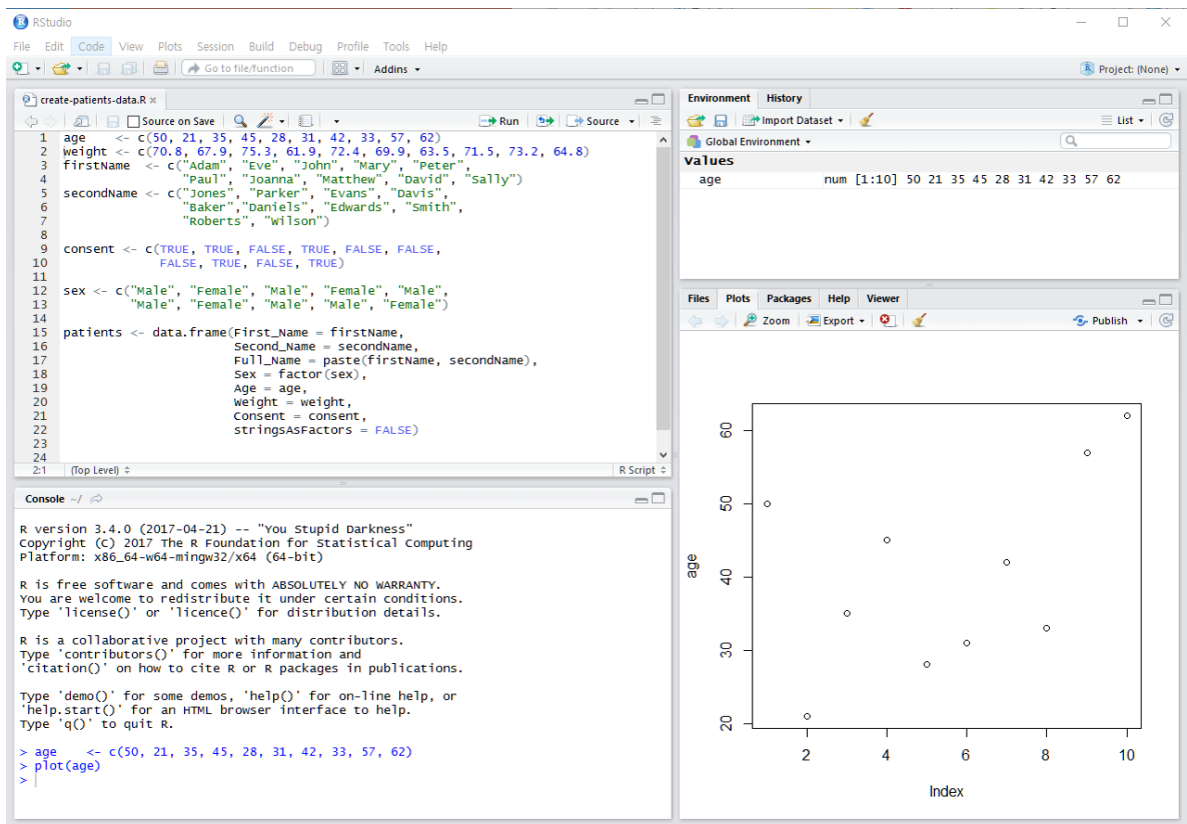
2.2.2 Rstudio 설치

Rstudio는 R 언어를 위한 오픈소스 기반 통합개발환경(IDE)으로 R 프로그래밍을 위한 편리한 기능들을 제공해 줍니다.

- 사이트에 접속 (<https://www.rstudio.com/>), 상단의 Products > RStudio 클릭
- RStudio Desktop 선택
- Download RStudio Desktop 클릭
- RStudio Desktop Free 버전의 Download를 선택하고
- Download RStudio for Windows 클릭, 다운로드
- 로컬 컴퓨터에 다운로드된 RStudio-x.x.x.exe 실행
- 설치 가이드에 따라 설치 완료

2.3 Rstudio interface

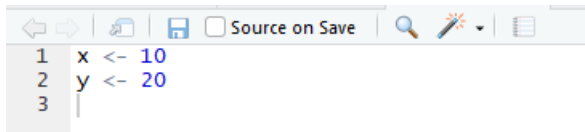
- 좌측 상단의 공간은 코드편집창, 좌측 하단은 콘솔창 이며 각 위치를 기호에 따라서 바꿀 수 있습니다.



2.3.1 Keyboard shortcuts

- 참고사이트
 - <https://support.rstudio.com/hc/en-us/articles/200711853-Keybaord-Shortcuts>
 - Tools -> Keyboard shortcut Quick Reference (Alt + Shift + K)

- 코드편집창 이동 (Ctrl+1) 콘솔창 이동(Ctrl+2)
- 한 줄 실행 (Ctrl+Enter)
- 주석처리 (Ctrl + Shift + C)
 - 또는 # 으로 시작하는 라인
- 실습
 - 코드편집창에서 다음 입력



- 단축키 Ctrl + enter 로 코드 실행
- 단축키 Ctrl + 2 로 커서 콘솔창으로 이동
- x 값 x+y 값 확인
- 단축키 Ctrl + 1 로 코드편집창 이동
- 단축키 Ctrl + Shift + C 사용

```
# x <- 10
# y <- 20
```

2.3.2 Set a working directory

- 시작 전 항상 작업 디렉토리 설정하는 것이 좋습니다. 예를 들어 c:\ 아래 새로운 디렉토리 kribb-r 을 만들고 작업공간으로 설정할 수 있습니다.

```
getwd()
dir()
setwd("C:\\kribb-r")
getwd()
dir()
```

- 또는 RStudio > Session > Set Working Directory > Choose Directory

2.3.3 Set a project

- 프로젝트를 만들어서 사용할 경우 파일이나 디렉토리, 내용 등을 쉽게 구분하여 사용 가능합니다. 아래와 같이 원하는 위치에 원하는 이름의 프로젝트를 생성하고 프로젝트를 시작할 때는 해당 디렉토리의 xxx.Rproj 파일을 클릭합니다.
- File > New Project > New Directory > New Project > "kribb-R" > Create Project
- File > New File > R Script > "day1.R"

2.4 R coding practice

2.4.1 Console calculator

$$2 + 2$$

$$((2 - 1)^2 + (1 - 3)^2)^{(1/2)}$$

$$2 + 2; 2 - 2$$

2.4.1.1 Exercise

다음 공식들을 계산하는 R 코드를 작성하시오

$$\sqrt{(4 + 3)(2 + 1)}$$

$$2^3 + 3^2$$

$$\frac{0.25 - 0.2}{\sqrt{0.2(1 - 0.2)/100}}$$

2.4.2 Variables and values

- 프로그래밍 언어의 공통적 개념 변수 , 함수 , 자료형 , 조건문 , 반복문
- Assignment operator (<- OR =)
 - Valid object name <- value
 - 단축키: Alt + - (the minus sign)

```
x <- 2
y <- x^2 - 2*x + 1
y
x <- "two"
some_data <- 9.8
```

- 내장 변수 Built-in variables

pi

- 변수이름 작명법
 - 문자, 숫자, “_”, “.” 등으로 구성
 - 대소문자 구분
 - 가독성, 의미있는 변수 이름
 - 길이 제한 없음

```
i_use_snake_case <- 1
otherPeopleUseCamelCase <- 2
some.people.use.periods <- 3
And_aFew.People_RENOUNCEconvention <- 4
```

- 자동 완성 기능 (Tab completion) in RStudio
- 이전 명령은 콘솔에서 위 아래 화살표
- 내장 함수 (Built-in functions)

```
x <- pi
sin(x)
sqrt(x)
log(x)
log(x, 10)
x <- c(10, 20, 30)
x + x
mean(x)
sum(x)/length(x)
```

2.4.2.1 Exercise

변수 `x` 에 1, 3, 5, 7, 9를, 변수 `y` 에 2, 4, 6, 8, 10을 저장하는 코드를 작성하고 `x` 와 `y` 를 더한 값을 `z` 에 저장하는 코드를 작성하시오

2.5 Supports

2.5.1 Help

```
help("mean")
?mean
example("mean")
help.search("mean")
??mean
help(package="MASS")
```

2.5.2 Cheatsheet

<https://rstudio.com/resources/cheatsheets/>

Base R Cheat Sheet

Getting Help

Accessing the help files

?mean

Get help of a particular function.

help.search('weighted mean')

Search the help files for a word or phrase.

help(package = 'dplyr')

Find help for a package.

More about an object

str(iris)

Get a summary of an object's structure.

class(iris)

Find the class an object belongs to.

Using Packages

install.packages('dplyr')

Download and install a package from CRAN.

library(dplyr)

Load the package into the session, making all its functions available to use.

dplyr::select

Use a particular function from a package.

data(iris)

Load a built-in dataset into the environment.

Working Directory

getwd()

Find the current working directory (where inputs are found and outputs are sent).

setwd('C:/file/path')

Change the current working directory.

Use projects in RStudio to set the working directory to the folder you are working in.

Vectors

Creating Vectors

<code>c(2, 4, 6)</code>	<code>2 4 6</code>	Join elements into a vector
<code>2:6</code>	<code>2 3 4 5 6</code>	An integer sequence
<code>seq(2, 3, by=0.5)</code>	<code>2.0 2.5 3.0</code>	A complex sequence
<code>rep(1:2, times=3)</code>	<code>1 2 1 2 1 2</code>	Repeat a vector
<code>rep(1:2, each=3)</code>	<code>1 1 1 2 2 2</code>	Repeat elements of a vector

Vector Functions

sort(x) Return x sorted.	rev(x) Return x reversed.
table(x) See counts of values.	unique(x) See unique values.

Selecting Vector Elements

By Position

<code>x[4]</code>	The fourth element.
<code>x[-4]</code>	All but the fourth.
<code>x[2:4]</code>	Elements two to four.
<code>x[-(2:4)]</code>	All elements except two to four.
<code>x[c(1, 5)]</code>	Elements one and five.

By Value

<code>x[x == 10]</code>	Elements which are equal to 10.
<code>x[x < 0]</code>	All elements less than zero.
<code>x[x %in% c(1, 2, 5)]</code>	Elements in the set 1, 2, 5.

Named Vectors

<code>x['apple']</code>	Element with name 'apple'.
-------------------------	----------------------------

Programming

For Loop

```
for (variable in sequence){  
  Do something  
}
```

Example

```
for (i in 1:4){  
  j <- i + 10  
  print(j)  
}
```

While Loop

```
while (condition){  
  Do something  
}
```

Example

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

If Statements

```
if (condition){  
  Do something  
} else {  
  Do something different  
}
```

Example

```
if (i > 3){  
  print('Yes')  
} else {  
  print('No')  
}
```

Functions

```
function_name <- function(var){  
  Do something  
  return(new_variable)  
}
```

Example

```
square <- function(x){  
  squared <- x*x  
  return(squared)  
}
```

Reading and Writing Data

Also see the **readr** package.

Input	Output	Description
<code>df <- read.table('file.txt')</code>	<code>write.table(df, 'file.txt')</code>	Read and write a delimited text file.
<code>df <- read.csv('file.csv')</code>	<code>write.csv(df, 'file.csv')</code>	Read and write a comma separated value file. This is a special case of read.table/write.table.
<code>load('file.Rdata')</code>	<code>save(df, file = 'file.Rdata')</code>	Read and write an R data file, a file type special for R.

Conditions	a == b	Are equal	a > b	Greater than	a >= b	Greater than or equal to	is.na(a)	is missing
	a != b	Not equal	a < b	Less than	a <= b	Less than or equal to	is.null(a)	is null

RStudio® is a trademark of RStudio, Inc. • CC BY-Mharr McNeill • mharr@mcneill@gmail.com

Learn more at [web page](#) or [vignette](#) • package version • Updated: 3/15

Types

Converting between common data types in R. Can always go from a higher value in the table to a lower value.

as.logical	TRUE, FALSE, TRUE	Boolean values (TRUE or FALSE)
as.numeric	1, 0, 1	Integers or floating point numbers.
as.character	'1', '0', '1'	Character strings. Generally preferred to factors.
as.factor	'1', '0', '1', levels: '1', '0'	Character strings with preset levels. Needed for some statistical models.

Maths Functions

log(x)	Natural log.	sum(x)	Sum.
exp(x)	Exponential.	mean(x)	Mean.
max(x)	Largest element.	median(x)	Median.
min(x)	Smallest element.	quantile(x)	Percentage quantiles.
round(x, n)	Round to n decimal places.	rank(x)	Rank of elements.
signif(x, n)	Round to n significant figures.	var(x)	The variance.
cor(x, y)	Correlation.	sd(x)	The standard deviation.

Variable Assignment

```
> a <- 'apple'  
> a  
[1] 'apple'
```

The Environment

ls()	List all variables in the environment.
rm(x)	Remove x from the environment.
rm(list = ls())	Remove all variables from the environment.

You can use the environment panel in RStudio to browse variables in your environment.

Matrices

`m <- matrix(x, nrow = 3, ncol = 3)`
Create a matrix from x.

<code>m[2,]</code>	Select a row	<code>t(m)</code> Transpose
<code>m[, 1]</code>	Select a column	<code>m %*% n</code> Matrix Multiplication
<code>m[2, 3]</code>	Select an element	<code>solve(m, n)</code> Find x in: m * x = n

Lists

`l <- list(x = 1:5, y = c('a', 'b'))`
A list is a collection of elements which can be of different types.

<code>l[[2]]</code>	<code>l[1]</code>	<code>l\$x</code>	<code>l['y']</code>
Second element of l	New list with only the first element.	Element named x	New list with only element named y.

Also see the **dplyr** package.

Data Frames

`df <- data.frame(x = 1:3, y = c('a', 'b', 'c'))`
A special case of a list where all elements are the same length.

x	y
1	a
2	b
3	c

Matrix subsetting

<code>df[, 2]</code>	
<code>df[2,]</code>	
<code>df[2, 2]</code>	

List subsetting

<code>df\$x</code>	<code>df[[2]]</code>

Understanding a data frame

<code>View(df)</code>	See the full data frame.
<code>head(df)</code>	See the first 6 rows.

nrow(df)

Number of rows.

ncol(df)

Number of columns.

dim(df)

Number of columns and rows.

cbind

- Bind columns.

rbind

- Bind rows.

Strings

Also see the **stringr** package.

<code>paste(x, y, sep = '')</code>	Join multiple vectors together.
<code>paste(x, collapse = '')</code>	Join elements of a vector together.
<code>grep(pattern, x)</code>	Find regular expression matches in x.
<code>gsub(pattern, replace, x)</code>	Replace matches in x with a string.
<code>toupper(x)</code>	Convert to uppercase.
<code>tolower(x)</code>	Convert to lowercase.
<code>nchar(x)</code>	Number of characters in a string.

Factors

factor(x)	Turn a vector into a factor. Can set the levels of the factor and the order.
cut(x, breaks = 4)	Turn a numeric vector into a factor by 'cutting' into sections.

Statistics

lm(y ~ x, data=df) Linear model.	t.test(x, y) Perform a t-test for difference between means.	prop.test Test for a difference between proportions.
glm(y ~ x, data=df) Generalised linear model.	pairwise.t.test Perform a t-test for paired data.	aov Analysis of variance.
summary Get more detailed information out a model.		

Distributions

Random Variates	Density Function	Cumulative Distribution	Quantile
Normal	<code>rnorm</code>	<code>dnorm</code>	<code>pnorm</code>
Poisson	<code>rpois</code>	<code>dpois</code>	<code>qpois</code>
Binomial	<code>rbinom</code>	<code>dbinom</code>	<code>qbinom</code>
Uniform	<code>runif</code>	<code>dunif</code>	<code>qunif</code>

Plotting

Also see the **ggplot2** package.

plot(x) Values of x in order.	plot(x, y) Values of x against y.	hist(x) Histogram of x.
---	---	-----------------------------------

Dates

See the **lubridate** package.

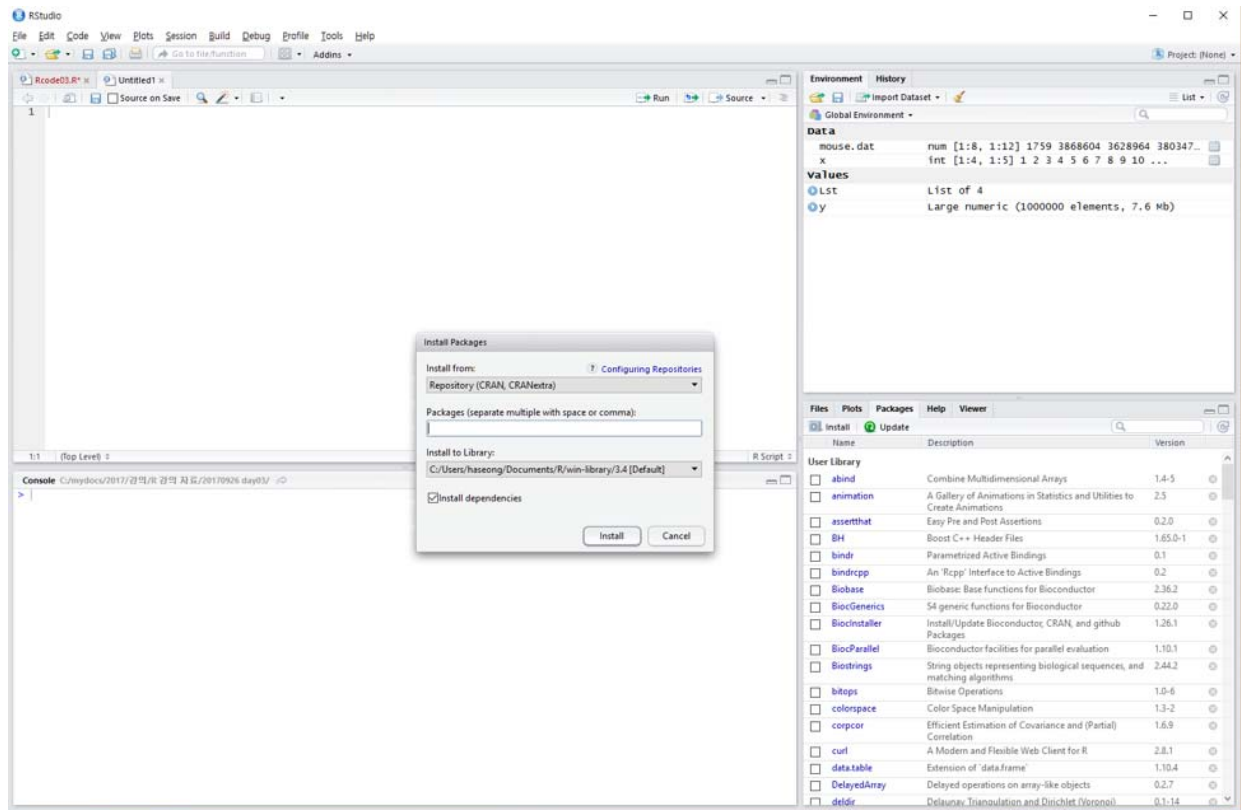
RStudio® is a trademark of RStudio, Inc. • CC BY-Mharr McNeill • mharr@mcneill@gmail.com • 844-448-1212 • [rstudio.com](#)

Learn more at [web page](#) or [vignette](#) • package version • Updated: 3/15

2.6 R packages and Dataset

2.6.1 R packages

- R 패키지는 함수들의 모음으로 다른 사람들이 만들어 놓은 함수를 가져와서 사용할 수 있음
- 예) `sum()` 은 `base` package에 있고 `sd()` 함수는 `stats` package에서 제공
- 패키지를 구할 수 있는 가장 대표적인 사이트
- The Comprehensive R Archive Network (CRAN) - <http://cran.r-project.org/web/views/>
- Bioconductor - <http://www.bioconductor.org/packages/release/bioc/>



- UsingR package installation


```
install.packages("UsingR")
```

- UsingR package loading

```
library(UsingR)
```

```
help(package="UsingR")
```

2.6.2 Data sets

- 일반적으로 패키지 안에 관련된 데이터도 같이 저장
- data() function를 이용해서 패키지 데이터를 사용자 작업공간에 복사해서 사용 가능

```
head(rivers)
```

```
length(rivers)
```

```
class(rivers)
```

```
data(rivers)
```

```
data(package="UsingR")
```

```
library(HistData)
```

```
head(Cavendish)
```

```
str(Cavendish)
```

```
head(Cavendish$density2)
```

이 저작물은 크리에이티브 커먼즈 저작자표시-비영리-변경금지 4.0 국제 라이선스에 따라 이용할 수 있습니다.