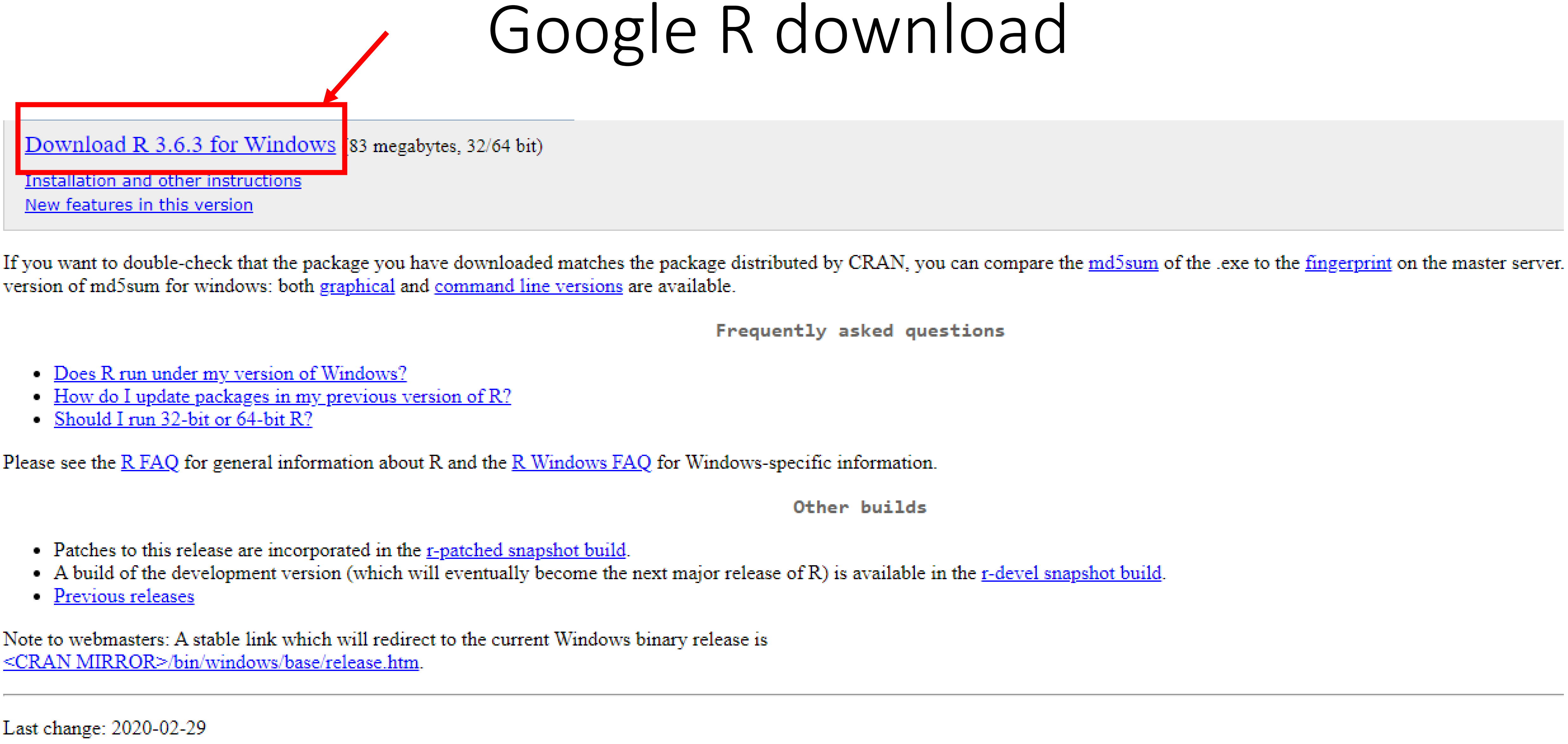
2020
生物統計應用方法

R基礎介紹

Google R download



Download R 3.6.3 for Windows (83 megabytes, 32/64 bit)
Installation and other instructions
New features in this version

If you want to double-check that the package you have downloaded matches the package distributed by CRAN, you can compare the md5sum of the .exe to the fingerprint on the master server. version of md5sum for windows: both graphical and command line versions are available.

Frequently asked questions

- Does R run under my version of Windows?
- How do I update packages in my previous version of R?
- Should I run 32-bit or 64-bit R?

Please see the R FAQ for general information about R and the R Windows FAQ for Windows-specific information.

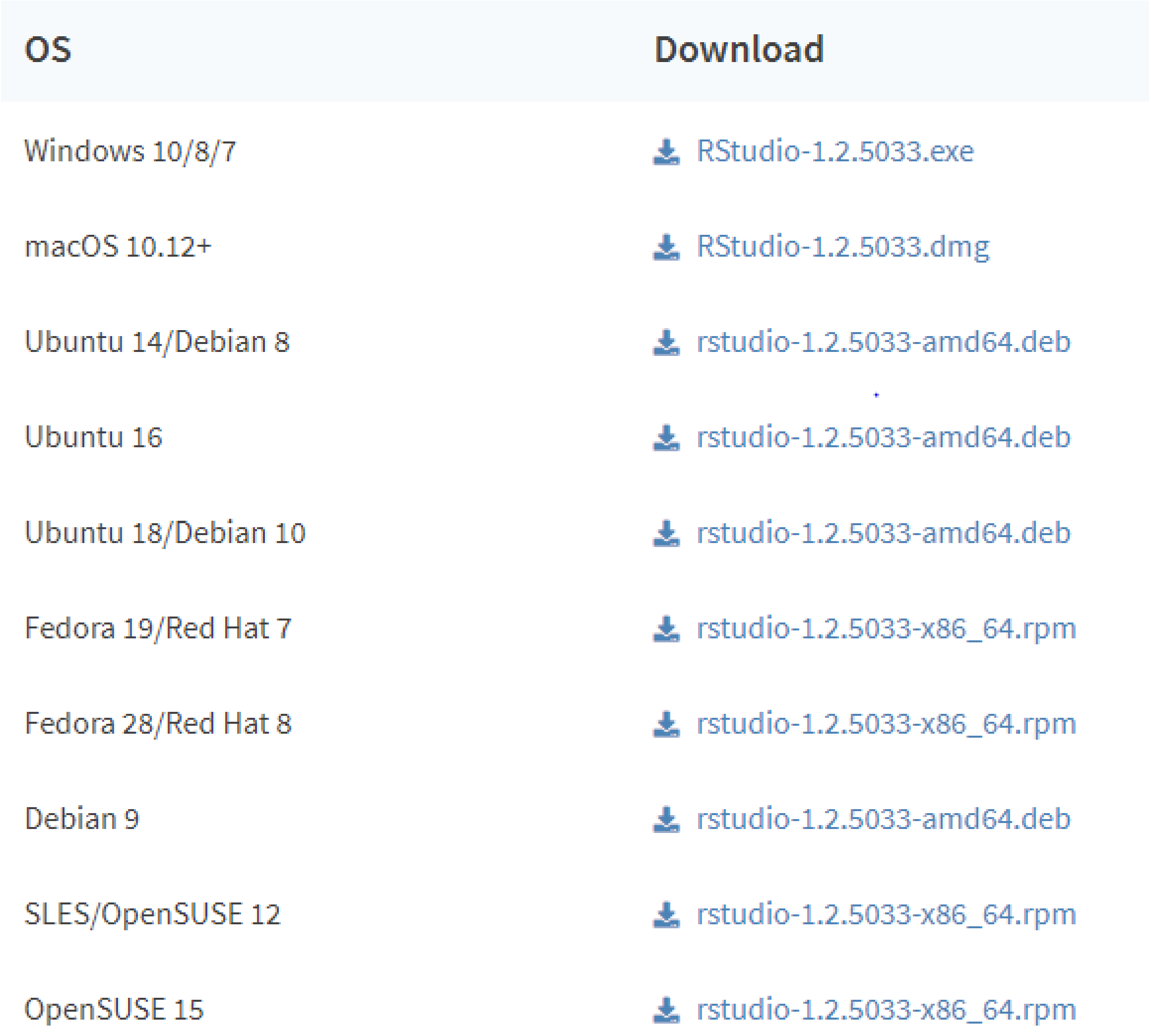
Other builds

- Patches to this release are incorporated in the r-patched snapshot build.
- A build of the development version (which will eventually become the next major release of R) is available in the r-devel snapshot build.
- Previous releases

Note to webmasters: A stable link which will redirect to the current Windows binary release is
<CRAN MIRROR>/bin/windows/base/release.htm.

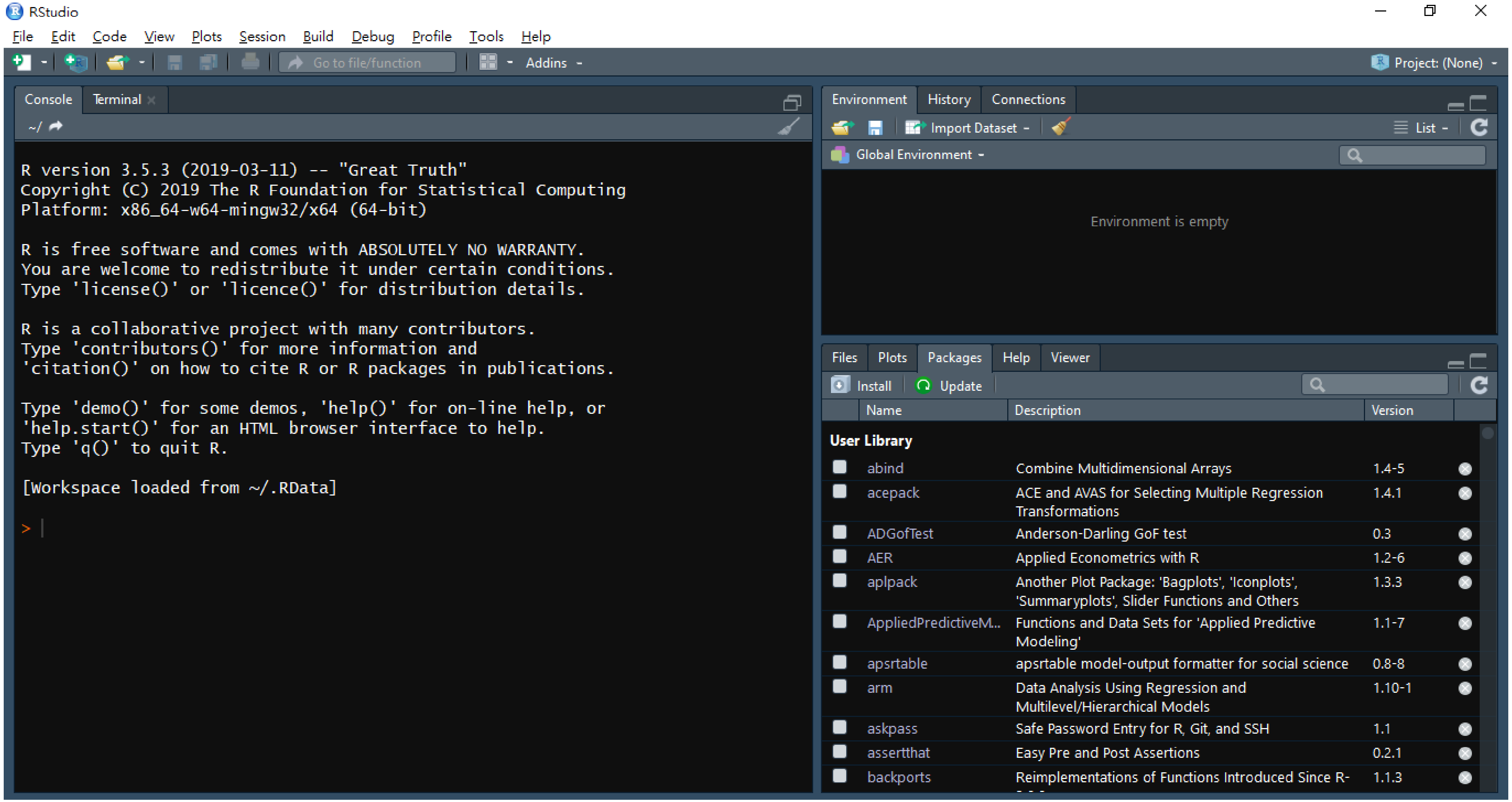
Last change: 2020-02-29

Google R Studio download



OS	Download
Windows 10/8/7	RStudio-1.2.5033.exe
macOS 10.12+	RStudio-1.2.5033.dmg
Ubuntu 14/Debian 8	rstudio-1.2.5033-amd64.deb
Ubuntu 16	rstudio-1.2.5033-amd64.deb
Ubuntu 18/Debian 10	rstudio-1.2.5033-amd64.deb
Fedora 19/Red Hat 7	rstudio-1.2.5033-x86_64.rpm
Fedora 28/Red Hat 8	rstudio-1.2.5033-x86_64.rpm
Debian 9	rstudio-1.2.5033-amd64.deb
SLES/OpenSUSE 12	rstudio-1.2.5033-x86_64.rpm
OpenSUSE 15	rstudio-1.2.5033-x86_64.rpm

R Studio window

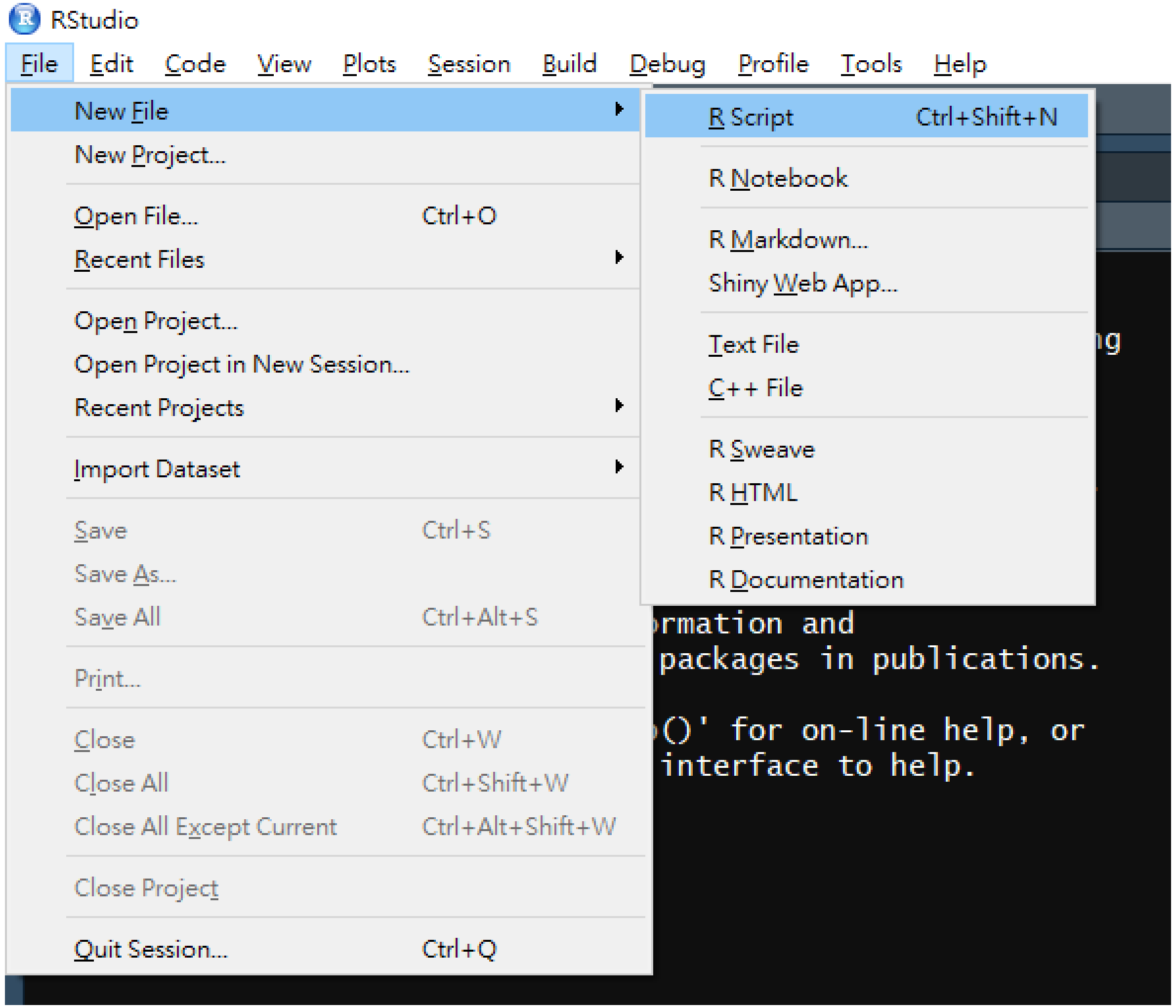


```
R version 3.5.3 (2019-03-11) -- "Great Truth"
Copyright (C) 2019 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (64-bit)

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

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

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

[Workspace loaded from ~/.RData]

> 
```

User Library

	Name	Description	Version	
□	abind	Combine Multidimensional Arrays	1.4-5	⊗
□	acepack	ACE and AVAS for Selecting Multiple Regression Transformations	1.4.1	⊗
□	ADGofTest	Anderson-Darling GoF test	0.3	⊗
□	AER	Applied Econometrics with R	1.2-6	⊗
□	aplpack	Another Plot Package: 'Bagplots', 'Iconplots', 'Summaryplots', Slider Functions and Others	1.3.3	⊗
□	AppliedPredictiveM...	Functions and Data Sets for 'Applied Predictive Modeling'	1.1-7	⊗
□	apsrtable	apsrtable model-output formatter for social science	0.8-8	⊗
□	arm	Data Analysis Using Regression and Multilevel/Hierarchical Models	1.10-1	⊗
□	askpass	Safe Password Entry for R, Git, and SSH	1.1	⊗
□	assertthat	Easy Pre and Post Assertions	0.2.1	⊗
□	backports	Reimplementations of Functions Introduced Since R-	1.1.3	⊗

Record your Code on Script



```
ng
rmation and
packages in publications.

()' for on-line help, or
interface to help.
```

Assignment and Basic Arithmetic operators

Assignment

- x=5
- x < - 5(alt -)
- y=3

Basic operators

- x+y,x-y ,x*y,x/y(加減乘除)
- x^y,x**y(次方)
- x%%y(餘數)
- x>y,x==y,x!=y,x<=y(條件判斷)

Common functions

- sqrt(),exp(),log(),sin(),…
- mean(),median(),sd(),quantile(),cor(),list(),…
- sum(),prod(),cumsum(),sort(),…
- lm(),glm(),anova(),summary(),…

Vector

- c():The most fundamental object in R

```
c(1,2,3) :numeric
c(‘Tony’,’Mary’) :character
```

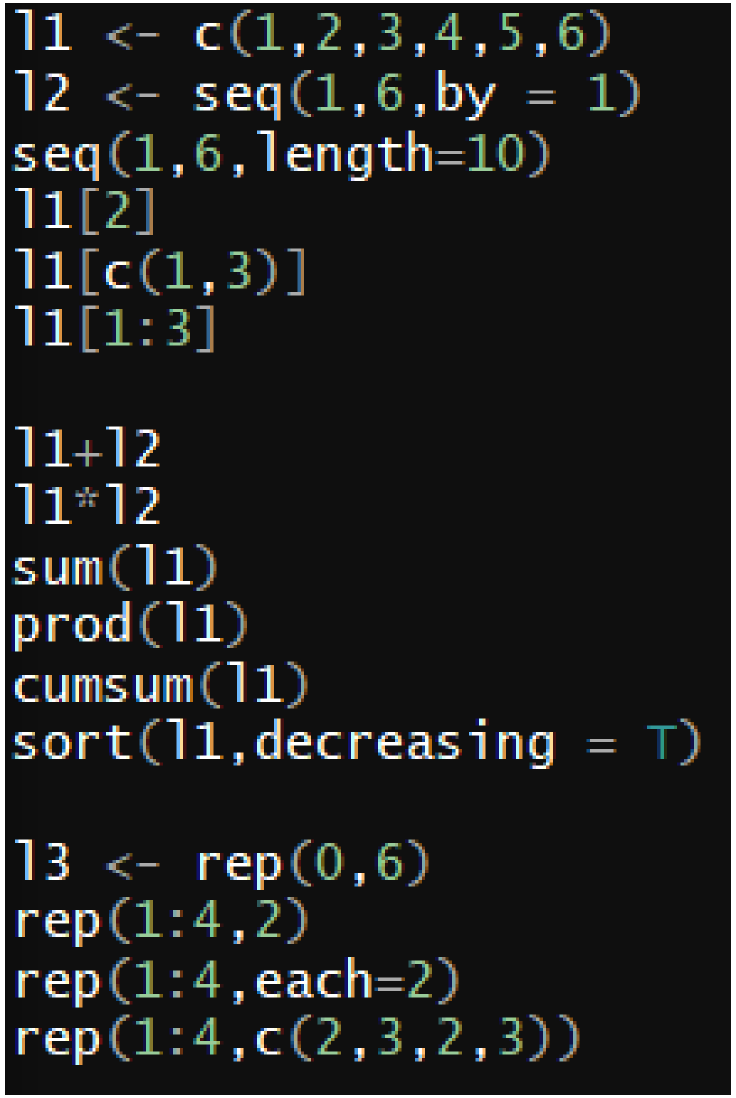
- seq(),rep() are common functions to construct vectors

```
rep(0,5),seq(0,10,by=1)
```

- Vector Manipulation

```
c(56, 78, 77) + c(77, 88, 99)
c(56, 78, 77) * c(77, 88, 99)
mean(c(56, 78, 77))
```

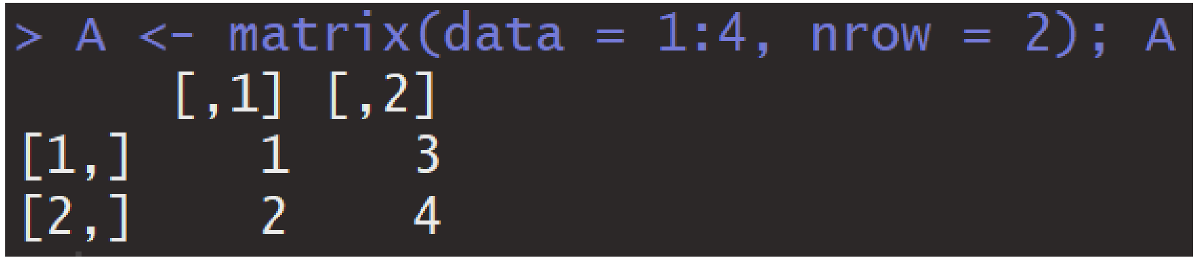
Example for vector



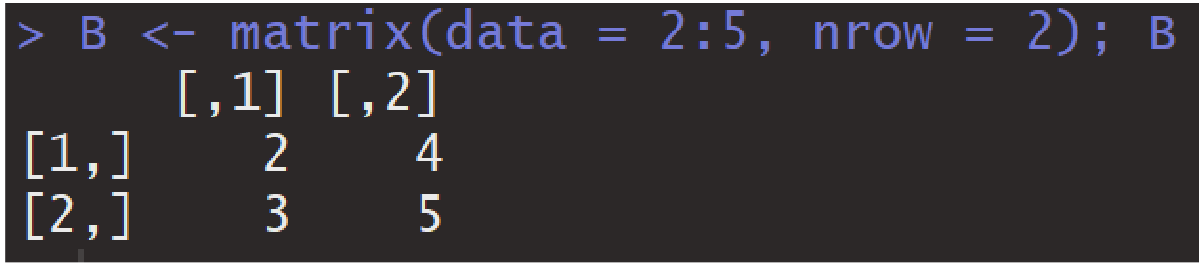
```
l1 <- c(1,2,3,4,5,6)
l2 <- seq(1,6,by = 1)
seq(1,6,length=10)
l1[2]
l1[c(1,3)]
l1[1:3]

l1+l2
l1*l2
sum(l1)
prod(l1)
cumsum(l1)
sort(l1,decreasing = T)

l3 <- rep(0,6)
rep(1:4,2)
rep(1:4,each=2)
rep(1:4,c(2,3,2,3))
```

Matrix

- matrix()
- R supports the arithmetic operations on matrix.
- * v.s. %*%



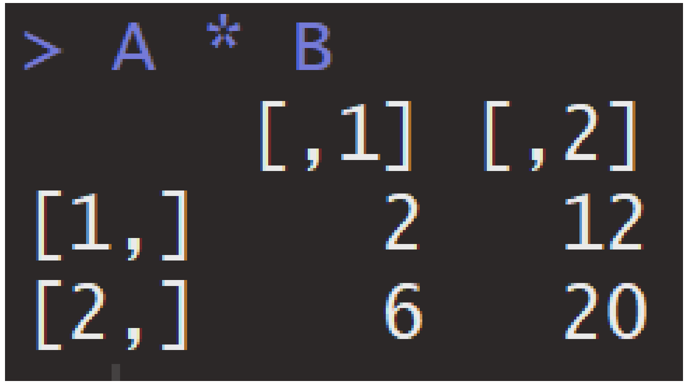
```
> A <- matrix(data = 1:4, nrow = 2); A
     [,1] [,2]
[1,]    1    3
[2,]    2    4
```



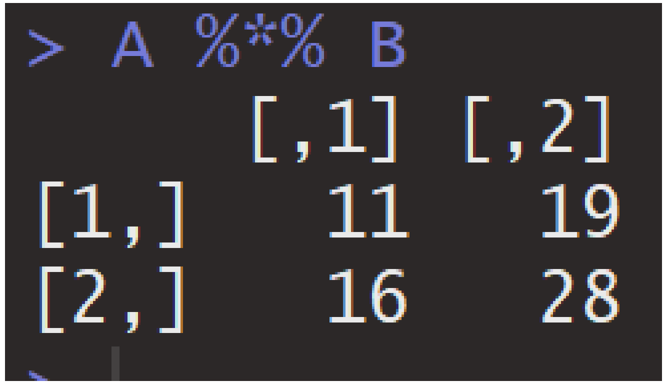
```
> B <- matrix(data = 2:5, nrow = 2); B
     [,1] [,2]
[1,]    2    4
[2,]    3    5
```

Matrix

- matrix()
- R supports the arithmetic operations on matrix.
- * v.s. %*%
- A[1,1],A[1,],cbind(),rbind()



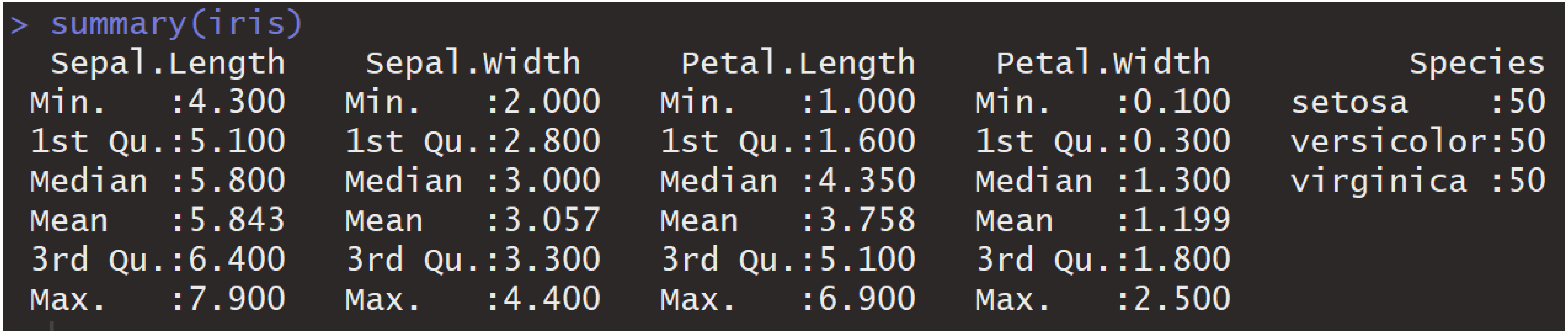
```
> A * B
     [,1] [,2]
[1,]    2   12
[2,]    6   20
```



```
> A %*% B
     [,1] [,2]
[1,]   11   19
[2,]   16   28
```

data.frame

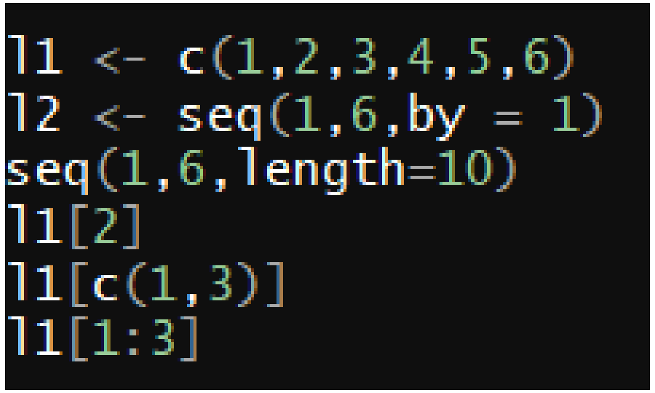
- data.frame()
- Most of the data would be stored as this form



```
> summary(iris)
  Sepal.Length    Sepal.Width     Petal.Length    Petal.Width          Species
 Min.   :4.300   Min.   :2.000   Min.   :1.000   Min.   :0.100   setosa    :50
 1st Qu.:5.100   1st Qu.:2.800   1st Qu.:1.600   1st Qu.:0.300   versicolor:50
 Median :5.800   Median :3.000   Median :4.350   Median :1.300   virginica :50
 Mean   :5.843   Mean   :3.057   Mean   :3.758   Mean   :1.199
 3rd Qu.:6.400   3rd Qu.:3.300   3rd Qu.:5.100   3rd Qu.:1.800
 Max.   :7.900   Max.   :4.400   Max.   :6.900   Max.   :2.500
```

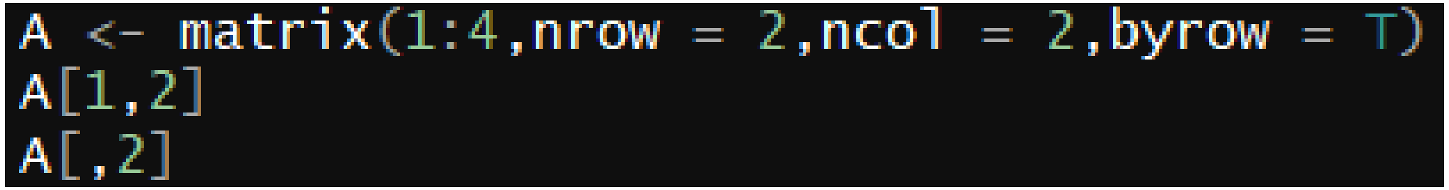
Index

- Vector



```
l1 <- c(1,2,3,4,5,6)
l2 <- seq(1,6,by = 1)
seq(1,6,length=10)
l1[2]
l1[c(1,3)]
l1[1:3]
```

- Matrix



```
A <- matrix(1:4,nrow = 2,ncol = 2,byrow = T)
A[1,2]
A[,2]
```

Index

- data.frame

```
iris$Sepal.Length, iris[, 1]
iris[, c(1, 3)], iris[, c(“Sepal.Length”, “Petal.Length”)]
```

Loop:for

- for() {…}：Do the same thing for many times
- for(i in 1:nrow(iris)) {

```
iris[i, 1] <-iris[i, 1] + 1
}
```

Loop:while

- while() {…}：Do the same thing until the condition fails
- j=5
- while(j>0) {

  ```
      print(j)
      j=j-1
      }
  ```

Flow Control：if

```
for(i in 1:nrow(iris)) {
        if(iris[i, 1] > 5) {
                iris[i, 1] <-5
                }
        }
```

```
iris$Sepal.Length<-ifelse(iris$Sepal.Length> 5,5,iris$Sepal.Length)
```

Define a function

- func_name <-function(…) {

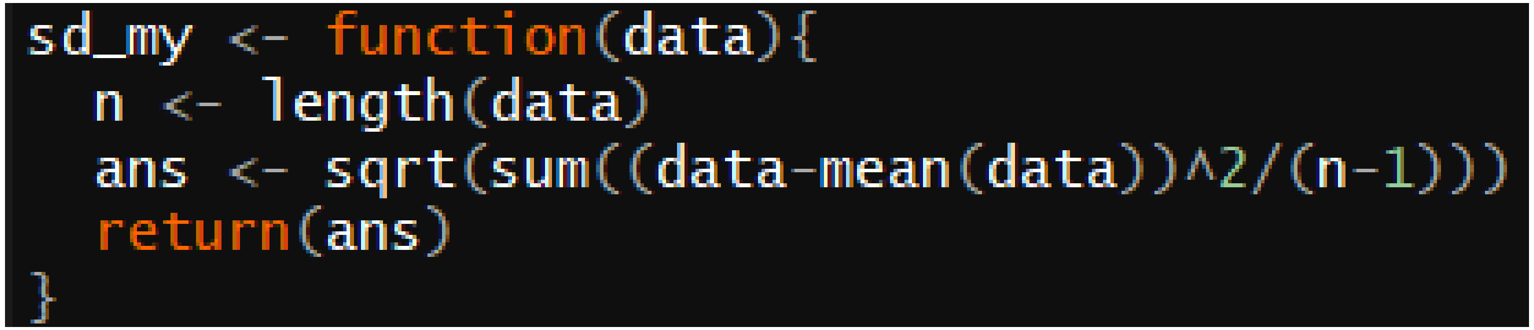
  …

  …

  …

  return(…)

  }



```
sd_my <- function(data){
  n <- length(data)
  ans <- sqrt(sum((data-mean(data))^2/(n-1)))
  return(ans)
}
```

Install Packages into R

- Install the package **ggplot2**

  install.packages("ggplot2")

- Load the package **ggplot2**

  library(ggplot2)

Read and write the Data

- Check your work directory

  getwd()
- Check the type and place of your data
- setwd(…)
- read.csv(…)
- read.table(…)
- Using write.csv(…) to rewrite the data

Example: Linear Regression

There are 10 students in the class. Their height(cm) and weight(kg) are shown below.

Height	159	165	156	148	157	174	161	166	165	170
weight	50	53	48	45	50	68	62	43	52	55

1.Describe the pattern of the data.

2.Are there any possible outlier?

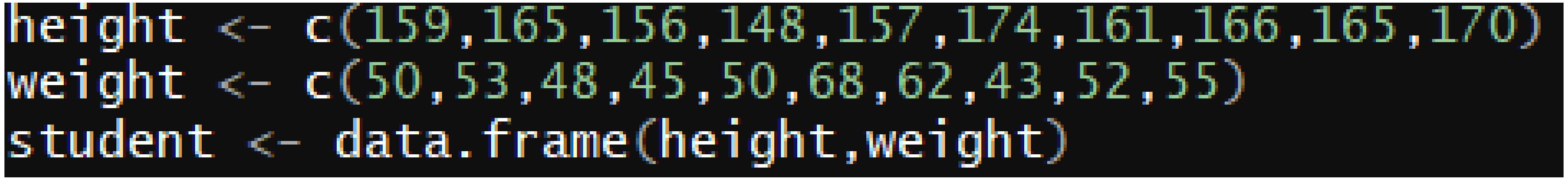
3.Compute the Pearson’s correlation.

4.Fit a least-squares line

5.Compute the measure of R^2

Example: Linear Regression

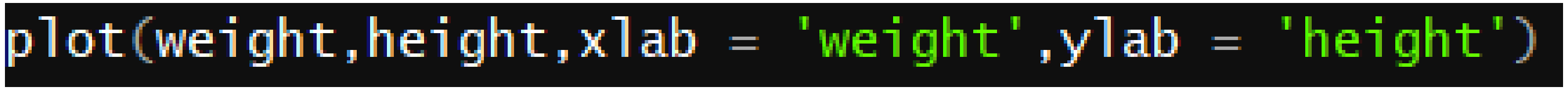
Keyin the data and form a data.frame called student.



```
height <- c(159,165,156,148,157,174,161,166,165,170)
weight <- c(50,53,48,45,50,68,62,43,52,55)
student <- data.frame(height,weight)
```

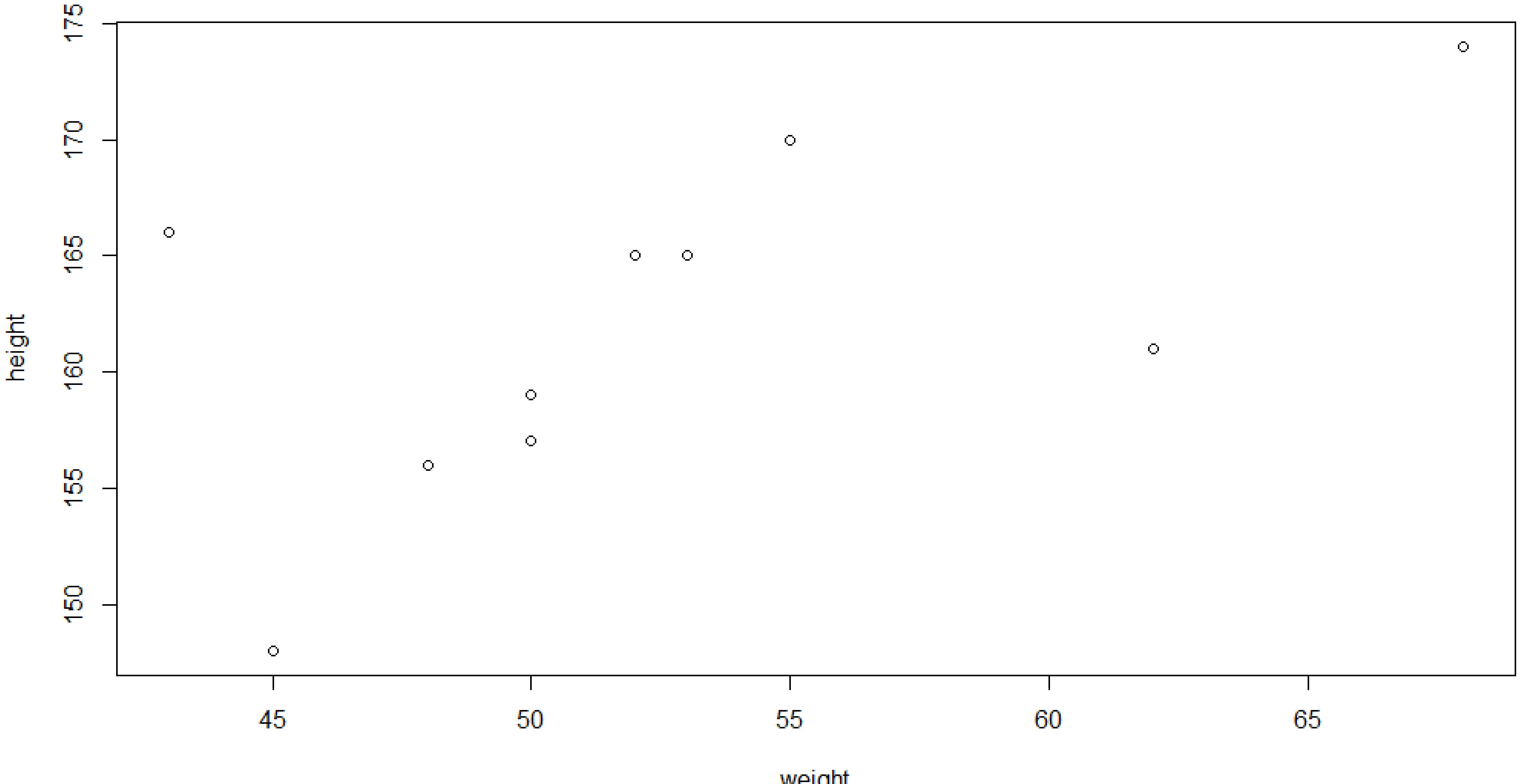
Example: Linear Regression

- 1.Describe the pattern of the data
- Idea：Draw the scatterplot and do some observations



```
plot(weight,height,xlab = 'weight',ylab = 'height')
```

Example: Linear Regression



Example: Linear Regression

- 2.Any possible outlier?

What is the meaning of outlier? How to define it?

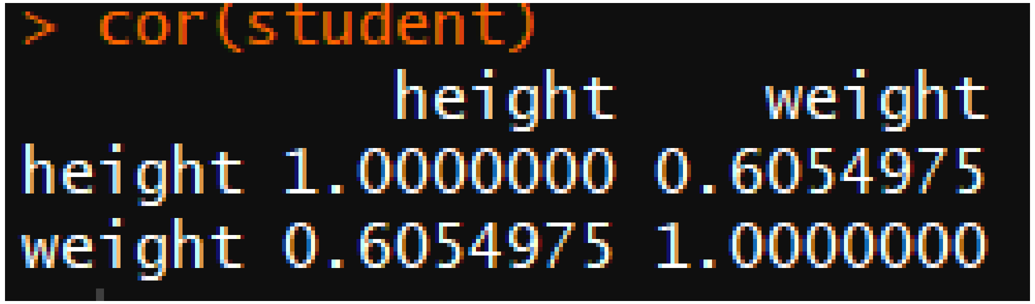
Actually, there are multiple criterions of an outlier.

The answer may depend on your choice and objective.

Example: Linear Regression

- 3.Compute Pearson’s correlation

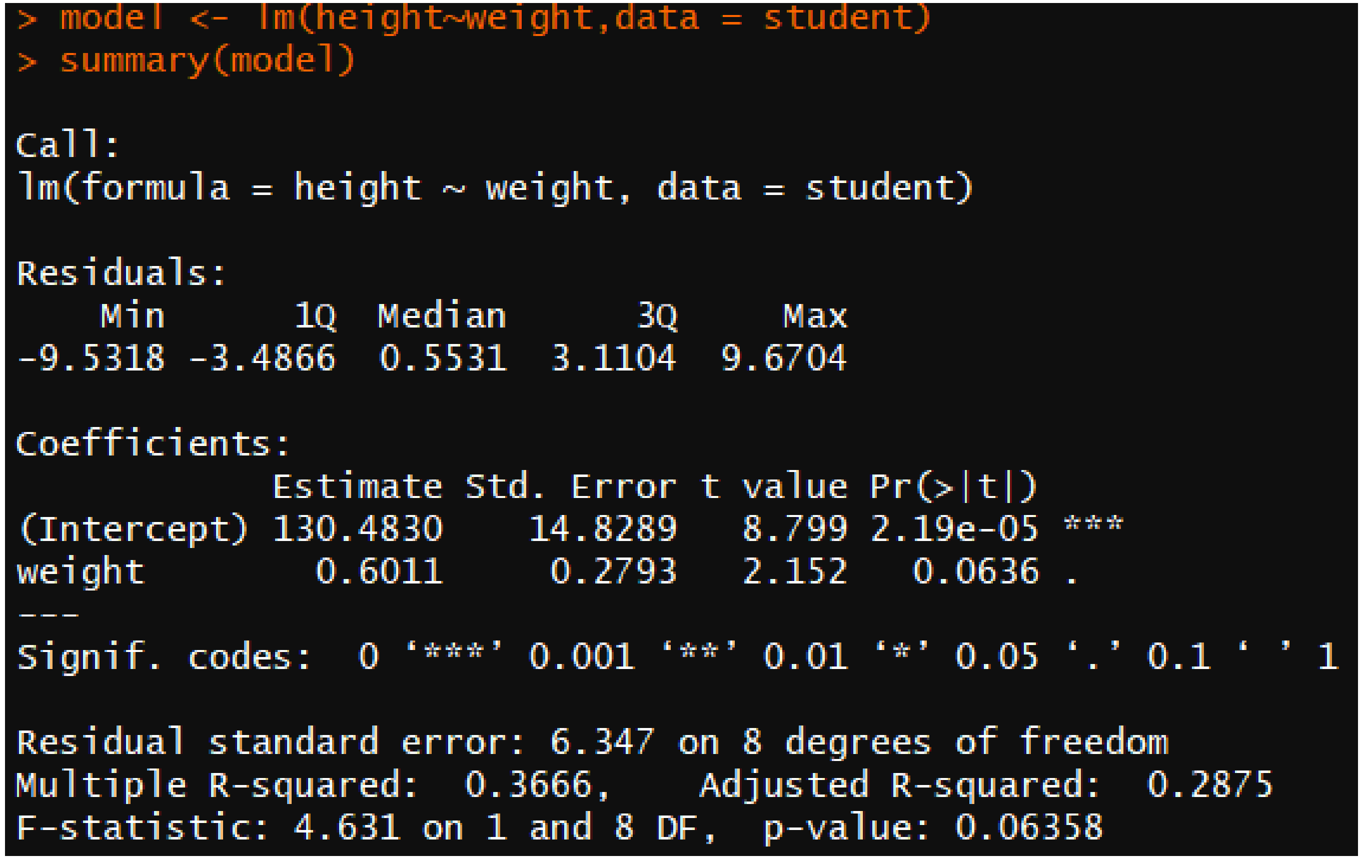
>cor(student)



```
> cor(student)
          height    weight
height 1.0000000 0.6054975
weight 0.6054975 1.0000000
```

Example: Linear Regression

- 4.Fit a least-squares line

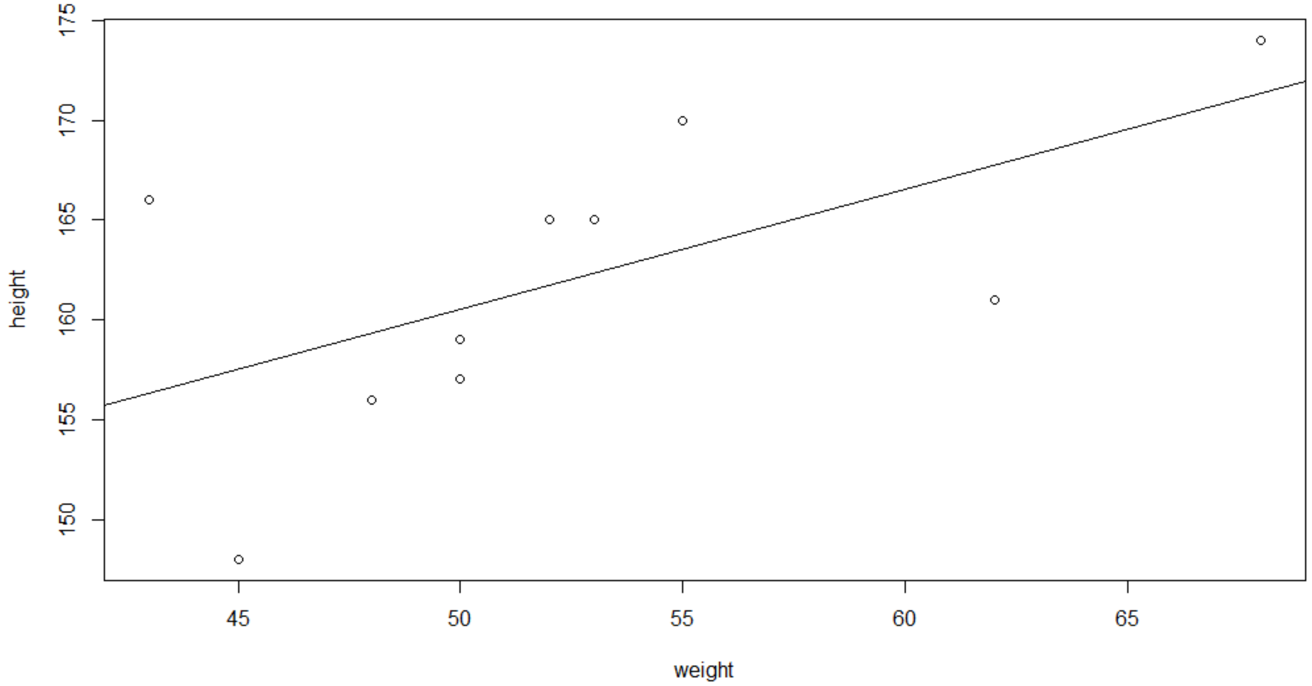


```
> model <- lm(height~weight,data = student)
> summary(model)

Call:
lm(formula = height ~ weight, data = student)

Residuals:
    Min      1Q  Median      3Q     Max 
-9.5318 -3.4866  0.5531  3.1104  9.6704 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept) 130.4830    14.8289   8.799 2.19e-05 ***
weight        0.6011     0.2793   2.152   0.0636 .  
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 6.347 on 8 degrees of freedom
Multiple R-squared:  0.3666,	Adjusted R-squared:  0.2875 
F-statistic: 4.631 on 1 and 8 DF,  p-value: 0.06358
```

Example: Linear Regression

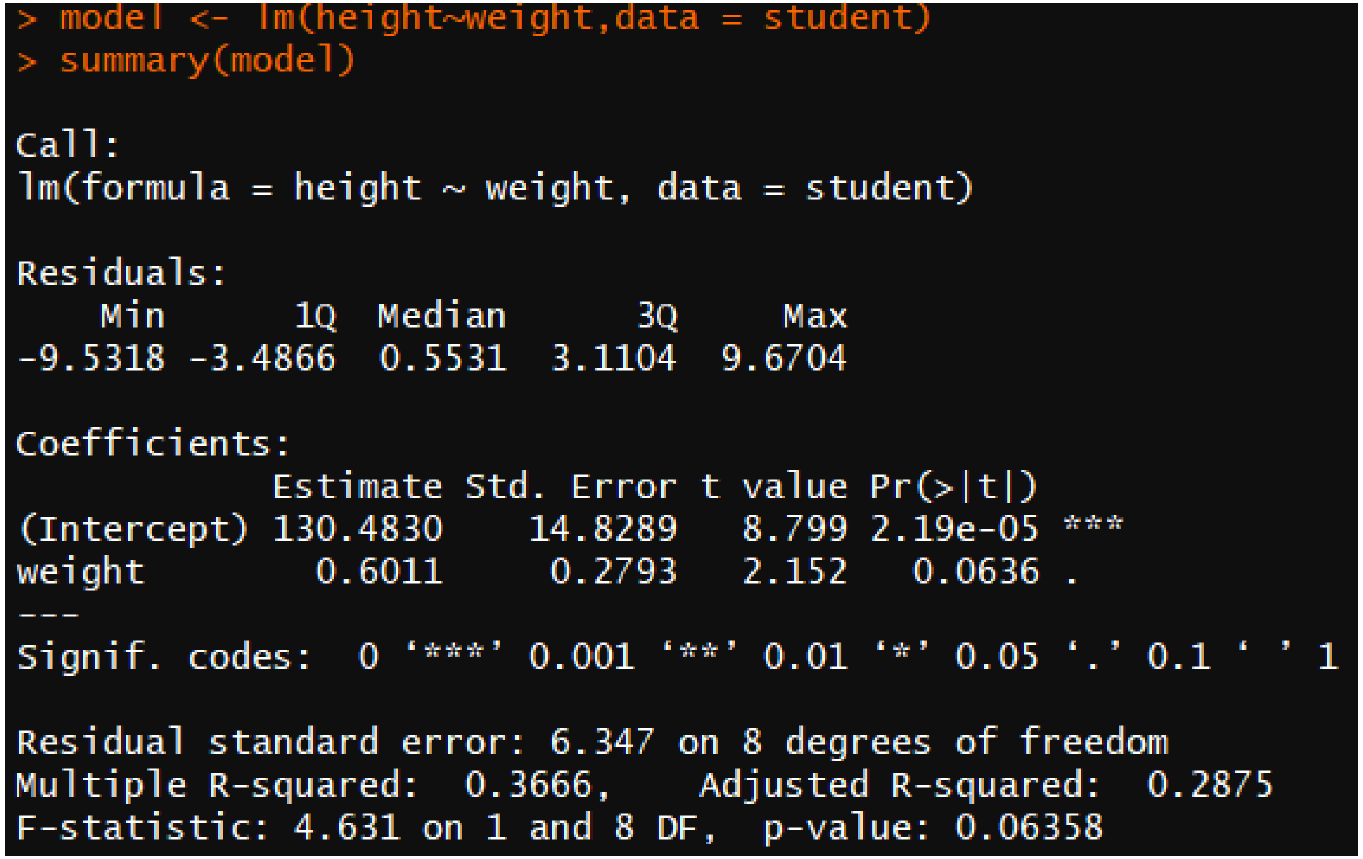
- 4.Fit a least-squares line

```
>abline(model)
```



Example: Linear Regression

- Compute the measure of R^2



```
> model <- lm(height~weight,data = student)
> summary(model)

Call:
lm(formula = height ~ weight, data = student)

Residuals:
    Min      1Q  Median      3Q     Max 
-9.5318 -3.4866  0.5531  3.1104  9.6704 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept) 130.4830    14.8289   8.799 2.19e-05 ***
weight        0.6011     0.2793   2.152   0.0636 .  
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 6.347 on 8 degrees of freedom
Multiple R-squared:  0.3666,	Adjusted R-squared:  0.2875 
F-statistic: 4.631 on 1 and 8 DF,  p-value: 0.06358
```

Furthermore

Some useful packages：

- Dplyr:data manipulation

https://rpubs.com/justmarkham/dplyr-tutorial

- ggplot2 :data visualization

http://r-statistics.co/Complete-Ggplot2-Tutorial-Part1-With-R-Code.html

- Rmarkdown:write your report in R

https://rmarkdown.rstudio.com/

An interactive way to learn R：

- swirl https://swirlstats.com/students.html