Data analysis and visualization using R

Zhengtao Xiao
8.28.2020

zhengtao.xiao@duke.edu

sas

SPSS

Prism

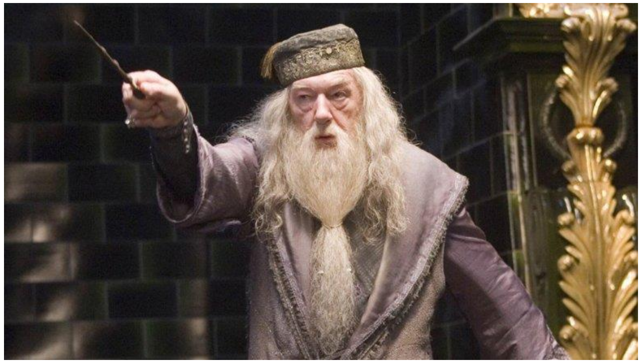
=

Muggles

Users have to reply on the functions that have been developed for them. The way to approach a problem constrained by how their developers thought to approach them. Users have to pay money to their constrained functions.

R

=



Wizard

Lots of functions or packages (spells) that developed by statistical researchers, but users can also create their own. Users don’t have to pay. Once experienced enough, users are almost unlimited in their ability to change their environment.

Learning R ……

History of R

- R initially written & released as an open source software by Ross Ihaka and Robert Gentleman during 90s.
- Since 1997: R Development Core Team ~ 37 people & thousands of code writers and statisticians share their libraries!

Advantages of R

- An interpreted computer language
- R is very powerful for data manipulation, statistics, and graphics.
- Fast and free
- Active user community
- Force you to think about your analysis

Disadvantages of R

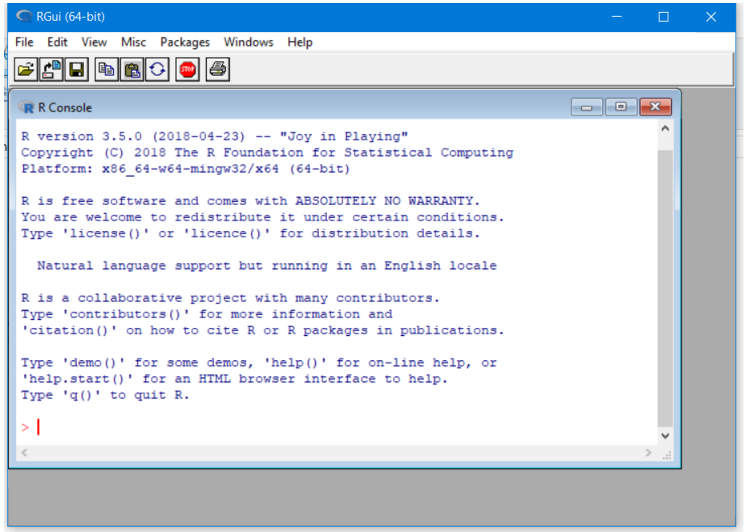
- Not user friendly, very simple GUI.
- No commercial support
- Easy to make mistakes.

Download R

https://www.r-project.org/

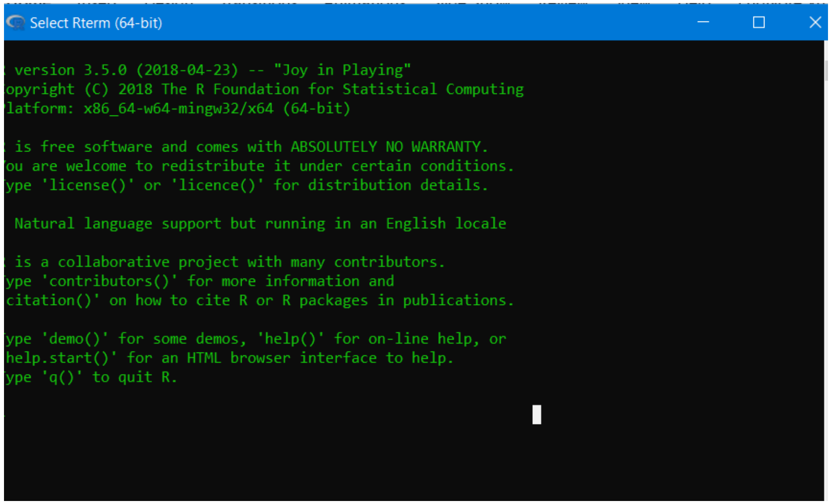
R workplace

R console



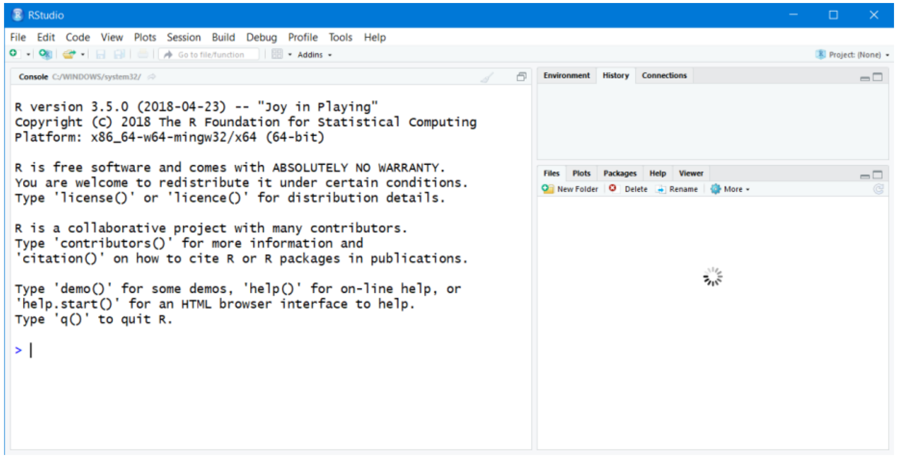
```
R version 3.5.0 (2018-04-23) -- "Joy in Playing"
Copyright (C) 2018 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (64-bit)

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.

  Natural language support but running in an English locale

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.

> 
```

R terminal



```
 version 3.5.0 (2018-04-23) -- "Joy in Playing"
opyright (C) 2018 The R Foundation for Statistical Computing
latform: x86_64-w64-mingw32/x64 (64-bit)

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

 Natural language support but running in an English locale

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

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

Rstudio: an integrated development environment (IDE) for R


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
Project: (None)
Console C:/WINDOWS/system32/
Environment History Connections
Files Plots Packages Help Viewer
New Folder Delete Rename More

```
R version 3.5.0 (2018-04-23) -- "Joy in Playing"
Copyright (C) 2018 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.

> |
```

Values in R

Numeric: 1

Character: “abc”

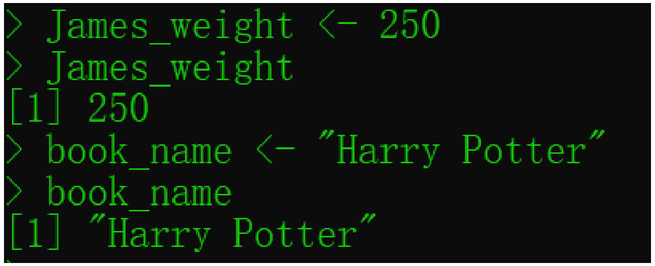
Logical: TRUE FALSE

Date: "2020-08-25"

R has objects

can be assigned using “<-” or “=”. Object can be a value, object, or function.

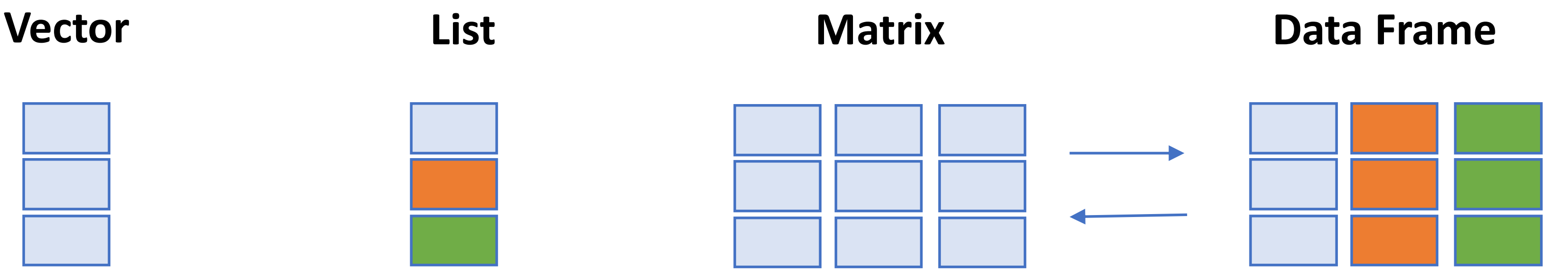
Try to assign an object:



```
> James_weight <- 250
> James_weight
[1] 250
> book_name <- "Harry Potter"
> book_name
[1] "Harry Potter"
```

How R store the values

Basic data structures in R



Vector

Elements should be the same type (numeric, character, logical or date)

List

Elements can be the same or different types

Matrix

Same type

Data Frame

Can be multiple types

Create and extract an object

<- is equal to =

- **Vector**: an ordered collection of single values of the same type, elements are accessed by the index.

  a <- 34

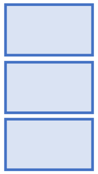
  b <- c(2, 4, 5) # c means concatenate

  b[1]

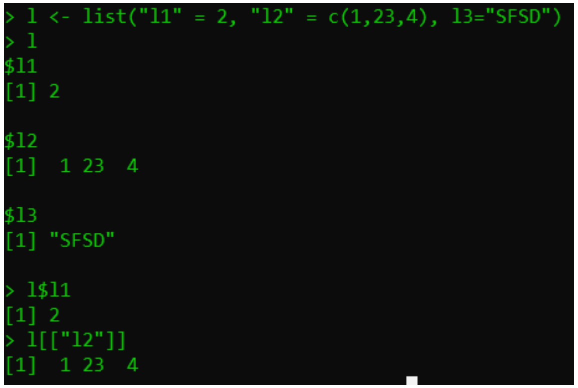
  fruits <- c(“apple”, “orange”, “strawberry”)

  Fruits[1] # object name is case sensitive

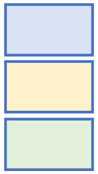
  ? What if I create a vector contains different types of values



- **List**: an ordered collection of elements of different type, elements are accessed by their names

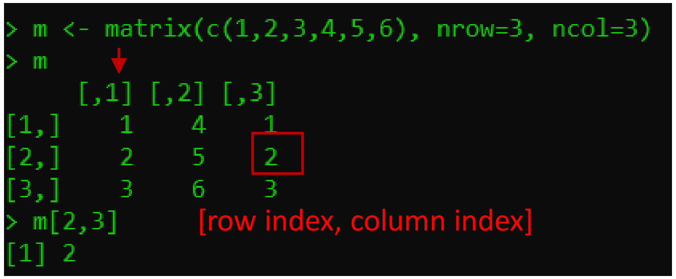


```
> l <- list("l1" = 2, "l2" = c(1,23,4), l3="SFSD")
> l
$l1
[1] 2

$l2
[1]  1 23  4

$l3
[1] "SFSD"

> l$l1
[1] 2
> l[["l2"]]
[1]  1 23  4
```



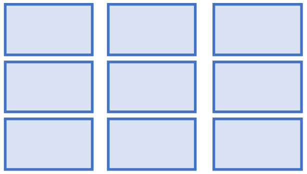
Create and extract object

- **Matrix**: a rectangular table of data of the same type

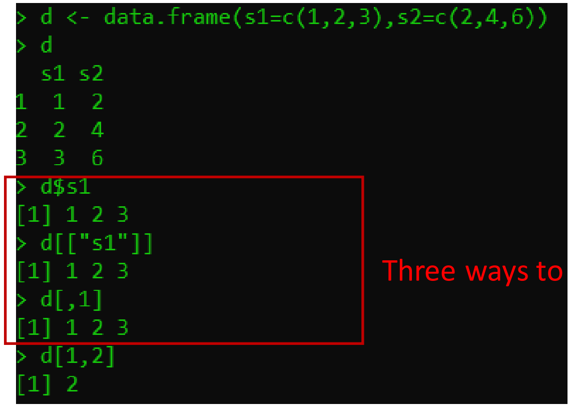


```
> m <- matrix(c(1,2,3,4,5,6), nrow=3, ncol=3)
> m
     [,1] [,2] [,3]
[1,]    1    4    1
[2,]    2    5    2
[3,]    3    6    3
> m[2,3]
[1] 2
```

[row index, column index]

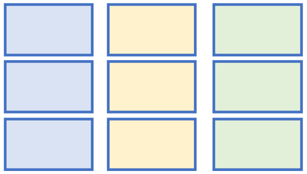


- **Data frame**: a rectangular table; each column has the same type, but different columns may have different types.



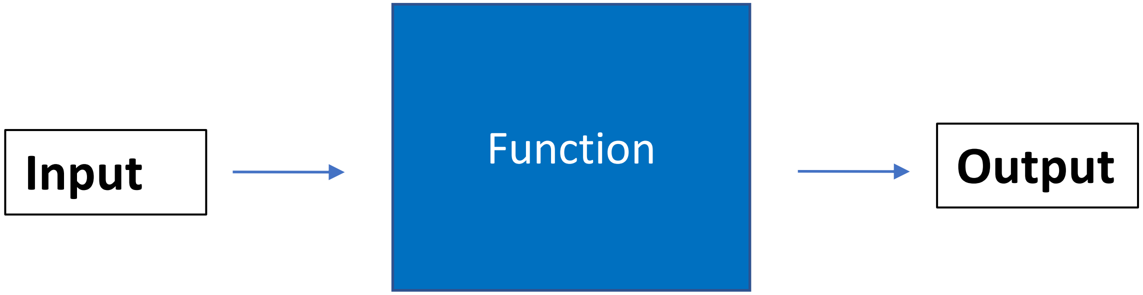
```
> d <- data.frame(s1=c(1,2,3),s2=c(2,4,6))
> d
  s1 s2
1  1  2
2  2  4
3  3  6
> d$s1
[1] 1 2 3
> d[["s1"]]
[1] 1 2 3
> d[,1]
[1] 1 2 3
> d[1,2]
[1] 2
```

Three ways to get the first column



R function

Function is like a box



do_this(input, parameters)

c, list, matrix, data.frame, mean(a)

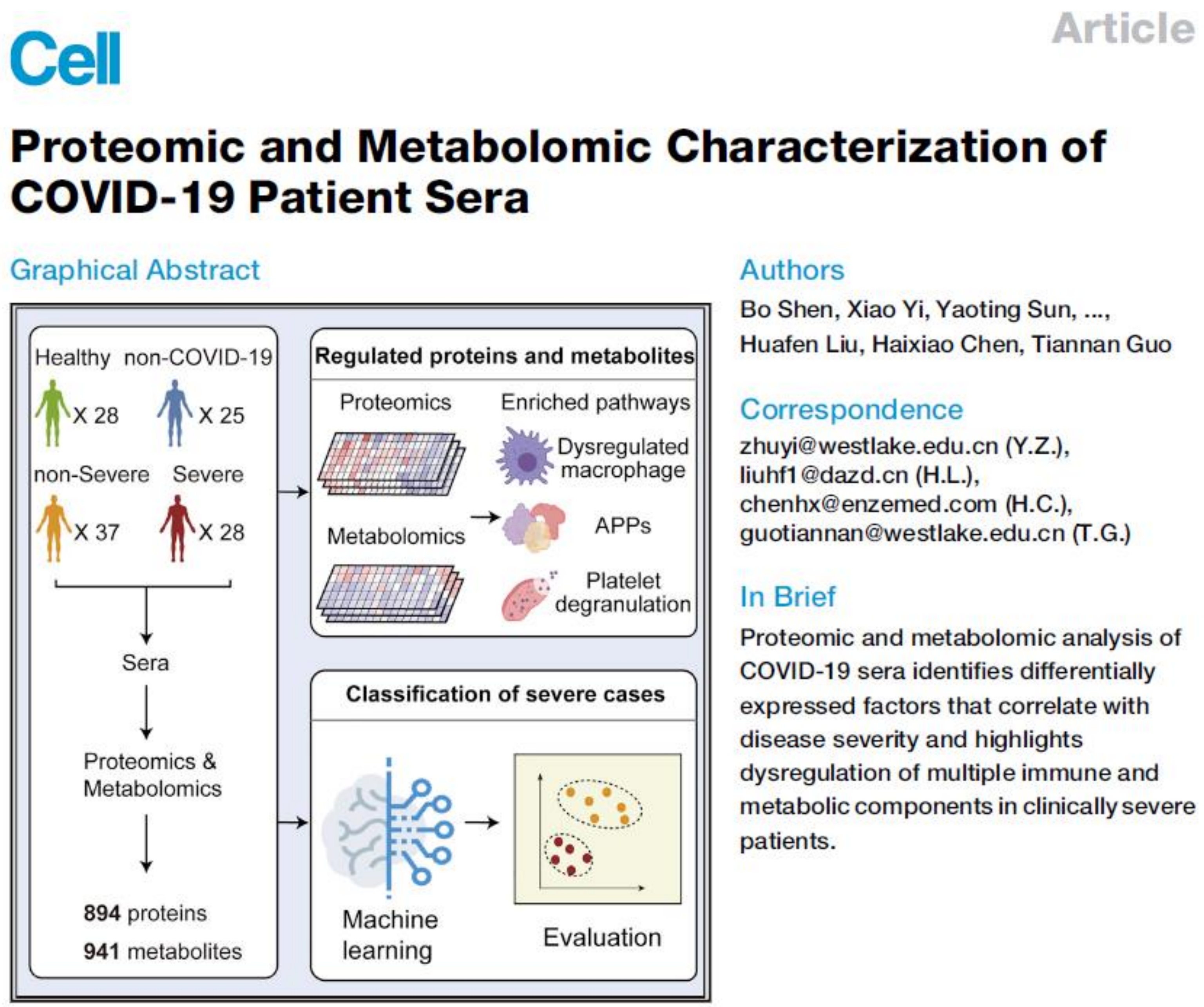
How to get more information about a built-in function: ?, help

How to define my own function:

```
cal_square  <- function(x) {
        y <- x * x
        return (y)  ## can be omitted.
}
cal_square(2)
```

Now let's play with R

Practice dataset:



Cell

Article

Proteomic and Metabolomic Characterization of COVID-19 Patient Sera

Graphical Abstract

Authors

Bo Shen, Xiao Yi, Yaoting Sun, ...,
Huafen Liu, Haixiao Chen, Tiannan Guo

Correspondence

zhuyi@westlake.edu.cn (Y.Z.),
liuhf1@dazd.cn (H.L.),
chenhx@enzemed.com (H.C.),
guotiannan@westlake.edu.cn (T.G.)

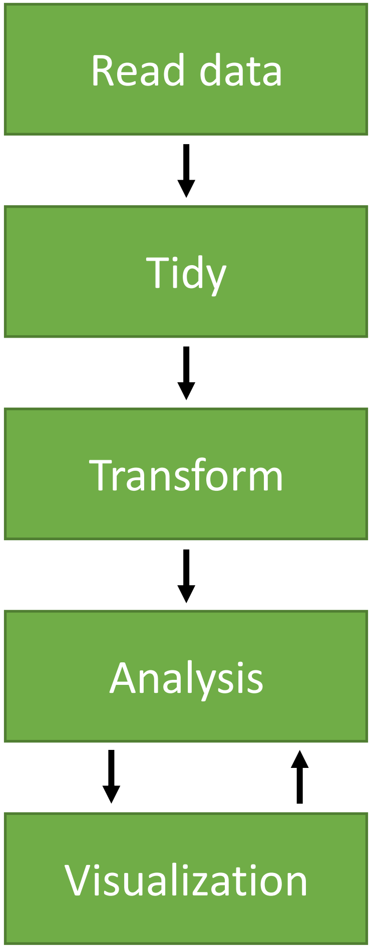
In Brief

Proteomic and metabolomic analysis of COVID-19 sera identifies differentially expressed factors that correlate with disease severity and highlights dysregulation of multiple immune and metabolic components in clinically severe patients.

Using R to perform analysis and visualize data

1. Read the dataset into R
2. Log2 transformation
3. Analyze the data (t-test)
4. Graphing and export

General data analysis flow



1. Read the dataset into R

Command: read.csv (for .csv file) .csv: separated by “,”

read.table (for .txt file) .txt: the columns separated by “ ” or “\t”

read_excel (for excel file) Need to load the readxl package

Parameters:

Filename: local file or an url

header: TRUE

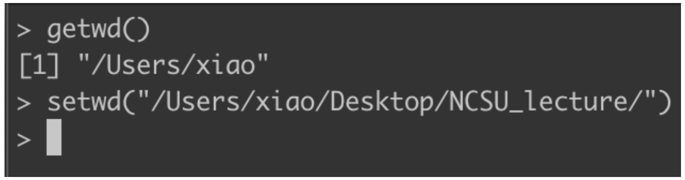
row.names: 1 (use the first column strings as the row name of dataset)

How to get your working directory?
How to change your working directory?
Write the full path of a file in windows system and unix like system.

More information about absolute path and relative path.

1. Read the dataset into R

use “getwd()” to get the default R working directory.
“setwd(path)” to change it



```
> getwd()
[1] "/Users/xiao"
> setwd("/Users/xiao/Desktop/NCSU_lecture/")
> 
```

Read the dataset and store it into a variable (for example mydata):

```
covid19_data <- read.csv(“RawData_plasma metabolites.csv”, header=TRUE,
row.names=1)
```

or

```
covid19_data <-
read.csv(“https://raw.githubusercontent.com/zhengtaoxiao/NCSU_R/master/
RawData_plasma metabolites.csv”, header=TRUE, row.names=1)
```

Change the data type from dataframe to matrix

```
covid19_data <- data.matrix(covid19_data)
```

2. Log2 transformation

Inspecting your dataset: View(covid19_data)

See also : head(covid19_data) → First 10 rows

tail(covid19_data) → Last 10 rows

Check the dimension of the data: dim

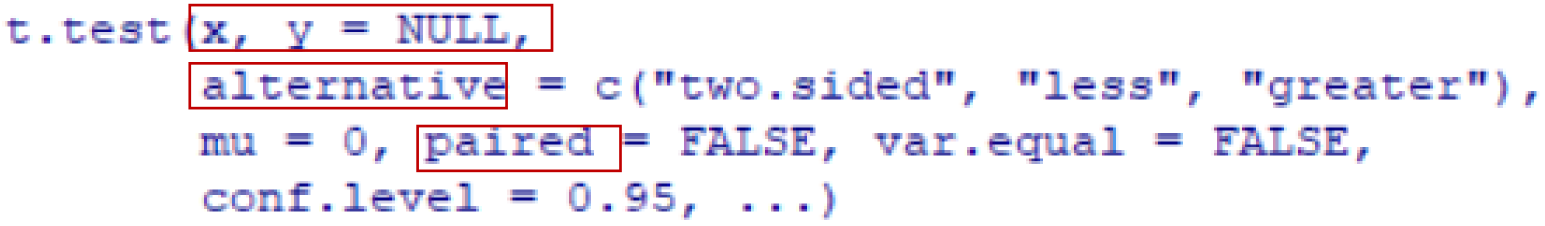
Log2 your dataset: covid19_log2_data <- log2(covid19_data+1)

The distribution of metabolites in a patients before and after log2 transformation:

```
hist(covid19_data)
hist(covid19_log2_data)
```

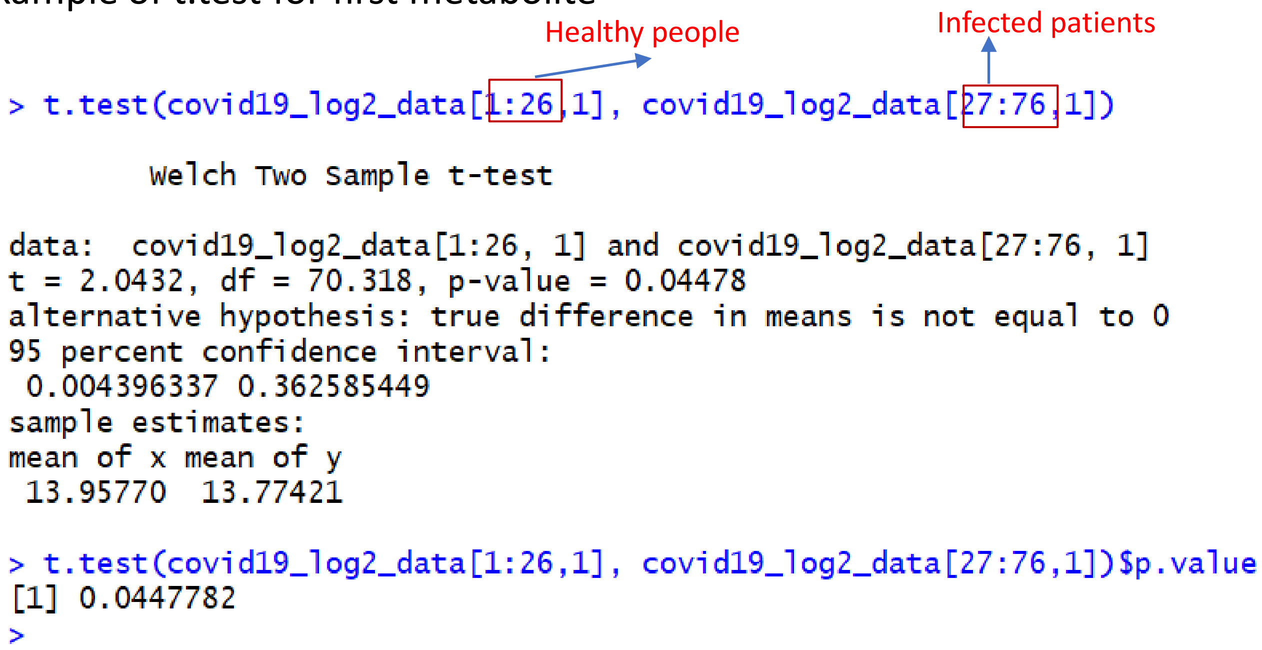
3. Analyze the data

Using of the t-test function in R



```
t.test(x, y = NULL,
       alternative = c("two.sided", "less", "greater"),
       mu = 0, paired = FALSE, var.equal = FALSE,
       conf.level = 0.95, ...)
```

Example of t.test for first metabolite



Healthy people

Infected patients

```
> t.test(covid19_log2_data[1:26,1], covid19_log2_data[27:76,1])

	Welch Two Sample t-test

data:  covid19_log2_data[1:26, 1] and covid19_log2_data[27:76, 1]
t = 2.0432, df = 70.318, p-value = 0.04478
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 0.004396337 0.362585449
sample estimates:
mean of x mean of y 
 13.95770  13.77421 

> t.test(covid19_log2_data[1:26,1], covid19_log2_data[27:76,1])$p.value
[1] 0.0447782
> 
```

The result of t.test is a list. Using the “$” to get the elements.

3. Analyze the data

We can define function to perform t-test on specific metabolite and return p-value

Function is like a box

Input

metabolite

Function

Output

p-value

```
myttest <- function(idx){
           res <- t.test(x = covid19_log2_data[1:26, idx],
                         y = covid19_log2_data[27:76, idx])

           return (res$p.value)

                               }
```

```
  myttest(1)
```

3. Analyze the data

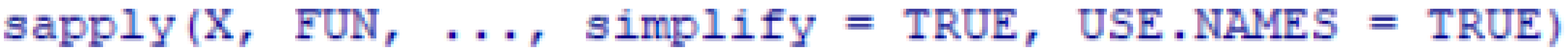
Perform test for all metabolites

```
pvals <- c()
for(i in 1:nrow(covid19_log2_data)){
  p <- myttest(i)
  pvals <- c(pvals,p)
}
```

1:10 generates a vector c(1,2,3,4,5,6,7,8,9,10). Here 1:nrow(mydata) return the row

indexes of mydata

Using of command sapply to perform t-test on each row



```
sapply(X, FUN, ..., simplify = TRUE, USE.NAMES = TRUE)
```

```
pvals <- sapply(X=1:ncol(covid19_log2_data),FUN=myttest)
```

nrow return the number of rows of a matrix or dataframe

3. Analyze the data

Let’s calculate the log2 fold change using the sapply

```
cal_log2fc <- function(idx){
           control_mean <- mean(covid19_data[1:26,idx])
           infected_mean <- mean(covid19_data[27:76,idx]
           return log2(infected_mean / control_mean)
}
```

```
log2_fcs <- sapply(X=1:ncol(covid19_data), FUN=cal_log2fc)
```

Learn more similar functions: apply, lapply → Return a list

↓

Input is an array or matrix

4. Graph and export: Volcano plot

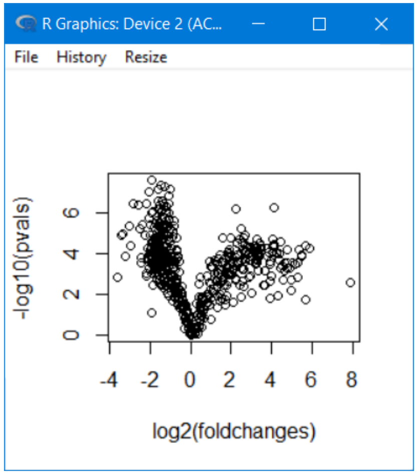
Command: plot

Parameters:

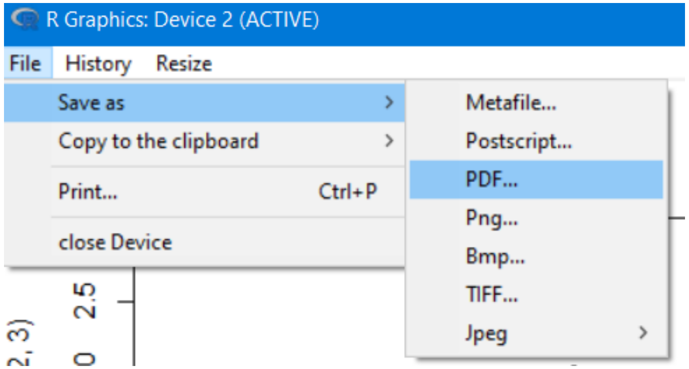
x (x axis)
y (y axis)

```
plot(x = log2_fcs , y = -log10(pvals))
```

Log2 transformed values are easy to understand: >0 means increased; <0 decreased;
-log10 transformed p-values: 0.1, 0.01, 0.001, 0.0001 → 1, 2, 3, 4



Export the graph



4. Graph and export: the t-test result

Command: write.csv (for .csv file)
write.table (for .txt file)

.csv: separated by “,”

.txt: the columns separated by “ ” or “\t”

Parameters:

Data: matrix or data frame
Filename: the name of file, default is print to console

Read the dataset and store it into a variable :

```
write.csv(cbind(log2_fcs, pvals), “ttest_result.csv”)
```

Using cbind to combine vectors by column and generate a matrix

Powerful package: ggplot

Focus on the analysis and data, using ggplot to create the graph quickly.

You provide the data, tell ggplot2 how to map variables to aesthetics,

what graphical primitives to use.

```
install.packages(“ggplot2”)
library(ggplot2)
group_data <- read.csv("group_info.csv", header=T,row.names=1)
plot_data <- data.frame(value=covid19_data[,1], + group=group_data[,1]
ggplot(plot_data, aes(x=group, y=value)) +
  geom_boxplot()
```

## try geom_col(), geom_violin(), geom_point(), geom_jitter()

ggplot: Add error bars to a bar

```
ggplot(plot_data, aes(x=group, y=value,fill=group)) +
  stat_summary(geom = "bar", fun = mean)+
  stat_summary(geom = "errorbar", fun.data = mean_se, width=0.5)
```

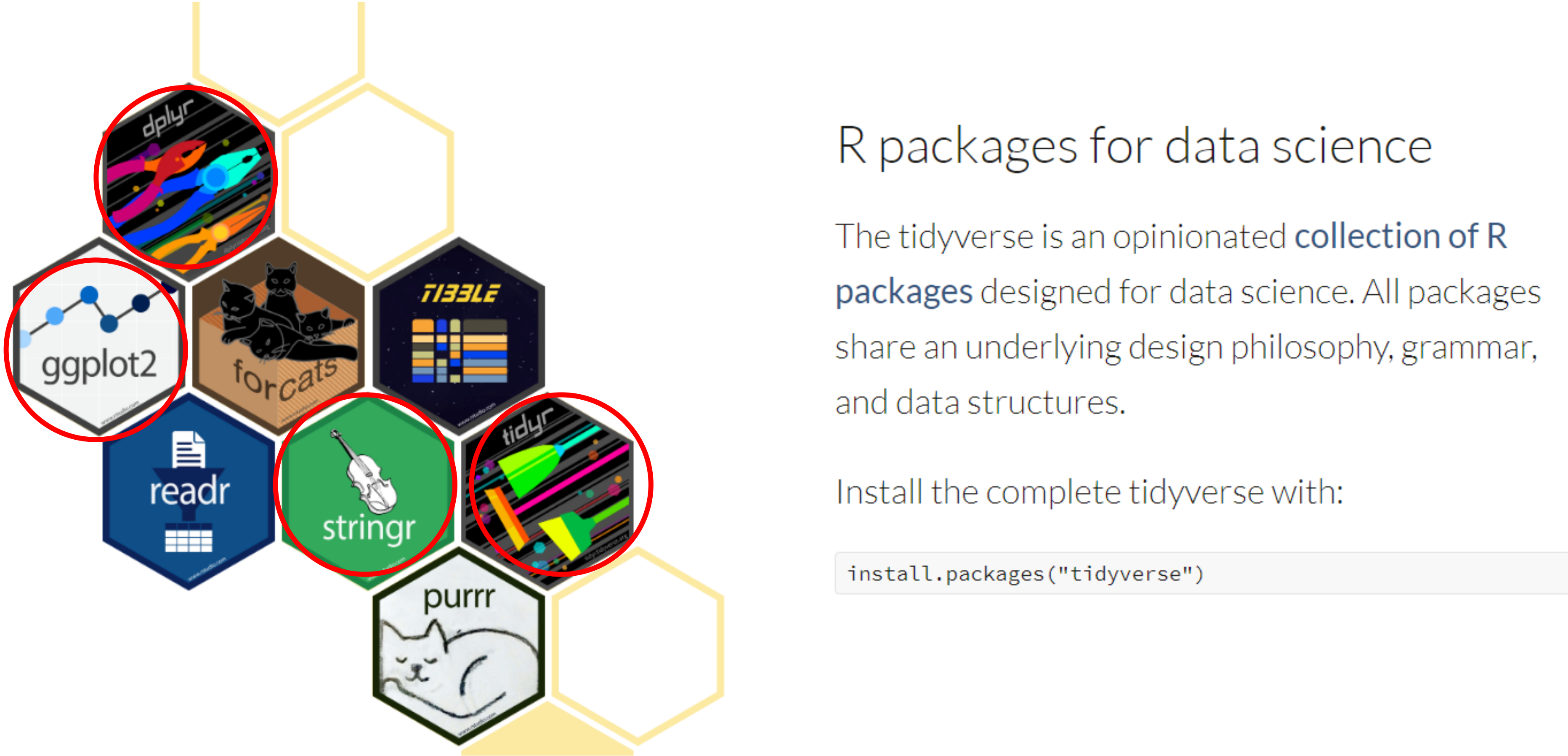
ggplot: more groups

```
plot_data2 <- covid19_data[,c("Citric.acid","Isocitric.acid",
                               "Fumaric.acid", "Succinic.acid")]
head(plot_data2)
library(reshape2)
plot_data2 <- melt(plot_data2) #Convert an object
head(plot_data2)
plot_data2$group <- rep(group_data[,1],4)
ggplot(plot_data2, aes(x=group, y=value, fill=variable)) +
  stat_summary(geom = "bar", fun = mean, position= "dodge")+
  stat_summary(geom = "errorbar", fun.data = mean_se, position="dodge")
```

ggplot: more groups, using the relative intensities

```
for(m in unique(plot_data2$variable)){
  control_mean <- mean(plot_data2[(plot_data2$variable == m) &
(plot_data2$group=="CONTROL"),"value"])
  plot_data2[plot_data2$variable == m, "relative"] <-
    plot_data2[plot_data2$variable == m, "value"] / control_mean
}
ggplot(plot_data2, aes(x=variable, y= relative, fill=group)) +
  stat_summary(geom = "bar", fun = mean, position= "dodge")+
  stat_summary(geom = "errorbar", fun.data = mean_se,
position="dodge") +
  scale_fill_manual(values =
c("#ffbfbf","#ff4040","#bf0000","#6f0000"))
```

More powerful packages



R packages for data science

The tidyverse is an opinionated collection of R packages designed for data science. All packages share an underlying design philosophy, grammar, and data structures.

Install the complete tidyverse with:

```
install.packages("tidyverse")
```

Any questions:

An Introduction to R

Notes on R: A Programming Environment for Data Analysis and Graphics
Version 3.6.1 (2019-07-05)

https://cran.r-project.org/doc/manuals/r-release/R-intro.pdf

https://www.r-project.org/help.html

stackoverflow

zhengtao.xiao@duke.edu