EC435 ภาค 1/2562

การสอนโปรแกรม R

เฉลิมพงษ์ คงเจริญ

คณะเศรษฐศาสตร์ ม.ธรรมศาสตร์

Why R?

- Powerful and flexible
- Free(open source)
- Extensive add-on software(package)
- Designed for statistical computing
- High level language

Installing R

- Install the latest version from: http//cran.r-project.org
- Install R Studio (Makes R easier) from http://www.rstudio.com

นอกจากนี้ เราสามารถใช้ Rstudio บน internet โดยไม่ต้องลงโปรแกรมผ่าน https://rstudio.cloud/

Working with R

- The R Console 'interprets' whatever you type
  - Calculator
  - Creating variables
  - Applying function
- "Analysis" Script+ Interactive Exploration
  - Static copy of what you did (reproducibility)
  - Try things out interactively, then add to your script
- R revolves around functions
  - Commands that take input, performs computations, and returns results
  - Many come with R Base (installation), but people write external functions you can download and use

R as a calculator

```
2+2

## [1] 4

2*4

## [1] 8

2^3

## [1] 8

2+(2*3)^2

## [1] 38
```

R variables

- You can create variables from within the R environment and from files on your computer
- R uses "=" or "<-" to assign values to a variable name
- Variable names are case-sensitive, i.e. X and x are different.

```
x<-2
x

## [1] 2
```

The 'combine' function – the function c( ) collects/combines/joins single R objects into a vector of R objects.

```
x<- c(1,4,6,8)
x

## [1] 1 4 6 8
```

What is a function?

The c( ) command is called a function: it takes inputs and gives an output. - In R, functions always go function(input) or name of function, then parentheses. The input can be many different things, such as function(x,y,z)

R Help

If you know the name of a function, help(function name)

Data Classes:

```
One dimensional classes('vectors')
```

- Character
- Numeric
- Integer
- Factor
- Logical Two dimensional classes
- data.frame: traditional 'Excel' spreadsheet
- Matrix: two-dimensional data, composed of rows and columns

Vector functions

```
z<-1:100
head(z)

## [1] 1 2 3 4 5 6

tail(z)

## [1]  95  96  97  98  99 100

str(z)

##  int [1:100] 1 2 3 4 5 6 7 8 9 10 ...
```

Data Subsetting

```
x1<- 10:20
length(x1)

## [1] 11

x1[1]

## [1] 10

x1[3:4]

## [1] 12 13

x1[c(1,5,7)]

## [1] 10 14 16
```

Matrices

```
m<-1:9
m

## [1] 1 2 3 4 5 6 7 8 9

mat<-matrix(m,nrow=3)
mat

##      [,1] [,2] [,3]
## [1,]    1    4    7
```

```
## [2,]    2    5    8
## [3,]    3    6    9

nrow(mat)

## [1] 3

ncol(mat)

## [1] 3

dim(mat)

## [1] 3 3
```

Data Selection

Matrices have tow 'slots' you can use to select data, which represent rows and columns, that are separated by a comma, so the syntax is matrix[row, column]

```
mat[1,1]

## [1] 1

mat[1, ]

## [1] 1 4 7

mat[ ,1]

## [1] 1 2 3
```

Data Frame

```
pttstock <- read.csv("https://raw.githubusercontent.com/chaleampong/EC435/master/pttstock.csv", stringsAsFactors=FALSE)
class(pttstock)

## [1] "data.frame"

names(pttstock)

## [1] "date"  "price"

str(pttstock)

## 'data.frame':    2089 obs. of  2 variables:
##  $ date : chr  "6/17/2003" "6/18/2003" "6/19/2003" "6/20/2003" ...
##  $ price: num  65.5 66 67 66.5 68 66.5 66 67.5 66 66.5 ...

head(pttstock$price)

## [1] 65.5 66.0 67.0 66.5 68.0 66.5
```

More on Packages

Packages are add-ons that are commonly written by users comprised of functions, data and vignettes. - Use library( ) or require( ) to load the package into memory so you can use its functions. - Install packages using install.packages("PackageName") . - Use help(package="PackageName") to see what contents the package has.

Import Data

In RStudio, we can import data files CSV, Excel, SPSS, SAS, Stata

Saving R data

It's very useful to be able to save collections of R objects for future analyses. Save(..., file="name.rda") When you close R, you might notice the prompt about saving your workspace in ".Rdata".

การคำนวณผลได้ตอบแทน

```
ptt.lret<-diff(log(pttstock$price))
n<-length(pttstock$price)
ptt.sret<-(pttstock$price[2:n]-pttstock$price[1:n-1])/(pttstock$price[1:n-1])
pttstock$sret<-c(NA,ptt.sret)
head(pttstock)
```

```
##        date price         sret
## 1 6/17/2003  65.5           NA
## 2 6/18/2003  66.0  0.007633588
## 3 6/19/2003  67.0  0.015151515
## 4 6/20/2003  66.5 -0.007462687
## 5 6/23/2003  68.0  0.022556391
## 6 6/24/2003  66.5 -0.022058824
```

```
write.csv(pttstock,file="pttstock.csv", row.names=FALSE)
```

Adding to data frames

```
pttstock$lret<-c(NA,ptt.lret)
head(pttstock)
```

```
##        date price         sret         lret
## 1 6/17/2003  65.5           NA           NA
## 2 6/18/2003  66.0  0.007633588  0.007604599
## 3 6/19/2003  67.0  0.015151515  0.015037877
## 4 6/20/2003  66.5 -0.007462687 -0.007490672
## 5 6/23/2003  68.0  0.022556391  0.022305758
## 6 6/24/2003  66.5 -0.022058824 -0.022305758
```

Adding rows and columns

```
m1<-matrix(1:9, nrow=3)
m2<-matrix(10:18, nrow=3)
cbind(m1,m2)
```

```
##      [,1] [,2] [,3] [,4] [,5] [,6]
## [1,]    1    4    7   10   13   16
## [2,]    2    5    8   11   14   17
## [3,]    3    6    9   12   15   18
```

```
rbind(m1,m2)
```

```
##      [,1] [,2] [,3]
## [1,]    1    4    7
## [2,]    2    5    8
## [3,]    3    6    9
## [4,]   10   13   16
## [5,]   11   14   17
## [6,]   12   15   18
```

Other manipulations

- abs(x)
- sqrt(x)
- log(x)
- log10(x)
- exp(x)

Data Summarization

Basic statistical summarization

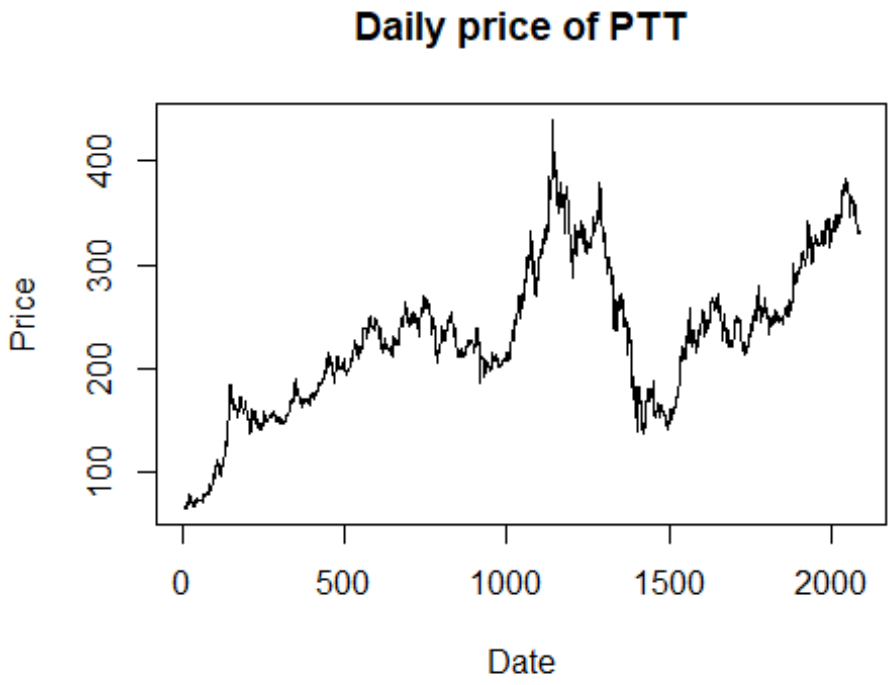
- mean(x)
- sd(x)
- median(x)
- quantile(x)
- range(x)

Basic summarization plots

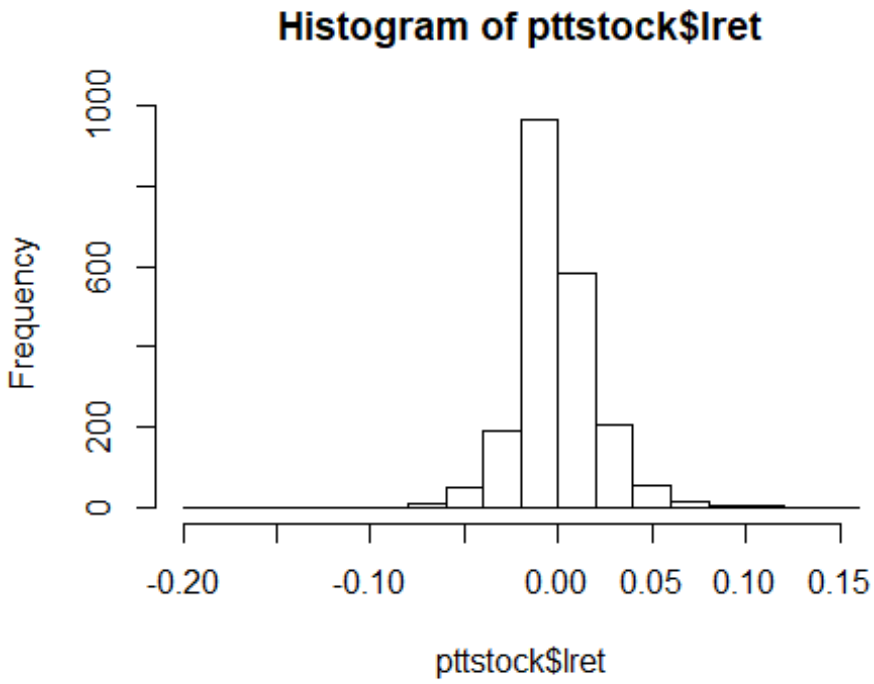
- plot(x,y)
- hist(x)
- plot(density(x))

Basic plot

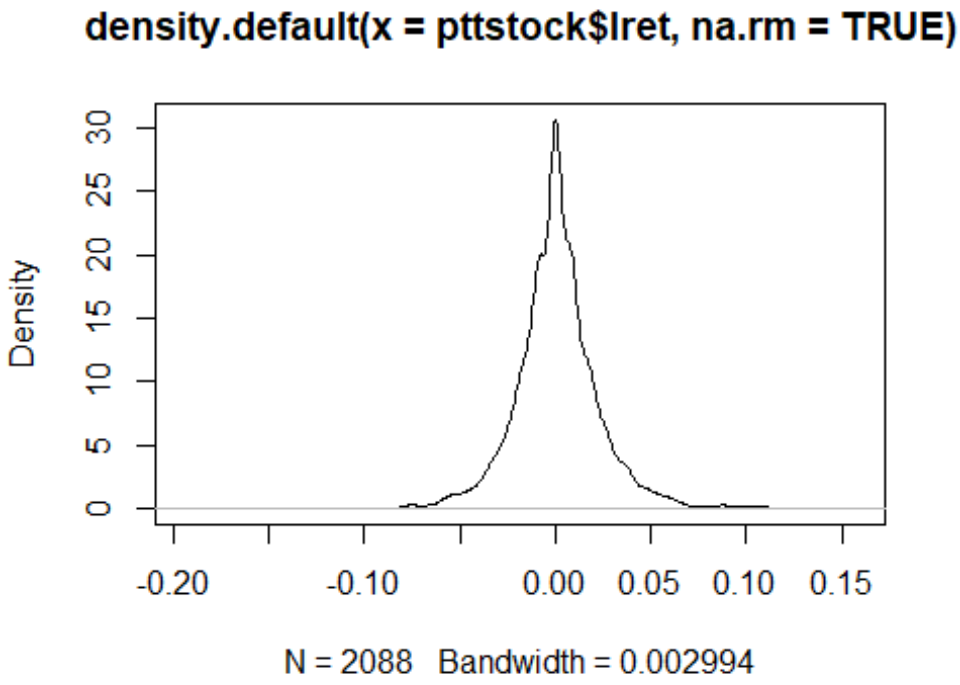
```
plot(pttstock$price, type="l", main="Daily price of PTT", xlab="Date",
ylab="Price")
```




```
hist(pttstock$lret)
```




```
plot(density(pttstock$lret, na.rm=TRUE))
```



Probability Distributions

- r for random number generation [e.g. rnorm( )]
- d for density [e.g. dnorm( )]
- p for probability [e.g. pnorm( )]
- q for quantile [e.g. qnorm( )]

Basic Statistics and tests

```
mean(pttstock$lret)
```

```
## [1] NA
```

```
mean(pttstock$lret, na.rm = TRUE)
```

```
## [1] 0.0007744456
```

```
sd(pttstock$lret, na.rm= TRUE)
```

```
## [1] 0.02240456
```

```
library(fBasics)
```

```
## Warning: package 'fBasics' was built under R version 3.5.3
```

```
## Loading required package: timeDate
```

```
## Warning: package 'timeDate' was built under R version 3.5.2
```

```
## Loading required package: timeSeries
```

```
## Warning: package 'timeSeries' was built under R version 3.5.3
```

```
s3<-skewness(pttstock$lret, na.rm= TRUE)
s3
```

```
## [1] -0.06886163
## attr(,"method")
## [1] "moment"
```

```
T<-nrow(pttstock)
t3<-s3/sqrt(6/T)
t3
```

```
## [1] -1.284905
## attr(,"method")
## [1] "moment"
```

```
k4<-kurtosis(pttstock$lret, na.rm= TRUE)
k4
```

```
## [1] 6.228529
## attr(,"method")
## [1] "excess"
```

```
t4<-k4/sqrt(24/T)
t4
```

```
## [1] 58.10976
## attr(,"method")
## [1] "excess"
```

```
normalTest(pttstock$lret,method=c("jb"))
```

```
##
## Title:
##  Jarque - Bera Normalality Test
##
## Test Results:
##   STATISTIC:
##   P VALUE:
##
## Description:
##  Mon Aug 26 12:14:39 2019 by user: User
```

```
basicStats(pttstock$lret)
```

```
##             X..pttstock.lret
## nobs             2089.000000
## NAs                 1.000000
## Minimum            -0.185899
## Maximum             0.149532
## 1. Quartile        -0.009569
## 3. Quartile         0.010989
## Mean                0.000774
## Median              0.000000
## Sum                 1.617043
## SE Mean             0.000490
## LCL Mean           -0.000187
## UCL Mean            0.001736
## Variance            0.000502
## Stdev               0.022405
## Skewness           -0.068862
## Kurtosis            6.228529
```

Package zoo และ PerformanceAnalytics

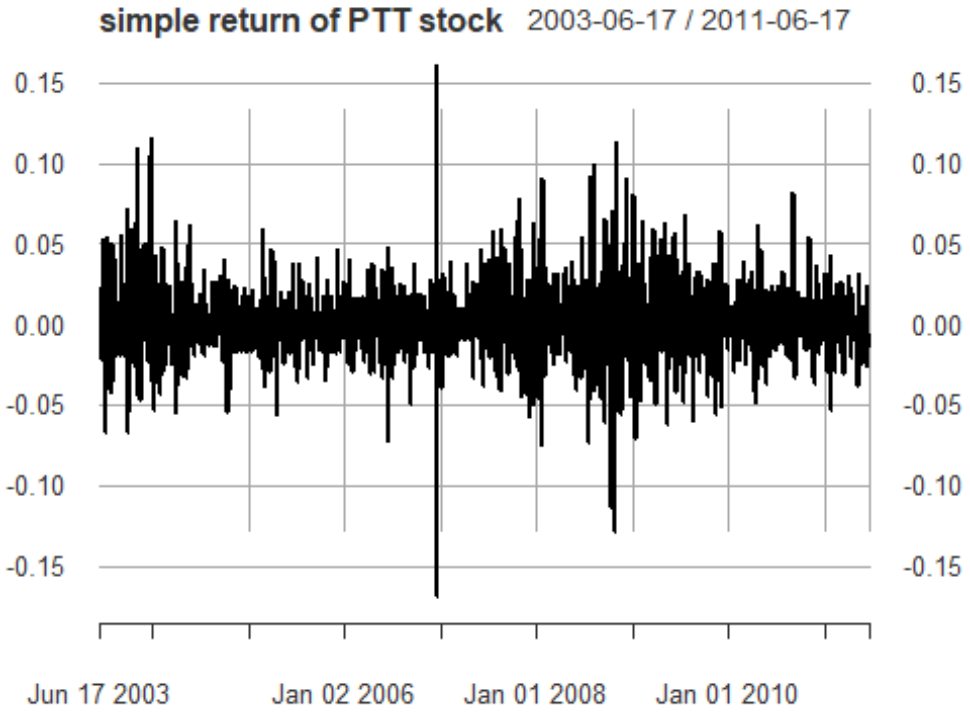
```
library(PerformanceAnalytics)
```

```
## Warning: package 'PerformanceAnalytics' was built under R version 3.5.3
```

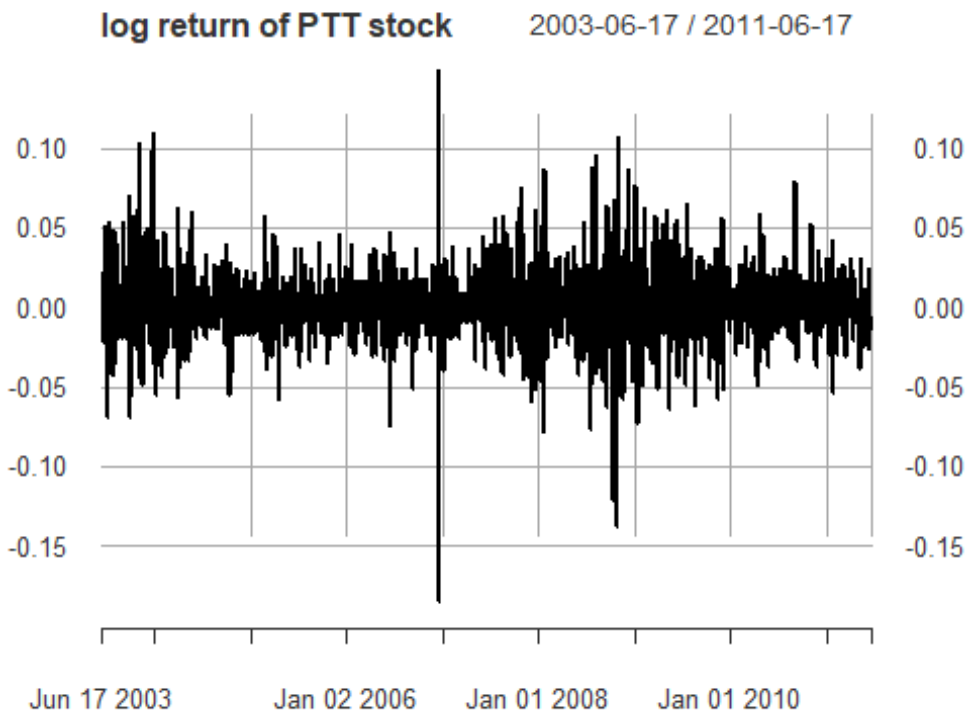
```
## Loading required package: xts
```

```
## Warning: package 'xts' was built under R version 3.5.2

## Loading required package: zoo

## Warning: package 'zoo' was built under R version 3.5.2

##
## Attaching package: 'zoo'

## The following object is masked from 'package:timeSeries':
##
##     time<-

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric

##
## Attaching package: 'PerformanceAnalytics'

## The following objects are masked from 'package:timeDate':
##
##     kurtosis, skewness

## The following object is masked from 'package:graphics':
##
##     legend

library(xts)
#convert data.frame to xts
pttstock <- read.csv("https://raw.githubusercontent.com/chaleampong/EC435/mas
ter/pttstock.csv", stringsAsFactors=FALSE)
class(pttstock$date)

## [1] "character"

pttstock$date<-as.Date(as.character(pttstock$date), "%m/%d/%Y")
pttstock.xts<-xts(pttstock$price, order.by=pttstock$date)
head(pttstock.xts)

##            [,1]
## 2003-06-17 65.5
## 2003-06-18 66.0
## 2003-06-19 67.0
## 2003-06-20 66.5
## 2003-06-23 68.0
## 2003-06-24 66.5

plot(pttstock.xts)
```


```
pttstock.xts.sret<-Return.calculate(pttstock.xts, method=c("discrete"))
plot(pttstock.xts.sret, main = "simple return of PTT stock")
```



```
pttstock.xts.lret<-Return.calculate(pttstock.xts, method=c("log"))
plot(pttstock.xts.lret, main="log return of PTT stock")
```



การทำงานบน R สามารถเปลี่ยน Folder(Directory) ที่ทำงานได้ โดยเราสามารถตรวจสอบด้วยคำสั่ง getwd( ) และเปลี่ยนด้วยคำสั่ง setwd("Folder")

```
getwd()

## [1] "G:/My Drive/teaching/ec435/notes"

#setwd()
```

เราสามารถ Save Data.Frame ได้ด้วยคำสั่ง write.csv โดยไฟล์ดังกล่าวจะปรากฏใน Folder ที่เราทำงาน

```
write.csv(pttstock, file="pttreturn.csv")
```

นอกจากนี้เราสามารถ Save ทุกอย่างที่เราทำงานได้ด้วยคลิก File>Save และตั้งชื่อไฟล์แล้ว Save (ไฟล์จะอยู่ในนามสกุล .Rdata)

```
save.image(file="pttsession.Rdata")
```