Getting start with R

Ting-Shuo Yo

September 23, 2016

大綱

- 什麼是 R？能吃嗎？
- 動手前的前置作業
- 安裝 R / Rstudio
- 實際的例子，動手做做看
- 基本概念

為什麼要用 R ？

- 免費
- 整合開發環境
- 套件豐富
    - 資料讀取
    - 資料清理
    - 分析工具
    - 繪圖報表
- 開發者社群資源
- 套件之間整合度高

R Programming

- 變數型態
- 輸入/輸出
- 控制結構
- 函數
- 進階控制結構
- 除錯
- 最佳化

動手做做看

下載資料

http://opendata.epa.gov.tw/Data/DownloadFile/ATM00240/

```
furl <- "http://service.dataqualia.com/misc/test01.csv"
download.file(furl, destfile="test01.csv")
```

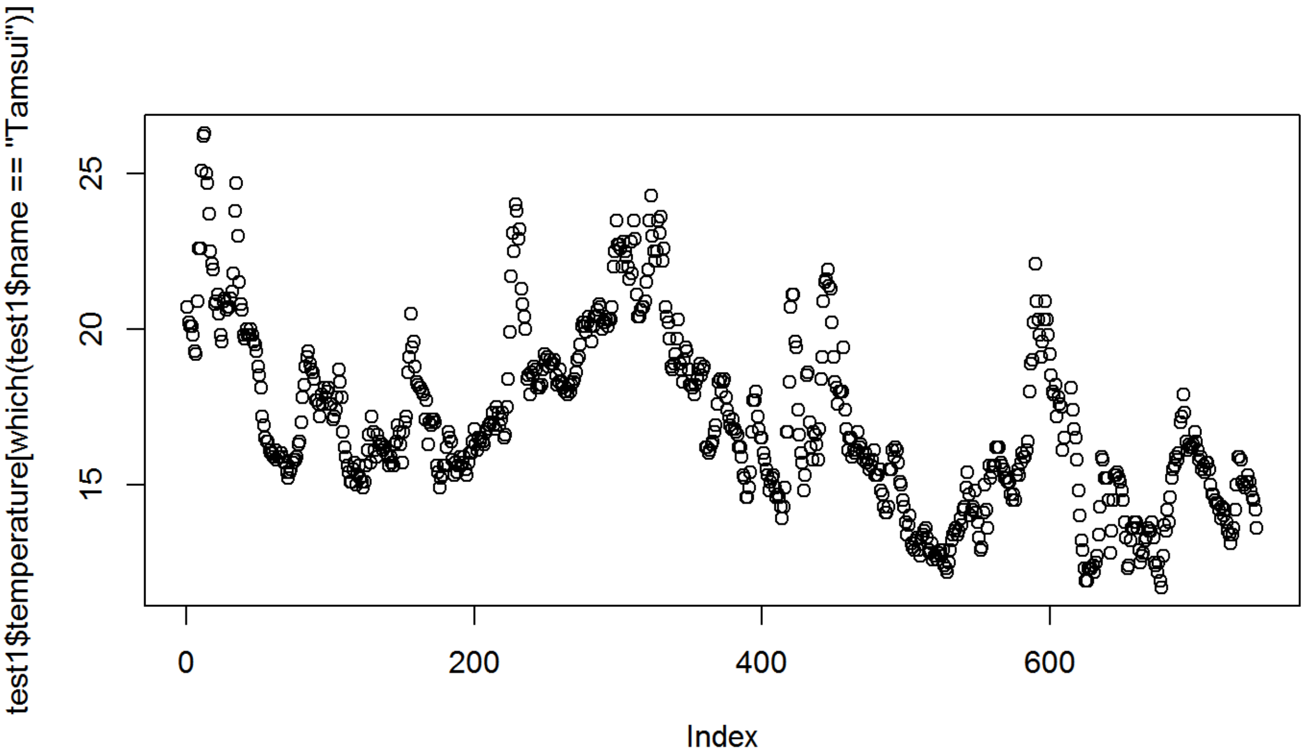
讀取資料

```
data1 <- read.csv("test01.csv", header = F, encoding="UTF-8", stringsAsFactors=F)
test1 <- data.frame("id"=data1$V1, "name"=data1$V2, "time"=as.Date(data1$V4), "pressure"=as.numeric(data
head(test1)
```

```
##       id   name       time pressure temperature rh wind_speed
## 1 466900 Tamsui 2015-12-23   1018.6        20.7 88        2.6
## 2 466900 Tamsui 2015-12-23   1018.7        20.2 89        1.8
## 3 466900 Tamsui 2015-12-23   1018.4        20.1 88        1.8
## 4 466900 Tamsui 2015-12-23   1018.1        20.1 88        0.7
## 5 466900 Tamsui 2015-12-23   1017.4        19.8 88        0.0
## 6 466900 Tamsui 2015-12-23   1017.8        19.3 90        0.3
##   wind_direction
## 1             NW
## 2             NW
```

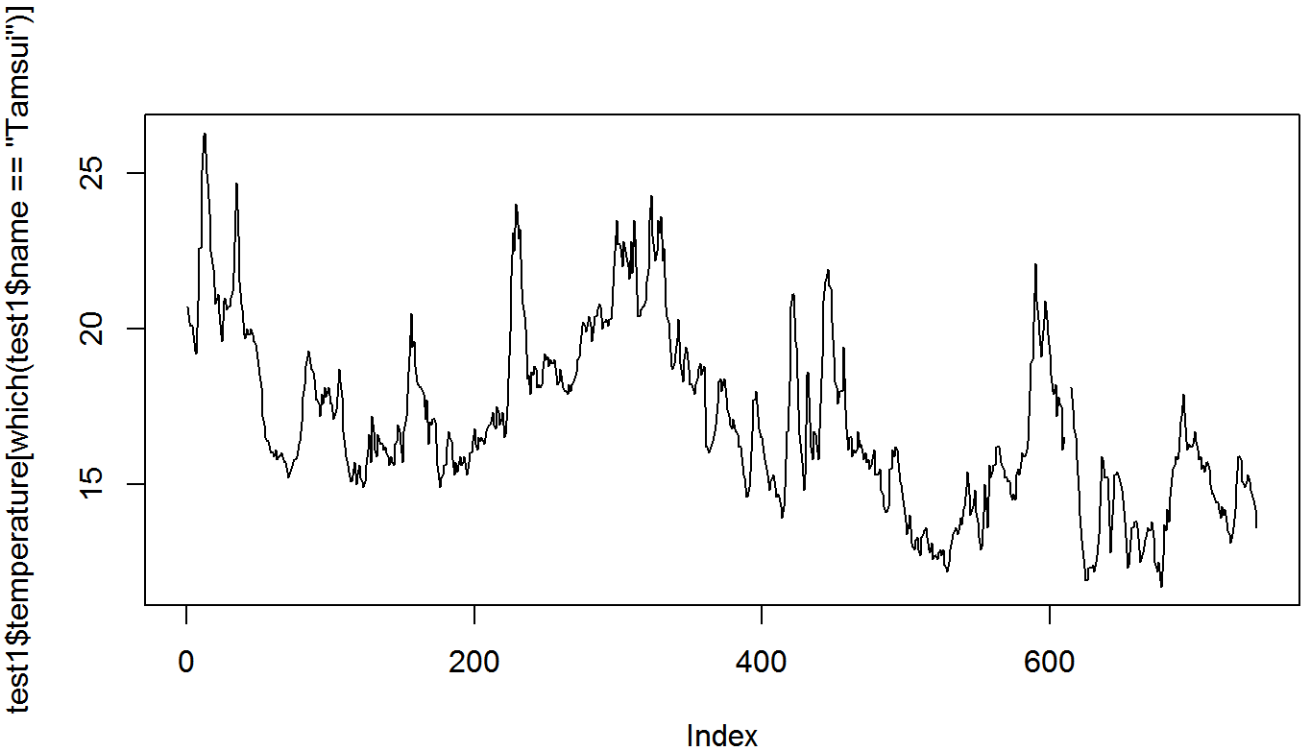
簡單繪圖 1/2

```
plot(test1$temperature[which(test1$name=="Tamsui")])
```



簡單繪圖 2/2

```
plot(test1$temperature[which(test1$name=="Tamsui")], type="line")
```



快速查詢手冊

?[函數名稱] = help([函數名稱])

```
help(plot, help_type="text")
?plot
```

args(function) = 顯示函數的輸入/輸出參數

```
args(plot)
```

```
## function (x, y, ...)
## NULL
```

常用系統指令 1/2

- `getwd()` = 顯示目前工作目錄
- `setwd()` = 設定目前工作目錄
- `dir.create("path/foldername", recursive = TRUE)` = 產生新目錄
- `unlink(directory, recursive = TRUE)` = 刪除目錄
- `list.files(recursive = TRUE)` = 顯示目前目錄的所有檔案

常用系統指令 **2/2**

- `ls()` = 顯示目前工作環境裡的所有物件
- `file.create("name")` = create file
  - `exists("name")` = return true/false exists in working directory
  - `info("name")` = return file info
  - `info("name")$property` = returns value for the specific attribute
  - `rename("name1", "name2")` = rename file
  - `copy("name1", "name2")` = copy file
  - `path("name1")` = return path of file

R 常見符號

- `<-` = assignment operator
- `#` = comment
- 輸入算式並按下 enter,即會自動計算結果,並且輸出在螢幕上
- 輸入變數的名稱並按下 enter = `print(x)`
- `[1]` 會出現在螢幕輸出的最前面

```
temp.tamsui <- data1$V6[which(data1$V2=="Tamsui")]
temp.tamsui
```

```
##   [1] "20.7" "20.2" "20.1" "20.1" "19.8" "19.3" "19.2" "20.9" "22.6" "22.6"
##  [11] "25.1" "26.2" "26.3" "25"   "24.7" "23.7" "22.5" "22.1" "21.9" "20.8"
##  [21] "20.9" "21.1" "20.5" "19.8" "19.6" "20.9" "21"   "20.6" "20.7" "20.7"
##  [31] "21"   "21.2" "21.8" "23.8" "24.7" "23"   "21.5" "20.8" "20.6" "19.8"
##  [41] "19.7" "20"   "19.8" "19.8" "20"   "19.8" "19.6" "19.5" "19.3" "18.8"
##  [51] "18.5" "18.1" "17.2" "16.9" "16.5" "16.4" "16.4" "16.1" "16"   "16"
##  [61] "15.9" "16.1" "15.8" "15.9" "15.9" "16"   "15.9" "15.7" "15.7" "15.4"
##  [71] "15.2" "15.4" "15.5" "15.7" "15.8" "15.8" "15.9" "16.3" "16.4" "17"
##  [81] "17.8" "18.2" "18.8" "19.1" "19.3" "18.9" "18.7" "18.6" "18.4" "17.7"
##  [91] "17.7" "17.6" "17.2" "17.9" "17.6" "18.1" "17.8" "18"   "18.1" "17.6"
## [101] "17.6" "17.1" "17.2" "17.4" "17.8" "18.7" "18.3" "17.8" "16.7" "16.2"
## [111] "15.9" "15.6" "15.4" "15.1" "15.1" "15.5" "15.7" "15"   "15.3" "15.6"
```

R 語言基礎概念

基本資料型態

- 基本物件
- Vectors and List
- Matrices and Data Frames
- Arrays
- Factors
- Missing Values
- Subsetting

R基本物件

- 5 種基本物件
  1. character
  2. numeric
  3. integer
  4. complex
  5. logical

```r
a <- "a"
is.character(a)
```

```
## [1] TRUE
```

```r
is.integer(a)
```

```
## [1] FALSE
```

R數值資料

- 簡單的數值變數型態
  - 浮點數一律以 `numeric` 物件表示 (double precision)
  - 整數`Integer`一律是長整數，可用數字後面加 `L` 表示(ex. `1L`)
  - `Inf` = 無窮大，可以在運算中使用
  - `NaN` = 非數值資料
- 豐富的數學和統計函數
  - `sqrt(value)` = 開根號
  - `sum(numbers)` / `mean(numbers)` = 加總/平均數
  - `var(numbers)` / `sd(numbers)` = 變異數 / 標準差
  - `cor(A, B)` / `cov(A,B)` = A 與 B 兩串資料的相關係數 / 共變數
  - `prcomp(matrix)` = 主成分分析
  - `fa(matrix)` = 因素分析

Vectors and Lists 1/3

- atomic vector: 數個同型態的資料

```
vector <- c(1,2,3,4,5)      # c()=concatenate
print(vector)
```

```
## [1] 1 2 3 4 5
```

```
print(vector * vector)      # 元件相乘 = vector[1] * vector[1] + ...
```

```
## [1]  1  4  9 16 25
```

```
print(t(vector) %*% vector) # 向量乘法
```

```
##      [,1]
## [1,]   55
```

Vectors and Lists 2/3

- list() = 特殊的 vector, 可包含不同型態的成員

```
l <- list("id"=101, "name"="John Doe", "scores"=c(8, 7, 8, 3, 9))
print(l)
```

```
## $id
## [1] 101
##
## $name
## [1] "John Doe"
##
## $scores
## [1] 8 7 8 3 9
```

```
sum(l$scores)    # = sum(l[[3]])
```

```
## [1] 35
```

Vectors and Lists 3/3

- 資料型態轉換

```
x <- "0"
as.numeric(x)
```

```
## [1] 0
```

```
as.logical(x)
```

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

```
as.complex(x)
```

```
## [1] 0+0i
```

Matrices and Data Frames 1/3

- `matrix` 包含同型態的資料, `data.frame` 可包含不同型態的資料
- `matrix(values, nrow = n, ncol = m)`

```
x <- c(1,2,3,4,5,6)                # x is a vector
mx <- matrix(x, nrow=3, ncol=2)   # mx is a 3x2 matrix
mx
```

```
##      [,1] [,2]
## [1,]    1    4
## [2,]    2    5
## [3,]    3    6
```

```
dim(mx)                            # get the dimension with dim()
```

```
## [1] 3 2
```

Matrices and Data Frames 2/3

```
# initiate a vector
x <-c(NA, 1, "cx", NA, 2, "dsa")
class(x)
```

```
## [1] "character"
```

```
# convert to matrix
dim(x) <- c(3, 2)
class(x)
```

```
## [1] "matrix"
```

Matrices and Data Frames 3/3

- `data.frame` 可包含不同型態的資料

```
df <- data.frame("id"=101:105, "name"=c("Alex","Bob","Carl","Dan","Eve"))
df
```

```
##    id name
## 1 101 Alex
## 2 102  Bob
## 3 103 Carl
## 4 104  Dan
## 5 105  Eve
```

```
df$name
```

```
## [1] Alex Bob  Carl Dan  Eve 
## Levels: Alex Bob Carl Dan Eve
```

Factors

- `factor`: 類別資料

```
df$name
```

```
## [1] Alex Bob  Carl Dan  Eve 
## Levels: Alex Bob Carl Dan Eve
```

```
as.numeric(df$name)
```

```
## [1] 1 2 3 4 5
```

```
levels(df$name)
```

```
## [1] "Alex" "Bob"  "Carl" "Dan"  "Eve"
```

- 在分析時要確認`character`跟`factor`的差異.
- `data1 <- read.csv("test01.csv", header = F, encoding="UTF-8", stringsAsFactors=F)`

Arrays

- `array(data, dim, dimnames)` 多維度資料

```
x <- 1:8
array(x, c(2,2,2))
```

```
## , , 1
## 
##      [,1] [,2]
## [1,]    1    3
## [2,]    2    4
## 
## , , 2
## 
##      [,1] [,2]
## [1,]    5    7
## [2,]    6    8
```

Missing Values

- `NaN` or `NA` = missing values
- `NaN` = undefined mathematical operations
- `NA` = any value not available or missing in the statistical sense
    - any operations with `NA` results in `NA`
    - `NA` can have different classes potentially (integer, character, etc)
    - Note: `NaN` is an `NA` value, but `NA` is not `NaN`
- `is.na()`, `is.nan()` = use to test if each element of the vector is `NA` and `NaN`
- `sum(my_na)` = sum of a logical vector (`TRUE` = 1 and `FALSE` = 0) is effectively the number of `TRUE`s

Removing NA Values

- `is.na()` = creates logical vector where T is where value exists, F is NA
  - subsetting with the above result can return only the non NA elements
- `complete.cases(obj1, obj2)` = creates logical vector where `TRUE` is where both values exist, and `FALSE` is where any is NA
  - can be used on data frames as well
  - `complete.cases(data.frame)` = creates logical vectors indicating which observation/row is good
  - `data.frame[logicalVector, ]` = returns all observations with complete data

Imputing Missing Values

- replacing missing values with estimates (can be averages from all other data with the similar conditions)

```
x <- c(1,2,3,NA,4,5)
x
```

```
## [1]  1  2  3 NA  4  5
```

```
is.na(x)
```

```
## [1] FALSE FALSE FALSE  TRUE FALSE FALSE
```

```
x[is.na(x)] <- 0
x
```

```
## [1] 1 2 3 0 4 5
```

Sequence of Numbers

```
1:10    # creates a sequence of numbers from first number to second number
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10
```

```
seq(1, 10, by=2)
```

```
## [1] 1 3 5 7 9
```

```
rep(0, times=10)
```

```
##  [1] 0 0 0 0 0 0 0 0 0 0
```

Subsetting

- R uses *one based index* –> starts counting at 1
- `[]` = 用來指定`vector`裡的成員，也可以同時指定多個成員，例如： `[1:2]`
- `[[]]` = 用來指定`list / data.frame`裡的成員
- `$` = 用來指定`list / data.frame`裡有`name`的成員

Indexing 1/2

- `data[x, y, ...]` can be used to index a specific element in a data collection (array, matrix, or data.frame)

```
x <- array(1:8, c(4,2))
x
```

```
##      [,1] [,2]
## [1,]    1    5
## [2,]    2    6
## [3,]    3    7
## [4,]    4    8
```

```
x[1,2]
```

```
## [1] 5
```

Indexing 2/2

- Use "sapce" to indicate "every element" in the data collection.

```
x <- array(1:8, c(4,2))
x
```

```
##      [,1] [,2]
## [1,]    1    5
## [2,]    2    6
## [3,]    3    7
## [4,]    4    8
```

```
x[2:3,]
```

```
##      [,1] [,2]
## [1,]    2    6
## [2,]    3    7
```

學寫程式沒有捷徑，**just do it!**

Quiz 1

- 在命令列輸入 2**4，會得到：

  1. 2
  2. 4
  3. 16
  4. 32

Quiz 2

- 要從 data frame test1 裡挑選前兩欄，應該用以下哪個指令：

1. test1[1,2]
2. test1[c(1,2)]
3. test1[,c(1,2)]
4. test1[c(1,2),]

Quiz 3

- 在命令列輸入 `2+NA`，會得到：

1. `NA`
2. `NULL`
3. 2
4. 0

Quiz 4

- 在命令列輸入 `x <- 4`，然後`class(x)`會得到：

  1. character
  2. numeric
  3. integer
  4. logical

Quiz 5

- 在命令列輸入 `x <- c(4, "a", TRUE)`，然後`class(x)`會得到：

  1. character
  2. numeric
  3. integer
  4. logical

Quiz 6

- 在命令列輸入 `x <- c(1,3,5)`，`y <- c(3,2,10)`，然後`cbind(x,y)`會得到什麼？

```
x <- c(1,3,5)
y <- c(3,2,10)
rbind(x,y)
```

```
##   [,1] [,2] [,3]
## x    1    3    5
## y    3    2   10
```

Quiz 7

- 在命令列輸入`x <- list(2, "a", "b", TRUE)`，`x[[1]]`是什麼？

```r
x <- list(2, "a", "b", TRUE)
x
```

```
## [[1]]
## [1] 2
## 
## [[2]]
## [1] "a"
## 
## [[3]]
## [1] "b"
## 
## [[4]]
## [1] TRUE
```

Quiz 8

- 在命令列輸入 `x <- 1:4`，`y <- 2:3`，請問`x + y`的答案是什麼？

```r
x <- 1:4
y <- 2:3
x + y
```

```
## [1] 3 5 5 7
```

Quiz 9

- `x <- c(17, 14, 4, 5, 13, 12, 10)`，如果希望將其中大於10的元素都換成4，應該用哪個指令？

```r
x <- c(17, 14, 4, 5, 13, 12, 10)
x > 10
```

```
## [1]  TRUE  TRUE FALSE FALSE  TRUE  TRUE FALSE
```

```r
x[x > 10] <- 4
x
```

```
## [1]  4  4  4  5  4  4 10
```