R language basics, part 2

HUST Bioinformatics course series

Wei-Hua Chen (CC BY-NC 4.0)

07 September, 2022

section 1: TOC

前情提要

vector & matrix:

- declaration
- manipulation
- arithmetic
- transposition

vectorization

- every is a vector!!
- vectorization verses loop (will be explained later)
- advantages using vectorization

今次预报

1. data.frame, tibble
2. read files from harddrive (IO)
3. exercises & homework

section 2: data.frame and tibble

data.frame, outline

1. what is a data.frame???
2. how to make a data.frame
3. how to add row(s)/col(s) to an existing data.frame how to combine two data.frames
4. how to manipulate a data.frame

2.1 what is a data.frame?

眼见为实:

```
library(tidyverse); ## 装入包
library(kableExtra);
kbl( head(mpg), booktabs = T); ## 显示前几行数据
```

manufacturer	model	displ	year	cyl	trans	drv	cty	hwy	fl	class
audi	a4	1.8	1999	4	auto(l5)	f	18	29	p	compact
audi	a4	1.8	1999	4	manual(m5)	f	21	29	p	compact
audi	a4	2.0	2008	4	manual(m6)	f	20	31	p	compact
audi	a4	2.0	2008	4	auto(av)	f	21	30	p	compact
audi	a4	2.8	1999	6	auto(l5)	f	16	26	p	compact
audi	a4	2.8	1999	6	manual(m5)	f	18	26	p	compact

注意 `head()` `tail()` 的用法和参数

head 和 tail 的用法

```
nrow(mpg); ## total number of rows
```

```
## [1] 234
```

```
kbl( head(mpg,  n=3), booktabs = T); ## 显示前几行数据
```

manufacturer	model	displ	year	cyl	trans	drv	cty	hwy	fl	class
audi	a4	1.8	1999	4	auto(l5)	f	18	29	p	compact
audi	a4	1.8	1999	4	manual(m5)	f	21	29	p	compact
audi	a4	2.0	2008	4	manual(m6)	f	20	31	p	compact

```
kbl( tail(mpg,  n=3), booktabs = T); ## 显示最后 3 行数据
```

manufacturer	model	displ	year	cyl	trans	drv	cty	hwy	fl	class
volkswagen	passat	2.8	1999	6	auto(l5)	f	16	26	p	midsize
volkswagen	passat	2.8	1999	6	manual(m5)	f	18	26	p	midsize
volkswagen	passat	3.6	2008	6	auto(s6)	f	17	26	p	midsize

data.frame 的组成与常用函数

组成

- 二维表格
- 由不同列组成；每列是一个 **vector**，不同列的数据类型可以不同，但一列只包括一种数据类型（int, num, chr ...）
- 各列的长度相同

常用 functions

- nrow( );
- ncol( );
- dim( );
- ...

structure of data.frame: str 函数

```
str( mpg );
```

```
## tibble [234 x 11] (S3: tbl_df/tbl/data.frame)
##  $ manufacturer: chr [1:234] "audi" "audi" "audi" "audi" ...
##  $ model       : chr [1:234] "a4" "a4" "a4" "a4" ...
##  $ displ       : num [1:234] 1.8 1.8 2 2 2.8 2.8 3.1 1.8 1.8 2 ...
##  $ year        : int [1:234] 1999 1999 2008 2008 1999 1999 2008 1999 1999 2008 ...
##  $ cyl         : int [1:234] 4 4 4 4 6 6 6 4 4 4 ...
##  $ trans       : chr [1:234] "auto(l5)" "manual(m5)" "manual(m6)" "auto(av)" ...
##  $ drv         : chr [1:234] "f" "f" "f" "f" ...
##  $ cty         : int [1:234] 18 21 20 21 16 18 18 18 16 20 ...
##  $ hwy         : int [1:234] 29 29 31 30 26 26 27 26 25 28 ...
##  $ fl          : chr [1:234] "p" "p" "p" "p" ...
##  $ class       : chr [1:234] "compact" "compact" "compact" "compact" ...
```

注：Tibble class 是 data.frame 的升级版本；本课程将二者混用，以 tibble 为主。用?mpg 命令查看 mpg 各列的意义

2.2 make a new data.frame

使用 data.frame 函数创建新的 data.frame:

```
##  data.frame()
( dat2 <-
  data.frame( data = sample( 1:100, 10 ),
        group = sample( LETTERS[1:3], 10, replace = TRUE),
        data2 = 0.1 )
);
```

```
##    data group data2
## 1    17     C   0.1
## 2    48     B   0.1
## 3    90     B   0.1
## 4    28     B   0.1
## 5    25     A   0.1
## 6    51     B   0.1
## 7   100     A   0.1
## 8    80     A   0.1
## 9    64     B   0.1
## 10   29     C   0.1
```

```
str(dat2);
```

```
## 'data.frame':    10 obs. of  3 variables:
##  $ data : int  17 48 90 28 25 51 100 80 64 29
##  $ group: chr  "C" "B" "B" "B" ...
```

2.3 how to add row(s)/col(s) to an existing data.frame

先创建“表头”，再填充

```
df2 <- data.frame( x = character(), y = integer(), z = double() , stringsAsFactors = FALSE );

##
df2 <- rbind( df2, data.frame( x = "a", y = 1L, z = 2.2 ) );
df2 <- rbind( df2, data.frame( x = "b", y = 2, z = 4.4 ) );

df2;
```

```
##   x y   z
## 1 a 1 2.2
## 2 b 2 4.4
```

注意

- 使用 rbind 函数
- 新的一行用 data.frame 定义，其“表头”需要与合并表相同

问题：

以下代码能顺利执行吗？

```
##  注意这里的 data.frame 会有多行 ...
df2 <- rbind( df2, data.frame( x = c("a","b","c"), y = 1L, z = 2.2 ) );
```

data.frame, add column

用 cbind 函数增加列：column bind

```
m <- cbind(1, 1:7) ; ## 产生两列数据 7 行数据 ..
( m <- cbind(m, 8:14) ); ## 增加一列 也有 7 行数据 ...
```

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

data.frame, add column, cont.

自行练习，回答代码中的问题：

```
## 1. 生成一个 10 行 2 列的 data.frame
df3 <- data.frame( data = 1:10, group = c("A","B") );

## 2. 增加一列，其长度是 1，可以吗？
cbind(df3, newcol = 1);

## 3. 增加一列，其长度是 10，可以吗？
cbind(df3, newcol = 1:10);

## 4. 增加一列，其长度是 2，可以吗？
cbind(df3, newcol = 1:2);

## 5. 增加一列，其长度是 3，可以吗？
cbind(df3, newcol = 1:3);
```

data.frame, 以列方式合并两个 data.frame

同样使用 cbind

```
df4 <- data.frame( data = 1:10, group = c("A","B") );
df5 <- data.frame( length = sample(1:100, 10), width = sample(1:100, 10) );

## --
head( cbind( df4, df5 ) );
```

```
##   data group length width
## 1    1     A     71    81
## 2    2     B     47    15
## 3    3     A      7    70
## 4    4     B      5    55
## 5    5     A     33    19
## 6    6     B     82    98
```

```
## 如果 一个 df 的行数少于另一处怎么办？
df6 <- data.frame( length = sample(1:100, 5), width = sample(1:100, 5) );
head( cbind( df4, df6 ) );
```

```
##   data group length width
## 1    1     A     90    92
## 2    2     B     81    47
## 3    3     A     73    27
## 4    4     B     44    10
## 5    5     A     78    90
```

2.4 how to manipulate a data.frame

自行尝试以下操作

```
## 取行:
df4[ 1:2, ];

## 取列
df4[, 2]

## 取行, 重新排序
df4[c(2,3,1), ]

## 取列, 重新排序
df4[, c(2,1)]

## 替换一行:
df4[1, ] <- data.frame( data = 100, group = "A" );

## 替换一列:
df4[, "data"] <- sample( 1:100, 5 );
```

tibble, outline

5. how to make a tibble
6. how to add row(s)/col(s) to an existing tibble how to combine two tibble
7. how to manipulate a tibble

2.5 make a new tibble

tibble 相关功能由 tibble 或 tidiverse 包提供

```
library(tibble); ## 或 library(tidiverse);
## 用 tibble 函数创建，用法和 data.frame() 相似
( dat <-
  tibble( data = sample( 1:100, 10 ),
          group = sample( LETTERS[1:3], 10, replace = TRUE),
          data2 = 0.1 )
);
```

```
## # A tibble: 10 x 3
##     data group data2
##    <int> <chr> <dbl>
##  1    50 A       0.1
##  2    84 C       0.1
##  3    37 B       0.1
##  4    60 C       0.1
##  5    45 A       0.1
##  6    76 A       0.1
##  7    75 A       0.1
##  8     9 B       0.1
##  9    11 A       0.1
## 10    13 A       0.1
```

- 注意每列的数据类型
- 长度不足时，比如 **data2** 列，会循环使用

str(dat)

查看得到的数据结构

```
str(dat);
```

```
## tibble [10 x 3] (S3: tbl_df/tbl/data.frame)
##  $ data : int [1:10] 50 84 37 60 45 76 75 9 11 13
##  $ group: chr [1:10] "A" "C" "B" "C" ...
##  $ data2: num [1:10] 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1
```

创建 tibble 的另一种方式 (by row)

```
## 另一种方式创建 tibble
tribble(
  ~x, ~y,  ~z,
  "a", 2,  3.6,
  "b", 1,  8.5
)
```

```
## # A tibble: 2 x 3
##   x         y     z
##   <chr> <dbl> <dbl>
## 1 a         2   3.6
## 2 b         1   8.5
```

2.6 how to add row(s)/col(s) to an existing tibble

```
## 新 tibble, with defined columns ... 创建表头
tb <- tibble( x = character(), y = integer(), z = double() );
dim(tb);
```

```
## [1] 0 3
```

```
## 增加行 ...
tb <- add_row( tb, x = "a", y = 2, z = 3.6  );
tb <- add_row( tb, x = "b", y = 1, z = 8.5  );

## 显示
tb;
```

```
## # A tibble: 2 x 3
##   x         y     z
##   <chr> <dbl> <dbl>
## 1 a         2   3.6
## 2 b         1   8.5
```

tibble, add_row 插入时指定行号

```
## 生成一个 tibble
df <- tibble(x = 1:3, y = 3:1);

# 在第二行之前插入
df <- add_row(df, x = 4, y = 0, .before = 2);

df;
```

```
## # A tibble: 4 x 2
##       x     y
##   <dbl> <dbl>
## 1     1     3
## 2     4     0
## 3     2     2
## 4     3     1
```

tibble, add_row 插入多行, 插入另一个 tibble??

```
## 插入多行
df <- add_row(df, x = 4:5, y = 0:-1);

## 插入另一个 tibble (与另一个 tibble 合并) ???
df2 <- tibble( x = as.double(200:202), y = as.double(1000:1002) );
df3 <- add_row( df, df2 ); ## 可以运行 ...
```

tibble, 合并多个 tibble

```
df3 <- bind_rows( df, df2 );
df3;
```

```
## # A tibble: 9 x 2
##       x     y
##   <dbl> <dbl>
## 1     1     3
## 2     4     0
## 3     2     2
## 4     3     1
## 5     4     0
## 6     5    -1
## 7   200  1000
## 8   201  1001
## 9   202  1002
```

tibble, add column

```
tb3 <- tribble(
  ~x, ~y,  ~z,
  "a", 2,  3.6,
  "b", 1,  8.5
);

tb3 <- add_column( tb3, a = 98 ); ## recycle ...
tb3 <- add_column( tb3, b = LETTERS[1:2], c = c("CHEN", "WANG") );
tb3;
```

```
## # A tibble: 2 x 6
##   x         y     z     a b     c
##   <chr> <dbl> <dbl> <dbl> <chr> <chr>
## 1 a         2   3.6    98 A     CHEN
## 2 b         1   8.5    98 B     WANG
```

tibble, 按列合并两个 tibble?

练习:

1. 尝试用 `add_column` 合并两个 tibble
2. 使用 `bind_cols` 合并两个 tibble

2.7 how to manipulate a tibble

自行练习以熟悉以下操作:

```
## 取得行
tb3[c(1,2), ];

## 取得列, 按顺序取列
tb3[, c("z", "y")];

## 替换列
tb3[["z"]] <- c(4.6, 5.5);

## 替换行
tb3[ 1, ] <- tibble( x = "d", y = 20, z = 46, a = 10, b = "C", c = "LILI" );
```

2.8 tibble 与 data.frame 之间相互转换

```
library(tibble)
head( as_tibble(iris) );
```

```
## # A tibble: 6 x 5
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
##          <dbl>       <dbl>        <dbl>       <dbl> <fct>
## 1          5.1         3.5          1.4         0.2 setosa
## 2          4.9         3            1.4         0.2 setosa
## 3          4.7         3.2          1.3         0.2 setosa
## 4          4.6         3.1          1.5         0.2 setosa
## 5          5           3.6          1.4         0.2 setosa
## 6          5.4         3.9          1.7         0.4 setosa
```

note: iris data set gives the measurements in centimeters of the variables sepal length and width and petal length and width, respectively, for 50 flowers from each of 3 species of iris (鸢尾属植物). The species are Iris setosa, versicolor, and virginica.

tibble to dataframe

```
library(tibble)
as.data.frame(  head( as_tibble(iris) ) );
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 1          5.1         3.5          1.4         0.2  setosa
## 2          4.9         3.0          1.4         0.2  setosa
## 3          4.7         3.2          1.3         0.2  setosa
## 4          4.6         3.1          1.5         0.2  setosa
## 5          5.0         3.6          1.4         0.2  setosa
## 6          5.4         3.9          1.7         0.4  setosa
```

2.9 differences between tibble and data.frame

tibble evaluates columns sequentially

```
rm(x,y); ## 删除可能存在的 x , y
tibble(x = 1:5, y = x ^ 2); ## 可以用 tibble 这样做
```

```
## # A tibble: 5 x 2
##       x     y
##   <int> <dbl>
## 1     1     1
## 2     2     4
## 3     3     9
## 4     4    16
## 5     5    25
```

练习:

```
data.frame(x = 1:5, y = x ^ 2); ## 但 data.frame 不行
```

```
## Error in data.frame(x = 1:5, y = x^2): object 'x' not found
```

differences between tibble and data.frame, cont.

data.frame 在取 subset 操作时，会造成困扰

```
df1 <- data.frame(x = 1:3, y = 3:1);
class(df1[, 1:2]);
```

```
## [1] "data.frame"
```

```
## subset 操作 : 取一列, 期待得到一个 data.frame ()
class(df1[, 1]); ## 结果得到一个 vector ...
```

```
## [1] "integer"
```

```
## 而 tibble 则不会
df2 <- tibble(x = 1:3, y = 3:1);
class(df2[, 1]); ## 永远都是 tibble
```

```
## [1] "tbl_df"     "tbl"        "data.frame"
```

differences between tibble and data.frame, cont.

tibble 可以进行可控的数据类型转换：

```
class(df2[[1]]); ## 取一列，转换为 vector
```

```
## [1] "integer"
```

```
class(df2$x); ## 用 [[]] 或 $ 都可以哦
```

```
## [1] "integer"
```

differences between tibble and data.frame, cont.

recycling

```
data.frame(a = 1:6, b = LETTERS[1:2]); ##  data.frame 可以!!!
```

```
##   a b
## 1 1 A
## 2 2 B
## 3 3 A
## 4 4 B
## 5 5 A
## 6 6 B
```

```
tibble(a = 1:6, b = LETTERS[1:2]); ## 但 tibble 不行!!!
```

```
## Error:
## ! Tibble columns must have compatible sizes.
## * Size 6: Existing data.
## * Size 2: Column `b`.
## i Only values of size one are recycled.
```

注意 tibble 的 recycling 仅限于长度为 1 或等长；而 data.frame 则为整除即可。

practises for recycling

```
tibble(a = 1, b = 1:3);
```

```
## # A tibble: 3 x 2
##       a     b
##   <dbl> <int>
## 1     1     1
## 2     1     2
## 3     1     3
```

```
tibble(a = 1:3, b = 1);
```

```
## # A tibble: 3 x 2
##       a     b
##   <int> <dbl>
## 1     1     1
## 2     2     1
## 3     3     1
```

```
tibble(a = 1:3, c = 1:2);
```

```
## Error:
## ! Tibble columns must have compatible sizes.
## * Size 3: Existing data.
## * Size 2: Column `c`.
## i Only values of size one are recycled.
```

differences between tibble and data.frame, cont.

data.frame will do partial matching

```
df <- data.frame(abc = 1)
df$ab; ## unwanted result ...
```

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

```
## -- but tibble will never do it;
df2 <- tibble(abc = 1)
df2$a; ## produce a warning and return NULL
```

```
## Warning: Unknown or uninitialised column: `a`.

## NULL
```

2.10 data.frame 和 tibble 的高级使用技巧

attach 和 detach

```
head( iris, n = 3 );
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 1          5.1         3.5          1.4         0.2  setosa
## 2          4.9         3.0          1.4         0.2  setosa
## 3          4.7         3.2          1.3         0.2  setosa
```

```
head(  iris$Sepal.Length , n = 10 ); ## 用 $ 操作符取得一列 ...
```

```
##  [1] 5.1 4.9 4.7 4.6 5.0 5.4 4.6 5.0 4.4 4.9
```

```
attach( iris );
head(  Sepal.Length , n = 10 ); ## 直接用列名获取数据;
```

```
##  [1] 5.1 4.9 4.7 4.6 5.0 5.4 4.6 5.0 4.4 4.9
```

```
detach(iris); ## 取消 attach 操作 --
```

with 函数

```
with( iris, head(  Sepal.Length, n  = 10 )); ## 用 with 也可以实现
```

```
##  [1] 5.1 4.9 4.7 4.6 5.0 5.4 4.6 5.0 4.4 4.9
```

within 函数

也可以用 within 对多列数据进行修改

```
head( airquality , n = 3 );
```

```
##   Ozone Solar.R Wind Temp Month Day
## 1    41     190  7.4   67     5   1
## 2    36     118  8.0   72     5   2
## 3    12     149 12.6   74     5   3
```

```
aq <- within(airquality, {      # Notice that multiple vars can be changed
    lOzone <- log(Ozone)
    Month <- factor(month.abb[Month])
    cTemp <- round((Temp - 32) * 5/9, 1) # From Fahrenheit to Celsius
    S.cT <- Solar.R / cTemp  # using the newly created variable
    rm(Day, Temp) ## 删除特定列 ...
});

head(aq, n = 3 );
```

```
##   Ozone Solar.R Wind Month     S.cT cTemp   lOzone
## 1    41     190  7.4   May 9.793814  19.4 3.713572
## 2    36     118  8.0   May 5.315315  22.2 3.583519
## 3    12     149 12.6   May 6.394850  23.3 2.484907
```

section 3: file IO: read a file into tibble & write tibble to a file

read from files

使用 functions from the `readr` package

```
## readr is part of tidyverse
library(tidyverse); ## or alternatively
library(readr);
```

available functions

- read_csv(): comma separated (CSV) files
- read_tsv(): tab separated files
- read_delim(): general delimited files
- read_fwf(): fixed width files
- read_table(): tabular files where columns are separated by white-space.
- read_log(): web log files

read a file into tibble

```
myiris <- read_csv("data/talk03/iris.csv");
```

```
## Rows: 150 Columns: 5
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (1): Species
## dbl (4): Sepal.Length, Sepal.Width, Petal.Length, Petal.Width
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

注意输出的 columns 定义

read with predifined column types

```
myiris2 <-  read_csv("data/talk03/iris.csv", col_types = cols(
  Sepal.Length = col_double(),
  Sepal.Width = col_double(),
  Petal.Length = col_double(),
  Petal.Width = col_double(),
  Species = col_character()
));
```

how to read from other formats??

try the following packages for other formats

- **haven** - SPSS, Stata, and SAS files
- **readxl** - excel files (.xls and .xlsx)
- **DBI** - databases
- **jsonlite** - json
- **xml2** - XML
- **httr** - Web APIs
- **rvest** - HTML (Web Scraping)

write to files

use the following functions to write object(s) to external files

- Comma delimited file: **write_csv**(x, path, na = "NA", append = FALSE, col_names = !append)
- File with arbitrary delimiter: **write_delim**(x, path, delim = " ", na ="NA", append = FALSE, col_names = !append)
- CSV for excel: **write_excel_csv**(x, path, na = "NA", append = FALSE, col_names = !append)
- String to file: **write_file**(x, path, append = FALSE)
- String vector to file, one element per line: **write_lines**(x,path, na = "NA", append = FALSE)
- Object to RDS file: **write_rds**(x, path, compress = c("none", "gz", "bz2", "xz"), ...)
- Tab delimited files: **write_tsv**(x, path, na = "NA", append = FALSE, col_names = !append)

练习

```
## write iris to outfiles of various formats
write_csv( iris, "iris.csv" );
write_tsv(iris, "iris.tsv", quote_escape = "none");
```

check `readr` cheatsheet (please Google).

section 4: 练习 & 作业

练习 & 作业

- Exercises and homework 目录下 talk03-homework.Rmd 文件;
- 完成时间：见钉群的要求

小结

今次提要

1. data.frame, tibble
2. 定义、区别、转化
3. read files from harddrive (IO)

下次预告

- factor ： R 另一个超级重要且难以上手的概念
- 基础和进阶绘图（配合 factor 讲解）

important

- all codes are available at Github:
  `https://github.com/evolgeniusteam/R-for-bioinformatics`