

# R language basics, part 2

## HUST Bioinformatics course series

Wei-Hua Chen (CC BY-NC 4.0)

02 三月, 2020

# section 1: TOC

# 前情提要

## vector & matrix:

- declaration
- manipulation
- arithmetic
- transposition

## vectorization

- every is a vector!!
- vectorization versys loop (will be explained later)
- advantages using vectorization (<https://www.noamross.net/blog/2014/4/16/vectorization-in-r--why.html>)

# 今次预报

- ① data.frame, tibble
- ② read files from harddrive (IO)
- ③ factors
- ④ exercises

## section 2: data.frame and tibble

# what is a data.frame???

眼见为实：

```
library(tidyverse); ## 装入包
knitr::kable( head(mpg) ); ## 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
```

```
knitr::kable( head(mpg, n=3) ); ## 显示前 3 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

```
knitr::kable( tail(mpg, n=3) ); ## 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, cont.

## 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( mpg );
```

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

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

# make a new tibble

```
## 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     19 C      0.1
## 2     68 C      0.1
## 3     49 B      0.1
## 4     18 C      0.1
## 5     86 B      0.1
## 6     82 C      0.1
## 7     15 B      0.1
## 8     25 C      0.1
## 9     87 C      0.1
## 10    80 C      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
- `sample()` 函数的用法

# str( dat )

## 查看得到的数据结构

```
str(dat);
```

```
## Classes 'tbl_df', 'tbl' and 'data.frame':    10 obs. of  3 variables:
## $ data : int  19 68 49 18 86 82 15 25 87 80
## $ group: chr   "C" "C" "B" "C" ...
## $ data2: num   0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1
```

# make a new tibble row-by-row

```
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
```

# make a new tibble then add rows

```
## 新 tibble, with defined columns ...
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 的其它参数

```
df <- tibble(x = 1:3, y = 3:1);  
  
# You can specify where to add the new rows  
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 );
```

## New rows in `add\_row()` must use columns that already exist:

## \* Can't find column `df2` in `.data`.

# 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

# make a new 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      56      A  0.1
## 2      78      A  0.1
## 3      81      B  0.1
## 4      11      C  0.1
## 5       2      A  0.1
## 6      47      A  0.1
## 7      93      A  0.1
## 8      45      B  0.1
## 9      67      B  0.1
## 10     89      C  0.1
```

```
str(dat2);
```

```
## 'data.frame':    10 obs. of  3 variables:
## $ data : int  56 78 81 11 2 47 93 45 67 89
## $ group: Factor w/ 3 levels "A","B","C": 1 1 2 3 1 1 1 2 2 3
## $ data2: num  0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1
```

# make a new data.frame & add data by row

```
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
```

# data.frame, add column

```
m <- cbind(1, 1:7);  
( m <- cbind(m, 8:14) );
```

```
##      [,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, 以列方式合并两个 data.frame

```
(dg <- diag(3));
```

```
##      [,1] [,2] [,3]
## [1,]    1    0    0
## [2,]    0    1    0
## [3,]    0    0    1
```

```
## 报错 ??
cbind( m, dg );
```

**## Error in cbind(m, dg): 矩阵的行数必需相符(见arg2)**

```
##
cbind(1:7, dg);
```

**## Warning in cbind(1:7, dg): number of rows of result is not a multiple of  
## vector length (arg 1)**

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

```
##
cbind(dg, 1:7);
```

# 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



# 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): 找不到对象'x'
```

# 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：取一行，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!!!
```

```
## Tibble columns must have consistent lengths, only values of length one are recycled:
## * Length 2: Column `b`
## * Length 6: Column `a`
```

# 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);
```

```
## Tibble columns must have consistent lengths, only values of length one are recycled:
## * Length 2: Column `c`
## * Length 3: Column `a`
```

# 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
```

# attach and 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");
```

```
## Parsed with column specification:
## cols(
##   Sepal.Length = col_double(),
##   Sepal.Width = col_double(),
##   Petal.Length = col_double(),
##   Petal.Width = col_double(),
##   Species = col_character()
## )
```

注意输出的 columns 定义

## read with predefined 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:

<https://rawgit.com/rstudio/cheatsheets/master/data-import.pdf>



## section 4: 练习与作业

# 练习

## data frame 练习

- 1 `https://www.r-exercises.com/2016/01/04/data-frame-exercises/`
- 2 `https://www.r-exercises.com/2016/11/28/data-frame-exercises-vol-2/`

## tibble 练习

- 1 `https://r4ds.had.co.nz/tibbles.html#exercises-18`
- 2 `http://uc-r.github.io/tibbles`
- 3 more to read: `http://www.sthda.com/english/wiki/tibble-data-format-in-r-best-and-modern-way-to-work-with-`

## 其它练习

- 1 file IO
- 2 with and within

# 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>