

# Note for R Lec 4-5

Luo Beier

## 目录

<b>1 R 语言</b>	<b>2</b>
1.1 Pipe operator . . . . .	2
1.2 基本数据管理 . . . . .	3
1.2.1 mutate() . . . . .	3
1.2.2 选取列 select() . . . . .	5
1.2.3 修改列名 . . . . .	8
1.2.4 按行选取 filter() . . . . .	9
1.2.5 排序 arrange() . . . . .	10
1.2.6 合并数据 . . . . .	11
1.2.7 缺失值 . . . . .	12
1.2.8 数据规整 . . . . .	13
1.3 From data set to a random variables . . . . .	18
1.3.1 summary() . . . . .	18
1.3.2 Probability models . . . . .	20
1.4 Generate random variables by the build-in functions . . . . .	26
1.4.1 Common probability distributions . . . . .	26
1.4.2 Generate normal random variables . . . . .	28
1.4.3 Generate multivariate normal random variables . . . . .	28

# 1 R 语言

首先，我们先介绍一个新的传递数据的方法：**Pipe operator**

## 1.1 Pipe operator

A key package: **tidyverse**

Suppose that we want to find the following summation:

$$\sqrt{\sum_{i=-10}^{10} |i|}.$$

Base-R, we can:

```
sqrt(sum(abs(-10:10)))
```

```
## [1] 10.48809
```

We can use pipe operator to deal with multiple functions like this:

```
library(tidyverse)
```

```
## -- Attaching packages ----- tidyverse 1.3.2 --
## v ggplot2 3.3.6      v purrr   0.3.4
## v tibble  3.1.8      v dplyr  1.0.10
## v tidyr   1.2.1      v stringr 1.4.1
## v readr   2.1.2      v forcats 0.5.2
## -- Conflicts ----- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
-10:10 %>%
  abs() %>%
  sum() %>%
  sqrt()
```

```
## [1] 10.48809
```

### More logical!

When you have multiple arguments in a function:

```
matrix(1:10, nrow = 2, byrow = TRUE)
```

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

```
1:10 %>%  
matrix(nrow = 2, byrow = TRUE)
```

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

## 1.2 基本数据管理

```
library(tidyverse)  
library(palmerpenguins)
```

### 1.2.1 mutate()

我们想再创建一个新的变量:

```
df <- penguins  
attach(df)  
df$bill_sum <- bill_length_mm + bill_depth_mm  
detach(df)
```

在 tidyverse 包下, 可以使用 mutate() 函数:

```
library(tidyverse)
mutate(.data = df, bill_sum = bill_length_mm + bill_depth_mm)
```

```
## # A tibble: 344 x 9
##   species island  bill_length_mm bill_d~1 flipp~2 body_~3 sex   year bill_~4
##   <fct>   <fct>          <dbl>    <dbl>   <int>   <int> <fct> <int>   <dbl>
## 1 Adelie  Torgersen          39.1     18.7    181    3750 male   2007    57.8
## 2 Adelie  Torgersen          39.5     17.4    186    3800 fema~ 2007    56.9
## 3 Adelie  Torgersen          40.3      18     195    3250 fema~ 2007    58.3
## 4 Adelie  Torgersen          NA      NA      NA      NA <NA>   2007     NA
## 5 Adelie  Torgersen          36.7     19.3    193    3450 fema~ 2007     56
## 6 Adelie  Torgersen          39.3     20.6    190    3650 male   2007    59.9
## 7 Adelie  Torgersen          38.9     17.8    181    3625 fema~ 2007    56.7
## 8 Adelie  Torgersen          39.2     19.6    195    4675 male   2007    58.8
## 9 Adelie  Torgersen          34.1     18.1    193    3475 <NA>   2007    52.2
## 10 Adelie Torgersen          42      20.2    190    4250 <NA>   2007    62.2
## # ... with 334 more rows, and abbreviated variable names 1: bill_depth_mm,
## #   2: flipper_length_mm, 3: body_mass_g, 4: bill_sum
```

```
## or,
mutate(df, bill_sum = bill_length_mm + bill_depth_mm)
```

```
## # A tibble: 344 x 9
##   species island  bill_length_mm bill_d~1 flipp~2 body_~3 sex   year bill_~4
##   <fct>   <fct>          <dbl>    <dbl>   <int>   <int> <fct> <int>   <dbl>
## 1 Adelie  Torgersen          39.1     18.7    181    3750 male   2007    57.8
## 2 Adelie  Torgersen          39.5     17.4    186    3800 fema~ 2007    56.9
## 3 Adelie  Torgersen          40.3      18     195    3250 fema~ 2007    58.3
## 4 Adelie  Torgersen          NA      NA      NA      NA <NA>   2007     NA
## 5 Adelie  Torgersen          36.7     19.3    193    3450 fema~ 2007     56
## 6 Adelie  Torgersen          39.3     20.6    190    3650 male   2007    59.9
## 7 Adelie  Torgersen          38.9     17.8    181    3625 fema~ 2007    56.7
## 8 Adelie  Torgersen          39.2     19.6    195    4675 male   2007    58.8
```

```
## 9 Adelie Torgersen      34.1      18.1      193      3475 <NA>      2007      52.2
## 10 Adelie Torgersen      42        20.2      190      4250 <NA>      2007      62.2
## # ... with 334 more rows, and abbreviated variable names 1: bill_depth_mm,
## # 2: flipper_length_mm, 3: body_mass_g, 4: bill_sum
```

```
## or, using pipe operator:
```

```
df %>%
```

```
mutate(bill_sum = bill_length_mm + bill_depth_mm)
```

```
## # A tibble: 344 x 9
```

```
##   species island  bill_length_mm bill_d~1 flipp~2 body_~3 sex   year bill_~4
##   <fct>   <fct>          <dbl>    <dbl>    <int>    <int> <fct> <int>    <dbl>
## 1 Adelie Torgersen      39.1      18.7     181     3750 male   2007     57.8
## 2 Adelie Torgersen      39.5      17.4     186     3800 fema~  2007     56.9
## 3 Adelie Torgersen      40.3       18     195     3250 fema~  2007     58.3
## 4 Adelie Torgersen      NA        NA        NA        NA <NA>   2007     NA
## 5 Adelie Torgersen      36.7      19.3     193     3450 fema~  2007     56
## 6 Adelie Torgersen      39.3      20.6     190     3650 male   2007     59.9
## 7 Adelie Torgersen      38.9      17.8     181     3625 fema~  2007     56.7
## 8 Adelie Torgersen      39.2      19.6     195     4675 male   2007     58.8
## 9 Adelie Torgersen      34.1      18.1     193     3475 <NA>   2007     52.2
## 10 Adelie Torgersen      42        20.2     190     4250 <NA>   2007     62.2
## # ... with 334 more rows, and abbreviated variable names 1: bill_depth_mm,
## # 2: flipper_length_mm, 3: body_mass_g, 4: bill_sum
```

### 1.2.2 选取列 select()

按下列方法进行选取:

- 按名称选取

两种操作:

```
df <- penguins # 初始化 df 变量
df %>% select(bill_length_mm, bill_depth_mm)
```

```
## # A tibble: 344 x 2
##   bill_length_mm bill_depth_mm
##           <dbl>         <dbl>
## 1           39.1           18.7
## 2           39.5           17.4
## 3           40.3            18
## 4            NA            NA
## 5           36.7           19.3
## 6           39.3           20.6
## 7           38.9           17.8
## 8           39.2           19.6
## 9           34.1           18.1
## 10          42            20.2
## # ... with 334 more rows
```

```
df %>% select(-bill_length_mm, -bill_depth_mm) # 按名称删除不需要的列
```

```
## # A tibble: 344 x 6
##   species island flipper_length_mm body_mass_g sex   year
##   <fct>   <fct>           <int>         <int> <fct> <int>
## 1 Adelie Torgersen           181         3750 male   2007
## 2 Adelie Torgersen           186         3800 female 2007
## 3 Adelie Torgersen           195         3250 female 2007
## 4 Adelie Torgersen            NA            NA <NA>   2007
## 5 Adelie Torgersen           193         3450 female 2007
## 6 Adelie Torgersen           190         3650 male   2007
## 7 Adelie Torgersen           181         3625 female 2007
## 8 Adelie Torgersen           195         4675 male   2007
## 9 Adelie Torgersen           193         3475 <NA>   2007
## 10 Adelie Torgersen           190         4250 <NA>   2007
```

```
## # ... with 334 more rows
```

- 按名称所含字符选取

As follows:

#### 名称选取

如果选取的列很多，我们也可以先观察其命名特征，用特定的函数进行选取。比如，在企鹅数据集中，类型为 `double` 的变量名都是以 `bill` 开始的，所以我们可以

```
df %>% select(starts_with("bill"))
```

其他选择函数：

函数	作用
<code>starts_with()</code>	以某前缀开头
<code>ends_with()</code>	以某后缀结尾
<code>contains()</code>	包含某字符或字符串
<code>matches()</code>	匹配正则表达式
<code>num_range()</code>	匹配某数值范围

- 按数值类型选取

```
df %>% select(where(is.numeric))
```

```
## # A tibble: 344 x 5
```

```
##   bill_length_mm bill_depth_mm flipper_length_mm body_mass_g year
##           <dbl>         <dbl>           <int>      <int> <int>
## 1           39.1           18.7             181        3750  2007
## 2           39.5           17.4             186        3800  2007
## 3           40.3            18             195        3250  2007
## 4            NA            NA              NA         NA  2007
## 5           36.7           19.3             193        3450  2007
## 6           39.3           20.6             190        3650  2007
## 7           38.9           17.8             181        3625  2007
## 8           39.2           19.6             195        4675  2007
```

```
## 9          34.1          18.1          193          3475  2007
## 10         42           20.2          190          4250  2007
## # ... with 334 more rows
```

```
df %>% select(where(is.double))
```

```
## # A tibble: 344 x 2
##   bill_length_mm bill_depth_mm
##           <dbl>         <dbl>
## 1           39.1           18.7
## 2           39.5           17.4
## 3           40.3           18
## 4            NA            NA
## 5           36.7           19.3
## 6           39.3           20.6
## 7           38.9           17.8
## 8           39.2           19.6
## 9           34.1           18.1
## 10          42            20.2
## # ... with 334 more rows
```

- 混合选取

```
## 逻辑并
select(df, 条件一 & 条件二)
## 逻辑或
select(df, 条件一 | 条件二)
## 逻辑非
select(df, !条件一)
```

### 1.2.3 修改列名



```
df %>%
  rename(Bill.Length = bill_length_mm,
         Bill.Depth = bill_depth_mm,
         Flipper.Length = flipper_length_mm,
         Body.Mass = body_mass_g) # 等号前是新名字，等号后面是老名字
```

```
## # A tibble: 344 x 8
##   species island   Bill.Length Bill.Depth Flipper.Length Body.Mass sex   year
##   <fct>   <fct>         <dbl>     <dbl>         <int>     <int> <fct> <int>
## 1 Adelie  Torgersen         39.1      18.7           181       3750 male   2007
## 2 Adelie  Torgersen         39.5      17.4           186       3800 fema~ 2007
## 3 Adelie  Torgersen         40.3       18            195       3250 fema~ 2007
## 4 Adelie  Torgersen          NA        NA            NA         NA <NA>   2007
## 5 Adelie  Torgersen         36.7      19.3           193       3450 fema~ 2007
## 6 Adelie  Torgersen         39.3      20.6           190       3650 male   2007
## 7 Adelie  Torgersen         38.9      17.8           181       3625 fema~ 2007
## 8 Adelie  Torgersen         39.2      19.6           195       4675 male   2007
## 9 Adelie  Torgersen         34.1      18.1           193       3475 <NA>   2007
## 10 Adelie Torgersen         42       20.2           190       4250 <NA>   2007
## # ... with 334 more rows
```

#### 1.2.4 按行选取 filter()

比如，我们想要选择 species 为"Adelie" 的这些样本点：

```
df %>% filter(species == "Adelie")
```

```
## # A tibble: 152 x 8
##   species island   bill_length_mm bill_depth_mm flipper_~1 body_~2 sex   year
##   <fct>   <fct>         <dbl>     <dbl>         <int>     <int> <fct> <int>
## 1 Adelie  Torgersen         39.1      18.7           181       3750 male   2007
## 2 Adelie  Torgersen         39.5      17.4           186       3800 fema~ 2007
## 3 Adelie  Torgersen         40.3       18            195       3250 fema~ 2007
```

```
## 4 Adelie Torgersen      NA      NA      NA      NA <NA> 2007
## 5 Adelie Torgersen     36.7     19.3    193    3450 fema~ 2007
## 6 Adelie Torgersen     39.3     20.6    190    3650 male  2007
## 7 Adelie Torgersen     38.9     17.8    181    3625 fema~ 2007
## 8 Adelie Torgersen     39.2     19.6    195    4675 male  2007
## 9 Adelie Torgersen     34.1     18.1    193    3475 <NA>  2007
## 10 Adelie Torgersen     42      20.2    190    4250 <NA>  2007
## # ... with 142 more rows, and abbreviated variable names 1: flipper_length_mm,
## # 2: body_mass_g
```

进一步再选取 bill\_length\_mm 大于 40 的:

```
df %>% filter(species == "Adelie",
  bill_length_mm > 40)
```

```
## # A tibble: 51 x 8
##   species island  bill_length_mm bill_depth_mm flipper_~1 body_~2 sex  year
##   <fct>   <fct>      <dbl>         <dbl>      <int>    <int> <fct> <int>
## 1 Adelie Torgersen    40.3          18        195     3250 fema~ 2007
## 2 Adelie Torgersen    42           20.2       190     4250 <NA>  2007
## 3 Adelie Torgersen    41.1          17.6       182     3200 fema~ 2007
## 4 Adelie Torgersen    42.5          20.7       197     4500 male  2007
## 5 Adelie Torgersen    46           21.5       194     4200 male  2007
## 6 Adelie Biscoe     40.6          18.6       183     3550 male  2007
## 7 Adelie Biscoe     40.5          17.9       187     3200 fema~ 2007
## 8 Adelie Biscoe     40.5          18.9       180     3950 male  2007
## 9 Adelie Dream      40.9          18.9       184     3900 male  2007
## 10 Adelie Dream      42.2          18.5       180     3550 fema~ 2007
## # ... with 41 more rows, and abbreviated variable names 1: flipper_length_mm,
## # 2: body_mass_g
```

### 1.2.5 排序 arrange()

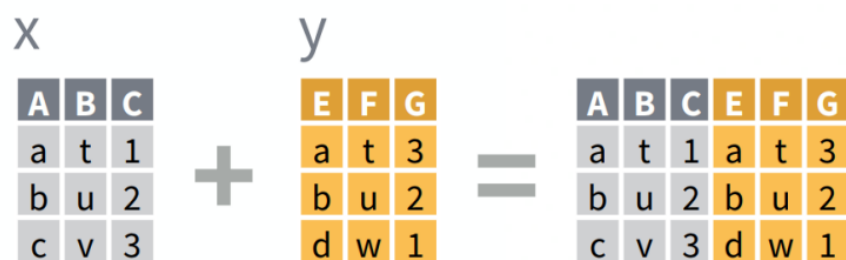
arrange(df, variable) (升序), 或者 arrange(df, -variable) (降序)。

也可以对多个变量依次排序。比如先对变量一排序，再对变量二排序，其公式为：

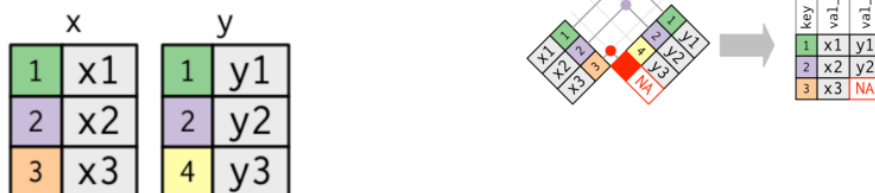
```
arrange(df, var1, var2)
```

### 1.2.6 合并数据

- `bind_cols(df1, df2, ...)`



- 连接数据 `_join()` 函数族
  - 左联结 `left_join()`;
  - 还有右联结 `right_join()`, `inner_join` 和 `full_join` 等。



## 其他的数据处理函数

### Data transformation with dplyr : CHEAT SHEET

**dplyr** functions work with pipes and expect **tidy data**. In tidy data:

- Each **variable** is in its own **column**
- Each **observation**, or **case**, is in its own **row**
- $x \% \rightarrow \% f(y)$  becomes  $f(x, y)$

**Summarise Cases**

Apply **summary functions** to columns to create a new table of summary statistics. Summary functions take vectors as input and return one value (see back).

**summary function**

```
summarise(data, ...)
# Compute table of summaries.
summarise(mtcars, avg = mean(mpg))
```

**count()** Count number of rows in each group defined by the variables in ... Also **tally()**.

```
count(data, ..., wt = NULL, sort = FALSE, name = NULL)
count(mtcars, cyl)
```

**Group Cases**

Use **group\_by()** to group data into individual rows. **dplyr** functions will compute results for each row. Also apply functions to list-columns. See tidy cheat sheet for list-column workflow.

```
group_by(data, ..., add = FALSE, drop = TRUE) to create a "grouped" copy of a table grouped by columns in ... dplyr functions will manipulate each "group" separately and combine the results.
group_by(mtcars, cyl) %>% summarise(avg = mean(mpg))
```

Use **rowwise()** to group data into individual rows. **dplyr** functions will compute results for each row. Also apply functions to list-columns. See tidy cheat sheet for list-column workflow.

```
rowwise(data, ...)
# Compute results for each row.
rowwise(mtcars) %>% summarise(lm_count = length(lm))
```

**ungroup()** Returns ungrouped copy of table.

```
ungroup(mtcars)
```

**Manipulate Cases**

**EXTRACT CASES**

Row functions return a subset of rows as a new table.

- filter()** `filter(data, ..., preserve = FALSE)` Extract rows that meet logical criteria. `filter(mtcars, mpg > 20)`
- distinct()** `distinct(data, ..., keep_all = FALSE)` Remove rows with duplicate values. `distinct(mtcars, gear)`
- slice()** `slice(data, ..., preserve = FALSE)` Select rows by position. `slice(mtcars, 10:15)`
- slice\_sample()** `slice_sample(data, ..., n, prop, weight_by = NULL, replace = FALSE)` Randomly select rows. Use `n` to select a number of rows and `prop` to select a fraction of rows. `slice_sample(mtcars, n = 5, replace = TRUE)`
- slice\_min()** `slice_min(data, order_by, ..., n, prop, with_ties = TRUE)` and **slice\_max()** Select rows with the lowest and highest values. `slice_min(mtcars, mpg, prop = 0.25)`
- slice\_head()** `slice_head(data, ..., n, prop)` and **slice\_tail()** Select the first or last rows. `slice_head(mtcars, n = 5)`

**Logical and boolean operators to use with filter()**

```
== < <= is.na() %in% | xor()
!= > >= is.na() ! &
```

See **Base:Logic** and **?Comparison** for help.

**ARRANGE CASES**

**arrange()** `arrange(data, ..., by, group = FALSE)` Order rows by values of a column or columns (low to high), use with **desc()** to order from high to low. `arrange(mtcars, mpg)`

**ADD CASES**

**add\_row()** `add_row(data, ..., before = NULL, after = NULL)` Add one or more rows to a table. `add_row(mtcars, speed = 1, dist = 1)`

**Manipulate Variables**

**EXTRACT VARIABLES**

Column functions return a set of columns as a new vector or table.

- pull()** `pull(data, var = 1, name = NULL, ...)` Extract column values as a vector, by name or index. `pull(mtcars, wt)`
- select()** `select(data, ...)` Extract columns as a table. `select(mtcars, mpg, wt)`
- relocate()** `relocate(data, ..., before = NULL, after = NULL)` Move columns to new position. `relocate(mtcars, mpg, cyl, after = last_col())`

**Use these helpers with select() and across()**

```
contains(match) num_range(prefix, range) # e.g. mpg:cyl
ends_with(match) all_of(x) any_of(x, ..., vars) ~ e.g. -gear
starts_with(match) matches(match) everything()
```

**MANIPULATE MULTIPLE VARIABLES AT ONCE**

- across()** `across(cols, funs, ..., names = NULL)` Summarise or mutate multiple columns in the same way. `summarise(mtcars, across(everything(), mean))`
- c\_across()** `c_across(cols)` Compute across columns in row-wise data. `transmute(pwrowwise(LMgas, total = sum(c_across(1:20)))`

**MAKE NEW VARIABLES**

Apply **vectorized functions** to columns. Vectorized functions take vectors as input and return vectors of the same length as output (see back).

**vectorized function**

- mutate()** `mutate(data, ..., keep = "all", before = NULL, after = NULL)` Compute new column(s). Also **add\_column()**, **add\_count()**, and **add\_tally()**. `mutate(mtcars, gpm = 1 / mpg)`
- transmute()** `transmute(data, ...)` Compute new column(s), drop others. `transmute(mtcars, gpm = 1 / mpg)`
- rename()** `rename(data, ...)` Rename columns. Use **rename\_with()** to rename with a function. `rename(mtcars, distance = dist)`

RStudio

RStudio® is a trademark of RStudio, PBC • CC BY SA RStudio • info@rstudio.com • 800-499-1212 • rstudio.com • Learn more at [dplyr.tidyverse.org](https://dplyr.tidyverse.org) • dplyr 1.0.7 • Updated: 2021-07

### 1.2.7 缺失值

- 运算中的数据一旦出现了NA，结果就会出现NA：

```
c(1, 2, 3, NA) %>% sum()
```

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

- 在运算的时候，可以强制忽略缺失值：

```
c(1, 2, 3, NA) %>% sum(na.rm = TRUE)
```

```
## [1] 6
```

- 很多函数都有na.rm这个选项，如mean(), var()等。

We can also do something to the original data to delete the NA value:

```
mean(penguins$body_mass_g)
```

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

```
penguins1 <- penguins[complete.cases(penguins),]  
mean(penguins1$body_mass_g)
```

```
## [1] 4207.057
```

### 1.2.8 数据规整

若我们有一组植物的高度数据

```
plant_height <- data.frame(  
  Day = 1:5,  
  A = c(0.7, 1.0, 1.5, 1.8, 2.2),  
  B = c(0.5, 0.7, 0.9, 1.3, 1.8),  
  C = c(0.3, 0.6, 1.0, 1.2, 2.2),  
  D = c(0.4, 0.7, 1.2, 1.5, 3.2)  
)
```

此时的数据形如：

```
plant_height
```

```
##   Day   A   B   C   D  
## 1    1 0.7 0.5 0.3 0.4  
## 2    2 1.0 0.7 0.6 0.7  
## 3    3 1.5 0.9 1.0 1.2  
## 4    4 1.8 1.3 1.2 1.5  
## 5    5 2.2 1.8 2.2 3.2
```

若我们想将其转化为：

```
## # A tibble: 20 x 3
##       Day plant height
##   <int> <chr>   <dbl>
## 1     1    1 A       0.7
## 2     2    1 B       0.5
## 3     3    1 C       0.3
## 4     4    1 D       0.4
## 5     5    2 A       1
## 6     6    2 B       0.7
## 7     7    2 C       0.6
## 8     8    2 D       0.7
## 9     9    3 A       1.5
## 10    10    3 B       0.9
## 11    11    3 C       1
## 12    12    3 D       1.2
## 13    13    4 A       1.8
## 14    14    4 B       1.3
## 15    15    4 C       1.2
## 16    16    4 D       1.5
## 17    17    5 A       2.2
## 18    18    5 B       1.8
## 19    19    5 C       2.2
## 20    20    5 D       3.2
```

则我们需要 `pivot_longer()` 来使表格变长:

- `pivot_longer()`

上述步骤的代码为:

```
long <- plant_height %>% pivot_longer(cols = A:D, names_to = "plant", values_to = "height")
long
```

```
## # A tibble: 20 x 3
```

```
##      Day plant height
##      <int> <chr>  <dbl>
##  1      1 A      0.7
##  2      1 B      0.5
##  3      1 C      0.3
##  4      1 D      0.4
##  5      2 A      1
##  6      2 B      0.7
##  7      2 C      0.6
##  8      2 D      0.7
##  9      3 A      1.5
## 10      3 B      0.9
## 11      3 C      1
## 12      3 D      1.2
## 13      4 A      1.8
## 14      4 B      1.3
## 15      4 C      1.2
## 16      4 D      1.5
## 17      5 A      2.2
## 18      5 B      1.8
## 19      5 C      2.2
## 20      5 D      3.2
```

- `pivot_wider()`

同样，我们也有使表格变宽的方法：

```
wide <- long %>% pivot_wider(names_from = "plant",
values_from = "height") # names_from= 表示转换后的表格的列的名字来源; values_from= 表示转换后的表格的值来源
wide
```

```
## # A tibble: 5 x 5
##      Day      A      B      C      D
##      <int> <dbl> <dbl> <dbl> <dbl>
##  1      1  0.7  0.5  0.3  0.4
```

```
## 2      2      1      0.7    0.6    0.7
## 3      3     1.5     0.9     1     1.2
## 4      4     1.8     1.3     1.2    1.5
## 5      5     2.2     1.8     2.2    3.2
```

复杂一点的例子:

```
plant_record <- data.frame(
  day = c(1L, 2L, 3L, 4L, 5L),
  A_height = c(1.1, 1.2, 1.3, 1.4, 1.5),
  A_width = c(2.1, 2.2, 2.3, 2.4, 2.5),
  A_depth = c(3.1, 3.2, 3.3, 3.4, 3.5),
  B_height = c(4.1, 4.2, 4.3, 4.4, 4.5),
  B_width = c(5.1, 5.2, 5.3, 5.4, 5.5),
  B_depth = c(6.1, 6.2, 6.3, 6.4, 6.5),
  C_height = c(7.1, 7.2, 7.3, 7.4, 7.5),
  C_width = c(8.1, 8.2, 8.3, 8.4, 8.5),
  C_depth = c(9.1, 9.2, 9.3, 9.4, 9.5)
)
as_tibble(plant_record)
```

```
## # A tibble: 5 x 10
##   day A_height A_width A_depth B_height B_width B_depth C_height C_width C_depth
##   <int>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
## 1     1     1.1     2.1     3.1     4.1     5.1     6.1     7.1     8.1
## 2     2     1.2     2.2     3.2     4.2     5.2     6.2     7.2     8.2
## 3     3     1.3     2.3     3.3     4.3     5.3     6.3     7.3     8.3
## 4     4     1.4     2.4     3.4     4.4     5.4     6.4     7.4     8.4
## 5     5     1.5     2.5     3.5     4.5     5.5     6.5     7.5     8.5
## # ... with abbreviated variable names 1: B_height, 2: C_height
```

```
plant_record_longer <- plant_record %>%
  tidyr::pivot_longer(
    cols = !day,
```



```
names_to = c("species", ".value"),
names_pattern = "(.*)_(.*)"
)
plant_record_longer %>% slice(1:10)
```

```
## # A tibble: 10 x 5
##   day species height width depth
##   <int> <chr>   <dbl> <dbl> <dbl>
## 1     1 A       1.1   2.1   3.1
## 2     1 B       4.1   5.1   6.1
## 3     1 C       7.1   8.1   9.1
## 4     2 A       1.2   2.2   3.2
## 5     2 B       4.2   5.2   6.2
## 6     2 C       7.2   8.2   9.2
## 7     3 A       1.3   2.3   3.3
## 8     3 B       4.3   5.3   6.3
## 9     3 C       7.3   8.3   9.3
## 10    4 A       1.4   2.4   3.4
```

变回去:

```
plant_record_wider <- plant_record_longer %>%
tidyr::pivot_wider(
names_from = species,
values_from = c(height, width, depth),
names_glue = "{species}_{.value}"
)
plant_record_wider
```

```
## # A tibble: 5 x 10
##   day A_height B_height C_height A_width B_width C_width A_depth B_depth C_depth
##   <int>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
## 1     1     1.1     4.1     7.1     2.1     5.1     8.1     3.1     6.1     9.1
```

```
## 2      2      1.2      4.2      7.2      2.2      5.2      8.2      3.2      6.2      9.2
## 3      3      1.3      4.3      7.3      2.3      5.3      8.3      3.3      6.3      9.3
## 4      4      1.4      4.4      7.4      2.4      5.4      8.4      3.4      6.4      9.4
## 5      5      1.5      4.5      7.5      2.5      5.5      8.5      3.5      6.5      9.5
## # ... with abbreviated variable names 1: B_height, 2: C_height
```

### 1.3 From data set to a random variables

#### 1.3.1 summary()

- Working with numbers:
  - center: sample mean, sample median, and so on
  - spread: standard deviation, range, quantiles, IQR(Interquartile range), and so on
  - skewness

```
body_mass_g <- penguins$body_mass_g
summary(body_mass_g)
```

```
##      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.     NA's
##      2700    3550    4050    4202    4750    6300         2
```

Or we can customize our summary output:

```
penguins %>%
  summarize(mean = mean(body_mass_g),
            median = median(body_mass_g),
            sd = sd(body_mass_g),
            IQR = IQR(body_mass_g, na.rm=TRUE))
```

```
## # A tibble: 1 x 4
##   mean median    sd  IQR
##   <dbl> <int> <dbl> <dbl>
## 1    NA     NA    NA  1200
```

If we want to calculate the mean of one particular *species*, we can use `group_by()` and `summarize`

```
penguins %>%
  group_by(species) %>%
  summarize(n = length(body_mass_g),
            mean = mean(body_mass_g, na.rm = TRUE),
            sd = sd(body_mass_g, na.rm = TRUE))
```

```
## # A tibble: 3 x 4
##   species      n mean   sd
##   <fct>    <int> <dbl> <dbl>
## 1 Adelie    152 3701.  459.
## 2 Chinstrap  68 3733.  384.
## 3 Gentoo   124 5076.  504.
```

We can add more information like *sex*:

```
penguins[complete.cases(penguins),]%>% # 去掉所有的 NA
  group_by(species,sex) %>%
  summarize(n = length(body_mass_g),
            mean = mean(body_mass_g, na.rm = TRUE),
            sd = sd(body_mass_g, na.rm = TRUE))
```

```
## `summarise()` has grouped output by 'species'. You can override using the
## `.groups` argument.
```

```
## # A tibble: 6 x 5
## # Groups:   species [3]
##   species sex      n mean   sd
##   <fct>   <fct> <int> <dbl> <dbl>
## 1 Adelie female   73 3369.  269.
## 2 Adelie male    73 4043.  347.
## 3 Chinstrap female 34 3527.  285.
```

```
## 4 Chinstrap male      34 3939.  362.
## 5 Gentoo    female    58 4680.  282.
## 6 Gentoo    male      61 5485.  313.
```

### 1.3.2 Probability models

- Joint count:

```
(joint_table <- penguins %>%
  xtabs(~species + sex, data = .)) %>% # 管道操作时，传过来的数据做为非第一参数时，必须用
  addmargins() # 计算 sum
```

```
##           sex
## species  female male Sum
##   Adelie      73   73 146
##   Chinstrap   34   34  68
##   Gentoo      58   61 119
##   Sum         165  168 333
```

- Joint probability

```
joint_table %>%
  prop.table() %>%
  round(digit = 3) %>% # 保留三位有效数字
  addmargins()
```

```
##           sex
## species  female male  Sum
##   Adelie    0.219 0.219 0.438
##   Chinstrap 0.102 0.102 0.204
##   Gentoo    0.174 0.183 0.357
##   Sum       0.495 0.504 0.999
```

- Marginal distribution

```
library(magrittr)
```

```
##
```

```
## 载入程辑包: 'magrittr'
```

```
## The following object is masked from 'package:purrr':
```

```
##
```

```
##      set_names
```

```
## The following object is masked from 'package:tidyr':
```

```
##
```

```
##      extract
```

```
joint_table %>%
```

```
margin.table(1) %T>% # 向左传递值到 print(), 但下一个%>% 仍然由 margin.table(1) 传递而不是
```

```
print() %>%
```

```
prop.table()
```

```
## species
```

```
##      Adelie Chinstrap      Gentoo
```

```
##      146          68        119
```

```
## species
```

```
##      Adelie Chinstrap      Gentoo
```

```
## 0.4384384 0.2042042 0.3573574
```

```
joint_table %>%
```

```
margin.table(2) %T>%
```

```
print() %>%
```

```
prop.table()
```

```
## sex
```

```
## female   male
```

```
##      165     168
```

```
## sex
##   female      male
## 0.4954955 0.5045045
```

- Conditional distribution

```
joint_table %>% prop.table(margin = 1) # 已知第一个变量 species 求条件分布
```

```
##           sex
## species   female    male
## Adelie    0.500000 0.500000
## Chinstrap 0.500000 0.500000
## Gentoo    0.487395 0.512605
```

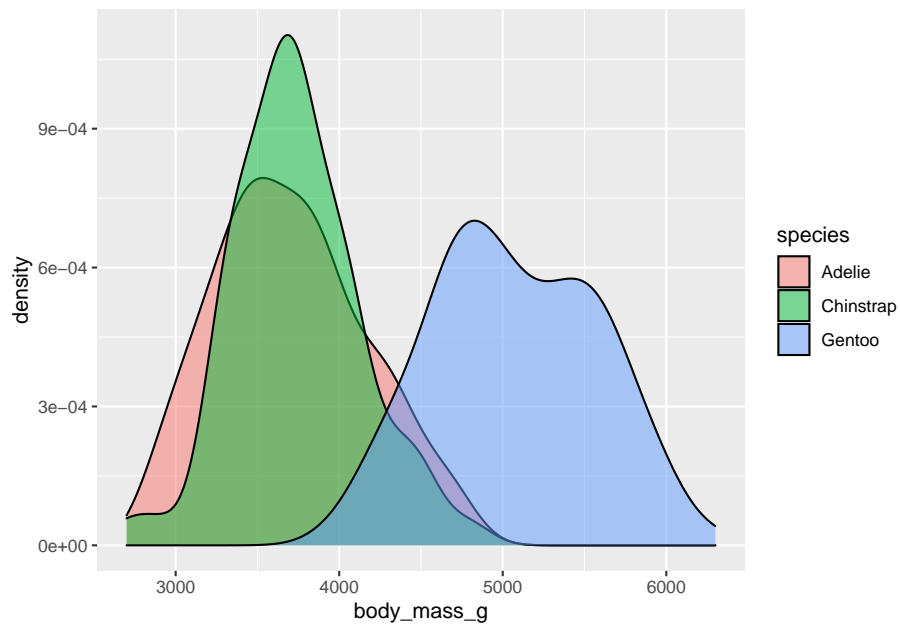
```
joint_table %>% prop.table(margin = 2)
```

```
##           sex
## species   female    male
## Adelie    0.4424242 0.4345238
## Chinstrap 0.2060606 0.2023810
## Gentoo    0.3515152 0.3630952
```

- Draw distribute plot

```
ggplot(data = penguins, aes(x = body_mass_g, y = ..density.., fill = species)) +  
  geom_density(color = "black", alpha = 0.5)
```

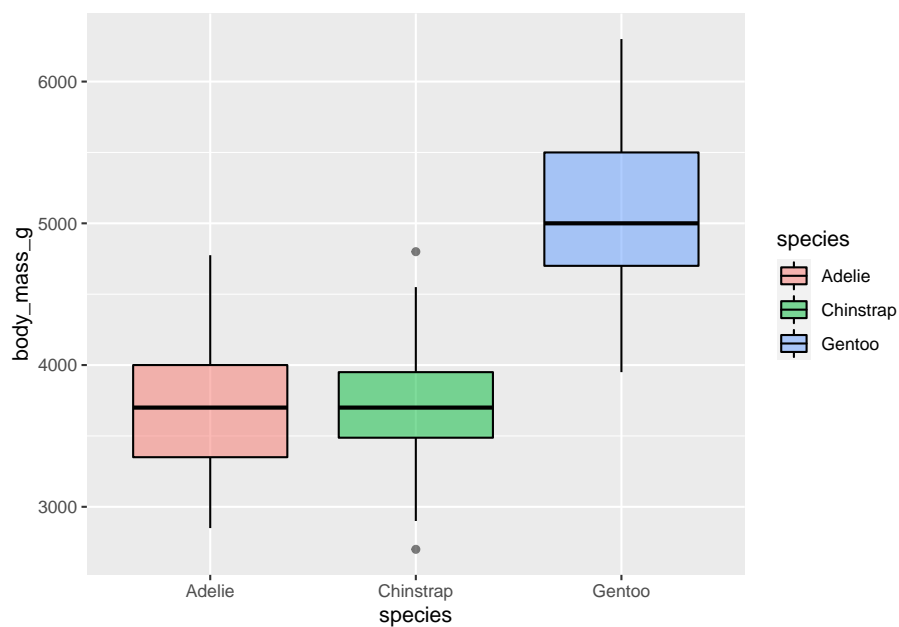
```
## Warning: Removed 2 rows containing non-finite values (stat_density).
```



Sometimes we can also use `boxplot()`

```
ggplot(data = penguins, aes(x=species, y=body_mass_g, fill=species)) +  
  geom_boxplot(color="black", alpha=0.5)
```

```
## Warning: Removed 2 rows containing non-finite values (stat_boxplot).
```



- Calculate Sample Mean and Covariance

### Mean

```
penguins1 %>%
  select(where(is.numeric)) %>%
  colMeans() %>%
  knitr::kable() # 用表格表示
```

	x
bill_length_mm	43.99279
bill_depth_mm	17.16486
flipper_length_mm	200.96697
body_mass_g	4207.05706
year	2008.04204

### Covariance Matrix



```
penguins1 %>%
  select(where(is.numeric)) %>%
  rename(X1 = bill_length_mm,
         X2 = bill_depth_mm,
         X3 = flipper_length_mm,
         X4 = body_mass_g,
         X5 = year) %>% # 改名防止矩阵名字过长影响美观
  var() %>%
  round(digit = 2) %>%
  knitr:: kable()
```

	X1	X2	X3	X4	X5
X1	29.91	-2.46	50.06	2595.62	0.15
X2	-2.46	3.88	-15.95	-748.46	-0.08
X3	50.06	-15.95	196.44	9852.19	1.72
X4	2595.62	-748.46	9852.19	648372.49	14.31
X5	0.15	-0.08	1.72	14.31	0.66

### Correlation Matrix

```
penguins1 %>%
  select(where(is.numeric)) %>%
  rename(X1 = bill_length_mm,
         X2 = bill_depth_mm,
         X3 = flipper_length_mm,
         X4 = body_mass_g,
         X5 = year) %>% # 改名防止矩阵名字过长影响美观
  cor() %>%
  round(digit = 2) %>%
  knitr:: kable()
```

	X1	X2	X3	X4	X5
X1	1.00	-0.23	0.65	0.59	0.03
X2	-0.23	1.00	-0.58	-0.47	-0.05
X3	0.65	-0.58	1.00	0.87	0.15
X4	0.59	-0.47	0.87	1.00	0.02
X5	0.03	-0.05	0.15	0.02	1.00

## 1.4 Generate random variables by the build-in functions

### 1.4.1 Common probability distributions

Letter	Description
d	density
p	probability, distribution function
q	quantile function
r	random generation

For example:

```
dnorm(0) #Normal distribution 的  $f(x)$  在  $x=0$  处的值
```

```
## [1] 0.3989423
```

```
pnorm(0) # $F(0)$ 
```

```
## [1] 0.5
```

```
qnorm(0.25) #Normal distribution 的  $F(x)$  小于等于 0.25 时  $x$  的值
```

```
## [1] -0.6744898
```

```
rnorm(100) # 生成 n 个服从 Normal distribution 的点
```

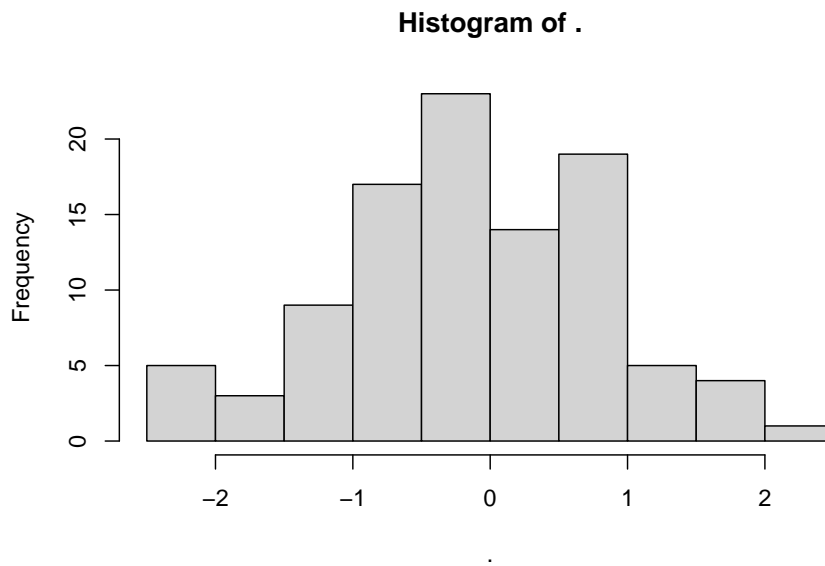
```
## [1] 1.18829096 -1.15645494 1.75901520 -1.62851463 -0.47929808 1.29158465
## [7] -1.37698614 0.81977064 1.13876763 -1.45196851 -0.52546849 0.47814181
## [13] 0.46604578 -1.07144375 0.51644292 -1.02343247 0.95530859 -0.49699640
## [19] 0.47972244 -1.05389021 0.47739010 0.26035254 0.50561321 -1.28990899
## [25] -2.34929216 0.80850840 0.82674754 -0.71999647 -0.53085095 2.32723527
## [31] 0.72535885 0.26664382 -1.33294735 -0.54392937 0.18923553 0.73232912
## [37] -0.81082234 -0.31146005 -1.92928874 -1.36538293 0.31933328 0.92352591
## [43] -0.48788156 -0.47385499 -0.16770338 0.40622148 0.35527935 -1.79442932
## [49] -1.70521416 -0.87196465 0.39392924 -0.04970255 -0.68412730 1.98220333
## [55] -0.29316760 -1.80714047 -0.56041460 2.17092448 0.14276964 -0.87940780
## [61] 0.42920082 0.78491586 0.81402102 0.05245506 2.17796729 1.04548216
## [67] -0.29899332 0.44371398 -1.17668937 -1.77741761 1.60512077 0.77862792
## [73] 0.32190387 0.16892348 1.42658837 0.53341022 -0.43232402 0.58773049
## [79] -0.76691045 -0.99110262 -0.79155863 1.54062052 1.44192545 -0.26641560
## [85] 1.18679781 -1.36513170 -0.51809399 -0.14828850 -1.03887170 -0.96044611
## [91] -0.79749702 -1.05890665 -2.58937365 -1.21071281 0.12490199 -0.10311855
## [97] 0.03490432 -1.15902916 -1.33368236 -0.56785331
```

## Common distributions

Distribution	Abbreviation	Distribution	Abbreviation
Beta	<b>beta</b>	Logistic	<b>logis</b>
Binomial	<b>binom</b>	Multinomial	<b>multinom</b>
Cauchy	<b>cauchy</b>	Negative binomial	<b>nbinom</b>
Chi-squared (noncentral)	<b>chisq</b>	Normal	<b>norm</b>
Exponential	<b>exp</b>	Poisson	<b>pois</b>
F	<b>f</b>	Wilcoxon Signed Rank	<b>signrank</b>
Gamma	<b>gamma</b>	T	<b>t</b>
Geometric	<b>geom</b>	Uniform	<b>unif</b>
Hypergeometric	<b>hyper</b>	Weibull	<b>weibull</b>
Lognormal	<b>lnorm</b>	Wilcoxon Rank Sum	<b>wilcox</b>

### 1.4.2 Generate normal random variables

```
rmnorm(100,mean=0,sd=1) %>%  
  hist()
```



```
rmnorm(100) %>%  
  summary()
```

```
##      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.  
## -2.01769 -0.86478 -0.11673 -0.06687  0.65905  2.23267
```

### 1.4.3 Generate multivariate normal random variables

```
library(MASS)
```

```
##  
## 载入程辑包: 'MASS'
```

```
## The following object is masked from 'package:dplyr':  
##  
##      select
```

```
set.seed(1234)  
mu <- c(10, 20)  
sigma <- matrix(c(1, 0.5, 0.5, 1), nrow = 2)  
mvrnorm(100, mu, sigma) %>%  
plot()
```

