时间序列分析与预测

第二讲

黄嘉平

深圳大学｜中国经济特区研究中心
粤海校区汇文楼办公楼 1510
课程网站 https://huangjp.com/TSAF/

1. The Tidyverse

1.1. 简介

Tidyverse (https://www.tidyverse.org) 是从数据科学的角度让 R 更好用的一系列工具包的集合。里面的所有工具包都建立在相同的理念之上，具有相同的语法，能处理相同的数据结构。

安装 `tidyverse` 工具包：运行 `install.packages("tidyverse")` 或通过 Tools > Install Packages … 菜单

调用 `tidyverse` 工具包：运行 `library(tidyverse)`，此命令可以调用核心工具包

如果你已经安装了 `fpp3`，那么你已经拥有了 `tidyverse` 中的一些核心工具包，因此只需调用 `fpp3` 即可。如果你在练习中遇到 `fpp3` 中没有包含的函数并因此报错，那么你只要安装并调用相应的工具包。

本讲内容基于 *R for Data Science* (2e) (https://r4ds.hadley.nz/) 的第一部份（1～8 章），因此在本讲中我们默认调用了 `tidyverse`。

1.2. tibble: tidyverse 中的通用数据结构

`mtcars` 是 base R 中提供的经典数据集之一，包含 1973-1974 年间美国市场上 32 种小汽车的油耗等 11 个变量的数据。我们可以用 `head()` 函数查看数据集的前 6 个观测值。

```
head(mtcars)
```

```
                   mpg cyl disp  hp drat    wt  qsec vs am gear carb
Mazda RX4         21.0   6  160 110 3.90 2.620 16.46  0  1    4    4
Mazda RX4 Wag     21.0   6  160 110 3.90 2.875 17.02  0  1    4    4
Datsun 710        22.8   4  108  93 3.85 2.320 18.61  1  1    4    1
Hornet 4 Drive    21.4   6  258 110 3.08 3.215 19.44  1  0    3    1
Hornet Sportabout 18.7   8  360 175 3.15 3.440 17.02  0  0    3    2
Valiant           18.1   6  225 105 2.76 3.460 20.22  1  0    3    1
```

```
class(mtcars)
```

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

通过 `class()` 可知 `mtcars` 的类型是 data frame。

1.2. tibble: tidyverse 中的通用数据结构

```
mtcars_tibble <- as_tibble(mtcars)        # 转换成 tibble 形式保存
mtcars_tibble                             # 显示内容
```

```
      mpg   cyl  disp    hp  drat    wt  qsec    vs    am  gear  carb
    <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
 1  21        6  160    110  3.9   2.62  16.5     0     1     4     4
 2  21        6  160    110  3.9   2.88  17.0     0     1     4     4
 3  22.8      4  108     93  3.85  2.32  18.6     1     1     4     1
 4  21.4      6  258    110  3.08  3.22  19.4     1     0     3     1
 5  18.7      8  360    175  3.15  3.44  17.0     0     0     3     2
 6  18.1      6  225    105  2.76  3.46  20.2     1     0     3     1
 7  14.3      8  360    245  3.21  3.57  15.8     0     0     3     4
 8  24.4      4  147.    62  3.69  3.19  20       1     0     4     2
 9  22.8      4  141.    95  3.92  3.15  22.9     1     0     4     2
10  19.2      6  168.   123  3.92  3.44  18.3     1     0     4     4
# ℹ 22 more rows
# ℹ Use `print(n = ...)` to see more rows
```

1.2. tibble: tidyverse 中的通用数据结构

```
install.packages("nycflights13")
library(nycflights13)
glimpse(flights)        # 横向显示数据集 flights 中每个变量的类型和前几个观测值
```

```
Rows: 336,776
Columns: 19
$ year           <int> 2013, 2013, 2013, 2013, 2013, 2013, 2013, 2013, …
$ month          <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, …
$ day            <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, …
$ dep_time       <int> 517, 533, 542, 544, 554, 554, 555, 557, 557, 558…
$ sched_dep_time <int> 515, 529, 540, 545, 600, 558, 600, 600, 600, 600…
$ dep_delay      <dbl> 2, 4, 2, -1, -6, -4, -5, -3, -3, -2, -2, -2, -2,…
$ arr_time       <int> 830, 850, 923, 1004, 812, 740, 913, 709, 838, 75…
$ sched_arr_time <int> 819, 830, 850, 1022, 837, 728, 854, 723, 846, 74…
$ arr_delay      <dbl> 11, 20, 33, -18, -25, 12, 19, -14, -8, 8, -2, -3…
$ carrier        <chr> "UA", "UA", "AA", "B6", "DL", "UA", "B6", "EV", …
$ flight         <int> 1545, 1714, 1141, 725, 461, 1696, 507, 5708, 79,…
```

1.2. tibble: tidyverse 中的通用数据结构

```
$ tailnum      <chr> "N14228", "N24211", "N619AA", "N804JB", "N668DN"…
$ origin       <chr> "EWR", "LGA", "JFK", "JFK", "LGA", "EWR", "EWR",…
$ dest         <chr> "IAH", "IAH", "MIA", "BQN", "ATL", "ORD", "FLL",…
$ air_time     <dbl> 227, 227, 160, 183, 116, 150, 158, 53, 140, 138,…
$ distance     <dbl> 1400, 1416, 1089, 1576, 762, 719, 1065, 229, 944…
$ hour         <dbl> 5, 5, 5, 5, 6, 5, 6, 6, 6, 6, 6, 6, 6, 6, 6, 5, …
$ minute       <dbl> 15, 29, 40, 45, 0, 58, 0, 0, 0, 0, 0, 0, 0, 0, 0…
$ time_hour    <dttm> 2013-01-01 05:00:00, 2013-01-01 05:00:00, 2013-…
```

由此可知，这个数据集中包含 333,776 个观测值，19 个变量。变量包含不同的类型

- `<int>` 代表 integer，保存的是整数
- `<dbl>` 代表 double，保存的是实数
- `<cht>` 代表 character，保存的是字符串（文字信息）
- `<dttm>` 代表 date-time，保存的是时间

1.3. 数据的提取和转换

`mtcars` 中的 `mpg` 变量保存了每个车型的油耗（miles per gallon）。如果我们想提取 $\text{mpg} > 25$ 的观测值应该如何实现呢?

以下两种方法都会给出同样的结果。

```
filter(mtcars_tibble, mpg > 25)
```

```
mtcars_tibble |> filter(mpg > 25)
```

```
# A tibble: 6 × 11
    mpg   cyl  disp    hp  drat    wt  qsec    vs    am  gear  carb
  <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
1  32.4     4  78.7    66  4.08  2.2   19.5     1     1     4     1
2  30.4     4  75.7    52  4.93  1.62  18.5     1     1     4     2
3  33.9     4  71.1    65  4.22  1.84  19.9     1     1     4     1
4  27.3     4  79      66  4.08  1.94  18.9     1     1     4     1
5  26       4 120.     91  4.43  2.14  16.7     0     1     5     2
6  30.4     4  95.1   113  3.77  1.51  16.9     1     1     5     2
```

1.3. 数据的提取和转换

这两个命令中先使用了 `dplyr` 工具包里面的 `filter()` 命令。

在调用 `tidyverse` 时，你是否注意到下面的信息？

```
── Conflicts ──────────────────────────────────────── tidyverse_conflicts() ──
✖ dplyr::filter() masks stats::filter()
✖ dplyr::lag()    masks stats::lag()
```

这是说原本在 base R 的 `stats` 工具包中存在 `filter()` 和 `lag()` 函数，但是现在它们的功能被 `dplyr` 中的同名函数覆盖了。如果要调用 `stats` 中的函数，则需使用 `stats::filter()` 或 `stats::lag()`。

其次，在命令 `mtcars_tibble |> filter(mpg > 25)` 中使用了符号 `|>`。它被称为 pipe operator，可以讲其左边的内容作为参数传递给右面的函数。因此，`x |> f(y)` 等同于 `f(x,y)`。

1.3. 数据的提取和转换

`|>` 可以多次传递。例如 `x |> f(y) |> g(z)` 等同于 `g(f(x,y),z)`。

```
mtcars_tibble |>
  filter(mpg > 20) |>                    # 选取 mpg > 20 的观测值
  group_by(gear) |>                      # 按照 gear 的取值分组
  summarise(mean_mpg = mean(mpg))        # 计算每组中 mpg 的平均值
```

```
# A tibble: 3 × 2
   gear mean_mpg
  <dbl>    <dbl>
1     3     21.4
2     4     25.7
3     5     28.2
```

在 RStudio 中可以利用快捷键 Ctrl + Shift + m 快速输入 `|>`，但首先需要在 Tools > Global Options > Code 中选中 “Use native pipe operator, |> (requires R 4.1 +)”。

1.3. 数据的提取和转换

`dplyr` 中包含关于数据操作的函数。

- 行操作
  - `filter()`：根据列的取值选择行
  - `arrange()`：根据列的取值改变行的排列顺序
  - `distinct()`：去掉重复的行
- 列操作
  - `mutate()`：将已有列的计算结果添加为新的列
  - `select()`：根据名称选择列
  - `rename()`：改变列的名称
  - `relocate()`：调整列的排列顺序
- 分组操作
  - `group_by()`：给数据添加分组（并不改变原数据）
  - `summarise()`：将每个组的数据归纳为一行

2. Data Tidying（数据整理）

2.1. 数据信息的不同保存方式

无论是 data frame 还是 tibble 类型的变量，都不会对数据本身做出太多要求。同样的信息可以用不同的方式保存。

让我们考虑下面的表格中包含的数据

Country	Year	Cases*	Population
Afghanistan	1999	745	19,987,071
Afghanistan	2000	2,666	20,595,360
Brazil	1999	37,737	172,006,362
Brazil	2000	80,488	174,504,898
China	1999	212,258	1,272,915,272
China	2000	213,766	1,280,428,583

* Cases of tuberculosis: 结核病例数

这里有四个变量 country, year, cases, population，六个观测值。（此类数据称为 panel/longitudinal data，包含每个个体在不同时间点的观测值。）

2.2. 什么样的数据可以称为 tidy？

```
table1
```

```
# A tibble: 6 × 4
  country      year  cases population
  <chr>       <dbl>  <dbl>      <dbl>
1 Afghanistan  1999    745   19987071
2 Afghanistan  2000   2666   20595360
3 Brazil       1999  37737  172006362
4 Brazil       2000  80488  174504898
5 China        1999 212258 1272915272
6 China        2000 213766 1280428583
```

`table1` 中的数据保存方式和前面的表格一致。

2.2. 什么样的数据可以称为 tidy？

`table2`

```
# A tibble: 12 × 4
   country      year type            count
   <chr>       <dbl> <chr>           <dbl>
 1 Afghanistan  1999 cases             745
 2 Afghanistan  1999 population    19987071
 3 Afghanistan  2000 cases            2666
 4 Afghanistan  2000 population    20595360
 5 Brazil       1999 cases            37737
 6 Brazil       1999 population   172006362
 7 Brazil       2000 cases            80488
 8 Brazil       2000 population   174504898
 9 China        1999 cases           212258
10 China        1999 population  1272915272
11 China        2000 cases           213766
12 China        2000 population  1280428583
```

`table2` 中的信息内容虽然和 `table1` 相同，但行数翻倍了。

2.2. 什么样的数据可以称为 tidy？

```
table3
```

```
# A tibble: 6 × 3
  country      year rate
  <chr>       <dbl> <chr>
1 Afghanistan  1999 745/19987071
2 Afghanistan  2000 2666/20595360
3 Brazil       1999 37737/172006362
4 Brazil       2000 80488/174504898
5 China        1999 212258/1272915272
6 China        2000 213766/1280428583
```

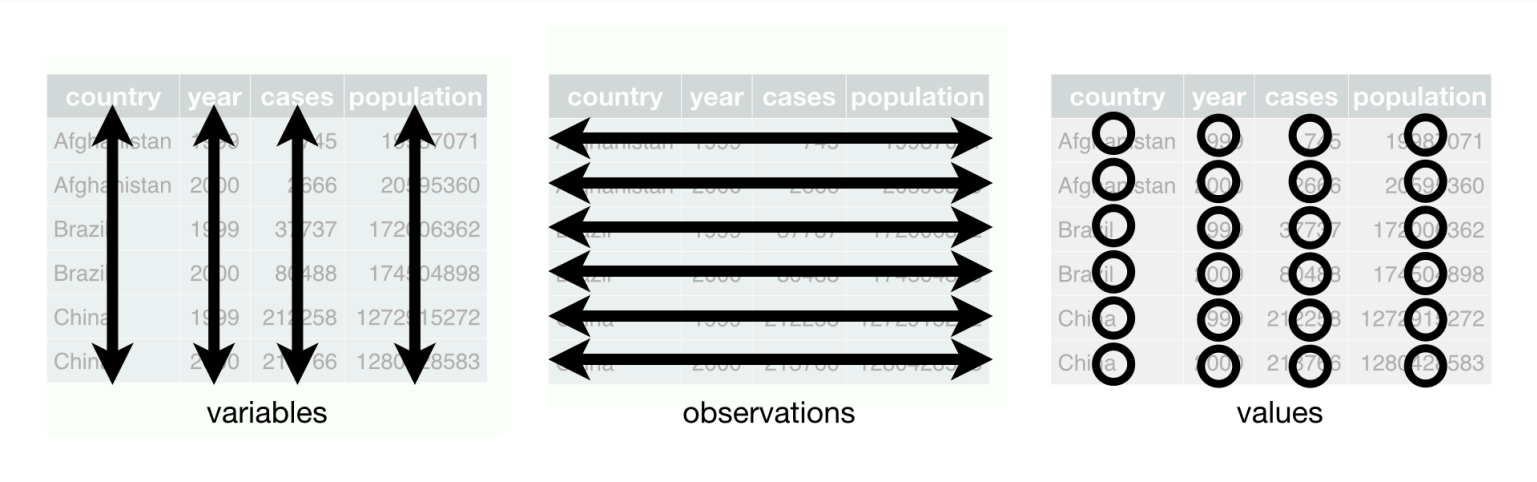
`table3` 将 cases/population 以字符串的形式保存在 rate 变量中。虽然内容和行数都和 `table1` 相同，但不方便处理信息。

2.2. 什么样的数据可以称为 tidy？

tidy: arrange neatly and in order/整洁的；有条理的。反义词是 messy。

tidy 数据应符合以下三个条件：

1. 一个变量（variable）只放在一列中，且每一列只包含一个变量。
2. 一个观测值（observation）只放在一行中，且每一行只包含一个观测值。
3. 一个数值（value）只放在一个单元格（cell）中，且每一个单元格只包含一个数值。



在前面的三个 tibble 数据中，只有 `table1` 是 tidy 的。

2.3. 整理数据（tidying）

现实世界中大多数数据都不是 tidy 的，这是因为人们在记录数据时很少考虑如何让它们分析起来更方便。因此，在获得数据后，我们首先需要对它进行整理（tidying）。

`tidyverse` 中工具包都是针对 tidy 数据开发的。其中 `tidyr` 给我们提供了很多整理数据的函数。下面介绍几个比较常用的：

- `pivot_longer()`：当很多列的名称本身可以构成一个分类变量，且内容相似时，此命令可以将它们重新构成两列（名称和内容），同时使数据集变得更长（即增加了观测值的数量）。
- `pivot_wider()`：和 `pivot_longer()` 相反的操作。
- `unite()`：将多列合并为一列（内容为字符串）。
- `separate_longer_delim()`, `separate_wider_delim()`：将一列拆分为多列。

想了解其他函数可参考 https://tidyr.tidyverse.org 。

2.3. 整理数据（tidying）

利用 `pivot_wider()` 将 `table2` 整理成 `table1`：

```
table2 |> pivot_wider(names_from = type, values_from = count)
```

若想将 `table3` 整理成 `table1`，则需要对 `rate` 列中的分子和分母进行分离。此操作可以用 `separate_wider_delim()`：

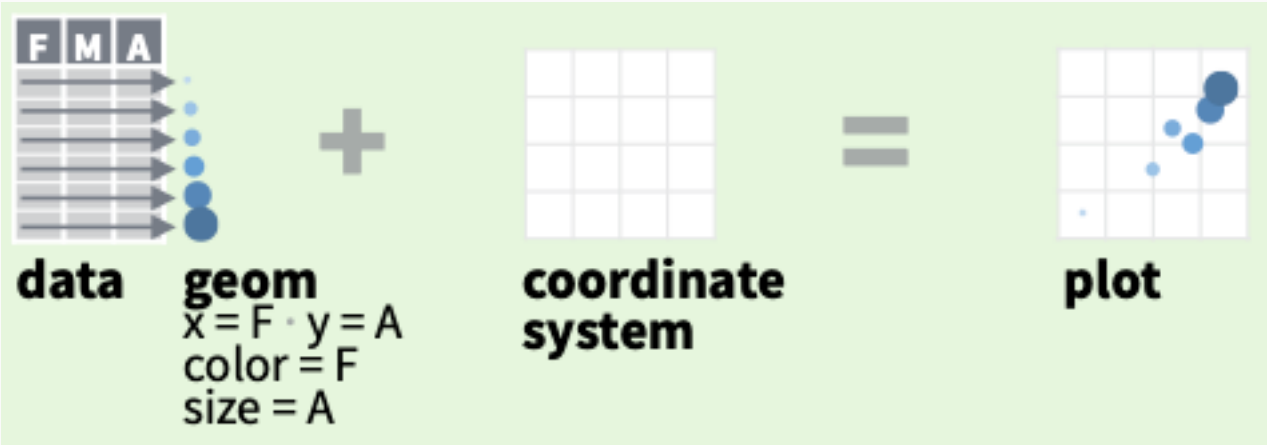
```
table3 |>
  separate_wider_delim(    # 将 rate 中 "/" 前后部份各自存为一列
    rate, delim = "/", names = c("cases", "pop")
        # 将 rate 中 "/" 前后部份各自存为一列，均为 chr 类型
  ) |>
  mutate(cases = as.numeric(cases), pop = as.numeric(pop))
        # 将 chr 类型的数值转变为 dbl 类型
```

3. 用 ggplot2 绘图

3.1. ggplot2

在 R 中进行绘图的方法有很多，不仅 base R 包含了多个绘图函数，也有不少工具包提供特殊用途的函数。其中，`tidyverse` 包含的 `ggplot2` 工具包是目前最流行也是功能最强大的一个。

`ggplot2` 的名称源自 *The Grammer of Graphics, 2e.* (Wilkinson, L., 2005, Springer)。它的基本绘图逻辑可以归纳为：数据 + 绘图领域 + 绘图形式（geom）。



3.2. 南极洲 Palmer 群岛企鹅数据集

`palmerpenguins` 工具包包含了南极洲 Palmer 群岛上记录的企鹅数据，共有 3 个种类 344 只个体。https://allisonhorst.github.io/palmerpenguins/

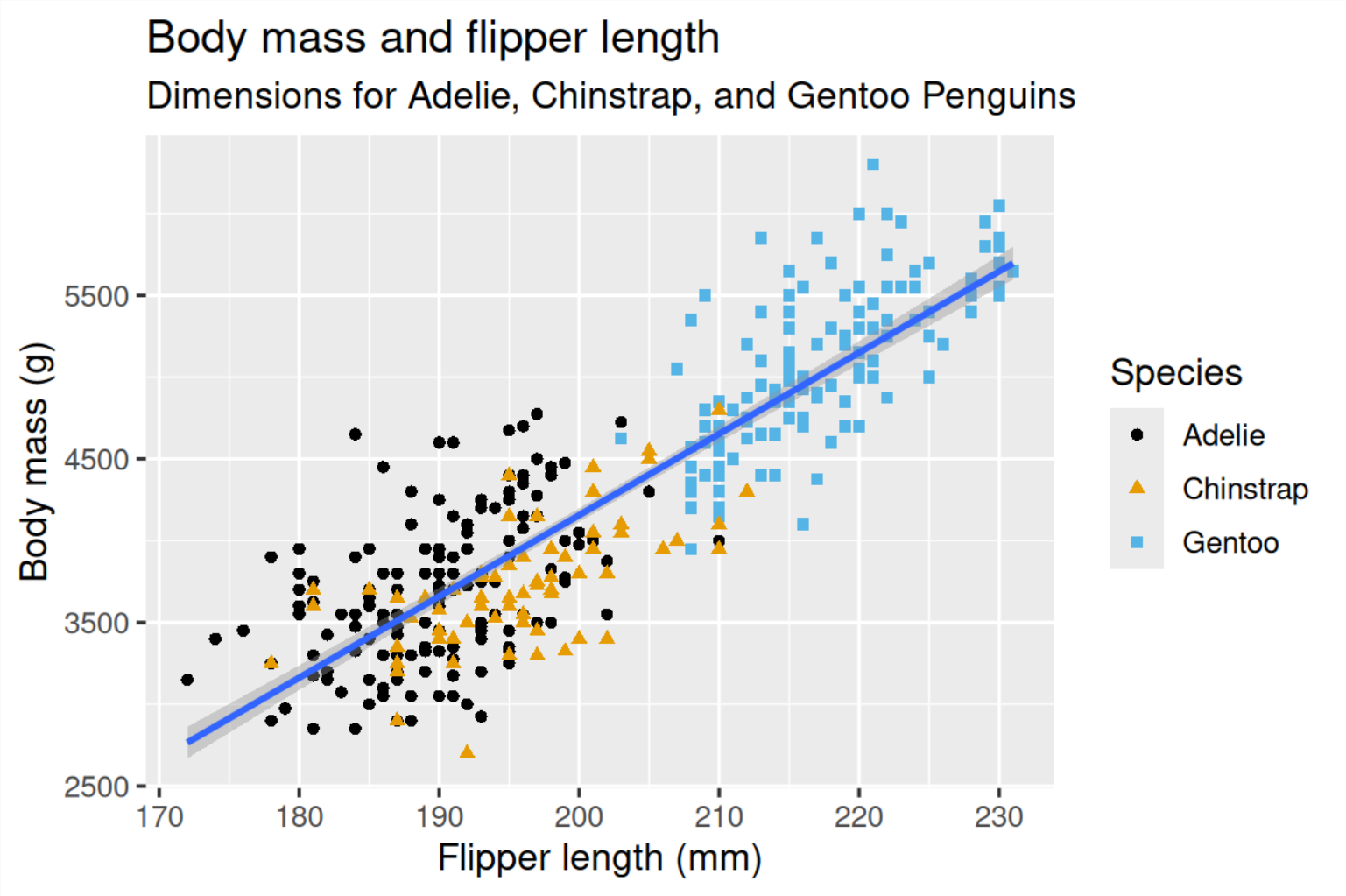
```
install.packages("palmerpenguins")
```

```
library(palmerpenguins)
```

```
glimpse(penguins)
```

```
Rows: 344
Columns: 8
$ species           <fct> Adelie, Adelie, Adelie, Adelie, Adelie, Adelie, Ad…
$ island            <fct> Torgersen, Torgersen, Torgersen, Torgersen, Torger…
$ bill_length_mm    <dbl> 39.1, 39.5, 40.3, NA, 36.7, 39.3, 38.9, 39.2, 34.1…
$ bill_depth_mm     <dbl> 18.7, 17.4, 18.0, NA, 19.3, 20.6, 17.8, 19.6, 18.1…
$ flipper_length_mm <int> 181, 186, 195, NA, 193, 190, 181, 195, 193, 190, 1…
$ body_mass_g       <int> 3750, 3800, 3250, NA, 3450, 3650, 3625, 4675, 3475…
$ sex               <fct> male, female, female, NA, female, male, female, ma…
$ year              <int> 2007, 2007, 2007, 2007, 2007, 2007, 2007, 2007, 20…
```

3.2. 南极洲 Palmer 群岛企鹅数据集



3.3. 用 ggplot2 绘图

第一步：指定数据

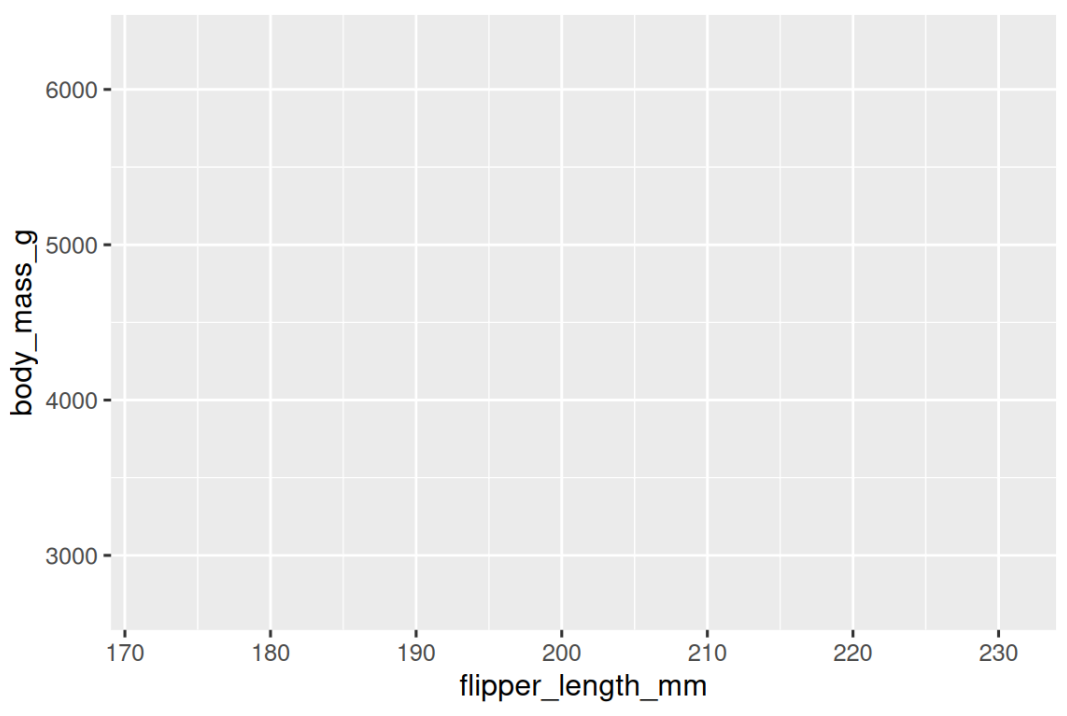
```
ggplot(data = penguins)
```

ggplot2 采用层层叠加的方式绘图。其基本命令只有一个 ggplot()。在尽指定数据的情况下，由于不清楚如何绘制，我们只能得到一个灰色的底板。

3.3. 用 ggplot2 绘图

第二步：指定坐标系

```
ggplot(
  data = penguins,
  mapping = aes(
    x = flipper_length_mm,
    y = body_mass_g
  )
)
```



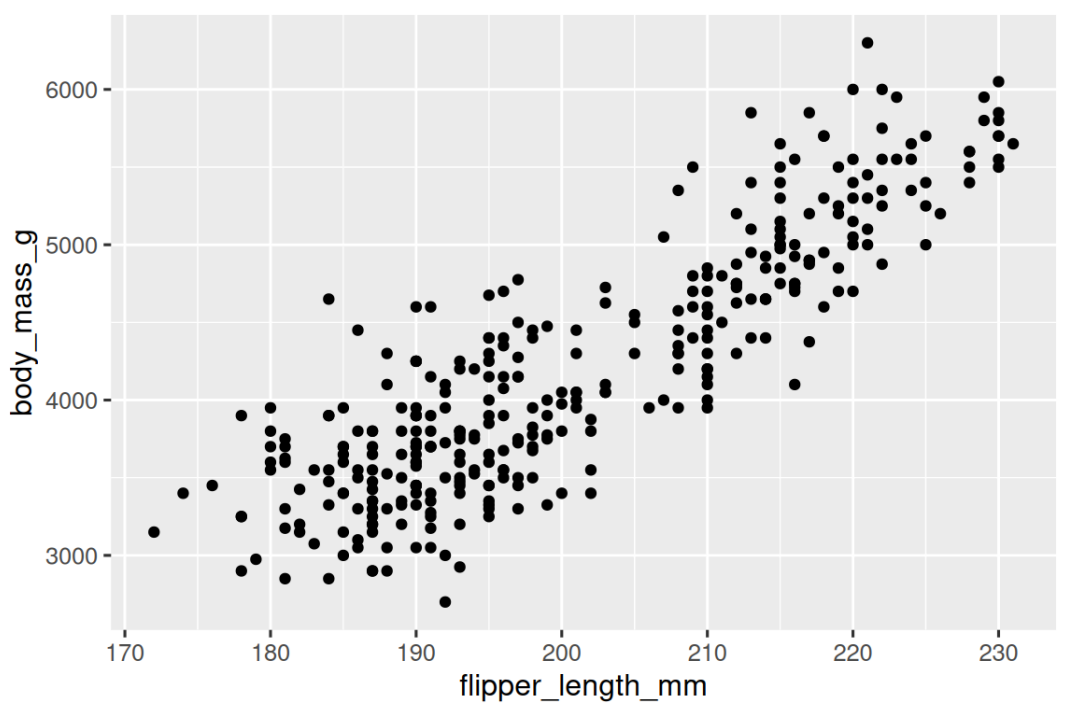
通过添加 `mapping` 参数后，我们获得了变量和坐标系的对应关系。

3.3. 用 ggplot2 绘图

第三步：指定绘图形式

```
ggplot(
  data = penguins,
  mapping = aes(
    x = flipper_length_mm,
    y = body_mass_g
  )
) + geom_point()
```

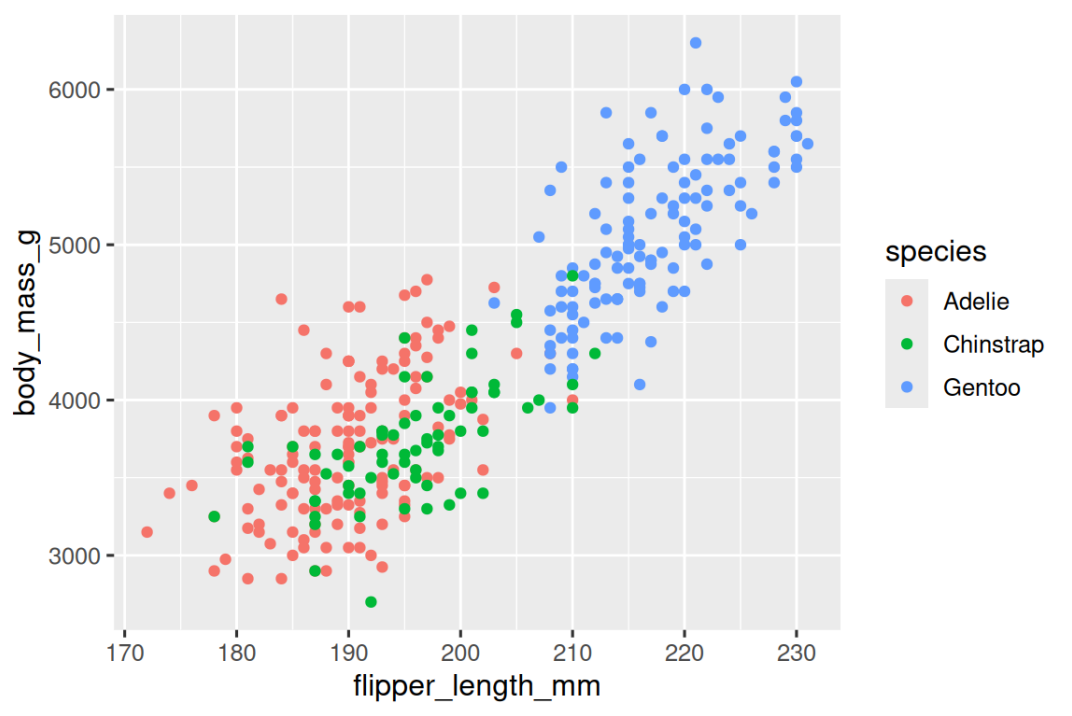
```
Warning message:
Removed 2 rows containing missing values or values outside the scale range (geom_point()).
```



3.3. 用 ggplot2 绘图

第四步：美化

```
ggplot(
  data = penguins,
  mapping = aes(
    x = flipper_length_mm,
    y = body_mass_g,
    color = species
  )
) + geom_point()
```

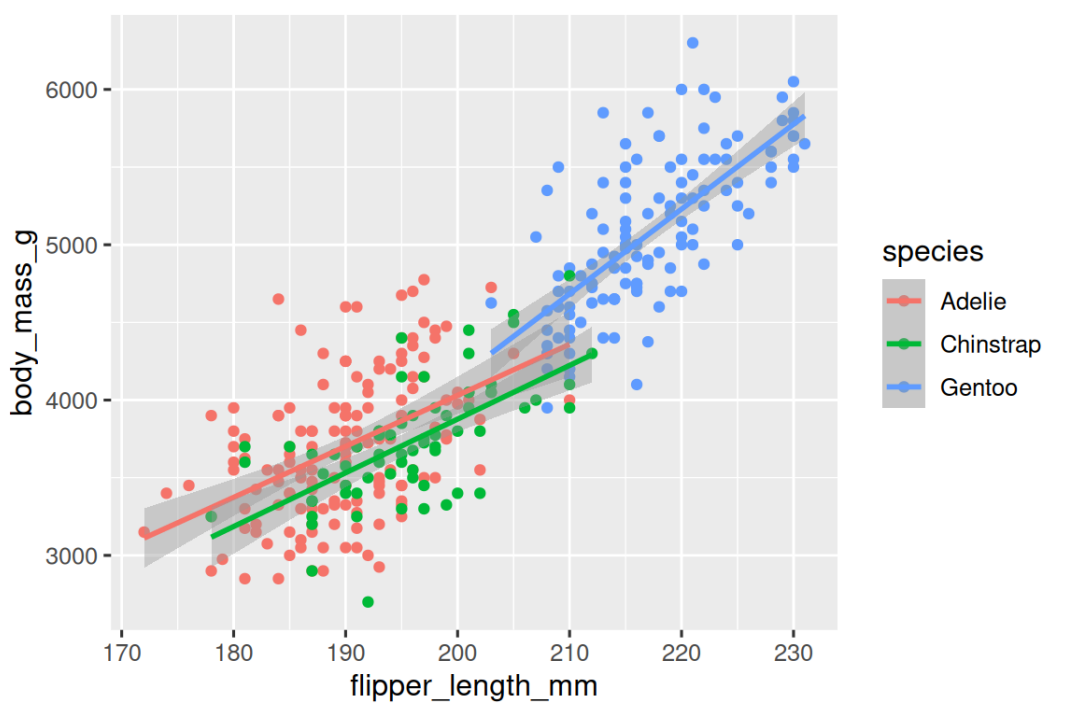


用颜色区分种类。

3.3. 用 ggplot2 绘图

第五步：添加其他要素

```
ggplot(
  data = penguins,
  mapping = aes(
    x = flipper_length_mm,
    y = body_mass_g,
    color = species
  )
) +
  geom_point() +
  geom_smooth(method = "lm")
```

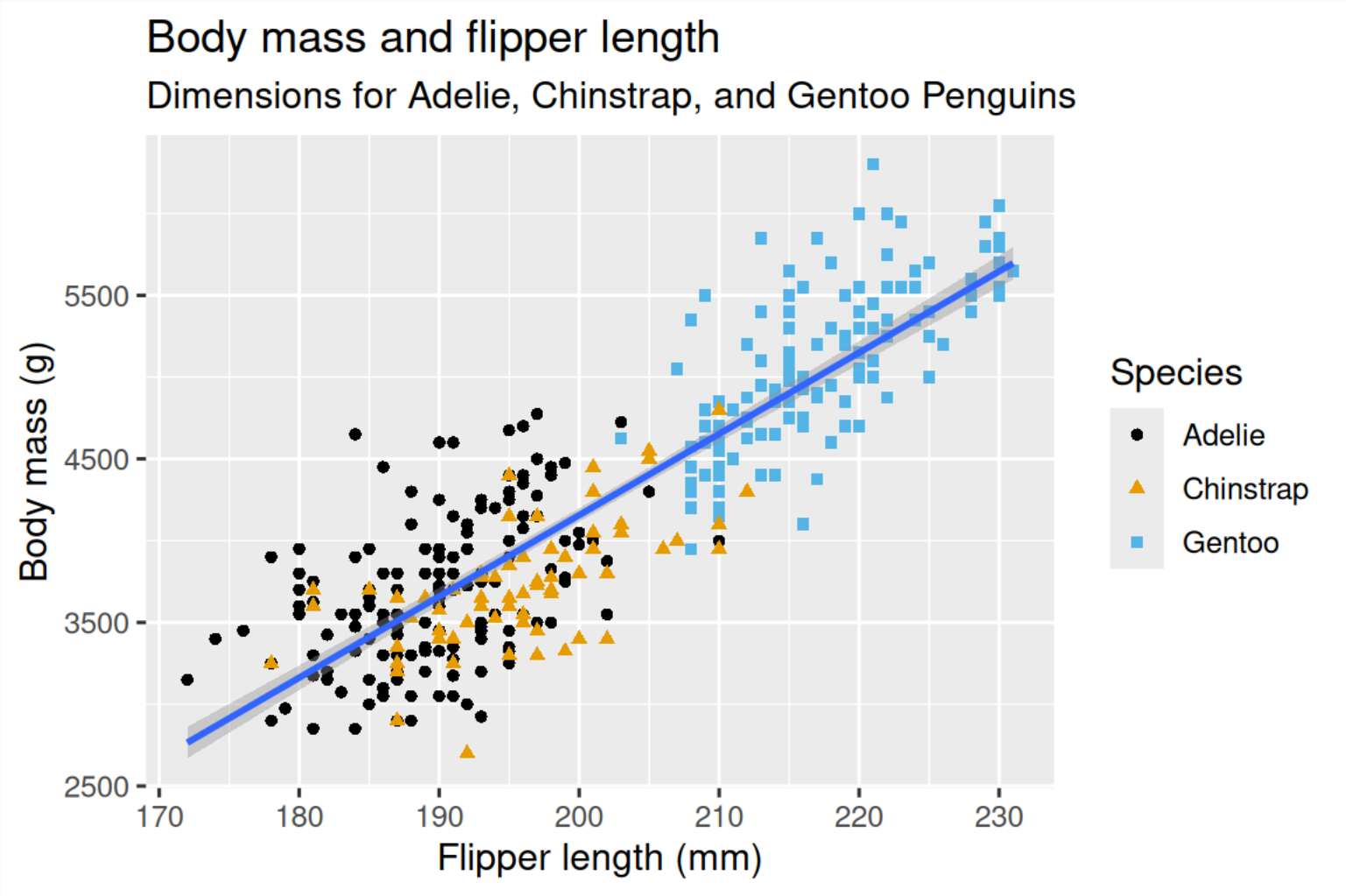


添加线性回归模型的拟合结果。

3.3. 用 ggplot2 绘图

```r
install.packages("ggthemes")
library(ggthemes)
ggplot(
  data = penguins,
  mapping = aes(x = flipper_length_mm, y = body_mass_g)
) +
  geom_point(aes(color = species, shape = species)) +    # 注意此处和前面的区别
  geom_smooth(method = "lm") +
  labs(      # 添加标题、副标题，更改坐标轴名称、图例名称
    title = "Body mass and flipper length",
    subtitle = "Dimensions for Adelie, Chinstrap, and Gentoo Penguins",
    x = "Flipper length (mm)", y = "Body mass (g)",
    color = "Species", shape = "Species"
  ) +
  scale_color_colorblind()     # 调用 ggthemes 工具包提供的色系（视觉障碍者也容易分辨）
```

3.3. 用 ggplot2 绘图



4. 课后练习

4. 课后练习

- 学习 *R for Data Science (2e.)* (https://r4ds.hadley.nz/) 第 1 ~ 8 章的内容。
- 了解 `ggplot2` 中的 `mpg` 数据集，并尝试回答下面的问题。

  1. 其中 `class` 变量是什么类型的数据？共有几种不同的取值？如果按其分类，每个类别有多少观测值？
  2. 抽取丰田产的所有四缸车的名称（剔除重复）。它们都是什么？
  3. 尝试画出下面的图表