

时间序列分析与预测

第三讲



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. 时间序列数据

1.1. tsibble 数据

在调用 `fpp3` 时，你会发现四个不属于 `tidyverse` 的工具包：`tsibble`, `tsibbledata`, `feasts`, `fable`。事实上，这四个工具包是 `tidyverts` 工具集的核心 (<https://tidyverts.org>)。

- `tsibble`: temporal data frames and tools
- `tsibbledata`: diverse datasets for `tsibble`
- `feasts`: features extraction and statistics
- `fable`: tidy forecasting

其中，`tsibble` 工具包提供了基于 `tibble` 的时间序列数据结构 `tsibble`，可以保存时间序列数据，并利用 `tidyverse` 中的其他函数进行数据整理。

1.1. tsibble 数据

考虑下面的时间序列数据：

Year	Observation
2015	123
2016	39
2017	78
2018	52
2019	110

下面我们将其保存为 tsibble 形式

```
y <- tsibble(  
  Year = 2015:2019,  
  Observation = c(123, 39, 78, 52, 110),  
  index = Year  
)  
y  
# A tsibble: 5 x 2 [1Y]  
  Year Observation  
  <int>         <dbl>  
1  2015         123  
2  2016          39  
3  2017          78  
4  2018          52  
5  2019         110
```

1.1. tsibble 数据

tsibble 数据比 tibble 数据多了两个参数，分别是 key 和 index，它们都是用来记录时间序列信息的。

在前面的例子中，我们将 Year 变量设置为 index (`index = Year`)，也就是指定了时间信息是保存在 Year 中的。此时，Year 承担了横截面数据中序号的作用，用来排列观测值的顺序。

我们还注意到，观测值的时间间隔也被保存在 tsibble 数据里

```
# A tsibble: 5 x 2 [1Y]
```

`tsibble()` 或 `as_tsibble()` 函数可以通过被指定为 index 的变量内容识别时间间隔，但需要提供正确的时间格式（见下页）。

1.1. tsibble 数据

假设我们有下面的 tibble 数据 `z`

```
z          # 该如何生成z?
```

```
# A tibble: 5 x 2
```

	Month	Observation
	<chr>	<dbl>
1	2019 Jan	50
2	2019 Feb	23
3	2019 Mar	34
4	2019 Apr	30
5	2019 May	25

这里 `Month` 变量是以字符串形式保存的，无法被识别为时间信息。因此我们需要将它转换为适当的时间格式。

转换时间数据的函数包括 `yearquarter()`, `yearmonth()`, `yearweek()`, `as_date()`, `as_datetime()` 等。

```
z |> mutate(  
  Month = yearmonth(Month)  
) |> as_tsibble(index = Month)
```

```
# A tsibble: 5 x 2 [1M]
```

	Month	Observation
	<mth>	<dbl>
1	2019 Jan	50
2	2019 Feb	23
3	2019 Mar	34
4	2019 Apr	30
5	2019 May	25

1.1. tsibble 数据

如果每个时间点上都有多个观测对象（例如各省的年度 GDP），我们也可以将它们都保存在一个 tsibble 数据里。这时候我们就需要用到 key 参数来排列观测对象。

tsibbledata 中提供的 `olympic_running` 数据集记录了每届奥运会（四年一度，缺少 1916、1940 和 1944 年）中跑步项目的最短用时数据。由于存在多种跑步项目，该数据集用 Length 和 Sex 的组合作为 key。

```
olympic_running
```

```
# A tsibble: 312 x 4 [4Y]
```

```
# Key:           Length, Sex [14]
```

	Year	Length	Sex	Time
	<int>	<int>	<chr>	<dbl>

1	1896	100	men	12
---	------	-----	-----	----

2	1900	100	men	11
---	------	-----	-----	----

3	1904	100	men	11
---	------	-----	-----	----

4	1908	100	men	10.8
---	------	-----	-----	------

5	1912	100	men	10.8
---	------	-----	-----	------

6	1916	100	men	NA
---	------	-----	-----	----

7	1920	100	men	10.8
---	------	-----	-----	------

8	1924	100	men	10.6
---	------	-----	-----	------

9	1928	100	men	10.8
---	------	-----	-----	------

10	1932	100	men	10.3
----	------	-----	-----	------

```
# i 302 more rows
```

```
# i Use `print(n = ...)` to see more rows
```

1.2. 对 tsibble 数据进行处理

下面我们对 `olympic_running` 数据进行整理。首先选出所有男子项目的成绩：

```
olympic_running |> filter(Sex == "men")
```

```
# A tsibble: 208 x 4 [4Y]
```

```
# Key:      Length, Sex [7]
```

	Year	Length	Sex	Time
	<int>	<int>	<chr>	<dbl>
1	1896	100	men	12
2	1900	100	men	11
3	1904	100	men	11
4	1908	100	men	10.8
5	1912	100	men	10.8
6	1916	100	men	NA
7	1920	100	men	10.8
8	1924	100	men	10.6
9	1928	100	men	10.8
10	1932	100	men	10.3

```
# i 198 more rows
```

```
# i Use `print(n = ...)` to see more rows
```


1.2. 对 tsibble 数据进行处理

此时 Sex 列就失去了意义，因此我们只保留其他三列：

```
olympic_running |> filter(Sex == "men") |> select(-Sex)
```

```
# A tsibble: 208 x 3 [4Y]
```

```
# Key:           Length [7]
```

```
  Year Length  Time  
  <int>  <int> <dbl>
```

1	1896	100	12
2	1900	100	11
3	1904	100	11
4	1908	100	10.8
5	1912	100	10.8
6	1916	100	NA
7	1920	100	10.8
8	1924	100	10.6
9	1928	100	10.8
10	1932	100	10.3

```
# i 198 more rows
```

```
# i Use `print(n = ...) ` to see more rows
```

1.2. 对 tsibble 数据进行处理

新增一列平均速度 $\text{aveSpeed} = \text{Length} / \text{Time}$ ，然后选取 1992 年以后历届奥运会 400 米项目的结果：

```
olympic_running |> filter(Sex == "men") |> select(-Sex) |>  
  mutate(aveSpeed = Length/Time) |>  
  filter(Year >= 1992 & Length == 400)
```

```
# A tsibble: 7 x 4 [4Y]  
# Key:      Length [1]  
   Year Length  Time aveSpeed  
   <int> <int> <dbl>    <dbl>  
1  1992    400  43.5     9.20  
2  1996    400  43.5     9.20  
3  2000    400  43.8     9.12  
4  2004    400  44      9.09  
5  2008    400  43.8     9.14  
6  2012    400  43.9     9.10  
7  2016    400  43.0     9.30
```

1.2. 对 tsibble 数据进行处理

我们也可以将 `olympic_running` 中的 `Length` 的取值看作变量，也就是把每个竞技项目的数据单独放在一列中：

```
olympic_running |> pivot_wider(names_from = Length, values_from = Time)
```

```
# A tsibble: 54 x 9 [4Y]
```

```
# Key:           Sex [2]
```

	Year	Sex	`100`	`200`	`400`	`800`	`1500`	`5000`	`10000`
	<int>	<chr>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>
1	1896	men	12	NA	54.2	131	273.	NA	NA
2	1900	men	11	22.2	49.4	121.	246.	NA	NA
3	1904	men	11	21.6	49.2	116	245.	NA	NA
4	1908	men	10.8	22.6	50	113.	243.	NA	NA
5	1912	men	10.8	21.7	48.2	112.	237.	877.	1881.
6	1916	men	NA	NA	NA	NA	NA	NA	NA
7	1920	men	10.8	22	49.6	113.	242.	896.	1906.
8	1924	men	10.6	21.6	47.6	112.	234.	871.	1823.
9	1928	men	10.8	21.8	47.8	112.	233.	878	1819.
10	1932	men	10.3	21.2	46.2	110.	231.	870	1811.

```
# i 44 more rows
```

```
# i Use `print(n = ...)` to see more rows
```

1.3. 导入外部数据

真实世界中的数据存在于数据库、csv 文件或者 excel 文件中，我们需要把它们导入 R 后，再转换为 tsibble 形式。

例如，https://OTexts.com/fpp3/extrfiles/prison_population.csv 提供了澳大利亚监狱收监人数的季度数据。

Date	State	Gender	Legal	Indigenous	Count
2005-03-01	ACT	Female	Remanded	ATSI	0
2005-03-01	ACT	Female	Remanded	Non-ATSI	2
2005-03-01	ACT	Female	Sentenced	ATSI	0
2005-03-01	ACT	Female	Sentenced	Non-ATSI	5
2005-03-01	ACT	Male	Remanded	ATSI	7
2005-03-01	ACT	Male	Remanded	Non-ATSI	58
⋮	⋮	⋮	⋮	⋮	⋮

1.3. 导入外部数据

`tidyverse` 包含的 `readr` 工具包中提供了导入或识别数据的函数。我们可以用 `read_csv()` 函数直接从网上将数据导入 R。

```
prison <- readr::read_csv("https://OTexts.com/fpp3/extrfiles/prison_
population.csv")
```

```
Rows: 3072 Columns: 6
```

```
— Column specification —————
```

```
Delimiter: ","
```

```
chr   (4): State, Gender, Legal, Indigenous
```

```
dbl   (1): Count
```

```
date  (1): Date
```

```
Use `spec()` to retrieve the full column specification for this data.
```

```
Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

1.3. 导入外部数据

```
prison
```

```
# A tibble: 3,072 × 6
```

	Date	State	Gender	Legal	Indigenous	Count
	<date>	<chr>	<chr>	<chr>	<chr>	<dbl>
1	2005-03-01	ACT	Female	Remanded	ATSI	0
2	2005-03-01	ACT	Female	Remanded	Non-ATSI	2
3	2005-03-01	ACT	Female	Sentenced	ATSI	0
4	2005-03-01	ACT	Female	Sentenced	Non-ATSI	5
5	2005-03-01	ACT	Male	Remanded	ATSI	7
6	2005-03-01	ACT	Male	Remanded	Non-ATSI	58
7	2005-03-01	ACT	Male	Sentenced	ATSI	5
8	2005-03-01	ACT	Male	Sentenced	Non-ATSI	101
9	2005-03-01	NSW	Female	Remanded	ATSI	51
10	2005-03-01	NSW	Female	Remanded	Non-ATSI	131

```
# i 3,062 more rows
```

```
# i Use `print(n = ...) ` to see more rows
```

由此可知，导入后的数据形式为 tibble，我们需要将其变更为 tsibble。

1.3. 导入外部数据

首先需要将保存为日度数据的 Date 更改为季度，然后指定 index 和 key。

```
prison |>
  mutate(Quarter = yearquarter(Date)) |>      # 将 Date 中的日期改为季度
  select(-Date) |>                             # 删除 Date 列
  as_tsibble(
    key = c(State, Gender, Legal, Indigenous), # 指定 key
    index = Quarter                             # 指定 index
  ) -> prison      # 将以上结果重新代入 prison
```

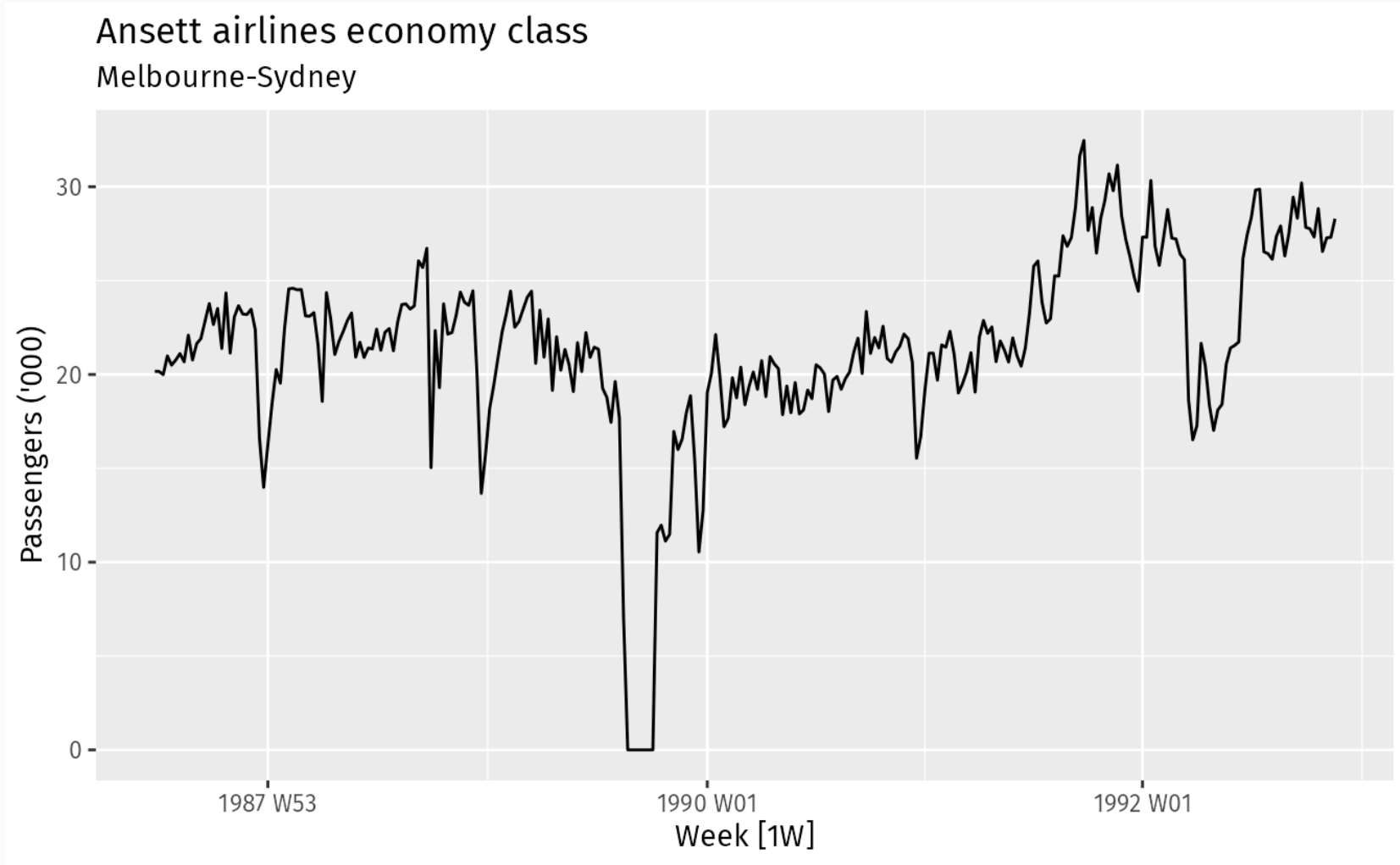
注意最后一行用了向右代入符号 ->，这中写法并不常见，但放在连续几个 |> 后就显得非常自然了。

1.3. 导入外部数据

```
prison      # 显示转换为 tsibble 后的 prison 数据
# A tsibble: 3,072 x 6 [1Q]
# Key:      State, Gender, Legal, Indigenous [64]
   State Gender Legal      Indigenous Count Quarter
   <chr> <chr>  <chr>      <chr>          <dbl>   <qtr>
1 ACT    Female Remanded ATSI          0 2005 Q1
2 ACT    Female Remanded ATSI          1 2005 Q2
3 ACT    Female Remanded ATSI          0 2005 Q3
4 ACT    Female Remanded ATSI          0 2005 Q4
5 ACT    Female Remanded ATSI          1 2006 Q1
6 ACT    Female Remanded ATSI          1 2006 Q2
7 ACT    Female Remanded ATSI          1 2006 Q3
8 ACT    Female Remanded ATSI          0 2006 Q4
9 ACT    Female Remanded ATSI          0 2007 Q1
10 ACT   Female Remanded ATSI          1 2007 Q2
# i 3,062 more rows
# i Use `print(n = ...)` to see more rows
```


2. 图形分析

2.1. 时序图 (time plot)



2.1. 时序图 (time plot)

我们用 `ggplot2` 提供的 `autoplot()` 命令画时序图。顾名思义，该函数可以根据输入的数据性质输出最合适的图表，如果是 `tsibble` 数据，则会输出时序图。

`autoplot()` 的基本语法是 `autoplot(Data, Measure)`，它会自动将 `index` 作为横轴，将指定的 `Measure` 变量作为纵轴，并针对每个 `key` 值组合画一条折线图。如果不指定 `Measure`，它会自动选择第一个数值变量（非 `index` 和 `key` 变量）进行绘图。

2.1. 时序图 (time plot)

`tsibbledata` 中包含的前澳洲安捷航空 (Ansett Airlines) 的周度旅客数量数据。1989 年, 澳大利亚的民航飞行员曾发起抗议活动, 这导致数据中存在 0 旅客记录。

```
ansett
```

```
# A tsibble: 7,407 x 4 [1W]
# Key:      Airports, Class [30]
   Week Airports Class  Passengers
   <week> <chr>      <chr>      <dbl>
1 1989 W28 ADL-PER Business      193
2 1989 W29 ADL-PER Business      254
3 1989 W30 ADL-PER Business      185
4 1989 W31 ADL-PER Business      254
5 1989 W32 ADL-PER Business      191
6 1989 W33 ADL-PER Business      136
7 1989 W34 ADL-PER Business         0
8 1989 W35 ADL-PER Business         0
```

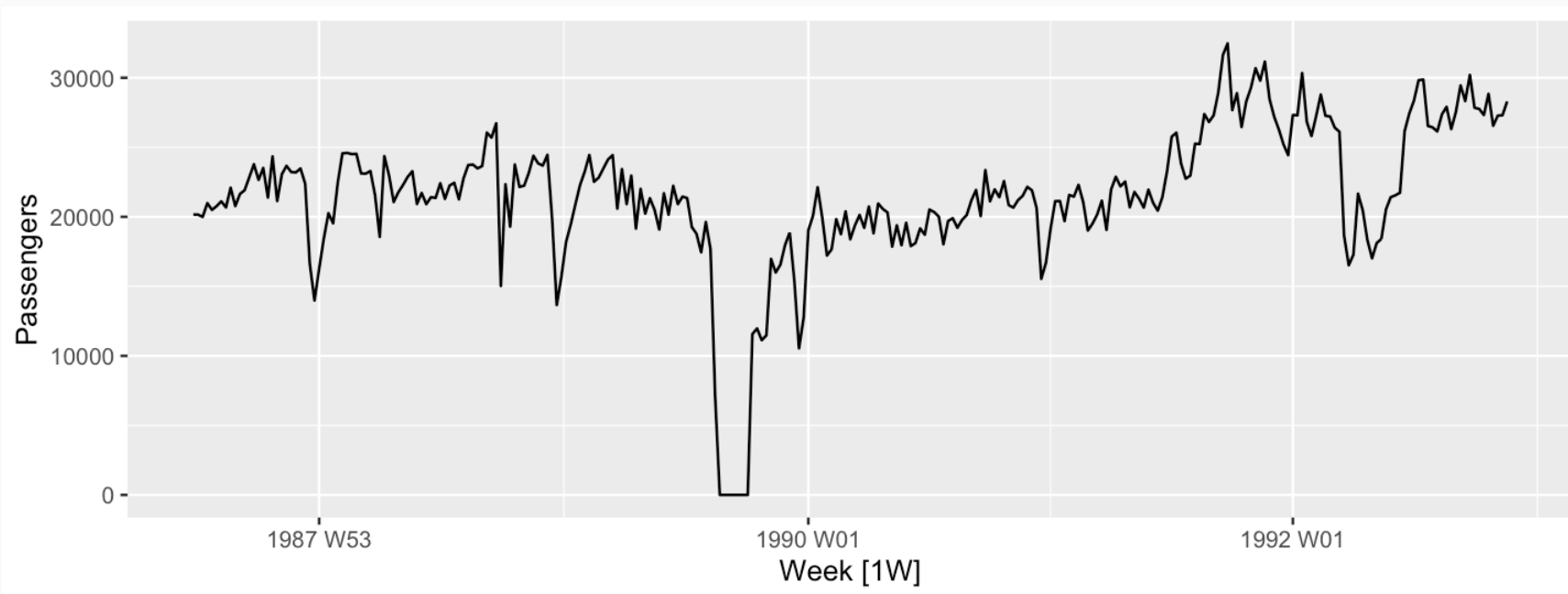
2.1. 时序图 (time plot)

```
ansett |>
```

```
  filter(Airports == "MEL-SYD", Class == "Economy") |>
```

```
  autoplot()
```

Plot variable not specified, automatically selected `vars = Passengers`



2.2. 时间序列特征

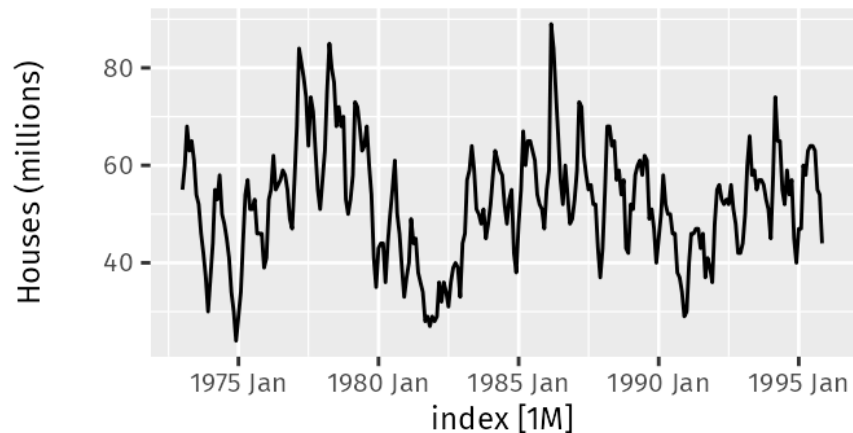
在描述时序图所体现的数据特征时，我们常会用到以下概念：

- **趋势 (trend)**：数据体现出的长期增加或减少。
- **季节性 (seasonality)**：数据中和季节性因素相吻合的变化，例如一年中的固定时期或一周中的固定日期。
- **周期 (cycle)**：没有固定频率的增加和减少，通常和经济周期 (business cycle) 相关，时间跨度一般大于两年。

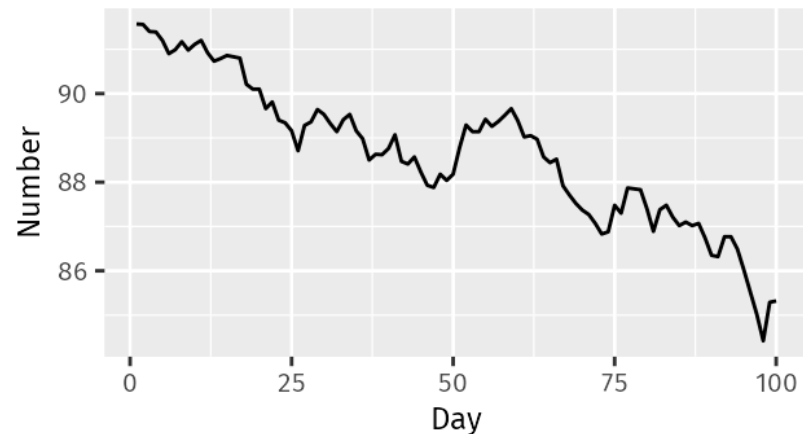
不同模型对这几种特征有不同的解释能力，因此在选择模型之前，我们首先应该判断数据中体现的是哪几种特征的组合。

2.2. 时间序列特征

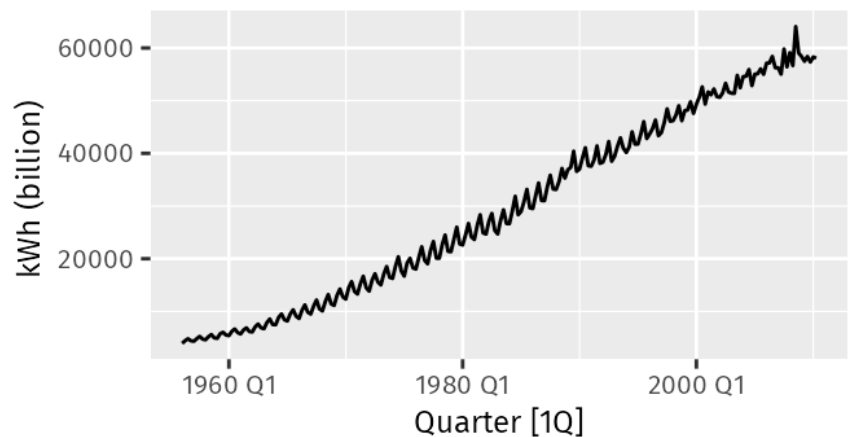
Sales of new one-family houses, USA



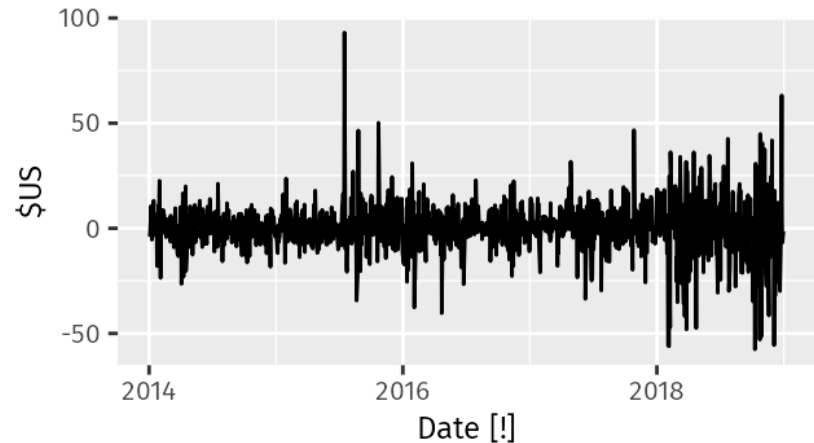
US treasury bill contracts



Australian quarterly electricity production

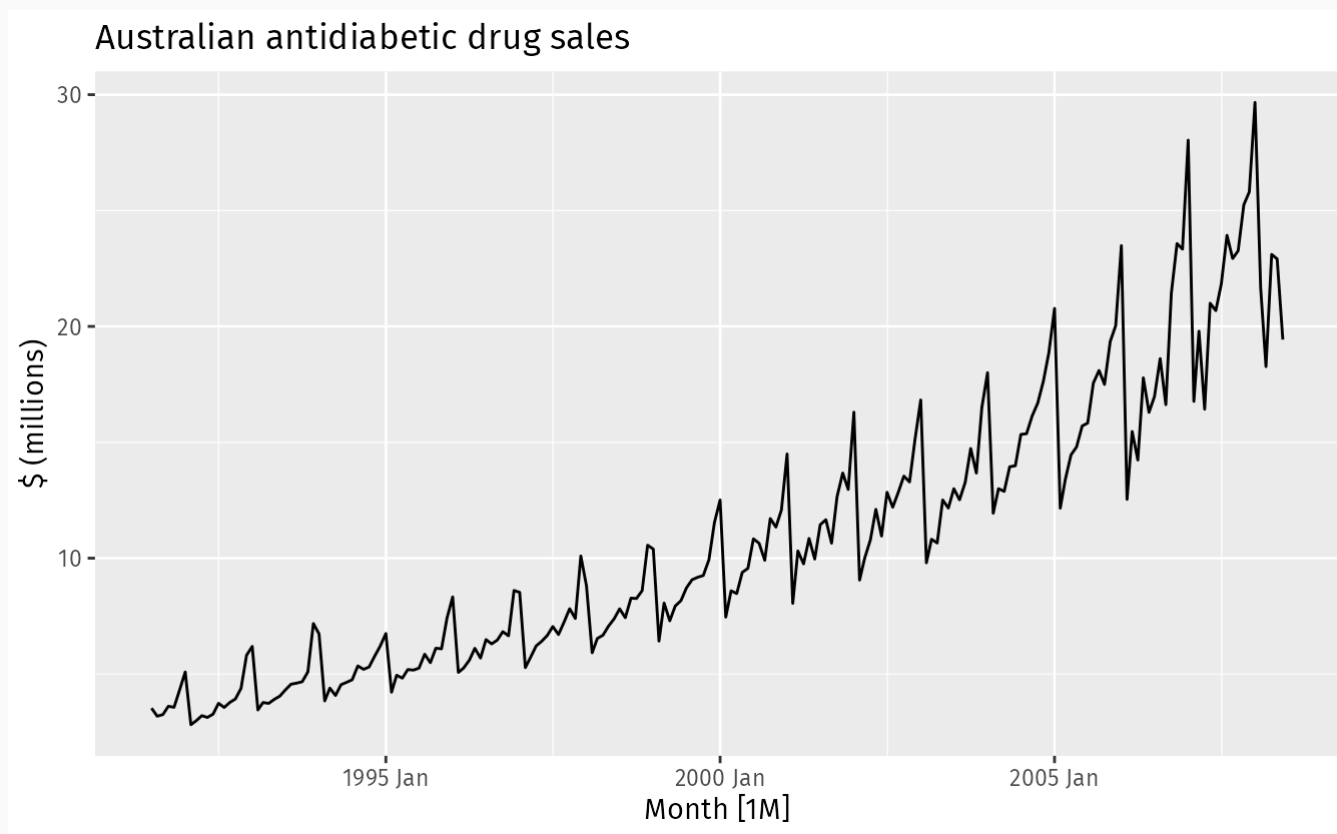


Daily changes in Google closing stock price



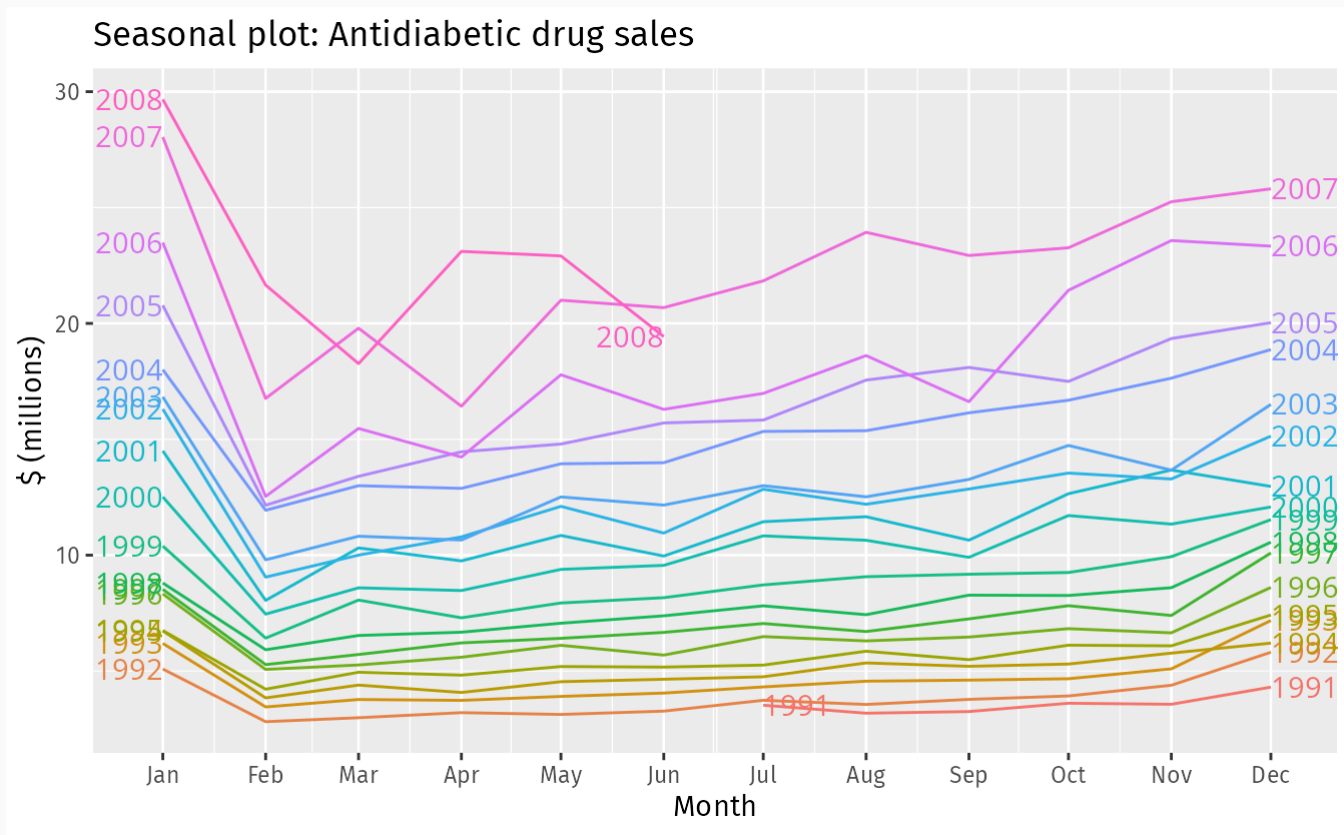
2.3. 通过季节图展现季节性特征

下图显示了澳大利亚糖尿病药物月度销售额，可以看到明显的季节性特征



2.3. 通过季节图展现季节性特征

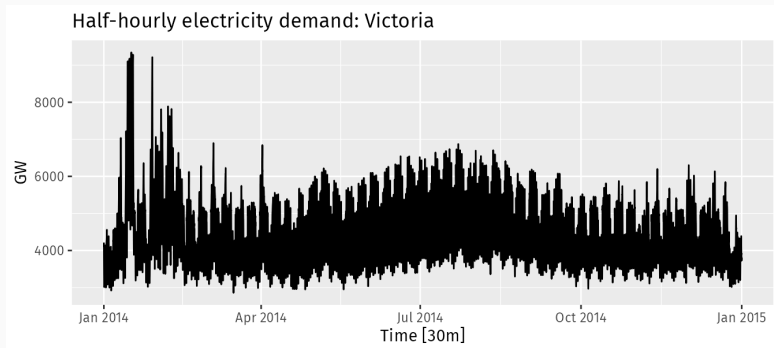
下面的季节图（seasonal plot）将横轴取为 12 个月，并将每年的数据堆叠绘制。这样可以更清晰地观察季节性特征及其整体变化趋势。（更多信息参考第 2.4 – 2.5 节）



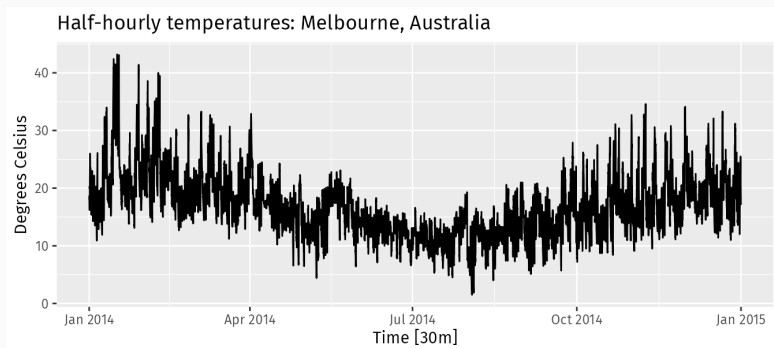
3. 散点图和相关性

3.1. 用散点图展示不同序列间的关系

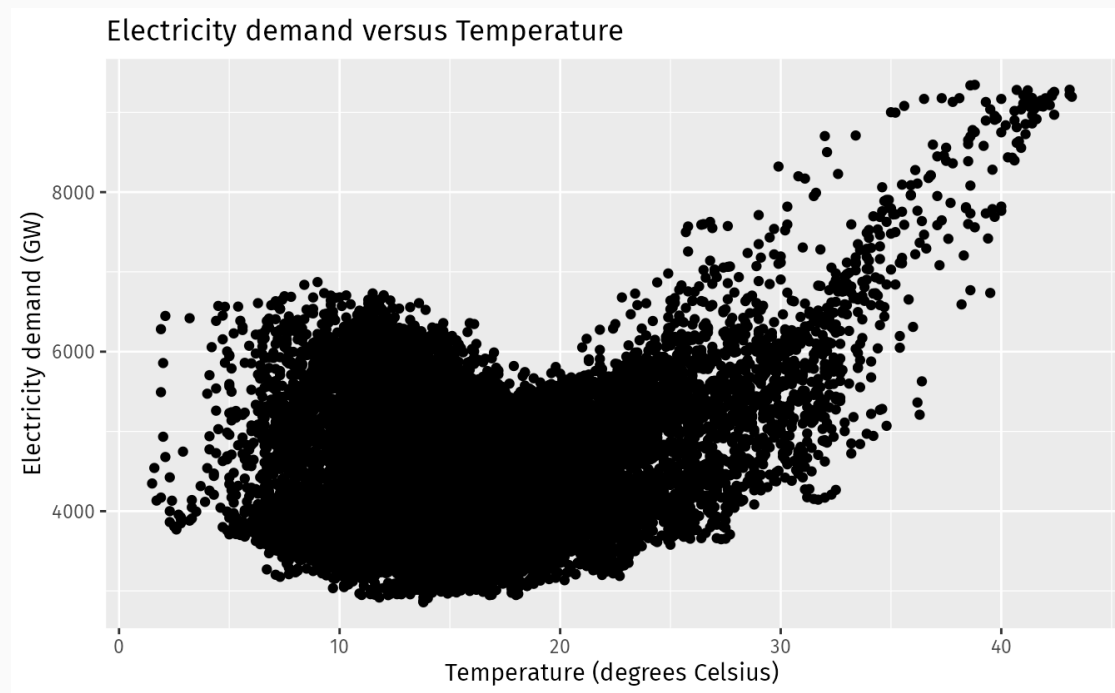
电力需求



气温



气温和电力需求间的关系



3.2. 相关系数

相关系数 (correlation coefficient)

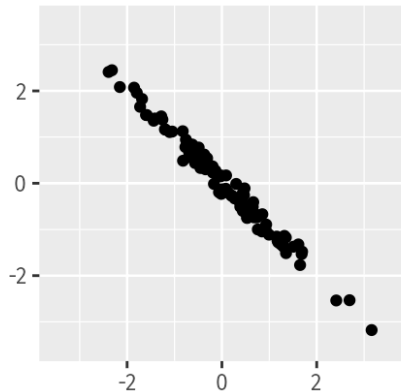
$$r = \frac{\sum_t (x_t - \bar{x})(y_t - \bar{y})}{\sqrt{\sum_t (x_t - \bar{x})^2} \sqrt{\sum_t (y_t - \bar{y})^2}}$$

- $-1 < r < 1$
- $|r|$ 的大小表示**线性相关**的强弱

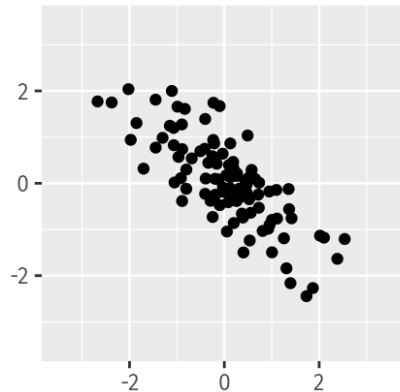
电力需求和气温间的相关系数为 0.28。两者间的非线性关系明显强于线性相关性。

3.2. 相关系数

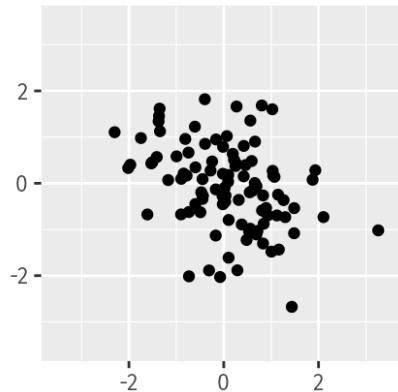
Correlation = -0.99



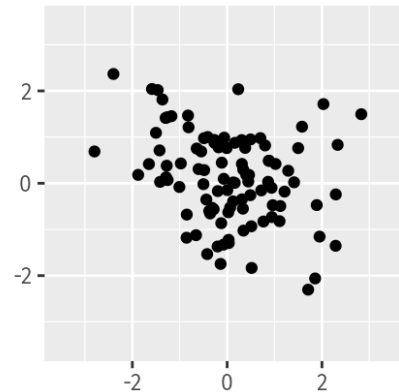
Correlation = -0.75



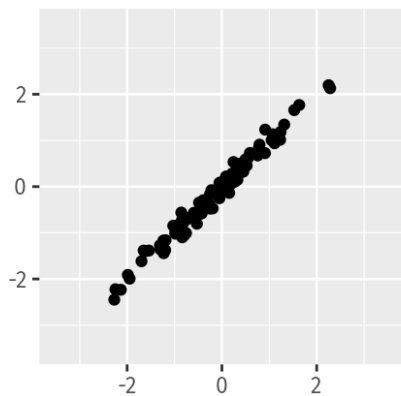
Correlation = -0.50



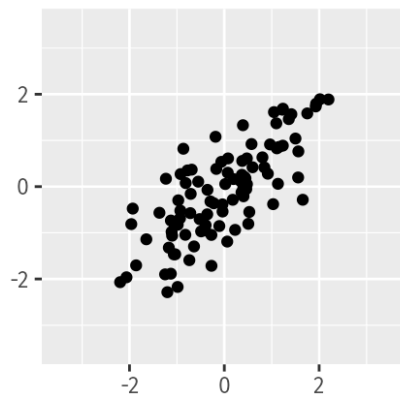
Correlation = -0.25



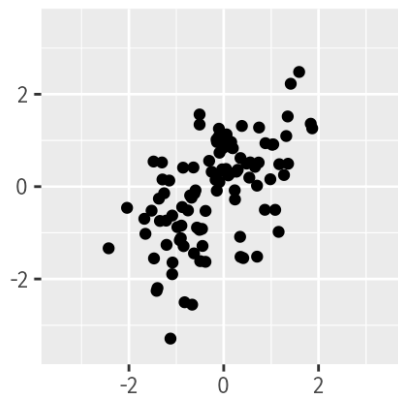
Correlation = 0.99



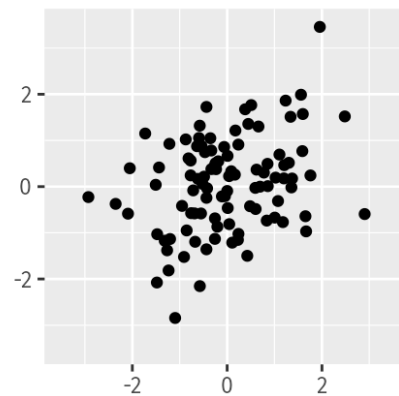
Correlation = 0.75



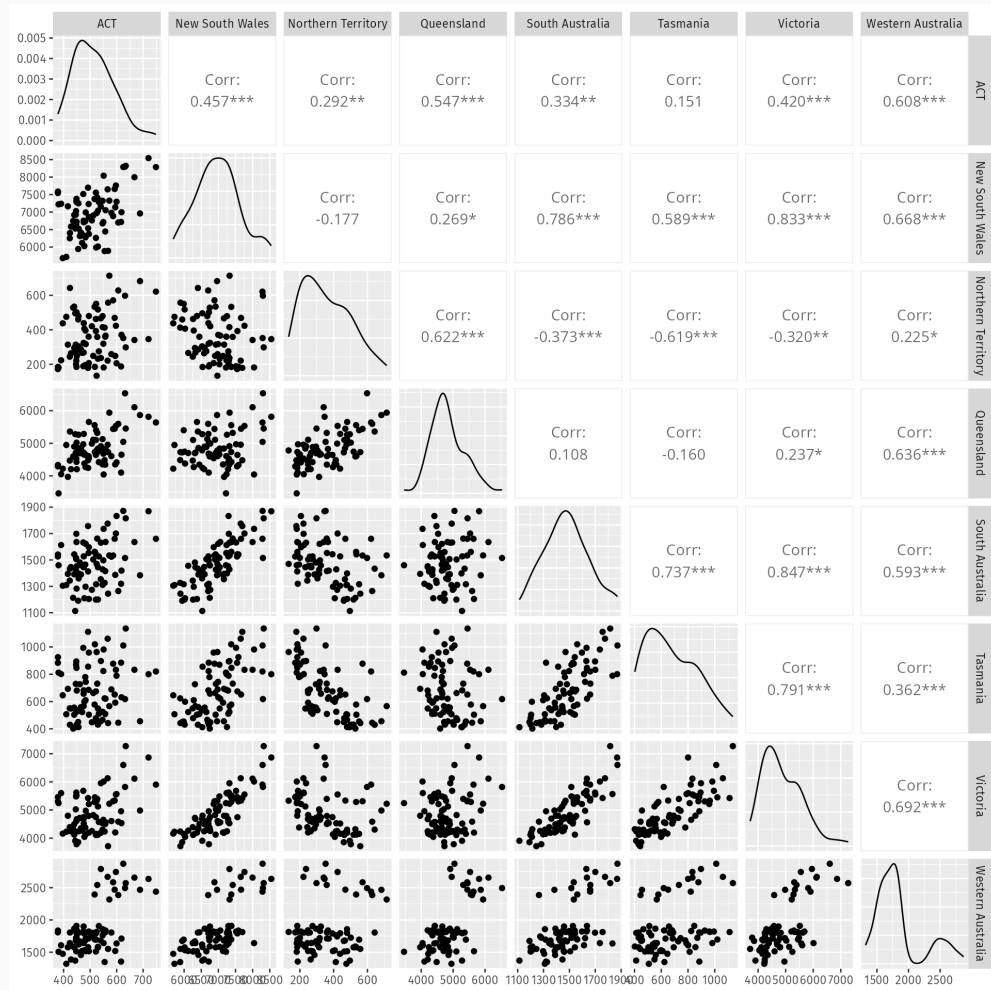
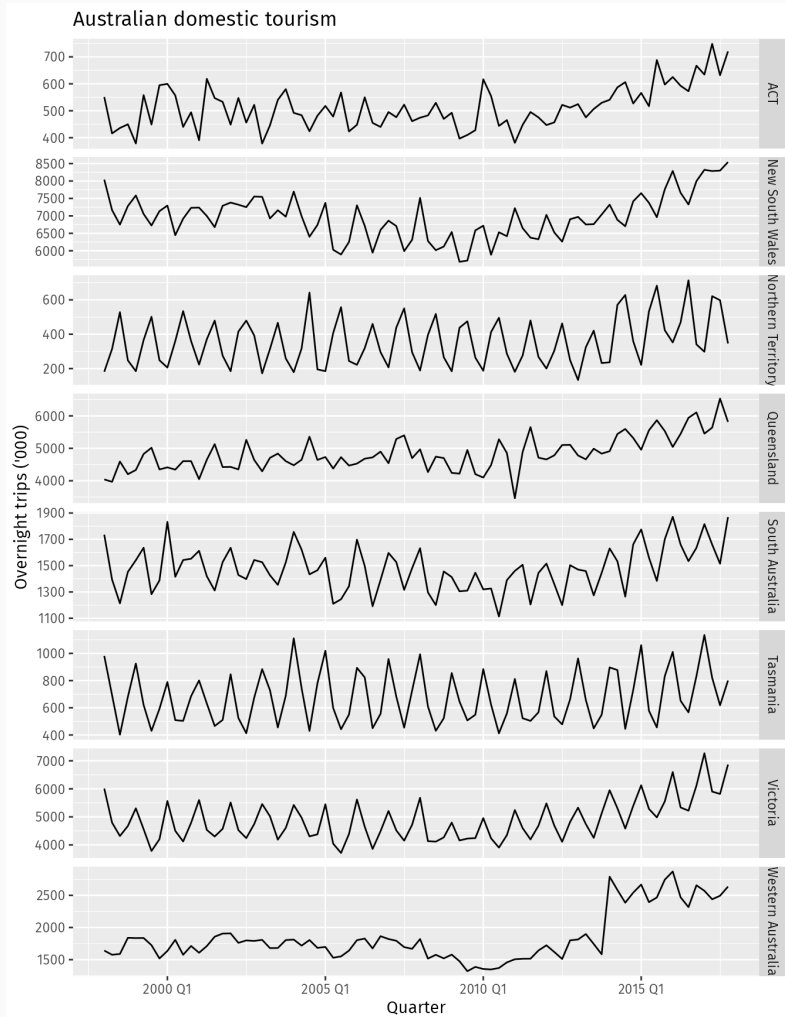
Correlation = 0.50



Correlation = 0.25



3.2. 相关系数



3.3. 滞后图 (lag plot)

滞后 (lag) 是时间序列分析中的一个重要概念。例如 2 月 10 日的一日滞后是 2 月 9 日，两日滞后是 2 月 8 日。

当我们讨论一个序列 y_t 时，我们指的是观测值的集合

$$\{y_t \mid t = 0, 1, \dots, T\}$$

此时， y_t 的 k 期滞后序列 y_{t-k} 指的则是

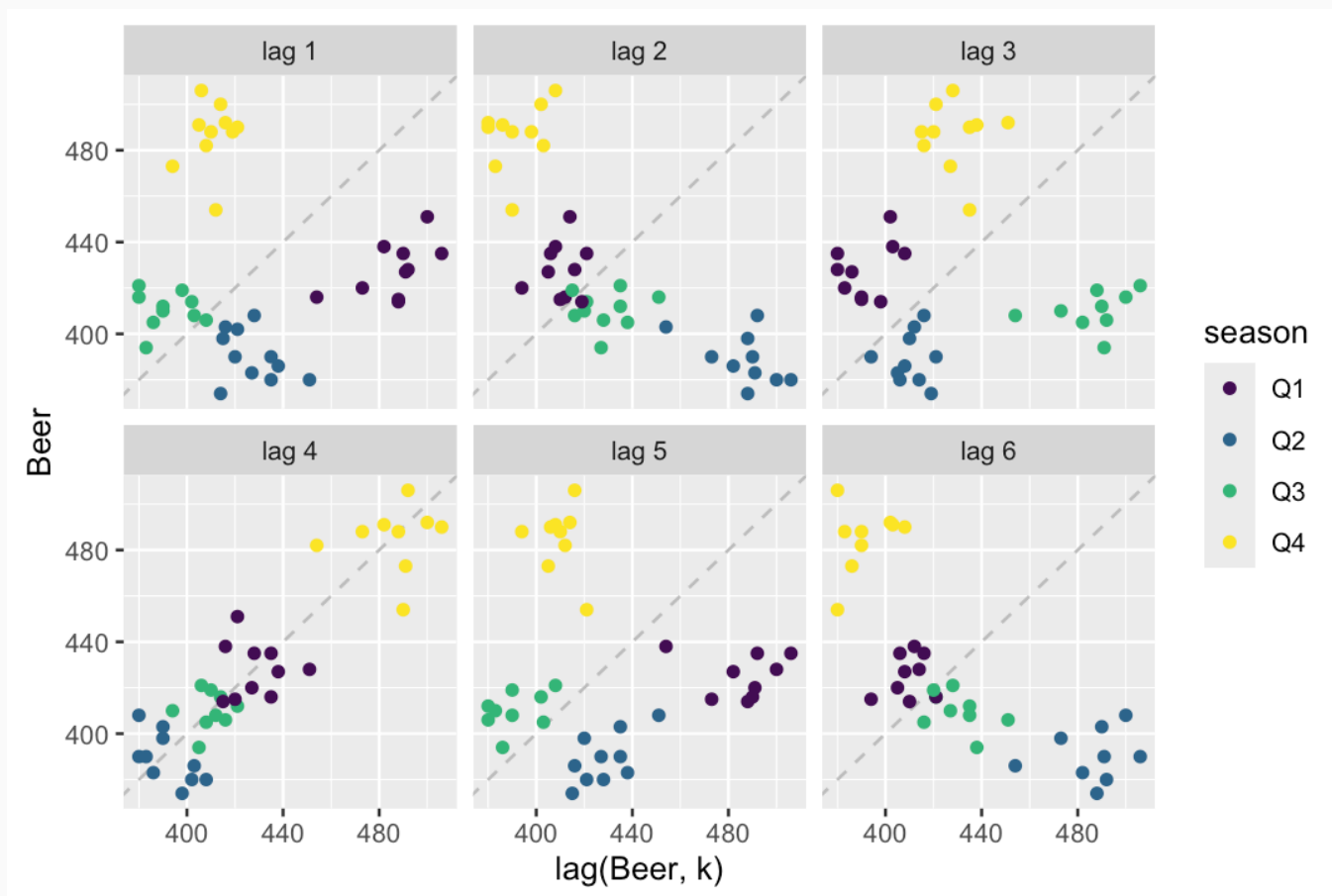
$$\{y_s \mid t = -k, 1 - k, \dots, T - k\}$$

其中 y_{-k}, \dots, y_{-1} 的值往往是缺失的。

因此，我们可以比较序列 y_t 和它自身的滞后序列 y_{t-k} 之间的关系。用二者绘制的散点图称为**滞后图 (lag plot)**，用二者计算的相关系数称为**自相关系数 (autocorrelation coefficient)**。

3.3. 滞后图 (lag plot)

用 `feasts` 包提供的函数 `gg_lag()` 可以绘制滞后图 (详见第 2.7 节)。



3.4. 自相关系数

序列 y_t 和 y_{t-k} 间的**自相关系数** (autocorrelation coefficient) r_k 定义为

$$r_k = \frac{\sum_{t=k+1}^T (y_t - \bar{y})(y_{t-k} - \bar{y})}{\sum_{t=1}^T (y_t - \bar{y})^2}$$

也称为**自相关函数** (autocorrelation function, ACF) 。

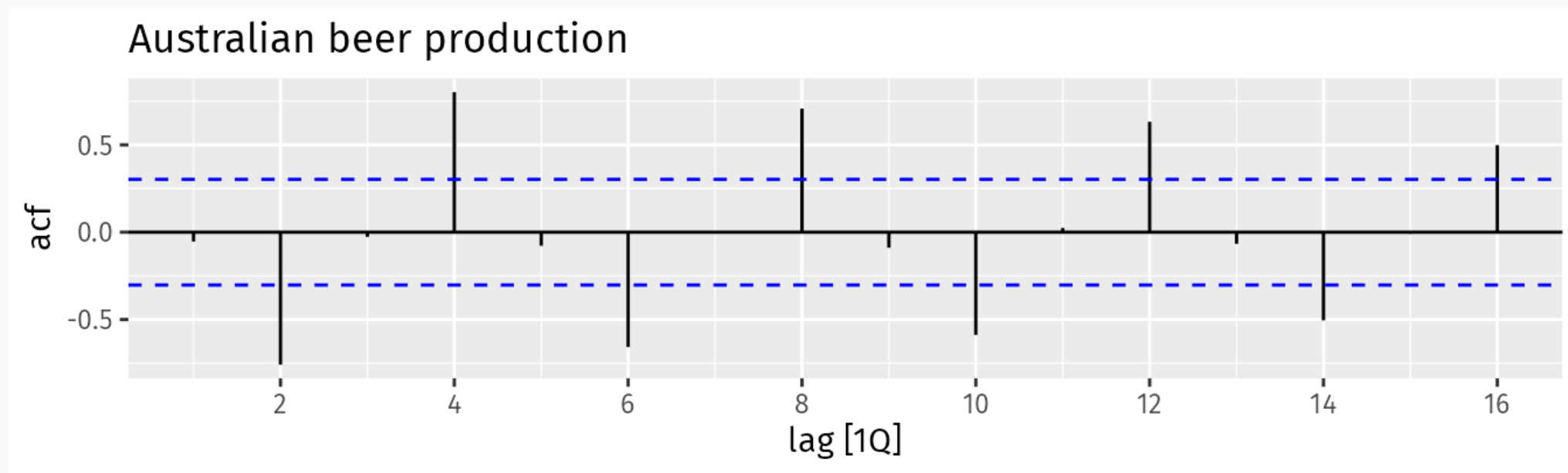
`feasts` 包中的 `ACF()` 函数 (注意大小写) 提供了计算自相关系数的功能。

```
aus_production |> filter(year(Quarter) >= 2000) |> ACF(Beer, lag_max = 3)
# A tsibble: 3 x 2 [1Q]
      lag      acf
  <cf_lag> <dbl>
1      1Q -0.0530
2      2Q -0.758
3      3Q -0.0262
```

3.4. 自相关系数

用多个自相关系数 ($k = 1, 2, \dots, K$) 绘制的图表称为**自相关图 (correlogram)**。

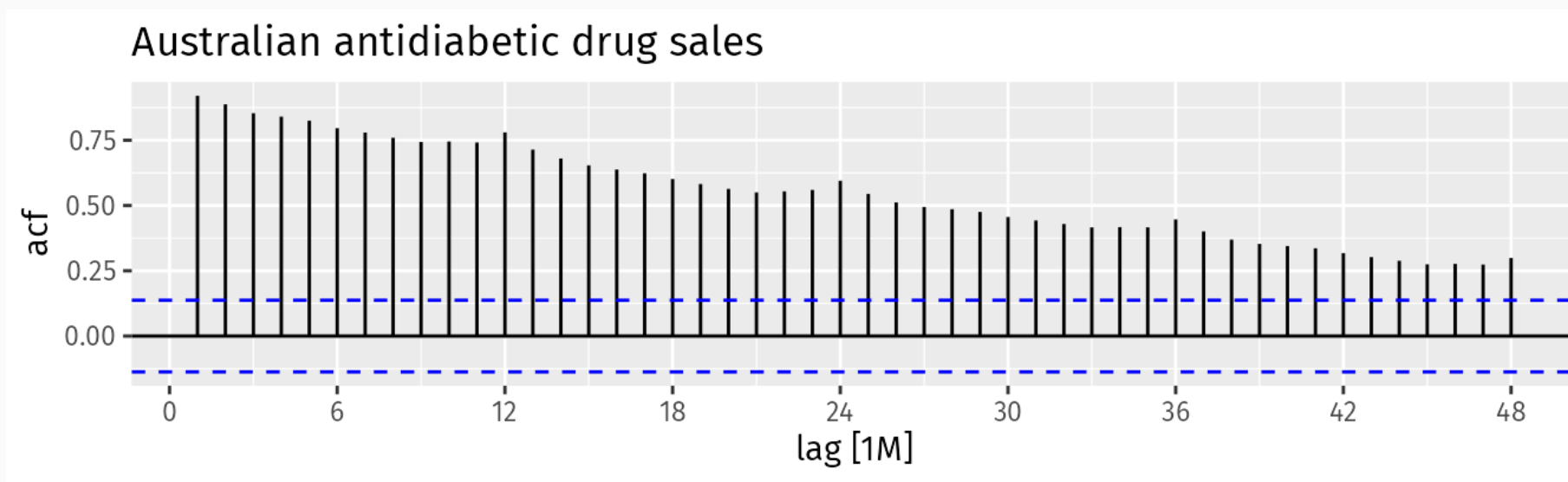
```
aus_production |>  
  filter(year(Quarter) >= 2000) |> ACF(Beer) |>  
  autoplot() + labs(title="Australian beer production")
```



3.4. 自相关系数

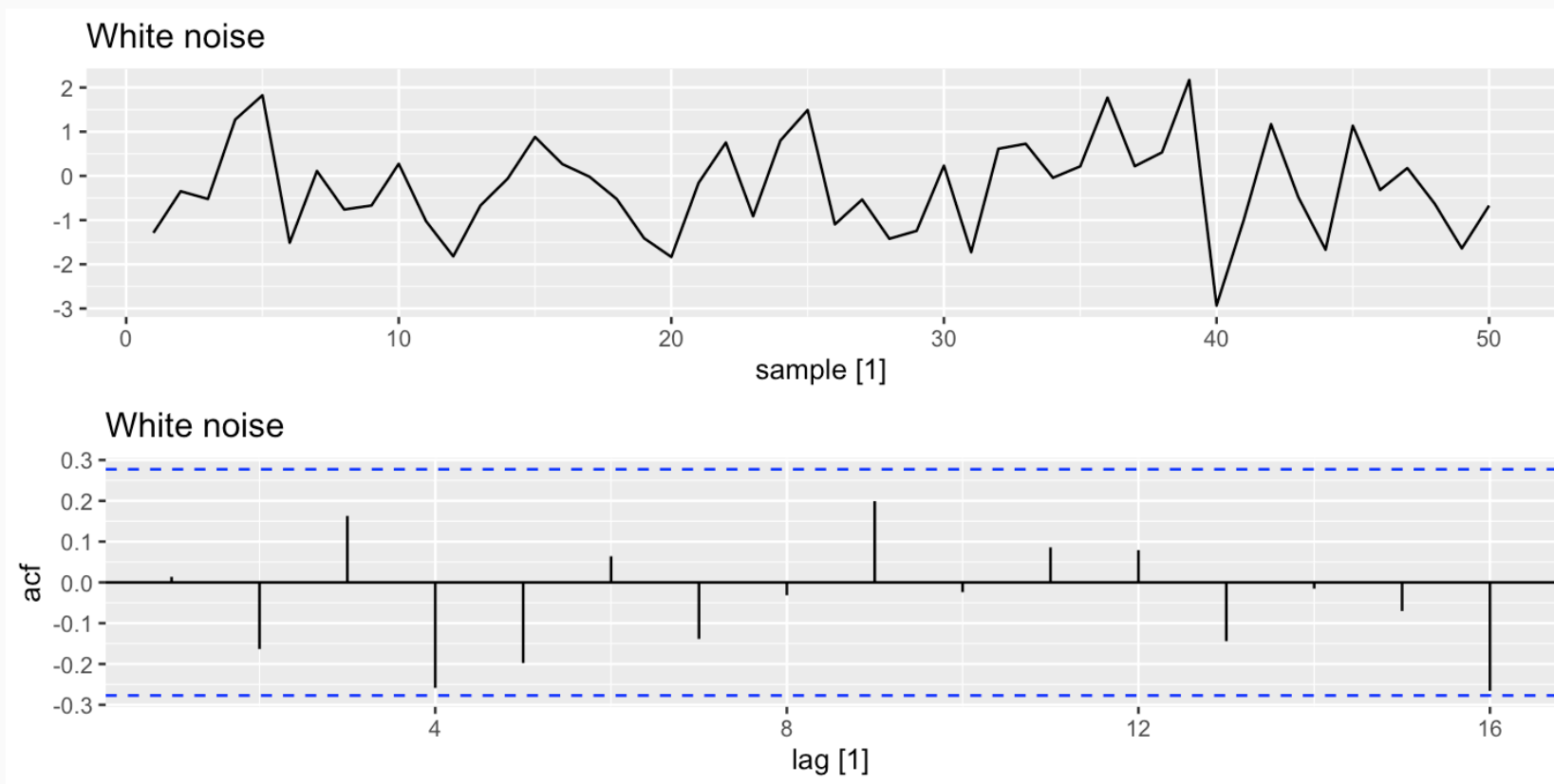
- 当序列存在趋势时，自相关系数通常为正且慢慢减小。
- 当序列存在季节性时，滞后期为季节周期的整数倍时，自相关系数会大于其他滞后期。

下图中的数据是典型的趋势和季节性共存的序列，季节周期为 12 期。



3.5. 白噪声 (white noise)

没有自相关的序列称为**白噪声 (white noise)**。



4. 课后练习

4. 课后练习

- 学习教科书第 2 章 (Time Series Graphics) 中的内容，并尝试在自己的电脑上复现书中的结果。
- 回答下列问题：
 1. 列出 `olympic_running` 数据集共包含了哪些跑步项目。
 2. `feasts` 包中的 `ACF()` 函数计算的自相关系数和根据定义计算的结果是否一致？请举例验证。