预测：方法与实践 2 时间序列图形

## 第二章 时间序列图形

对于任何数据分析工作而言，其首要任务是数据可视化。图示化数据可以清晰地展现数据的特征，包括数据的形态、异常值、随时间变化情况以及变量间的相互关系。我们在预测时应尽可能地将图中显示的特征纳入考虑。正如数据类型决定使用什么预测方法一样，数据类型也决定了使用什么图形来展示数据。

在画图之前，首先我们应该在R中设置我们的时间序列数据。

library(fpp2)

## Registered S3 method overwritten by 'quantmod':  
## method from  
## as.zoo.data.frame zoo

## ── Attaching packages ────────────────────────────────────────────── fpp2 2.4 ──

## ✔ ggplot2 3.3.6 ✔ fma 2.4   
## ✔ forecast 8.16 ✔ expsmooth 2.3

##

### 2.1 ts 对象

时间序列是一组按照时间发生先后顺序进行排列，并且包含一些信息的数据点序列。在R中，这些信息可以被储存在ts对象中。

假设我们有某个变量在过去几年中每年的观测值：

| 年份 | 观测值 |
| --- | --- |
| 2012 | 123 |
| 2013 | 39 |
| 2014 | 78 |
| 2015 | 52 |
| 2016 | 110 |

我们可以采用ts()函数将数据转化为ts类型：

y <- ts(c(123, 39, 78, 52, 110), start = 2012)

假如我们只有年度数据，即每年只有一个观测值，此时只需要给出起始年份（或者截止年份）。

假如一个变量的观测频率大于每年一次，我们可以通过设置frequency参数来设置频率。例如，如果我们的月度数据已经被储存在了数值型向量z中，那么我们可以用如下的方式将其转换为时间序列：

# y <- ts(z, start=2003, frequency=12)

在本书中，几乎所有的数据都已被储存在ts对象中。但是，如果需要处理自己收集的数据，首先应使用ts()函数将原始数据转化为ts对象。

#### 时间序列的频率

“频率”是季节模式重复之前的观测值个数1。在R中使用ts()函数时，频率有以下值可供选择。

| 年份 | 观测值 |
| --- | --- |
| 2012 | 123 |
| 2013 | 39 |
| 2014 | 78 |
| 2015 | 52 |
| 2016 | 110 |

实际上，一年并不是精准的52周。由于每四年是一个闰年，平均来看每年有 365.25/7 = 52.18 周。但大多数使用ts对象的函数都需要频率为整数。

如果观测频率大于每周一次，可以采用多种方法来处理频率。例如，日观测数据可能具有周季节性（frequency=7）或者具有年度季节性（frequency=365.25）。类似地，一个每分钟观测一次的数据可能具有时季节性（frequency=60），可能是日季节性（frequency=24x60=1440），还可能是周季节性（frequency=24x60x7=10080），甚至可能具有年度周期性（frequency=24x60x365.25=525960）。在我们处理时间序列之前，确定其频率至关重要。

在第 11 章中，我们将会处理多季节性时间序列数据类型，而不必仅局限于其中某一个频率。

这与物理学中或傅立叶分析中的频率定义相反，这里的频率在那里被称为“周期”。↩︎

### 2.2 时间图

对于时间序列数据而言，我们从最简单的时间图开始。时间图是用将观测值与观测时间点作图，散点之间用直线连接。例如图2.1表示在澳大利亚两个最大的城市之间，Ansett航空公司的每周客流量。

autoplot(melsyd[, "Economy.Class"]) +  
 ggtitle("墨尔本 - 悉尼经济舱乘客客流量") +  
 xlab("年份") +  
 ylab("千") +  
 theme(text = element\_text()) +  
 theme(plot.title = element\_text(hjust = 0.5))

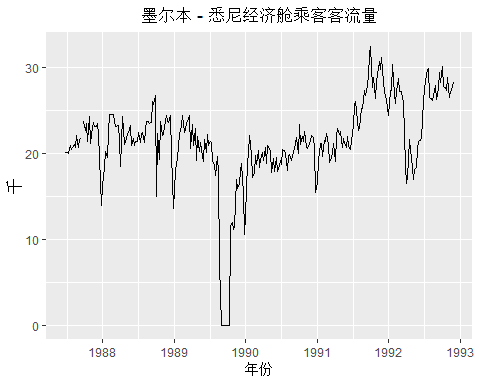


图 2.1: Ansett航空公司经济舱的客流量

之后我们会频繁使用autoplot()函数，它可以自动画出你传给其第一个参数的内容。在本例中，它将melsyd[,"Economy.Class"]识别为一个时间序列，进而自动生成时间图。

该时间图直观地展现出数据具有的一些特征：

由于1989年当地的工业纠纷，当年的客流量为0。 在1992年中，由于一部分经济舱被商务舱取代，导致客流量大幅减少。 1991年下半年客流量大幅上升。 由于假日效应，在每年年初，客流量都会有一定幅度的下降。 这是序列存在长期波动，在1987年向上波动，在1988年向下波动，而在1990年和1991年又再次向上波动。 在某些时期存在缺失值。 对该数据进行预测建模时，需要考虑上述所有的特征，以便有效预测未来的客流量。

如图 2.2 所示，是一个简单的例子。

autoplot(a10) +  
 ggtitle("澳大利亚降糖药物的销量") +  
 ylab("百万（美元）") +  
 xlab("年份") +  
 theme(text = element\_text()) +  
 theme(plot.title = element\_text(hjust = 0.5))

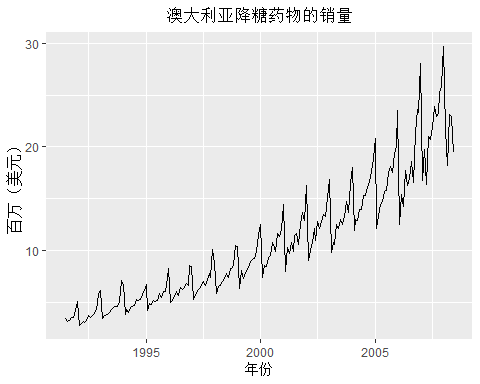


图 2.2: 澳大利亚降糖药物的月销量

显然，图示的时间序列具有明显增长的趋势。同时，在上升趋势中伴随着明显的季节性。在每年年底，由于政府补贴计划，使得降糖药品更便宜，所以人们倾向于在年底囤积药物，从而导致年初的销售额大幅下降。因此，当我们对降糖药物的销量进行预测时，需同时考虑其趋势和季节性因素。

### 2.3 时间序列模式

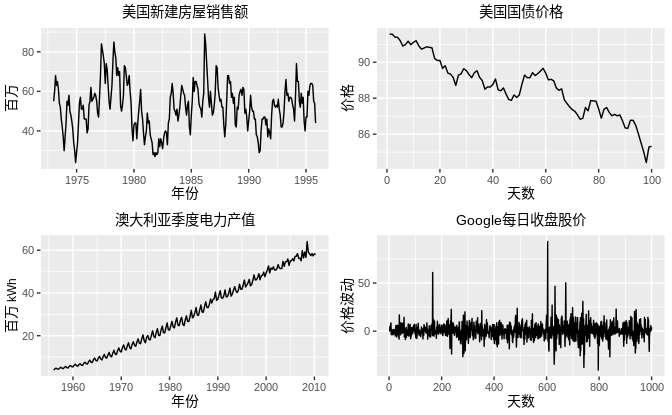
我们通常使用例如“趋势”、“季节性”等词语描述时间序列。在深入研究时间序列模式时，应该更精确的定义这些词语。

#### 趋势

当一个时间序列数据长期增长或者长期下降时，表示该序列有 趋势 。在某些场合，趋势代表着“转换方向”。例如从增长的趋势转换为下降趋势。在图 2.2 中，明显存在一个增长的趋势。 #### 季节性 当时间序列中的数据受到季节性因素（例如一年的时间或者一周的时间）的影响时，表示该序列具有 季节性 。季节性总是一个已知并且固定的频率。由于抗糖尿病药物的成本在年底时会有变化，导致上述抗糖尿药物的月销售额存在季节性。 #### 周期性 当时间序列数据存在不固定频率的上升和下降时，表示该序列有 周期性 。这些波动经常由经济活动引起，并且与“商业周期”有关。周期波动通常至少持续两年。 许多初学者都不能很好的区分季节性和周期，然而这两个概念是完全不同的。当数据的波动是无规律时，表示序列存在周期性；如果波动的频率不变并且与固定长度的时间段有关，表示序列存在季节性。一般而言，周期的长度较长，并且周期的波动幅度也更大。

许多时间序列同时包含趋势、季节性以及周期性。当我们选择预测方法时，首先应该分析时间序列数据所具备的特征，然后再选择合适的预测方法抓取特征。

以下四个示例分别是上述三个特征的不同组合。



时间序列不同模式的四个示例

图 2.3: 时间序列不同模式的四个示例

美国新建房屋销售额（左上）表现出强烈的年度季节性，以及周期为6~10年的周期性。但是数据并没有表现出明显的趋势。 美国国债价格（右上）表示1981年美国国债在芝加哥市场连续100个交易日的价格。可以看出，该序列并没有季节性，但是有明显下降的趋势。假如我们拥有该序列更多的观测数据，我们可以看到这个下降的趋势是一个长期循环的一部分。但是现在我们只有连续100天的数据，它表现出下降的趋势。 澳大利亚月度电力产值数据（左下）明显表现出向上增长的趋势，以及强季节性。但是并不存在周期性。 Google收盘股价格（右下）的价格波动没有趋势，季节性和周期性。随机波动没有良好的形态特性，不能很好地预测。

### 2.4 季节图

季节图和时间序列图很相似，不同之处是季节图是针对观察数据的“季节性”绘制的。下面的例子是降糖药物的销售情况。

ggseasonplot(a10, year.labels = TRUE, year.labels.left = TRUE) +  
 xlab("月份") +  
 ylab("百万（美元）") +  
 ggtitle("季节图：降糖药物销量") +  
 theme(text = element\_text()) +  
 theme(plot.title = element\_text(hjust = 0.5))

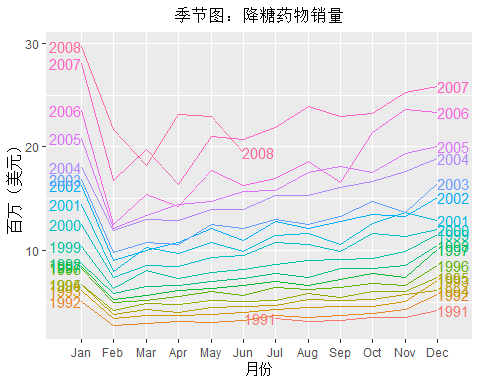


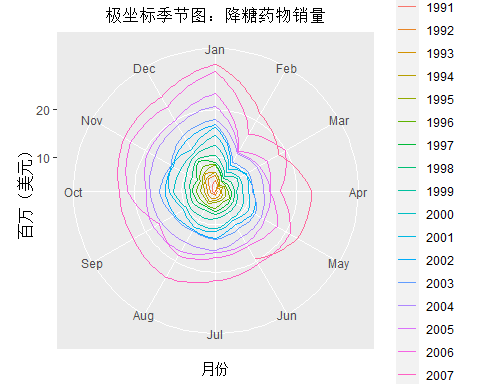
图 2.4: 澳大利亚月度降糖病药物销售量的季节图

在早些年，数据形态基本相同，但是近些年数据存在相互堆叠的情况。季节图可以很清晰的显示季节形态，这对识别数据形态是否发生变化非常有效。

在本例中，在每年一月份降糖药物的销量都会大幅下降。实际上，患者会在12月下旬大量购买降糖药物，但是这部分销量会在一两周后才向政府登记。从上图还可以看出，2008年3月销量大幅下降（其他年份2月份至3月份的销量增加）。2008年6月份销量较少可能是由于销量数据收集不完整导致。

季节图中可以将直角坐标转换为极坐标。设置参数polar=TRUE即可转化为极坐标，如下图所示。

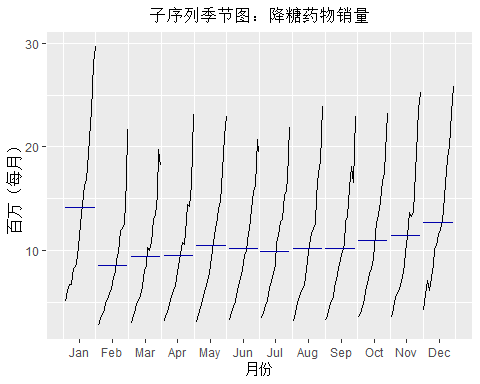
ggseasonplot(a10, polar = TRUE) +  
 xlab("月份") +  
 ylab("百万（美元）") +  
 ggtitle("极坐标季节图：降糖药物销量") +  
 theme(text = element\_text()) +  
 theme(plot.title = element\_text(hjust = 0.5))



### 2.5 子系列季节图

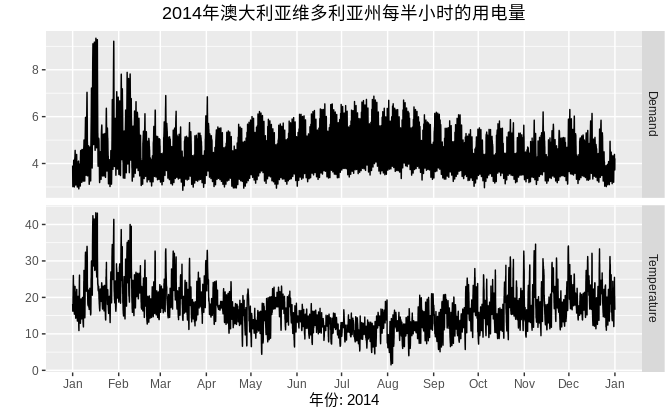
如下图所示，我们可以通过ggsubseriesplot()函数着重描绘在相同月份数据的同比变化情况。

ggsubseriesplot(a10) +  
 xlab("月份") +  
 ylab("百万（每月）") +  
 ggtitle("子序列季节图：降糖药物销量") +  
 theme(text = element\_text()) +  
 theme(plot.title = element\_text(hjust = 0.5))

 position the caret at any line or the code chunk, then click “+”.

### 2.6 散点图

在此之前，我们所讨论的内容都是单个时间序列的可视化。此外，多个时间序列的可视化也是非常有用的。 图 2.7 分别展示了两个时间序列：2014年澳大利亚维多利亚州每半小时的用电量（以千兆瓦为单位）和温度（以摄氏度为单位）。温度选取的是维多利亚州最大城市墨尔本的当地温度，而用电量是整个维多利亚州的用电量。



2014年澳大利亚维多利亚州每半小时的用电量和温度

图 2.7: 2014年澳大利亚维多利亚州每半小时的用电量和温度

我们可以在一张图上绘制两个时间序列的散点图来研究用电量和温度之间的关系。

qplot(Temperature, Demand, data = as.data.frame(elecdemand)) +  
 ylab("用电量 (千兆瓦)") +  
 xlab("温度 (摄氏度)") +  
 theme(text = element\_text()) +  
 theme(plot.title = element\_text(hjust = 0.5))

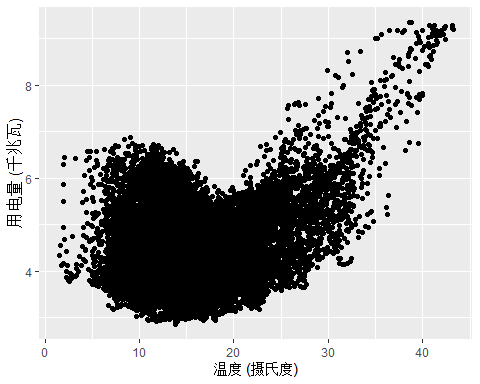


图 2.8: 2014年澳大利亚维多利亚州每半小时的用电量对温度的散点图

这个散点图可以很好的帮助我们理解变量之间的相互关系。从图中我们可以看出，当温度很高时，人们会大量的使用空调进行降温，进而导致用电量随之增加；当温度很低时，人们会使用空调取暖，也会使得用电量一定程度上增加。

#### 相关性

我们经常会用 相关系数 衡量两个两个变量之间的相关强度。假如已知两个变量x和y，那么它们之间的相关系数为

r的值始终在-1到1之间。当两个变量完全负相关时，r值为-1；当两个变量完全正相关时，r为1.图 2.9 分别展示了不同相关强度的例子。

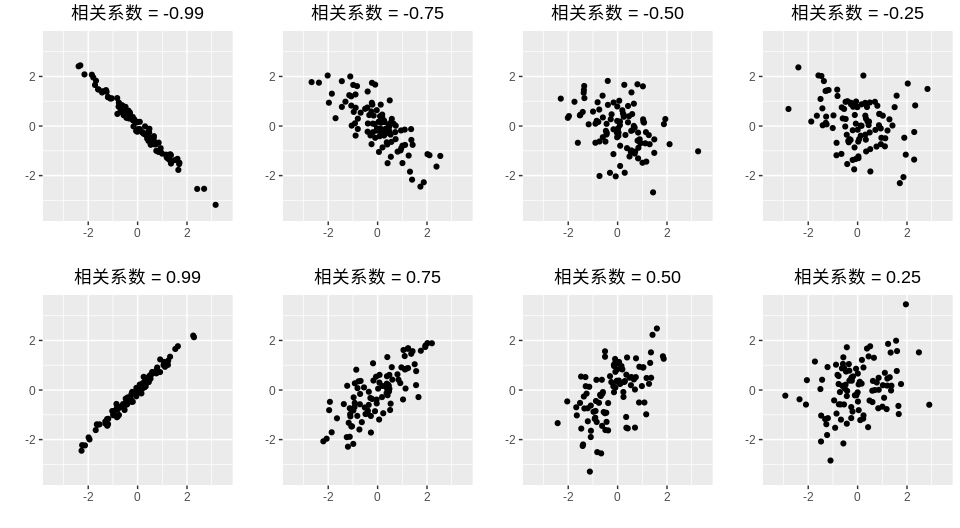


图 2.9: 不同相关强度的例子

图 2.9: 不同相关强度的例子

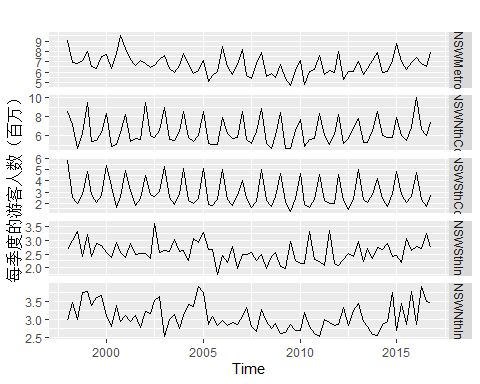
需要注意的是，相关系数仅仅衡量了变量之间的线性关系，并且有时会导致错误的结果。例如，在图 2.10中，所有例子的相关系数均为0.82，但是它们有着完全不同的形态。 这表明，在分析变量之间关系时，不仅要看相关系数值，而且要关注生成的图形。

所有示例的相关系数均为0.82。数据来 FJ Anscombe (1973) Graphs in statistical analysis. American Statistician, 27, 17–21. 图 2.10: 所有示例的相关系数均为0.82。数据来 FJ Anscombe (1973) Graphs in statistical analysis. American Statistician, 27, 17–21. 在图 2.8中，用电量和温度之间的相关系数仅为0.2798，但并不代表用电量和温度之间存在很强的非线性关系。

#### 散点图矩阵

当所分析的数据有多个变量时，将每个变量与其他变量进行比较也很有意义。如图2.11所示，表示澳大利亚新南威尔士五个地区的季度游客人数。

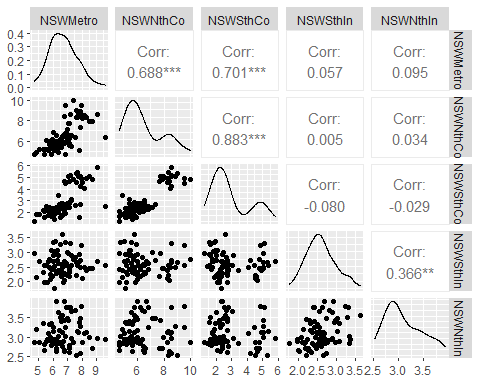
autoplot(visnights[, 1:5], facets = TRUE) +  
 ylab("每季度的游客人数（百万）") +  
 theme(text = element\_text()) +  
 theme(plot.title = element\_text(hjust = 0.5))



如图 2.12所示，我们可以绘制出它们的散点图矩阵。

# todo 不懂  
visnights[, 1:5] %>%  
 as.data.frame() %>%  
 GGally::ggpairs()

## Registered S3 method overwritten by 'GGally':  
## method from   
## +.gg ggplot2



#> Registered S3 method overwritten by 'GGally':  
#> method from  
#> +.gg ggplot2

图 2.12: 澳大利亚新南威尔士五个地区的季度游客人数的散点图矩阵

对于图中的每一块区域，其行变量是垂直轴行的变量，列变量是水平轴的变量。有许多设置可以控制生成的图形的形态。在默认设置中，相关系数在图的右上方显示，散点图在左下方显示，对角线上是密度曲线。

我们可以通过散点图矩阵快速查看所有变量之间的相关关系。在本例中，由图中第二列数据可知，新南威尔士州北部海岸游客与新南威尔士南部海岸游客之间存在强烈的正关系，而新南威尔士州北部海岸的游客与新南威尔士内陆游客之间几乎没有相关关系。同时，我们可以通过散点图矩阵检测到异常值。由于2000年悉尼奥运会，新南威尔士大都会地区存在异常大的客流量。

### 2.7 滞后图

图 2.13是澳大利亚每季度啤酒产量的散点图，横轴表示时间序列的滞后阶数。各图分别显示了不同 k值下和的散点图。

# todo 不懂  
beer2 <- window(ausbeer, start = 1992)  
gglagplot(beer2)

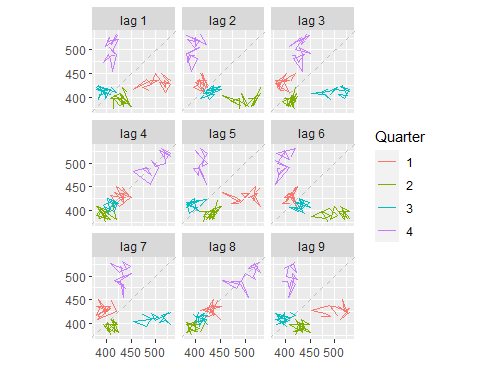


图 2.13: 澳大利亚每季度啤酒产量不同滞后阶数散点图

图中不同颜色代表不同季节，每条线都按时间顺序连接。从图中可以看出，滞后四阶和滞后八阶有正相关关系，说明数据具有很强的季节性。二阶滞后图和六阶滞后图显示，第四季度的峰值对应第二季度的最低点。

window()函数可以截取时间序列的一部分。在本例中，我们截取序列ausbeer1992年之后的部分。

### 2.8 自相关

正如相关系数可以衡量两个变量之间的线性相关关系一样，自相关系数可以测量时间序列 滞后值 之间的线性关系。

以下几个不同的自相关系数，对应于滞后图中的不同情况。例如， 衡量和之间的关系；衡量和$Y\_{t-2}之间的关系。 定义如下：

其中， T是时间序列的长度。 澳大利亚啤酒产量数据的前九个自相关系数如下表所示。 | r1 | r2 | r3 | r4 | r5 | r6 | r7 | r8 | r9 | | — | — | — | — | — | — | — | — | — | | -0.102 | -0.657 | -0.060 | 0.869 | -0.089 | -0.635 | -0.054 | 0.832 | -0.108 | 各值分别对应于图 2.13 中的九个散点图。通过绘制自相关系数图可以描绘 自相关函数 或者是ACF。因此也被称为相关图。

ggAcf(beer2) +  
 ggtitle('')

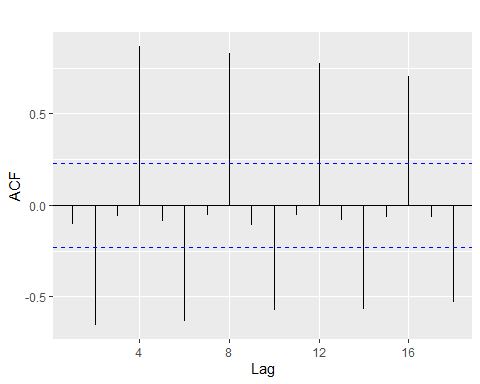


图 2.14: 每季度啤酒产量的自相关函数

在该图中： - r4值最大。这是由于数据的季节性形态：顶峰往往出现在第四季度，谷底往往出现在第二季度。 - r2值最小。这是由于谷底往往在高峰之后的两个季度出现。 - 蓝色虚线之内的区域自相关性可近似看做0。这将会在下节详细阐述。

#### ACF 图中的趋势性和季节性

当数据具有趋势性时，短期滞后的自相关值较大，因为观测点附近的值波动不会很大。时间序列的ACF一般是正值，随着滞后阶数的增加而缓慢下降。

当数据具有季节性时，自相关值在滞后阶数与季节周期相同时（或者在季节周期的倍数）较大。

当数据同时具有趋势和季节性时，我们会观察到组合效应。如图 2.15 是澳大利亚用电量，该序列同时具有趋势和季节性。它的ACF值如图 2.16 所示。

aelec <- window(elec, start = 1980)  
autoplot(aelec) +  
 xlab("年份") +  
 ylab("百万千瓦") +  
 theme(text = element\_text()) +  
 theme(plot.title = element\_text(hjust = 0.5))

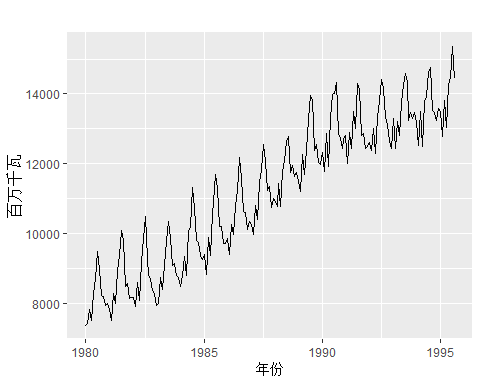


图 2.15: 1980-1995年间澳大利亚月度用电量

ggAcf(aelec, lag = 48) +  
 ggtitle('')

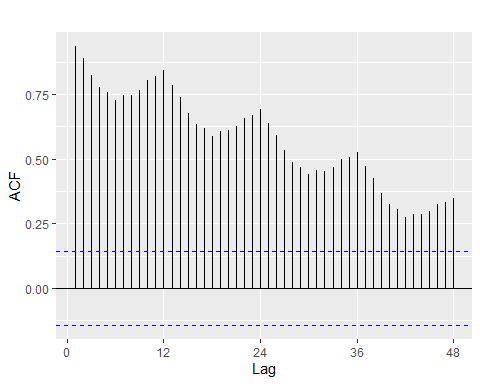


图 2.16: 澳大利亚月度用电量的ACF值

自相关系数值随着滞后阶数增加而缓慢降低，是因为原时间序列中具有趋势变化，而图中的“圆齿状”形状是来源于原时间序列中的季节性变化。

### 2.9 白噪声

“白噪声”是一个对所有时间其自相关系数为零的随机过程。 图 2.17是一个白噪声的例子。

set.seed(30)  
y <- ts(rnorm(50))  
autoplot(y) +  
 ggtitle("白噪声") +  
 theme(text = element\_text()) +  
 theme(plot.title = element\_text(hjust = 0.5))

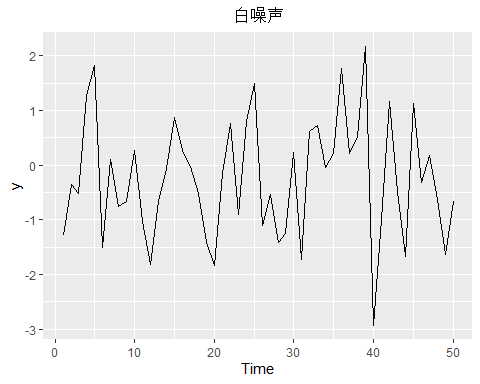


图 2.17: 白噪声时间序列

ggAcf(y) +  
 ggtitle('')

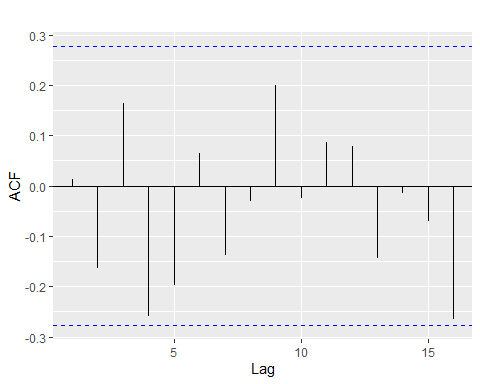


图 2.18: 白噪声函数的自相关函数

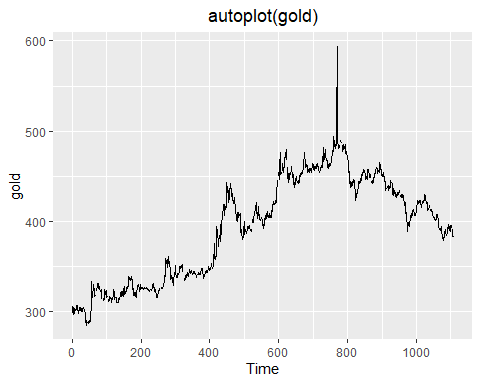
对于白噪声而言，我们期望它的自相关值接近0。但是由于随机扰动的存在，自相关值并不会精确地等于0。对于一个长度为T的白噪声序列而言，我们期望在0.95的置信度下，它的自相关值处于±2/√T之间。我们可以很容易的画出ACF的边界值（图中蓝色虚线）。如果一个序列中有较多的自相关值处于边界之外，那么该序列很可能不是白噪声序列。

在上例中，序列长度T=50，边界为±2/√50=±0.28。所有的自相关值均落在边界之内，证明序列是白噪声。

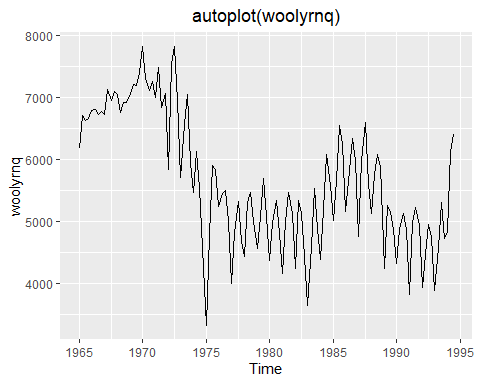
### 2.10 练习

1. 使用帮助函数了解gold、woolyrnq和gas时间序列
2. 使用autoplot()分别绘制上述序列。

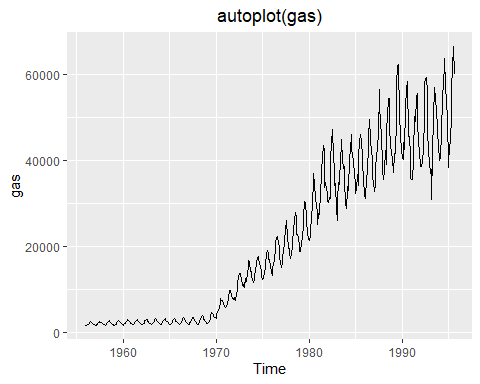
autoplot(gold) +  
 ggtitle("autoplot(gold)") +  
 theme(text = element\_text()) +  
 theme(plot.title = element\_text(hjust = 0.5))



autoplot(woolyrnq) +  
 ggtitle("autoplot(woolyrnq)") +  
 theme(text = element\_text()) +  
 theme(plot.title = element\_text(hjust = 0.5))



autoplot(gas) +  
 ggtitle("autoplot(gas)") +  
 theme(text = element\_text()) +  
 theme(plot.title = element\_text(hjust = 0.5))

 b. 各个序列的频率是多少？ 提示：运用frequency()函数。

frequency(gold)

## [1] 1

frequency(woolyrnq)

## [1] 4

frequency(gas)

## [1] 12

1. 使用which.max()函数找出gold序列的离群点。它是哪个观测点？

which.max(gold)

## [1] 770

1. 从[OTexts.org/fpp2/extrafiles/tute1.csv](https://otexts.com/fppcn/extrafiles/tute1.csv)上下载文件tute1.csv，打开并查看内容。从B列到D列，是三组季度性序列。分别标记为：Sales、AdBudget 和 GDP。Sales是1981年至2005年期间小型公司的季度销售额。 AdBudget 是广告预算，GDP是国内生产总值。三组时间序列均已经剔除了通货膨胀的影响。
2. 首先将数据读入R中,并初步观察数据形态：

tute1 <- read.csv("C:/Users/hsuhau/PycharmProjects/forecasting-methods-and-practice/data/2.10/tute1.csv", header = TRUE)  
View(tute1)

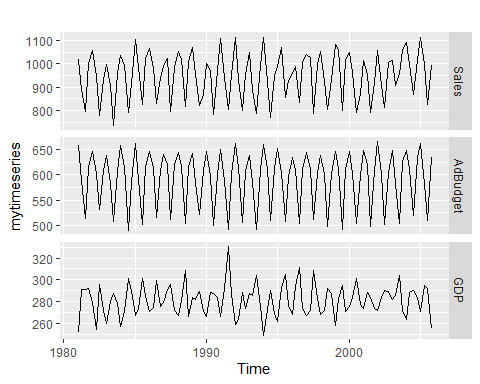
1. 将数据转化为时间序列

mytimeseries <- ts(tute1[, -1], start = 1981, frequency = 4)

（使用`[,-1]`可以移除第一列的数据）

1. 画出每个序列的时间序列图

autoplot(mytimeseries, facets=TRUE)



尝试不设置参数`facets=TRUE`。

1. 从[OTexts.org/fpp2/extrafiles/retail.xlsx](https://otexts.com/fppcn/extrafiles/retail.xlsx)下载澳大利亚月度零售数据。这些代表不同澳大利亚各州的不同类别的零售额，并存储在MS-Excel文件中。
2. 首先将数据读入R中：

retaildata <- readxl::read\_excel("C:/Users/hsuhau/PycharmProjects/forecasting-methods-and-practice/data/2.10/retail.xlsx", skip=1)

必须设置第二个参数`skip=1`。

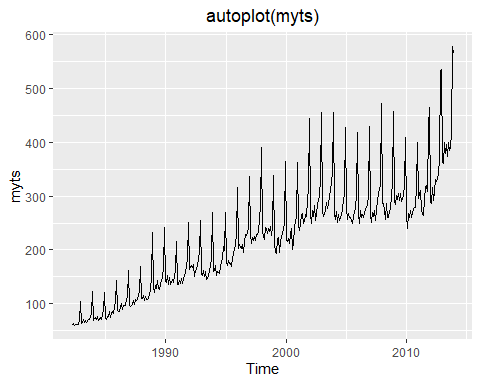
1. 选择其中的一个时间序列：

myts <- ts(retaildata[,"A3349873A"], frequency=12, start=c(1982,4))

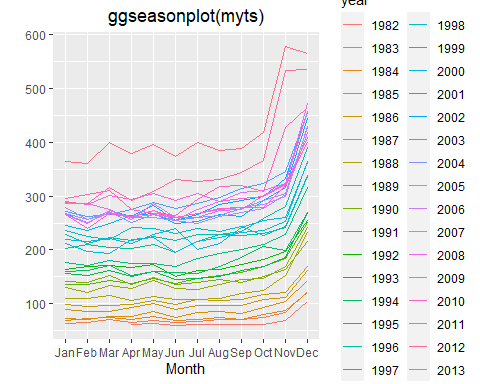
1. 使用下列函数观察时间序列：

`autoplot()`,`ggseasonplot()`,`ggsubseriesplot()`,`gglagplot()`,`ggAcf()`

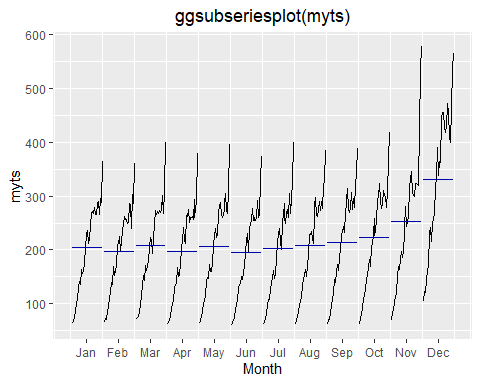
autoplot(myts) +  
 ggtitle("autoplot(myts)") +  
 theme(text = element\_text()) +  
 theme(plot.title = element\_text(hjust = 0.5))



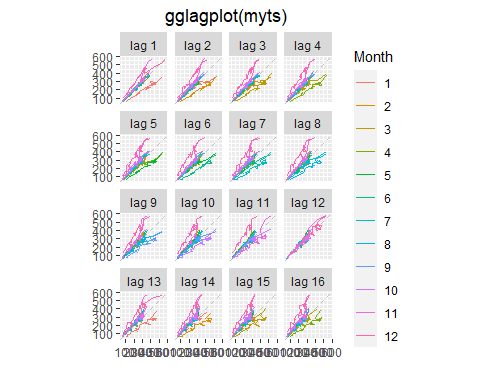
ggseasonplot(myts) +  
 ggtitle("ggseasonplot(myts)") +  
 theme(text = element\_text()) +  
 theme(plot.title = element\_text(hjust = 0.5))



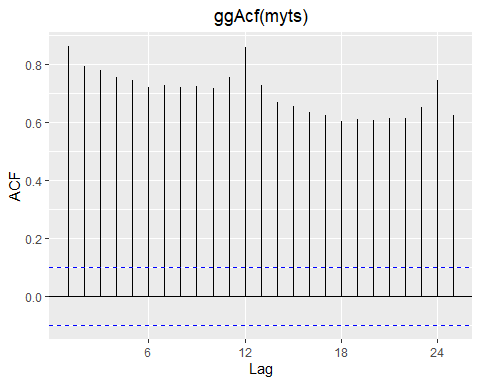
ggsubseriesplot(myts) +  
 ggtitle("ggsubseriesplot(myts)") +  
 theme(text = element\_text()) +  
 theme(plot.title = element\_text(hjust = 0.5))



gglagplot(myts) +  
 ggtitle("gglagplot(myts)") +  
 theme(text = element\_text()) +  
 theme(plot.title = element\_text(hjust = 0.5))



ggAcf(myts) +  
 ggtitle("ggAcf(myts)") +  
 theme(text = element\_text()) +  
 theme(plot.title = element\_text(hjust = 0.5))

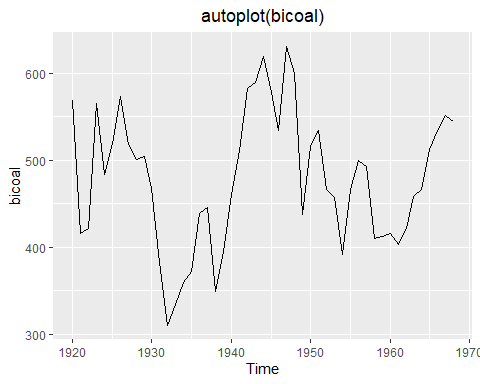


你发现了趋势、季节性或者周期性吗？你对这个时间序列有什么认识？

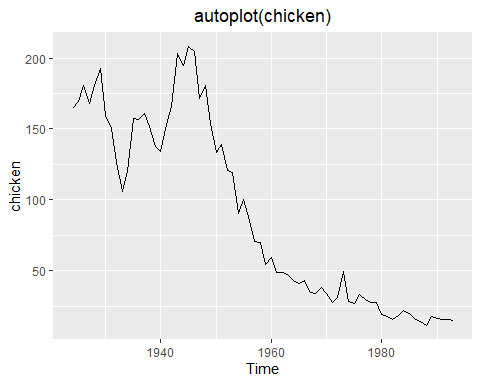
1. 画出以下时间序列的时间序列图：bicoal,chicken,dole,usdeaths,lynx,goog,writing,fancy,a10,h02。

* 使用help()函数找出每个时间序列的数据。
* 在goog的时间序列图上，修改坐标轴名称和标题。

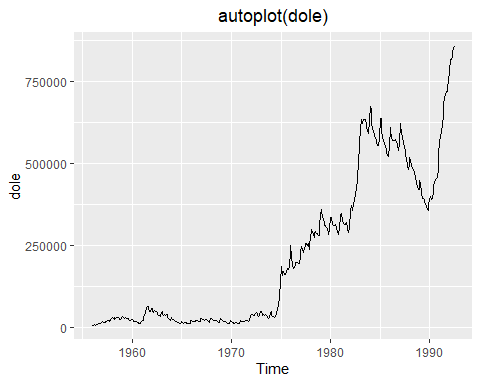
autoplot(bicoal) +  
 ggtitle("autoplot(bicoal)") +  
 theme(text = element\_text()) +  
 theme(plot.title = element\_text(hjust = 0.5))



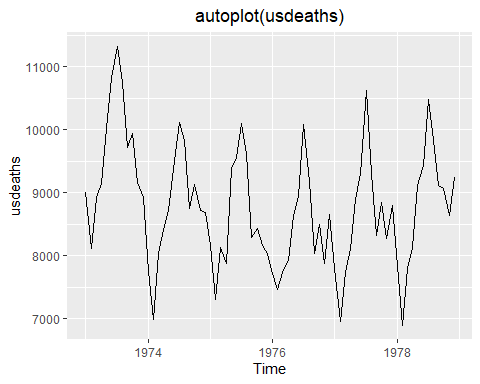
autoplot(chicken) +  
 ggtitle("autoplot(chicken)") +  
 theme(text = element\_text()) +  
 theme(plot.title = element\_text(hjust = 0.5))



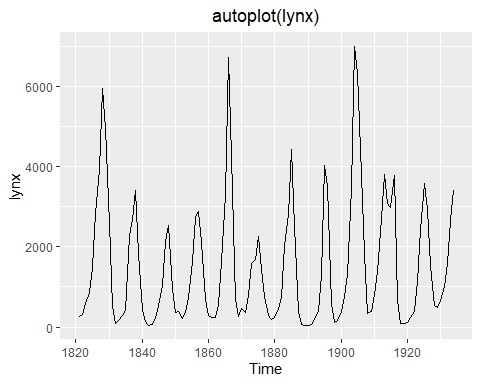
autoplot(dole) +  
 ggtitle("autoplot(dole)") +  
 theme(text = element\_text()) +  
 theme(plot.title = element\_text(hjust = 0.5))



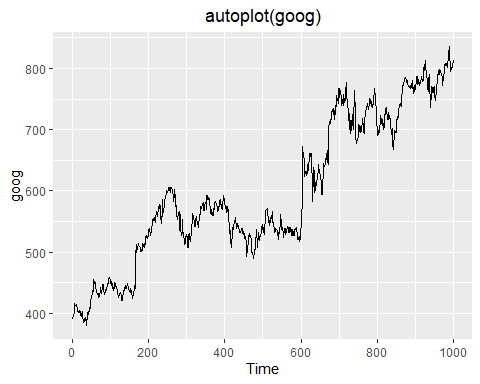
autoplot(usdeaths) +  
 ggtitle("autoplot(usdeaths)") +  
 theme(text = element\_text()) +  
 theme(plot.title = element\_text(hjust = 0.5))



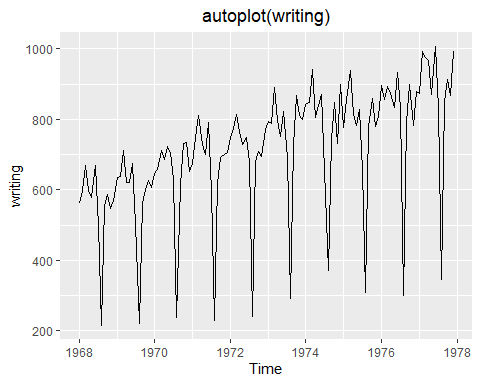
autoplot(lynx) +  
 ggtitle("autoplot(lynx)") +  
 theme(text = element\_text()) +  
 theme(plot.title = element\_text(hjust = 0.5))



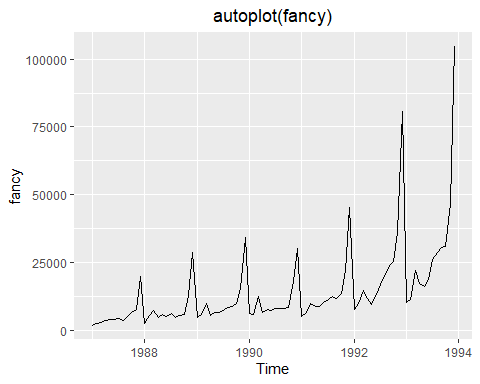
autoplot(goog) +  
 ggtitle("autoplot(goog)") +  
 theme(text = element\_text()) +  
 theme(plot.title = element\_text(hjust = 0.5))



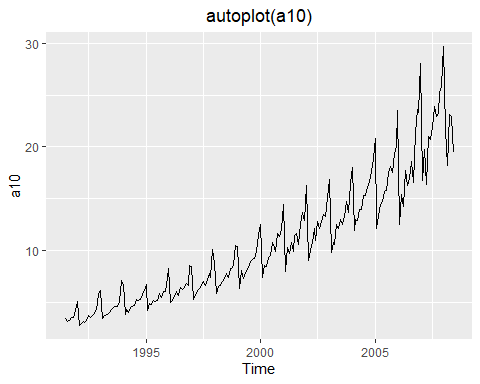
autoplot(writing) +  
 ggtitle("autoplot(writing)") +  
 theme(text = element\_text()) +  
 theme(plot.title = element\_text(hjust = 0.5))



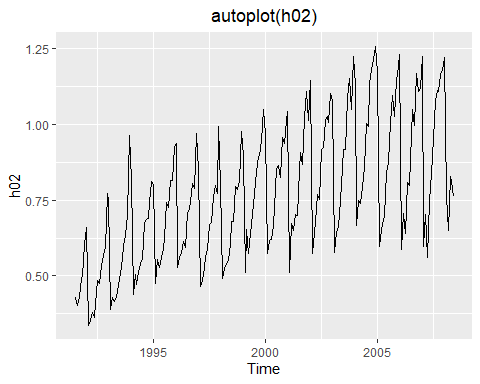
autoplot(fancy) +  
 ggtitle("autoplot(fancy)") +  
 theme(text = element\_text()) +  
 theme(plot.title = element\_text(hjust = 0.5))



autoplot(a10) +  
 ggtitle("autoplot(a10)") +  
 theme(text = element\_text()) +  
 theme(plot.title = element\_text(hjust = 0.5))

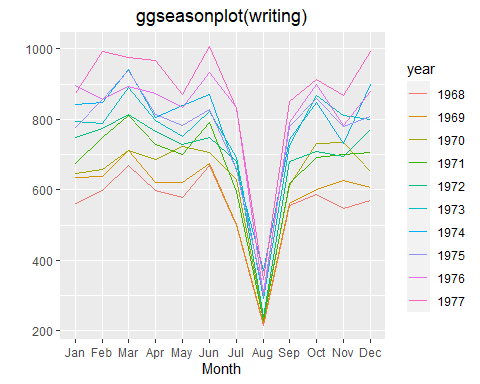


autoplot(h02) +  
 ggtitle("autoplot(h02)") +  
 theme(text = element\_text()) +  
 theme(plot.title = element\_text(hjust = 0.5))

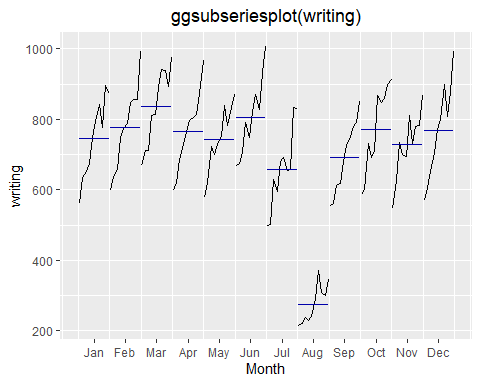
 5. 使用ggseasonplot()函数和ggsubseriesplot()函数来观察writing、fancy、a10和h02四个序列的季节性特征。

* 你对时间序列的季节性有什么认识？
* 你可以识别出异常的年份吗？

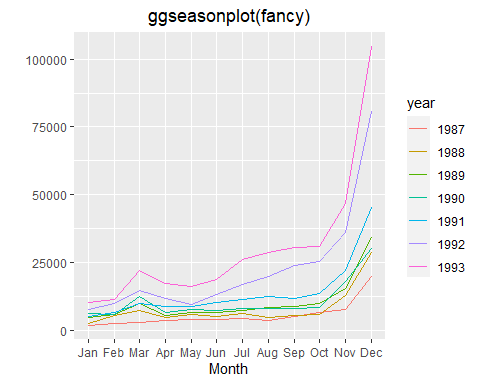
ggseasonplot(writing) +  
 ggtitle("ggseasonplot(writing)") +  
 theme(text = element\_text()) +  
 theme(plot.title = element\_text(hjust = 0.5))



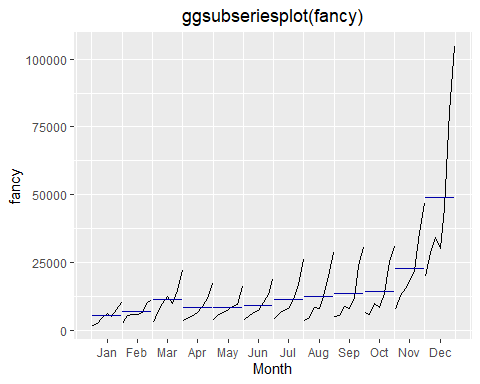
ggsubseriesplot(writing) +  
 ggtitle("ggsubseriesplot(writing)") +  
 theme(text = element\_text()) +  
 theme(plot.title = element\_text(hjust = 0.5))



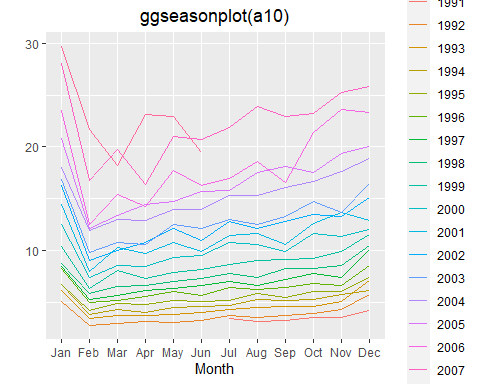
ggseasonplot(fancy) +  
 ggtitle("ggseasonplot(fancy)") +  
 theme(text = element\_text()) +  
 theme(plot.title = element\_text(hjust = 0.5))



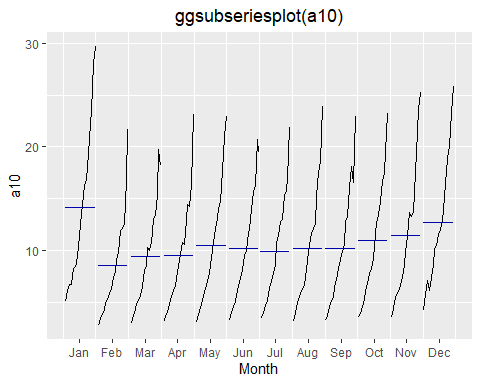
ggsubseriesplot(fancy) +  
 ggtitle("ggsubseriesplot(fancy)") +  
 theme(text = element\_text()) +  
 theme(plot.title = element\_text(hjust = 0.5))



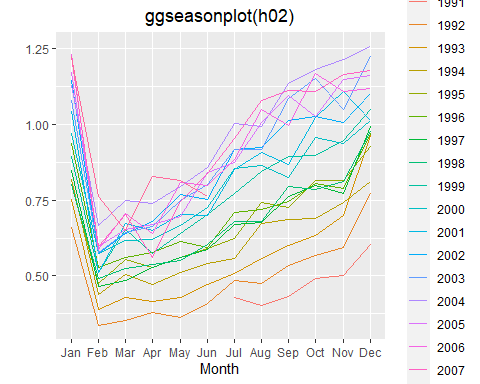
ggseasonplot(a10) +  
 ggtitle("ggseasonplot(a10)") +  
 theme(text = element\_text()) +  
 theme(plot.title = element\_text(hjust = 0.5))



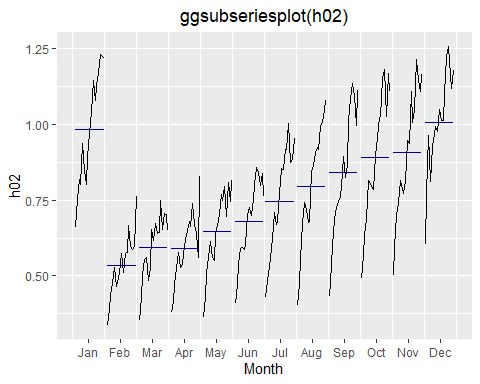
ggsubseriesplot(a10) +  
 ggtitle("ggsubseriesplot(a10)") +  
 theme(text = element\_text()) +  
 theme(plot.title = element\_text(hjust = 0.5))



ggseasonplot(h02) +  
 ggtitle("ggseasonplot(h02)") +  
 theme(text = element\_text()) +  
 theme(plot.title = element\_text(hjust = 0.5))



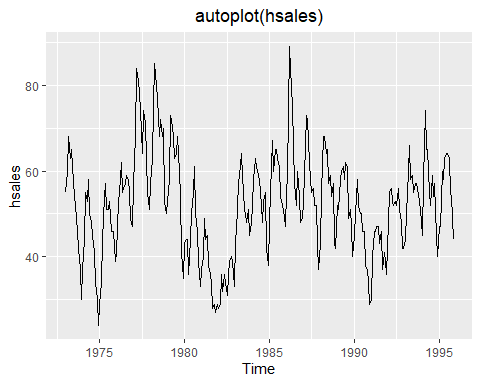
ggsubseriesplot(h02) +  
 ggtitle("ggsubseriesplot(h02)") +  
 theme(text = element\_text()) +  
 theme(plot.title = element\_text(hjust = 0.5))

 6. 使用autoplot()、ggseasonplot()、ggsubseriesplot()、gglagplot()和ggAcf()这些画图函数，探索序列hsales、usdeaths、bricksq、sunspotarea和gasoline的特征。

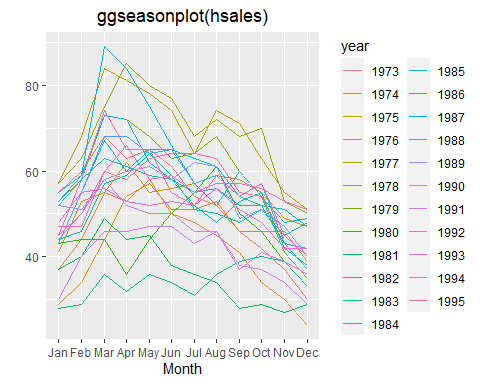
* 你可以发现季节性、周期性或者趋势吗？
* 从这些序列中你发现了什么？

hsales

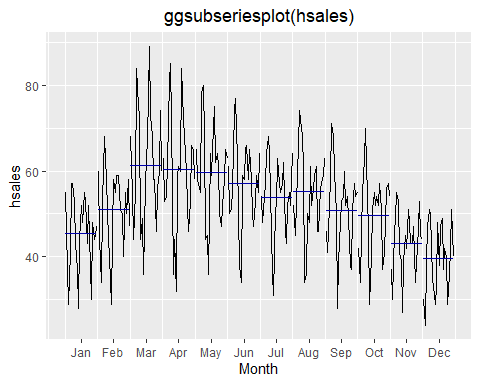
autoplot(hsales) +  
 ggtitle("autoplot(hsales)") +  
 theme(text = element\_text()) +  
 theme(plot.title = element\_text(hjust = 0.5))



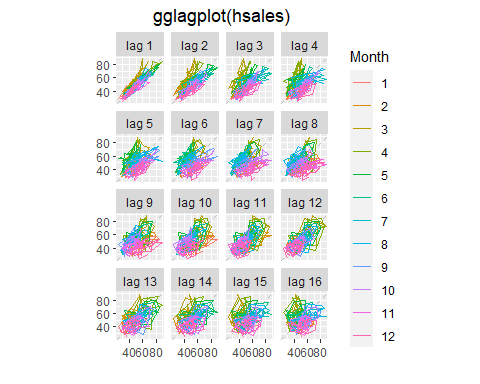
ggseasonplot(hsales) +  
 ggtitle("ggseasonplot(hsales)") +  
 theme(text = element\_text()) +  
 theme(plot.title = element\_text(hjust = 0.5))



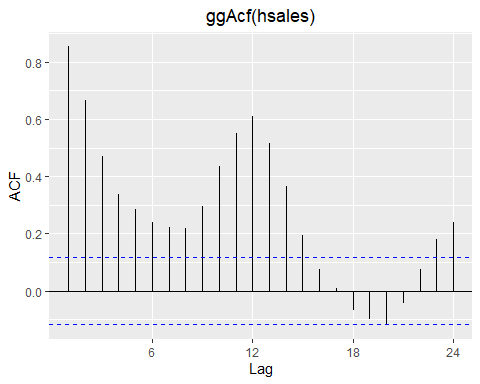
ggsubseriesplot(hsales) +  
 ggtitle("ggsubseriesplot(hsales)") +  
 theme(text = element\_text()) +  
 theme(plot.title = element\_text(hjust = 0.5))



gglagplot(hsales) +  
 ggtitle("gglagplot(hsales)") +  
 theme(text = element\_text()) +  
 theme(plot.title = element\_text(hjust = 0.5))

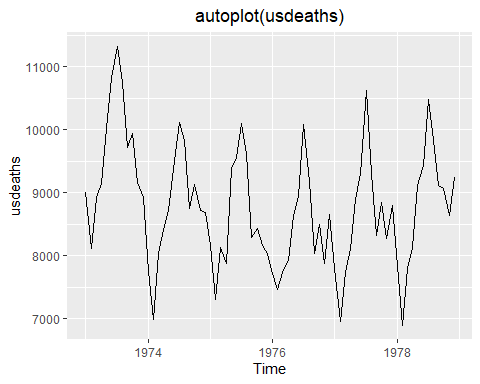


ggAcf(hsales) +  
 ggtitle("ggAcf(hsales)") +  
 theme(text = element\_text()) +  
 theme(plot.title = element\_text(hjust = 0.5))

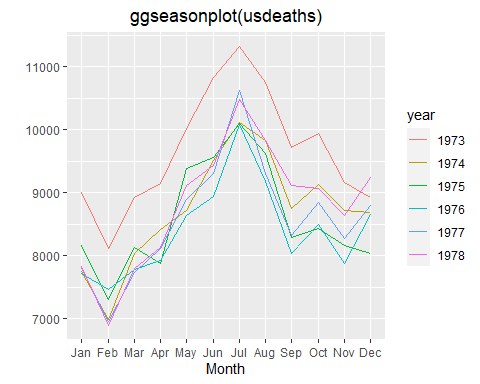


usdeaths

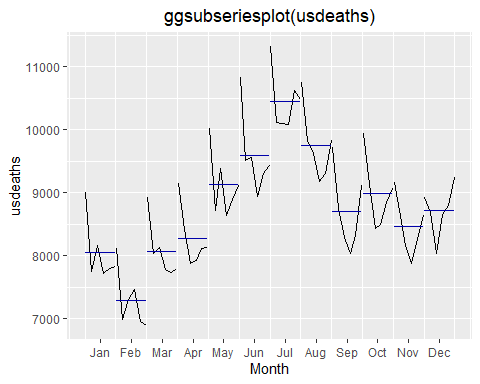
autoplot(usdeaths) +  
 ggtitle("autoplot(usdeaths)") +  
 theme(text = element\_text()) +  
 theme(plot.title = element\_text(hjust = 0.5))



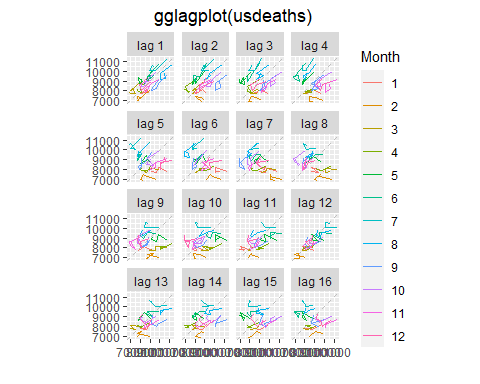
ggseasonplot(usdeaths) +  
 ggtitle("ggseasonplot(usdeaths)") +  
 theme(text = element\_text()) +  
 theme(plot.title = element\_text(hjust = 0.5))



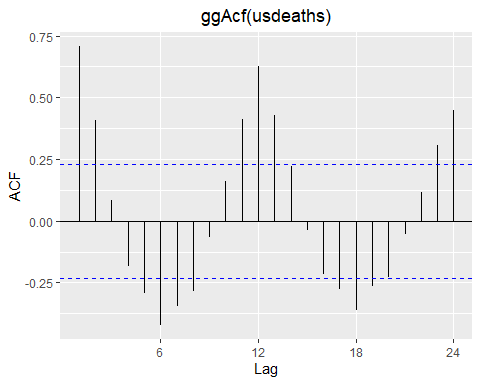
ggsubseriesplot(usdeaths) +  
 ggtitle("ggsubseriesplot(usdeaths)") +  
 theme(text = element\_text()) +  
 theme(plot.title = element\_text(hjust = 0.5))



gglagplot(usdeaths) +  
 ggtitle("gglagplot(usdeaths)") +  
 theme(text = element\_text()) +  
 theme(plot.title = element\_text(hjust = 0.5))

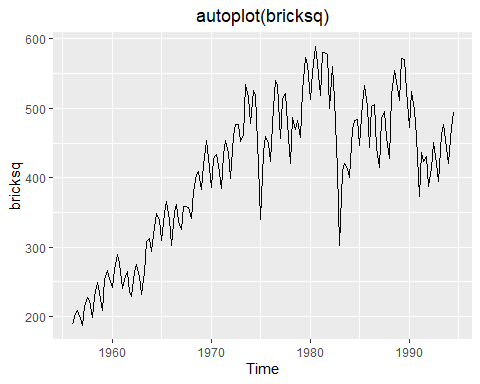


ggAcf(usdeaths) +  
 ggtitle("ggAcf(usdeaths)") +  
 theme(text = element\_text()) +  
 theme(plot.title = element\_text(hjust = 0.5))

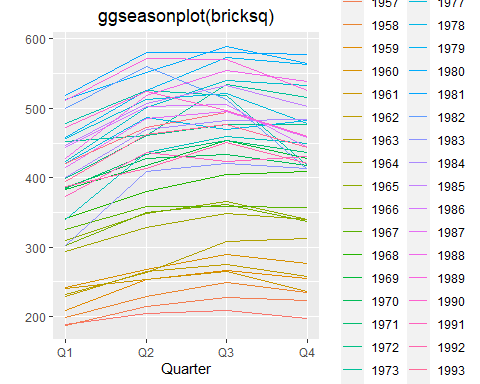


bricksq

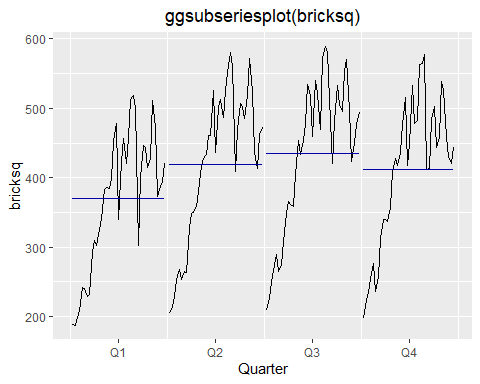
autoplot(bricksq) +  
 ggtitle("autoplot(bricksq)") +  
 theme(text = element\_text()) +  
 theme(plot.title = element\_text(hjust = 0.5))



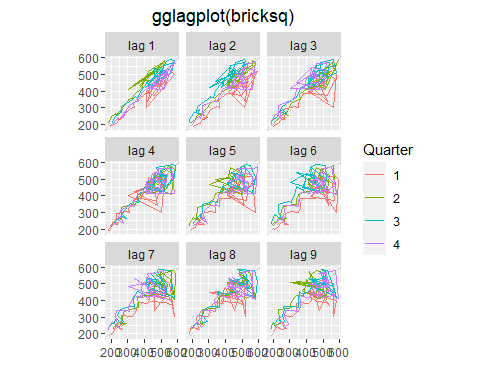
ggseasonplot(bricksq) +  
 ggtitle("ggseasonplot(bricksq)") +  
 theme(text = element\_text()) +  
 theme(plot.title = element\_text(hjust = 0.5))



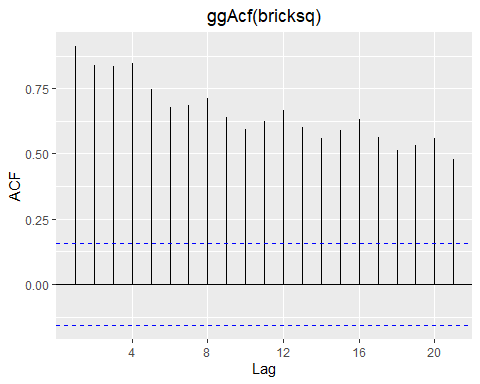
ggsubseriesplot(bricksq) +  
 ggtitle("ggsubseriesplot(bricksq)") +  
 theme(text = element\_text()) +  
 theme(plot.title = element\_text(hjust = 0.5))



gglagplot(bricksq) +  
 ggtitle("gglagplot(bricksq)") +  
 theme(text = element\_text()) +  
 theme(plot.title = element\_text(hjust = 0.5))

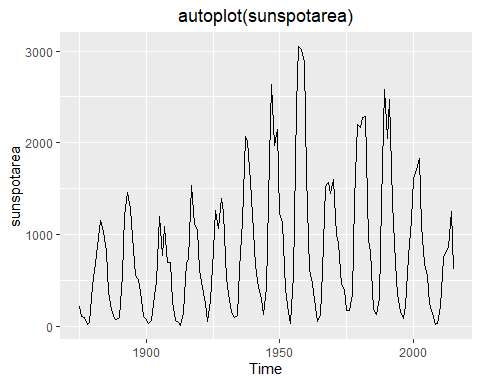


ggAcf(bricksq) +  
 ggtitle("ggAcf(bricksq)") +  
 theme(text = element\_text()) +  
 theme(plot.title = element\_text(hjust = 0.5))



sunspotarea

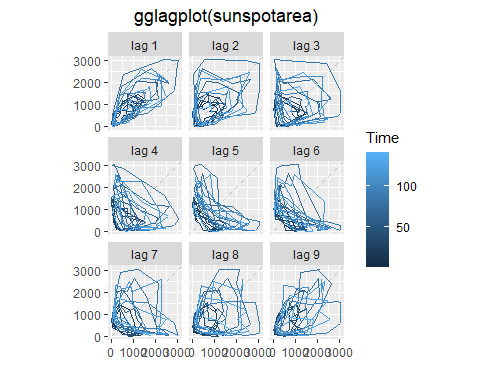
autoplot(sunspotarea) +  
 ggtitle("autoplot(sunspotarea)") +  
 theme(text = element\_text()) +  
 theme(plot.title = element\_text(hjust = 0.5))



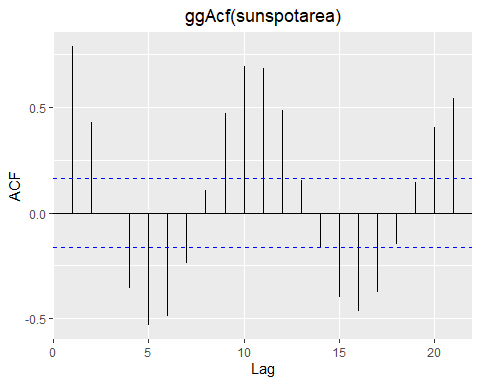
# ggseasonplot(sunspotarea) +  
# ggtitle("ggseasonplot(sunspotarea)") +  
# theme(text = element\_text()) +  
# theme(plot.title = element\_text(hjust = 0.5))  
  
# ggseasonplot(sunspotarea) 中的错误 (Chapter-2-Time-series-graphics.rmd#606): Data are not seasonal

# ggsubseriesplot(sunspotarea) +  
# ggtitle("ggsubseriesplot(sunspotarea)") +  
# theme(text = element\_text()) +  
# theme(plot.title = element\_text(hjust = 0.5))  
  
# ggsubseriesplot(sunspotarea) 中的错误 (Chapter-2-Time-series-graphics.rmd#614): Data are not seasonal

gglagplot(sunspotarea) +  
 ggtitle("gglagplot(sunspotarea)") +  
 theme(text = element\_text()) +  
 theme(plot.title = element\_text(hjust = 0.5))

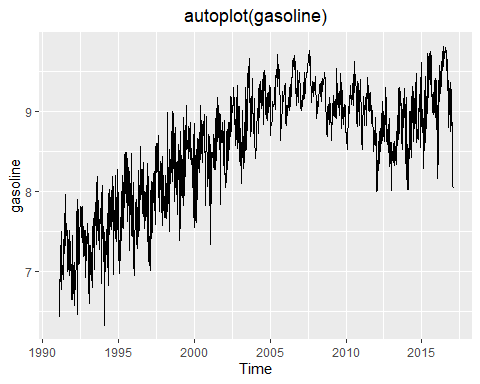


ggAcf(sunspotarea) +  
 ggtitle("ggAcf(sunspotarea)") +  
 theme(text = element\_text()) +  
 theme(plot.title = element\_text(hjust = 0.5))

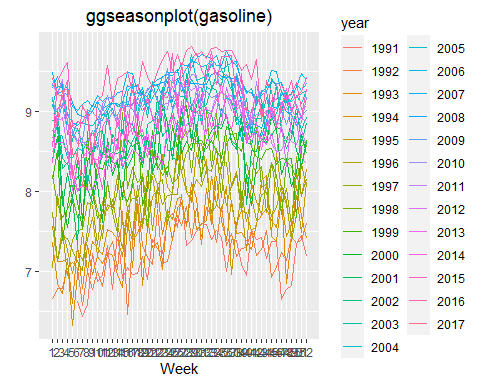


gasoline

autoplot(gasoline) +  
 ggtitle("autoplot(gasoline)") +  
 theme(text = element\_text()) +  
 theme(plot.title = element\_text(hjust = 0.5))

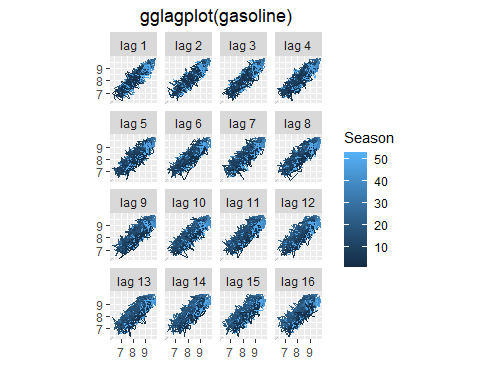


ggseasonplot(gasoline) +  
 ggtitle("ggseasonplot(gasoline)") +  
 theme(text = element\_text()) +  
 theme(plot.title = element\_text(hjust = 0.5))

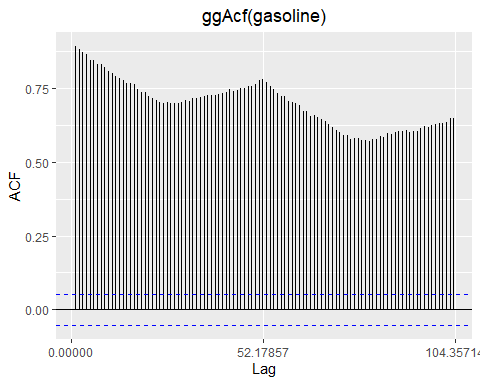


# ggsubseriesplot(gasoline) +  
# ggtitle("ggsubseriesplot(gasoline)") +  
# theme(text = element\_text()) +  
# theme(plot.title = element\_text(hjust = 0.5))  
  
# ggsubseriesplot(gasoline) 中的错误 (Chapter-2-Time-series-graphics.rmd#648): Each season requires at least 2 observations. This may be caused from specifying a time-series with non-integer frequency.

gglagplot(gasoline) +  
 ggtitle("gglagplot(gasoline)") +  
 theme(text = element\_text()) +  
 theme(plot.title = element\_text(hjust = 0.5))

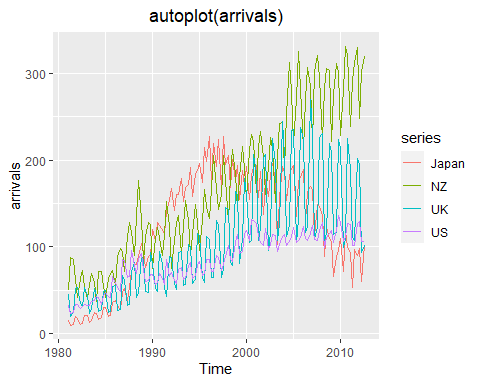


ggAcf(gasoline) +  
 ggtitle("ggAcf(gasoline)") +  
 theme(text = element\_text()) +  
 theme(plot.title = element\_text(hjust = 0.5))

 7. arrivals数据包括从日本、新西兰、英国和美国到达澳大利亚的人数（千）。

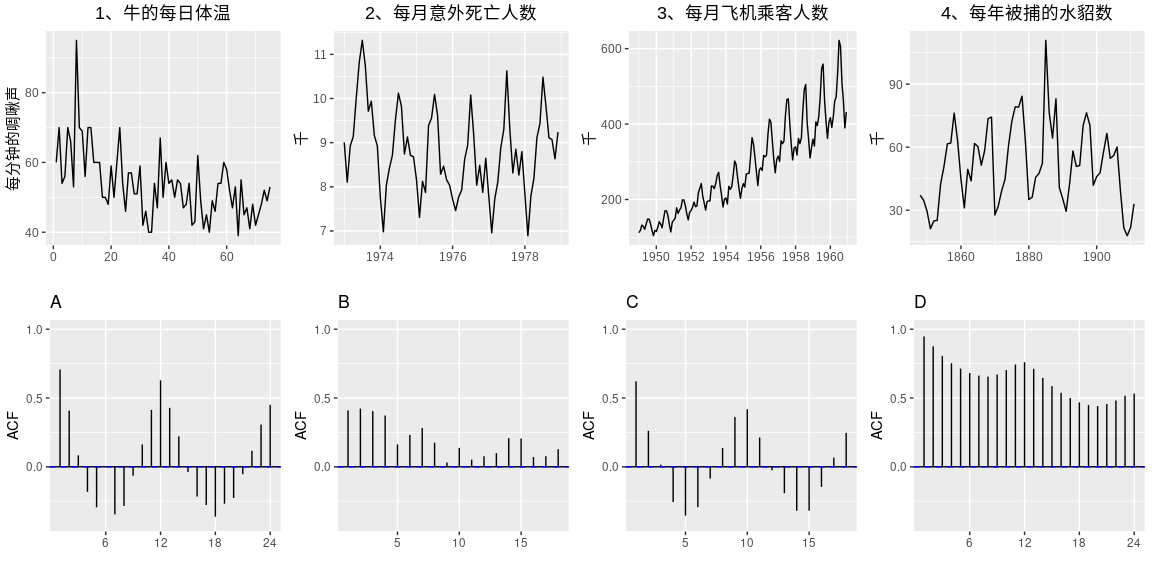
* 使用autoplot()、ggseasonplot()和ggsubseriesplot()函数比较四个国家到达澳大利亚人数的不同之处。
* 你可以找出异常观测点吗？

autoplot(arrivals) +  
 ggtitle("autoplot(arrivals)") +  
 theme(text = element\_text()) +  
 theme(plot.title = element\_text(hjust = 0.5))



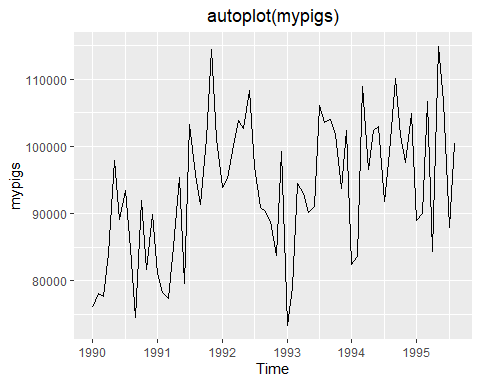
# ggseasonplot(arrivals) +  
# ggtitle("ggseasonplot(arrivals)") +  
# theme(text = element\_text()) +  
# theme(plot.title = element\_text(hjust = 0.5))  
  
# data.frame(y = as.numeric(x), year = trunc(round(time(x), 8)), 中的错误 (Chapter 2 Time series graphics.rmd#679): 参数值意味着不同的行数: 508, 127

# ggsubseriesplot(arrivals) +  
# ggtitle("ggsubseriesplot(arrivals)") +  
# theme(text = element\_text()) +  
# theme(plot.title = element\_text(hjust = 0.5))  
  
# data.frame(y = as.numeric(x), year = trunc(time(x)), season = as.numeric(phase)) 中的错误 (Chapter 2 Time series graphics.rmd#687): 参数值意味着不同的行数: 508, 127

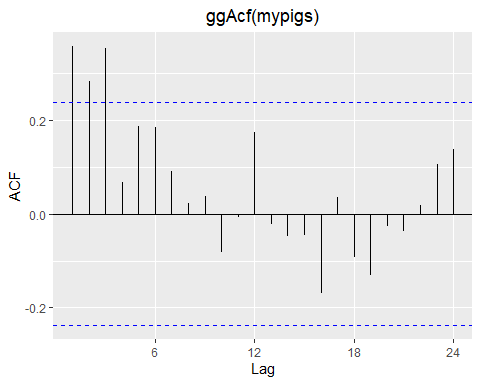
1. 下列时间序列图和ACF图分别对应四个不同的时间序列。你的任务是将第一行的时间序列图和第二行的ACF图分别对应。 
2. 数据集pigs是从1980年1月到1995年8月，维多利亚州每月屠宰猪的总量。使用mypigs <- window(pigs, start=1990)截取1990年之后的数据。运用autoplot和ggAcf函数分别画出mypigs及其ACF图，并将其结果与白噪声的结果（如图[2.17](https://otexts.com/fppcn/wn.html#fig:wnoise)和[2.18](https://otexts.com/fppcn/wn.html#fig:wnoiseacf)所示）进行比较。

mypigs <- window(pigs, start=1990)

autoplot(mypigs) +  
 ggtitle("autoplot(mypigs)") +  
 theme(text = element\_text()) +  
 theme(plot.title = element\_text(hjust = 0.5))



ggAcf(mypigs) +  
 ggtitle("ggAcf(mypigs)") +  
 theme(text = element\_text()) +  
 theme(plot.title = element\_text(hjust = 0.5))



1. 数据集dj包含了连续292个交易日的道琼斯指数。采用ddj <- diff(dj)来计算道琼斯指数的日变化量。画出ddj和它的ACF图。道琼斯指数的日变化类似一个白噪声吗？

ddj <- diff(dj)  
ggAcf(ddj) +  
 ggtitle("ggAcf(ddj)") +  
 theme(text = element\_text()) +  
 theme(plot.title = element\_text(hjust = 0.5))

