

《金融软件与编程》课程教案

宁红泉

目录

1	第一章经济与金融计量学概述	3
1.1	课程工具介绍	3
1.2	经济计量学与金融计量学的含义及建模步骤	6
1.3	经济计量学与金融计量学建模过程	6
1.4	经济与金融模型中的数据	6
1.5	经济与金融计量软件简介	7
1.6	作业	8
2	第二章 R 语言的下载、安装与启动	8
2.1	R 语言的介绍	8
2.2	R 与 Rstudio 软件的界面介绍	9
2.3	R 与 Rstudio 个性化设置	9
2.4	作业	11
3	第三章 R 语言编程基础	12
3.1	数据对象与数据读写	12
3.2	数据集的基本处理	19
3.3	函数与控制流	26
3.4	绘制基本图形	30
3.5	高级绘图	36
3.6	作业	45
4	第四章参数估计假设检验的 R 语言应用	45

4.1	参数估计理论知识	45
4.2	参数估计的 R 语言应用	45
4.3	假设检验的 R 语言应用	45
5	第五章线性回归分析的 R 语言应用	45
5.1	一元线性回归分析基本理论	46
5.2	一元线性回归分析的 R 语言应用	46
5.3	多元线性回归分析基本理论	46
5.4	多元线性回归分析的 R 语言应用	46
5.5	稳健线性回归分析的 R 语言应用	46
6	第六章多重共线性的 R 语言应用	46
6.1	多重共线性的概念	47
6.2	多重共线性的后果	47
6.3	多重共线性的识别和检验	47
6.4	消除多重共线性的方法	47
6.5	多重共线性的 R 语言应用	47
7	第七章异方差问题的 R 语言应用	47
7.1	异方差的概念	48
7.2	异方差的原因与后果	48
7.3	异方差的识别检验	48
7.4	异方差的消除方法	48
7.5	异方差的 R 语言应用	48
8	第八章自相关问题人 R 语言应用	48
8.1	自相关的概念	49
8.2	自相关的产生原因	49
8.3	自相关的识别与检验	49
8.4	自相关性诊断与消除的 R 语言应用	49
9	第九章时间序列分析	49
9.1	单位根检验	49
9.2	ARMA 模型介绍	49
9.3	ARMA 模型的确定	49

1 第一章经济与金融计量学概述	3
9.4 ARMA 模型的确定与检验	49
10 第十章时间序列分析 garch 模型的 R 语言应用	49
10.1 GARCH 模型的介绍	50
10.2 ARCH 效应检验	50
10.3 GARCH 模型的 R 语言应用	50

1 第一章经济与金融计量学概述

教学目的:

- 1.熟悉金融计量软件的种类;
- 2.了解各种金融计量软件的特点;
- 3.掌握金融计量软件使用范围以及编程基础特点。

教学重难点:

- 1.重点: 掌握金融计量软件使用范围以及编程基础特点;
- 2.难点: 了解各种金融计量软件的特点;

1.1 课程工具介绍

学生在课程中推进使用 rmarkdown, 它的编辑器是 rstudio。在进行编写 rmarkdown 的时候, 首先要简单的学一些 latex, 但在 rmarkdow 中显示中文有点麻烦。中文 LaTeX 文档并非难题, 需要借助 CTeX 才行。最佳方式就是使用 tinytex 包。需要注意的是电脑上其他 latex 版本会和它冲突。可以在 Rtudion 上进行设置,

选择 tool-global option-sweave-xelatex, 运行 latex 选用 tinytex.

安装 tinytex 的步骤: `install.packages("devtools"),library(devtools),`

`devtools::install_github('yihui/tinytex')`

`tinytex::install_tinytex(),`

`tlmgr_search('framed.sty')` % 搜索包含 framed.sty 文件的 LaTeX 包

`tlmgr_install('framed')` % 安装 framed 包

`tlmgr_update()` % 更新 TeX Live

latex 的使用:

- documentclass{ctexart} % 或者 ctexrep/ctexbook
- usepackage{ctex} 可以添加文件头:

```
---
documentclass: ctexart
output: rticles::ctex
---
```

1.1.1 R 代码段插入

R 代码用 R Markdown 的语法嵌入, 即三个反引号开始一段代码````{r}`和三个反引号````` 结束一段代码:

```
options(digits = 4)
fit = lm(dist ~ speed, data = cars)
coef(summary(fit))

##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -17.579      6.7584  -2.601 1.232e-02
## speed          3.932      0.4155   9.464 1.490e-12

b = coef(fit)
```

上面回归方程中的斜率是 3.9324, 完整的回归方程为:

$$Y = -17.5791 + 3.9324x$$

画图可以直接通过命令方式实现:

```
par(mar = c(4, 4, .1, .1), las = 1)
plot(cars, pch = 19)
abline(fit, col = 'red')
```

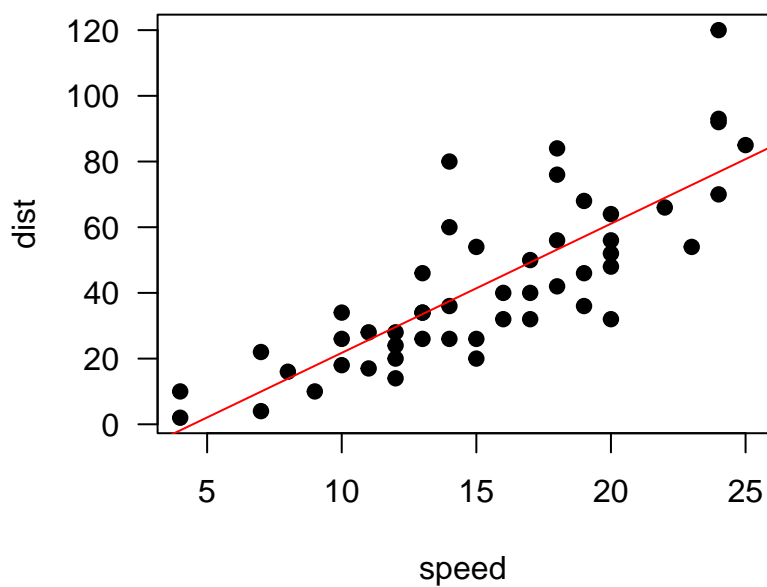


图 1: cars 数据散点图以及回归直线。

1.1.2 主题修改

Rstudio 提供的 rarticles 模板可能由于种种原因不能满足论文格式的要求, LaTeX 用户就是这样永无止境地调格式。

Pandoc: <https://github.com/jgm/pandoc/blob/master/data/templates/default.latex>

它是一个文本文件。若熟悉 LaTeX 的话一看就明白, 只不过里面有些 Pandoc 变量而已。

1.1.3 小结

Rmarkdown 是一款 markdown, 可以通过其他方式打开。它不能像 word 那样, 当时修改就可以看到效果。

但它是一款可以运行 r 代码的 markdown。

1.2 经济计量学与金融计量学的含义及建模步骤

计量经济学是将经济理论实用化、数理化的实证经济学。它是利用经济理论、数学、统计推断等工具对经济现象进行分析的经济学科的分支。

具体的过程包括：

- 模型设计
- 模型建立
- 参数估计
- 检验
- 预测

1.3 经济计量学与金融计量学建模过程

- 关于研究问题的概述
- 样本数据收集
- 选择合适的计量方法来估计模型
- 对模型进行实证检验
- 模型应用

1.4 经济与金融模型中的数据

1. 经济与金融数据的类型

- 横截面数据：同一时间在不同空间的观测数据
- 时间序列数据：按时间序列排列的数据
- 面板数据：时间序列数据和横截面数据相结合的数据

2. 经济与金融数据的来源

- 专业性网站
- 专业数据公司和信息公司

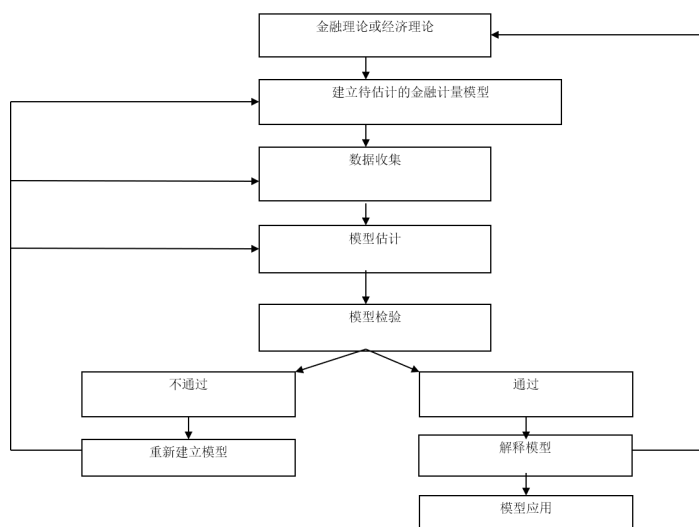


图 2: 金融计量建模流程

- 抽样调查

1.5 经济与金融计量软件简介

1. Eviews 软件
2. SAS 软件
3. Matlab
4. SPSS
5. Stata

通过利用实际的软件操作，简单的介绍它们之间的各自特点。重点介绍 R 与 python 软件的开源思想及它们的发展趋势。

1.6 作业

1. 金融计量学与经济计量学的区别是什么？2. 自行下载 topyra 软件，了解什么是 markdown？

2 第二章 R 语言的下载、安装与启动

教学目的：

1. 熟悉R语言使用流程；
2. 了解选择R语言编程的意义；
3. 掌握使用R语言正确设置。

教学重难点：

1. 重点：掌握R计量软件界面与特点；
2. 难点：正确设置R软件；

2.1 R 语言的介绍

为什么选择 R 语言？

- 免费的软件
- 编程方便，语言灵活，图形功能强大
- 优秀的内在帮助系统
- 高质量、广泛的统计分析、数据挖掘平台
- 国际上 R 语言已然是专业数据分析领域的标准 R 的起源
- R 是 S 语言的一种实现。S 语言是由 AT&T 贝尔实验室开发的一种用来进行数据探索、统计分析、作图的解释型语言。

- R 的使用与 S-PLUS 有很多类似之处，两个软件有一定的兼容性。##
R 与 Rstudio 软件下载与安装 R 可以在 CRAN(Comprehensive R
Archive Network)
<http://cran.r-project.org/mirrors.html> 。
- (利用 R 软件进行现场演示操作，然后要求学生课堂练习)

2.2 R 与 Rstudio 软件的界面介绍

R 自身带的编辑器很不好用，因此可以寻找很多的替代方案，比如可以选择 Emacs 和 Vim 来替代。这里推荐 Rstudio，它是专门用于 R 语言环境的 IDE。
Rstudio 可以从其官网 <http://www.rstudio.com/> 上免费下载安装。

2.3 R 与 Rstudio 个性化设置

2.3.1 获取 R 的帮助

R 提供了大量的帮助功能，学会如何使用这些帮助文档可以在很大程度上助力你的编程工作。通过命令窗口输入代码查看帮助。

函数	功能
help.start()	打开帮助文档
help("plot")或者?plot	查看函数plot的帮助（引号可以省略）
help.search("plot")或者??plot	以plot为关键词搜索本地帮助文档
example("plot")	函数plot的使用示例（引号可以省略）
RSiteSearch("plot")	以plot为关键词搜索在线文档个邮件列表存档
apropos("plot",mode="function")	列出名称中含有plot的所有可用函数
data()	列出当前以加载包中所含的所有可用示例数据集
vignette()	列出当前已经安装包中所有可能的vignette文档
vignette("plot")	为主题plot显示指定的vignette文档

图 3: help 命令使用

课堂演示，学生操作练习。

练习例子：查看 tseries 包，查看数据说明。

2.3.2 R 工作目录

- 工作空间 (workspace) 就是当前 R 的工作环境，它储存着所有用户定义的对象 (向量、矩阵、函数、数据框、列表)。
- 在一个 R 会话结束时，你可以将当前工作空间保存到一个镜像中，并在下次启动 R 时自动载入它。
- 当前的工作目录 (working directory) 是 R 用来读取文件和保存结果的默认目录。
- 可以使用函数 `getwd()` 来查看当前的工作目录，或使用函数 `setwd()` 设定当前的工作目录。
- 如果需要读入一个不在当前工作目录下的文件，则需要在调用语句中写明完整的路径。

2.3.3 Rstudio 常用命令

2.3.4 R 包介绍

- 包是 R 函数、数据、预编译代码以一种定义完善的格式组成的集合。
- 计算机上存储包的目录称为库 (library)。
- 函数 `libPaths()` 能够显示库所在的位置。
- 函数 `library()` 则可以显示库中有哪些包。
- R 自带了一系列默认包 (包括 `base`、`datasets`、`utils`、`grDevices`、`graphics`、`stats` 以及 `methods`)，它们提供了种类繁多的默认函数和数据集。其他包可通过下载来进行安装。
- 使用命令 `install.packages("package_name", "dir")` 即可。
- 查看包帮助： `library(help="package_name")`
- 加载包： `library(package_name)` 或者 `require(package_name)`

函数	功能
<code>getwd()</code>	显示当前的工作目录。
<code>setwd("new_path")</code>	修改当前的工作目录为new_path。
<code>ls()</code>	列出当前工作空间中的对象。
<code>rm(objectList)</code>	移除(删除)一个或多个对象。
<code>rm(list = ls())</code>	移除当前工作空间的所有对象，即清除R工作空间中的内存变量。
<code>help(options)</code>	显示可用选项的说明。
<code>options()</code>	显示或设置当前选项。
<code>history(n)</code>	显示最近使用过的n个命令(默认值为25)。
<code>savehistory("myfile")</code>	保存命令历史到文件myfile中(默认值为.Rhistory)。
<code>loadhistory("myfile")</code>	载入一个命令历史文件(默认值为.Rhistory)。
<code>save.image("myfile")</code>	保存工作空间到文件myfile中(默认值为.RData)。
<code>save(objectlist,file="myfile")</code>	保存指定对象到一个文件中。
<code>load("myfile")</code>	读取一个工作空间到当前会话中(默认值为.RData)。
<code>q()</code>	退出R，并将会询问是否保存工作空间。

图 4: Rstudio 常用命令

2.3.5 常用包

- 空间数据分析类
- 机器学习与统计学习类
- 多元统计类
- 药物动力学数据分析类
- 计量经济类
- 金融分析类
- 并行计算类
- 数据库访问类

2.4 作业

1. 下载安装 R 与 Rstudio，并正确设置界面。
2. 下载包，并使用包查看相关函数命令及数据集。

3 第三章 R 语言编程基础

教学目的：

1. 熟悉R语言对象的属性与分类；
2. 了解R语言各类对象的建立与存取；
3. 掌握R编程基础知识。

教学重难点：

1. 重点：掌握R编程基础知识；
2. 难点：掌握R软件的编程方法；

3.1 数据对象与数据读写

3.1.1 数据类型

基本赋值语句

```
x<-8
```

R 语言的对象常见的数据类型有：字符型、数值型、逻辑型、复数型。此外，也可能是缺省值 (NA)。

类型	辨别	转换
character	is.character()	as.character()
complex	is. complex()	as. complex()
integer	is. integer()	as. integer()
logical	is. logical()	as. logical()
NA	is.na()	as.na()
numeric	is. numeric()	as. numeric()

图 5: R 数据类型

对于未知类型的对象,在 R 中有 3 个函数可以查看对象的类型: `class`、`mode`、`typeof`。

使用格式: `class(x)`

其中 `x` 为需要查看类型的对象, `mode`、`typeof` 函数使用格式与 `class` 函数相同。

实例: 创建 3 个不同类型的数据, 展示 3 个辨别函数的区别。

```
df <- data.frame(c1 = letters[1:3], c2 = 1:3, c3 = c(1, -1, 3.0),
  stringsAsFactors = F)
sapply(df, mode)
```

```
##           c1           c2           c3
## "character" "numeric"    "numeric"
```

```
sapply(df, class)
```

```
##           c1           c2           c3
## "character" "integer"    "numeric"
```

```
sapply(df, typeof)
```

```
##           c1           c2           c3
## "character" "integer"    "double"
```

在展现数据的细节上, `mode`<`class`<`typeof`。 `mode` 函数只查看数据的大类, `class` 函数查看数据的类, `typeof` 函数则更加细化, 查看数据的细类。

3.1.2 数据结构

R 拥有许多用于存储数据的对象类型, 包括向量、矩阵、数组、数据框和列表。

1. 数据框 (data frame) 是 R 中用于存储数据的一种结构: 列表示变量, 行表示观测。

在同一个数据框中可以存储不同类型 (如数值型、字符型) 的变量。

2. 向量是以一维数组的方法管理数据的一种对象类型。可以说向量是 R 语

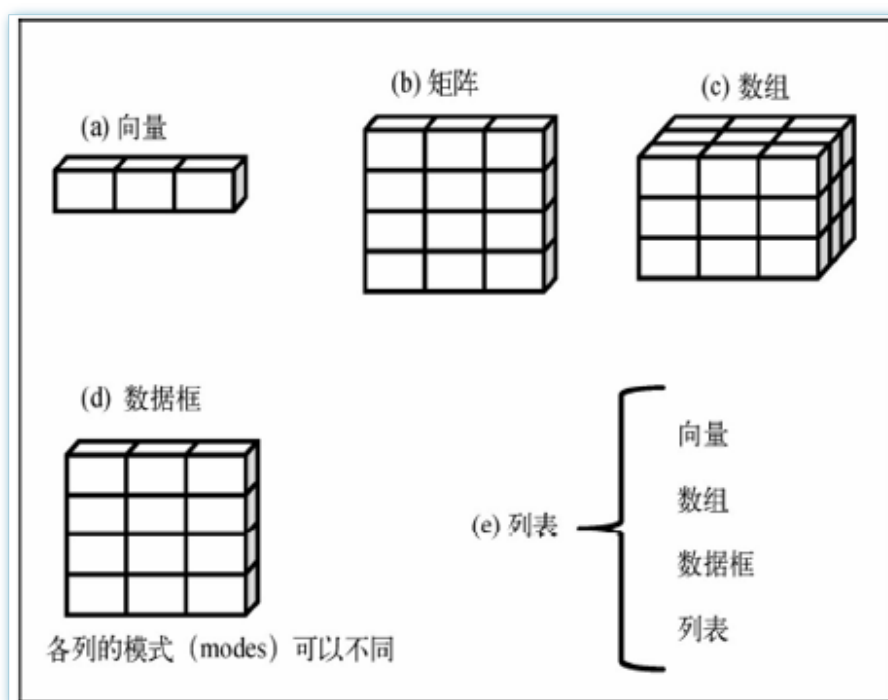


图 6: R 数据结构

言中最基本的数据类型，很多算法函数都是以向量的形式输入的。
向量可以是数值型、字符型、逻辑值型 (T、F) 和复数型。

```
x1 <- c(1, 2, 3, 4) # 创建数值型向量，可写成 x1=c(1:4)
x2 <- c("a", "b", "c", "d") # 创建字符型变量
x3 <- c(TRUE, FALSE, FALSE, TRUE) # 创建逻辑型变量
```

字符型、逻辑值型 (T、F)、数值型和复数型；

一个向量的所有元素都必须属于相同的类型。如果不是，R 将强制执行类型转换。

3.R 语言最强大的方面之一就是函数的向量化。

```
w<-seq(1:10)
x<-sqrt(w)
tow_number<-w+x
```

提问：如果两个向量长度不同，能相加吗？

4. 特殊向量的创建

- seq(1:n:m)
- rep(x,n) 5. 索引向量

只要访问向量中的部分或个别元素。这就是所谓的索引，它用方括号 [] 来实现。

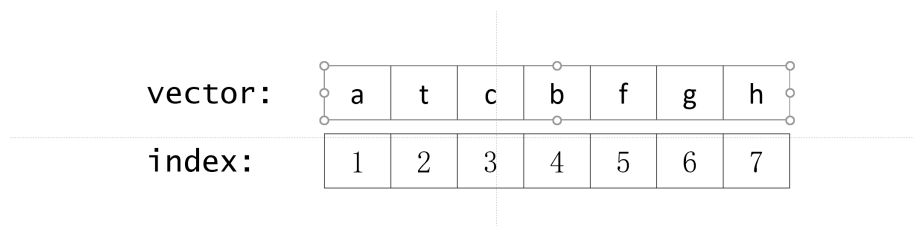


图 7: 向量索引

R 语言中，提供如下多种索引方法。

- 给向量传入正数，它会返回此位置上的向量元素切片。它的第一个位置是 1（而不像其他某些语言一样是 0）。
- 给向量传入负数，它会返回一个向量切片，它将包含除了这些位置以

外的所有元素。

- 给向量传入一个逻辑向量，它会返回一个向量切片，里面只包含索引为 TRUE 的元素。
- 对于已命名的向量，给向量传入命名的字符向量，将会返回向量中包含这些名字的元素切片。

另外，还可以通过掩码方式进行操作：

- which 函数将返回逻辑向量中为 TRUE 的位置。R 语言可以对已经创建好的向量直接进行元素扩展及删除等编辑操作。
向量中元素的删除通过减号加元素下标的形式实现。

向量排序（sort 函数）

常用参数	参数描述	选项
x	排序的对象	排序的对象为数值型，也可以是字符型。
decreasing	排序的顺序	默认设置为FALSE，即升序排序。设置为TRUE时，为降序排序。
na.last	是否将缺失值放到序列的最末尾。	默认设置为FALSE，设置为TRUE时将向量中的NA值放到序列的最末尾。

图 8: 向量排序

6. 矩阵和数组

利用矩阵 matrix 可以描述二维数据，和向量相似，其内部元素可以是实数、复数、字符、逻辑型数据。

矩阵 matrix 使用两个下标来访问元素，A[i,j] 表示矩阵 A 第 i 行、第 j 列的元素。

多维数组 array 可以描述多维数据。array 有一个特征属性叫维数向量（dim 属性），

它的长度是多维数组的维数，dim 内的元素则是对应维度的长度。

矩阵是数组的特殊情况，它具有两个维度。

矩阵的合并

- cbind 函数把其自变量横向拼成一个大矩阵，rbind 函数把其自变量纵向拼成一个大矩阵。
- cbind 函数的自变量是矩阵或看作列向量的向量时，自变量的高度（行数）应该相等。

矩阵的拉直

设 A 是一个矩阵，则函数 `as.vector(A)` 可以将矩阵转化为向量。

```
A <- matrix(1:6,2,3)
AA<-as.vector(A)
```

矩阵的行或列计算的函数

- `colSums` 对矩阵各列求和
- `colMeans` 求矩阵各列的均值
- `rowSums` 对矩阵各行求和
- `rowMeans` 求矩阵各列的均值

矩阵运算函数：

函数	功能
<code>+*/</code>	四则运算，要求矩阵的维数相同，对对应位置的各元素进行运算
<code>colSums()</code>	对矩阵的各列求和
<code>rowSums()</code>	对矩阵的各行求和
<code>colMeans()</code>	对矩阵的各列求均值
<code>rowMeans()</code>	对矩阵的各行求均值
<code>t()</code>	对矩阵的行列进行转置
<code>det()</code>	求解方阵的行列式
<code>outer()</code>	求解矩阵的外积（叉积）
<code>%*%</code>	矩阵乘法，要求第一个矩阵的列数与第二个矩阵的行数相同
<code>diag()</code>	对矩阵取对角元素，若对象为向量，则生成以向量为对角元素的对角矩阵
<code>solve()</code>	对矩阵求解逆矩阵，要求矩阵可逆

图 9: 矩阵运算

数组创建

数组是矩阵的扩展，它把数据的维度扩展到两个以上。可以通过 `array` 函数方便地创建数组。

- 对于矩阵和数组，`dim` 函数将返回其维度的整数值向量。
- 对于矩阵，函数 `nrow` 和 `ncol` 将分别返回行数和列数。

列表和数据框

列表 `list` 和数据框 `data.frame` 也是一个二维数据。在使用 R 语言进行数据分析和挖掘的过程中，向量和数据框的使用频率是最高的，`list` 则在存储较

复杂的数据时作为数据对象类型。

数据框创建

`data.frame` 函数可以直接把多个向量建立为一个数据框，并为列设置名称。

```
my.datasheet <- data.frame(site = c('A','B','A','A','B'),
+season = c('winter','summer','summer','spring','fall'),
+pH = c(7.4,6.3,8.6,7.2,8.9))
names(my.datasheet)
names(my.datasheet)
```

可以通过 `names()` 来读取并编辑列名称。

数据框索引通过 [行下标,]，可以直接获取相应行的所有元素，并以数据框对象形式返回。**列表创建**

```
(my.list <- list(stud.id = 34453,
+               stud.name = '张三',
+               stud.marks = c(14.3,12,15,19)))
$stud.name
$stud.marks
```

3.1.3 数据读写

R 暂时没有很好用的可视化的数据导入工具，所有需要使用命令来导入导出数据。

如果使用 Rstudio 编辑器，可以使用其提供的简单的数据导入功能。

使用它可以从 txt、csv 等文本格式的文件或者从网络中获取数据。

(使用软件进行演示)

1. CSV 文件读取

```
library(learningr)
deer_file <- system.file('extdata','RedDeerEndocranialvolume.dlm',package
='learningr')
deer_data <- read.table(deer_file,header = TRUE,fill = TRUE)
head(deer_data)
```

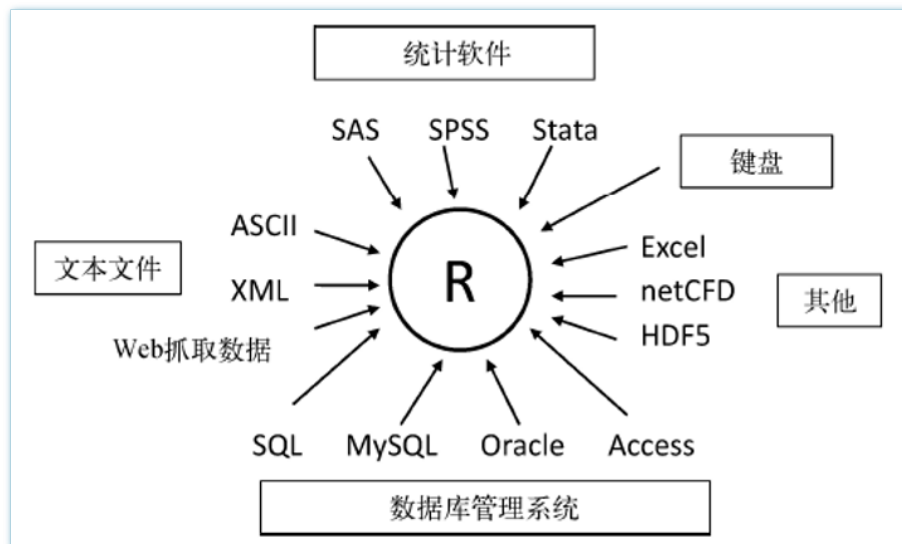


图 10: 数据源读取类型

2. 在 Windows 系统中，可以使用 RODBC 包来访问 Excel 文件，或直接使用 xlsx 包和 XLConnect 包来访问 Excel2007 文件。
其他的数据类型读取，不作介绍，大家可以 help 相关函数与包。

3.1.4 作业

要求学生自行下载一只股票的数据，然后利用 R 软件读取数据，并保存。

3.2 数据集的基本处理

1. 访问数据框变量

```
data.iris <- data.frame(Sepal.Length = c(5.1, 4.9, 4.7, 4.6),
  Sepal.Width = c(3.5, 3.0, 3.2, 3.1),
  Petal.Length = c(1.4, 1.4, 1.3, 1.5), Petal.Width = rep(0.2, 4))
data.iris[, 1] # 索引第一列
data.iris$Sepal.Length # 按列的名称索引
data.iris[1:3, ] # 索引第一至三行
```

```
subset(data.iris, Sepal.Length < 5) # 按条件索引行  
data.iris["Sepal.Length"][1] # 索引Sepal.Length列第一个元素
```

2. 如何创建新变量?

```
mydata <- data.frame(x1 = c(2, 2, 6, 4), x2 = c(3, 4, 2, 8))  
# 创建新变量方法一  
mydata$sumx <- mydata$x1 + mydata$x2  
mydata$meanx <- (mydata$x1 + mydata$x2) / 2  
# 创建新变量方法二  
mydata <- transform(mydata, sumx = x1 + x2, meanx = (x1 + x2) / 2)
```

3. 修改变量名

交互式编辑器修改变量名
(软件演示操作)

3.2.1 数据属性列增加

1. 变量的重命名

R 修改变量名的方式有很多种, 常用 `rename` 函数, `names` 函数, `colnames` 和 `rownames` 等函数实现对变量名字的修改。

rename 函数

`reshape` 包中有一个 `rename` 函数, 可用于修改数据框和列表的变量名, 但不能用于修改矩阵的变量名。

使用格式:

```
rename(dataframe, c(oldname = "newname", oldname = "newname", ...))
```

`names()` 函数和 `rename()` 函数一样, 可修改数据框和列表的变量名, 而不能用于修改矩阵的变量名, 不同点在于,

`names()` 函数会在原数据集中修改变量名, 但 `rename()` 函数并不会直接改变原数据集中的变量名。

使用格式:

```
names(x) <- value
```

2. 属性列增加

`colnames()` 函数和 `rownames()` 函数

`rownames()` 和 `colnames()` 函数可修改矩阵行名和列名, 同时, 也能够修改

数据框的行名和列名。
使用格式：

```
rownames(x) <- value
rownames(x) <- value
```

3.2.2 清洗数据

缺失值分析

在 R 中，缺失值以符号 NA（Not Available，不可用）表示。不可能出现的值（例如，被 0 除的结果）通过符号 NaN（Not a Number，非数值）来表示。与 SAS 等程序不同，R 中字符型和数值型数据使用的缺失值符号是相同的。

函数	描述
is.na(x)	返回一个与x等长的逻辑向量，并且由相应位置的元素是否是NA来决定这个逻辑向量相应位置的元素是TRUE还是FALSE。TRUE表示该位置的元素是缺失值。
anyNA(x, recursive = FALSE)	判断数据中是否存在缺失值，返回TRUE或FALSE值。若存在缺失值则返回TRUE，否则返回FALSE。
na.omit(x)	删除含有缺失值的观测
complete.cases(x)	返回一个逻辑向量，不存在缺失值的行的值为TRUE，存在缺失值的行的值为FALSE。

图 11: 缺失值处理

complete.cases 检查哪行有缺失值。
na.omit 能删除数据框中所有带缺失值的行

处理日期变量

日期值通常以字符串的形式传入 R 中，然后转化为以数值形式存储的日期变量。在 R 中，
字符型的日期值无法进行日期变量的计算，因此可通过日期值处理函数，将字符型的日期值转换成日期变量。

日期格式

（利用软件演示操作）

处理日期变量

R 软件处理日期函数主要有两个：strptime,strftime

函数	功能
<code>Sys.Date</code>	返回系统当前的日期。
<code>Sys.time</code>	返回系统当前的日期和时间。
<code>date</code>	返回系统当前的日期和时间（返回的值为字符串）。
<code>as.Date</code>	将字符串形式的日期值转换为日期变量。
<code>as.POSIXlt</code>	将字符串转化为包含时间及时区的日期变量。
<code>strptime</code>	将字符型变量转化为包含时间的日期变量。
<code>strftime</code>	将日期变量转换成指定格式的字符型变量。
<code>format</code>	将日期变量转换成指定格式的字符串。

图 12: 日期数据处理

符号	含义	示例
<code>%d</code>	数字表示的日期（00~31）	01~31
<code>%a</code>	缩写的星期名	Mon
<code>%A</code>	非缩写的星期名	Monday
<code>%w</code>	数字表示的星期天数	0-6，周日为0
<code>%m</code>	数字表示的月份（00~12）	00~12
<code>%b</code>	缩写的月份	Jan
<code>%B</code>	非缩写的月份	January
<code>%y</code>	二位数的年份	16
<code>%Y</code>	四位数的年份	2016
<code>%H</code>	24小时制小时	00-23
<code>%I</code>	12小时制小时	01-12
<code>%p</code>	AM/PM指示	AM/PM
<code>%M</code>	十进制的分钟	00-60
<code>%S</code>	十进制的秒	00-60

图 13: 日期格式

```
x <- strptime(x, "%Y-%m-%d %H:%M:%S") % 用于输入数据 strptime(x,
format = "%Y/%m/%d") % 用于输出数据
format(x, "%d/%m/%Y"), % 输出的格式转换成 format 定义的格式
```

数据排序

R 中涉及排序的基本函数有 `order`、`sort` 和 `rank` 三个。下面看下其基本用法：

- `sort(x, decreasing = FALSE, ...)`
- `order(..., na.last = TRUE, decreasing = FALSE)`
- `rank(x, na.last = TRUE, ties.method = c("average", "first", "random", "max", "min"))` 演示列子：

```
x <- c(19, 84, 64, 2)
order(x)
```

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

```
rank(x)
```

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

```
sort(x)
```

```
## [1] 2 19 64 84
```

(利用列子说明三者的区别)

3.2.3 选取变量及数据

合并数据集

数据框的编辑可以通过 `rbind` 和 `cbind` 函数。需要注意的是，使用 `rbind` 和 `cbind` 函数对于数据框而言，

分别为增加新的样本数据和增加新属性变量。

`rbind` 的自变量的宽度（列数）应该与原数据框的宽度相等，`cbind` 的自变量的高度（行数）应该与原数据框的高度相等。

选取变量及数据

删除向量中的变量

```
vector <- c(1, 2, 3, 4)
```

```
vector[-1] % 删除第一个元素
```

使用 subset 函数选取数据

subset 函数是一种较为简便的方法用来选取变量与观测变量，功能为选取变量与观测变量。

代码：

```
df1 <- data.frame(name = c("aa", "bb", "cc"), age = c(20, 29, 30),
                  sex = c("f", "m", "f"))
selectresult1 <- subset(df1, name == "aa", select = c(age, sex))
selectresult2 <- subset(df1, name == "aa" & sex == "f", select = c(age, sex))
```

随机抽样

简单随机抽样可通过 srswr 函数，srswor 和 sample 函数实现。srswr 函数和 srswor 函数在 sampling 包中，

使用前需要先加载 sampling 包。下面看下其基本用法：

```
srswr(n, N)
```

```
srswor(n, N)
```

```
sample(x, size, replace = FALSE, prob = NULL)
```

3.2.4 整合数据

使用 SQL 语句操作数据：

```
name<-c(rep("张三", 1, 3), rep("李四", 3))
```

```
subject<-c("数学", "语文", "英语", "数学", "语文", "英语")
```

```
score<-c(89, 80, 70, 90, 70, 80)
```

```
stuid<-c(1, 1, 1, 2, 2, 2)
```

```
stuscore<-data.frame(name, subject, score, stuid)
```

```
library(sqldf)
```

```
% 计算每个人的总成绩并排名 (要求显示字段：姓名，总成绩)
```

```
sqldf("select name, sum(score) as allscore from stuscore group by name
```

```
order
```

数据汇总统计

使用格式: `aggregate(x, by, FUN)`

粘贴数据结构——R 中合并两个数据集可以通过专门的函数 `merge()` 来实现。

在 R 中，数据融合通过 `reshape2` 包中的 `melt()` 函数实现。

使用格式:

```
melt(data, varnames, value.name = "value", na.rm = FALSE)
```

数据重塑

`melt` 获得的数据可以用 `acast` 或 `dcast` 还原。`acast` 获得数组，`dcast` 获得数据框。

使用格式:

```
cast(data, formula, fun.aggregate = NULL,...)
```

3.2.5 处理字符数据

正则表达式是用于描述或匹配一个文本集合的表达式。所有英文字母、数字和很多可显示的字符本身就是正则表达式，用于匹配它们自己。

符号	描述
.	除了换行以外的任意字符
\\	转义字符，如要匹配括号就要写成 "\\(\\)"
	表示可选项，即 前后的表达式任选一个。
^	放在表达式开始处表示匹配文本开始位置， 放在方括号内开始处表示非方括号内的任一字符
\$	放在句尾，表示一行字符串的结束
()	提取匹配的字符串，(\\s*)表示连续空格的字符串
[]	选择方括号中的任意一个（如[a-z]表示任意一个小写字符）
{}	前面的字符或表达式的重复次数。 如{5,12}表示重复的次数不能少于5次，不能多于12次，否则都不匹配。
*	前面的字符或表达式重复零次或更多次
+	前面的字符或表达式重复一次或更多次
?	前面的字符或表达式重复零次或一次

图 14: 常用字符串式

常用的字符串处理函数:

(利用软件演示正则条件处理函数) 学生练习

函数	描述
<code>nchar(x)</code>	计算x中的字符数量
<code>substr(x, start, stop)</code>	提取或替换一个字符向量中的子串
<code>grep(pattern, x, ignore.case = FALSE, perl = FALSE, value = FALSE, fixed = FALSE, useBytes = FALSE, invert = FALSE)</code>	字符串查询，返回结果为匹配项的下标。
<code>sub(pattern, replacement, x, ignore.case = FALSE, fixed=FALSE)</code>	对第一个满足条件的匹配做替换，原字符串并没有改变，要改变原变量只能通过再赋值的方式。
<code>gsub(pattern, replacement, x, ignore.case = FALSE, perl = FALSE, fixed = FALSE, useBytes = FALSE)</code>	把所有满足条件的匹配都做替换，原字符串并没有改变，要改变原变量只能通过再赋值的方式。
<code>strsplit(x, split, fixed = FALSE, perl = FALSE, useBytes = FALSE)</code>	在split处分割字符向量x中的元素。
<code>paste(..., sep = " ", collapse = NULL)</code>	连接字符串，分隔符为sep

图 15: 字符串处理函数

3.3 函数与控制流

3.3.1 常用函数处理数据

和其它数据分析软件一样，在 R 语言中，也有许许多多可应用于数值计算和统计分析的数值函数，

主要可以分成数学函数，统计函数和概率函数三大类。（利用 PPT 简单讲解）

1.apply 家族

apply 包括 `apply`、`lapply`、`tapply`、`sapply`、`vapply`、`mapply` 等几个函数，这些函数都能够

把函数按照某种方式运用于数据。

这些函数的使用方法为：

- `apply(x, MARGIN, FUN, ...)` # 把 FUN 函数运用到 x 数据的第 MARGIN 维度上
- `lapply(x, FUN, ...)` # 把函数 FUN 运用到列表的每一个元素
- `tapply(x, INDEX, FUN=NULL, ..., simplify=TRUE)` # 把 FUN 函数根据 INDEX 索引应用到 x 数据
- `sapply(x, FUN, ..., simplify=TRUE, USE.NAMES=TRUE)`
% 是 `lapply` 函数更加友好的版本，可以使用 `simplify` 参数来调整输出的数据格式
- `vapply(x, FUN, FUN.VALUE, ..., USE.NAMES=TRUE)` % 类似于 `sapply`，但是返回值只能按照预先制定的方式输出

- `mapply(FUN, ..., MoreArgs = NULL, SIMPLIFY = TRUE, USE.NAMES = TRUE)` # 运用于多变量情况（利用软件演示 `apply` 函数示例）

3.3.2 编写条件分支语句

3.3.3 编写循环语句

1.if/else 判断语句

在 R 语言中创建 if-else 语句的基本语法如下所示。

```
if (boolean expression) {  
    // statement(s) will execute if the boolean expression is true.  
} else {  
    // statement(s) will execute if the boolean expression is false.  
}
```

if/else 判断语句

例子: if 语句

```
# if-else 语句  
x <- c("what", "is", "truth")  
if ("Truth" %in% x) {  
    print("Truth is found")  
} else {  
    print("Truth is not found")  
}
```

```
## [1] "Truth is not found"
```

该语句可以实现多重条件的嵌套，三重嵌套的条件语句的基本语法如下所示。

```
if (boolean_expression 1) {  
    // Executes when the boolean expression 1 is true.
```

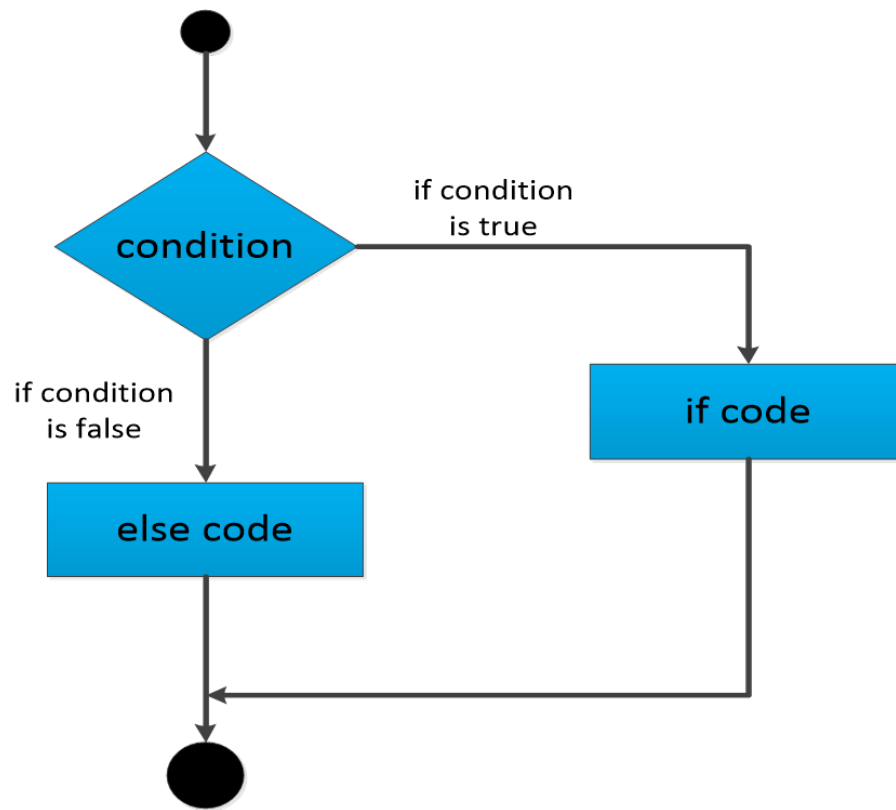


图 16: if 语句

```
} else if ( boolean_expression 2) {  
    // Executes when the boolean expression 2 is true.  
} else if ( boolean_expression 3) {  
    // Executes when the boolean expression 3 is true.  
} else {  
    // executes when none of the above condition is true.
```

2.switch 分支语句

switch 分支语句的使用格式如下:

```
switch(expression,list)
```

3.for 循环语句

使用格式如下:

```
for (name in expr1) {expr2}
```

在循环过程中, 若需要输出每次循环的结果, 可使用 cat 函数或 print 函数, 接下来将介绍 cat 函数的使用方法.

使用格式如下:

```
cat(expr1,expr2,...)
```

3.while 循环语句

使用格式如下:

```
while (cond) {expr}
```

4.repeat-break 循环语句

使用格式如下:

```
repeat expr 或者 repeat {if(cond){break}}
```

3.3.4 编写自定义函数

一个函数的结构大致如下所示。

```
myfunction <- function(arglist) {  
    statements  
    return(object)  
}
```

函数体通常包括三个部分, 异常处理、运算过程和返回值.

- 异常处理：输入的数据不能满足函数计算的要求，或者类型不符，这时应设计相应的机制提示哪个地方出现错误。
- 运算过程：包括具体的运算步骤。运算过程和该函数要完成的功能有关。
- 返回值：用 return 函数给出，返回对象的数据类型是任意的，从标量到列表皆可。函数在内部处理过程中，一旦遇到 return，就会终止运行，将 return 内的数据作为函数处理的结果给出。

3.4 绘制基本图形

3.4.1 绘制基础图形

分析数据第一件要做的事情就是观察它。对于每个变量需要注意的是最常见的值，值域，不寻常的观测，多个变量的关系，是否符合模型假设等。对数据进行可视化就显的很重要。

函数	图形	功能
hist	直方图	分布
sm.density.compare	密度图	分布
boxplot	箱线图	分布
vioplot	小提琴图	分布
barplot	条形图	分布
dotchart	Cleveland点图	分布
pie	饼图	分布
plot	根据作图对象而异，最简单的是散点图	关系（对散点图），图形不同功能不同
pairs	散点图矩阵	关系
corrgram	相关图	关系

图 17: 画图函数

qqplot	QQ图	假设检验
mosaicplot	马赛克图	假设检验
stars	星状图	突出特征
sunflowerplot	向日葵散点图	突出特征
contour	等高图	聚类
heatmap	热图	聚类

图 18: 画图函数

1. 直方图

在 R 中, hist 函数可用于绘制直方图, 显示连续数据的分布情形。hist 函数的具体用法如下所示。

```
hist(x,breaks= ,freq=,...)
```

2. 条形图

在 R 中, 可以使用 barplot 函数绘制条形图, 展示各类数据的数量分布情形。条形图的 x 轴是数据类别,

y 轴是相应类别的频数。barplot 函数的具体用法如下所示。

```
barplot(height, beside =, horiz =, , ...)
```

3. 饼图

R 语言里, 提供的绘制饼图函数为 pie 函数, 其具体用法如下所示。

```
pie(x, labels = names(x), radius = 0.8,...)
```

4. 箱线图

R 中, boxplot 函数用于绘制箱型图。单独的箱线图的使用格式如下所示。box-

```
plot(x,...,range=,width=,varwidth=,notch=,names=,horizontal=,add=FALSE,...)
```

5. 散点图

在 R 语言中, 绘制散点图的函数是 plot 函数, 其具体用法如下所示。

```
plot(x, y, ...)
```

6. 多变量相关矩阵图

在 R 的 corrgram 包中的 corrgram 函数可绘制多变量相关矩阵图。使用格

式如下所示。

```
corrgram(x,order=, lower.panel= , upper.panel=,text.panel=,diag.panel=,...)
```

7. 核密度图

sm.density.compare 函数可以直接堆放多条密度曲线。使用格式如下。

```
corrgram(x,order=, lower.panel= , upper.panel=,text.panel=,diag.panel=,...)
```

8.QQ 图

QQ 图的原理是如果一批数据服从某种理论分布，看其经验分布和理论分布是否一致。将排序后的数据

和理论分布的分位数进行比较后大致相等，说明了经验分布和理论分布相似。

使用格式如下：

```
qqplot(x, y,...); qqnorm(y,...);qqline(y)
```

(利用实际数据演示)

3.4.2 修改图形参数

1. 修改颜色
- R 语言提供了自带的固定种类的颜色，主要涉及的是 colors 函数
 - 通过 palette 函数固定调色板，只要设定好了调色板，它的取值就不会再改变（直到下一次重新设定调色板）。
2. 线条样式

函数	说明	使用格式
lines	在画布中添加曲线，可以设置的参数包括：线条样式（lty）、颜色（col）、粗细（lwd）等	lines(x, lty = , lwd = ,...)
abline	在画布中添加参考线，可以设置的参数包括：直线的截距（a）、直线的斜率（b）、水平线的纵轴值（h）、垂直线的横轴值（v）等	abline(a = NULL, b = NULL, h = NULL, v = NULL, reg = NULL, lty = , lwd = ,...)
segments	两点之间绘制线段，绘制对象是两端点的坐标	segments(x0, y0, x1, y1, lty = , lwd = ,...)
arrows	在线段端点上加箭头，箭头与线段之间的夹角（angle）可调	arrows(x0, y0, x1, y1, angle = , lty = , lwd = ,...)
grid	在绘图的基础上添加网格线，其中ny用户设置水平网格的数目，nx用于设置垂直网格的数据。equilogs是当坐标取了对数之后，是否仍使用等距的网格线	grid(nx = NULL, ny = nx, col = , lty = , lwd = , equilogs = TRUE)
rug	x为坐标轴须的位置；ticksize为坐标轴须的长度；side为坐标轴须的位置，默认值1为x轴，取值2时为y轴	rug(x, ticksize = , side = , col = , lty = , lwd = ,...)

图 19: 线条样式

3. 修改文本属性

title 函数，text 函数和 mtext 函数可以在打开的画布上添加文字元素，其中 title 函数是在图形上添加标题元素，text 函数可以在图形中任意位置添加文本，mtext 函数则是对图形的四条边上添加文本。

函数	说明	使用格式
title	添加标题元素，其中main是主标题，sub是副标题，xlab是x轴标题，ylab是y轴标题，选项都是一个列表list(text, font=, col=, cex=,...)或者简单的text，text是文本内容	title(main = NULL, sub = NULL, xlab = NULL, ylab = NULL, line = NA, outer = FALSE,...)
text	在图形中任意位置添加文本，其中x, y 确定 标签位置，labels是文本内容	text(x, y = NULL, labels = seq_along(x), cex = 1, col = NULL, font = NULL,...)
mtext	在图形的四条边上添加文本，其中text与labels一样指文字的内容，side取值为整数1~4分别把周边文本作在表示图形的下、左、上、右边。line设置一个距离图形边缘的行数	mtext(text, side = 3, line=0, cex = NA, col = NA, font = NA, ...)

图 20: 文本属性

4. 设置坐标轴

绘图函数中，设置坐标轴展示和范围的参数

参数	描述
axes	逻辑参数，如果axes=TRUE（默认），则显示坐标轴，如果axes=FALSE，则隐藏坐标轴。
xaxt/yaxt	坐标轴样式，默认值“s”，表示x /y轴以标准样式显示，取值“n”表示隐藏x /y轴
xaxs/yaxs	坐标轴计算方式，默认值“r”表示把原始数据的范围向外扩大4%，作为x /y轴范围，取值“l”表示x /y轴范围为原始数据范围
xlim/ylim	坐标轴范围，设置为c(from,to) , from是x /y轴的首坐标,to是尾坐标

图 21: 设置坐标轴

axis 函数来创建自定义的坐标轴

axis 函数绘图例子：

参数名称	参数解释
side	坐标轴所在的边，1,2,3,4分别表示下，左，上，右
at	通过向量来设置坐标轴内各刻度标记的位置，at参数要与labels向量一一对应
labels	一个向量字符，表示坐标轴各刻度的名称（刻度标记），labels参数要与at向量一一对应
font.axis	刻度标记的字体，1（默认）为正常字体，2表示粗体，3表示斜体，4表示粗斜体。
cex.axis	刻度标记的大小，1（默认）表示正常大小，小于1表示缩放，大于1表示放大
col.axis	刻度标记的颜色，对应颜色名称即可
tick	设置是否画出坐标轴，为TRUE（默认）时，表示画出坐标轴，为FALSE时则不画出，此时并不影响刻度标记labels的展示

图 22: 坐标轴刻度

```
plot(c(1:12), col="white", xaxt="n", yaxt="n", ann = FALSE)
axis(1, at=1:12, col.axis="red", labels=month.abb)
axis(2, at=seq(1,12,length=10), col.axis="red", labels=1:10, las=2)
axis(3, at=seq(1,12,length=7), col.axis="blue", cex.axis=0.7, tck=-0.01,
labels = c("Mon", "Tues", "Wed", "Thu", "Fri", "Sat", "Sun"))
axis(4, at=seq(1,12,length=11), col.axis="blue", cex.axis=0.7, tck=-0.01,
labels=seq(0, 1, 0.1), las=2)
```

5. 添加图例

legend 函数的具体用法如下所示:

```
legend(x, y = NULL, legend, col = par("col"), lty, pch, bty = "o",
bg = par("bg"), ncol = 1, horiz=FALSE, xpd = FALSE, title = NULL)
```

legend 函数参数:

图例位置设置:

3.4.3 保存图形

R 语言提供将图片输出到屏幕的 windows 函数和 X11 函数，其中 windows 函数用于 Windows 系统，

参数名称	参数解释
x, y	设置图例的位置（默认左上角位置），除使用x和y参数外，也可以使用字符"bottomright", "bottom", "bottomleft", "left", "topleft", "top", "topright", "right", "center"
legend	一个字符向量，表示图例中的文字
horiz	图例的排列方式，为FALSE（默认）时，图例垂直排列，为TRUE时，图例水平排列
ncol	图例的列数目，当horiz=TRUE时，该项无意义
pch	图例中点的样式，可取0-25，其中0-14为空心点，15-25为实心点；也可以直接通过pch="+"的方式定义点的样式。
lty	图例中线的样式，0表示不画线，1表示实线，2表示虚线，3表示点线
col	图例中点和线的颜色，令col等于对应颜色名称即可
bg	图例的背景颜色，令bg等于对应颜色名称即可。在bty参数为"n"时无效
bty	设置图例框的样式，取"o"（默认）时表示显示边框，取"n"时表示无边框
xpd	是否在作图区域外作图,默认FALSE，即不允许在作图区域外作图
title	设定图例的标题

图 23: 图例设置

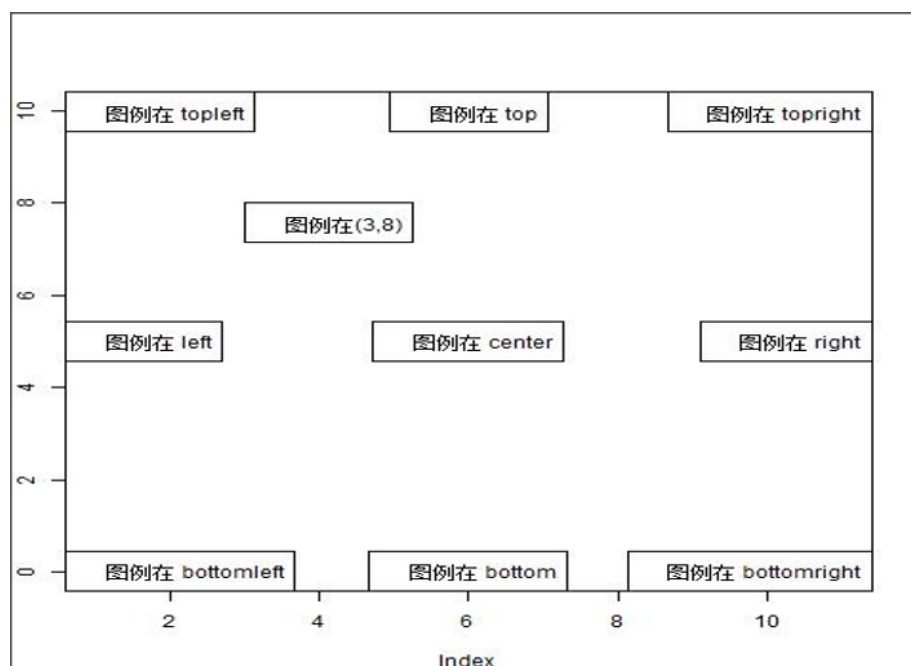


图 24: 图例位置设置

X11 函数用 UNIX 类型的系统的 X11 桌面系统。

`windows()` % 打开图形设备界面，UNIX 用 `X11()`

`plot(iris)` % 在打开的图形界面绘制散点矩阵图

选择“文件”菜单下的“另存为”可将图片以不同的文件形式输出到屏幕。

#保存为jpg格式

```
jpeg(filenamees = 'C:\\Users\\45543\\Desktop\\iris.jpg') # 保存的路径及文件名称、类型
```

```
plot(iris[, 1:4]) # 绘制图形
```

```
dev.off() # 关闭关联
```

如果希望将当前画布的图形保存到指定路径，可以使用 `win.metafile` 函数，

`bmp` 函数，`tiff` 函数，

`svg` 函数，`postscript` 函数，`png` 函数，`jpeg` 函数等将接下来的图形绘制到指定文件，并用 `dev.off` 函数结束关联。

3.4.4 作业

要求完成 7 种图的绘制，并对图进行正确设置坐标轴、刻度、图例等。

3.5 高级绘图

3.5.1 使用 lattice 包绘图

`lattice` 是由 Deepayan Sarkar 基于 `grid` 包的一套统计图形系统，它的图形设计理念来自于 Cleveland 的 Trellis 图形。

`lattice` 包通过栅栏（trellis）图形来对多元变量关系进行直观展示，为单变量和多变量数据

的可视化提供了一个全面的图形系统。一些用标准绘图很难实功能，`lattice` 却能很轻易地实现。

以 `mtcars` 数据集为例，绘制车身重量（`wt`）与每加仑汽油行驶的英里数（`mpg`）的散点图。

```
library('lattice')
```

```
xyplot(wt ~ mpg, data = mtcars, xlab = 'Weight', ylab = 'Miles per Gallon',
```

```
main ='lattice包绘制散点图')
```

1. 图形参数

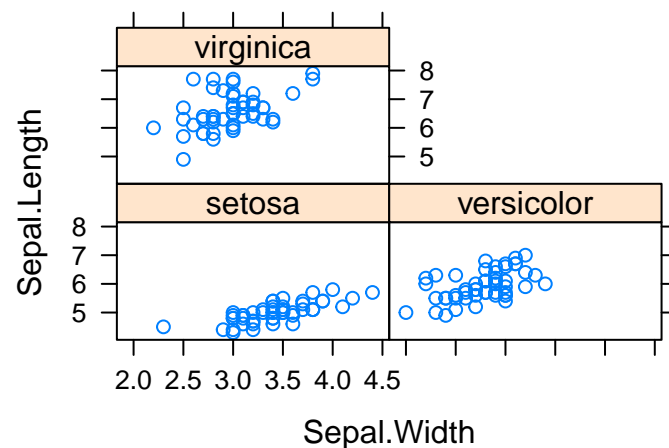
- 可使用 `trellis.par.get` 函数获取参数列表，`trellis.par.set` 函数则可以修改参数列表
- `names(trellis.par.get())` # 查看参数列表的名称
- `show.settings()` # 图形化显示所有参数
- lattice 图形参数是有层次的，可以将它们看作列表的列表，即可使用 `$` 对相关图形参数进行索引。
- 在 lattice 包，默认以不同颜色去区分不同类别数据。lattice 包中，可以通过管道符号 (`|`) 来添加条件变量 `v`，其表达方式如下所示。

```
graph_function(formula|v,data=,options)
```

例子：以 `iris` 数据集为例，以 `Species`（鸢尾花种类）为条件变量绘制 `Sepal.Length`（花萼长度）

与 `Sepal.Width`（花萼宽度）的散点图。

```
library(lattice)
attach(iris)
xyplot(Sepal.Length ~ Sepal.Width | Species)
```



```
detach(iris)
```

2. 条件变量

- 在栅栏图形中，单个面板要依据条件变量的各个水平来创建。
 - 设小写字母（x，y）代表数值型变量，大写字母（A，B）代表类别型变量（因子），在高级绘图函数中，表达式形式通常如下所示： $x \sim y \mid A + B$
 - 竖线左边的变量为主要变量，右边的变量为条件变量。主要变量将变量映射到每个面板的坐标轴上。
3. 面板函数在 lattice 包中，每个高级绘图函数都调用了一个默认的函数来绘制面板。默认函数：`panel.graph_function`
- 面板函数参数：

函数	描述
<code>panel.abline</code>	在面板的图表区域增加线
<code>panel.curve</code>	在面板的图表区域增加曲线
<code>panel.rug</code>	在面板上增加轴须
<code>panel.mathdensity</code>	给定分布函数，绘制概率分布图
<code>panel.average</code>	按照因子变量，绘制平均值
<code>panel.fill</code>	对面板填充具体的颜色
<code>panel.grid</code>	绘制网格线
<code>panel.loess</code>	增加一条光滑曲线
<code>panel.lmline</code>	为数据增加一条回归线
<code>panel.refline</code>	在面板的图表区增加一条线
<code>panel.qqmathline</code>	在样本和理论分布的25分位点和75分位点加一条线
<code>panel.violin</code>	绘制小提琴图，通常用于箱线图

图 25: 面板函数参数

使用 `xyplot` 函数绘制的散点图上添加回归线，光滑曲线，轴须和网格线，只需要将 `panel` 参数设置为一个整合了多个面板函数的函数即可。

```
library('lattice')
my_panel<- function(x,y){panel.lmline(x, y, col = "red", lwd = 1,
                                     lty = 2) # 为数据增加一条回归线
  panel.loess(x,y) # 增加一条光滑曲线
  panel.grid(h = -1, v = -1) # 绘制网格线
```

```

panel.rug(x, y) # 在面板上增加轴须
panel.xyplot(x, y)} # 绘制散点图
xyplot(mpg ~ wt, data = mtcars, xlab = "Weight", ylab = "Miles per Gallon",
       main = "Miles per Gallon on Weight", panel = my_panel)

```

4. 分组变量

通过添加条件变量 ($x \sim y \mid A + B$)，可以创建出各个水平下的面板。若想要把不同水平的图形结果叠加到一起，

则可以将变量设定为分组变量 (group 参数)，其具体用法如下所示。

```
graph_function(formula, data=, group=v)
```

例子：

以 iris 数据集为例，绘制 Sepal.Length（花萼长度）与 Sepal.Width（花萼宽度）的散点图，并根据 Species（鸢尾花种类）对数据进行分组。

```

library('lattice')
xyplot(Sepal.Length ~ Sepal.Width, group = Species, data = iris,
       pch = 1:3, col = 1:3, main = 'Sepal.Length VS Sepal.Width',
       key = list(space = "right", title = "Species", cex.title = 1, cex = 1,
                  text = list(levels(factor(iris$Species))),
                  points=list(pch = 1:3, col= 1:3)))

```

5. 图形组合

以 iris 数据集为例，在同一个画布上分别 Sepal.Length（花萼长度）与 Sepal.Width（花萼宽度）的散点图，

其中左图是以 Species（鸢尾花种类）为条件变量的栅栏图，右图是根据 Species（鸢尾花种类）分组的散点图。

```

library(lattice)
graph1 <- xyplot(Sepal.Length ~ Sepal.Width | Species,
                 data = iris, main = '栅栏图')
graph2 <- xyplot(Sepal.Length ~ Sepal.Width, group = Species, data = iris,
                 main = '散点图1')
graph3 <- xyplot(Petal.Length ~ Petal.Width, group = Species, data = iris,
                 main = '散点图2')

```

```
# split参数
plot(graph1, split = c(1,1,3,1))
plot(graph2, split = c(2,1,3,1), newpage = FALSE)
plot(graph3, split = c(3,1,3,1), newpage = FALSE)
# position参数, 绘图效果同上
plot(graph1, position = c(0, 0, 1/3, 1))
plot(graph2, position = c(1/3, 0, 2/3, 1), newpage = FALSE)
plot(graph3, position = c(2/3, 0, 1, 1), newpage = FALSE)
```

lattice 包中包含的基本绘图函数及相关绘图对象

函数名 (graph_function)	函数功能	绘制对象 (formula)
barchart	条形图	数组, 表达式, 矩阵, 数值型向量, 表格
dotplot	点图	数组, 表达式, 矩阵, 数值型向量, 表格
histogram	直方图	因子, 表达式, 数值型向量
densityplot	核密度图	表达式, 数值型向量
stripplot	带状图	表达式, 数值型向量
qqmath	Q-Q图	表达式, 数值型向量
qq	Q-Q图	表达式
bwplot	箱线图	表达式, 数值型向量
xyplot	散点图	表达式
splo	散点图矩阵	数据框, 表达式, 矩阵
levelplot	三维水平图	数组, 表达式, 矩阵, 表格
contourplot	三维等高线图	数组, 表达式, 矩阵, 表格
cloud	三维散点图	表达式, 矩阵, 表格
wireframe	三维曲面图	表达式, 矩阵

图 26: lattice 基本绘图函数

其通用函数格式如下所示:

```
graph_function(formula, data = , options)
```


3.5.2 使用 ggplot2 包绘图

ggplot2 的核心理念是将绘图与数据分离，数据相关的绘图与数据无关的绘图分离，按图层作图。

```
1.qplot qplot 的用法和 R base 包的 plot 函数很相似，其使用格式如下所示。
qplot(x, y = NULL, ..., data, facets = NULL, margins = FALSE, geom =
“auto”,
stat = list(NULL), position = list(NULL), xlim = c(NA,NA), ylim = c(NA,
NA),
log = ””, main = NULL,xlab = deparse(substitute(x)),
ylab = deparse(substitute(y)), asp = NA)
qplot 函数参数：
```

常见参数	描述
data	数据框（data.frame）类型；如果有这个参数，那么x，y的名称必需对应数据框中某列变量的名称
facets	图形/数据的分面，这是ggplot2作图比较特殊的一个概念，它把数据按某种规则进行分类，每一类数据做一个图形，所以最终效果就是一页多图
geom	图形的几何类型（geometry），这又是ggplot2的作图概念。ggplot2用几何类型表示图形类别，比如point表示散点图、line表示曲线图、bar表示柱形图等。
stat	图形的统计类型（statistics），这个更加特殊。直接将数据统计和图形结合，这是ggplot2强大和受欢迎的原因之一
position	可对图形或者数据的位置调整。相同数据的几何对象位置相同，是放在一个位置相互覆盖还是用别的排列方式：dodge为并排模式；fill为堆叠模式，并归一化为相同的高度；stack为纯粹的堆叠模式；jitter会在X和Y两个方向增加随机的扰动来防止对象之间的覆盖
margins	逻辑值，表示是否显示边界
...	其他参数与plot函数的参数类似

图 27: qplot 函数参数

```
library(ggplot2)
qplot(Species, Sepal.Length, data = iris, geom = "boxplot", fill = Species,
      main = "依据种类分组的花萼长度箱线图")
qplot(Sepal.Length, Sepal.Width, data = iris, colour = Species, shape = Species,
      main = "绘制花萼长度和花萼宽度的散点图")
```

2.ggplot2 包的语言逻辑
通过 plot 与 ggplot2 的实际例子对比来说明。

```
plot(iris$Sepal.Length, iris$Sepal.Width)
library(ggplot2)
ggplot(data= iris, aes(x = Sepal.Length, y = Sepal.Width)) + #绘制底层画布
geom_point(color = "darkred") #在画布上添加点
```

ggplot 的绘图有以下几个特点:

- 有明确的起始（以 ggplot 函数开始）与终止（一句语句一幅图）。
- ggplot2 语句可以理解为一句语句绘制一幅图，然后进行图层叠加，而叠加是通过“+”号把绘图语句拼接实现的。ggplot 函数的使用格式如下所示:

```
ggplot(data = NULL, mapping = aes(), ..., environment = parent.frame())
```

示例:

data = iris 时, mapping = aes(x = Sepal.Length, y = Sepal.Width) 表示将花萼长度作为 x 轴变量, 花萼宽度作为 y 轴变量。如果需要将 Species 映射到形状或者颜色属性, 可以添加参数 shape = Species 或者 colour = Species。

```
ggplot(data = iris, aes(x = Sepal.Length, y = Sepal.Width, colour = Species,
                        shape = Species)) # 底层画布
```

ggplot2 中可用的几何对象函数:

几何对象函数	描述
geom_abline	线：由斜率和截距指定
geom_area	面积图
geom_bar	条形图
geom_bin2d	二维封箱的热图
geom_blank	空的几何对象，什么也不画
geom_boxplot	箱线图
geom_contour	等高线图
geom_crossbar	Crossbar图（类似于箱线图，但没有触须和极值点）
geom_density	密度图
geom_density2d	二维密度图
geom_errorbar	误差线（通常添加到其他图形上，比如柱状图、点图、线图 etc）
geom_errorbarh	水平误差线
geom_freqpoly	频率多边形（类似于直方图）
geom_hex	六边形图（通常用于六边形封箱）
geom_histogram	直方图

图 28: ggplot 几何对象函数

geom_hline	水平线
geom_jitter	点、自动添加了扰动
geom_line	线
geom_linerange	区间，用竖直线表示
geom_path	几何路径，由一组点按顺序链接
geom_point	点
geom_pointrange	一条垂直线，线的中间有一个点(与Crossbar图和箱线图有关)
geom_polygon	多边形
geom_quantile	一组分位数线（来自分位数回归）
geom_rect	二维的长方形
geom_ribbon	彩虹图
geom_rug	触须
geom_segment	线段
geom_smooth	平滑的条件均值
geom_step	阶梯图
geom_text	文本
geom_tile	瓦片（即一个个的小长方形或多边形）

如果将数据定义在 ggplot 函数中，那么所有图层都可以共用这个数据；如果将数据定义在 geom_point 函数中，那么这个数据就只供这个几何对象使用。

统计类型 stat 是指对数据所应用的统计类型/方法，ggplot2 为每一种几何类型指定了一种默认的统计类型。

标尺设置

- `scale__continuous`: 将数据的连续取值映射为图形属性的取值。
- `scale__discrete`: 将数据的离散取值映射为图形属性的取值。
- `scale__identity`: 使用数据的值作为图形属性的取值。
- `scale__manual`: 将数据的离散取值作为手工指定的图形属性的取值。

示例:

随机从 iris 数据集的 150 个样本中抽取 100 个样本, 并绘制条形图反映 100 个样本中各个鸢尾花种类的数量情况。

```
set.seed(1234) # 设置随机种子
my_iris <- iris[sample(1:150, 100, replace = FALSE),] # 随机抽样
p <- ggplot(my_iris) + geom_bar(aes(x = Species, fill = Species))
p$scales # 查看p的标尺参数
p + scale_fill_manual(
  values = c("orange", "olivedrab", "navy"), # 颜色设置
  breaks = c("setosa", "versicolor", "virginica"), # 图例和轴要显示的分段点
  name = "my_Species", # 图例和轴使用的名称
  labels = c("set", "ver", "vir") # 图例使用的标签
```

在这个实际例子, 可以通过修改标尺参数做前后对比图, 进而理解标尺在 ggplot2 包中的作用。

ggplot2 使用技巧:

- 使用 `scale_color_manual` 或 `scale_color_brewer` 函数修改图形的颜色。
- ggplot2 默认的坐标系是笛卡尔坐标系,
- 分组绘图 (`facet_grid` 函数)
- 保存图片 (`ggsave(filename,width,height,...)`)

3.6 作业

1. 利用 p2p 数据，要求学生清洗数据，得到订单的最后统计数据。
2. 要求利用 lattice,ggplot2 分别画图（7 种图）

4 第四章参数估计假设检验的 R 语言应用

教学目的：

1. 熟悉参数估计与假设检验的理论知识；
2. 了解R软件参数估计与各类假设检验的流程；
3. 掌握各类参数估计与各类假设检验的编程基础特点。

教学重难点：

1. 重点：掌握各类参数估计与各类假设检验的R语言应用编程特点；
2. 难点：了解各类参数估计与各类假设检验的R语言应用流程；

4.1 参数估计理论知识

4.2 参数估计的 R 语言应用

4.3 假设检验的 R 语言应用

5 第五章线性回归分析的 R 语言应用

教学目的：

1. 熟悉线性回归的定义以及约束条件；
2. 了解线性回归模型的R语言编程汇编语言；
3. 掌握线性回归模型的R语言编程的关键问题的解决。

教学重难点：

- 1.重点：掌握线性回归模型的R语言编程的关键问题的解决；
- 2.难点：了解线性回归模型的R语言编程汇编语言；

5.1 一元线性回归分析基本理论

5.2 一元线性回归分析的 R 语言应用

5.3 多元线性回归分析基本理论

5.4 多元线性回归分析的 R 语言应用

5.5 稳健线性回归分析的 R 语言应用

6 第六章多重共线性的 R 语言应用

教学目的：

- 1.熟悉多重共线性的定义；
- 2.掌握多重共线性的理论知识；
- 3.掌握多重共线性的控制条件在R语言上的编程。

教学重难点：

- 1.重点：掌握多重共线性的控制条件在R语言上的编程；
- 2.难点：了解多重共线性的控制条件；

6.1 多重共线性的概念

6.2 多重共线性的后果

6.3 多重共线性的识别和检验

6.4 消除多重共线性的方法

6.5 多重共线性的 R 语言应用

7 第七章异方差问题的 R 语言应用

教学目的：

1. 熟悉异方差的定义以及本质特点；
2. 了解异方差的控制手段；
3. 掌握异方差控制在R语言编程上的特点。

教学重难点：

1. 重点：掌握异方差控制在R语言编程上的特点；
2. 难点：异方差的R软件控制实现；

7.1 异方差的概念

7.2 异方差的原因与后果

7.3 异方差的识别检验

7.4 异方差的消除方法

7.5 异方差的 R 语言应用

8 第八章自相关问题人 R 语言应用

教学目的：

- 1.熟悉自相关的定义以及本质特点；
- 2.了解自相关的控制手段；
- 3.掌握自相关控制在R语言编程上的特点。

教学重难点：

- 1.重点：掌握自相关控制在R语言编程上的特点；
- 2.难点：了解自相关的控制手段；

8.1 自相关的概念

8.2 自相关的产生原因

8.3 自相关的识别与检验

8.4 自相关性诊断与消除的 R 语言应用

9 第九章时间序列分析

教学目的:

1. 熟悉时间序列的定义以及ARIMA模型的适用条件;
2. 了解时间序列与ARIMA模型的逻辑特点;
3. 掌握时间序列与ARIMA模型在R语言编程上的特点。

教学重难点:

1. 重点: 掌握时间序列与ARIMA模型在R语言编程上的特点;
2. 难点: 了解时间序列与ARIMA模型的逻辑特点;

9.1 单位根检验

9.2 ARMA 模型介绍

9.3 ARMA 模型的确定

9.4 ARMA 模型的确定与检验

10 第十章时间序列分析 garch 模型的 R 语言应用

教学目的:

1. 熟悉时间序列的定义以及GARCH模型的适用条件；
2. 了解时间序列与GARCH模型的逻辑特点；
3. 掌握时间序列与GARCH模型在R语言编程上的特点。

教学重难点：

1. 重点：掌握时间序列与GARCH模型在R语言编程上的特点；
2. 难点：了解时间序列与GARCH模型的逻辑特点；

10.1 GARCH 模型的介绍

10.2 ARCH 效应检验

10.3 GARCH 模型的 R 语言应用