统计计算

李丰　　康雁飞

2020-03-17

献给李慷小朋友

目录

表格

插图

简介

R 软件的 bookdown 扩展包是 R Markdown 的增强版，支持自动目录、文献索引、公式编号与引用、定理编号与引用、图表自动编号与引用等功能，可以作为 LaTeX 的一种替代解决方案，在制作用 R 进行数据分析建模的技术报告时，可以将报告文字、R 程序、文字性结果、表格、图形都自动地融合在最后形成的网页或者 PDF 文件中。

Bookdown 使用的设置比较复杂，对初学者不够友好。这里制作了一些模板，用户只要解压缩打包的文件，对某个模板进行修改填充就可以变成自己的中文图书或者论文。Bookdown 的详细用法参见https://bookdown.org/yihui/bookdown/，在李东风的《统计软件教程》[1]也有部分介绍。

一些常用功能的示例在 `0101-usage.Rmd` 文件中，用户可以在编辑器中打开此文件参考其中的做法。

Bookdown 如果输出为网页，其中的数学公式需要 MathJax 程序库的支持，用如下数学公式测试浏览器中数学公式显示是否正常：

$$\text{定积分} = \int_a^b f(x)\,dx$$

如果显示不正常，可以在公式上右键单击，选择 “Math Settings–Math Renderer”，依次使用改成 “Common HTML”，“SVG” 等是否可以变成正常显示。PDF 版本不存在这样的问题。

[1] http://www.math.pku.edu.cn/teachers/lidf/docs/Rbook/html/_Rbook/index.html

第一章　R 基础

1.1　安装设置

使用 RStudio 软件完成编辑和转换功能。在 RStudio 中，安装 bookdown 等必要的扩展包。

本模板在安装之前是一个打包的 zip 文件，在适当位置解压（例如，在 `C:/myproj` 下），得到 `MathJax`, `Books/Cbook`, `Books/Carticle` 等子目录。本模板在 `Books/Cbook` 中。

为了利用模板制作自己的中文书，将 `Books/Cbook` 制作一个副本，改成适当的子目录名，如 `Books/Mybook`。

打开 RStudio 软件，选选单 “File - New Project - Existing Directory”，选中 `Books/Mybook` 子目录，确定。这样生成一本书对应的 R project (项目)。

为了将模板内容替换成自己的内容，可以删除文件 `0101-usage.Rmd`，然后将 `1001-chapter01.Rmd` 制作几份副本，如 `1001-chapter01.Rmd`, `2012-chapter02.Rmd`, `3012-chapter03.Rmd`。各章的次序将按照前面的数值的次序排列。将每个`.Rmd` 文件内的 `{#chapter01}`, `{#chapter02-sec01}` 修改能够反映章节内容的标签文本。所有的标签都不允许重复。参见本模板中的 `0101-usage.Rmd` 文件。

后面的 §1.3.1 和 §1.3.2 给出了将当前的书转换为网页和 PDF 的命令，复制粘贴这些命令到 RStudio 命令行可以进行转换。

1.2 编写自己的内容

1.2.1 文档结构

除了 index.Rmd 以外，每个.Rmd 文件是书的一章。每章的第一行是用一个井号（#）引入的章标题。节标题用两个井号开始，小节标题用三个井号开始。标题后面都有大括号内以井号开头的标签，标签仅用英文大小写字母和减号。

1.2.2 图形自动编号

用 R 代码段生成的图形，只要具有代码段标签，且提供代码段选项 fig.cap="图形的说明文字"，就可以对图形自动编号，并且可以用如\@ref(fig:label) 的格式引用图形。如：

```
plot(1:10, main="程序生成的测试图形")
```

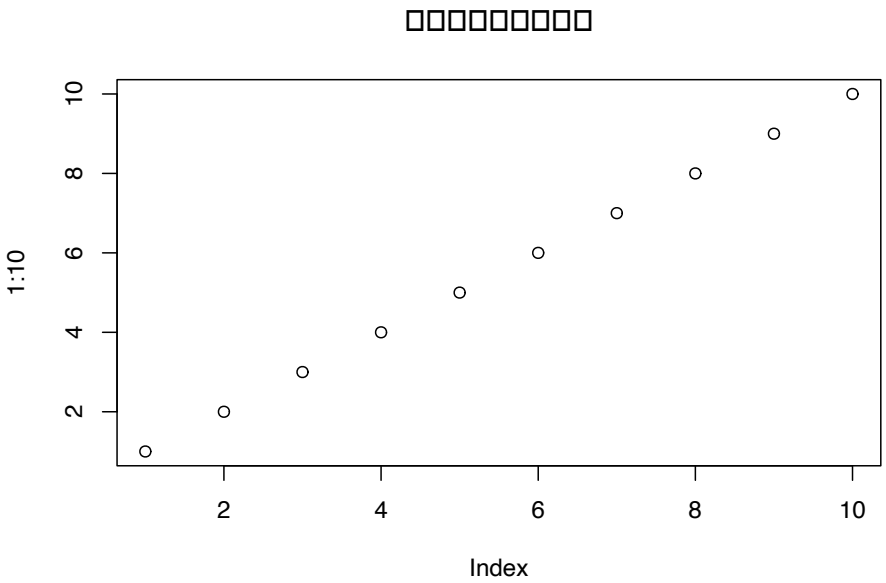


图 1.1: 图形说明文字

引用如：参见图1.1。引用中的 fig: 是必须的。

表 1.1: 表格说明文字

自变量	因变量
1	1
2	4
3	9
4	16
5	25
6	36
7	49
8	64
9	81
10	100

在通过 LaTeX 转换的 PDF 结果中，这样图形是浮动的。

1.2.3 表格自动编号

用 R 代码 `knitr::kable()` 生成的表格，只要具有代码段标签，并且在 `knitr::kable()` 调用时加选项 `caption="表格的说明文字"`，就可以对表格自动编号，并且可以用如`\@ref(tab:label)` 的格式引用表格。如：

```
d <- data.frame("自变量"=1:10, "因变量"=(1:10)^2)
knitr::kable(d, caption="表格说明文字")
```

引用如：参见表1.1。引用中的 `tab:` 是必须的。

在通过 LaTeX 转换的 PDF 结果中，这样的表格是浮动的。

1.2.4 数学公式编号

不需要编号的公式，仍可以按照一般的 Rmd 文件中公式的做法。需要编号的公式，直接写在`\begin{align}` 和`\end{align}` 之间，不需要编

号的行在末尾用\nonumber 标注。需要编号的行用 (\#eq:mylabel) 添加自定义标签，如

$$\begin{aligned}\Sigma =&(\sigma_{ij})_{n\times n}\\ =&E[(\boldsymbol{X}-\boldsymbol{\mu})(\boldsymbol{X}-\boldsymbol{\mu})^T]\end{aligned} \tag{1.1}$$

引用如：协方差定义见(1.1)。

1.2.5 文献引用与文献列表

将所有文献用 bib 格式保存为一个.bib 文献库，如模板中的样例文件 mybib.bib。可以用 JabRef 软件来管理这样的文献库，许多其它软件都可以输出这样格式的文件库。

为了引用某一本书，用如：参见 (**?**)。

被引用的文献将出现在一章末尾以及全书的末尾，对 PDF 输出则仅出现在全书末尾。

1.3 转换

1.3.1 转换为网页

用如下命令将整本书转换成一个每章为一个页面的网站，称为 gitbook 格式：

```
bookdown::render_book("index.Rmd",
  output_format="bookdown::gitbook", encoding="UTF-8")
```

为查看结果，在 _book 子目录中双击其中的 index.html 文件，就可以在网络浏览器中查看转换的结果。重新编译后应该点击“刷新”图标。

在章节和内容较多时，通常不希望每次小修改之后重新编译整本书，这时类似如下的命令可以仅编译一章，可以节省时间，缺点是导航目录会

变得不准确。命令如：

```
bookdown::preview_chapter("1001-chapter01.Rmd",
  output_format="bookdown::gitbook", encoding="UTF-8")
```

单章的网页可以通过网络浏览器中的“打印”功能，选择一个打印到 PDF 的打印机，可以将单章转换成 PDF 格式。

1.3.2 生成 PDF

如果想将 R Markdown 文件借助于 LaTeX 格式转换为 PDF，需要在系统中安装一个 TeX 编译器。现在的 rmarkdown 包要求使用 tinytex 扩展包以及配套的 TinyTeX 软件包[1]，好像不再支持使用本机原有的 LaTex 编译系统，如果不安装 tinytex，编译为 PDF 格式时会出错。TinyTeX 优点是直接用 R 命令就可以安装，更新也由 R 自动进行，不需要用户干预。但是，安装时需要从国外网站下载许多文件，有因为网络不畅通而安装失败的危险。

为了安装 R 的 tinytex 扩展包和单独的 TinyTeX 编译软件，应运行：

```
install.packages('tinytex')
tinytex::install_tinytex()
```

安装过程需要从国外的服务器下载许多文件，在国内的网络环境下有可能因为网络超时而失败。如果安装成功，TinyTeX 软件包在 MS Windows 系统中一般会安装在 `C:\Users\`用户名`\AppData\Roaming\MikTex` 目录中，其中“用户名”应替换成系统当前用户名。如果需要删除 TinyTeX 软件包，只要直接删除那个子目录就可以。

为了判断 TinyTeX 是否安装成功，在 RStudio 中运行

```
tinytex::is_tinytex()
```

[1] https://yihui.name/tinytex/

结果应为 TRUE, 出错或者结果为 FALSE 都说明安装不成功。在编译 pdf_book 时，可能会需要联网下载 LaTeX 所需的格式文件。

Bookdown 借助操作系统中安装的 LaTeX 编译软件 TinyTeX 将整本书转换成一个 PDF 文件，这需要用户对 LaTeX 有一定的了解，否则一旦出错，就完全不知道如何解决。用户如果需要进行 LaTeX 定制，可修改模板中的 preamble.tex 文件。

转换为 PDF 的命令如下：

```
bookdown::render_book("index.Rmd",
  output_format="bookdown::pdf_book", encoding="UTF-8")
```

在 _book 子目录中找到 CBook.pdf 文件，这是转换的结果。CBook.tex 是作为中间结果的 LaTeX 文件，如果出错可以从这里查找错误原因。

转换 PDF 对于内容多的书比较耗时，不要过于频繁地转换 PDF，在修改书的内容时，多用 bookdown::preview_chapter 和转换为 gitbook 的办法检验结果。定期地进行转换 PDF 的测试。每增加一章后都应该试着转换成 PDF 看有没有错误。命令如：

```
bookdown::preview_chapter("1001-chapter01.Rmd",
  output_format="bookdown::gitbook", encoding="UTF-8")
```

1.3.3 上传到网站

如果书里面没有数学公式，则上传到网站就只要将 _book 子目录整个地用 ftp 软件传送到自己的网站主目录下的某个子目录即可。但是，为了支持数学公式，就需要进行如下的目录结构设置：

1. 设自己的网站服务器目录为/home/abc，将 MathJax 目录上传到这个目录中。
2. 在/home/abc 中建立新目录 Books/Mybook。
3. 将 _book 子目录上传到/home/abc/Books/Mybook 中。
4. 这时网站链接可能类似于 http://dept.univ.edu.cn/~abc/Books/Mybooks/_book/in
   具体链接地址依赖于服务器名称与主页所在的主目录名称。

如果有多本书，MathJax 仅需要上传一次。因为 MathJax 有三万多个文件，所以上传 MathJax 会花费很长时间。

第二章　求根算法

2.1　介绍

2.2　牛顿法

牛顿法（Newton Method，or Newton-Raphson method）是最常用且高效的求根算法之一。假设函数 $f(x)$ 可导并一阶导 $f'(x)$ 连续，且 x^* 为 $f(x)=0$ 的根。设 x_0 是对 x^* 的一个好的估计，令 $x^*=x_0+h$。因为 x^* 是 $f(x)=0$ 的根而且 $h=x^*-x_0$，所以 h 描述了 x_0 距离真实根的距离。

因为 h 很小，我们通过线性逼近可以得到：

$$0=f(x^*)=f(x_0+h)\approx f(x_0)+hf'(x_0),$$

所以除非 $f'(x_0)$ 接近于 0，我们得到：

$$h\approx -\frac{f\left(x_0\right)}{f'\left(x_0\right)}.$$

从而，

$$x^*=x_0+h\approx x_0-\frac{f\left(x_0\right)}{f'\left(x_0\right)},$$

我们可以把这个新的估计 x_1 做为对 x^* 更好的估计：

$$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)}.$$

如果我们继续更新，就可以得到 x_2：

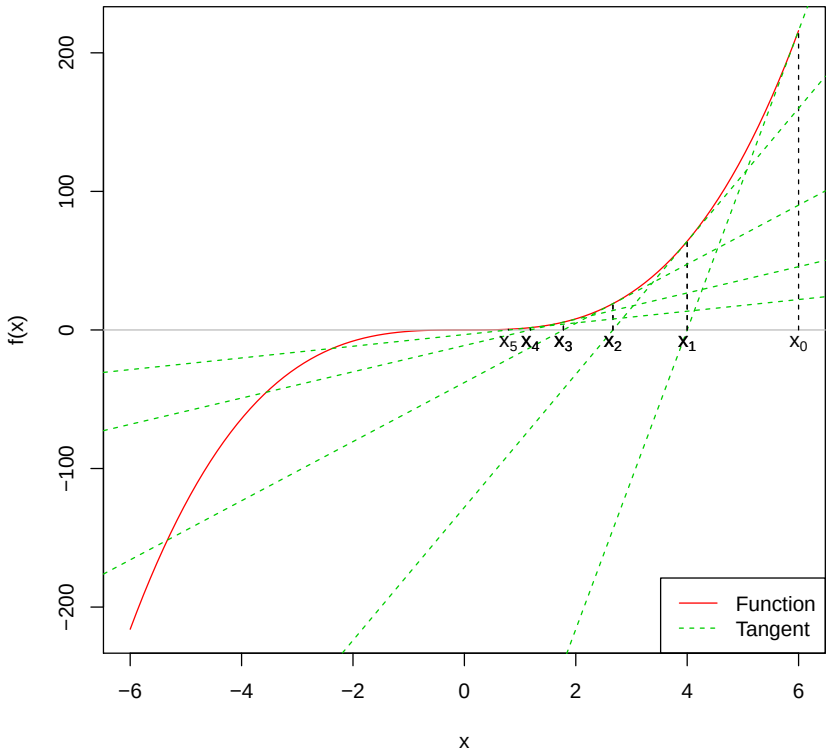
$$x_2 = x_1 - \frac{f(x_1)}{f'(x_1)}.$$

按照这种思路继续迭代下去，就是牛顿法求根。牛顿法更一般的写法如下。

牛顿法

1. 选择初始猜测点 x_0，设置 $n = 0$。
2. 按照以下迭代过程进行迭代：
$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}. \tag{2.1}$$
3. 计算 $|f(x_{n+1})|$。
    1. 如果 $|f(x_{n+1})| \leq \epsilon$，停止迭代；
    2. 否则，返回第 2 步

下面我们来看一下牛顿法更直观的解释。



现在我们将牛顿法在 R 中实现，以下 `newton.root()` 函数中的第一个参数为待求根的函数 $f(x)$，因为牛顿法中每一次迭代都需要求 $f(x)$ 和 $f'(x)$ 在 x_n 处的取值，因此我们定义了 R 函数 `ftn()`，它的输入参数为 $f(x)$ 的表达式及 x 的取值，输出为 $f(x)$ 和 $f'(x)$。具体代码如下。

```
newton.root <- function(f, x0 = 0, tol = 1e-09, max.iter = 100) {
    x <- x0
    fx <- ftn(f, x)
    iter <- 0
    # xs用来保存每步迭代得到的x值
    xs <- list()
    xs[[iter + 1]] <- x
    # 继续迭代直到满足停止条件
```

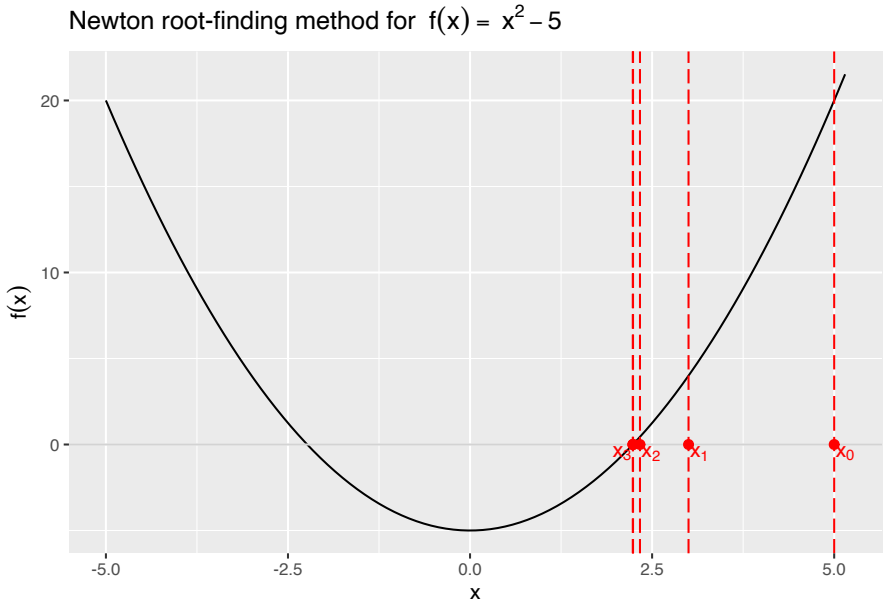
```
    while ((abs(fx$f) > tol) && (iter < max.iter)) {
        x <- x - fx$f/fx$fgrad
        fx <- ftn(f, x)
        iter <- iter + 1
        xs[[iter + 1]] <- x
        cat("迭代第", iter, "次，x = ", x, "\n")
    }

    # output depends on success of algorithm
    if (abs(fx$f) > tol) {
        cat("算法无法收敛\n")
        return(NULL)
    } else {
        cat("算法收敛\n")
        return(xs)
    }
}

ftn <- function(f, x) {
    df <- deriv(f, "x", func = TRUE)
    dfx <- df(x)
    f <- dfx[1]
    fgrad <- attr(dfx, "gradient")[1, ]
    return(list(f = f, fgrad = fgrad))
}
```

例 2.1. 求 $f(x) = x^2 - 5$ 的根。

```
f <- expression(x^2 - 5)
roots <- unlist(newton.root(f, 5))
```

```
## 迭代第 1 次，x =  3
## 迭代第 2 次，x =  2.333333
## 迭代第 3 次，x =  2.238095
```

```
## 迭代第 4 次, x =  2.236069
## 迭代第 5 次, x =  2.236068
## 算法收敛
```



“‘