Quarto 中文书稿写作手册

文学化统计编程

汤银才

2024-08-23

目录

前言

这本小册子是在我 2021 年 7 月在 github 上出版的《Bookdown 中文书稿写作手册》的基础上改篇过来的，基本的章节结构与内容基本保持不变.

Bookdown 是谢益辉基于 R Markdown 开发出来的系列文学化写作工具之一, 主要是针对 R 语言开发出来，而 Quarto Markdown 是新一代的 R Markdown, 可支持包括 R, Python, Julia, Oberservable 等在内的多种语言, 因此作为 Quarto 应用场景之一的 Quarto Book 可视为 Bookdown 的拓展或新的一代, 因此这本小册子会在多种编程语言和 Quarto 本身的特性上尽可能体现出来.

这本小册子可视为一个写中文书稿的 `quarto` 模板，也是中文 `quarto book` 写作的一本说明书，其中汇总了书稿中几大核心要素的写作技巧. 除了原有的 `bookdown` 参考书, 我主要参考了 Quarto 网站Quarto Guide中Quarto Books的相关内容及其中提及的在线电子书. `Quarto Gallery` 提供的电子书在不断增加，在此罗列一些，也表示对 Posit 及这些电子书的作者表示敬意和感谢.

1. Hadley Wickham, R for Data Science, 2E
2. Wes McKinney, Python for Data Analysis, 3E
3. Garrett Grolemund, Hands on Programming with R
4. Bernd Bischl, Raphael Sonabend, Lars Kotthoff, Michel Lang （eds.), Applied Machine Learning Using mlr3 in R, mlr3book
5. Rohan Alexander, Telling Stories with Data: With Applications in R and Python
6. Jeffrey Heer, Dominik Moritz, Jake VanderPlas, and Brock Craft, Visualization Curriculum
7. Michael Dorman, Anita Graser, Jakub Nowosad, Robin Lovelace, Geocomputation with Python

8. V. Lyubchich and Y. R. Gel, Time Series Analysis: Lecture Notes with Examples in R

9. Damien Dotta, Cookbook Polars for R: A side-by-side comparison of polars, R base, dplyr, tidyr and data.table packages.

10. Kevin Heavey, Modern Polars: A side-by-side comparison of the Polars and Pandas libraries.

11. Malcolm Barrett, Lucy D'Agostino McGowan, Travis GerkeCausal Inference in R

12. 汤银才, Bookdown 中文书稿写作手册, 2021-07-21.

13. 李东风，R 语言教程，第 22 章：Quarto 格式文件, 2023-07-27.

第一章　引言

这是第 一 章的内容，回顾中国图书发展的历史及最新趋势 (R Core Team, 2021; Xie, 2015).

1.1　中国图书出版的变化

中国的图书出版经历一些折腾，方正系统一度非常流行，但现在看来是非常失败的，主要是它在解决中文字库和公式排版等的同时没有解决兼容性、标准化及便利性等问题，其本质上想实现在其自创的中文系统中将 TeX 命令做一个映射以实现数学公式排版，同时完成格式的定制.

中国学术界也经历了一些折腾，如中科院张林波研究员等开发的 CCT 系统和华东师范大学肖刚与陈志杰等老师开发的天元系统，它们是 TeX 系统汉化版，较好地解决了汉字生成与调用，但因没有考虑普适性或可拓展性而像方正系统一样随着 CJK 的逐渐成熟而逐渐被抛弃，均退出了历史舞台. 而同期中科院吴凌云博士等在普及 TeX 的同时开发的 TeX 中文套餐 C TeX 相当成功，主要的工作是会对汉字排版的 ctex 宏包及套餐的开发 (后者类似于针对苹果 OS 系统的 MacTeX, 但后期的更新显得太慢)，并对三个主流的文档类 book, article, report 进行了定制，推出了相应的中文文档类 ctexbook, ctexart, ctexrep, 由此避免了传统的基于 CJK 宏包需要的大量定制，同时保证了与原有 TeX 系统的兼容性. 这样我们始终可以使用跨平台的 TeXLiVe 进行排版或各类模板的开发，例如各个出版社的图书模板、各个期刊的模板、各高校的硕士和博士毕业论文模板等, 现在这样的模板都开源到 Overleaf 上.

TeX 的出现[1]，而且始终屹立不倒的原因是什么? 第一个原因是它解决了一个 Word 之类文字编辑系统的痛点，即从所见所得 (WYSIWYG, What you see is what you get) 到所想所得

[1]是伟大的 Knuth 创造了排版的奇迹

(WYTIWYG), 即通过一些 TeX 命令集构成一个完整的编程语言，由它完成一个封闭的体系，具有类似 C 语言一样非常强大的开发功能，由此形成了后来的 latex, miktex, latex2.09, luatex, xetex 等 TeX 编译引擎，它们在充分利用电脑系统资源的同时实现高质量输出需要的精度.

TeX 屹立不倒的另一个原因是浮动对象的处理，即包括公式，表格、图形、页码、章节、文献、定理等的标签化与引用，实现文档内部的自由跳转，结合 Acrobat Reader 这样强大的 pdf 阅读器的支持，使得读者的阅读体验得到大幅改善，并为图书的电子化奠定了基础.

随着数据科学这一新兴学科的出现， 开源的 R 和 Python (还有正在逐步流行的 Julia) 编程语言越来越强大. 为了增加这类图书的可读性，需要将代码较完整地呈现在读者面前，并且要求代码的即时可复现能力，即数据的变化，其分析的结果 (包括图形和表格) 也随之发生变化并同步呈现出来. 这就是现在逐步流行的文学化编程 (literate programming), 它实现上最早也是 TeX 的鼻祖 Knuth 提出的, 后来被谢益辉得到重视并广泛推广，并通过 Rstudio 和 R Markdown 和现在的 Quarto Markdown 传递给 R 的用户. 我们姑且把这种将写作与数据分析相结合的统计分析称为文学化统计编程 (literate statistical programming). 它在为数据科学爱好者带来便利的同时也通过 Bookdown 或 quarto 为论文、分析报告和图书的写作和电子化带来了极大的好处，越来越多的包括网页版电子书在内的文档出现在 (https://github.com/) 等网站上.

这本小册子主要考虑书稿类文档的写作，它可以包括图书、毕业论文和分析报告，在些统称为书稿. 现在写书选择什么类型的模板，下面我们来作进一步的探索与比较.

1.2 统计类图书的核心要素

统计或数据科学类图书的排版除普通图书的页面及文字风格等静态元素外，核心要素体现在浮动的对象上，使得图书的阅读体验更好地发挥出来，即在不同页面之间快速切换、跟踪、搜索，以及必要的 R 和 Python 等代码以语法高亮方式显示，必要时同步将执行结果呈现出来.

1. **章节标题**是浮动的，最主要用于书签的生成;
2. **公式**是浮动的，这是数学、统计等理科书的特点，公式引用必不可少;
3. **图形**是浮动的，统计图形作为可视工具，在说明数据或展示分析结果时经常会引用相应的图形;

4. **表格**是浮动的，通常是原始数据或统计分析的结果以表格形式展示出来，它们可能被多次在不同的章节中引用;

5. **定理**是浮动的，这里定理是指与之相关的一大类，包括常用的定理、引理、推论、命题、例子等，它们在文中也会被反复引用;

6. **文本**可以设置浮动标签后被引用，最为常见的是图形与表格的题图 (caption) 通过文本方式来引用;

7. **文献**是浮动的，这在是谈及前人的已有工作、成果比较或进行综述时经常要引用大量已经发表的论文、图书、会议报告等.

TeX 有一套成熟的浮动对象的排版方式，通过给浮动对象打标签 (label)，然后引用 (ref). `Quarto Book`(或之前的 `Bookdown`) 思路一样，但比 TeX 的处理稍复杂些 (可能因我们还没有习惯引起). 我们在后面章节中分别举例说明.

1.3 统计数据分析类图书模板的选择

统计数据分析类图书既有理论或原理的讲解，又会有一些案例分析，包括这些案例分析实现的代码. 我们可以考虑的模板主要有三种类型.

1.3.1 基于纯 TeX 模板

全世界 90% 的书是由 TeX 排版的，包括硕士和博士毕业论文模板，这要感谢鼻祖 Knuth! 开源成就了 TeX !

如果仅仅是统计理论方面的书集，这显然是最好的选择，因为高质量公式的排版离不开 TeX. 基于 TeX 的排版存在三个明显的缺陷或不足:

1. 大量的 TeX 命令需要记忆;

2. 对于代码的排版非常不便，特别是 `R` 或 `Python` 代码执行后的输出，尤其是图形与表格;

3. 代码以 `listing` 等包来呈现, 无法实时呈现代码运行的结果，不符合文学化编程的要求.

1.3.2 R, Markdown 与 TeX 的结合

数据科学时代更注重文学化统计编程，**代码伴随**是这类图书的特点，自 Springer 出版 R 系列统计图书后，这种风格成为新趋势，大大方便了数据科学爱好者" 便学习便练习" 的学习方式. 针对代码伴随，早期对这类图书有二个解决方案：

1. Sweave/knitr + R

本质上它是在 TeX 嵌入 R 代码块，并由 R 在后台运行后将结果也嵌入到 TeX 中，再由 TeX 的编译引擎生成 pdf. 这个方案的基本沿用 TeX 的方式，它仅解决了上面提到的第二个问题. 在数据科学时代，报告的快速生成成为新的要求，效率优先! 随着 knitr 的出现 Sweave 退出舞台.

2. R Markdown + Mathjax/ TeX

Markdown 作为一种轻量级的标记语言成为网页作为文字主要载体的互联网时代首先的写作工作，但它显然不适合数学与统计类论文或图书的撰写，但 knitr 和 pandoc 的出现使不同风格的内容整合与转换成为可能，而不同风格的内容各有善长的工具实现，作为统计类专业论文或图书类文档主要的内容有：

a. 文字，由 markdown 完成

b. 公式，由 TeX 完成

c. 代码，由 R (或 Python) 完成

要说明的是，在网页端，TeX 的实现可由 Mathjax 来完或渲染 (转化或生成标准的公式)，见第 四 章说明.

1.3.3 从 Markdown、R Markdown 到 Quarto

Markdown 是一种轻量级标记语言， 其语法简单易用， 容易上手. 人们可以通过 Markdown 给纯文本文档添加格式化元素，轻松转换为其他格式文件，适用不同场景，从而使创作者专注于内容写作，而非格式排版. Markdown 文档的扩展名为 .md. Markdown 的语法结构十分简单，使用者稍微阅读一下相关文档和案例，便可动手写作.

R Markdown 是基于 Markdown 的扩展，专门为 R 语言和数据分析而设计. 它允许在 Markdown 文档中嵌入 R 代码块，并提供了方便的方式来执行和显示代码的结果. 通过 R Markdown，用户可以编写结合了文本、代码和结果展示的动态报告、文档、演示文稿等. R

Markdown 文件可以通过 RStudio 等工具进行编辑和转换，得到包括 HTML、PDF、Word 文档等多种输出格式. R Markdown 文档的扩展名为 `.rmd`.

过去 10 年，R 的功能与生态得到不断完善, 得益于 Hadley Wickham 和谢益辉等开发的许多优秀包的出现，如 `knitr`, `kableExtra`, `ggplot2`, `dplyr` 等, 以及基于 `rmarkdown` 适用于新的特定应用场景的程序包, 如 `bookdown`，`pagedown`, `rticle` 等的出现和普及, `Rstudio` 作为一个 IDE 也变得越来越成熟与强大，正在成为一个非常优秀的论文、幻灯片及图书等撰写与出版系统 (PUB).

Quarto 则是定位 R Markdown 的下一代产品，由 Posit [2]（原 RStudio 的开发团队）开发，是一个开源的数据分析与科技出版系统. Quarto 不仅支持 R 生态系统的扩展，还支持更多的编程语言（如 `Python`、`Julia` 和 `Observable` 等），同时支持近 40 种输出格式 (包括 `html`,`pdf`, `ePub`, `Word`, `Asciidoc` 等)，从而在更大范畴内为科技类（不仅仅限于统计类）出版活动提供解决方案.

`Quarto` 已经完成能够实现 `bookdown` 的书稿排版涉及的定制及核心要素的排版，而且考虑到对更多编程语言的支持, 我们完成可以将原来基于 `bookdown` 的写作转换到 `Quarto` 上来. 若非必要, 我们这本手册之后也用 `Quarto Book` 来指基于 `Quarto` 的书稿格式排版.

1.3.4 格式转换流程

Quarto 根据以下工作流程导出各类格式文档：

其中，`Knitr` 是 Quarto 中使用的一个 R 包，负责将 `.qmd` 文件转换为普通 Markdown 格式（`.md` 文件），以便后续使用 `Pandoc` 进行格式转换。在转换前，会运行 `.qmd` 文件中的代码块，将得到的文字、表格、图形结果再插入到文档中. 代码块可以在 R 的 `knitr` 包支持下运行，也可以在 Jupyter 软件的支持下运行.

`Pandoc` 是一个开源的文档转换工具，用于将一个格式的文档转换为另一种格式，广泛

[2] Posit 由 RStudio 更名而来，于 2021 年开始着手打造新一代的 `R Markdown`，称为 `Quarto`. 2022 年 12 月，`Rstudio` 作为公司于 2021 年作出了重大战略调整，不仅将公司名称改为 `Popsit`，而且将更多的开发放在应用更为广泛的 `VS Code` 上，但同时保证对 `Rstudio` 和 `Python Notebook` 的支持. 该公司专注于数据科学的资源集成，近年来一直在布局打通大数据科学的创作与分享.

应用于科技写作领域。Pandoc 支持多种输入格式（如 Markdown、HTML、LaTeX 等）和输出格式（如 HTML、PDF、Word、ePub 等），并提供了丰富的选项和灵活的配置来满足各种转换需求。特别地, 对于 pdf 输出, 由 pandoc 完成由 md 转化为 TeX , 并由 laTeX 编译生成 pdf; 对于 HTML 输出, 由 pandoc 由 md 转化为 html, 其中的数学公式由 Mathjax 或 KaTeX 完成渲染.

想在 VS Code 中进行格式转换，可以在命令行下运行如下命令：

```
quarto render --to all
```

1.3.5 多格式图书出版的实现

在科技高度发达的互联系时代，读者使用的媒介基本有三类：较为专业的电脑，较为轻便的平板 (电脑) 和全功能的智能手机. 前者以 pdf 类图书为主呈现给读者，同时可以完成标注等工作；后者以文字型的电子图书为主，消磨时间为主；而平板的使用者逐渐成为电子类图书的新势力，包括 pdf 和 epub 之类的电子书. 基于 Quarto 创建的 Book 类, 注重不同类型读者的媒体使用的差异，并很好地实现统一编写与差异化输出. 目前 Quarto Book 可以一键生成三类图书:

1. gitbook，可自由出版在 git pages 上
2. ePub, 发表到大量的电子图书平台上
3. pdf, 正规的图书出版公司以电子或纸质形式出版

> **注释**
>
> Quarto 与 TeX 做到了无缝衔接，我们可以在 Markdown 文档中嵌入和管理 TeX 代码，而且我们还可以通过 TeX 或 $CSS/SCSS$ 的高度定制实现符合 PDF、HTML 或 ePub 的的输出。

第二章　Quarto 书稿速览

这是第 二 章的内容，概要性地讲解基于 `quarto` 拓展包进行图书排版的整体思路与实现方式. (R Core Team, 2021; Xie, 2015)

2.1　Quarto 的结构

Quarto 作为基于 Markdown 的写作工具，旨在简化和增强出版流程，它提供了一种灵活且易于使用的方式来创建和发布各种类型的文档。

Quarto 包括以下几个关键组成部分：

1. **Quarto Markdown 文件**：Quarto 使用 Markdown 作为主要的文档格式，文件扩展名为 `.qmd`。

2. **YAML 配置文件**：Quarto 使用 `_quarto.yml` 文件作为项目的配置文件。这个 YAML 配置文件就是一个文本文件，它允许用户指定输出格式、模板、外观等选项，自定义 Quarto 文档的元数据。

3. **输出格式**：Quarto 使用 Pandoc 转换文档格式，将 Markdown 文档输出为 HTML、PDF、ePub、docx 等格式，用户可以根据需要配置相应选项，选择输出格式。

4. **模板**：Quarto 使用模板来定义文档的整体布局、样式和体例等，用户可以选择预定义的模板，也可以自定义自己的模板，使用 HTML、CSS 或 SCSS、LaTeX 等方式来控制文档的形式。

5. **资源**：在 Quarto 中，你可以包含各种类型的资源，如图片、代码、数据、交互列表等，这些资源可以与 Markdown 文件一起存储在同一目录中，方便管理和引用。

通过将 Markdown 文件、YAML 配置文件、模板和各类资源整合到一起，Quarto 提供了一种完整的文档构建和发布解决方案，它使得创建专业、美观的多格式文档变得更加简单和高效。

Quarto 目前发行的版本为 1.4.554。

2.2 关于 Quarto Book

Quarto Markdown 作为新一代的 R Markdown，是另一个增强 markdown 格式的扩展，使得 qmd 格式可以支持书箱编写所需的章节、数学公式、定理、表格、图形、文献等浮动对象的自动编号，并按设定的标签在正文中引用。在 quarto 的管理下一本书的内容可以按章节分解成多个 qmd 文件，其中可以包含可执行的 R/Python/Julia/observable 等代码，代码生成的统计汇总结果、表格、图形可以自动插入到生成的内容中，而且其中的表格和图形也是浮动。书的输出格式包括支持 gitbook 格式的网页图书，也可以经 LaTeX 编译器转换的 PDF 图书，还可以生成 ePub 等格式的电子书。

目前 Quarto 支持在 RStudio、VS Code(甚至 Python Jupyter) 的集成环境中编辑、管理和生成图书.

关于浮动对象标签的设定与应用见第 2.6 节的汇总表格及后续各章的介绍与示例.

2.3 书的基本设置

一本用 bookdown 管理的书，一般放置在某个子目录下，并作为一个 RStudio 项目 (project) 用 RStudio 管理。该目录中的所有的文本文件都要使用 UTF-8 编码。

2.3.1 index.Rmd 文件

一本 bookdown 书，一般都需要有一个 index.Rmd 文件，这是最后生成的网站的主页的原始文件. 这个文件的开始是 YAML 元数据部分, 进行全书的有关设置，包括标题、作者、日期及影响全书的一些选项等，放在三个减号组成的两行之间。然后写一些这本书的说明，如书的前言部分。index.Rmd 中 YAML 元数据部分的一个例子如下：

```
title: "bookdown书稿模板"
```

```
author: "汤银才"
date: "2024-06-15"
documentclass: book
bibliography: [myrefs.bib]
biblio-style: apa
link-citations: yes
site: bookdown::bookdown_site
description: "bookdown写书体验非常好."
```

注意:

1. site 选项很重要，一定要有，site: bookdown::bookdown_site 使得 RStudio 能辨认这是一个 bookdown 图书项目，从而为其生成一键编译的 build 快捷方式；

2. 在 bookdown 项目中与 index.Rmd 同级的所有.Rmd 文件都自动作为书的一章，其好处是作者可以任意地增删章节，编译整本书时将按照文件名的字典序依次进行。实际上，也可以在 _output.yml 文件中设置一项 rmd_files，列出所有需要作为一章的文件，并以列出次序编译；

3. 在 index.Rmd 的元数据中也可以指定一些 LaTeX 的选项，例如

```
fontsize: 12pt
indent: true
linestretch: 2.0
link-citations: yes
colorlinks: yes
lot: true
lof: true
geometry:
- a4paper
- tmargin=2.5cm
- bmargin=2.5cm
- lmargin=3.5cm
- rmargin=2.5cm
```

2.3.2 _bookdown.yml 文件

一个 bookdown 图书项目除了 index.Rmd 文件之外，还有一些设置文件从 index.Rmd 文件的元数据部分抽离出来。一个是 _bookdown.yml 文件，它存放与整本书的处理有关的 YAML 元数据。例如

```
book_filename: "bookdown-template"
new_session: yes
language:
  label:
    fig: "图 "
    tab: "表 "
    thm: '定理'
    def: '定义'
    exm: '例'
    proof: '证明: '
  ui:
    chapter_name: ["第 ", " 章"]
```

其中 new_session: true 设置很重要，这使得每一个 Rmd 文件中的 R 程序都在一个单独的 R 会话中独立地运行，避免了不同 Rmd 文件之间同名变量和同名标签的互相干扰。book_filename 是最终生成的 PDF 图书或者 ePub 电子书的主文件名。language 下可以定制一些与章节名、定理名等有关的名称。

2.3.3 _output.yml 文件

另一个设置文件是 _output.yml，用于图书输出格式的设置[1]，本小册子的 _output.yml 文件内容如下

```
bookdown::gitbook:
  css: css/style.css
  split_by: chapter
  includes:
    in_header: _header.html
```

[1]这部分内容也可以包含在 index.Rmd 的元数据部分

```
  config:
    toc:
      collapse: subsection
      scroll_highlight: yes
      before: |
        <li><a href="./">bookdown排版模板</a></li>
      after: |
        <li><a href="https://bookdown.org" target="blank">本书由 bookdown 强力驱动</
a></li>
    download: [pdf, epub]
    edit: https://github.com/yihui/bookdown-chinese/edit/master/%s
    sharing:
      github: yes
      facebook: no
  pandoc_args: [ "--csl", "apa-6th-edition-no-ampersand.csl" ]
bookdown::pdf_book:
  includes:
    in_header: latex/preamble.tex
    before_body: latex/before_body.tex
    after_body: latex/after_body.tex
  keep_tex: yes
  dev: "cairo_pdf"
  latex_engine: xelatex
  citation_package: biblatex
  template: latex/template.tex
  toc_depth: 3
  toc_unnumbered: no
  toc_appendix: yes
  quote_footer: ["\\begin{flushright}", "\\end{flushright}"]
  pandoc_args: [ "--top-level-division=chapter" ]
bookdown::epub_book:
  stylesheet: css/style.css
  pandoc_args: [ "--csl", "apa-6th-edition-no-ampersand.csl" ]
```

它分别对 gitbook、pdf_book 和 epub_book 三种输出格式设置了一些输出选项。其中一些选项是通过文件形式给出设置的，我们再补充说明一下。

1. style.css 是自定义的 CSS 显示格式，在 gitbook 和 epub_book 中使用;

2. _header.html 是插入了一部分个性化的 HTML 代码，其内容将出现在每个生成的 HTML 文件的 head 部分。我们在此文件中给出了使用本地的 Mathjax 实现数学公式离线显示的设置，内容为

```
<script type="text/x-mathjax-config">
MathJax.Hub.Config({
  jax: ["input/TeX","output/SVG"],
  extensions: ["tex2jax.js","MathMenu.js","MathZoom.js"],
  TeX: {
    Macros: {
      bm: ["{\\boldsymbol #1}",1],
    },
    extensions: ["AMSmath.js","AMSsymbols.js","noErrors.js","noUndefined.js"]
  }
});

</script>
<script type="text/javascript"
  src="http://127.0.0.1/MathJax/MathJax.js">
</script>
```

其中 http://127.0.0.1/MathJax/是本地服务器上 Mathjax 的位置。有关 Mathjax 的本地安装与启动可参考第附录 A 章的介绍;

3. apa-6th-edition-no-ampersand.csl 是 gitbook 和 epub_book 处理文献使用的风格文件;

4. preamble.tex 是处理 (编译)bookdown 文件经 pandoc 转化生成的 tex 文件时导言区需要额外的宏包和设置;

5. before_body.tex 是 tex 书稿类正文前面的设置，最基本的是

```
\frontmatter
```

6. after_body.tex 是 tex 书稿类正文之后的设置，最基本的是

```
\backmatter
```

7. `template.tex` 是针对`bookdown`编译经`pandoc`转化生成的`tex`文件时的模板，由它生成

其他选项说明：

1. split_by: chapter: 按章分割书稿；
2. collapse: subsection: 目录中隐藏子节 (仅显示二级标题)；
3. scroll_highlight: yes: 目录滚动时高亮显示，便于定位；
4. keep_tex: yes: 保留中间生成的 tex 源文件，便于查错；
5. dev: "cairo_pdf": 使用 cairo_pdf() 生成 LaTeX 编译需要的图片文件;
6. latex_engine: xelatex: TeX 文件的排版引擎为 XeTeX，针对 UTF-8 编码;
7. citation_package: biblatex: 文献引用库指定为 biblatex, 另一个为 natbib;
8. toc_depth: 3: 目录提取至三级标题;
9. toc_unnumbered: no: 指定目录编号;
10. toc_appendix: yes: 附录添加到目录中.

2.4 章节结构

如前所述, 除了 index.Rmd 文件，项目中每个.Rmd 文件都作为一章，其第一行是以一个 # 号和空格开头的一级标题。

每一章可以有若干节与子节，分别用 markdown 的二级标题 (二个 # 开始) 和三级标题 (三个 # 开始) 编写。bookdown 的章、节、子节标题单独成一行，其后可以添加标签, 章节的标签是标题后加空格，然后是大括号内以 # 号开头的标签，如

```
# 引言 {#intro}

## 关于bookdown {#bookdown}
```

bookdown 中有二个特殊的标题：

1. 部分

   内容相近的章节可以作为一个”部分”。为此，在一个部分的第一个章节文件的章标题前面增加一行，以 `#（PART）`开头，以 `{-}`结尾，例如

```
# （PART）bookdown中的浮动对象 {-}
```

2. 附录

   一本书的最后可以有附录，附录的章节将显示为 `A.1, B.1` 这样的格式。为此，在附录章节的第一个文件开头加如下的第一行标题行：

```
# （APPENDIX）附录 {-}


# biblatex介绍 {#biber}
```

2.5 书的编译

在 `index.Rmd` 或者 `_bookdown.yml` 中设置 `site: bookdown::bookdown_site` 后，RStudio 就能识别这个项目是一个 `bookdown` 项目，这时 RStudio 会有一个 `Build` 按钮，其中有 `Build book` 快捷图标，从下拉菜单中选择一个输出格式（包括 `gitbook`、`pdf_book`、`epub_book`），就可以编译整本书 (见图 2.1)。

经 `build` 编译生成的图书默认保存在 `_book` 子目录中[2]。

1. 对 `gitbook` 格式 (即 HTML 网页格式)，编译完成后会弹出一个预览窗口，点击”Show in new window “按钮可以将内容在操作系统默认的网络浏览器中打开。我们也可以用其他浏览器 (建议使用 Google chrome 浏览器) 打开 `_book` 子目录中的 `index.html` 文件来查看 `gitbook` 格式的图书。

2. 对于 `pdf_book` 格式，如果成功编译[3]，也会弹出一个 PDF 预览窗口。可以在 `_book` 子目录中找到这个 PDF 文件。

3. 对于 `epub_book` 格式，如果成功编译，会在操作系统默认的 ePub 软件 (如苹果电脑的 book) 中打开，并在 `_book` 子目录中找到这个 ePub 文件。

[2]可以在 `_bookdown.yml` 中设置 `output_dir` 项改变图书保存的子目录.

[3]需要 LaTeX 支持，建议安排 `TeXLive`，也可以仅安装谢益辉为 `rmarkdown` 开发的 `tinytex`

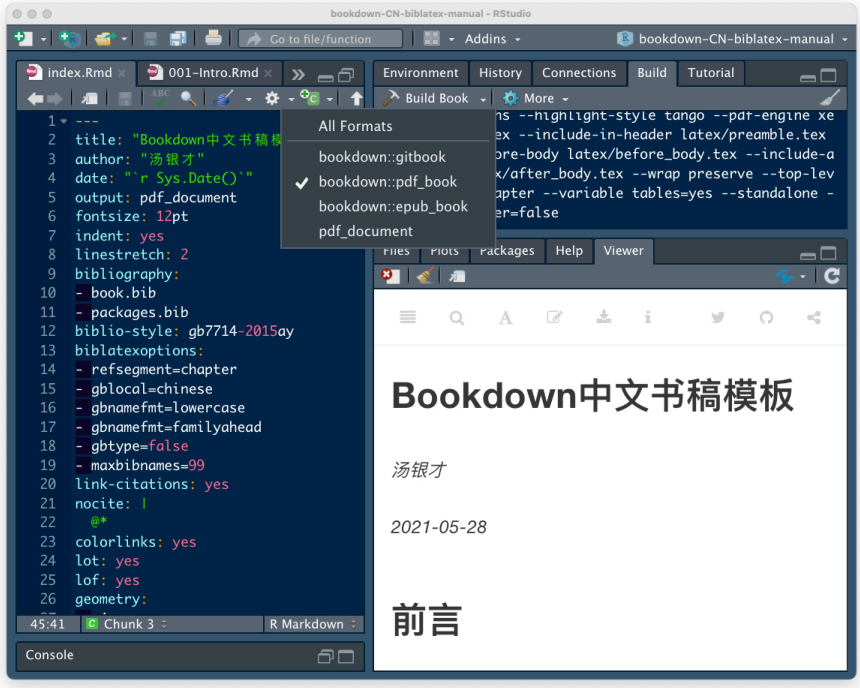


图 2.1: R Bookdown 编译界面.

2.6 浮动对象标签与引用汇总

浮动对象	标签设置	引用格式
标题	`{#sec-label}`	`@sec-label`
公式	`{#eq-label}`	`(@eq-label)`
图形	`label="label"`	`\@ref(fig:label)`
表格	`label="label"`	`\@ref(tab:label)`
定理	`label="label"`	`\@ref(prefix:label)`
文本	`(ref:label)`	`` `(ref:label)` ``
文献	`biblabel`	`@biblabel`

注：

1. 定理泛指定理类，包括定理 (thm)、引理 (lem)、推论 (cor)、命题 (prp)、设想 (cnj)、定义 (def)、例子 (exm)、习题 (exr) 等，其中括号中是引用时的前缀 (prefix)；
2. 文本标签在单独一行中设定，可用在表格与图形的 `caption` 中引用，即在 `fig.caption`, `tab.caption` 选项的设置中引用；
3. 定理类环境标签前缀的汉化可在 `_bookdown.yml` 中通过 `language` 进行[4]，例如

```
language:
  label:
    fig: "图 "
    tab: "表 "
    thm: '定理'
    def: '定义'
    exm: '例'
```

R Core Team, 2021. R: A Language and Environment for Statistical Computing[M/OL]. Vienna, Austria: R Foundation for Statistical Computing. https://www.R-project.org/.

Xie Y, 2015. Dynamic Documents with R and knitr[M/OL]. 2nd 版本. Boca Raton, Florida: Chapman; Hall/CRC. http://yihui.name/knitr/.

[4]图形与表格也可同理汉化

第三章　Quarto 书稿中的章节标题

我们在第 三 章讲述章节标题的设置、标签与引用.

3.1　章节标题

章节标题用遵从 `markdown` 的规则，用 `#` 设置，

- 一级标题用一个井号 `#`, 在 `quarto` 书稿中表示章, 相当于 TeX 中的`\chapter{}`
- 二级标题用二个井号 `##`, 在 `quarto` 书稿中表示节, 相当于 TeX 中的`\section{}`
- 三级标题用三个井号 `###`, 在 `quarto` 书稿中表示子节, 相当于 TeX 中的`\subsection{}`

还可以有更深的标题设置.

3.2　章节标题标签的设定与引用

Quarto 书稿中章节标题标签可在标题后用 `{#sec-label}` 来设定，引用方式为 `@sec-label`. 例如

```
第 @sec-sections 章 第 @sec-sec3-2 节讨论标题标签的设定与引用.
```

显示为：

第 三 章第 3.2 节讨论标题标签的设定与引用.

第四章　Quarto 书稿中的公式与定理

这是第 四 章的内容, 讲述浮动对象定理与公式的标签与引用.

4.1　公式标签的设定与引用

QUarto 借助 mathjax(或 katex) 处理数学公式的渲染; 尽管可通过联网由 cdn 上的 mathjax.js 进行渲染, 但速度随因公式的增加, 渲染变得很慢, 甚至出错。mathjax 的本地化是提速的主要解决方案, 详见附录 A 介绍;

4.1.1　{#eq-label} 标签

Quarto 中无论是单行公式还是多行公式, 均通过公式无标号的公式环境 (一对 $$) 实现, 其后添加标签, 格式为 {#eq-label}, 其中 eq 是关键字, 例如

```
$$
  \left(k\right) = \binom{n}{k} p^k\left(1-p\right)^{n-k}
$$ {#eq-binom}
```

显示为单行公式

$$f\left(k\right) = \binom{n}{k} p^k\left(1-p\right)^{n-k} \tag{4.1}$$

对于多行公式可以采用 aligned 环境, 但目前仅可对整个公式组设置标签, 例如

```
$$
\begin{aligned}
g(X_{n}) &= g(\theta)+g'({\tilde{\theta}})(X_{n}-\theta) \notag \\
```

```
\sqrt{n}[g(X_{n})-g(\theta)] &= g'\left({\tilde{\theta}}\right)
  \sqrt{n}[X_{n}-\theta ] (\#eq:align)
\end{aligned}
$$ {#eq-align}
```

显示为

$$g(X_n) = g(\theta) + g'(\tilde{\theta})(X_n - \theta)$$
$$\sqrt{n}[g(X_n) - g(\theta)] = g'\left(\tilde{\theta}\right)\sqrt{n}[X_n - \theta]$$

在这种情形下, 公式的引用采用 `@eq-label`, 例如上面的二个公式可引用为：公式 (4.1) 和公式 (4.2).

4.1.2 \label 标签

- 为了和 LaTeX 一样使用 `\label` 和 `\ref` 或 `\eqref` 去设定与引用公式标签，我们需要在上面创建的 `mathjax.html` 文件中，添加如下脚本:

```
<script>
  MathJax = {
    tex: {
      tags: 'ams',  // should be 'ams', 'none', or 'all'
    }
  };
</script>
```

- 我们可以将 `equation` 和 `aligned` 环境结合起来输入多行公式，对整个公式组设置一个 `\label`，例如：

```
\begin{equation}
\begin{aligned}
g(X_{n}) &= g(\theta)+g'({\tilde{\theta}})(X_{n}-\theta)\\
\sqrt{n}[g(X_{n})-g(\theta)] &= g'\left({\tilde{\theta}}\right)
  \sqrt{n}[X_{n}-\theta ]
\end{aligned}\label{eq-1}
```

```
\end{equation}
```

显示为

$$\begin{aligned} g(X_n) &= g(\theta) + g'(\tilde{\theta})(X_n - \theta) \\ \sqrt{n}[g(X_n) - g(\theta)] &= g'\left(\tilde{\theta}\right)\sqrt{n}[X_n - \theta] \end{aligned} \tag{4.2}$$

- 我们还可以用 `align` 等环境输入多行公式，每行公式设置一个 `\label`，例如：

```
\begin{align}
a^n &=  a\cdot a \cdot a \cdot \ldots \label{eq-2}\\
a^n &= p \label{eq-3}
\end{align}
```

显示为

$$a^n = a \cdot a \cdot a \cdot \ldots \tag{4.3}$$

$$a^n = p \tag{4.4}$$

在这种情形下，我们可以用和 TeX 一样的方式来引用公式，引用方式为 `\eqref{label}`，例如：公式 (4.2) 和 (4.4).

注释

`{#eq-label}` 和 `\label` 标签有各自适用的应用场景.

1. 在 Markdown 转换 (render) 为 PDF 时, Quarto 先将 Markdown 转换为 TeX，再利用 TeX 系统编译输出 PDF. 在此过程中，`{#eq-label}` 标签被自动转换为`\label` 标签，因此两种标签会在 PDF 中合并编号;
2. 在 Markdown 转换 (render) 为 HTML 时, `{#eq-label}` 标签会转换为静态的 `\tag` 标签，而 `\label` 标签保持不变，因此会出现两种标签相互独立、分别编号的情况;
3. 在 Markdown 转换 (render) 为 ePub 时, `{#eq-label}` 标签会被保留，而 `\label` 标签无法显示.

因此，我们不建议混用 `{#eq-label}` 和 `\label` 这两种标签，尤其在多格式出版中，为尽量保证不同格式文件内容的一致性，应优先考虑使用 `{#eq-label}` 标签，暂时舍弃 `\label` 标签以及多行公式的逐行引用，将这个问题留给 Quarto 在将来的升级版本中解决.

4.2 定理标签的设定与引用

这里我们给出几个 Quarto 书稿中定理类环境的例子.

引理 4.1. *带标签的引理.*

定理 4.1 (无限群)**.** *带标签与名字的定理.*

证明. 这里是定理的证明部分. □

定义 4.1. 带标签的定义.

例 4.1. 带标签的例子.

定理类引用的示例: 4.1, 定义 4.1 定理 4.1.

4.3 数学公式的扩展

有些公式无法用 TeX 中包的命令来实现，例如粗体数学符号，尽管在 TeX 中有个 `bm` 包在数学环境下通过`\bm{\alpha}` 来实现`\boldsymbol{\alpha}` 的功能，但在 `html` 下需要给 `mathjax` 做个 TeX 宏 (`macro`)`bm`[1]:

```
TeX: {
  Macros: {
    bm: ["{\\boldsymbol #1}",1],
  },
}
```

此时由 `$\bm\alpha$`，出来的效果为 $\bm\alpha$.

[1]配置在 `MathJax.Hub.Config` 下进行，具体参见 Mathjax 技术文档说明

有关数据公式的标签与应用可参考mathjax 官方文档, `Mathjax 2.7` 的本地化安装参考附录 A 介绍.

第五章　Quarto 书稿中的图形的处理

第 五 章将通过具体的示例展示浮动对象图形的标签与引用.

5.1　由 R 生成单个图形示例

这是最为常见的情形, 只涉及一个 R 的绘图主命令.

```
#| label: fig-fig5-1
#| fig: TRUE
#| echo: FALSE
#| fig.cap: "`iris`数据集`Petal.Length ~ Species`的箱线图."
boxplot(Petal.Length ~ Species, data = iris,
        ylab = "Petal length", xlab = "Species", col = "gray")
```

显示为

> **注释**
>
> 图形的标签由 R 代码块中的 `label: fig-xxx` 给出, 其中 `fig` 是关键字.

5.2　由 R 生成两个图形并置或堆叠

在 R 的代码块选项中设置 `fig-subcap: true` 或 `fig-subcap` 加上子图的图例, 并添加 `layout-ncol: 2` 就可获得二个子图的并置. 例如下面的代码

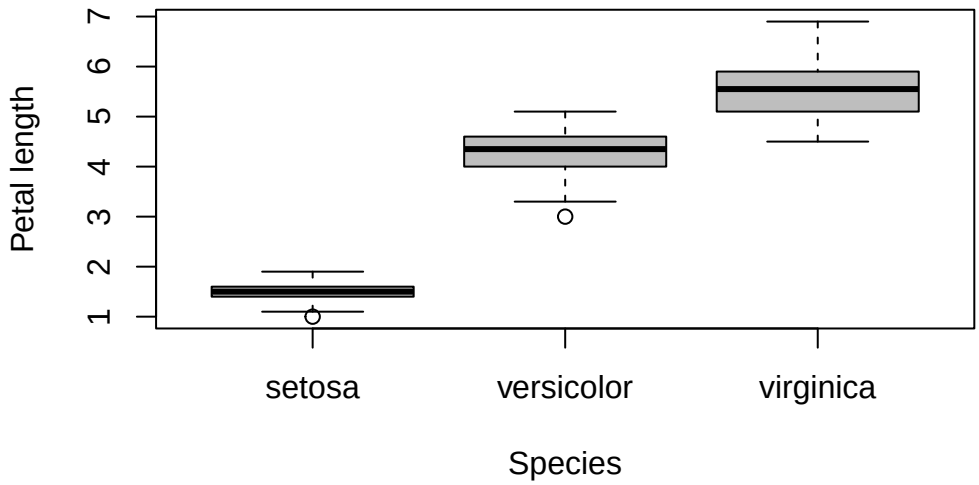


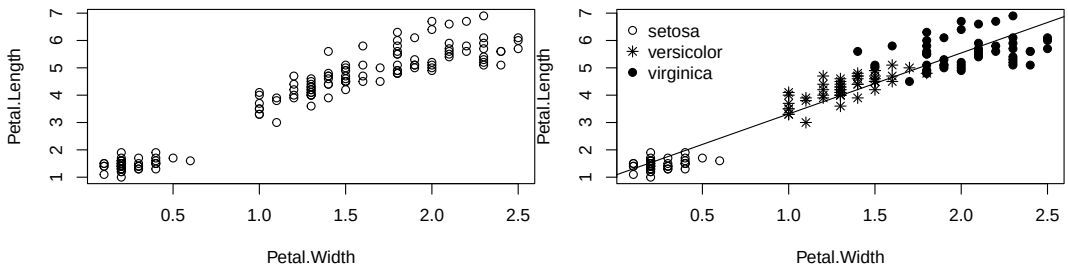
图 5.1: iris 数据集 Petal.Length ~ Species 的箱线图.

```
#| label: fig-fig5-2
#| fig-cap: "`iris`数据集`Petal.Length ~ Species` 的散点图. 右侧的图像中散点类型通过`Spe
#| fig-subcap:
#|   - " 散点图"
#|   - " 添加回归线"
#| layout-ncol: 2
plot(Petal.Length ~ Petal.Width, data = iris)
plot(Petal.Length ~ Petal.Width, data = iris,
     pch = c(21, 8, 19)[as.numeric(iris$Species)])
legend("topleft", pch = c(21, 8, 19),
       legend = levels(iris$Species), box.lty = 0)
lm1 <- lm(Petal.Length ~ Petal.Width, data = iris)
abline(lm1)     # 在散点图中添加拟合线
```

在 html 正常的浏览中得到两个左右并置的图形.

```
plot(Petal.Length ~ Petal.Width, data = iris)
plot(Petal.Length ~ Petal.Width, data = iris,
     pch = c(21, 8, 19)[as.numeric(iris$Species)])
legend("topleft", pch = c(21, 8, 19),
       legend = levels(iris$Species), box.lty = 0)
lm1 <- lm(Petal.Length ~ Petal.Width, data = iris)
```

```
abline(lm1)      # 在散点图中添加拟合线
```



(a) 散点图 (b) 添加回归线

图 5.2: iris 数据集 Petal.Length ~ Species 的散点图. 右侧的图像中散点类型通过 Species 因子的水平给出, 见图例. 直线为数据集拟合线性模型的结果.

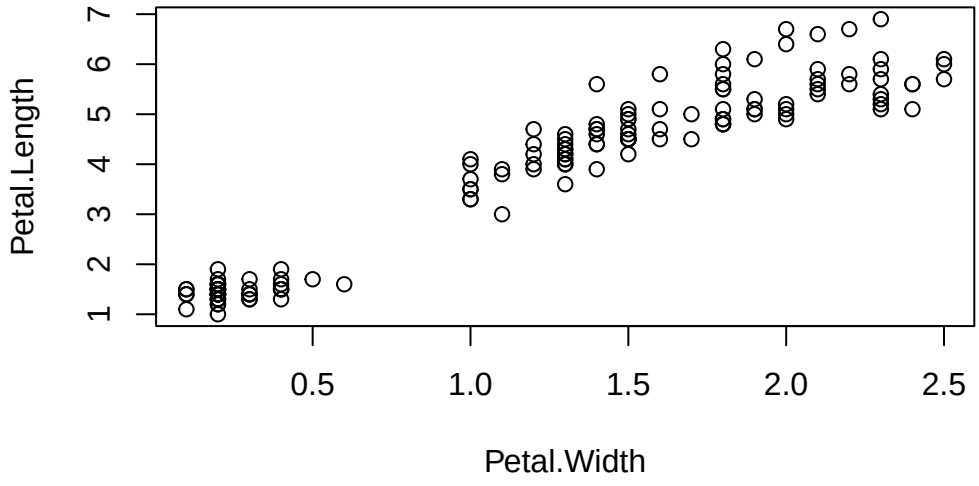
类似地, 通过选项 layout-nrow: 2 就可将二个子图堆叠起来. R 代码

```
#| label: fig-fig5-3
#| fig-cap: "`iris`数据集`Petal.Length ~ Species` 的散点图. 下图的图像中散点类型通过`Sp
#| fig-subcap:
#|   - " 散点图"
#|   - " 添加回归线"
#| layout-nrow: 2
plot(Petal.Length ~ Petal.Width, data = iris)
plot(Petal.Length ~ Petal.Width, data = iris,
     pch = c(21, 8, 19)[as.numeric(iris$Species)])
legend("topleft", pch = c(21, 8, 19),
       legend = levels(iris$Species), box.lty = 0)
lm1 <- lm(Petal.Length ~ Petal.Width, data = iris)
abline(lm1)      # 在散点图中添加拟合线
```

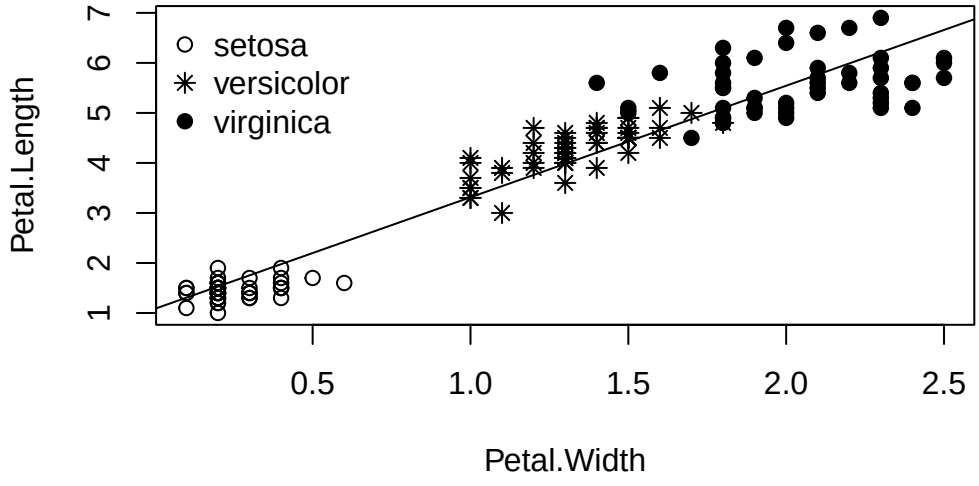
在 html 正常的浏览中得到两个上下堆叠的图形.

```
plot(Petal.Length ~ Petal.Width, data = iris)
plot(Petal.Length ~ Petal.Width, data = iris,
     pch = c(21, 8, 19)[as.numeric(iris$Species)])
legend("topleft", pch = c(21, 8, 19),
       legend = levels(iris$Species), box.lty = 0)
lm1 <- lm(Petal.Length ~ Petal.Width, data = iris)
```

```
abline(lm1)     # 在散点图中添加拟合线
```



(a) 散点图



(b) 添加回归线

图 5.3: iris 数据集 Petal.Length ~ Species 的散点图. 下图的图像中散点类型通过 Species 因子的水平给出，见图例. 直线为数据集拟合线性模型的结果.

5.3 单个静态图形的插入

在 Quarto 中插入本地图形可使用命令. 代码

```
#| label: fig-fig4-4
#| fig.align: 'center'
#| out.width: '70%'
#| fig.cap: 'R logo'
knitr::include_graphics("figures/Rlogo.png")
```

给出了 R 的 logo 图.

```
knitr::include_graphics("figures/Rlogo.png")
```

图 5.4: R logo

5.4 静态图形的并置

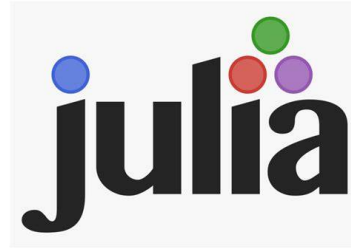
```
::: {#fig-program layout-ncol="3"}
![R](figures/Rlogo.png){#fig-R}

![Python](figures/python.png){#fig-python}

![Julia](figures/julia.png){#fig-julia}
```

```
数据科学的编程语言
:::
```

得到 `R`, `Python` 和 `Julia` 三个 logo 的并置.

(a) R

(b) Python



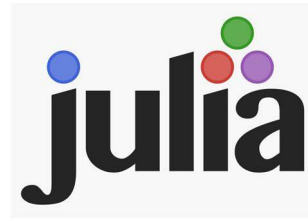
(c) Julia

图 5.5: 数据科学中的编程语言

> **注释**
>
> 静态图形的插入我们可以使用 `knitr` 程序包中的命令 `include_graphics()`. 例如上面的代码等价于

```
knitr::include_graphics("figures/Rlogo.png")
knitr::include_graphics("figures/python.png")
knitr::include_graphics("figures/julia2.png")
```

(b) Python



(c) Julia

(a) R

图 5.6: 数据科学中的编程语言

5.5 图形引用

图形引用通过 `R` 代码块的标签引用, 并带前缀 `fig-`, 例如

```
图 @fig-fig5-2 和图 @fig-fig5-3 为两个图的并置与堆叠.
```

输出为:

图 5.2 和图 5.3 为两个图的并置与堆叠.

第六章　Quarto 书稿中的表格的处理

第 六 章将通过具体的示例展示浮动对象 “表格” 的标签与引用技巧.

6.1　表格示例 1：由数据生成表格

代码

```
#| label: tbl-mtcars
#| tbl-cap: "mtcars 数据的前 5 行."
library(knitr)
kable(head(mtcars[, 1:5], 10))
```

输出为

```
library(knitr)
kable(head(mtcars[, 1:5], 10))
```

表 6.1: mtcars 数据的前 5 行.

	mpg	cyl	disp	hp	drat
Mazda RX4	21.0	6	160.0	110	3.90
Mazda RX4 Wag	21.0	6	160.0	110	3.90
Datsun 710	22.8	4	108.0	93	3.85
Hornet 4 Drive	21.4	6	258.0	110	3.08
Hornet Sportabout	18.7	8	360.0	175	3.15
Valiant	18.1	6	225.0	105	2.76
Duster 360	14.3	8	360.0	245	3.21

表 6.1: mtcars 数据的前 5 行.

	mpg	cyl	disp	hp	drat
Merc 240D	24.4	4	146.7	62	3.69
Merc 230	22.8	4	140.8	95	3.92
Merc 280	19.2	6	167.6	123	3.92

> **注释**
>
> 表格的标签由 R 代码块中的 `label: tbl-xxx` 给出, 其中 `tbl` 是关键字.

6.2 表格示例 2: 由数据框构造表格

代码

```
#| label: tbl-tab3-2
#| tbl-cap: '遗传连接模型例子中观测到的频数 $y_i$ 和频率 $p(y_i|\pi)$, $i=1, \dots ,4$,
#| echo: FALSE
#| results: "asis"
tab.pis <- data.frame(
  C = c("Frequency $y_i$", "Probability $p(y_i\\mid\\pi)$"),
  V1 = c("125", "$\\frac{1}{2}+\\frac{\\pi}{4}$"),
  V2 = c("18","$\\frac{1}{4}(1-\\pi)$"),
  V3 = c("20","$\\frac{1}{4}(1-\\pi)$"),
  V4 = c("34","$\\frac{1}{4}$")
)
names(tab.pis) <- c("Category", "1", "2", "3", "4")

knitr::kable(tab.pis, escape = FALSE)
```

输出为

表 6.2: 遗传连接模型例子中观测到的频数 y_i 和频率 $p(y_i|\pi)$，$i = 1, \ldots, 4$，197 个动物.

Category	1	2	3	4
Frequency y_i	125	18	20	34
Probability $p(y_i \mid \pi)$	$\frac{1}{2} + \frac{\pi}{4}$	$\frac{1}{4}(1-\pi)$	$\frac{1}{4}(1-\pi)$	$\frac{1}{4}$

注释

表格中的题图通过 `tbl-cap:` 由文本引用生成.

6.3 表格示例 3: 由 markdwon 表格直接生成

INLA 可处理的分布/似然有 60 种, 6.3 仅列出了其中的一部分.

```
| **Value**  | **Likelihood** |
|------------|----------------|
| `poisson`  | Poisson        |
| `binomial` | Binomial       |
| `t`        | Student's t    |
| `gamma`    | Gamma          |

: `INLA`提供的一些似然. {#tbl-liks}
```

输出为

表 6.3: INLA 提供的一些似然.

Value	**Likelihood**
`poisson`	Poisson
`binomial`	Binomial
`t`	Student's t
`gamma`	Gamma

注释

1. 表格中% 用\\%
2. 表格中 latex 命令用\\代替\
3. 前二种方法中表格的标签由 R 代码块中的'label:' 给出.

6.4 markdown 子表的并置

```
::: {#tbl-panel layout-ncol="2"}
| Col1 | Col2 | Col3 |
|------|------|------|
| A    | B    | C    |
| E    | F    | G    |
| A    | G    | G    |

: 第一个表格 {#tbl-first}

| Col1 | Col2 | Col3 |
|------|------|------|
| A    | B    | C    |
| E    | F    | G    |
| A    | G    | G    |

: 第二个表格 {#tbl-second}

二个子表的并置
:::
```

输出为

表 6.4: 二个子表的并置

(a) 第一个表格

Col1	Col2	Col3
A	B	C
E	F	G
A	G	G

(b) 第二个表格

Col1	Col2	Col3
A	B	C
E	F	G
A	G	G

6.5 数据子表的并置与堆叠

在 R 的代码块选项中设置 tbl-subcap: true 或 tbl-subcap 加上子表的图例，并添加 layout-ncol: 2 就可获得二个子表的并置.

```
#| label: tbl-tables
#| tbl-cap: " 两个表格并置示例 2"
#| tbl-subcap:
#|   - "Cars 数据"
#|   - "Pressure 数据"
#| layout-ncol: 2

library(knitr)
kable(head(cars))
kable(head(pressure))
```

输出为

```
library(knitr)
kable(head(cars))
kable(head(pressure))
```

类似地，通过选项 layout-nrow: 2 就可将二个子图堆叠起来.

```
#| label: tbl-tables-by-row
#| tbl-cap: " 两个表格堆叠示例"
#| tbl-subcap:
#|   - "mtcars 数据"
#|   - "iris 数据"
```

表 6.5: 两个表格并置示例 2

(a) Cars 数据

speed	dist
4	2
4	10
7	4
7	22
8	16
9	10

(b) Pressure 数据

temperature	pressure
0	0.0002
20	0.0012
40	0.0060
60	0.0300
80	0.0900
100	0.2700

```
#| layout-nrow: 2


library(knitr)
kable(head(mtcars))
kable(head(iris))
```

输出为

```
library(knitr)
kable(head(mtcars))
kable(head(iris))
```

6.6 表格引用

表格引用由代码块的标签 (设为 `label`) 引用实现, 并带前缀 `tbl-`, 由 `@tbl-label` 实现. 例如

```
本章共出现二个表格并置的示例, 即 表 @tbl-panel, 表 @tbl-tables 和 @tbl-second.
```

输出为:

本章共出现二个表格并置的示例, 即表 6.4, 表 6.5 和 6.4b.

表 6.6: 两个表格堆叠示例

(a) mtcars 数据

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

(b) iris 数据

Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
5.1	3.5	1.4	0.2	setosa
4.9	3.0	1.4	0.2	setosa
4.7	3.2	1.3	0.2	setosa
4.6	3.1	1.5	0.2	setosa
5.0	3.6	1.4	0.2	setosa
5.4	3.9	1.7	0.4	setosa

第七章　Quarto 书稿中的文献排版

这是第 七 章的内容，讲解文献库的建立、文献格式与排版方法.

7.1　文献库建立样例

根据文献的类型，文献库的格式有 14 种类型，它们由 title,author, journal,address 等域构成. 最为常见的文献类型是论文 (article) 和图书 (book), 它们的格式如下

```
@article{CitekeyArticle,
  author    = "P. J. Cohen",
  title     = "The independence of the continuum hypothesis",
  journal   = "Proceedings of the National Academy of Sciences",
  year      = 1963,
  volume    = "50",
  number    = "6",
  pages     = "1143--1148",
}
```

```
@book{CitekeyBook,
  author    = "Leonard Susskind and George Hrabovsky",
  title     = "Classical mechanics: the theoretical minimum",
  publisher = "Penguin Random House",
  address   = "New York, NY",
```

```
  year      = 2014
}
```

其余类型参见网页:The 14 BibTeX entry types. 文献库通过 BiBTeX 或 BiBer 的运行生成文献目录 (bibliography).

7.2 文献风格

不同的期刊、书集对文献目录中参考文献的呈现方式有不同的要求，BiBTeX 是通过风格 (style) 文件来控制 (配合宏包 natbib 使用), BiBer 是通过选项来控制 (配合 biblatex 使用).

文献呈现的风格分为作者-日期格式 (author-date style) 和数字格式 (numeric). 作者-日期格式共有 141 个式样，其中最为常用的式样有 2 个，即 alpha 和 apalike. 数据格式共 88 个式样，其中有 8 类是标准式样，也是最常用的式样，即 abbrv, acm,ieeetr,plain,siam, 和 unsrt. 这些式样的具体形式与介绍见

中文的文献排版根据国标 GB/T77114-2015 的规范，对文献的类型要求提供文献的标识代码，共有 18 个，例如图书用 M 标识，期刊论文用 J 标识.

7.3 中文文献风格的设置

基于 BiBTeX, 中文文献通过宏包 gbt7714 实现[1]，兼容宏包 natbib，在 LaTeX 中使用方法如下：

1. 在导言区调用宏包 gbt7714；
2. 在正文中 \cite{} 等引用文献；
3. 使用 \bibliographystyle{} 选择参考文献表的样式；
4. 使用 \bibliography{} 命令生成参考文献表;
5. 编译生成带文献的 pdf 文件，基于 xelatex 引擎的编译方式如下 (foo.tex 为文件名)

[1]详见 https://github.com/87ouo/gbt7714-bibtex-style 说明

```
xelatex foo.tex
bibtex foo
xelatex foo.tex
xelatex foo.tex
```

基于 BiBer, 中文文献通过宏包 biblatex-gb7714-2015 实现[2]，兼容宏包 biblatex. 本质上 biblatex-gb7714-2015 是个式样宏包，依附于宏包 biblatex, 通过式样选项使用. 在 LaTeX 中使用方法如下：

1. 在导言区调用宏包 biblatex, 并设定文献样本和，例如；
   - 使用顺序编码制：

```
\usepackage[backend=biber,style=gb7714-2015]{biblatex}
```

   - 使用著者-出版年制[3]：

```
\usepackage[backend=biber,style=gb7714-2015ay]{biblatex}
```

2. 在\begin{document} 加载参考文献库，命令为\addbibresource{}
3. 在正文中 \cite{} 等引用文献；
4. 在需要出现文献目录的地方使用 \printbibliography， 可通过 title, heading, segment 选项对输出进行控制;
5. 编译生成带文献的 pdf 文件， 基于 xelatex 引擎的编译方式如下 (foo.tex 为文件名)

```
xelatex foo.tex
biber foo
xelatex foo.tex
xelatex foo.tex
```

7.4 文献库的建立工具

- Zotero
- JabRef

[2]详见 https://github.com/hushidong/biblatex-gb7714-2015
[3]尽管后端 (backend) 可以使用 bibtex, 但不建议使用.

有关文献库扩展包 biblatex 的在 TeX 和 bookdown 中的使用请参考附录 B.

附录 A　Mathjax 的离线安装与使用

附录 A 介绍了网页显示数学公式的插件 mathjax，本地化安装和使用方法等. (R Core Team, 2021; Xie, 2015)

A.1　mathjax 简介

- MathJax是一款相当强悍的在网页显示数学公式的插件.
- 基于 Mathjax, 就可通过 LaTeX 的命令输出精美的数学公式. 加载 Mathjax 后[1]，就可通过一对美元符号 $(或左\(右\))输入行内公式,通过一对双美元符号 $$(或左\[右\])输入行间公式，例如

```
$$
J\alpha(x) = \sum_{m=0}^\infty
\frac{(-1)^m}{m! \Gamma (m + \alpha + 1)}
{\left({ \frac{x}{2} }\right)}^{2m + \alpha}
$$
```

显示出下面的数学公式

$$J\alpha(x) = \sum_{m=0}^{\infty} \frac{(-1)^m}{m!\Gamma(m + \alpha + 1)} \left(\frac{x}{2}\right)^{2m+\alpha}.$$

- 可以使用 LaTeX 自带的复杂的数学环境和数学字体命令，如排版多行公式的 align 和 split 环境，字母加黑命令\mathbb{}

```
\begin{align}
3x-1 &= \mathbb{A} \\
```

[1]需要远程或本地支持

```
  3x &= \mathbf{B} \\
   x &= \mathscr{C}
\end{align}
```

输出为

$$3x - 1 = \mathbb{A} \tag{A.1}$$

$$3x = \mathbf{B} \tag{A.2}$$

$$x = \mathscr{C} \tag{A.3}$$

A.2 调用远程服务器上的 mathjax

一般情况下，只需要使用远程加载 Mathjax 的 js 库就行了，例如在需要渲染数学公式的网页上增加 html 命令

```
</script>
<script type="text/javascript" async
  src="https://cdn.mathjax.org/mathjax/latest/
  MathJax.js?config=TeX-MML-AM_CHTML">
</script>
```

A.3 mathjax 本地服务器的安装与使用

我们以 Macbook 的 Apache 服务器为例说明步骤[2]

1. 服务器的启动

   在终端 (terminal) 下输入命令

```
sudo apachectl start
```

2. 检查服务是否启动成功

   在浏览器中输入网址

[2]Windows 10 下的本地服务器的启动可参考 (https://www.jianshu.com/p/d86c77942181)

```
http://127.0.0.1/
```

如果显示 It Works! 就表示服务器已经成功启动. 请记住: 服务器上文件在本地的位置为

```
/Library/WebServer/Documents
```

3. 关闭服务器 (不用时)

在终端 (terminal) 下输入命令

```
sudo apachectl stop
```

4. 将 Mathjax2.6 或 Mathjax2.7 下载并解压到/Library/WebServer/Documents 下[3], 目录名为 Mathjax

5. 启动本地 Mathjax

在运行 Bookdown(或其他 R Mmarkdown, Quarto Markdown 文件) 时, 须加载下面的由 html 命令组成的文件 mathjax_27.html

```
</script>
<script type="text/javascript"
   src="http://127.0.0.1/MathJax/MathJax.js">
</script>
```

并在 Bookdown 或 Quarto Markdown 中由 _output.yml 的命令

```
include-in-header: mathjax_27.html
```

加载进来.

[3]暂且不要用最新的 3.2 版本

附录 B　基于 biblatex 生成参考文献

附录 B 介绍了文献库扩展包 biblatex 的使用，及其与 natbib 的比较.

B.1　概述

- bibtex 与 biber 是处理参考文献信息的二个外部程序 (backend), 它们起到将 LaTeX 文档与 bib 文档交互作用的目的；
- natbib 与 biblatex 是二个处理参考文献 (bibliography) 展示和引用 (citation) 的 TeX 宏包; natbib 仅通过 bibtex 起作用，而 biblatex 可通过 biber 起作用；

B.1.1　natbib 的优缺点

- natbib 的优点：
  - 有大量与期刊/出版商对应的风格文件 (.sty);
- natbib 的缺点:
  - 自定义的局限: 修改风格文件较为困难；
  - 设计的局限性：主要为自然科学类期刊的 Author-Year 及数字引用方式设计.

B.1.2　biblatex 的优缺点

- biblatex 被认为是 LaTeX 中处理参考文献 (bibliography) 很有前途的宏包，功能强大，提供了很多可定制的选项.
- biblatex 的优点
  - 提供 natbib 的 Author-year 和数字引用方式，因此可视为 natbib 的扩展；
  - 可通过 LaTeX 宏来控制文献的风格，方便修改;

- 提供许多人性化的引用风格 (例如 author-title);
- 提供人性化的文献数据库域名 (字段);
- 直接支持多文献库与文献分类;
- 提供大量标准与拓展的 biblatex 风格 (见使用手册);
- 文献可按主题分割为部分.

- biblatex 的优点
  - 学习曲线陡: 选项多, 不易掌握;
  - 编译时可能会依赖于最新 LaTeX 版本, 从而导致版本的更新, 由此带来不必要的麻烦.

B.2 biblatex 的在 TeX 中的使用样例

1. 使用示例

```
\usepackage[style=authoryear,backend=biber]{biblatex}
```

   代替\usepackage[authoryear]{natbib};

2. 加载一个或多个 bib 文献库文件

```
\addbibresource{file1.bib}
\addbibresource{file2.bib}
```

3. 文献目录输入: 出现文献的地方输入

```
\printbibliography
```

4. 引用:

   使用\textcite 代替\citet; 使用\parencite 代替\citep

5. 编译方式, 以 XeTeX 引擎为例

```
XeLaTeX
biber (代替 bibtex)
XeLaTeX
XeLaTeX
```

B.3 biblatex 在 Quarto 书稿中的使用

1. 在 index.Rmd 文件的 yml 部分增加选项

```
biblatexoptions: [refsegment=chapter]
biblio-style: gb7714-2015ay
```

注意`[]`内可增加其他`biblatex`选项（参考`biblatex`使用手册）

2. 在需要出现文献的地方 (如每一章后) 加

```
\printbibliography[segment=\therefsegment, heading=subbibliography, title={参考文献}
```

3. 在 _output.yml 文件的 bookdown::pdf_book: 增加选项 (前面空二格)

```
citation_package: biblatex
```

> **注释**
>
> 若要 bibtex 驱动文献的排版，只需要在这一步改为 citation_package: natbib

4. 如上所述，LaTeX 要生成最终的 PDF 文档. 如果论文中含有交叉引用 (图形、表格、公式、章节、文献等)、bibtex/biber 命令和术语表等，通常需要让 TeX 编译多次. 而使用 Latexmk, 只需要让 TeX 编译一次, 它会自动帮你做好其它所有事情. 由于系统在你已经安装的 LaTeX 发行版本时已经包含了 latexmk，但我们需要人为地运行 Latexmk 命令 (编译 TeX 文件)，使用 XeLaTeX 的编译格式为

```
latexmk -pvc -xelatex file.tex
```

其中选项 pvc 表示检查输入文件的更改并预览结果. 在 Rstudio 中你只需要添加下面的代码块.

```
Sys.setenv(RSTUDIO_PDFLATEX = "latexmk")
```

附录 C　tinytex 安装

我们在第附录 C 章讲述 tinytex 的安装与使用.

参考文献

R Core Team, 2021. R: A Language and Environment for Statistical Computing[M/OL]. Vienna, Austria: R Foundation for Statistical Computing. https://www.R-project.org/.

Xie Y, 2015. Dynamic Documents with R and knitr[M/OL]. 2nd 版本. Boca Raton, Florida: Chapman; Hall/CRC. http://yihui.name/knitr/.