简单粗暴 LaTeX

K.L Wu

本手册是wklchris-GitHub的 LaTeX-cn 项目

当前版本号：v1.6.2
最后更新于：2018 年 3 月 22 日

目录 | 0

序 1

第一稿序

其实在之前我是有一稿手册的，开始撰写的日期大概在 2015 年 4 月。但是自己觉得写得太烂，因此索性推倒重写了这一版。这一版的主要特征是：

1. 我希望能够吸引初学者快速上手，解决手头的问题。因此去掉了枯燥的讲解和无穷无尽的宏包用法介绍，直接使用实例；
2. 力求突出实用性。当然，也会提点一些可以深入学习的内容，读者可以自行查阅，或者阅读本手册中的扩展阅读章节（即带星号 * 的章节）。
3. 本手册使用的编辑器为 TeXStudio，而非之前的商业软件 WinEdt. 这使得学习 LaTeX 的门槛更低。当然了，你有权使用任何编辑器。

手册主体分为六大部分[1–5]：

写给读者 * 介绍 LaTeX 背景、优缺点、适用情形。

基础 包括标点、缩进、距离、章节、字体、颜色、注释、引用、封面、目录、列表、图表、页面等详细内容。

数学排版 包括数学符号、公式、编号等内容。

进阶 主要是自定义命令，帮助你更高效、更简洁地书写你的文档。

Tikz 绘图 * 附加章节，需要读者取消注释后重编译。

附录 帮助你快速查找一些你想要的东西。

由于工作全部由我一人完成，限于视野，难免存在错漏之处。恳请读者指正。如遇到的手册中无法解决的问题，欢迎向我提出。推荐书目可参考本手册附录。

最后，还要感谢在 LaTeX 学习中为我解答疑惑的同学，特别是来自 LaTeX 度吧的吧友；本手册中许多的解决方案都是由他们提供的。我谨在此记录。

Mail: wklchris@hotmail.com

Chris Wu
September 17, 2016 于 Davis, CA

更新日志：

版本号以 $x.y.z$ 的形式公示。当 $z = 0$ 时，为决定正式 release 的版本。
v1.6.0 更新 — 2017 年 6 月 15 日：
v1.6.1 更新 — 2017 年 8 月 9 日；
v1.6.2 更新 — 2017 年 10 月 5-15 日。
v1.6.3 更新 — 2018 年 3 月 22 日。

以下内容中，未以括号注明的项是在 v1.6.0 中更新的。

- 重要更新：
  - ~~**TikZ 相关的内容停止更新。原因是现有的其他软件绘图功能强大，导出为 pdf 格式的矢量图后也易于调用；TikZ 相比之下学习成本过高。笔者又决定重启 TikZ 章节**~~。（v1.6.3）
  - **重新添加了 TikZ 章节**——不过仍是之前弃笔时的版本，预计将于 v1.7 更新。
  - **增加了加快 XeLaTeX 调用字体速度的方式**。参考“中文支持与 CJK 字体”一节 `fc-cache` 相关内容。（v1.6.1）
- 字体更换：思源宋体。
- 编辑了 Head.tex 文件，使之更易阅读。

添加 宏包`animate`：在 PDF 中展示动态图。（v1.6.1）

添加 宏包`tocbibind`：
- 将目录本身编入目录项。
- 将参考文献章节编号、编入目录项。
- 将索引章节编号、编入目录项。

更新 宏包`dcolumn`，更详细说明了如何在表格中使用小数点对齐。（v1.6.2）

更新 宏包`fancyvbr`：更详细说明了抄录环境`BVerbatim`，如何提供居中支持。(v1.6.2)

更新 宏包`xeCJK`：
- 参数 CJKspace。该功能在新版宏包中已修复。
- 命令 `\setCJKmainfont`，可指定字体文件名。

- 数学内容：

添加 宏包`extarrows`：长等号命令。

添加 命令`\numberthis`，用于给多行公式中的某行编号。(v1.6.2)

添加 命令`\allowdisplaybreaks`，用于支持多行公式环境换页。

- 拆分了文档，并修改了“文档拆分”一节的内容。（v1.6.3）
- 修复了一些错别字与无效的文档内跳转链接。

更早版本的更新细节，请到Project Release Webpage浏览。

写给读者* 2

我见过许多朋友初试 LaTeX，他们都感到非常不能理解。主要有以下几个疑问：

1. “我平常使用 MS Word，似乎也能完成科技排版工作，为什么我还需要 LaTeX?” ——见“为什么需要 *LaTeX*?”一节。
2. “LaTeX 看上去不像是排版工具，更像是编程语言。我讨厌用写代码一样的方式来写文章。” ——文本文件使你更专注于内容而不是排版细节。
3. “LaTeX 能生成 doc 文件吗？我平时上交作业/提交汇报时难道使用不便修改 pdf 文件么?” ——见“*LaTeX* 生成的文件格式?”一节。

本章希望能解决读者的这些疑问，让读者对于 LaTeX 有基础的了解，再决定是否需要学习。当然，如果你是被迫进入了 LaTeX 这个坑，你也可以阅读本章，或许本章能让你喜欢上 LaTeX 呢。

2.1 什么是 LaTeX?

先讲 TeX（读音类似于“泰赫”）。

TeX 是 Knuth[1]研发的免费、开源的排版系统，其初衷是为了“改变排版界糟糕的排版技术”，并用于排版他的著作《计算机程序设计艺术》。

TeX 对于读者来说应该是底层的内容，如果你有兴趣，可以阅读 Knuth 著的 *The TeXbook*，这本书是学习 TeX 最权威的材料，没有之一。在本手册的参考文献中也给出了其他 TeX 学习的资料。

再讲 LaTeX（读音类似于“拉泰赫”）。

LaTeX 是基于 TeX 的宏[2]集，其作者是 Dr. Lamport[3]；其姓氏开头两个字母

[1] Donald Ervin Knuth(高德纳，1938–)，现代计算机科学的先驱者，斯坦福大学计算机系的终身荣誉教授，图灵奖和冯诺依曼奖得主，TeX 和 METAFONT 的发明人。同时也是业内经久不衰的著作 *The Art of Computer Programming* 的作者。

[2] “宏 (Macro)”是一个计算机概念，指用单个命令或操作完成一系列底层命令或操作的组合。

[3] Leslie Lamport(1941–)，美国计算机科学家，图灵奖和冯诺依曼奖得主。

La 与底层排版系统 TEX 相结合，就组成了名称 LATEX。LATEX 在 TEX 基础上定义了众多的宏命令，使得用户可以更方便地进行排版。本手册的参考文献中就有他的作品。

LATEX 现在的版本是LATEX 2ε，意思是 2.x 版而没到 3.0 版。错位排版的字母 E 和字母 A 暗示了它是排版系统。在无法这样输出的场合，请写作 LaTeX 和 LaTeX 2e。

2.2 TEX 与 LATEX 的优缺点

TEX 的优点：**稳定、精确、美观**。底层的 TEX 系统已经很多年没有进行大的变动了，因为它注重稳定；TEX 系统可以让你把排版内容通过数字参数的方式写到任意的位置，量化的参数意味着精确；TEX 底层的空距调整机制，以及对于数学公式近乎完美的支持，则确保了排版效果的美观。

LATEX 是基于 TEX 的，自然不会抛弃上述 TEX 的优点。具体包括：

- 排版出来就是印刷品。专业而美观。
- 易用、全面的数学排版支持，无出其右。
- 撰写文档时不会被文档排版细节干扰精力。你可以使用之前自定义的模板，或者方便地在文字组织完毕后调整你的模板，以轻松达到满意的效果。
- 复杂的排版功能支持，比如图表目录、索引、参考文献管理、高度自定义的目录样式、双栏甚至多栏排版。
- 丰富的功能以及易寻的帮助文档。众多的 LATEX 宏包赋予了 LATEX 强大的扩展功能，它们都自带文档供你学习。
- 源文件是文本文件[4]。你可以在任何设备、任何文本编辑器中书写文档内容，无须担心复制时格式的变化；最后粘贴到同一个 tex 文件中编译即可。
- 跨平台，免费，开源。

那缺点呢？我认为主要有：

- 入门门槛高。想要熟练地使用 LATEX 并轻松地编写有自己的风格的文档，不是一两天就能够达到的。
- 并非“所见即所得”，需要编译才能看到效果。编译查错有时令人恼火。
- 完善一个自己的模板可能需要很长的时间。尽管 LATEX 原生定义的模板能够满足绝大多数场合的需要。
- 排版长表格有些复杂。但作为补充，在表格内插入数学公式是非常简单的。

2.3 为什么需要 LATEX？

你可能基于以下原因学习 LATEX：

1. 你的投稿对象要求你使用 LATEX 排版，而不是 MS Word——这种情况对没听说过 LATEX 的你来说，真是糟糕透了。
2. 你需要在多个设备上撰写同一份文档。但你发现把内容在多个文档间复制粘贴时，格式总是会出现问题。
3. 你受够了 MS Word 自带的公式编辑器，或者你觉得购买的插件 MathType 的效果也不尽如人意。但你经常需要排版公式。

[4]文本文件的另一个优点是易于进行版本控制，比如利用 git. 你可以方便地比较你相比上次修改了什么内容，也可以方便地恢复到之前某个时刻的版本。

4. 你想参加某个科学竞赛,比如 MCM,然后你发现你的朋友用的一个叫 LaTeX 的东西似乎还不错。
5. 你想出版一本书，或者投稿你的作品——结果他们告知你如果你使用 LaTeX 而不是 MS Word 攥写原稿件，他们会更快地把作品印刷出来。
6. 呃……也许，你只是喜欢学习新事物?

对于科研工作者或者在校研究生，我认为 LaTeX 是非常优秀的工具。如果你是本科生，或者更年轻的群体，你也可以先学习 LaTeX，因为到了研究生和工作中，学习的时间可能就非常有限了。

2.4 MS Word 难道不优秀吗?

我想说的是，**MS Word 当然是优秀的软件**。但是它与 LaTeX 的定位不同，所以它们分别适用于不同场合。前者注重简单组织内容，后者注重排版效果。

在排版书籍、科学文档方面，LaTeX 非常专业、美观，公式支持性极佳，几乎所有参数你都可以量化调整。如果你想高度自定义一份文件，比如拥有特殊几何、颜色元素，且易于更改模板的简历，LaTeX 可以完全独当一面。在这些这方面，MS Word 是无法匹敌 LaTeX 的。

但是如果你只是为了生成非正式的文档，比如 1–2 页的作业稿；或者只是一份易于别人修改的非科学稿件，比如一份需要同事修改的演讲稿……那你无须使用 LaTeX。这些方面，LaTeX 无疑是比不上 MS Word 的。

2.5 LaTeX 生成的文件格式?

一般广为使用的是 pdf，以及 dvi 的格式。LaTeX 无法生成 doc 或者 docx 扩展名文件，因为那是属于 MS 的商用格式，两者的工作机理也完全不同。

所以很遗憾,如果你身处一个要求你“必须提交 docx”的环境中,那么 LaTeX 对你并不是一个好选择。但我想指出的是，这是稳定、优秀的 pdf 格式没有得到你身处环境认可的遗憾——pdf 也可以方便地添加批注，并在不同设备的显示上有更好的稳定性。

3 LaTeX 基础

3.1 第一份文稿

编辑器的配置大概是需要讲解一下的，毕竟对于初学者来说是很头疼的事情。本手册就以 TeXStudio 为例进行配置。首先你应该安装一个 TeX Live，它是完全免费的，网址：http://tug.org/texlive/.

虽然它体积较大，但是却是最一劳永逸、最不需要花时间取配置的方法，同时它大概也是功能支持最强的 LaTeX 发行版。

打开 TeXStudio 后，选择选项 → 设置 TeXStudio → 构建 → 默认编译器，选择 XeLaTeX. 这主要是基于中文文档编译的考虑，同时 XeLaTeX 也能很好地编译英文文档。我建议始终使用它作为默认编译器。

之后你可以在窗口输入一篇小文档，并保存为 tex 扩展名的文件进行测试：

```
\documentclass{ctexart}
\begin{document}
    Hello, world!
    你好，世界！
\end{document}
```

点击编译按钮生成，F7 查看。生成的 pdf 在你的 tex 文件保存目录中。具体各行的含义我们会在后文介绍。

3.2 认识 LaTeX

3.2.1 命令与环境

LaTeX 中的**命令**通常是由一个反斜杠加上命令名称，再加上花括号内的参数构成的（有的命令不带参数，例如：`\TeX`）。

```
\documentclass{ctexart}
```

如果有一些选项是备选的，那么通常会在花括号前用方括号标出。比如：

```
\documentclass[a4paper]{ctexart}
```

还有一种重要指令叫做**环境**。它被定义于控制命令`\begin{environment}`和`\end{environment}`间的内容。比如：

```
\begin{document}
...内容...
\end{document}
```

环境如果有备选参数，只需要写在`\begin[...]{name}`这里就行。

注意：不带花括号的命令后面如果想打印空格，请加上一对内部为空的花括号再键入空格。否则空格会被忽略。例如：`\LaTeX{} Studio`.

3.2.2 保留字符

LaTeX 中有许多字符有着特殊的含义，在你生成文档时不会直接打印。例如每个命令的第一个字符：反斜杠。单独输入一个反斜杠在你的行文中不会有任何帮助，甚至可能产生错误。LaTeX 中的保留字符有：

```
# $ % & _ { } \
```

它们的作用分别是：

`#`：自定义命令时，用于标明参数序号。
`$`：数学环境命令符。
`%`：注释符。在其后的该行命令都会视为注释。如果在回车前输入这个命令，可以防止行末 LaTeX 插入一些奇怪的空白符。
`^`：数学环境中的上标命令符。
`&`：表格环境中的跳列符。
`_`：数学环境中的下标命令符。
`{`与`}`：花括号用于标记命令的必选参数，或者标记某一部分命令成为一个整体。
`\`：反斜杠用于开始各种 LaTeX 命令。

以上除了反斜杠外，均能用前加反斜杠的形式输出。即你只需要键入：

```
\# \$ \% \^ \& \_ \{ \}
```

唯独反斜杠的输出比较头痛，你可以尝试：

```
1 $\backslash$ \textbackslash
2 \texttt{\char92}
```

\ \\

其中命令`\char[num]`是一个特殊的命令，使用环境需要是 tt 字体环境，用于输出 USCII 码对应的字符；92 对应的即反斜杠。你也可以用`\char`后加字符的方式输出你想输出的命令，但需要包裹在`\texttt`或者`\ttfamily`内。如果想输出的字符是保留字符，需要再加一个反斜杠。

```
\texttt{\char`~} % 输出一个波浪线
\texttt{\char`\\} % 输出保留字反斜杠
\texttt{\char`@} % 实际上可直接输入@
```

另外需要说明的是，上例提及的波浪线~ 用来输出一个禁止在该处断行的空格，也不能够直接输出。尝试：

```
1 a $\sim$ b
2 a\~ b
3 a\~{} b
4 a\textasciitilde b
```

a ∼ b ab̆ a˜ b a~b

3.2.3 导言区

任何一份 LaTeX 文档都应当包含以下结构：

```
\documentclass[options]{doc-class}
\begin{document}
    ...
\end{document}
```

其中，在语句`\begin{document}`之前的内容称为**导言区**。导言区可以留空，以可以进行一些文档的准备操作。你可以粗浅地理解为：导言区即模板定义。

文档类的参数 doc-class 和可选选项 *options* 有表3.1取值：

表 3.1: 文档类和选项

doc-class 文档类[1]	
article	- 科学期刊，演示文稿，短报告，邀请函。
proc	- 基于 article 的会议论文集。
report	- 多章节的长报告、博士论文、短篇书。
book	- 书籍。
slides	- 幻灯片，使用了大号 Scans Serif 字体。
options	
字体	- 默认 10pt，可选 11pt 和 12pt.
页面方向	- 默认竖向 portrait，可选横向 landscape。
纸张尺寸	- 默认 letterpaper，可选用 a4paper, b5paper 等。
分栏	- 默认 onecolumn，还有 twocolumn。
双面打印	- 有 oneside/twoside 两个选项，用于排版奇偶页。article/report 默认单面。
章节分页	- 有 openright/openany 两个选项，决定是在奇数页开启新页或是任意页开启新页。注意 article 是没有 chapter（“章”）命令的，默认任意页。
公式对齐	- 默认居中，可改为左对齐 fleqn；默认编号居右，可改为左对齐 leqno。
草稿选项	- 默认 final，可改为 draft，使行溢出的部分显示为黑块。

在本文中，多数的文档类提及的均为 report/book 类。如果有 article 类将会特别指明。其余的文档类不予说明。本手册排版即使用了 report 类。

在导言区最常见的是**宏包**的加载工作，命令形如：`\usepackage{package}`。通俗地讲，宏包是指一系列已经制作好的功能“模块”，在你需要使用一些原生 LaTeX 不带有的功能时，只需要调用这些宏包就可以了。比如本文的代码就是利用`listings`宏包实现的。

宏包的具体使用将参在各部分内容说明中进行讲解。如果你想学习一个宏包的使用，按 Win+R 组合键呼出运行对话框，输入 texdoc 加上宏包名称即可打

开宏包帮助 pdf 文档。例如：`texdoc xeCJK`。

3.2.4 错误的排查

在编辑器界面上，下方的日志是显示编译过程的地方。在你编译通过后，会出现这样的字样：

- **Errors 错误**：严重的错误。一般地，编译若通过了，该项是零。
- **Warnings 警告**：一些不影响生成文档的瑕疵。
- **Bad Boxes 坏箱**[2]：指排版中出现的长度问题，比如长度超出（Overfull）等。后面的 Badness 表示错误的严重程度，程度越高数值越大。这类问题需要检查，排除 Badness 高的选项。

你可以向上翻阅日志记录（即.log 文件），来找到 Warning 开头的记录，或者 Overfull/ Underfull 开头的记录。这些记录会指出你的问题出在哪一行 (比如 line 1-2) 或者在 pdf 的哪一页（比如 active [12]。注意，这个 12 表示计数器计数页码，而不是文件打印出来的真实页数)。此外你还需要了解：

- 值得指出的是，由于 LaTeX 的编译原理（第一次生成 aux 文件，第二次再引用它），目录想要合理显示**需要连续编译两次**。在连续编译两次后，你会发现一些 Warnings 会在第二次编译后消失。在 TeXStudio 中，你可以只单击一次“构建并查看”，它会检测到文章的变化并自动决定是否需要编译两次。
- 对于大型文档，寻找行号十分痛苦。你需要学会合理地拆分 tex 文件，参阅第3.13节的内容。

这里也推荐宏包`syntonly`，在导言区加入它支持的`\syntaxonly`命令，会只排查语法错误而不生成任何文档，这可以使你更快地编译。不过它似乎不太稳定，例如本文档可以正常编译，但是使用该命令时则会出错。

3.2.5 文件输出

LaTeX 的输出一般推荐 pdf 格式，由 LaTeX 直接生成 dvi 的方法并不推荐。你在 tex 文档的文件夹下可能看到的其他文件类型：

.sty	宏包文件
.cls	文档类文件。
.aux	用于储存交叉引用信息的文件。因此，在更新交叉引用（公式编号、大纲级别）后，需要编译两次才能正常显示。
.log	日志。记录上次编译的信息。
.toc	目录文件。
.lof	图形目录。
.lot	表格目录。
.idx	如果文档中包含索引，该文件用于储存索引信息。
.ind	索引记录文件。
.ilg	索引日志文件。
.bib	BibTeX 参考文献数据文件。
.bbl	BibTeX 生成的参考文献记录。
.bst	BibTeX 模板。
.blg	BibTeX 日志。

[1]此外还有`beamer`宏包定义的 beamer 文档类，常用于创建幻灯片。
[2]Box 是 LaTeX 中的一个特殊概念，具体将在这里进行讲解。

.out hyperref 宏包生成的 pdf 书签记录。

有时 LaTeX 的编译出现异常，你需要删除文件夹下除了 tex 以外的文件再编译。此外，在某些独占程序打开了以上的文件时（比如用 Acrobat 打开了 pdf），编译可能出现错误。请在编译时确保关闭这些独占程序。

3.3 标点与强调

英文符号 | <> + = 一般用于数学环境中，如果在文本中使用，请在它们两侧加上“$”。如果你在 LaTeX 中直接输入大于、小于号而不把它们放在数学环境中，它们并不会被正确地打印。你应该使用`\textgreater`, `\textless`命令。

在科技文章中，中文的句号也一般使用全角圆点“．”[3]而不是平常的“。”，也不是正常的英文句点“.”。这个符号很难正常输入；你可以先输入正常句点，最后再替换。

3.3.1 引号

英文单引号并不使用两个' 符号组合。左单引号是重音符`（键盘上数字 1 左侧），而右单引号是常用的引号符。英文中，左双引号就是连续两个重音符。

英文下的引号嵌套需要借助`\thinspace`命令分隔，比如：

```
``\thinspace`Max' is here.''
```

中文下的单引号和双引号你可以用中文输入法直接输入。

3.3.2 破折、省略号与短横

英文的短横分为三种：

- 连字符：输入一个短横：-，效果如 daughter-in-law
- 数字起止符：输入两个短横：--，效果如：page 1–2
- 破折号：输入三个短横：---，效果如：Listen—I'm serious.

中文的破折号你也许可以直接使用日常的输入方式。中文的省略号同样。但是注意，英文的省略号使用`\ldots`这个命令而不是三个句点。

3.3.3 强调：粗与斜

LaTeX 中专门有个叫做`\emph{text}`的命令，可以强调文本。对于通常的西文文本，上述命令的作用就是斜体。如果你对一段已经这样转换为斜体的文本再使用这个命令，它就会取消斜体，而成为正体。

西文中一般采用上述的斜体强调方式而不是粗体，例如在说明书名的时候可能就会使用以上命令。关于字体的更多内容参考字体这一节。

[3]这个标点是 U+FF0E，称为“FULLWIDTH FULL STOP”。

3.3.4 下划线与删除线

LaTeX 原生提供的\underline命令简直烂的可以，建议你使用ulem宏包下的 uline 命令代替，它还支持换行文本。ulem宏包还提供了一些实用命令：

```
\uline{下划线} \\
\uuline{双下划线} \\
\dashuline{虚下划线} \\
\dotuline{点下划线} \\
\uwave{波浪线} \\
\sout{删除线} \\
\xout{斜删除线}
```

下划线
双下划线
虚下划线
点下划线
波浪线
删除线
斜删除线

需要注意的是，ulem宏包重定义了\emph命令，使得原来的加斜强调变成了下划线、原来的两次强调就取消强调变成了两次强调就双下划线。通过宏包的 normalem 选项可以取消这个更改：\usepackage[normalem]{ulem}。

3.3.5 其他

角度符号或者温度符号需要借助数学模式$...$ 输入：

```
$30\,^{\circ}$三角形 \\
$37\,^{\circ}\mathrm{C}$
```

30° 三角形
37 °C

欧元符可能需要用到textcomp宏包支持的\texteuro命令。

其次是千位分隔位，比如1\,000\,000。如果你不想它在中间断行就在外侧再加上一个\mbox命令：\mbox{1\,000\,000}。

再次是注音符号，比如 ô，也常用于拼音声调，参考注音符号表部分的附录内容。如果你想输入音标，请使用 tipa 宏包[4]，同样参考附录 A。

最后，介绍hologo宏包，它可以输出许多 TeX 家族标志。其实 LaTeX 原生自带了\LaTeX, \TeX等命令。而 hologo 宏包支持的命令有：

```
% 大写H表示符号的首字母也大写
\hologo{XeLaTeX} \Hologo{BibTeX}
```

XeLaTeX BibTeX

3.4 格式控制

首先了解一下 LaTeX 的长度单位：

pt point，磅。

pc pica。1pc=12pt，四号字大小

in inch，英寸。1in=72.27pt

bp bigpoint，大点。1bp=$\frac{1}{72}$in

[4]tipa 会重定义\!命令，因此请使用\negthinspace代替；或在xeCJK与amsmath宏包前加载，并使用 safe 选项。具体可以参见本手册的 Head.tex 文件。

cm centimeter，厘米。1cm=$\frac{1}{2.54}$in
mm millimeter，毫米。1mm=$\frac{1}{10}$cm
sp scaled point。TeX 的基本长度单位，1sp=$\frac{1}{65536}$pt
em 当前字号下，大写字母 M 的宽度。
ex 当前字号下，小写字母 x 的高度。
然后是几个常用的长度宏，更多的长度宏使用会在表格、分栏等章节提到。

```
\textwidth % 页面上文字的总宽度，即页宽减去两侧边距。
\linewidth % 当前行允许的行宽。
```

有时候你可以使用可变长度，比如“`5pt plus 3pt minus 2pt`”，表示一个能收缩到 3pt 也能伸长到 8pt 的长度。直接使用倍数也是允许的，例如：1.5`\parindent`等。

我们通常使用`\hspace{len}`和`\vspace{len}`这两个命令控制特殊的空格，具体的使用方法参考水平和竖直距离这一节。

3.4.1 空格、换行与分段

在 LaTeX 中，多个空格会被视为一个，多个换行也会被视为一个。如果你想要禁止 LaTeX 在某个空格处的换行，将空格用 `~` 命令替代即可，比如“`Fig.~8`”。

通常的换行方法非常简单：LaTeX 会自动转行，然后在每一段的末尾，只需要输入两个回车即可完成分段。如果需要一个空白段落（实质是一个空白行），先输入两个回车，再输入`\mbox{}`，最后再输入两个回车即可。你可以用`\par`来产生一个带缩进的新段。

在下划线一节的例子中已经给出了强制换行的方式，即两个反斜：`\\`. 不过这样做的缺点在于下一行段首缩进会消失，这个命令也的确一般不用于正文换行；**正文中想要换行，请直接使用两个回车**。

段落之间的距离由`\parskip`控制，默认 `0pt plus 1pt`.

```
\setlength{\parskip}{0pt}
```

宏包`lettrine`能够产生首字下沉的效果：

```
\lettrine{T}{his} is an example.
    Hope you like this package,
    and enjoy your \LaTeX\ trip!
```

THIS is an example. Hope you like this package, and enjoy your LaTeX trip!

3.4.2 分页

用`\newpage`命令开始新的一页。
用`\clearpage`命令清空浮动体队列[5]，并开始新的一页。
用`\cleardoublepage`命令清空浮动体队列，并在偶数页上开始新的一页。

[5]参见浮动体这一节的内容。

注意：以上命令都是基于`\vfill`的。如果要连续新开两页，请在中间加上一个空的箱子（`\mbox{}`），如`\newpage\mbox{}\newpage`。

3.4.3 缩进、对齐与行距

英文的段首不需要缩进。但是对中文而言，段首缩进需要借助`indentfirst`宏包来完成。你可能还需要使用`\setlength\parindent{2em}`这样的命令来设置缩进距离。如果在句首强制取消缩进，你可以在段首使用`\noindent`命令。

LaTeX 默认使用两端对齐的排版方式。你也可以使用`flushleft`, `flushright`, `center`这三种环境来构造居左、居中、居右三种效果。特殊的`\centering`命令常常用在环境内部（或者一对花括号内部），以实现居中的效果。但请尽量用`center`环境代替这个老旧的命令。类似的命令还有`\raggedleft`来实现居右，`\raggedright`来实现居左。更多的空格控制请参考这一节。

插入制表位、悬挂缩进、行距等复杂的调整参考这部分的内容。

3.5 字体与颜色

这一节只讨论行文中字体使用。数学环境内字体使用请参考这一节的内容。

3.5.1 字族、字系与字形

字体 (typeface) 的概念非常令人恼火，在电子化时代，基本上也都以字体 (font) 作为替代的称呼。宋体、黑体、楷体，这属于**字族**；对应到西文就是罗马体、等宽体等。加粗、加斜属于**字系和字形**。五号、小四属于**字号**。这三者大概可以并称**字体**[6]。

3.5.2 中西文“斜体”

首先需要明确一点：汉字没有加斜体。平常我们看到的加斜汉字，通常是几何变换得到的结果，非常的粗糙，并不严格满足排版要求；而真正的字形是需要精细的设计的。同时，汉字字体里面也很少有加粗体的设计。

西文一般设有加斜，但是这与“斜体”并不是同一回事。加斜是指某种字族的 Italy 字系；而斜体，是指 Slant 字族。在行文中表强调时使用的是前者；在 Microsoft Word 等软件中看到的倾斜的字母 *I*，也代表前者。

3.5.3 原生字体命令

LaTeX 提供了基本的字体命令，包括表3.2中显示的内容。

字族、字系、字形三种命令是互相独立的，可以任意组合使用。但这种复合字体的效果有时候无法达到（因为没有对应的设计），比如`\scshape`字形和`\bfseries`字系。LaTeX 会针对这种情况给出警告，但仍可以编译，只是效果会不同于预期。

[6] 本文中的字族、字系等称呼难以找到统一标准，可能并不是准确的名称。

表 3.2: LaTeX 字体命令表

字族	-	\rmfamily	-	把字体置为 Roman 罗马字族。
	-	\sffamily	-	把字体置为 Sans Serif 无衬线字族。
	-	\ttfamily	-	把字体置为 Typewriter 等宽字族。
字系	-	\bfseries	-	粗体 **BoldSeries** 字系属性。
	-	\mdseries	-	中粗体 MiddleSeries 字系属性。
字形	-	\upshape	-	竖直 Upright 字形。
	-	\slshape	-	斜体 *Slant* 字形。
	-	\itshape	-	强调体 *Italic* 字形。
	-	\scshape	-	小号大写体 Scap 字形。
如果临时改变字体，使用\textrm，\textbf这类命令。				

如果在文中多次使用某种字体变换，可以将其自定义成一个命令。这时请使用 text 系列的命令而不要使用 family, series 或 shape 系列的命令。否则需要多加一组花括号防止“泄露”。以下二者等价：

```
\newcommand{\concept}[1]{\textbf{#1}}
\newcommand{\concept}[1]{{\bfseries #1}}
```

更多自定义命令的语法请参考这一节。

然后就是字号的命令。行文会有一个默认的“标准”字号，比如你在 documentclass 的选项中设置的 12pt（如果你设置了的话）。LaTeX 给出了一系列“相对字号命令”，列出如表3.3。此外，ctex宏包的\zihao命令，参数 0–8 以及 −0−−8 表示初号到八号、小初到小八[7]。

表 3.3: 相对字号命令表

命令	10pt	11pt	12pt
\tiny	5pt	6py	6pt
\scriptsize	7pt	8pt	8pt
\footnotesize	8pt	9pt	10pt
\small	9pt	10pt	11pt
\normalsize	10pt	11pt	12pt
\large	12pt	12pt	14pt
\Large	14pt	14pt	17pt
\LARGE	17pt	17pt	20pt
\huge	20pt	20pt	25pt
\Huge	25pt	25pt	25pt

[7]日常使用的小四为 12pt，五号为 10.5pt。

如果你想设置特殊的字号，使用：

```
\fontsize{font-size}{line-height}{\selectfont <text>}
```

其中 font-size 填数字，单位 pt；一般而言，line-height 填`\baselineskip`[8]。

默认全文的字体使用`\rmfamily`族的字体。你可以通过重定义的方式更改它，使`\rmfamily`，`\textrm`命令都指向新的字体。甚至把默认字体改为 sf/tt 字族。

```
\renewcommand{\rmdefault}{font-name}
% 默认字体改为sf字族，也可用\ttdefault
\renewcommand{\familydefault}{\sfdefault}
\renewcommand{\sfdefault}{font-name}
% 如果你排版CJK文档，还需要更改CJK的默认字体
\renewcommand{\CJKfamilydefault}{\CJKsfdefault}
```

3.5.4 西文字体

LaTeX 预包含字体如表3.4（参考http://www.tug.dk/FontCatalogue/）：

表 3.4: 部分 LaTeX 西文字体

命令	字体名
cmr	Computer Modern Roman (默认)
lmr	Latin Modern Roman
pbk	Bookman
ppl	Palatino
lmss	Latin Modern Roman Serif
phv	Helvetica
lmtt	Latin Modern

以上可以这样使用：

```
\newcommand{\myfont}[2]{{\fontfamily{#1}\selectfont #2}}
\renewcommand{\rmdefault}{ptm} % 可更改默认字体，同理可改sfdefault等
% 以上在导言区定义。在正文中：
Let's change font to \myfont{ppl}{Palatino}!
```

在 XeLaTeX 编译下，一般使用`fontspec`宏包来选择**本地安装**的字体。注意：该宏包可能会明显增加编译所需的时间。

```
\usepackage{fontspec}
  \newfontfamily{\lucida}{Lucida Calligraphy}
  \lucida{This is Lucida Calligraphy}
```

该宏包的`\setmathrm/sf/tt`与`\setboldmathrm`命令可以支持你更改数学环境中调用的字体。

[8]这个命令的意义是行与行之间的基线间距（即行距），默认是 1.2 倍文字高。

另外，你也可以通过简单地加载txtfont宏包，设置西文字体为 Roman 体，且同时会为你设置好数学字体。其他的简单字体宏包还有cmbright，提供的 CM Bright 与 TeX 默认字体 Computer Modern 协调的不错；以及提供 Palatino 字体的pxfonts。另外的字体宏包在此不再介绍。

3.5.5 中文支持与 CJK 字体

中文方面，ctex宏包直接定义了新的中文文档类 ctexart, ctexrep 与 ctexbook，以及 ctexbeamer 幻灯文档类。例如本手册 Head.tex 中：

```
\documentclass[a4paper, zihao=-4, linespread=1]{ctexrep}
  \renewcommand{\CTEXthechapter}{\thechapter}
```

以上设置字号为小四，行距因子为 1（故行距为 $1 \times 1.2 = 1.2$ 倍，其中 1.2 是 LaTeX 默认的基线间距）。而 a4paper 选项继承与原生文档类 report，可见 ctex 文档类还是很好地保留了原生文档类的特征。值得注意的是，ctex 文档类会用\CTEX 开头的计数器命令代替原有的，除非你使用 scheme=plain 来让 ctex 文档类仅支持中文而不做任何文档细节更改。具体的使用参考 ctex 宏包文档。

ctex宏包支持以下字体命令：

表 3.5: ctex 宏包支持的字体命令

宋体	\songti	黑体	\heiti	仿宋	\fangsong	楷书	\kaishu
雅黑	\yahei	隶书†	\lishu	幼圆†	\youyuan		

† 标注了此符号的字体不受 *ubuntu* 字库支持。

再者参考 XeLaTeX 编译下的xeCJK宏包的使用。在使用 XeLaTeX 时，如果你使用 ctex 文档类，它会在底层调用xeCJK宏包，所以你无须再显式地加载它。当然你也可以使用原生文档类，然后逐一汉化参数内容。

TeX Live 配合 XeLaTeX 时，调用字体非常慢。Windows 下，把 xelatex.exe 与 TeXStudio 设为管理员运行，能大幅缩短编译用时。另外，安装新字体后，管理员命令行 fc-cache 能够刷新字体缓存（很慢），有时也能改善用时[9]。

比如在导言区：

```
\usepackage[slantfont,boldfont]{xeCJK}
  \xeCJKsetup{CJKMath=true}
  \setCJKmainfont[BoldFont=SimHei]{SimSun}
% 这里把SimHei直接写成中文“黑体”也可以
% 也可以直接通过字体文件名调用
% \setCJKmainfont{SourceHanSerifCN-Regular.otf}
```

其中，加载 xeCJK 宏包时使用了 slantfont 和 boldfont 两个选项，表示允许设置中文的斜体和粗体字形。在 setCJKmainfont 命令中，把 SimSun（宋体）设置为了主要字体，SimHei（黑体）设置为主要字体的粗体字形，即 textbf 或者 bfseries 命令的变换结果。你也可以使用 SlantFont 来设置它的斜体字形。

[9]提供这两种方式的网页链接：StackExchange 页面。

除了 setCJKmainfont,还有 setCJKsansfont(对应\textsf),setCJKmonofont(对应\texttt),以及 setCJKmathfont(对应数学环境下的 CJK 字体,但需要载 xeCJKsetup 中设置 CJKMath=true)。

上面提到的 xeCJKsetup 有下列可以定制的参数,下划线为默认值:

- CJKspace=true/false:是否保留行文中 CJK 文字间的空格,默认忽略空格。
- CJKMath=true/false:是否支持数学环境 CJK 字体。如果想在数学环境中直接输入汉字,请开启该选项;否则在数学环境内,需要将汉字写在\textrm、或者amsmath宏包支持的\text命令中。
- CheckSingle=true/false:是否检查 CJK 标点单独占用段落最后一行。此检查在倒数二、三个字符为命令时可能失效。
- LongPunct={——……}:设置 CJK 长标点集,默认的只有中文破折号和中文省略号。长标点不允许在内部产生断行。你也可以用 += 或者-= 号来修改 CJK 长标点集。
- MiddlePunct={—— • }:设置 CJK 居中标点集,默认的只有中文破折号和中文间隔号(中文输入状态下按数字 1 左侧的重音符号键)。居中标点保证标点两端距前字和后字的距离等同,并禁止在其之前断行。你同样可以使用 +=/-= 进行修改。
- AutoFakeBold=true/false:是否启用全局伪粗体。如果启用,在 setCJKmainfont 等命令中,将用 AutoFakeBold=2 参数代替原有的 BoldFont=SimHei 这种参数。其中,数字 2 表示将原字体加粗 2 倍实现伪粗体。
- AutoFakeSlant=true/false:是否启用全局伪斜体。仿上。

如果预定义一种 CJK 字体,可以在导言区使用如下命令。比如这里定义了宋体,后文中直接使用\songti来调用 SimSun 字体:

```
% 参数: [family]\font-switch[features]{font-name}
\newCJKfontfamily[song]\songti{SimSun}
```

如果要临时使用一种 CJK 字体,使用\CJKfontspec命令。其中的 FakeSlant 和 FakeBold 参数根据全局伪字体的启用情况决定;如果未启用则使用 BoldFont、SlantFont 参数指定具体的字体。

```
{\CJKfontspec[FakeSlant=0.2,FakeBold=3]{SimSun} text}
```

对于 Windows 系统,想要获知电脑上安装的中文字体,使用 CMD 命令:

```
fc-list -f "%{family}\n" :lang=zh-cn >d:\list.txt
```

然后到d:\list.txt 文件中查看中文字体列表。

3.5.6 颜色

使用xcolor宏包来方便地调用颜色。比如本文中代码的蓝色:

```
\usepackage{xcolor}
  \definecolor{keywordcolor}{RGB}{34,34,250}
% 指定颜色的text
{\color{color-name}{text}}
```

xcolor宏包预定义的颜色:

表 3.6: xcolor 宏包预定义颜色

black	darkgray	lime	pink	violet	blue	gray
magenta	purple	white	brown	green	olive	red
yellow	cyan	lightgray	orange	teal		

还可以通过“调色”做出新的效果：

```
{\color{red!70} 百分之70红色}\\
{\color{blue!50!black!20!white}
  50蓝20黑30白}\\
{\color{-yellow}黄色的互补色}
```

百分之 70 红色
50 蓝 20 黑 30 白
黄色的互补色

还有一些方便的颜色命令，比如带背景色的箱子，参考第5.2.9节。

3.6 引用与注释

电子文档的最大优越在于能够使用超链接，跳转标签、目录，甚至访问外部网站。这些功能实现都需要“引用”。

3.6.1 标签和引用

使用\label命令插入标签（在 MS Word 中称为“题注”），然后在其他地方用\ref或者\pageref命令进行引用，分别引用标签的序号、标签所在页的页码。

```
\label{section:this}
\ref{section:this}
\pageref{section:this}
```

宏包amsmath提供了\eqref命令，默认效果如 (3.1)，实质上是调用了原生的\ref命令。

但是更常用的是hyperref宏包。由于它经常与其他宏包冲突，一般把它放在导言区的最后。比如本手册：

```
\usepackage[colorlinks,bookmarksopen=true,
    bookmarksnumbered=true]{hyperref}
```

宏包选项也可以以\hypersetup的形式另起一行书写，键值包括：

colorlinks 默认 false，即加上带颜色的边框，[10]而不是更改文字的颜色。默认 linkcolor=red, anchorcolor=black, citecolor=green, urlcolor=magenta.

hidelinks 无参数，取消链接的颜色和边框。

bookmarks 默认 true，用于生成书签。

bookmarksopen 默认 false，是否展开书签。

bookmarksopenlevel 默认全部展开。设置为 secnumdepth 对应的值可以指定展开到这一级。比如对 report 指定 2，就是展开到 section 为止。

[10]这个边框在打印时并不会打印出来。

bookmarksnumbered 默认 false，书签是否带章节编号。
unicode 无参数，使用 UTF-8 编码时可以指定的选项。
pdftitle pdf 元数据：标题。
pdfauthor pdf 元数据：作者。
pdfsuject pdf 元数据：主题。
pdfkeywords pdf 元数据：关键词。
pdfstartview 默认值 Fit，设置打开 pdf 时的显示方式。Fit 适合页面，FitH 适合宽度，FitV 适合高度。

如果章节标题中带有特殊内容无法正常显示在 pdf 书签中，这样使用：

```
\section{质能公式\texorpdfstring{$E=mc^2$}{E=mc\textasciicircum 2}}
```

在加载了hyperref宏包后，可以使用的命令有：

```
% 文档内跳转
\hyperref[label-name]{print-text}
\autoref{label-name} % 自动识别label上方的命令
% 链接网站
\href{URL}{print-text}
\url{URL} %彩色可点击
\nolinkurl{URL} % 黑色可点击
```

其中\autoref命令会先检查\label引用的计数器，再在检查其 autref 宏是否存在。比如图表环境会检查是否有\figureautorefname这个宏，如果有则引用之；而正常的\ref命令只会引用\figurename。以下列出hyperref宏包支持的计数器宏（请自行插入）：

表 3.7: autoref 命令支持的计数器宏

命令	默认值	命令	默认值
\figurename	Figure	\tablename	Table
\partname	Part	\appendixname	Appendix
\equationname	Equation	\Itemname	item
\chaptername	chapter	\sectionname	section
\subsectionname	subsection	\subsubsectionname	subsubsection
\paragraphname	paragraph	\Hfootnotename	footnote
\AMSname	Equation	\theoremname	Theorem
\page	page. 但常使用\autopageref命令代替。		

比如，通过重定义\figureautorefname，就能用“图 3.1”的效果代替默认的“Figure 3.1”：

```
\renewcommand\figureautorefname{图}
```

另一个宏包nameref不满足于只引用编号，提供了引用对象的标题内容的功能。使用\nameref命令可以利用位于标题下方的标签来引用标题内容。

关于页码引用，如果想要生成“第 × 页，共 × 页”的效果，可能需要借助lastpage宏包。它提供的标签 LastPage 可以保证在输出页面的最后（如果你自行添加标签，可能还会有后续浮动体），因此可以：

```
This is page \thepage\ of
    \pageref{LastPage}
```

This is page 23 of 100

3.6.2 脚注、边注与尾注

■ 脚注

脚注是一种简单标注，使用方法是：

```
\footnote{This is a footnote.}
```

在某些环境内（如表格），脚注无法正常使用，可以先用\footnotemark依次插入位置，再在 tabular/table 环境外用\footnotetext依次指明脚注的内容。

minipage 环境是支持脚注，在其内部或正文内这样可以写表格脚注：

```
\begin{minipage}{\linewidth}
\begin{tabular}{l}
This is an exmaple\footnotemark.
\end{tabular}
\footnotetext{You don't need
    more.}
\end{minipage}
```

This is an exmaple[11].

You don't need more.

行文中切忌过多地使用脚注，它会分散读者的注意力。默认情况下脚注按章编号。脚注相关的命令：

```
% 在大纲或者\caption命令中使用脚注，需要加\protect
\caption{Titel\protect\footnote{This is footnote.}}
% 脚注之间的距离：\footnotesep
% 每页脚注之上横线：\footnoterule，默认值：
\renewcommand\footnoterule{\rule{0.4\columnwidth}{0.4pt}}
% 调整脚注到正文的间距，例如：
\setlength{\skip\footins}{0.5cm}
```

更多的参考footmisc宏包，比如其选项 perpage 让脚注编号每页清零。

■ 边注

LaTeX 的边注命令\marginpar不会进行编号。必选参数表示在页右显示边注；可选参数表示如果边注在偶数页，则在页左显示。例如右边这个音符：♫

```
这一行有边注\marginpar[左侧]{右侧}
```

如果想要改变边注的位置，使用`\reversemarginpar`命令。此外，有关边注的长度命令`\marginparwidth/sep/push`分别控制边注的宽、边注到正文的距离、边注之间的最小距离。可以使用`geometry`宏包来设置前两者，参考第3.10节。

■ 尾注

尾注用于注释较长、无法使用脚注的场合，需要`endnotes`宏包。

3.6.3 援引环境

援引环境有 quote 和 quotation 两个。前者首行不缩进；后者首行缩进，且支持多段文字。

```
鲁智深其师有偈言曰：
\begin{quote}
逢夏而擒，遇腊而执。
听潮而圆，见信而寂。
\end{quote}
圆寂之后，其留颂曰：
\begin{quotation}
平生不修善果，只爱杀人放火。
忽地顿开金绳，这里扯断玉锁。

咦！钱塘江上潮信来，今日方知我是我。
\end{quotation}
```

鲁智深其师有偈言曰：

> 逢夏而擒，遇腊而执。
> 听潮而圆，见信而寂。

圆寂之后，其留颂曰：

> 平生不修善果，只爱杀人放火。忽地顿开金绳，这里扯断玉锁。
>
> 咦！钱塘江上潮信来，今日方知我是我。

另外一个诗歌援引环境叫 verse，是悬挂缩进的。一般很少用到。

```
Rabindranath Tagore wrote this in
his \emph{The Gardener}:
\begin{verse}
Constant thrusts from your eyes
keep my pain fresh for ever.
\end{verse}
```

Rabindranath Tagore wrote this in his *The Gardener*:

> Constant thrusts
> from your eyes
> keep my pain
> fresh for ever.

3.6.4 摘要

article 和 report 文档类支持摘要,在`\maketitle`命令之后可以使用`abstract`环境。在单栏模式下，其相当于一个带标题的`quotation`环境，而这个标题可以通过重定义`\abstractname`更改；双栏下则相当于`\section*`命令定义的一节。

3.6.5 参考文献

参考文献主要使用的命令是`\cite`，与`\label`相似。通过`natbib`宏包的使用可以定制参考文献标号在文中的显示方式等格式，下面`natbib`宏包的选项含义为：数字编号、排序且压缩、上标、外侧方括号，总体像这样：`[1,3-5]`。[12]

```
\documentclass{ctexart}
% 如果是book类文档，把\refname改成\bibname
\renewcommand{\refname}{参考文献}
\usepackage[numbers,sort&compress,super,square]{natbib}
\begin{document}
This is a sample text.\cite{author1.year1,author2.year2}
This is the text following the reference.
% “99”表示以最多两位数来编号参考文献，用于对齐
\begin{thebibliography}{99}
    \addtolength{\itemsep}{-2ex} % 用于更改行距
    \bibitem{author1.year1}Au1. ArtName1[J]. JN1. Y1:1--2
    \bibitem{author2.year2}Au2. ArtName2[J]. JN2. Y2:1--2
\end{thebibliography}
\end{document}
```

当然以上只是权宜之计的书写方法。更详尽的参考文献使用（BibTeX 方法）在BibTeX 这一节进行介绍。

如果想要将参考文献章节正常编号，并加入到目录中，可以使用 `tocbibind` 宏包。注意，此时需要重命名`\tocbibname`（而不是 `\refname` 或 `\bibname`）来指定参考文献章节的标题。例如：

```
\usepackage[nottoc,numbib]{tocbibind}
\renewcommand{\tocbibname}{References}
```

该宏包对于将索引、目录本身、图表目录编入目录页同样有效。选项 `nottoc` 表示目录本身不编入，`notlof/lot` 表示图/表目录不编入，`notindex` 表示索引不编入，`notbib` 表示参考文献不编入。而选项 `numindex/bib` 表示给索引/参考文献章节正常编号。选项 `none` 表示禁用所有。

3.7 正式排版：封面、大纲与目录

3.7.1 封面

封面的内容在导言区进行定义，一般写在所有宏包、自定义命令之后。主要用到的如：

```
\title{Learning LaTeX}
\author{wklchris}
\date{text}
```

然后在`document`环境内第一行，写上：`\maketitle`，就能产生一个简易的封面。其中`\title`和`\author`是必须定义的，`\date`如果省略会自动以编译当天

[12]这里的 LaTeX 代码实际为：`\textsuperscript{ttfamily [1,3-5]}`

的日期为准，格式形如：January 1, 1970。如果你不想显示日期，可以写`\date{}`。

标题页的脚注用`\thanks`命令完成。

3.7.2 大纲与章节

LaTeX 中，将文档分为若干大纲级别。分别是：

`\part` 部分。这个大纲不会打断 chapter 的编号。

`\chapter` 章。基于 article 的文档类不含该大纲级别。

`\section` 节。

`\subsection` 次节。默认 report/book 文档类本级别及以下的大纲不进行编号，也不纳入目录。

`\subsubsection` 小节。默认 article 文档类本级别及以下的大纲不进行编号，也不纳入目录。

`\paragraph` 段。极少使用。

`\subparagraph` 次段。极少使用。

对应的命令例如：`\section{第一节}`。

以上各级别在 LaTeX 内部以“深度”参数作为标识。第一级别 part 的深度是 -1，以下级别深度分别是 $0, 1, \ldots$，类推。注意到由于 article 文档类缺少 chapter 大纲，其 part 深度又是从 0 开始的，故 section 及以下的深度数值与 book/report 文档类是一致的。

另外的一些使用技巧：

```
% 大纲编号到深度2，并纳入目录
\setcounter{tocdepth}{2}
% 星号命令：插入不编号大纲，也不纳入目录
\chapter*{序}
% 将一个带星号的大纲插入目录
\addcontentsline{toc}{chapter}{序}
% 可选参数用于在目录中显示短标题
\section[Short]{Looooooooong}
% 自定义章节标题名
\renewcommand{\chaptername}{CHAPTER}
```

book 文档类还提供了以下的命令：

`\frontmatter` 前言。页码为小写罗马字母，其后的章节不编号，但生成页眉页脚和目录项。

`\mainmatter` 正文。页码为阿拉伯数字；其后的章节编号，页眉页脚和目录项正常运作。

`\backmatter` 后记。页码格式不变，继续计数。章节不编号，但生成页眉页脚和目录项。

关于附录`\appendix`部分的大纲级别问题不在此讨论，请参考这一节。在 book 文档类中，附录一般放在正文与后记之间；当然你也可以在非 book 文档类中使用附录。关于章节样式自定义的问题，则请看这里。

3.7.3 目录

目录在大纲的基础上生成，使用命令\tableofcontents即可插入目录。目录在加载了hyperref宏包后，可以实现点击跳转的功能。你可以通过重定义命令更改\contentsname，即“目录”的标题名。

```
\renewcommand{\contensname}{目录}
```

你也可以插入图表目录，分别是\listoffigures，\listoftables。通过重定义\listfigurename和\listtablename可以更改图表目录的标题。如果要更改目录的显示的大纲级别深度，设置计数器:

```
\setcounter{tocdepth}{2} % 这是到subsection
```

想要将目录本身编入目录项，使用 tocbibind 宏包，参考26。
目录的高级自定义需要借助titletoc宏包，参考第5.5节。

3.8 计数器与列表

3.8.1 计数器

LaTeX 中的自动编号都借助于内部的计数器来完成。包括:
章节 part, chapter, section, subsection, subsubsection, paragraph, subparagraph
编号列表 enumi, enumii, enumiii, enumiv
公式和图表 equation, figure, table
其他 page, footnote, mpfootnote[13]
用\the 接上计数器名称的方式来调用计数器，比如\thechapter。如果只是想输出计数器的数值，可以指定数值的形式，如阿拉伯数字、大小写英文字母，或大小写罗马数字。常用的命令包括:

```
\arabic{counter-name}
\Alph \alph \Roman \roman
% ctex文档类还支持\chinese
```

比如本文的附录，对章和节的编号进行了重定义。注意：**章的计数器包含了节在内**。以下的命令写在appendices环境中（或者\appendix命令后），因此对于此外的编号不产生影响；同理你也可以这样对列表编号进行局部重定义。

```
\renewcommand{\thechapter}{\Alph{chapter}}
\renewcommand{\thesection}
    {\thechapter-\arabic{section}}
\renewcommand{\thefootnote}{[\arabic{footnote}]}
```

计数器的命令:

[13] \mpfootnote命令用于实现 minipage 环境的脚注。

```
% 父级计数器变化，则子级计数器重新开始计数
\newcounter{counter-name}[parent counter-name]
\setcounter{counter-name}{number}
\addtocounter{counter-name}{number}
% 计数器步进1，并归零所有子级计数器
\stepcounter{counter-name}
```

3.8.2 列表

LaTeX 支持的预定义列表有三种，分别是无序列表`itemize`，自动编号列表`enumerate`，还有描述列表`description`.

■ *itemize* 环境

例子：

```
\begin{itemize}
  \item This is the 1st.
  \item[-] And this is the 2nd.
\end{itemize}
```

- This is the 1st.
- And this is the 2nd.

每个`\item`命令都生成一个新的列表项。通过方括号的可选参数，可以定义项目符号。默认的项目符号是圆点 (`\textbullet`)。更多的方法参考这一节。

■ *enumerate* 环境

例子：

```
\begin{enumerate}
  \item First
  \item[Foo] Second
  \item Third
\end{enumerate}
```

1. First
Foo Second
2. Third

方括号的使用会打断编号，之后的编号顺次推移。更多的方法参考这一节。

■ *description* 环境

例子：

```
\begin{description}
  \item[LaTeX] Typesetting
    System.
  \item[wkl] A Man.
\end{description}
```

LaTeX Typesetting System.
wkl A Man.

默认的方括号内容会以加粗显示。更多的方法参考这一节。

3.9 浮动体与图表

3.9.1 浮动体

浮动体将图或表与其标题定义为整体，然后动态排版，以解决图、表卡在换页处造成的过长的垂直空白的问题。但有时它也会打乱你的排版意图，因此使用与否需要根据情况决定。

图片的浮动体是`figure`环境，而表格的浮动体是`table`环境。一个典型的浮动体例子：

```
\begin{table}[!htb]
    \centering
    \caption{table-cap}
    \label{table-name}
    \begin{tabular}{...}
        ...
    \end{tabular}
\end{table}
```

其中，浮动体环境的参数`!htb` 含义是：！表示忽略内部参数（比如内部参数对一页中浮动体数量的限制）；h、t、b 分别表示插入此处、插入页面顶部、插入页面底部，故 htb 表示优先插入此处，再尝试插入到某页顶，最后尝试插入到页底。此外还有参数 p，表示允许为浮动体单独开一页。LaTeX 的默认参数是 tbp. 请不要单独使用 htbp 中的某个参数，以免造成不稳定。

`\caption`命令给表格一个标题，写在了表格内容（即`tabular`环境）之前，表示标题会位于表格上方。对于图片，一般将把此命令写在图片插入命令的下方。注意：label 命令请放在 caption 下方，否则可能出现问题。

浮动体的自调整属性可能导致它“一直找不到合适的插入位置”，然后多个浮动体形成排队（因为靠前的浮动体插入后，靠后的才能插入）。如果在生成的文档中发现浮动体丢失的情况，请尝试更改浮动参数、去掉部分浮动体，或者使用`\clearpage`命令来清空浮动队列，以正常开始随后的内容。

如果希望浮动体不要跨过 section，使用：

```
\usepackage[section]{placeins}
```

其实质是重定义了`\section`命令，在之前加上了`\FloatBarrier`。你也可以自行在每个想要阻止浮动体跨过的位置添加。

3.9.2 图片

图片的插入使用`graphicx`宏包和`\includegraphics`命令，例子：

```
\begin{center}
    \includegraphics[width=0.8\linewidth]{ThisPic}
\end{center}
```

可选参数指定了图片宽度为 0.8 倍该行文字宽。类似地可以指定 height(图片高)，scale(图片缩放倍数)，angle(图片逆时针旋转角度)，origin(图片旋转中心

lrctbB[14]) 这样的命令。前三个命令不建议同时使用。旋转的图片基线会变化，故一般用 totalheight 代替 height.

对于 Thispic 这个参数的写法，XeLaTeX 支持 pdf, eps, png, jpg 图片扩展名。你可以书写带扩展名的图片名称 ThisPic.png，也可以不带扩展名。如果不给出扩展名，将按上述四个扩展名的顺序依次搜索文件。

如果你不想把图片放在 LaTeX 文档主文件夹下，可以使用下面的命令加入新的图片搜索文件夹：

```
\graphicspath{{c:/pics/}{./pic/}}
```

用正斜杠代替 Windows 正常路径中的反斜杠。你可以加入多组路径，每组用花括号括起，并确保路径以正斜杠结束。用./指代主文件夹路径，也可省略。

图文混排可参考wrapfig宏包，后文的箱子一节即是例子。

```
% \usepackage{wrapfig}
\begin{wrapfigure}[linenum]{place}[overhang]{picwidth}
    \includegraphics...
    \caption...
\end{wrapfigure}
```

各参数的含义：(1) ***linenum:*** （可选）图片所占行数，一般不指定；(2) ***place:*** 图片在文字段中的位置——R, L, I, O 分别代表右侧、左侧、近书脊、远书脊；(3) ***overhang:*** （可选）允许图片超出页面文本区的宽度，默认是 0pt。在该项可以使用\width代替图片的宽度，填入\width将允许把图片全部放入页边区域，以及：(4) ***picwidth:*** 指定图片的宽度，默认情形下图片的高度会自动调整。。

3.9.3 表格

LaTeX 原生的表格功能非常有限，甚至不支持单元格跨行和表格跨页。但是这些可以通过宏包longtable, supertabular, tabu等宏包解决。跨行的问题只需要multirow宏包。下面是一个例子（没有写在浮动体中）：

```
\begin{center}
  \begin{tabular}[c]{|l|c||p{3em}
    r@{-}} \hline\hline
    A & B & C & d\\D & E & F &
    g\\
    \cline{1-2}
    \multicolumn{2}{|c|}{G}&H&i\\
    \hline
  \end{tabular}
\end{center}
```

A	B	C	d-
D	E	F	g-
G		H	i-

各参数的说明如下：

[14]这六个字母分别代表左、右、中、顶、底，以及基线。

- 可选参数**对齐方式**：[t] 表示表格上端与所在行的网格线对齐。如果与它同一行的有文字的话，文字是与表格上端同高的。如果使用参数 [b]，就是下端同高。[c] 是中央同高。t=top, b=buttom, c=center.
- 必选参数**列格式**：用竖线符号“|”来表示竖直表线，连续两个“|”表示双竖直表线。最右边留空了，表示没有竖直表线。或者你可以使用“@{}”表示没有竖直表线。你也可以用“@{-}”这样的形式把竖直表线替换成“-”，具体效果不再展示。而此处的 l、c、r 分别表示从左往右一共三列，分别**左对齐、居中对齐、右对齐**文字。在使用 l、c、r 时，表格宽度会自动调整。你可以用”p2em” 这样的命**指定某一列的宽度**，这时文字自动左对齐。注意：单元格中的文字默认向上水平表线对齐，即竖直居上。
- 在 tabular 环境内部，命令`\hline`来绘制水平表线。命令`\cline{i-j}`用于绘制横跨从 i 到 j 行的水平表线。两个连续的`\hline`命令可以画双线，但是双线之间相交时可能存在问题。
- 在 tabular 环境内部，命令”`&`” 用于把光标跳入该行下一列的单元格。每行的最后请使用两个反斜杠命令跳入下一行。命令`\hline`或`\cline`不能算作一行，因此它们后面没有附加换行命令。
- 在 tabular 环境内部，跨列命令`\multicolumn{number}{format}{text}`用于以 format 格式合并该行的 number 个单元格，并在合并后的单元格中写入文本 text。如果一行有了跨列命令，请注意相应地减少”`&`” 的数量。

文章中出现了表格，几乎就一定会加载`array`宏包。在`array`宏包支持下，`cols` 参数除了 l, c, r, p{}, @{} 以外，还可以使用：

- `m{}`，`b{}`: 指定宽度的竖直居中，居下的列。
- `>{decl}`，`<{decl}`: 前者用在 lcrpmb 参数之前，表示该列的每个单元格都以此 decl 命令开头；后者用于结尾。比如[15]：

```
\begin{tabular}{|>{\centering\ttfamily}p{5em}
    |>{$}c<{$}|}
...
\end{tabular}
```

- `!{symbol}`: 使用新的竖直表线，类似于原生命令 `@{}`，不同在于`!{}` 命令可以在列间保持合理的空距，而`@{}` 会使两列紧贴。

你甚至可以自定义 lcrpmb 之外的列参数，但需要保证是单字母。比如定义某一列为数学环境：

```
\newcolumntype{T}{>{$}c<{$}}
```

■ array，multirow 宏包

来一个`array`宏包下的例子：

[15]例中的`\centering`命令后可加入`\arraybackslash`以应对可能的表格换行命令异常。

```latex
% 记得\usepackage{array}
\begin{tabular}{|>{\setlength
  \parindent{5mm}}m{1cm}|
  >{\large\bfseries}m{1.5cm}|
  >{$}c<{$}|}
  \hline A & 2 2 2 2 2 2 & C\\
  \hline 1 1 1 1 1 1  & 10 &
    \sin x \\ \hline
\end{tabular}
```

A	**2 2 2 2 2 2**	C
1 1 1 1 1 1	**10**	$\sin x$

然后一个跨行跨列的例子。如果同时跨行跨列，必须把 multirow 命令放在 multicolumn 内部。用`\multirow`和`\multicolumn`作用于单独的 1 行或 1 列，能临时改变某单元格的对齐方式。如果用星号代替列样式，表示自适应宽度。

```latex
% \usepackage{multirow}
\begin{center}
\begin{tabular}{|c|c|c|}
  \hline
  \multirow{2}{2cm}{A Text!}
    & ABC & DEF \\
  \cline{2-3} & abc & def \\
  \hline
  \multicolumn{2}{|c|}
    {\multirow{2}*{Nothing}} &
    XYZ \\
  \multicolumn{2}{|c|}{} & xyz \\
  \hline
\end{tabular}
\end{center}
```

<table>
<tr><td rowspan="2">A Text!</td><td>ABC</td><td>DEF</td></tr>
<tr><td>abc</td><td>def</td></tr>
<tr><td colspan="2" rowspan="2">Nothing</td><td>XYZ
xyz</td></tr>
</table>

表格的第一个单词是默认不断行的，这在单元格很窄而第一个词较长时会出现问题。可以通过下述方法解决：

```latex
% \usepackage{array}
\newcolumntype{P}[1]{>{#1
  \hspace{0pt}\arraybackslash}
  p{14mm}}
% \arraybackslash用于修复换行符
\begin{center}
\begin{tabular}{|P{\raggedleft}|}
  \hline Superconsciousness \\
    \hline
\end{tabular}
\end{center}
```

Super- con- scious- ness

此外，**表格还可以嵌套**，以方便地“拆分单元格”。注意下例中如何确保嵌套单元格表线显示正常：

```
\begin{tabular}{|c|l|c|}
\hline
a & bbb & c \\ \hline
a & \multicolumn{1}{@{}l@{}|}
{\begin{tabular}{c|c}
a & b \\ \hline
aa & bb \\
\end{tabular}}
& c \\ \hline
a & b & c \\ \hline
\end{tabular}
```

a	bbb		c
a	a	b	c
	aa	bb	
a	b		c

用\firsthline和\lasthline能够解决行内表格竖直方向对齐问题。

■ makecell 宏包

宏包makecell提供了一种方便在单元格内换行的方式，并可以配合参数 tblrc；带星表示有更大的竖直空距。此外，命令\multirowcell由multirow宏包与该宏包共同支持。命令\thead则有更小的字号，通常用于表头。

```
\begin{tabular}{|c|c|}
\hline
\thead{双行\\表头} &
    \thead{双行\\表头}\\
\hline
\multirowcell{2}{简单\\粗暴} &
    \makecell[l]{ABCD\\EF} \\
\cline{2-2} &
    \makecell*{更大的竖直空距} \\
\hline
\end{tabular}
```

双行 表头	双行 表头
简单 粗暴	ABCD EF
	更大的竖直空距

该宏包还提供了\Xhline和\Xcline命令，可以指定横线的线宽。例如模仿三线表[16]：

```
\begin{tabular}{ccc}
\Xhline{2pt}
\multirow{2}*{X} &
\multicolumn{2}{c}{Hey}\\
\Xcline{2-3}{0.4pt}
& Left & Right \\
\Xhline{1pt}
a & A & B \\
b & C & D \\
\Xhline{2pt}
\end{tabular}
```

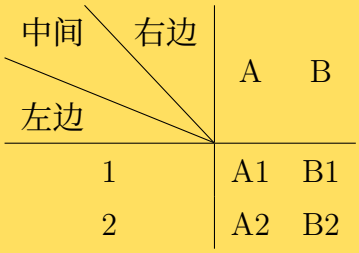
X	Hey	
	Left	Right
a	A	B
b	C	D

[16] 更正规的三线表绘制，参考后文的booktabs宏包。

■ diagbox 宏包

该宏包提供了分割表头的命令\diagbox。虽然斜线表头并不是规范的科技排版内容，但是在许多场合也可能用到。命令支持两或三参数。

```
\begin{tabular}{c|cc}
\diagbox{左边}{中间}{右边} & A &
    B \\
\hline
1 & A1 & B1 \\
2 & A2 & B2
\end{tabular}
```



■ 其他

关于表格的间距：

- \tabcolsep或者\arraycolsep控制列与列之间的间距,取决于你使用tabular还是array环境。默认 6pt。
- 列格式 @ 能够去除列间的空距，比如“@{}”。而命令\extracolsep{1pt}于 @ 的参数中，那么会将其右侧的列间隔都增加 1pt。
- 表格内行距用\arraystretch控制，默认为 1。

一些其他的使用技巧：

1. 输入同格式的列：参数|*{7}{c|}r|，相当于 7 个居中和 1 个居右。
2. 表格重音：原本的重音命令\`, \' 与\=，改为\a`, \a' 与\a=。
3. 控制整表宽度：tabularx宏包提供\begin{tabular*}{width}[pos]{cols}，比如你可以把 width 取值为\0.8\linewidth之类。
4. 如想实现单元格内换行，使用makecell宏包支持的\makecell命令。
5. 宏包dcolumn提供了新的列对齐方式 D，并调用array宏包。故你可以利用后者支持的命令，这样定义：

```
% 表示输入小数点、显示为小数点、支持小数点后2位
\newcolumntype{d}{D{.}{.}{2}}
% 使用 d{2} 这样的参数进行控制
\newcolumntype{d}[1]{D{.}{.}{#1}}
```

注意：

- 表头请用\multicolumn1c类似的语句进行处理。
- 第三参数不能帮你截取、舍入，只用于预设列宽；小心超宽。
- 第三参数可以是“-1”，表示小数点居中；可以形如“2.1”，表示在小数点左侧预留 2 位宽、右侧预留 1 位宽。

3.9.4 非浮动体图表和并排图表

如果不使用浮动体，又想给图、表添加标题，请在导言区加上：

```
\makeatletter
\newcommand\figcaption{\def\@captype{figure}\caption}
\newcommand\tabcaption{\def\@captype{table}\caption}
\makeatother
```

这部分是底层的 TeX 代码，在此就不多介绍了。在如上定义后，你可以在浮动体外使用`\figcaption`和`\tabcaption`命令。注意：为了防止标题和图表不在一页，可以用`minipage`环境把它们包起来。

同样的，如果排版并排图片，请用`minipage`把每个图包起来，指定宽度，然后放在浮动体内。注意灵活运用`\\[10ex]` 这样的命令来排版 2×2 的图片。

如果需要给每个图片定义小标题，参考`subfig`宏包的相关内容。这里给一个简单的例子：

```
\begin{figure}
\centering
\subfloat[...]{\label{sub-fig-1}
    \begin{minipage}
    \centering
    \includegraphics[width=...]{...}
    \end{minipage}}
\quad\subfloat[...]
```

3.10 页面设置

3.10.1 纸张、方向和边距

主要借助`geometry`宏包。先看一张页面构成，如图3.1：

geometry 宏包的具体的选项参数有：

paper=<papername> ：其中纸张尺寸有 [a0–a6, b0–b6, c0–c6]paper, ansi[a–e]paper, letterpaper, executivepaper, legalpaper.

papersize={<width>,<height>} ：自定义尺寸。也可以单独对 paperwidth 或者 paperheigth 赋值。

landscape ：切换到横向纸张。默认的是 portrait.

body 部分分为两个概念：一个是总文本区 (total body)，另一个是主文本区 (body). 总文本区可以由主文本区加上页眉 (head)、页脚 (foot)、侧页边 (marginalpar) 组成。默认的选项为 includehead, 表示总文本区包含页眉。要包括其他内容, 可使用：`includefoot`，`includeheadfoot`，`includemp`，`includeall`，以及以上各个参数将 include 改为 ignore 后的参数。

总文本区在默认状态下占纸张总尺寸的 0.7，由 scale=0.7 控制，你也可以分别用 `hscale` 和 `vscale` 指定宽和高的占比。用具体的长度定义也是可以的，使用 `(total)width` 和 `(total)height` 定义总文本区尺寸，或者用 `textwidth` 和 `textheight` 定义主文本区的尺寸[17]。或者直接用 `total={width,height}`，`body={width,height}` 定义。甚至你可以用 `lines=<num>` 行数指定 textheight。

[17]当 totalwidth 和 textwidth 都定义时，优先采用后者的值。

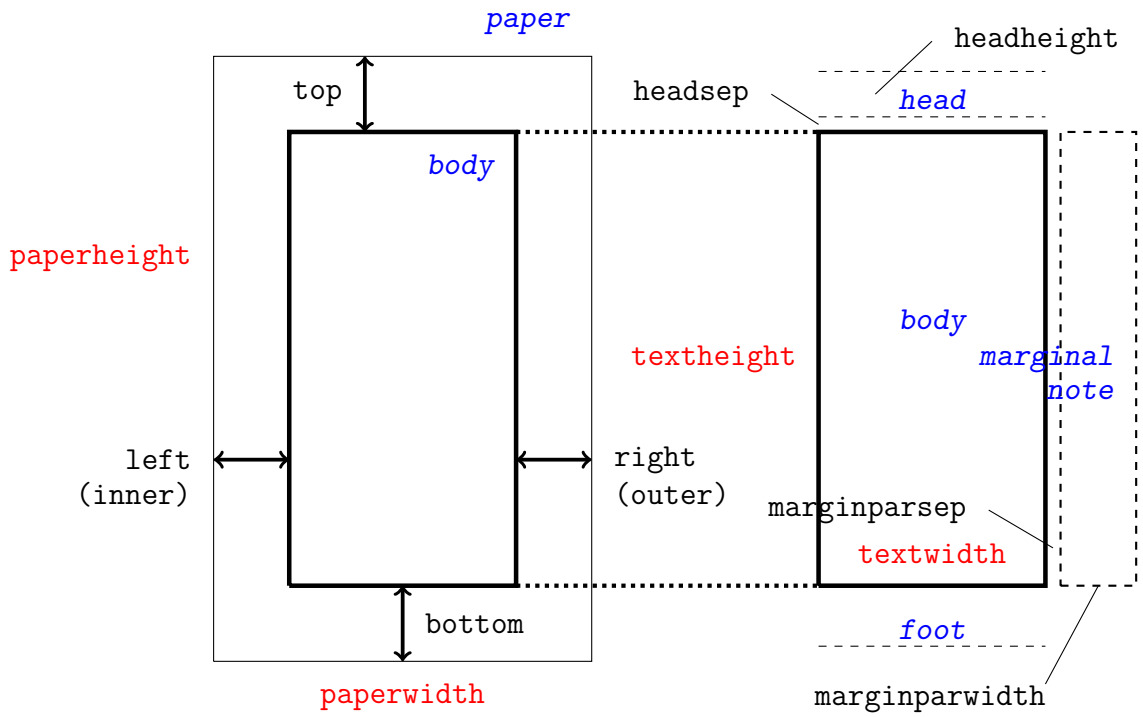


图 3.1: 页面构成示意图

页边的控制最为常用,分别用 left/inner, right/outer, top, bottom 来定义四向的页边。其中 inner, outer 参数只在文档的 twoside 参数启用时才有意义。你可以用 hmarginratio 来给定 left(inner) 与 right(outer) 页边宽的比例,默认是单页 1:1、双页 2:3。top 和 bottom 之间的比由 vmarginratio 给定。你也可以用 vcentering, hcentering, centering 来指定页边比例为 1:1. 在文档的左侧（内侧），可以指定装订线宽度 bindingoffset，使页边不会侵入。

页眉和页脚是位于 top 和 bottom 页边之内的文档元素。对于页眉和页脚的高度，分别使用 headheight/head, footskip/foot 参数指定。hmargin, vmargin 来指定侧两和顶底的边距。它们到主文本区的参数分别是 headsep, footnotesep, marginparsep. 你可以用 nohead, nofoot, nomarginpar 参数来清除总文本区中的页眉，页脚和侧页边。

对于在文档类 documentclass 命令中能使用的参数，geometry 有不少也能做。比如 twoside, onecolumn, twocolumn。甚至还能用文档类中不能用的 columnsep（启用多栏分隔线）。

最后，这是几个小例子：

```
% 与Microsoft Word的默认样式相同:
\usepackage[hmargin=1.25in,vmargin=1in]{geometry}
% 书籍中靠书脊一侧的边距较小:
\usepackage[inner=1in,outer=1.25in]{geometry}
```

3.10.2 页眉和页脚

主要借助 fancyhdr 宏包。LaTeX 中的页眉页脚定义主要借助了两个命令，一个是`\pagestyle`，参数有：

empty 无页眉页脚。
plain 无页眉，页脚只包含一个居中的页码。
headings 无页眉，页脚包含章/节名称与页码。
myheadings 无页眉，页脚包含页码和用户定义的信息。

另一个命令是`\pagenumbering`，与计数器一样，拥有 `arabic`，`[Rr]oman`，`[Aa]lph` 五种页码形式。

`fancyhdr`宏包给出了一个叫 fancy 的`\pagestyle`，将页眉和页脚分别分为左中右三个部分，分别叫`\lhead`, `\chead`, `\rhead`, 以及类似的 [lcr]foot. 页眉页脚处的横线粗细也可以定义，默认页眉为 0.4pt、页脚为 0pt. 下面是一个例子：

```
\usepackage{fancyhdr}
\pagestyle{fancy}
    \lhead{}
    \chead{}
    \rhead{\bfseries wklchris}
    \lfoot{Leftfoot}
    \cfoot{\thepage}
    \rfoot{Rightfoot}
\renewcommand{\headrulewidth}{0.4pt}
\renewcommand{\footrulewidth}{0.4pt}
```

加载这个宏包，更多地是为了解决双页（twoside）文档的排版问题。对于双页文档，`fancyhdr`宏包给出了一套新的指令：用 E, O 表示单数页和双数页，L, C, R 表示左中右，H, F 表示页眉和页脚。其中 H, F 需要配合`\fancyhf`命令使用。如果不使用 H, F 这两个参数，也可以使用`\fancyhead`, `\fancyfoot`两个命令代替。一个新的例子：

```
\fancyhead{} % 清空页眉
    \fancyhead[RO,LE]{\bfseries wklchris}
\fancyfoot{} % 清空页脚
    \fancyfoot[LE,RO]{Leftfoot}
    \fancyfoot[C]{\thepage}
    \fancyfoot[RE,LO]{Rightfoot}
```

该宏包在定义双页文档时，采用了如下的默认设置：

```
\fancyhead[LE,RO]{\slshape \rightmark}
\fancyhead[LO,RE]{\slshape \leftmark}
\fancyfoot[C]{\thepage}
```

上例中的`\rightmark`表示较低级别的信息，即当前页所在的 section，形式如“1.2 sectionname ”，对于 article 则是 subsection；而`\leftmark`表示较高级别的信息，即对应的 chapter，对于 article 则是 section. 命令`\leftmark`包含了页面上`\markboth`[18]下的最后一条命令的左参数，比如该页上出现了 section

[18] `\markboth`是一个会被`\chapter`等命令调用的命令，默认右参数是空。注意，带星号的大纲不调用这

1–2，那么 leftmark 就是“Section 2”；命令\rightmark则包含了页面上的第一个\markboth命令的右参数或者第一个\markright命令的唯一参数，比如可能是“Subsection 1.2”。

这听起来可能难以理解，但是\markboth命令有两个参数，分别对应显示在文档的左页和右页（但是默认右参数留空，用\markright去指定右页），故有左右之分；而\markright命令只有一个参数。你可以试着再去理解一下双页文档下的宏包的默认设置。利用这一点来重定义 chaptermark(book/report), sectionmark, subsectionmark(article) 命令，举个例子：

```
% 这里的参数#1是指输入的section/chapter的标题
% 效果：“1.2. The section”
\renewcommand{\sectionmark}[1]{\markright{\thesection.\ #1}}
% 效果：“CHAPTER 2. The chapter”
\renewcommand{\chaptermark}[1]{\markboth{\MakeUppercase{%
    \chaptername}\ \thechapter.\ #1}{}}
```

如果你对于默认的\pagestyle不满意，可以用\fancypagestyle命令进行更改。例如更改 plain 页面类型：

```
\fancypagestyle{plain}{
    \fancyhf{} % 清空页眉页脚
    \fancyhead[c]{\thesection}
    \fancyfoot{\thepage}}
```

3.11 抄录与代码环境

抄录是指将键盘输入的字符（包括保留字符和空格）不经过 TeX 解释，直接输出到文档。默认的字体参数是等宽字族（ttfamily）。用法是\verb(*)命令或者verbatim(*)环境，区别在于带星号的会将空格以“␣”（\textvisiblespace）的形式标记出来。

注意，\verb命令是一个特殊的命令，可以用一组花括号括住抄录内容，也可以任意两个同样的符号（但不能是 *）。比如：

```
\verb|fooo{}bar|
\verb+fooo{}bar+
```

\verb(*)以及verbatim(*)环境很脆弱，不能隐式地用于自定义环境，也一般不能用作命令的参数。verbatim宏包提供了更多的抄录支持，fancyvrb宏包提供了\SaveVerb, \UseVerb命令，以及便于实现居中的BVerbatim环境（置于center环境内即可），详情读者可自行查阅。

宏包shortverb支持以一对符号代替\verb命令，比如竖线号：

```
% \usepackage{shortverb}
\MakeShortVerb|
Verbatim between this pair of verts: |#\?*^|
```

一命令，你需要这样书写：\chapter*{This\markboth{This}{}}。

代码环境的输出，比如本文中带行号的代码块，参见这一节。

3.12 分栏

这部分内容使用文档类的 twocolumn 可选参数就能实现。在 LaTeX 的双栏模式下，\newpage命令只能进行换栏操作，而\clearpage命令才会换进行换页操作。同时，文中随时可以使 用\twocolumn或 者\onecolumn命令执行换页、清空浮动队列，并切换分栏模式。在双栏上方的跨栏内容，如摘要，可以写在\twocolumn[…]可选参数中。

栏之间的间距由\columnsep控制；栏宽为\columnwidth，但请不要手工修改这个值。它可以被用作参数传递给其他命令。栏之间的分隔线宽由长度\columnseprule给出，默认值为 0pt，一般需要可以将其设置为 0.4pt。

如果在同一页内需要分栏与单栏并存，或者想要分成多栏，可以尝试使用multicol宏包。它提供一个支持任意多栏、但是边注和浮动体[19]无法使用的环境。比如本节：

```
\begin{multicols}{2}
  [\section{分栏}]
  ...
\end{multicols}
```

同时，该宏包会对齐每一栏的下边缘；在该环境下，使用\columnbreak来强制切换到新的一栏。还需要指出的是，该宏包并不保证各栏之间每行的网格都是对齐的。如果你需要此功能，可以参考grid宏包。

3.13 文档拆分

文档拆分只需要在主文件中使用\input{filename.tex}或\include{filename}命令,后者不写扩展名默认为.tex。两者区别在于\include命令将会插入\clearpage再读取文件。

拆分的优势在于可以根据 chapter（或其他）分为多个文件，省去了长文档浏览时的一些不便。你也可以把整个导言区做成一个文件，然后在不同的 LaTeX 文档中反复使用，即充当模板的功能。你还可以把较长的 tikz 绘图代码写到一个 tex 中，在需要时\input即可。

在导言区定义\includeonly加上 filename，可以确保只引入列表中的文件。在被引入文件的最后加入\endinput命令，其后的内容会被忽略。

一种较规范的拆分文件的文件头，以本文的章节放在次级目录中为例：

```
%!TEX root = ../LaTeX-cn.tex
```

[19]带星号的浮动体或许可以使用，如figure*，但参数 h 会失效。

3.14 西文排版及其他

3.14.1 连写

LaTeX 排版以及正规排版中，如果你输入 ff, fl, fi, ffi 等内容，它们默认会连写。在字母中间插入空白的箱子以强制不连写：f `\mbox{}` l。

3.14.2 断词

行末的英文单词太长，LaTeX 就会以其音节断词。如果你想指定某些单词的断词位置，使用如下命令断词。例子：

```
\hyphenation{Hy-phen-a-tion FORTRAN}
```

这个例子允许 Hyphenation, hyphenation 在短横处断词，同时**禁止** FORTRAN, Fortran, fortran 断词。如果你在行文中加入\-命令，则可以实现允许在对应位置断词的效果。比如：

```
I will show you this:
su\-per\-cal\-i\-frag\-i\-lis\-%
tic\-ex\-pi\-al\-i\-do\-cious
```

I will show you this: supercali-fragilisticexpialidocious

如果你不想断词，比如电话号码，巧妙利用`\mbox`命令吧：

```
My telephone number is: \mbox{012 3456 7890}
```

3.14.3 硬空格与句末标点

如果你想在某个不带参数的命令后输入空格，请接上一个空的花括号确保空格能够正常输出。例如：`\这是\TeX{} Live.`

在 LaTeX 中还有一个命令“\␣”，用于产生一个硬空格（区别于软空格\space），所以你也可以用`\TeX\␣Live`。

西文排版下，LaTeX 会判断一种**句末标点**，即小写字母后的“.”，“?”或者“!”三个英文标点。句末标点后如果键入空格，LaTeX 会自动增加空格的距离。如果句子以大写字母结尾，LaTeX 会认为这是人名而不增加空格，这时候需要手动添加命令`\@`：

```
OK. That's fine.\\
OK\@. That's fine.
```

OK. That’s fine.
OK. That’s fine.

相反，有些并非句末标点的情况会被识别为句末标点，这时候需要在标点后插入一个`\␣`或者~ 来缩小间距；区别在于前者允许断行，后者不允许。

```
Prof. Smith is a nice man.\\
Prof.~Smith is a nice man.
```

Prof. Smith is a nice man.
Prof. Smith is a nice man.

在标点后使用\frenchspacing命令，可以调整为极小的空距。这个命令在排版参考文献列表时可能被使用。

在 XeLaTeX 编译模式下的中文字符，与西文或者符号之间会产生默认的空距[20]。如果你不想要这个空距，把中文放在\mbox内即可，比如：

```
1 \mbox{例子}-1
```

例子-1

3.14.4 特殊符号

符号的总表可以参照 symbols-a4 文档，运行 texdoc symbols-a4 即可调出。包括希腊字母在内的一些数学符号将会在下一章介绍。这里给出基于wasysym宏包的一些常用符号：

表 3.8: wasysym 宏包符号

‰	\permil	♂	\male	♀	\female
✓	\checked	☒	\XBox	☑	\CheckedBox
✳	\hexstar	☎	\phone	♫	\twonotes

[20]这个问题在 ctex 文档类下似乎被已解决。

4 数学排版

4.1 行间与行内公式

行内公式指将公式嵌入到文段的排版方式，主要要求公式垂直距离不能过高，否则影响排版效果。行内公式的书写方式：

```
$...$ 或者 \(...\) 或者 \begin{math}...\end{math}
```

一般推荐第一种方式。例如：`$\sum_{i=1}^{n}a_i$`，即：$\sum_{i=1}^{n} a_i$.

另外一种公式排版方式是**行间公式**，也称行外公式，使用：

```
\[...\] 或者 \begin{displaymath}...\end{displaymath}
或者 amsmath 提供的 \begin{equation*}...\end{equation*}
```

一般也推荐第一种命令[1]，例如：`\[\sum_{i=1}^n{a_i}\]`，得到：

$$\sum_{i=1}^{n} a_i$$

从上面的两个例子可以看出，即使输出相同的内容，行内和行间的排版也是有区别的，比如累计符号上标是写在正上方还是写在右上角。

如果行间公式需要编号，使用`equation`环境[2]，还可以插入标签：

[1]还有一种 `$$...$$` 的写法，源自底层 TEX，不建议使用。

[2]需要注意有一个已被放弃的多行公式编号环境叫 `eqnarray`，请不要再使用。

```
\begin{equation}
\label{eq:NoExample}
  |\epsilon|>M
\end{equation}
```

$$|\epsilon| > M \tag{4.1}$$

4.2 数学字体、字号与空格

4.2.1 空格

在数学环境中，行文空格是被忽略的。比如`$x,y$` 和`$x， y$` 并没有区别。数学环境有独有的空格命令，最后一个是 $-3/18$ 的空格：

```
  $没有空格,3/18空\,格$ \\
  $4/18空\:格,5/18空\;格$ \\
  $9/18空\ 格,一个空\quad 格$ \\
  $两个空\qquad 格,负3/18空\!格$
```

$没有空格,3/18空格$
$4/18空格,5/18空格$
$9/18空格,一个空\quad 格$
$两个空\qquad 格,负3/18空格$

事实上，以上命令也可以在数学模式外使用，其中使用最广泛的是`\,`，比如上文提到过的千位分隔符。在数学环境中它也应用广泛：

```
\[ \int_0^1 x \ud{}x
= \frac{1}{2} \]
```

$$\int_0^1 x \,\mathrm{d}x = \frac{1}{2}$$

其中`\ud` 命令是自定义的，这也是微分算子的正常定义：

```
\newcommand{\ud}{\mathop{}\negthinspace\mathrm{d}}
```

4.2.2 间距

命令`\abovedisplayskip`和`\belowdisplayskip`控制了行间公式与上下文之间的间距，并且该值不会随字号调整而调整。有时你需要自行指定。默认值 `12pt plus 3pt minus 9pt`。多行公式之间的间距用`\jot`来控制，默认 `3pt`。命令`\mathsurround`给出了行内公式与文字间，除了预留空格之外的间距，默认值为 `0pt`。另外一个有趣的命令`\smash`，可以忽略参数的全高：

```
\[\underline{\smash{\int f(x)\ud
    x}}=1\]
```

$$\underline{\smash{\int f(x)\,\mathrm{d}x}} = 1$$

也能够通过参数，单独忽略参数的高度 (t) 或深度 (b)：

```
$\sqrt{A_{n_k}} \qquad
\sqrt{\smash[b]{A_{n_k}}}$
```

$\sqrt{A_{n_k}} \qquad \sqrt{\smash[b]{A_{n_k}}}$

4.2.3 字号

LaTeX 提供四种字号尺寸命令：

`\displaystyle` 行间公式尺寸。如 $\sum_{i=1}^{n} a$

`\textstyle` 行内公式尺寸。如 $\sum_{i=1}^{n} a$

`\scriptstyle` 上下标尺寸。如 $\sum_{i=1}^{n} a$

`\scriptscriptstyle` 次上下标尺寸。如 $\sum_{i=1}^{n} a$

4.2.4 数学字体

将字体转为正体使用`\mathrm`命令。如需保留空格，使用`\textrm`命令——这与正文一致。但是，`\textrm`命令内的字号可能不会自适应，`\mathrm`则表现起来稳定得多。

例如自然对数的底数 e，在本文中就是这样定义的：

```
\newcommand{\ue}{\mathrm{e}}
```

以下简单介绍几种数学字体。数学字体的总表参见表4.1。

■ 数学粗体

数学粗体使用`amsmath`宏包支持的`\boldsymbol`命令。命令`\boldmath`的问题在于它只能加粗一个数学环境，其中很可能包括了标点符号，而这是不严谨的。命令`\mathbf`就差的更远，它只能把字体转为正粗体，而数学字体都是斜体的。

```
$\mu,M$\\ $\boldsymbol{\mu},
\boldsymbol{M}$
```

μ, M
$\boldsymbol{\mu}, \boldsymbol{M}$

■ 空心粗体

空心粗体使用`amsfonts`或`amssymb`宏包的`\mathbb`命令。这里用`\textrm`而不是`\mathrm`，是为了保留空格。

```
$x^2 \geq 0 \qquad
\textrm{for all }x\in\mathbb{R}$
```

$x^2 \geq 0 \qquad \textrm{for all } x \in \mathbb{R}$

4.3 基本命令

基本函数默认用正体书写，包括：

```
\sin \cos \tan \cot \arcsin \arccos \arctan \cot \sec \csc
\sinh \cosh \tanh \coth \log \lg \ln \ker \exp \dim \arg \deg
\lim \limsup \liminf \sup \inf \min \max \det \Pr \gcd
```

以上函数，最后一行的 10 个是可以带上下限参数的，即在行间公式模式下，上标和下标将在函数正上方和正下方书写内容。

amsmath宏包允许\DeclareMathOperator命令自定义基本函数，用法类似于\newcommand命令。如果命令带星号\DeclareMathOperator*，则可以带上下限参数。

此外有一个叫\mathop的命令，可以把参数转换为数学对象，使其能够堆叠上下标；\mathbin与\mathrel则分别能把参数转换为二元运算符、二元关系符，并正确设置两侧的空距。

4.3.1 上下标与虚位

用低划线和尖角符表示上标和下标，请仔细体会下述例子：

```
$a^3_{ij}$ \\
${a_{ij}}^3\text{或}a_{ij}{}^3$\\
$\mathrm{e}^{x^2}\geq 1$
```

a_{ij}^3
$a_{ij}{}^3$或$a_{ij}{}^3$
$\mathrm{e}^{x^2}\geq 1$

上面的指数 3 的位置读者可以多多体会一下。此外，\phantom被称为虚位命令，从下例你也能够体会到他的作用：

```
${}^{12}_{6}\mathrm{C}$ \\
${}^{12}_{\phantom{1}6}
\mathrm{C}$ \\
$a^3_{ij}$ \\
$a^{\phantom{ij}3}_{ij}$
```

${}^{12}_{6}\mathrm{C}$
${}^{12}_{6}\mathrm{C}$
a^3_{ij}
a^{3}_{ij}

宏包mathtools提供了\prescript来避免手工调整：

```
$\prescript{12}{6}{\mathrm{C}}$
```

${}^{12}_{6}\mathrm{C}$

4.3.2 微分与积分

导数直接使用单引号'，积分使用\int符号：

```
$y'=x \qquad \dot{y}(t)=t$ \\
$\ddot{y}(t)=t+1$
$\dddot{y}+\ddddot{y}=0$ \\
$\iint_{D}f(x)=0$
$\int_{0}^{1}f(x)=1$
```

$y'=x \qquad \dot{y}(t)=t$
$\ddot{y}(t)=t+1$ $\dddot{y}+\ddddot{y}=0$
$\iint_{D}f(x)=0$ $\int_{0}^{1}f(x)=1$

有时候需要更高级的微分或积分号，其中\ud命令在上文这里定义过：

```
\[\left.\frac{\ud y}{\ud
    x}\right|_{x=0}\quad
\frac{\partial f}{\partial x}
\quad\oint\;\varoiint_S \]
```

$$\left.\frac{\mathrm{d}y}{\mathrm{d}x}\right|_{x=0}\quad\frac{\partial f}{\partial x}\quad\oint\;\varoiint_S$$

其中的\dot系的导数形式 LaTeX 只原生支持到二阶导数。后面的三阶、四阶需要amsmath宏包。\int系的积分命令类似。而环形双重积分命令\varoiint需要esint宏包[3]。

\left.或\right.命令[4]只用于匹配，本身不输出任何内容。

4.3.3 分式、根式与堆叠

分式使用\frac命令。或者amsmath宏包支持的\dfrac、\tfrac命令来强制获得行间公式、行内公式大小的分数。如果想自定义分式样式，参考第4.3.8节一节的\genfrac命令。

```
\[\frac{x}{y}+\dfrac{x}{y}
+\tfrac{a}{b}\]
```

$$\frac{x}{y}+\dfrac{x}{y}+\tfrac{a}{b}$$

该宏包还支持另一个命令\cfrac，用于输入连分式。

```
\[\cfrac{1}{1+\cfrac{2}{1+x}}\]
```

$$\cfrac{1}{1+\cfrac{2}{1+x}}$$

空根式用\surd输出，更常用的是\sqrt：

```
$\sqrt{2} \qquad \surd$\\
$\sqrt[\beta]{k}$
```

$\sqrt{2}$ $\qquad$ $\surd$
$\sqrt[\beta]{k}$

开方次数的位置可以用这两个命令微调，参数是整数：

```
$\sqrt[\leftroot{-2}\uproot{2}
    \beta]{k}$
```

$\sqrt[\leftroot{-2}\uproot{2}\beta]{k}$

划线命令使用\underline和\overline，水平括号使用 brace 或者 bracket 代替 line，例如\underbrace：

```
$\overline{m+n}$ \\
$\underbrace{a_1+\ldots+a_n}_{n}$
$\overbrace{a_1+\ldots+a_n}^{n}$
% 可选参数：线宽；竖直空距
$\underbracket[0.4pt][1ex]
  {a_1+\cdots+a_n}_n$
```

$\overline{m+n}$
$\underbrace{a_1+\ldots+a_n}_{n}$ $\overbrace{a_1+\ldots+a_n}^{n}$
$\underbracket[0.4pt][1ex]{a_1+\cdots+a_n}_n$

两个互有重叠的括号需要一个箱子命令\rlap，会在后面提到。不过在 j 之前的空距有些异常，可能需要\,进行修正。

[3]该宏包可能与amsmath冲突，即便使用也请其放在amsmath之后加载。
[4]参考定界符部分的内容。

```
\[b+\rlap{$\overbrace{\phantom{
  c+d+e+f+g}}^x$}c+d+\underbrace{
  e+f+g+h+i}_y+\,j \]
```

$$b+\overbrace{c+d+\underbrace{e+f+g}_{y}+h+i}^{x}+j$$

事实上\overline命令也存在问题，请比较：

```
$\overline{A}\overline{B}$ \\
$\closure{A}\closure{B}$
$\closure{AB}$
```

$$\overline{A}\overline{B}$$
$$\overline{A}\,\overline{B}\ \overline{AB}$$

其中\closure是在导言区定义的：

```
\newcommand{\closure}[2][3]{{}\mkern#1mu
    \overline{\mkern-#1mu#2}}
```

还可以输出能堆叠到其他对象上的箭头符，比如向量符号：

```
  $\vec
    a\quad\overrightarrow{PQ}$
  $\overleftarrow{EF}$
```

$$\vec{a}\quad\overrightarrow{PQ}\ \overleftarrow{EF}$$

你也许还需要能够添加上下堆叠的箭头符：

```
\[ a\xleftarrow{x+y+z} b \]
\[ c\xrightarrow[x<y]{a*b*c}d \]
```

$$a\xleftarrow{x+y+z}b$$
$$c\xrightarrow[x<y]{a*b*c}d$$

尖帽符号、波浪符号，还有yhmath宏包支持的圆弧符号：

```
$\hat{A}\quad\widehat{AB}$\\
$\tilde{C}\quad\widetilde{CD}
\qquad\wideparen{APB}$
```

$$\hat{A}\quad\widehat{AB}$$
$$\tilde{C}\quad\widetilde{CD}\qquad\wideparen{APB}$$

强制堆叠命\stackrel，位于上方的符号与上标同等大小。如果有amsmath宏包，可以使用\overset或者\underset 命令，前者与\stackrel命令完全等同：

```
$\int f(x) \stackrel{?}{=} 1$\\
$A\overset{abc}{=}B$ \quad
    $C\underset{def}{=}D$
```

$$\int f(x)\stackrel{?}{=}1$$
$$A\overset{abc}{=}B\quad C\underset{def}{=}D$$

一个很强大的堆叠放置命令\sideset，只用于巨算符：

```
\[\sideset{_a^b}{_c^d}\sum\]
\[\sideset{}{'}\sum_{n=1}\text{或}
\,{\sum\limits_{n=1}}'\]
```

$$\sideset{_a^b}{_c^d}\sum$$
$$\sideset{}{'}\sum_{n=1}\text{或}\,{\sum\limits_{n=1}}'$$

去心邻域\mathring大概也属于堆叠符的一种？这样输出：

```
$\mathring{U}$
```

$\mathring{U}$

在下一次节：累加与累积中，还介绍了更多的堆叠命令。

4.3.4 累加与累积

使用\sum和\prod命令，效果如下：

```
\[\sum_{i=1}^{n}a_i=1 \qquad
\prod_{j=1}^{n}b_j=1\]
```

$$\sum_{i=1}^{n} a_i = 1 \qquad \prod_{j=1}^{n} b_j = 1$$

有时需要复杂的堆叠方式，效果如下：

```
\[\sum_{\substack{0<i<n \\
  0<j<m}} p_{ij}=
  \prod_{\begin{subarray}{l}
  i\in I \\  1<j<m
  \end{subarray}}q_{ij}\]
```

$$\sum_{\substack{0<i<n \\ 0<j<m}} p_{ij} = \prod_{\substack{i\in I \\ 1<j<m}} q_{ij}$$

有时候需要强制实现堆叠的效果，可以使用\limits命令。如果堆积目标不是数学对象，还需要使用\mathop命令：

```
\[\max\limits_{i>1}^{x}\quad
\mathop{xyz}\limits_{x>0}\quad
\lim\nolimits_{x\to \infty}\]
```

$$\max\limits_{i>1}^{x} \quad \mathop{xyz}\limits_{x>0} \quad \lim\nolimits_{x\to \infty}$$

4.3.5 矩阵与省略号

矩阵的排版可以通过array环境和自适应定界符完成：

```
\[\mathbf{A}=
\left(\begin{array}{ccc}
x_{11} & x_{12} & \ldots \\
x_{21} & x_{22} & \ldots \\
\vdots & \vdots & \ddots
\end{array}\right)\]
```

$$\mathbf{A} = \left(\begin{array}{ccc} x_{11} & x_{12} & \ldots \\ x_{21} & x_{22} & \ldots \\ \vdots & \vdots & \ddots \end{array}\right)$$

还有就是\cdots命令。mathdots宏包支持省略号缩放，并提供了\iddots：

$\cdots$ 或许什么时候需要使用呢？

LaTeX 原生支持自动添加定界符的形式，不过要放在数学环境内：

```
\centering $\begin{matrix}
0 & 1 \\ 1 & 0 \end{matrix}\qquad
\begin{pmatrix} 0 & 2 \\
2 & 0 \end{pmatrix}$
```

$$\begin{matrix} 0 & 1 \\ 1 & 0 \end{matrix} \qquad \begin{pmatrix} 0 & 2 \\ 2 & 0 \end{pmatrix}$$

方括号和花括号使用`\[Bb]matrix`命令：

```
\centering $\begin{bmatrix}
0 & 3 \\ 3 & 0
    \end{bmatrix}\qquad
\begin{Bmatrix} 0 & 4 \\
4 & 0 \end{Bmatrix}$
```

$$\begin{bmatrix} 0 & 3 \\ 3 & 0 \end{bmatrix} \qquad \begin{Bmatrix} 0 & 4 \\ 4 & 0 \end{Bmatrix}$$

行列式使用`\[Vv]matrix`命令：

```
\centering $\begin{vmatrix}
0 & 5 \\ 5 & 0
    \end{vmatrix}\qquad
\begin{Vmatrix} 0 & 6 \\
6 & 0 \end{Vmatrix}$
```

$$\begin{vmatrix} 0 & 5 \\ 5 & 0 \end{vmatrix} \qquad \begin{Vmatrix} 0 & 6 \\ 6 & 0 \end{Vmatrix}$$

宏包`mathtools`的带星 matrix 命令，可更改列对齐：

```
$\begin{pmatrix*}[r]
100 & -200 \\ 20 & 10
\end{pmatrix*}$
```

$$\begin{pmatrix*}[r] 100 & -200 \\ 20 & 10 \end{pmatrix*}$$

在矩阵中排版`\dfrac`分式时，处理行距如下例的 `\\[8pt]`：

```
\[\mathbf{H}=\begin{bmatrix}
\dfrac{\partial^2 f}{\partial
    x^2} &
\dfrac{\partial^2 f}
{\partial x \partial y} \\[8pt]
\dfrac{\partial^2 f}
{\partial x \partial y} &
\dfrac{\partial^2 f}{\partial
    y^2}
\end{bmatrix}\]
```

$$\mathbf{H}=\begin{bmatrix} \dfrac{\partial^2 f}{\partial x^2} & \dfrac{\partial^2 f}{\partial x \partial y} \\[8pt] \dfrac{\partial^2 f}{\partial x \partial y} & \dfrac{\partial^2 f}{\partial y^2} \end{bmatrix}$$

宏包`amsmath`还支持行内小矩阵`\smallmatrix`，需手动加括号。

```
矩阵 $\left(\begin{smallmatrix}
x & -y\\ y & x\end{smallmatrix}
\right)$ 可以显示在行内。
```

矩阵 $\left(\begin{smallmatrix} x & -y \\ y & x \end{smallmatrix}\right)$ 可以显示在行内。

最后，一种带边注的矩阵`\bordermatrix`，用法有些奇怪：

```
\[\bordermatrix{& 1 & 2\cr
1 & A & B \cr
2 & C & D \cr} \]
```

$$\bordermatrix{& 1 & 2\cr 1 & A & B \cr 2 & C & D \cr}$$

4.3.6 分段函数与联立方程

用cases环境书写分段函数，它自动生成一个比\left{更紧凑的花括号：

```
\[y=\begin{cases}
\int x, & x>0 \\
0,   & x=0 \\
x-1, & x<0
\end{cases},\,x\in\mathbb{R}\]
```

$$y=\begin{cases}\int x, & x>0\\ 0, & x=0\\ x-1, & x<0\end{cases},\,x\in\mathbb{R}$$

如果想要生成 display 样式的内容（比如上面的积分号只是 text 样式的），使用mathtools宏包的dcases环境代替cases环境。如果cases环境的第二列条件不是数学语言而是一般文字，可以考虑使用dcases*环境，列中用 & 隔开。

```
\[y=\begin{dcases}
  \int x, & x>0 \\
  x^2, & x\leqslant 0
  \end{dcases}\]
\[z=\begin{dcases*}
  y, & when $y$ is prime\\
  y^2, & otherwise
  \end{dcases*}\]
```

$$y=\begin{dcases}\int x, & x>0\\ x^2, & x\leqslant 0\end{dcases}$$

$$z=\begin{dcases*}y, & when y is prime\\ y^2, & otherwise\end{dcases*}$$

4.3.7 多行公式及其编号

多行公式可以使用amsmath下的align环境——因为原生的eqnarray环境真的很差！而且align环境不需要像array环境那样给出列的数目和参数，能够根据 & 符号的数量来自调整。**这个环境会自动对齐等号或者不等号，所以必要时请用 & 指定对齐位置。**下面是一个例子：

```
\begin{align}
  a^2  &= a\cdot a \\
       &= a*a      \\
       &= a^2
\end{align}
```

$$\begin{aligned}a^2 &= a\cdot a &(4.2)\\ &= a*a &(4.3)\\ &= a^2 &(4.4)\end{aligned}$$

LaTeX 中长公式不能自动换行[5]，请如上自行指定断行位置和缩进距离。

至于多行公式换页，可以在导言区加上\allowdisplaybreaks实现（可选参数：1 为尽量避免换页，2 至 4 为倾向于换页），或在特定位置加上\displaybreak（可选参数：0 为允许在下个换行符后换页，但不倾向换页；2-3 介中；4 为强制换页）。两种的默认可选参数都是 4。

上例给出三个编号，如果你只需要一个，可以：

[5]不过breqn宏包的dmath环境可以实现自动换行，读者可以自行尝试效果。

```
\begin{align}
  a^2&= a\cdot a& b&=c\nonumber\\
  g  &= a*a & d&>e>f  \nonumber\\
  step&= a^2 & &Z^3
\end{align}
```

$$a^2 = a \cdot a \qquad b = c$$
$$g = a * a \qquad d > e > f$$
$$step = a^2 \qquad Z^3 \tag{4.5}$$

如果你想让编号显示在这三行的中间而不是最下面一行，可以尝试把公式写在aligned或者gathered环境中，然后再嵌套到equation环境内。如果你根本不想给多行公式编号，尝试align*环境。

另外，amsmath宏包的multline环境将自动把编号放在末行。首行左对齐，末行右对齐，中间的行居中。

```
\begin{multline}
a>b \\
b>c \\
\therefore a>c
\end{multline}
```

$$a > b$$
$$b > c$$
$$\therefore a > c \tag{4.6}$$

如果想在环境中插入小段行间文字,使用\intertext命令,或者mathtools宏包的\shortintertext命令。区别是后者的垂直间距更小一些。

```
\begin{align*}
\shortintertext{If}
 y &= 0 \\
 x &< 0\\
\shortintertext{then}
 z &= x+y
\end{align*}
```

If
$$y = 0$$
$$x < 0$$
then
$$z = x + y$$

当然，align环境用于分列对齐的。如果仅想所有行居中，使用amsmath宏包的gather环境即可。这是一个非常实用的环境，你也可以用gather*环境排版居中的、非编号的多行公式。

```
\begin{gather}
  X=1+2+\cdots+n \\
  Y=1
\end{gather}
```

$$X = 1 + 2 + \cdots + n \tag{4.7}$$
$$Y = 1 \tag{4.8}$$

4.3.8 二项式

二项式可能需要借助amsmath宏包的\binom命令。它也有像分式一样的行间和行内两个命令\tbinom与\dbinom：

```
$\mathrm{C}_n^k=\binom{n}{k}
\qquad a_n=\dbinom{n}{k}$
```

$\mathrm{C}_n^k = \binom{n}{k} \qquad a_n = \dbinom{n}{k}$

你也可以通过该宏包支持的`\genfrac`自定义类似二项式命令：

```
\genfrac{left-delim}{right-delim}{thickness}{mathstyle}
{numerator}{denominator}
% thickness为分式线线宽，留空空表示默认
% mathstyle从0-3由\displaystyle减至\scriptscriptstyle
\newcommand{\Bfrac}[2]{\genfrac{[}{]}{0pt}{}{#1}{#2}}
```

你可以借此得到新的命令`\Bfrac`：

```
\[\text{We define
    }\Bfrac{n}{k}=\binom{k}{n}\]
```

$$\text{We define } \begin{bmatrix} n \\ k \end{bmatrix} = \binom{k}{n}$$

4.3.9 定理

在使用下述定理内容时，请加载`amsthm`宏包。

首先是定理环境格式的自定义。如同定义命令一样，在导言区加上：

```
\newtheorem{envname}[counter]{text}[section]
```

其中 *name* 表示定理的引用名称，即下文将其作为一个环境名来识别；*text* 表示定理的显示名称，即下文中定理将以其作为打印内容。而 *counter* 参数表示你是否与先前声明的某定理共同编号。*section* 参数表示定理的计数层级，如果是 section，表示每节分别计数；chapter 表示每章分别计数。

来看一个例子。首先在导言区定义如下三个样式：

```
\theoremstyle{definition}\newtheorem{laws}{Law}[section]
\theoremstyle{plain}\newtheorem{ju}[laws]{Jury}
\theoremstyle{remark}\newtheorem*{marg}{Margaret}
```

以上三个`\theoremstyle`即是它预定义的所有样式类型。definition 标题粗体，内容罗马体；plain 标题粗体，内容斜体；remark 标题斜体，内容罗马体。带星号表示不进行计数。在环境的使用中可以添加可选参数，用于以括号的形式注释定理。然后这是示例：

```
\begin{laws}
Never believe easily.
\end{laws}
\begin{ju}[The 2nd]
Never suspect too much.
\end{ju}
\begin{marg}Nothing
    else.\end{marg}
```

Law 4.3.1. Never believe easily.

Jury 4.3.2 (The 2nd)**.** *Never suspect too much.*

Margaret. Nothing else.

`amsthm`宏包还提供了`proof`环境，并且用`\qedhere`来指定证毕符号的位置。如果不加指定，将会自动另起一行。

```
\begin{proof}
For an right triangle, we have:
  \[a^2+b^2=c^2 \qedhere\]
\end{proof}
```

证明. For an right triangle, we have:

$$a^2+b^2=c^2 \qquad \square$$

4.4 数学符号与字体

4.4.1 数学字体

原生的数学字体命令：

表 4.1: 原生数学字体表

`\mathrm{ABCDabcde 1234}`	ABCDabcde1234
`\mathit{ABCDabcde 1234}`	*ABCDabcde123*
`\mathnormal{ABCDabcde 1234}`	*ABCDabcde*1234
`\mathcal{ABCDabcde 1234}`	$\mathcal{ABCD}\dashv\lfloor\rfloor\lceil\rceil\infty\in\ni\triangle$

需要其他宏包支持的数学字体：

表 4.2: 宏包数学字体表

`\mathscr{ABCDabcde 1234}` $\mathscr{ABCD}$	mathrsfs
`\mathfrak{ABCDabcde 1234}` $\mathfrak{ABCDabcde}1234$	amsfonts 或者 amssymb
`\mathbb{ABCDabcde 1234}` $\mathbb{ABCD}\beth\nvDash\nVdash\nvdash$	amsfonts 或者 amssymb

4.4.2 定界符

表4.3给出了一些数学环境中使用的定界符。
使用`\left`, `\right`还有`\middle`能够使定界符自适应式子的高度：

```
\[P\left(X \middle\vert
   Y=0\right)
=\left.\int_0^1 p(t)\ud
   t\middle/ N\right.\]
```

$$P\left(X\middle|Y=0\right)=\left.\int_0^1 p(t)\,\mathrm{d}t\middle/N\right.$$

如果希望手动指定定界符的尺寸，这时使用后：

```
% 加l, r, m对应上述三种自适应命令
$(\big(\Big(\bigg(\Bigg<\qquad
\bigl[\frac{x+y}{x^2}\bigr]$
```

$$(\big(\Big(\bigg(\Bigg\langle \qquad \bigl[\tfrac{x+y}{x^2}\bigr]$$

表 4.3: 定界符

(	(	[	[or \lbrack	↑	\uparrow
)	)	]	] or \rbrack	↓	\downarrow
{	{ or \lbrace	}	} or \rbrace	↕	\updownarrow
⟨	\langle	⟩	\rangle	\	\backslash
⌊	\lfloor	⌋	\rfloor	⇕	\Updownarrow
⌈	\lceil	⌉	\rceil	⇑	\Uparrow
‖	| or \Vert	|	| or \vert	⇓	\Downarrow

– 以下需要 amssymb 宏包 –

⌜	\ulcorner	⌝	\urcorner
⌞	\llcorner	⌟	\lrcorner

有时\left.和\right.能灵活地用于跨行控制，因为它们并非实际配对：

```
\begin{align*}
  x &=\left(\frac{1}{2}x\right.\\
  &\left.\vphantom{\frac{1}{2}}
  +y^2+z_1\right)
\end{align*}
```

$$x = \left(\frac{1}{2}x \right. \\ \left. \vphantom{\frac{1}{2}} + y^2 + z_1\right)$$

其中\vphantom命令用于输出一个高度虚位，使得第二行的自适应定界符与第一行同等大小。特别地，命令\mathstrut表示一个有圆括号总高的虚位：

```
$\sqrt{b}\sqrt{y}\qquad
\sqrt{\mathstrut
    b}\sqrt{\mathstrut y}$
```

$\sqrt{b}\sqrt{y}$ $\qquad$ $\sqrt{\mathstrut b}\sqrt{\mathstrut y}$

4.4.3 希腊字母

希腊字母表如表4.4所示。表中包含了小写希腊字母、大写希腊字母，其中部分希腊字母的输入方式与英文字母一致。

4.4.4 二元运算符

二元运算符包括常见的加减乘除，还有集合的交、并、补等运算。表4.5只列出常用的二元运算符，更多的请参考 symbols-a4 文档。

4.4.5 二元关系符

二元关系符常常被用于判断两个数的大小关系，或者集合中的从属关系。表4.6和表4.7只列出常用的二元关系符，更多的请参考 symbols-a4 文档。

表4.7中的二元关系符需要amssymb宏包。

表 4.4: 希腊字母表

α	\alpha	θ	\theta	o	o	υ	\upsilon
β	\beta	ϑ	\vartheta	π	\pi	ϕ	\phi
γ	\gamma	ι	\iota	ϖ	\varpi	φ	\varphi
δ	\delta	κ	\kappa	ρ	\rho	χ	\chi
ϵ	\epsilon	λ	\lambda	ϱ	\varrho	ψ	\psi
ε	\varepsilon	μ	\mu	σ	\sigma	ω	\omega
ζ	\zeta	ν	\nu	ς	\varsigma	η	\eta
ξ	\xi	τ	\tau				
A	A	B	B	Γ	\Gamma	$\varGamma$	\varGamma
Δ	\Delta	$\varDelta$	\varDelta	E	E	Z	Z
H	H	Θ	\Theta	$\varTheta$	\varTheta	I	I
Λ	\Lambda	$\varLambda$	\varLambda	M	M	N	N
Ξ	\Xi	$\varXi$	\varXi	O	O	Π	\Pi
$\varPi$	\varPi	P	P	Σ	\Sigma	$\varSigma$	\varSigma
T	T	Υ	\Upsilon	$\varUpsilon$	\varUpsilon	Φ	\Phi
$\varPhi$	\varPhi	X	X	Ψ	\Psi	$\varPsi$	\varPsi
Ω	\Omega	$\varOmega$	\varOmega				

表 4.5: 二元运算符: \mathbin

$+$	+	$-$	-	$\times$	\times	$\div$	\div
$\pm$	\pm	$\mp$	\mp	$\circ$	\circ	$\triangleright$	\triangleright
$\cdot$	\cdot	$\star$	\star	$*$	\ast	$\triangleleft$	\triangleleft
$\cup$	\cup	$\cap$	\cap	$\setminus$	\setminus	$\bullet$	\bullet
$\oplus$	\oplus	$\ominus$	\ominus	$\otimes$	\otimes	$\oslash$	\oslash
$\odot$	\odot	$\bigcirc$	\bigcirc	$\vee$	\vee,\lor	$\wedge$	\wedge,\land
$\bigcup$	\bigcup	$\bigcap$	\bigcap	$\bigvee$	\bigvee	$\bigwedge$	\bigwedge

表 4.6: 二元关系符: \mathrel

$<$	<	$>$	>	$\leq$	\le(q)	$\geq$	\ge(q)
$\ll$	\ll	$\gg$	\gg	$\equiv$	\equiv	$\neq$	\neq
$\prec$	\prec	$\succ$	\succ	$\preceq$	\preceq	$\succeq$	\succeq
$\sim$	\sim	$\simeq$	\simeq	$\cong$	\cong	$\approx$	\approx
$\subset$	\subset	$\supset$	\supset	$\subseteq$	\subseteq	$\supseteq$	\supseteq
$\in$	\in	$\ni$	\ni	$\notin$	\notin	$\propto$	\propto
$\parallel$	\parallel	$\perp$	\perp	$\smile$	\smile	$\frown$	\frown
$\asymp$	\asymp	$\bowtie$	\bowtie	$\vdash$	\vdash	$\dashv$	\dashv

4.4.6 箭头与长等号

在表4.3中给出了几个箭头符号，但是不够全，这里给出总表如表4.8。

LaTeX 定义了逻辑命令\iff, \implies, \impliedby，与箭头符大小相同但

表 4.7: amssymb 二元关系符

$\leqslant$	\leqslant	$\geqslant$	\geqslant	$\because$	\because	$\therefore$	\therefore
$\nless$	\nless	$\ngtr$	\ngtr	$\lessdot$	\lessdot	$\gtrdot$	\gtrdot
$\lessgtr$	\lessgtr	$\gtrless$	\gtrless	$\lesseqgtr$	\lesseqgtr	$\gtreqless$	\gtreqless
$\subseteqq$	\subseteqq	$\supseteqq$	\supseteqq	$\subsetneqq$	\subsetneqq	$\supsetneqq$	\supsetneqq

表 4.8: 箭头

$\leftarrow$	\leftarrow	$\longleftarrow$	\longleftarrow
$\rightarrow$	\rightarrow	$\longrightarrow$	\longrightarrow
$\leftrightarrow$	\leftrightarrow	$\longleftrightarrow$	\longleftrightarrow
$\Leftarrow$	\Leftarrow	$\Longleftarrow$	\Longleftarrow
$\Rightarrow$	\Rightarrow	$\Longrightarrow$	\Longrightarrow
$\Leftrightarrow$	\Leftrightarrow	$\Longleftrightarrow$	\Longleftrightarrow
$\mapsto$	\mapsto	$\longmapsto$	\longmapsto
$\nearrow$	\nearrow	$\searrow$	\searrow
$\swarrow$	\swarrow	$\nwarrow$	\nwarrow
$\leftharpoonup$	\leftharpoonup	$\rightharpoonup$	\rightharpoonup
$\leftharpoondown$	\leftharpoondown	$\rightharpoondown$	\rightharpoondown
$\rightleftharpoons$	\rightleftharpoons	$\iff$	\iff (bigger space)

是两侧间距更大:

```
$x=y \implies a=b$\\
$x=y \impliedby a=b$\\
$x=y \iff a=b$
```

$$x=y \implies a=b$$
$$x=y \impliedby a=b$$
$$x=y \iff a=b$$

依旧另外给出一个基于`amssymb`宏包的附表4.9。

表 4.9: amssymb 箭头

$\dashleftarrow$	\dashleftarrow	$\dashrightarrow$	\dashrightarrow
$\circlearrowleft$	\circlearrowleft	$\circlearrowright$	\circlearrowright
$\leftrightarrows$	\leftrightarrows	$\rightleftarrows$	\leftrightarrows
$\nleftarrow$	\nleftarrow	$\nLeftarrow$	\nLeftarrow
$\nrightarrow$	\nrightarrow	$\nRightarrow$	\nRightarrow
$\nleftrightarrow$	\nleftrightarrow	$\nLeftrightarrow$	\nLeftrightarrow

最后，宏包`extarrows`给出了一些实用的长箭头与长等符号:

```
$\xlongequal{\Delta}$\quad
$\xLeftrightarrow{\Delta}$\\
$\xleftrightarrow{x=\tan t}$\\
$\xLongleftarrow{x}
    \xLongrightarrow{y}$
```

$$\xlongequal{\Delta}\quad \xLeftrightarrow{\Delta}$$
$$\xleftrightarrow{x=\tan t}$$
$$\xLongleftarrow{x} \xLongrightarrow{y}$$

4.4.7 其他符号

注意冒号如果从键盘输入，会识别为关系符，例如 :=。在表示比例时也可以借用，或者外加\mathbin命令 $a:b$。数学中可能用到的冒号，请使用\colon命令，像 $x:y\to\infty$ 这样。

像\一样在数学环境中使用*命令，提醒 LaTeX 此处可以断词。LaTeX 如果在此处断词，会自动补一个 × 符号。你也可以自定义：

```
1 \renewcommand{\*}{discretionaty{\,\mbox{$\cdot$}}{}{}}
```

最后是一些其他的难以归类的符号，也不全是数学领域会用到的，只不过它们可以在数学环境下输出出来，以及被amssymb宏包所支持。如表4.10和表4.11。

表 4.10: 其他符号

…	\dots	⋯	\cdots	⋮	\vdots	⋱	\ddots
∀	\forall	∃	\exists	ℜ	\Re	ℵ	\aleph
∠	\angle	∞	\infty	△	\triangle	∇	\nabla
ℏ	\hbar	𝚤	\imath	𝚥	\jmath	ℓ	\ell
♠	\spadesuit	♡	\heartsuit	♣	\clubsuit	♢	\diamondsuit
♭	\flat	♮	\natural	♯	\sharp		
非数学符号:							
£	\pounds	§	\S	©	\copyright	¶	\P
†	\dag	‡	\ddag	®	\textregistered		

表 4.11: amssymb 其他符号

□	\square	■	\blacksquare	ℏ	\hslash
★	\bigstar	▲	\blacktriangle	▼	\blacktriangledown
◊	\lozenge	⧫	\blacklozenge	∡	\measuredangle
℧	\mho	∅	\varnothing	ð	\eth

LaTeX 进阶 5

本章的内容多数与宏包的使用相关。记得使用 texdoc 命令查看宏包的使用手册，这是学习宏包最好的手段，没有之一。

5.1 自定义命令与环境

自定义命令是 LaTeX 相比于字处理软件 MS Word 之流最强大的功能之一。它可以大幅度优化你的文档体积，用法是：

```
\newcommand{cmd}[args][default]{def}
```

现在来解释一下各个参数：

cmd：新定义的命令，不能与现有命令重名。

args：参数个数。

default：首个参数，即 #1 的默认值。你可以定义只有一个参数、且参数含默认值的命令。

def：具体的定义内容。参数 1 以 #1 代替，参数 2 以 #2 代替，以此类推。

如果重定义一个现有命令，使用\renewcommand命令，用法与\newcommand一致。简单的例子：

```
% 加粗：\concept{text}
\newcommand{\concept}[1]{\textbf{#1}}
% 加粗#2并把#1#2加入索引，默认#1为空。
% 比如\cop{Sys}或者\cop[Sec.]{Sys}
\newcommand{\cop}[2][]{\textbf{#2}}\index{#1 #2}}
```

如果想定义一个用于数学环境的命令，借助\ensuremath命令。它保证其参数会在数学模式下运转，且即使已位于数学模式中也不会报错。

```
\renewcommand\qedsymbol{\ensuremath{\Box}}
```

自定义环境的命令是\newenvironment，也可以定义多个参数。注意后段定义中不能使用参数，但你可以“先保存后调用”。例子：

```
\newenvironment{QuoteEnv}[2][]
    {\newcommand\Qauthor{#1}\newcommand\Qref{#2}}
    {\medskip\begin{flushright}\small ——~\Qauthor\\
    \emph{\Qref}\end{flushright}}
```

下面是效果：

```
\begin{QuoteEnv}[William
    Butler]{When you are old}
But one man loved the pilgrim
    soul in you.
And loved the sorrows of your
    changing face.
\end{QuoteEnv}
```

But one man loved the pilgrim soul in you. And loved the sorrows of your changing face.

—— William Butler
When you are old

5.2 箱子：排版的基础

LaTeX 排版的基础单位就是“箱子 (box)”，例如整个页面是一个矩形的箱子，侧边栏、主正文区、页眉页脚也都是箱子。在正常排版中，文字应当位于箱子内部；如果单行文字过长、没能正确断行，造成文字超出箱子，这便是 Overfull 的坏箱 (bad box)；如果内容太少，导致文字不能美观地填满箱子，便是 Underfull 的坏箱。

如图5.1所示，箱子的三个参数：高度 (height)、宽度 (width) 和深度 (depth)。分隔高度和深度的是基线。

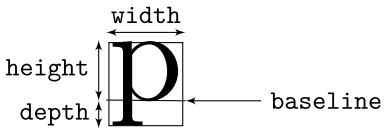


图 5.1: 箱子的参数

5.2.1 无框箱子

命令\mbox产生一个无框的箱子，宽度自适应。有时用它来强制“结合”一系列命令，使之不在中间断行。比如 TeX 这个命令的定义（其中\raisebox命令在后面介绍）：

```
\mbox{T\hspace{-0.1667em}\raisebox{-0.5ex}{E}\hspace{-0.125em}X}
```

或者也可以使用命令\makebox[width][pos]{text}，宽度由 width 参数指定。pos 参数的取值可以是 l, s, r 即居左、两端对齐、居右，还有竖直方向的 t, b 两个参数。

无框小页的使用方法是 minipage 环境，参数类似\parbox：

```
\begin{minipage}[pos]{width}
```

5.2.2 加框箱子

命令\fbox产生加框的箱子，宽度自动调整，但不能跨行。命令\framebox类似上面介绍的\makebox。如果是想在数学环境下完成加框，使用\boxed命令。

width 参数中，可以用\width, \height, \depth, \totalheight分别表示箱子的自然宽度、自然高度、自然深度和自然高深度之和。

```
\fbox{This is a frame box} \\
\framebox[2\width]{double-width}\\
\begin{equation}\boxed{x^2=4}
\end{equation}
```

This is a frame box

double-width

$$\boxed{x^2 = 4} \tag{5.1}$$

加宽盒子的宽度、以及内容到盒子的距离可以自行定义。默认定义是：

```
\setlength{\fboxrule}{0.4pt} \setlength{\fboxsep}{3pt}
```

加框小页使用boxedminipage环境 (需要boxedminipage宏包)。

5.2.3 竖直升降的箱子

命令\raisebox可以把文字提升或降低，它有两个参数：

```
A\raisebox{-0.5ex}{n} example.
```

A_n example.

5.2.4 段落箱子

段落箱子的强大之处在于它提供自动换行的功能，当然你需要指定宽度。

```
\parbox[pos]{width}{text}
```

以及例子：

```
This is \parbox[t]{3.5em}{an long
example to show} how \parbox[b]
{4em}{parbox' works perfectly.
```

This is an long example to show how 'parbox' works perfectly .

5.2.5 缩放箱子

宏包graphicx提供了一种可缩放的箱子`\scalebox{h-sc}[v-sc]{pbj}`，注意其中水平缩放因子是必要参数。缩放内容可以是文字也可以是图片，例子：

```
\LaTeX---\scalebox{-1}[1]{\LaTeX}\\
\LaTeX---\scalebox{1}[-1]{\LaTeX}\\
\LaTeX---\scalebox{-1}{\LaTeX}\\
\LaTeX---\scalebox{2}[1]{\LaTeX}
```

LaTeX—XeTeX
LaTeX—pdfLaTeX
LaTeX—XeLaTeX
LaTeX—LaTeX

此外还有`\resizebox{width}{heigh}{text}`命令。

5.2.6 标尺箱子

命令`\rule[lift]{width}{height}`能够画出一个黑色的矩形。你可以在单元格中使用 width, height 其一为 0 的该命令，作一个隐形的“支撑”来限定单元格的宽或高。而`\strut`命令则用当前字号大小设置高度与深度。例如：

```
\begin{tabular}{|c|}
  \hline
  \rule[-1em]{1em}{1ex}text
  \rule{0pt}{38pt} \\
  \hline
  2nd text\strut--- \\
  \hline
\end{tabular}
```

text
2nd text—

5.2.7 覆盖箱子

有时候需要把一段文字覆盖到另一段上面，使用`\llap`或`\rlap`。什么？你从没这么干过？但或许有一天你需要呢？

```
你看不清这些字\llap{是什么}\\
\rlap{这些}你也看不清
```

你看不清是什么
这些你也看不清

5.2.8 旋转箱子

宏包graphicx提供了\rotatebox命令，参数与插图命令相同。

```
\rotatebox[origin=c]{90}{专}治颈椎病。
```

专治颈椎病。

5.2.9 颜色箱子

xcolor宏包支持的颜色箱子命令有：

```
\textcolor{red}{红色}强调\\
\colorbox[gray]{0.95}{浅灰色背景}
    \\
\fcolorbox{blue}{cyan}{%
\textcolor{blue}{蓝色边框+文字，
  青色背景}}
```

红色强调
浅灰色背景
蓝色边框 + 文字，青色背景

命令\fcolorbox可以调整\fboxrule，\fboxsep参数，而\colorbox只能调整后者。参考前面的加框箱子一节。

强大的tcolorbox宏包专门定义了众多的箱子命令，参考第5.13.2节。

5.3 复杂距离

5.3.1 水平和竖直距离

长度单位参考这里介绍过的内容。水平距离命令有两种，一种禁止在此处断行，如表5.1；另一种允许换行，如表5.2。

表 5.1: 禁止换行的水平距离

\thinspace或\,	0.1667em	a b
\negthinspace或\!	-0.1667em	ab
\enspace	0.5em	a b
\nobreakspace或 ~	空格	a b

使用\hspace{length}命令自定义空格的长度，其中 *length* 的取值例如：-1em, 2ex, 5pt plus 3pt minus 1pt, 0.5\linewidth 等。如果想要这个命令在断行处也正常输出空格，使用带星命令\hspace*。

类似地使用\vspace和\vspace*命令，作为竖直距离的输出。

表 5.2: 允许换行的水平距离

\quad	1em	_ _
\qquad	2em	_ _
\enskip	0.5em	_ _
\␣	空格	_ _

要定义新的长度宏，使用\newlength命令；要重设现有长度宏的值，可以选择使用\setlength命令；要调整长度宏的值，则使用\addtolength命令。

```
\newlength{\mylatexlength}
\setlength{\mylatexlength}{10pt}
\addtolength{\mylatexlength}{-5pt}
```

此外，LaTeX 还定义了三个竖直长度\smallskip, \medskip, 和\bigskip:

```
\parbox[t]{3em}{TeX\par TeX}
\parbox[t]{3em}{TeX\par\smallskip
    TeX}
\parbox[t]{3em}{TeX\par\medskip
    TeX}
\parbox[t]{3em}{TeX\par\bigskip
    TeX}
```

TeX TeX TeX TeX
TeX TeX TeX TeX

5.3.2 填充距离与弹性距离

命令\fill用于填充距离，需要作为\hspace或\vspace的参数使用。另外还有单独使用的命令\hfill与\vfill，作用相同。

弹性距离指以一定比例计算得到的多个空白，命令是\stretch。例子:

```
Left\hspace{\fill}Right\\
Left\hspace{\stretch{1}}Center
\hspace{\stretch{2}}Right
```

Left Right
Left Center Right

你还可以使用类似\hfill的\hrulefill和\dotfill命令:

```
L\hfill R\\
L\hrulefill Mid\dotfill R
```

L R
L_____Mid . . . . . R

5.3.3 行距

LaTeX 的行距由基线计算，可以使用命令\linespread{num}，默认的基线距离\baselineskip是 1.2 倍的文字高。所以默认行距是 1.2 倍；如果更改 linespread 为 1.3，那么行距变为 $1.2 \times 1.3 = 1.56$ 倍——这也是 ctex 文档类的做法。

此外还有\lineskiplimit和\lineskip命令。有时候在两行之间，可能包含较高的内容（比如分式 $\frac{1}{2}$），使得前一行底部与后一行顶部的距离小于 limit 值，则此时行距会从由\linespread改为由\lineskip控制。本手册采用：

```
\setlength{\lineskiplimit}{3pt}
\setlength{\lineskip}{3pt}
```

5.3.4 制表位 *

制表位使用tabbing环境，需要指出，这是一个极其容易造成坏箱的环境。几个要点：

\= 在此处插入制表位。
\> 跳入下一个制表位。
\\ 制表环境内必须手动换行和缩进。
\kill 若行末用\kill 代替\\，那么该行并不会被实际输出到文档中。

一个丑陋的例子：

```
\begin{tabbing}
\hspace{4em}\=\hspace{8em}\=\kill
制表位 \> 就是这样 \> 使用的 \\
随时 \> 可以添加 \> 新的: \= 就这样 \\
也可以 \= 随时重设 \= 制表位 \\
这是 \> 新的 \> 一行
\end{tabbing}
```

制表位　就是这样　　使用的
随时　　可以添加　　新的: 就这样
也可以 随时重设 制表位
这是　新的　　一行

5.3.5 悬挂缩进 *

这种缩进在实际排版中并不常用，经常是列表需要的场合才使用，但那可以借助列表宏包enumitem进行定义。这里介绍的是正文中的悬挂缩进使用。

如果需要对单独一段进行悬挂缩进，例如使用：

```
\hangafter 2
\hangindent 6em
```

这两行放在某一段的上方，作用是控制紧随其后的段落从第 2 行开始悬挂缩进，并且设置悬挂缩进的长度是 6em。

如果需要对连续的多段进行悬挂缩进，可以改造编号列表环境或者verse环境[1]来实现。或者尝试：

[1] 事实上这是一个排版诗歌的环境，参考前文的这里。

```
正文...

{\leftskip=3em\parindent=-1em
\indent
    这是第一段。注意整体需要放在
一组花括号内，且花括号前应当有空白行
。第一段前需要加indent命令，最后一段
的末尾需额外空一行，否则可能出现异常。

这是第二段。

\ldots

这是最后一段。别忘了空行。

}
```

正文...

这是第一段。注意整体需要放在一组花括号内，且花括号前应当有空白行。第一段前需要加 indent 命令，最后一段的末尾需额外空一行，否则可能出现异常。

这是第二段。

...

这是最后一段。别忘了空行。

5.3.6 整段缩进 *

宏包changepage提供了一个adjustwidth环境，它能够控制段落两侧到文本区（而不是页边）两侧的距离。

```
\begin{adjustwidth}{1cm}{3cm}
本段首行缩进需要额外手工输入。本环境距文本区左侧1cm，右侧3cm。
\end{adjustwidth}
```

也可以尝试赋值\leftskip等命令，对奇偶页处理更有效。

5.4 自定义章节样式

这一节主要涉及titlesec宏包的使用。章节样式调整使用\titlelabel，\titleformat*命令。前者需要配合计数器使用，后者简单地设置章节标题的字体样式。例如：

```
\titlelabel{\thetitle.\quad}
\titleformat*{\section}{\itshape}
```

章节样式由标签和标题文字两部分构成。标签一般表明了大纲级别以及编号，比如“第一章”、“Section 3.1”等。标题文字比如“自定义章节样式”这几个字。还记得吗？在 report 与 book 类的 subsection 及以下，article 类的 paragraph 及以下是默认没有编号的。因此对应的级别也没有标签，除非人工进行设置。

对于需要详细处理标签、标题文字两部分的情况，titlesec宏包还提供了一个\titleformat命令。调用方式：

```
\titleformat{command}[shape]{format}{label}{sep}
    {before-code}{after-code}
```

它们对应的含义如下：

command: 大纲级别命令，如`\chapter`等。

shape: 章节的预定义样式，分为 9 种：

- **hang** 缺省值。标题在右侧，紧跟在标签后。
- **block** 标题和标签封装排版，不允许额外的格式控制。
- **display** 标题另起一段，位于标签的下方。
- **runin** 标题与标签同行，且正文从标题右侧开始。
- **leftmargin** 标题和标签分段，且位于左页边。
- **rightmargin** 仿上。右页边。
- **drop** 文本包围标题。
- **wrap** 类似 drop，文本会自动调整以适应最长的一行。
- **frame** 类似 display，但有框线。

format: 用于设置标签和标题文字的字体样式。这里可以包含竖直空距，即标题文字到正文的距离。

label: 用于设置标签的样式，比如“第`\chinese\thechapter`章”大概是 ctexbook 类的默认样式设置。

sep: 标签和标题文字的水平间距，必须是 LaTeX 的长度表达。当 shape 取 display 时，表示竖直空距；取 frame 时表示标题到文本框的距离。

before: 标题前的内容。

after: 标题后的内容。对于 hang, block, display，此内容取竖向；对于 runin, leftmargin, 此内容取横向；否则此内容被忽略。

宏包还给出了`\titlespacing`与`\titlespacing*`两个命令。使用方式是：

```
\titlespacing*{command}{left}{before-sep}{after-sep}[right-sep]
\titlespacing{command}{left}{*m}{*n}[right-sep]
```

各参数的含义：

command: 大纲级别命令，如`\chapter`

label: 缩进值。在 left/right margin 下表示标题宽；在 wrap 中表示最大宽；在 runin 中表示标题前缩进的空距。

before-sep: 标题前的垂直空距。

after-sep: 标题与正文之间的空距。hang, block, display 中是垂直空距；runin, wrap, drop, left/right margin 中是水平空距。

right-sep: 可选。仅对 hang, block, display 适用。

***m/*n:** 在`\titlespacing`命令中的 m, n 分别表示 before 与 after sep 的变动范围倍数，基数是默认值。

最后，宏包还给出了`\titleline`命令，用来绘制填充整行、同时又嵌有其他对象的行。对象可以嵌入到左中右 lcr 三个位置。如果你只是想填充一行而不嵌入对象，使用`\titlerule`及其带星号的命令形式。

```
% 嵌入对象的线
\titleline[c]{CHAPTER 1}
% 单纯填充一行
\titlerule[height]
\titlerule*[width]{text}
```

最后，给出本手册中的样式定义，作为例子。这个例子稍微有些复杂，只使用到了`\titleformat`相关的命令。

```
\newcommand{\chaformat}[1]{%
    \parbox[b]{.5\textwidth}{\hfill\bfseries #1}%
    \quad\rule[-12pt]{2pt}{70pt}\quad
    {\fontsize{60}{60}\selectfont\thechapter}}
% chapter样式定义中的\chaformat以章名作为隐式参数
\titleformat{\chapter}[block]{\hfill\LARGE\sffamily}
    {}{0pt}{\chaformat}[\vspace{2.5pc}\normalsize
    \startcontents\printcontents{}{1}
    {\setcounter{tocdepth}{2}}]
\titleformat*{\section}{\centering\Large\bfseries}
\titleformat{\subsubsection}[hang]
    {\bfseries\large}{\rule{1.5ex}{1.5ex}}{0.5em}{}
```

本例没有定义 subsection 样式。如果你想给 subsection 级别标号（即赋予它标签），使用：`\setcounter{secnumdepth}{3}`[2]。

临时更改`\secnumdepth`可以生成不编号的章节，但章节名仍会被使用在目录和`\markboth`中——有时这比带星号的章节命令更巧妙一些。

5.5 自定义目录样式

这一节主要涉及titletoc宏包，它与titlesec宏包的文档写在同一个 pdf 中。上节的例子（即本手册 Chapter）涉及`\startcontents`与`\printcontents`命令，旨在每一章的开始插入本章的一个目录。

首先是目录的标题，可以通过 renewcommand 更改。分别是 `\contentsname`, `\listfigurename`, `\listtablename`三个。

再来看命令`\dottecontents`与命令`\titlecontents`：

```
\dottecontents{section}[left]{above-code}
    {label-width}{leader-width}
\titlecontents{section}[left]{above-code}{numbered-entry-format}
    {numberless-entry-format}{filler-page-format}[below-code]
```

各参数的含义：

section: 目录对象。可以填 chapter, section, 或者 figure, table.
left: 目录对象左侧到左页边区的距离。请作必选使用。
above-code: 格式调整命令。可以包含垂直对象，也可以用`\contentslabel`, 即指定本级别目录标签箱子的宽度。
label-width: 标签宽。
leader-width: 填充符号宽。默认的填充符号是圆点。
numered-entry-format: 如果有标签，则在目录文本前输入的格式。
numberless-entry-format: 如果没有标签输入的格式。
filler-page-format: 填充格式。一般借助titlesec中的`\titlerule*`命令。
below-code: 在 entry 之后输入的格式，比如垂直空距。

本手册目录样式定义，其中 section 级别使用了填充命令`\titlerule*`：

[2] report/book 类 part 级别深度为 0，递增；article 类 part 为-1，无 chapter 级别。故 section 及以下深度一致。

```
\titlecontents{chapter}[1.5em]{}{\contentslabel{1.5em}}
    {\hspace*{-2em}}{\hfill\contentspage}
\titlecontents{section}[3.3em]{}
    {\contentslabel{1.8em}}{\hspace*{-2.3em}}
    {\titlerule*[8pt]{$\cdot$}\contentspage}
\titlecontents*{subsection}[2.5em]{\small}
    {\thecontentslabel{}}{}
    {, \thecontentspage}[;\qquad][.]
```

5.6 自定义图表

5.6.1 长表格

包括supertabular, longtable, tabu在内的多个宏包都能完成长表格的排版，大致的功能会包括：

表头控制： 首页的表头样式，以及转页后表头的样式。

转页样式： 在表格跨页时，页面最下方插入的特殊行，比如 to be continued.

这里主要介绍longtable宏包。主要命令：

\endhead 定义每页顶端的表头。在表头行用该命令代替\\\命令来换行即可。

\endfirsthead 如果首页的表头与其他页不同，使用该命令。

\endfoot 定义每页底端的表尾。

\endlastfoot 另外定义末页底端的表尾。

\caption 与原生 tabular 的该命令一致。如果你不想显示表格编号，使用带星的该命令；如果不想让其加入表格目录，在可选参数中留空\caption[]{...}。

\label 注意\label 命令不能被用在多页对象中，请在表体中或者 firsthead-/lastfoot 中使用。

\LTleft 表格左侧到主文本区边缘的距离，默认是\fill。你可以用：\setlength\LTleft{0pt}来取消这个距离，进行居左。

\LTright 类似。

\LTpre 表格上部到文本的距离，默认是\bigskipamount[3]。

\LTpost 类似。

\\[…] 在换行后插入竖直空距。

***** 禁止在该行后立刻进行分页。

\kill 该行不显示，但用于计算宽度。

\footnote(mark/text) 命令\footnote不能用于表头或表尾；在表头和表尾中，使用\footnotemark命令，并在表外用\footnotetext写明脚注内容。

longtable宏包支持的表格可选参数是 clr，不能使用 t 或 b. 此外，longtable中的跨列可能需要编译多次才能正常显示。最后给出一个例子：

[3]这个命令通常是一行左右的竖直距离，12pt±4pt 左右。

表 5.3: This is an example

This is the headfirst[4]	
First Col	Second Col
This is an example	and you can
see how longtable will	work.
Space after line are	allowed.
You can adjust LTright and	LTleft
if you want to. I'd like	to set
LTleft as "0pt", but it all	depends on
you. And maybe you can try	footnote
like this[5] and also	footnotetext[6].
As for footnotemark, you've seen it	in the firsthead.
And I think maybe it's long	enough to make
a table across pages, so go to the	next page and
check whether the head at next page is	different from
that on this page. Also you can have a look	at lastfoot.
This is the	bottom.

[4]Footnotemark: first footnote in table head.
[5]Footnote example.
[6]Footnotetext example.

–Continued Longtable–

This is the head of other page		
*	First Here	Second Here *
*	So do you get how to use this	package? *
*	Maybe you'll love it. So enjoy	`longtable`! *
*	**That's all**	**and thanks**. *

它的代码如下：

```
\begin{longtable}{@{*}r||p{3cm}@{*}}
KILLED & LINE! \kill

\caption[\texttt{longtable} Example]{This is an example}\\
\hline
\multicolumn{2}{@{}c@{}}{This is the headfirst\footnotemark}\\
First Col & Second Col \\
\hline\hline
\endfirsthead

\caption*{--Continued Longtable--}\\
\hline\hline
\multicolumn{2}{@{}c@{}}{This is the head of other page}\\
First Here & Second Here \\
\hline
\endhead

\hline\hline
This is the & bottom. \\
\hline
\endfoot

\hline
\textbf{That's all} & \textbf{and thanks}. \\
\hline
\endlastfoot

\footnotetext{Footnotemark: first footnote in table head.}
This is an example & and you can \\
see how longtable will & work. \\
Space after line are & allowed. \\[25ex]
You can adjust LTright and & LTleft \\
if you want to. I'd like & to set \\
LTleft as 0pt'', but it all & depends on\\
you. And maybe you can try & footnote \\
like this\footnote{Footnote example.} and also
& footnotetext\footnotemark
\footnotetext{Footnotetext example.}. \\
As for footnotemark, you've seen it & in the firsthead.\\[15ex]
And I think maybe it's long & enough to make \\
a table across pages, so go to the & next page and \\
check whether the head at next page is & different from \\
```

```
that on this page. Also you can have a look & at lastfoot.
\\[20ex]
So do you get how to use this & package? \\
Maybe you'll love it. So enjoy & \texttt{longtable}!
\end{longtable}
```

5.6.2 booktabs：三线表

booktabs宏包提供\toprule, \midrule与\bottomrule命令来绘制三线表。更多需要的横线可以通过\midrule添加。

```
\begin{tabular}{cccc}
\toprule
& \multicolumn{3}{c}{Numbers} \\
\cmidrule{2-4}
& 1 & 2 & 3 \\
\midrule
Alphabet & A & B & C \\
Roman & I & II& III \\
\bottomrule
\end{tabular}
```

	Numbers		
	1	2	3
Alphabet	A	B	C
Roman	I	II	III

命令\cmidrule如果连续使用，还能写成\cmiderule(lr)的形式，使其向内缩进一小段，造成相互“断开”的样子。

5.6.3 彩色表格

彩色表格依靠colortbl宏包，它会调用array和color宏包。但是可以在加载xcolor宏包时添加 table 选项来调用colortbl宏包。

首先是命令\columncolor，给表格某列加背景色。其中 mode 参数是指 rgb/cmyk 等。left/right-ex 参数表示向两侧填充的距离，默认是\tablecolsep。

```
\columncolor[mode]{colorname}[left-ex][right-ex]
```

命令\rowcolor和\cellcolor分别用于更改表头行的颜色和单个单元格的颜色，放置在表格内对应位置即可。在xcolor支持下还可以使用\rowcolors命令，但放在表格开始之前：

```
% 表线为单横，从第2行开始，奇数行绿，偶数行青
\rowcolors[\hline]{2}{green}{cyan}
\begin{tabular}...
```

要临时开关奇偶行颜色，使用\show/hide rowcolors命令。

彩色表格中跨行，需要把跨行命令放在最后一行，并跨负数行：

```latex
\rowcolors{2}{green}{cyan}
\begin{tabular}{ll}
\hline Col 1 & Col 2\\
& A\\ \multirow{-2}*{Hey} & B\\
\hline
\end{tabular}
```

Col 1	Col 2
Hey	A
	B

5.6.4 子图表

子图表输出用subcaption宏包，它需要与caption宏包共同加载。比如：

```latex
\usepackage{caption,subcaption}
  \captionsetup[sub]{labelformat=simple}
  \renewcommand{\thesubtable}{(\alph{subtable})}
% 用\ref引用得到如“图1.1(a)”的效果
\begin{table}
\caption{Parents}
\begin{subtable}[b]{0.5\linewidth}
  \centering
  \begin{tabular}{|c|c|}
  A & B \\ \end{tabular}
  \caption{First}\label{...}
\end{subtable}
\begin{subtable}[b]{0.5\linewidth}
  \centering
  \begin{tabular}{|c|c|}
  A & B \\ C & D \end{tabular}
  \caption{Second}
\end{subtable}
\end{table}
```

效果如表5.4(a)与表5.4(b)。更多的请参考caption宏包。

表 5.4: Parents

A	B

(a) First

A	B
C	D

(b) Second

5.6.5 动态图

使用 animate 宏包（当然，graphicx 宏包也是需要的），可以将多张图片以动态图的形式插入 PDF。需要注意的是，**动态图在一些功能较弱的 PDF 浏览器中可能无法正常工作**，推荐使用 Adobe 系列 PDF 浏览器以保证正常浏览。代码如下：

```
\begin{figure}[!hbt]
  \centering
  \animategraphics[controls, autoplay, loop,
    width=0.6\linewidth]{20}{Py3-matplotlib-}{0}{98}
\end{figure}
```

以上代码对应的动态图[7]给出如图5.2所示：

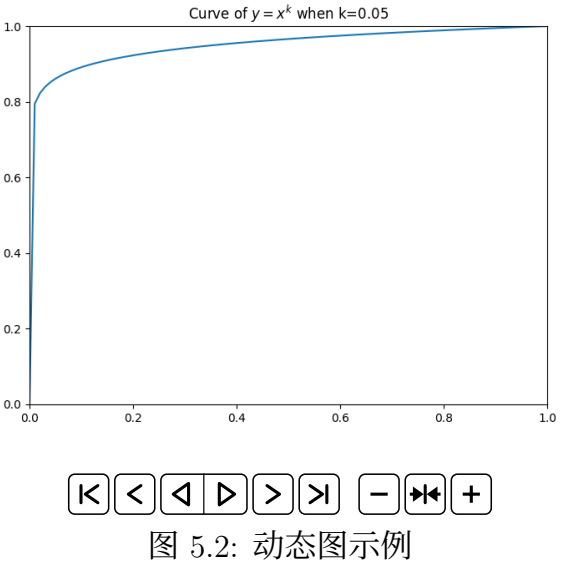


图 5.2: 动态图示例

以上会搜索文件夹（包括你在 `graphicx` 中设置的文件夹），找到图片依次序编号的从“Py3-matplotlib-0.png”到“Py3-matplotlib-98.png”的这 99 张图片，以每秒 20 帧为默认播放速度加载。参数 `controls` 表示在图片下方附加控制按钮，可以暂停/播放，正放/倒放，手动浏览帧，以及更改播放速度。参数 `autoplay` 表示当阅读者浏览到动态图所在页面时，动态图会自动开始播放。参数 `loop` 表示播放到尾帧后自动重播。最后，你可以像一般图片加载一样，指定它的 `width/height`。

注意：如果你只有 GIF 图像，但安装了ImageMagick，可以在图像文件夹下使用命令行命令：

```
convert Py3-matplotlib.gif -coalesce Py3-matplotlib.png
```

来将单个 GIF 转为符合上述要求的多个 png 图像。

5.7 自定义编号列表

编号列表的自定义主要使用`enumitem`宏包。主要的计数器有：

- enumerate:
  - **Counter:** enumi, enumii, enumiii, enumiv
  - **Label:** labelenumi, labelenumii, ...
- itemize: 只有 Label，该列表没有 Counter。

[7] 该例由 Python - matplotlib 绘制。可以参考此页面的附录。

  - **Label:** labelitemi, labelitemii, …
- description: 只有`\descriptionlabel`定义，默认：

```
\newcommand*{\descriptionlabel}[1]{\hspace\labelsep
    \normalfont\bfseries #1} % \labelsep标签间距，默认0.5em
```

列表`enumerate`, `itemize`的默认参数见表5.5。

表 5.5: 编号列表默认参数表

环境	层	Label	默认	Counter	默认
`enumerate`	1	`\labelenumi`	`\theenumi.`	`\theenumi`	`\arabic{enumi}`
	2	`\labelenumi`	`\(theenumii)`	`\theenumii`	`\alph{enumi}`
	3	`\labelenumiii`	`\theenumiii.`	`\theenumiii`	`\roman{enumi}`
	4	`\labelenumiv`	`\theenumiv.`	`\theenumiv`	`\Alph{enumiv}`
`itemsize`	1	`\labelitemi`	`\textbullet`	.....	•
	2	`\labelitemii`	`\textendash`	.....	–
	3	`\labelitemiii`	`\textasteriskcentered`	.....	*
	4	`\labelitemiv`	`\textperiodcentered`	.....	·

在`enumerate`列表中，编号样式按照：1.→(a)→i.→A 的顺序嵌套，分别代表`\theenumi`, `\theenumii`, `\theenumiii`, `\theenumiv`的值。你可以通过计数器命令来指定编号样式，不过要额外加上一个星号，比如`\arabic*`表示阿拉伯数字。一个例子：

```
\begin{enumerate}\item First
  \begin{enumerate}\item Second
    \begin{enumerate}\item Third
      \begin{enumerate}
      \item Fourth Layer
\end{enumerate}\end{enumerate}
\end{enumerate}\end{enumerate}
% 改为首层小写罗马数字，放于圆括号
\renewcommand{\theenumi}
  {\roman{enumi}}
\renewcommand{\labelenumi}
  {(\theenumi)}
\begin{enumerate}
\item First-layer symbol has
    changed!
\end{enumerate}
```

1. First
   (a) Second
      i. Third
         A. Fourth Layer
(i) First-layer symbol has changed!

你也可以在`ctex`宏包被调用（包括 ctex 文档类被使用）时，在导言区加入：

```
\AddEnumerateCounter{\chinese}{\chinese}{}
```

这样就可以将汉字指定为编号样式了。

宏包`enumitem`可添加参数于列表后，像`\begin{list}[options]`：

`label` 定义`enumerate`环境的编号样式，或者`itemize`环境的符号样式。

`ref` 设置嵌套序号格式，比如 `[ref=\emph{\alph*}]` 表示引用的上层序号是强调后的小写字母。你也可以写：`[label=\alph{enumi}. \roman*]`。

label* 加在enumerate上层序号上。比如上层是 2，那么就是 2.1, 2.1.1……
font/format 设置 label 的字体。如果环境是 description, 那么就会设置\item命令后方括号内的文本字体。
align 对齐方式默认 right, 也可以选择 left/parleft.
start 初始序号。start=2 表示初始序号是 2, b, B, ii 或 II.
resume 不需赋值的布尔参数。表示接着上一个enumerate环境的结尾进行编号。
resume* 不需赋值的布尔参数。表示完全继承上一个enumerate环境的参数。如果你常常使用这个命令，也许你可以新定义一个列表环境。
series 给当前列表起名（比如 mylist），可以在后文中用 resume=mylist 进行继续编号。
style 定义description列表的样式。

standard: label 放在盒子中。
unboxed: label 不放在盒子中，避免异常长度或空格。
nextline: 如果 label 过长，text 会另起一行。
sameline: 无论 label 多长，text 从 label 同一行开始。
multiline: label 会被放在一个宽为 leftmargin 的 parbox 中。

在列表定义中可能碰到的参数如图5.3。其中加粗加斜的 LaTeX 不原生支持。

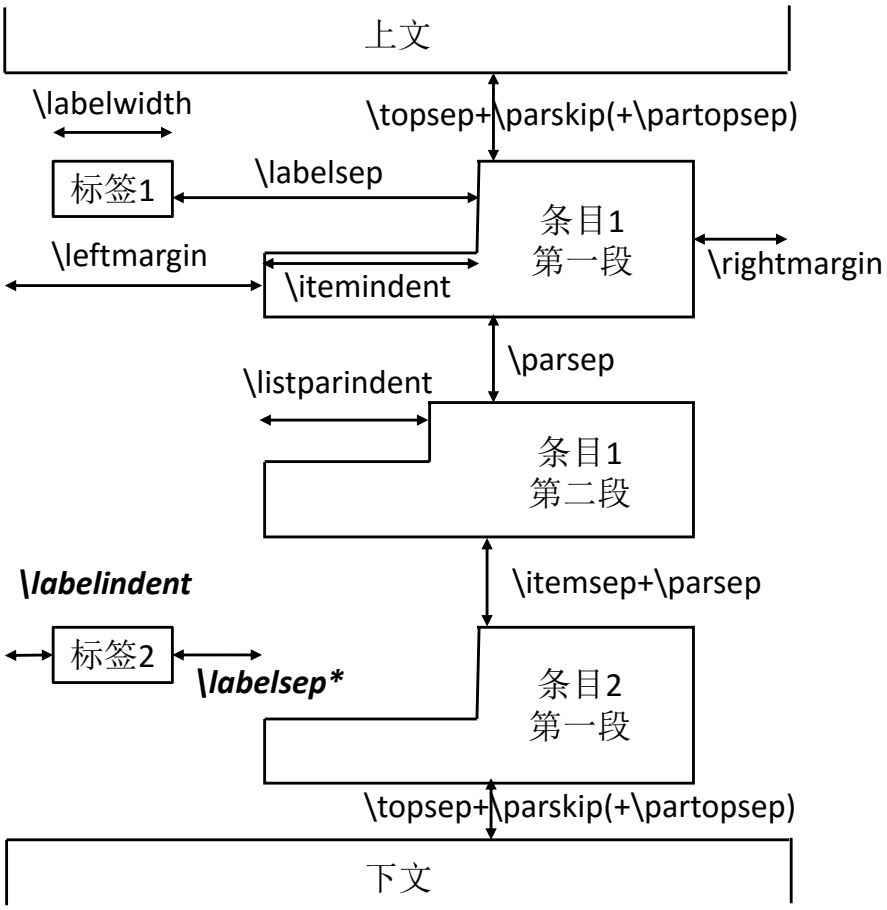


图 5.3: 列表长度参数总图

图5.3中的竖直空距 topsep, partopsep, parsep, itemsep，以及水平空距 left-

/rightmargin, listparindent, labelwidth, labelsep, itemindent 都是可以直接以 key=value 的形式写在列表环境后做参数的。

命令\setlist，用于定义列表环境的样式。比如可以更改原有的列表：

```
\setlist[enumerate]{label=\arabic* -,
    font=\bfseries, itemsep=0pt}
\setlist[itemize]{label=$\bullet$,
    font=\bfseries,leftmargin=\parindent}
\setlist[description]{font=\bfseries\uline}
```

最后，说一下行内列表。在加载enumitem宏包时使用 inline 选项即可启用，环境名是enumerate*. 参数有：

before 在行内列表插入前的文本，一般是冒号。

itemjoin 各\item之间的文本，一般是逗号或者分号。

itemjoin* 倒数第二个与最后一个\item间的文本，一般是 ", and" 或者 "，还有" 之类。

几个小例子。description环境：

```
\begin{description}
[font=\bfseries\uline]
    \item[This]is BFSERIES.
    \item[And]this also.
\end{description}
```

This is BFSERIES.

And this also.

编号数字左端与左页边平齐：

```
\begin{enumerate}[leftmargin=*]
```

Here we go. This is a very long sentence and you will find that it goes to the second line in order to show how long its parindent is.

1. The left sides
2. of the label number
3. have equal indent with
4. the text parindent.

编号数字左端与段首缩进位置平齐：

```
\begin{enumerate}[labelindent=\parindent,leftmargin=*]
```

Here we go. This is a very long sentence and you will find that it goes to the second line in order to show how long its parindent is.

1. The left sides
2. of the label number
3. have equal indent with
4. the text parindent.

编号项目正文与段首缩进位置平齐：

```
\begin{enumerate}[leftmargin=\parindent,start=3]
```

Here we go. This is a very long sentence and you will find that it goes to the second line in order to show how long its parindent is.

3. An item can be extremely long. You cannot know how its parindent works if it is too short to reach the second line.
4. This is short.

标签加框：

```
\begin{enumerate}[label=\fbox{\Roman*},labelindent=\parindent]
```

Here we go. This is a very long sentence and you will find that it goes to the second line in order to show how long its parindent is.

I An item can be extremely long. You cannot know how its parindent works if it is too short to reach the second line.

II This is short.

最后，本手册使用了如下 5 种：

```
\begin{description}[font=\bfseries\uline,labelindent=\parindent,
    itemsep=0pt,parsep=0pt,topsep=0pt,partopsep=0pt]
\begin{description}[font=\bfseries\ttfamily,itemsep=0pt,
    parsep=0pt,topsep=0pt,partopsep=0pt]
\begin{enumerate}[font=\bfseries,labelindent=0pt,itemsep=0pt,
    parsep=0pt,topsep=0pt,partopsep=0pt]
\begin{itemize}[font=\bfseries,itemsep=0pt,parsep=0pt,
    topsep=0pt,partopsep=0pt]
% 行内列表定义
\newenvironment{inlinee}
    {\begin{enumerate*}[label=(\arabic*), font=\rmfamily,
    before=\unskip{: }, itemjoin={{; }}, itemjoin*={{, 以及: }}]}
    {\end{enumerate*}。}
```

5.8 BibTeX 参考文献

首先说一下基础的使用。通过重定义`\refname`或`\bibname`，前者是 article 类，后者是 book 类。这点在第3.6.5节这一节已经介绍过。

关于怎样将参考文献正常编号并加入目录中，请参考第3.6.5节这一节。

```
\renewcommand{\bibname}{参考文献}
```

在文献目录之前、文献标题之下，用`\bibpreamble`插入一段文字：

```
\renewcommand{\bibpreamble}{以下是参考文献：}
```

用`\bibfont`更改参考文献的字体：

```
\renewcommand{\bibfont}{\small}
```

用\citenumfont定义在正文中引用时，文献编号的字体：

```
\renewcommand{\citenumfont}{\itshape}
```

用\bibnumfmt定义文献目录的编号，默认是 [1], ...形式。比如改成加点形式：

```
\renewcommand{\bibnumfmt}[1]{\textbf{#1.}}
```

文献项之间的间距更改，调整\bibsep即可：

```
\setlength{\bibsep}{1ex}
```

5.8.1 natbib 宏包

个人认为文献宏包首推natbib，不再推荐cite. natbib宏包的加载选项：

round (默认) 圆括号。
square/curly/angle 方括号/花括号/尖括号。
semicolon/comma 分号/逗号作为文献序号分隔符。
authoryear “作者 + 年代 (AuY)”模式显示参考文献。
numbers “数字编号 (num)”模式显示参考文献。
super 参考文献显示在上标。
sort(&compress) 排序文献序号（并压缩[8]）。
compress 压缩但不排序。
longnamefirst 长名称在前，缩写名称在后。
nonamebreak 防止作者名称中间出现断行。可能造成 Overfull 坏箱，但能解决某些 hyperref 异常。
merge 允许 * 形式的引用。
elide 在 merge 选项引用中，省略相同的作者或年份。

你也可以通过宏包提供的\setcitestyle命令：

1. 引用模式：authoryear, number 与 super 三种，含义同上。
2. 引用分隔符：semicolon, comma, 或者用 citesep={*sep*} 来指定。
3. 作者与年代间的符号：aysep={*sep*}
4. 同作者下多个年代间的符号：yysep={*sep*}
5. 说明文字后的符号：notesep={*sep*}

默认的参数是：

```
\setcitestyle{authoryear,round,comma,aysep={;},
    yysep={,},notesep={, }}
```

除了 LaTeX 原生的\cite命令，natbib宏包还提供了表5.6所示的引用命令。

[8]压缩是指：连续三个或以上的序号会显示为如 2–4 的形式。

表 5.6: natbib 宏包命令表

使用\Citet, \Citep, \Citealt, \Citealp确保姓名首字母大写			
\citet & \citet*			
AuY	citet{jon90}	⇒	Jones et al. (1990)
	citet[chap.2]{jon90}	⇒	Jones et al. (1990, chap.2)
	citet{jon90, jam91}	⇒	Jones et al. (1990); James et al. (1991)
	citet*{jon90}	⇒	Jones, Baker, and Williams (1990)
num	citet{jon90}	⇒	Jones et al. [21]
	citet[chap.2]{jon90}	⇒	Jones et al. [21, chap.2]
\citep & \citep*			
AuY	citep{jon90}	⇒	(Jones et al., 1990)
	citep[chap.2]{jon90}	⇒	(Jones et al., 1990, chap.2)
	citep[see][]{jon90}	⇒	(see Jones et al., 1990)
	citep[see][chap.2]{jon90}	⇒	(see Jones et al., 1990, chap.2)
	citep{jon90, jon91}	⇒	(Jones et al., 1990, 1991)
	citep*{jon90}	⇒	(Jones, Baker, and Williams, 1990)
num	citep{jon90}	⇒	[21]
	citep[chap.2]{jon90}	⇒	[21, chap.2]
	citep[see][]{jon90}	⇒	[see 21]
	citep[see][chap.2]{jon90}	⇒	[see 21, chap.2]
	citepjon90a,jon90b	⇒	[21, 32]
\cite			
AuY	此模式下与\citet相同		
num	此模式下与\citep相同		
\citealt: 与\citet相似，但没有括号。			
\citealp: 与\citep相似，但没有括号。			
\citenum: 引用文献编号。			
\citetext: 打印一段文本。			
\citeauthor & \citeauthor*: 引用文献的作者。带星表示显示该文献的全部作者。			
\citeyear & \citeyearpar: 引用文献的年份。par 的意思是在文献外加括号。			

5.8.2 BibTeX 使用

BibTeX 通过单独的.bib 扩展名文件管理文献，使多文档方便地共用一份文献列表（可以指引用其中的部分文献）成为可能。使用时请确保：

1. 确保你的文档定义了\bibliographystyle类型。LaTeX 预定义的类型分为：

   plain 按照第一作者字母顺序排序。“作者. 文献名. 出版商或刊物, 出版地, 出版时间. ”

   unsrt 按引用顺序排序。

   alpha 按作者名称和出版年份排序。

abbrv 缩写形式。

2. 在文档中插入了`\cite`等命令。
3. 在参考文献列表位置插入了`\bibliography`命令[9]。

一个简单的例子：

```
\bibliographystyle{plain}
\begin{document}
    ...
    ... and published here\cite{Smith93TRB}.
    ...
    \bibliography{myBib}
\end{document}
```

然后在你的 myBib.bib 文件中，你需要有类似这样的条目。等号后使用花括号或引号均可。

```
% 如果引用期刊
@article{Smith1993TRB,
    author = {作者，多个作者用and 连接},
    title = {标题},
    journal = {期刊名},
    volume = {卷20},
    number = {页码},
    year = {年份},
    abstract = {摘要，引用的时候自己参考，非必须}}
% 如果引用书籍
@book{Smith1993TRB,
    author ="作者",
    year="年份2008",
    title="书名",
    publisher ="出版社名称"}
```

这里只介绍了 article 和 book 两种类型。更多的文献类型以及它们的使用条目选项，参考表5.7。

最后，你可能需要编译 XeLaTeX，再编译 BibTeX，最后连续编译两次 XeLaTeX 来完成你的文档建立。

5.9 索引

使用`makeidx`宏包来建立索引。索引标题通过重定义`\indexname`更改。

1. 在导言区加载`makeidx`宏包，并输入`\makeindex`开始收集索引。
2. 在文中使用`\index`命令来插入索引标签。
3. 在需要插入索引列表的位置输入`\printindex`。

索引命令`\index`的用法如表5.8。注意：这四种符号`!|@"` 如果要写在参数中，请在它们之前添加一个双引号。

此外，`imakeidx`宏包可能更强，它允许索引分组：

[9]如果你在正文中使用`\nocite{ref-name}`命令，可以把 bib 文件中未 cite 的文献也加入到列表。想全部加入，在正文中使用`\nocite{*}`命令。

表 5.7: BibTeX 文献类型总表

article	期刊文献 必要：author, title, journal, year 选填：volume, number, pages, month, note
book	公开出版图书 必要：author/editor, title, publisher, year 选填：volume/number, series, address, edition, month, note
booklet	无出版商或作者的图书。必要：title 选填：author, howpublished, address, month, year, note
conference/ inproceedings	无出版商或作者的图书。必要：title 选填：author, howpublished, address, month, year, note
inbook	书籍章节 必要：author/editor, title, chapter and/or pages, publisher, year 选填：volume/number, series, type, address, edition, month, note
incollection	书籍含独立标题的章节，比如论文集的一篇 必要：author, title, booktitle, publisher, year 选填：editor, volume/number, series, type, chapter, pages, address, edition, month, note
manual	技术手册。必要：title 选填：author, organization, address, edition, month, year, note
mastersthesis	硕士论文 必要：author, title, school, year 选填：type, address, month, note
misc	其他 选填：author, title, howpublished, month, year, note
phdthesis	博士论文 必要：author, title, year, school 选填：address, month, keywords, note
techreport	教育，商业机构的技术报告 必要：author, title, institution, year 选填：type, number, address, month, note
unpublished	未出版的论文或图书 必要：author, title, note 选填：month, year

表 5.8: 索引命令\index 的使用

例子	效果
!: 分级索引，最多三级	
hello	hello, 1
hello!Foo	Foo, 2
hello!Foo!bar	bar, 3
@: 格式化，“排序字串 @ 显示样式”	
alpha@α	α, 4
BOLD@\textbf{BOLD}	**BOLD**, 5
|: 页码显示	
wow|(wow|)	wow, 6–13
Meow|textbf	Meow, **14**
Meow|see{hello}	Meow, *see* hello
Meow|seealso{wow}	Meow, *see also* wow

```
\makeindex[title={Group 1}]
\makeindex[title={Group 2},name=another]
% 以上在导言区，且需要\usepackage{imakeidx}
    ...\index{...}
    ...\index[another]{...}
\printindex
\printindex[another]
```

定制索引样式可使用imakeidx宏包；另一个宏包idxlayout也能实现这些功能，不过需要放置在前者之后加载。

想要将索引章节正常编号或编入目录项，使用 tocbibind 宏包，参考26。

关于索引，部分用户有制作词汇表的需求，请参考glossary宏包。

5.10 公式与图表编号样式

5.10.1 取消公式编号

取消单行公式的编号，用\[\]或者equation*环境，代替equation即可。

取消多行公式中某行的编号，使用amsmath宏包的\notag或\nonumber命令。命令放在对应行的末尾即可。该方法同样适用于equation环境。

取消多行公式中所有行的编号，请使用align*环境而不是align环境。你可以参考多行公式及其编号这一小节的内容。

5.10.2 增加公式编号

amsmath宏包提供了增加编号的\tag命令：

```
\[a^2>0 \tag{$\star$}\]
\begin{equation}
b^2 \geqslant 0
\tag*{[Axiom]}
\end{equation}
```

$$a^2 > 0 \qquad (\star)$$

$$b^2 \geqslant 0 \qquad [\text{Axiom}]$$

其中\tag*命令会去掉编号行间公式的小括号，从而定制性更强。如果想在多行公式中的某行添加编号，使用\numberthis命令。

5.10.3 父子编号：公式 1 与公式 1a

有时你需要叙述一些推论，你不希望这些推论被编号为公式，但是不进行编号又难以叙述。这时可以尝试amsmath宏包提供的subequations环境：

```
\begin{subequations}
This is an upright text.
\begin{align}
A' &=B+C \\
X &=0 \nonumber \\
D' &=E \times F
\end{align}
Enjoy this environment.
\end{subequations}
```

This is an upright text.

$$A' = B + C \qquad (5.2a)$$

$$X = 0$$

$$D' = E \times F \qquad (5.2b)$$

Enjoy this environment.

父子编号样式的定义参考这里。还有一种利用计数器的方式，可以插入公式 1 和公式 1’ 这样的效果，但是实现起来稍显麻烦。参考这里。

5.10.4 在新一节重新编号公式

只需要\numberwithin进行设置：

```
\numberwithin{equation}{section}
```

对于 article 文档类，显示为 (2.1), (2.2), ...; 对于 chapter 等文档类，显示为 (1.2.1), (1.2.2), ...这样。

5.10.5 公式编号样式定义

通过控制计数器的方式可以方便地自定义公式编号样式：

```
% 更改为: (2-i), (2-ii)
\renewcommand{\theequation}{\thechapter-\roman{equation}}
% 父子公式编号样式, 在subequation环境内使用
% 效果: (4.1-i), (4.1-ii)
\renewcommand{\theequation}
```

```
    {\theparentequation-\roman{equation}}
```

公式 1 和公式 1' 的实现方法可以这样：

```
\begin{equation}\label{eq:example}
    A=B,\ B=C
% 需要标记的公式内部给myeq计数器赋值
\setcounter{myeq}{\value{equation}}
\end{equation}
插入另一个式子：
\begin{equation}
    D>0
\end{equation}
由式\ref{eq:example}可以推出式\ref{eq:example'}：
\begin{equation}\label{eq:example'}
    \tag{\arabic{chapter}.\arabic{myeq}$'$}
    A=C
\end{equation}
```

5.11 附录

附录可以直接在文中需要开启的地方，使用下述命令。此后的最高大纲级别编号会变为大写英文字母：A, B, C, ...

```
\appendix
```

你也可以使用appendix宏包。加载时常用的选项：

titletoc： 目录中显示为 “Appendix A” 而不是只有一个 “A”。如果你不喜欢 Appendix 这个名称，用重定义\appendixname命令即可。

header： 在附录页的标题前插入同样的名称。

附录一般出现在document环境内部的最后，例如：

```
% 导言区：\usepackage[titletoc]{appendix}
\begin{appendices}
\renewcommand{\thechapter}{\Alph{chapter}}
\titleformat{\chapter}[display]{\Huge\bfseries}
    {附录\Alph{chapter}}{1em}{}
\chapter{...} ...
\end{appendices}
```

本手册使用的是\appendix命令，并在其后重定义了 chapter 的显示样式。读者可以从源码中参考。

5.12 自定义浮动体 *

可以使用newfloat宏包自定义浮动体。形如：

```
\DeclareFloatingEnvironment[options]{float-name}
```

选项包括：
name 标签内容。如 name= 插图。
listname 目录名。如 listname= 插图目录。
fileext 目录文件扩展名，默认是 *float-name* 前加 lo。
placement 位置参数 htbp。
within 父计数器名称。比如 chapter. 可以设为 none.
chapterlistsgaps 赋值 on/off. 是否允许浮动体目录中，不同章节的浮动体间有额外的空距。
要输出浮动体目录，请插入命令`\listof[float-name]s`。

5.13 编程代码与行号 *

5.13.1 listings 宏包

编程代码不是用`verbatim`环境输出的……`listings`宏包是个好选择。你可以打开它的宏包文档查看它支持的编程语言，包括 C/C++, Python, Java 等等。当然还有 LaTeX，如果它也算编程代码的话。你也可以自定义一个全新的语言。预定义或自定义的语言均可用`\lstset`命令来设置。

```
\lstdefinelanguage{languagename}{key=value}
% 设置
\lstset{language=languagename, key=value}
```

可调整的参数包括：
language 用于`\lstset`中，表示设置只对该语言生效。该参数有可选参数，比如 [Sharp]C 表示 C#，具体需要查看 listings 宏包。
basicstyle 基础输出格式，一般是`\small\ttfamily`
commentstyle 注释样式。
keywordstyle 保留词样式。
sensitive 保留词是否大小写敏感。默认 false，可选 true/false。
stringstyle 字符串样式。
showstringspaces 显示字符串中的空格。
numbers 行号样式，默认是 none. 可选 left/right.
stepnumber 默认是 1，即每行都显示行号。
numberfirstline 默认 false，即如果 stepnumber> 1，首行不显示数字。
numberstyle 行号样式。
numberblanklines 默认 true，即在空白的代码行也显示行号。
firstnumber 默认 auto，可选 last 或填入数字，表示起始行号。
frame 默认 single，即 trbl，分别表示 top, right, buttom, left 四条边的线都是单线。如果想变某边为双线，大写它：trBL.
frameround 在`\lstset`中，从右上角顺时针设置，代码框为直角或圆角。比如 fttt 表示仅右上为直角。
framerule 代码框的线宽。
backgroundcolor 定义 frame 里代码的背景色，如`\color{red}`。
belowskip 默认`\medskipamount`。代码框下端到下文正文的竖直距离。

aboveskip 类似。

emptylines 设置最多允许的空行，比如 =1，会使多于 1 行的空行全部删除，并不计入行号。如果写为 =*1，那么被删除的行的行号仍然会保留计数。

esacpeinside 暂时脱离代码环境而输入一些 LaTeX 支持的命令，比如临时输入斜体。一般设置为一对重音符号（键盘数字 1 左侧的符号）。

代码环境的调用方式是`lstlisting`环境，例如：

```
\begin{lstlisting}[language=Python]
for loopnum in lst:
    sum += lst[loopnum]
\end{lstlisting}
```

也可以利用`\lstnewenvironment`命令定义一个代码输出环境：

```
\lstnewenvironment{envi-name}
    [opt][opt default]
    {before-code}{after-code}
% 调用：
\begin{envi-name}...\end{envi-name}
```

如果只想在行内输出，可以像这样用`\lstinline`定义：

```
\newcommand{\inlatexline}[1]{{\lstinline
    [language=TeX,basicstyle=\small\ttfamily]{#1}}}
```

有时候，一些关键词并没有被宏包成功高亮，或者你需要更多种类的高亮方式，这时候你可以自己设置：

```
\lstset{language=..., classoffset=0,
    morekeywords={begin,end},
    keywordstyle=\color{brown},
    classoffset=1,...}
```

除了 morekeywords 外，你可以使用：

```
\lstdefinestyle{...}{
    morecomment=[l]{//}, % 单行注释
    morecomment=[s]{/*}{*/}, % 多行注释，不可嵌套
    morecomment=[n]{(*}{*)}, % 多行注释，可嵌套
    morestring=[b]", % 字符串
% 如果想在字符串内输出该符，前加反斜杠即可
```

代码环境在复制的时候怎么才能不复制前面的行号呢？在导言区加上，

```
\usepackage{accsupp}
\newcommand{\emptyaccsupp}[1]
    {\BeginAccSupp{ActualText={}}#1\EndAccSupp{}}
% 在lstset的numberstyle中加入
...numberstyle=...\emptyaccsupp...,
```

并请用 Adobe Reader 等功能健全的 PDF 阅读打开 pdf！如果是 Sumatra 仍然可能会选中前面的行号。

最后，给出本手册旧版中使用的 LaTeX 的`\lstset`，颜色另行定义：

```
\lstset{language=[LaTeX]TeX,
    basicstyle=\small\ttfamily,
    commentstyle=\color{commentcolor},
    keywordstyle=\color{keywordcolor},
    stringstyle=\color{stringcolor},
    showstringspaces=false,
    % Package/Tikz-Lib Using
    classoffset=0,
    morekeywords={begin,end,usetikzlibrary},
    keywordstyle=\color{keywordcolor},
    classoffset=1,
    morekeywords={article,report,book,xeCJK,tikz,calc},
    keywordstyle=\color{packagecolor},
    classoffset=2,
    morekeywords={document,tikzpicture},
    keywordstyle=\color{envicolor},
    % Line Number Style
    numbers=left,stepnumber=1,
    numberstyle=\tiny\emptyaccsupp,
    % Frame and Background Color
    frame=single,framerule=0pt,
    backgroundcolor=\color{backcolor},
    % Spaces
    emptylines=1,escapeinside=``}
```

5.13.2 tcolorbox 宏包

本手册在修订过程中发现了一个可以方便地“一侧写源代码，另一侧展示结果”的宏包，名叫`tcolorbox`；因此基本用其替换了原有的`listings`宏包——但是引擎仍然使用的是`listings`，在使用新宏包时也需要借助旧宏包选项来设置引擎。

该宏包支持下，本手册用`\newtcblisting`定义了 LaTeX 代码环境：

```
\usepackage{tcolorbox}
  \tcbuselibrary{listings,skins,breakable}
% listings是代码展示引擎，breakable为了可跨页
\newtcblisting{latex}{breakable,skin=bicolor,colback=gray!30!white,
  colbacklower=white,colframe=cyan!75!black,listing only,
  left=6mm,top=2pt,bottom=2pt,fontupper=\small,
  % listing style
  listing options={style=tcblatex,
  keywordstyle=\color{blue},commentstyle=\color{green!50!black},
  numbers=left,numberstyle=\tiny\color{red!75!black}\emptyaccsupp,
  emptylines=1,escapeinside=}}
```

其中很多选项意义显然，就不赘述了。需要指明的有：

skin bicolor，让源代码和显示结果可以分开设置背景色。

colbacklower 背景色。tcolorbox 分为两段，下段（或右段）叫 lower。

fontupper 上段（或左段）叫 upper，这是设置在进入 upper 前插入的格式命令，不局限于字号。

代码展示参数 该参数经常用到的是：

listing only 仅展示源代码。也有 text only 选项。
listing and text 上段源代码，下段结果。本手册的codeshowabove环境采用该参数。如果 text 与 listing 交换，即上段结果下段源代码。
listing side text 左段源代码，右段结果。本手册的codeshow环境就采用了该参数。同样也可以交换，变成左段结果右段源代码。
listing outside text 同上，只是结果“看起来”在盒子外。
listing option 除了特殊的 style 字段，这些参数都会被传递给引擎（本手册是 listings 宏包）。style=tcblatex 是tcolorbox宏包预定义的。

通过该宏包的\newtcbox命令，本手册实现了对于命令、环境、宏包的高亮。以下给出宏包高亮作为例子，参数含义显然：

```
\newtcbox{\pkg}[1][orange!70!red]{on line,before
    upper={\rule[-0.2ex]{0pt}{1ex}\ttfamily},
    arc=0.8ex,colback=#1!30!white,colframe=#1!50!black,
    boxsep=0pt,left=1.5pt,right=1.5pt,top=1pt,bottom=1pt,
    boxrule=1pt}
```

实际上tcolorbox的强大之处远不止此，它能做出颜值很高的箱子样式。更多的内容请自行查阅其宏包文档学习。

5.13.3 行号

行号使用lineno宏包进行生成，在此简单介绍宏包选项：
left 默认选项，行号出现在左页边。
right 右页边。
switch 对于双页排版的文档，偶数页左页边，奇数页右页边。
switch* 对于双页排版的文档，置于内侧页边。
running 默认选项。整个文档进行计数。
pagewise 每页行号重新从 1 计数。
modulo 每 5 行显示行号。
displaymath 自动将 LaTeX 默认行间公式放在新定义的linenomath环境中。
mathline 如果使用了linenomath环境，则对其中的数学公式也编行号。

如果你想开始编号，使用\linenumbers[number]命令，其中 *number* 表示起始行号；并用\nolinenumbers来结束。或者选择使用(running)linenumbers环境，并且仿上设置起始行号。如果使用\linenumbers*，\runninglinenumbers或者linenumbers*、runninglinenumbers*环境，那么行号会自动从 1 开始。

你可以使用\resetlinenumber[number]，在某处把行号设置为某个数值。

如果想要每 N 行显示行号，使用\modulolinenumbers[N]命令。比如：

```
\begin{linenumbers}
    \modulolinenumbers[3]
    ...TEXT...
\end{linenumbers}
```

本节的 subsection 后到代码段前的所有内容就是包含在如上的环境中的。

6 TikZ 绘图 *（编写中）

请注意：该章节的版本有些陈旧，作者将在下个较大的版本更新中进行重写。

6.1 所需宏包与命令

使用 TikZ，首先需要加载宏包`tikz`：

```
\usepackage{tikz}
% 有时也需要库：\usetikzlibrary{...}
```

接着，需要环境`tikzpicture`用于输出 TikZ 代码。在本手册中，如果没有特殊说明，代码都包括在这个环境中。

```
\begin{tikzpicture}
\draw (-0.5,0) -- (0.5,0);
\draw (0,-0.5) -- (0,0.5);
\end{tikzpicture}
```

你也可以用`\tikz`命令代替以上环境输出简单图像。它需要一个花括号参数，或者搜索分号来结束语句。

```
\tikz \draw (-0.5,0) -- (0.5,0); % 二者等价
\tikz{\draw (-0.5,0) -- (0.5,0)}
```

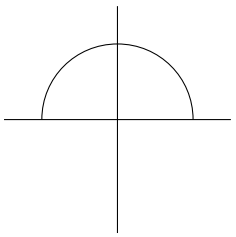
命令`\draw`是最基础的 TikZ 命令，用于画线。单位 1 的长度是 1cm。

6.2 基础命令

6.2.1 路径：直线与曲线

路径 (path) 是 TikZ 的基础单元，指一系列连接在一起的直线或曲线。绘制直线和曲线都使用`\draw`命令，只是曲线可以用 `controls` 指定曲线上任意两点间的切线方向。准确地说，这是 Bézier 曲线。

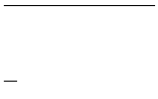
```
\draw (-1.5,0) -- (1.5,0);
\draw (0,-1.5) -- (0,1.5);
\draw (-1,0) .. controls (-1,0.555) and
    (-0.555,1) .. (0,1)
              .. controls (0.555,1) and
    (1,0.555) .. (1,0);
```



6.2.2 长度与单位

默认单位 1 的长度是 1cm，但是你也可以指定单位：

```
\draw (0,1)--(2,1);
\draw (0,0)--(5pt,0);
```



重定义单位长度也是可以的：

```
\draw[x=5pt,y=5pt] (0,0)--(0,2);
```


6.2.3 线宽

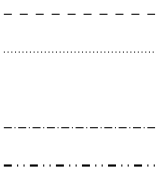
线宽可以通过 `ulrta thin` , `very thin` , `thin` , `semithick` , `thick` , `very thick` , `ultra thick` 这七种预定义的线宽来指定。它们的宽度依次上升。也可以直接用 `line width` 指定。

```
\draw[very thin] (0,0) -- (1,0);
\draw[ultra thick] (1,0) -- (1,1);
\draw[line width=10pt] (2,0) -- (2,1);
```

6.2.4 线型

常用的线型除实线外，有虚线 (`dashed`)、点线 (`dotted`)、点划线 (`dash dot`) 和双点画线 (`dash dot dot`) 四种。还可以配合 `loosely` 和 `densely` 进行调整。

```
\draw[dashed] (0,2.5) -- (2,2.5);
\draw[densely dotted] (0,2) -- (2,2);
\draw[densely dash dot] (0,1) -- (2,1);
\draw[dash dot dot,thick] (0,0.5) -- (2,0.5);
```



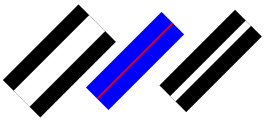
线型也可以自由设置，用选项 dash pattern 。并可用 dash phase 指定线型起始的距离：

```
\draw[dash pattern=on 1cm off .25cm on .25cm
    off .5cm,dash phase=1cm] (0,0)--(4,0);
```


6.2.5 双线

选项 double 用于设置双线，另有 double distance 及 double distance between line centers 可选：

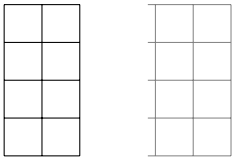
```
\draw[line width=.2cm,double distance=.3cm]
    (0,0)--(1,1);
\draw[line width=.2cm,draw=blue,double=red]
    (1,0)--(2,1);
\draw[line width=.2cm,double distance
    between line centers=.3cm] (2,0)--(3,1);
```



6.2.6 网格

TikZ 支持 grid 绘制网格，指定左下角和右上角即可。help lines 还预定义了一种颜色较浅、宽度较小的线型。

```
\draw[step=0.5] (0,0) grid (1,2);
\draw[step=0.5,help lines] (1.9,0) grid
    (3,2);
```



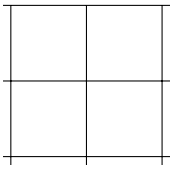
6.2.7 缩放：全局与局部

你可以在单个\draw后使用缩放 scale ，但更常见的是在环境后统一缩放：

```
\begin{tikzpicture}[scale=0.5]
\draw[step=0.5,help lines] (1.9,0) grid
    (3,2);
\end{tikzpicture}
```

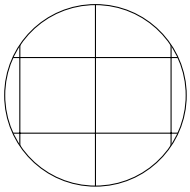
局部的缩放\clip实现了剪裁的效果：

```
\clip (-1.1,-1.1) rectangle (1.1,1.1);
\draw[step=1] (-2,-2) grid (2,1);
```



clip 与 draw 实质都是\path命令的参数的简写形式。使用该命令可以在剪裁时画出剪裁边缘：

```
\path[clip,draw] (0,-0.5) circle
    [radius=1.2];
\draw[step=1] (-2,-2) grid (2,1);
```

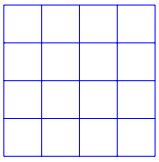


剪裁命令在剪裁后经常需要放大，配合 scale 选项较多。

6.2.8 自定义

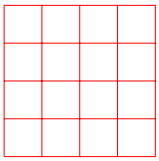
选项 help lines 实质是 style=very thin, gray 的联合。使用\tikzset命令设置它：

```
\tikzset{help lines/.style =
    {color=blue,very thin}}
\draw[step=0.5,help lines] (0,0) grid (2,2);
```



或者在使用环境时直接将其作为选项写上：

```
\begin{tikzpicture}[help lines/.style =
    {color=red,very thin}]
\draw[step=0.5,help lines] (0,0) grid (2,2);
\end{tikzpicture}
```



你也可以自定义选项，并且带参数：

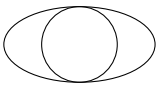
```
\tikzset{mystyle/.style={very
    thick,color=#1!50},
mystyle/.default=blue}
\draw[mystyle] (0,2) -- (3,2);
\draw[mystyle=green!50!black] (0,0) -- (3,0);
```

6.3 几何形状

6.3.1 圆与椭圆

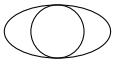
圆与椭圆有专门的选项 circle 与 ellipse，指定圆心和半径：

```
\draw (1,1) circle (0.5);
\draw (1,1) ellipse (1 and 0.5);
```



也可以写全参数：

```
\draw (0,0) circle [radius=10pt];
\draw (0,0) ellipse [x radius=20pt, y
    radius=10pt];
```



6.3.2 圆弧与椭圆弧

圆弧可以通过指定起点角度、终点角度、圆弧半径来绘制，选项 arc：

```
\draw (3,0) arc (0:30:3);
ndraw (-2,1) arc (180:-45:2 and 1)
%等价: \draw (3,0) arc [start angle=0,end
    angle=30,radius=3];
```

类似地绘制椭圆弧：

```
\draw (0,0) arc (0:270:0.7 and 0.4);
```

6.3.3 矩形

矩形的命令是 rectangle，指定左下角点和右上角点：

```
\draw (0,0) rectangle (2,1);
```

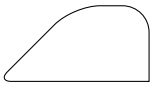
6.3.4 圆角与倒角

选项 rounded corners 和 sharp corners 控制圆角和倒角。默认值是 4pt。

```
\draw [rounded corners] (0,0) -- (1,1)
-- (2,0) .. controls (3,1) .. (4,0);
```

如果写在两个点之间，可以“开关”圆角和尖角：

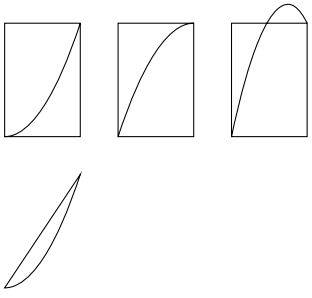
```
\draw (0,0) [rounded corners=10pt] -- (1,1)
    -- (2,1)
[sharp corners] -- (2,0)
[rounded corners=5pt] -- cycle;
```



6.3.5 抛物线 *

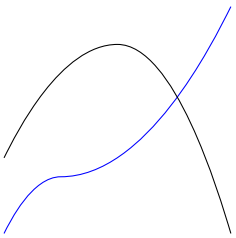
并不常用的 parabola 以及 bend，请读者自行理解：

```
\draw (0,0) rectangle (1,1.5) (0,0) parabola
    (1,1.5);
\draw[xshift=1.5cm] (0,0) rectangle (1,1.5)
    (0,0) parabola[bend at end] (1,1.5);
\draw[xshift=3cm] (0,0) rectangle (1,1.5)
    (0,0) parabola bend (.75,1.75) (1,1.5);
\draw[yshift=-2cm] (1,1.5) --(0,0) parabola
    cycle;
```



其中 bend 的含义是“拐点”，平常理解即一阶导数符号变化的点。理解：

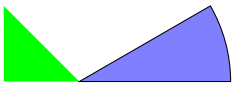
```
\draw[color=blue] (0,0) parabola[bend
    pos=0.25] (3,3);
\draw (0,1) parabola[parabola height=2cm]
    (3,0);
```



6.3.6 填充与渐变

填充命令\fill只能用于闭合对象，需要 cycle 选项或者手动闭合到第一个点。也可以在填充的同时绘制，使用\filldraw命令[1]。

```
\fill[green] (0,0)--(1,0)--(0,1)--cycle;
\filldraw[fill=blue!50!white,draw=black]
    (1,0)--(3,0) arc (0:30:2) -- cycle;
```

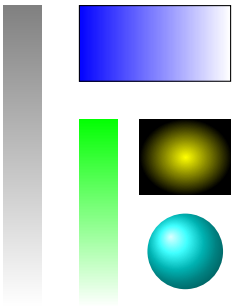


填充还能根据重叠次数来判断填充与否，even odd rule 例子：

```
\fill[even odd rule, blue] (0,0) -- (2,0.5)
    -- (1,1) circle (0.25);
```

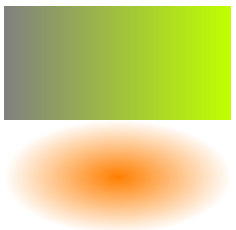
类似的渐变命令是\shade和\shadedraw：

```
\shade (0,0) rectangle (0.5,4);
\shade[top color=green,bottom color=white]
    (1,0) rectangle (1.5,2.5);
\shadedraw[left color=blue,right
    color=white] (1,3) rectangle (3,4);
\shade[inner color=yellow,outer color=black]
    (1.8,1.5) rectangle (3,2.5);
\shade[ball color=cyan] (2.4,0.75) circle
    (0.5);
```



上例的 ball 只是一种渐变类型，默认是 axis，另外还有 radial 可选。在 axis 下，还可以选择渐变角度 shading angle。

```
\shade[shading=radial,inner color=orange]
    (0,0) rectangle (3,1.5);
\shade[shading angle=45,right color=lime]
    (0,1.5) rectangle (3,3);
```



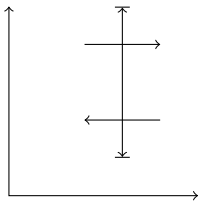
6.4 坐标系与坐标

6.4.1 箭头

绘制坐标系的方法非常基础，使用带箭头的直线：

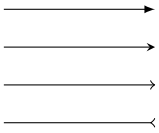
[1] 颜色命令可能需要xcolor宏包。

```
\draw[->] (1,2)--(2,2);
\draw[<-] (1,1)--(2,1); % 反过来画
\draw[<->] (0,2.5)--(0,0)--(2.5,0);
\draw[|<->|] (1.5,0.5)--(1.5,2.5); % 很实用
```



也有一些其他风格的箭头，更多的请调用arrows.meta这个库。

```
\draw[-latex] (0,3)--(2,3);
\draw[-stealth] (0,2.5)--(2,2.5);
\draw[-to] (0,2)--(2,2);
\draw[-to reversed] (0,1.5)--(2,1.5);
```

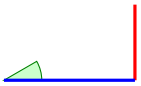


6.4.2 极坐标与相对坐标

直角坐标的表示非常简单，比如 (0,1)。也可以指定单位，比如 (0,1pt)。而极坐标则可以用 (30:1) 的方式表示在 30° 方向上的 1 单位长位置的点。

加号可以用来指代相对坐标，是相对该语句中最后一个绝对坐标而言的；而两个加号用于移动“笔”的位置，是相对该语句最后一个坐标而言的。体会：

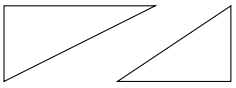
```
\filldraw[fill=green!20,draw=green!50!black]
    (0,0) -- (0.5,0)
arc [start angle=0, end angle=30,
    radius=0.5] -- cycle;
\draw[red,very thick] (30:2) -- +(0,-1);
\draw[blue,very thick] (30:2) ++(0,-1) --
    (0,0);
```



6.4.3 横纵坐标投影

这个命令是象形的：-| 与 |-两种。

```
\draw (0,0)--(2,1) -| cycle;
\draw (1.5,0)--(3,1) |- cycle;
```



6.5 输出单个文件

要输出为.svg 矢量文件，用于更多的插图场合。需要在电脑安装 pdf2svg[2]。不过在 LaTeX 使用的场合，可以去掉下述的 convert 参数，以输出.pdf 格式的矢量文件。

下例中的 multi=false 表示只输出为单页文件。

```
\documentclass[tikz,convert=pdf2svg,multi=false]{standalone}
% tikz package already loaded by 'tikz' option
\begin{document}
```

[2] http://www.cityinthesky.co.uk/opensource/pdf2svg/

```
\begin{tikzpicture}
  \draw (0,0) -- (10,10);
  \draw (10,0) -- (0,10);
\end{tikzpicture}
\end{document}
```

在编译时如果是 XeLaTeX，还需要添加参数：

```
% 如果上例的文件名为 example.tex
xelatex -shell-escape example.tex
```

7 参考文献

[1] Goossens, Michel, Frank Mittelbach, and Alexander Samarin. *The LaTeX Companion*, 2nd edition. Addison-Wesley Reading Mass, 2004.

[2] Hubert Partl, Irene Hyna, and Elisabeth Schlegl. 一份不太简短的 *LaTeX2ε* 介绍. ChinaTeX 论坛译, 2016.
English: https://www.ctan.org/tex-archive/info/lshort/english
中文: https://github.com/CTeX-org/lshort-cn/releases

[3] Jean Pierre Casteleyn. *Visual TikZ (version 0.62)*. IUT Génie Thermique et Énergie, 2016.
http://tug.ctan.org/info/visualtikz/VisualTikZ.pdf

[4] Leslie Lamport. *LaTeX: A Document Preparation System*, 2nd edition. Addison-Wesley Reading Mass, 1994.

[5] 刘海洋. *LaTeX* 入门. 电子工业出版社, 2013.

§ 附录 注音符号 | A

表 A.1: 注音符号与特殊符号

样式 - 命令	样式 - 命令	样式 - 命令	样式 - 命令
ō - \=o	ó - \'o	ǒ - \v o	ò - \`o
ô - \^o	ö - \"o	ȯ - \.o	ő - \H{o}
o̩ - \d{o}	ŏ - \u{o}	o̲ - \b{o}	o͡o - \t{oo}
$\tilde{o}$ — $\tilde{o}$		$\hat{o}$ — $\hat{o}$	
ø - \o	Ø - \O	ɪ - \i	ȷ - \j
å - \aa	Å - \AA	æ - \ae	Æ - \AE
œ - \oe	Œ - \OE	¡ - !`	¿ - ?`

表 A.2: 国际音标输入表（部分）

样式 - 命令	样式 - 命令	样式 - 命令
ʣ - \textdzlig	ʃ - \textesh	ʧ - \textteshlig
ʤ - \textdyoghlig	ʌ - \textturnv	ə - \textschwa
ɡ - \textscriptg	θ - \texttheta	ʊ - \textupsilon
ɑ - \textscripta	ð - \dh	ɛ - \textepsilon
ɔ - \textopeno	ʒ - \textyogh	ŋ - \ng
重音	次重音	长音节
ˈ - \textprimstress	ˌ - \textsecstress	ː - \textlengthmark

注：*\dh* 命令在非 *CJK* 文档中有时编译会出现问题。

§ 附录 建议与其他 | B

除了参考文献列表中给出的书籍以外，我还推荐你用控制台在 TeX Live 中能找到的以下书籍：

texdoc usrguide TeXLive 自带的用户手册。
texdoc clsguide TeXLive 自带的文档类和宏包编写手册。
texdoc fntguide TeXLive 自带的字体使用手册。
texdoc symbols-a4 一份速查表，基本上所有的 LaTeX 字符命令都在这里了。
texdoc latexcheat 很有趣的命令表，只有两页。
texdoc impatient *TeX for the Impatient*, 一本介绍底层 TeX 的书。这也是我阅读的第一本 TeX 书，Knuth 的 *The TeXbook* 虽然血统正但是难啃啊。本书中译本在：https://bitbucket.org/zohooo/impatient/wiki/Home
texdoc texbytopic *TeX by Topic*, 个人觉得不如上面那本，但也许只是叙述方式不一样吧。

你可能还需要的功能：

mhchem 该宏包用于输入化学式，提供了`\ce`命令。