

一份（不太）简短的 L^AT_EX 2_ε 介绍

或 *111* 分钟了解 L^AT_EX 2_ε

英文作者：Tobias Oetiker

Hubert Partl, Irene Hyna and Elisabeth Schlegl

英文版本：Version 6.4, May 5, 2023

中文翻译：C_T_EX 开发小组

中文版本：版本 6.04，二零二四年四月

Copyright © 2024 Chinese T_EX Society.

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.3 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled “GNU Free Documentation License”.

中文版致谢

6.04 中文版致谢

参与此次修订的朋友包括：

atxy-blip	SainoNamkho	OsbertWang	syvshc
chenxijun	yunhao94	hushidong	

完整的贡献者列表见 [GitHub contributors list](#)。

5.11 中文版致谢

对 lshort-zh-cn 的新一轮修订在 ChinaT_EX 交流 QQ 群 (91940767) 的网友支持下完成。参与此次修订的朋友包括：

网友 ID	E-mail
MkSwQi	1427799302@qq.com
hello	312439151@qq.com
ε	554502177@qq.com

4.20 中文版致谢

中文 T_EX 学会启动的 lshort-zh-cn 修正计划已经完工！本项计划历时八个月，参加的朋友有：

CT _E X 论坛 ID	翻译章节
zpxing	前言、第二章、第五章 1–2.4 & 3、第六章
Frogge	第一章
liwenjun	第三章
lijian605	第四章
gprsnl	第五章 2.5–2.11

haginile 和 Frogge 通读了全篇，并给出了详细的勘误表。blackold 对于第二章亦有所贡献。最后由 zpxing 统筹全书。

3.20 中文版致谢

本文档的翻译工作由 C_TE_X 版主“经典问题”倡议，历经近十个月才得以完成。期间参与翻译工作的朋友有：

C _T E _X 论坛 ID	翻译章节	源文件名
经典问题	前言	overview.tex
高原之狼	第一章	things.tex
controlong	第二章	typeset.tex
cxterm	第三章	math.tex, lssym.tex
aloft	第四章	spec.tex
ganzhi	第五章	custom.tex

在此特向这些奉献者表示感谢！

英文版致谢

Much of the material used in this introduction comes from an Austrian introduction to \LaTeX 2.09 written in German by:

Hubert Partl [<partl@mail.boku.ac.at>](mailto:partl@mail.boku.ac.at)

Zentraler Informatikdienst der Universität für Bodenkultur Wien

Irene Hyna [<Irene.Hyna@bmwf.ac.at>](mailto:Irene.Hyna@bmwf.ac.at)

Bundesministerium für Wissenschaft und Forschung Wien

Elisabeth Schlegl [<noemail>](mailto:noemail)

in Graz

If you are interested in the German document, you can find a version updated for \LaTeX 2 ϵ by Jörg Knappen at CTAN://info/lshort/german

The following individuals helped with corrections, suggestions and material to improve this paper. They put in a big effort to help me get this document into its present shape. I would like to sincerely thank all of them. Naturally, all the mistakes you'll find in this book are mine. If you ever find a word that is spelled correctly, it must have been one of the people below dropping me a line.

Eric Abrahamsen, Lenimar Nunes de Andrade, Eilinger August, Rosemary Bailey, Barbara Beeton, Marc Bevand, Connor Blakey, Salvatore Bonaccorso, Pietro Braione, Friedemann Brauer, Markus Brühwiler, Jan Busa, David Carlisle, Neil Carter, Carl Cerecke, Mike Chapman, Pierre Chardaire, Xingyou Chen, Christopher Chin, Diego Clavadetscher, Wim van Dam, Benjamin Deschwenden Jan Dittberner, Michael John Downes, Matthias Dreier, David Dureisseix, Hans Ehrbar, Elliot, Rockrush Engch, William Faulk, Robin Fairbairns, Johan Falk, Jörg Fischer, Frank Fischli, Daniel Flipo, Frank, Mic Milic Frederickx, David Frey, Erik Frisk, Hans Fugal, Robert Funnell, Greg Gamble, Andy Goth, Cyril Goutte, Kasper B. Graversen, Arlo Griffiths, Alexandre Guimond, Neil Hammond, Christoph Hamburger, Rasmus Borup Hansen, Joseph Hilferty, Daniel Hirsbrunner, Martien Hulsen, Björn Hvittfeldt, Morten Høgholm, Werner Icking, Eric Jacoboni, Jakob, Alan Jeffrey, Martin Jenkins, Byron Jones, David Jones, Johannes-Maria Kaltenbach, Nils Kanning, Andrzej Kawalec, Christian Kern, Alain Kessi, Axel Kielhorn, Sander de Kievit, Kjetil Kjernsmo, Tobias Klauser, Jörg Knappen, Michael Koundouros, Matt Kraai, Tobias Krewer, Flori Lambrechts, Mike Lee, Maik Lehradt, Rémi Letot, Axel Liljencrantz, Jasper Loy, Johan Lundberg, Martin Maechler, Alexander Mai, Claus Malten, Kevin Van Maren, Pablo Markin, I. J. Vera Marín, Hendrik Maryns, Chris McCormack, Aleksandar S. Milosevic, Henrik Mitsch, Stefan M. Moser, Philipp Nagele, Richard Nagy, Manuel Oetiker, Urs Oswald, Hubert Partl, Marcelo Pasin, Martin Pfister, Lan Thuy Pham, Breno Pietracci, Demerson Andre Polli, Maksym Polyakov, Nikos Pothitos, John Reffing, Mike Ressler, Brian Ripley, Kurt Rosenfeld, Bernd Rosenlecher, Chris Rowley, Young U. Ryu, Risto Saarelma, András Salamon, José Carlos Santos, Christopher Sawtell, Gilles Schintgen, Craig Schlenter, Hanspeter Schmid, Baron Schwartz, Jordi Serra i Solanich, Miles Spielberg, Susan Stewart, Matthieu Stigler, Geoffrey Swindale, Laszlo Szathmary, Boris Tobotras, Josef Tkadlec, Scott Veirs, Didier Verna, Carl-Gustav Werner, Fabian Wernli, Matthew Widmann, David Woodhouse, Chris York, Rick Zaccone, Fritz Zaucker, and Mikhail Zotov.

前言

L^AT_EX [1] 是一个文档准备系统 (Document Preparing System), 它非常适用于生成高印刷质量的科技类和数学类文档。它也能够生成所有其他种类的文档, 小到简单的信件, 大到完整的书籍。L^AT_EX 使用 T_EX [6] 作为它的排版引擎。

这份短小的手册描述了 L^AT_EX 2_ε 的使用, 对 L^AT_EX 的大多数应用来说应该是足够了。参考文献 [1, 2] 对 L^AT_EX 系统提供了完整的描述。

本手册在英文版 lshort 的基础上进行了适当的重新编排, 共有八章和两篇附录:

第一章 讲述 L^AT_EX 的来源, 源代码的基本结构, 以及如何编译源代码生成文档。

第二章 讲述在 L^AT_EX 中如何书写文字, 包括中文。

第三章 讲述文档排版的基本元素——标题、目录、列表、图片、表格等等。结合前一章的内容, 你应当能够制作内容较为丰富的文档了。

第四章 L^AT_EX 排版公式的能力是众人皆知的。本章的内容涉及了一些排版公式经常用到的命令、环境和符号。章节末尾列出了 L^AT_EX 常见的数学符号。

第五章 介绍了如何修改文档的一些基本样式, 包括字体、段落、页面尺寸、页眉页脚等。

第六章 介绍了 L^AT_EX 的一些扩展功能: 排版参考文献、排版索引、排版带有颜色和超链接的电子文档。

第七章 介绍了如何在 L^AT_EX 里使用 T_iKZ 绘图。作为入门手册, 这一部分点到为止。

第八章 当你相当熟悉前面几章的内容, 需要自己编写命令和宏包扩展 L^AT_EX 的功能时, 本章介绍了一些基本的命令满足你的需求。

附录 A 介绍了如何安装 T_EX 发行版和更新宏包。

附录 B 当新手遇到错误和需要寻求更多帮助时, 本章提供了一些基本的参考。

这些章节是循序渐进的, 建议刚刚熟悉 L^AT_EX 的读者按顺序阅读。一定要认真阅读例子的源代码, 它们贯穿全篇手册, 包含了很多的信息。

如果你已经对 L^AT_EX 较为熟练, 本手册的资源已不足够解决你的问题时, 请访问“Comprehensive T_EX Archive Network”(CTAN) 站点, 主页是 <https://www.ctan.org>。所有的宏包也可以从 <https://mirrors.ctan.org> 和遍布全球的各个镜像站点中获得。

在本书中你会找到其他引用 CTAN 的地方, 形式为 CTAN:// 和之后的树状结构。引用本身是一个超链接, 点击后将打开内容在 CTAN 上相应位置的页面。

要在自己的电脑上安装 T_EX 发行版, 请参考附录 A 中的内容。各个操作系统下的 T_EX 发行版位于 [CTAN://systems](https://www.ctan.org/systems)。

如果你有意在这份文档中增加、删除或者改变一些内容，请通知作者。作者对 L^AT_EX 初学者的反馈特别感兴趣，尤其是关于这份介绍哪些内容很容易理解，哪些内容可能需要更好地解释，而哪些内容由于太过难以理解、非常不常用而不适宜放在本手册。

CT_EX 开发小组

<https://github.com/CTeX-org>

lshort 的最新中文版本位于 [CTAN://info/lshort/chinese](http://ctan.org/info/lshort/chinese)。如果用户对其他语言的版本感兴趣，请浏览 [CTAN://info/lshort](http://ctan.org/info/lshort)。

目录

中文版致谢	i	2.3.6 拉丁文扩展与重音	14
英文版致谢	iii	2.3.7 其它符号	15
前言	v	2.3.8 L ^A T _E X 标志	15
目录	vii	2.4 断行和断页	15
		2.4.1 单词间距	15
		2.4.2 手动断行和断页	16
		2.4.3 断词	17
第一章 L ^A T _E X 的基本概念	1	第三章 文档元素	19
1.1 概述	1	3.1 章节和目录	19
1.1.1 T _E X	1	3.1.1 章节标题	19
1.1.2 L ^A T _E X	1	3.1.2 目录	20
1.1.3 L ^A T _E X 的优缺点	1	3.1.3 文档结构的划分	20
1.1.4 命令行基础	2	3.2 标题页	20
1.2 第一次使用 L ^A T _E X	3	3.3 交叉引用	22
1.3 L ^A T _E X 命令和代码结构	4	3.4 脚注和边注	22
1.3.1 L ^A T _E X 命令和环境	4	3.5 特殊环境	23
1.3.2 L ^A T _E X 源代码结构	5	3.5.1 列表	23
1.4 L ^A T _E X 宏包和文档类	5	3.5.2 对齐环境	25
1.4.1 文档类	5	3.5.3 引用环境	25
1.4.2 宏包	6	3.5.4 摘要环境	26
1.5 L ^A T _E X 用到的文件一览	7	3.5.5 代码环境	26
1.6 文件的组织方式	7	3.6 表格	27
1.7 L ^A T _E X 和 T _E X 相关的术语和概念	8	3.6.1 列格式	27
第二章 用 L ^A T _E X 排版文字	11	3.6.2 列宽	29
2.1 语言文字和编码	11	3.6.3 横线	29
2.1.1 ASCII 编码	11	3.6.4 合并单元格	30
2.1.2 扩展编码	11	3.6.5 嵌套表格	31
2.1.3 UTF-8 编码	12	3.6.6 行距控制	31
2.2 排版中文	12	3.7 图片	32
2.3 L ^A T _E X 中的字符	13	3.8 盒子	33
2.3.1 空格和分段	13	3.8.1 水平盒子	33
2.3.2 注释	13	3.8.2 带框的水平盒子	33
2.3.3 特殊字符	13	3.8.3 垂直盒子	34
2.3.4 连字	13	3.8.4 标尺盒子	34
2.3.5 标点符号	14	3.9 浮动体	35

3.9.1	浮动体的标题	35	5.1.6	在 ctex 宏包或文档类中 更改中文字体	65
3.9.2	并排和子图表	36	5.1.7	使用 unicode-math 宏包 配置 Unicode 数学字体	65
第四章	排版数学公式	39	5.2	文字装饰和强调	65
4.1	\mathcal{AMS} 宏集	39	5.3	段落格式和间距	66
4.2	公式排版基础	39	5.3.1	长度和长度变量	66
4.2.1	行内和行间公式	39	5.3.2	行距	67
4.2.2	数学模式	40	5.3.3	段落格式	68
4.3	数学符号	41	5.3.4	水平间距	68
4.3.1	一般符号	41	5.3.5	垂直间距	69
4.3.2	指数、上下标和导数	41	5.4	页面和分栏	69
4.3.3	分式和根式	42	5.4.1	利用 geometry 宏包设置 页面参数	71
4.3.4	关系符	42	5.4.2	页面内容的垂直对齐	72
4.3.5	算符	42	5.4.3	分栏	72
4.3.6	巨算符	43	5.5	页眉页脚	72
4.3.7	数学重音和上下括号	44	5.5.1	基本的页眉页脚样式	72
4.3.8	箭头	45	5.5.2	手动更改页眉页脚的内容	73
4.3.9	括号和定界符	45	5.5.3	fancyhdr 宏包	74
4.4	多行公式	46	第六章	特色工具和功能	75
4.4.1	长公式折行	46	6.1	参考文献和 BibTeX 工具	75
4.4.2	多行公式	46	6.1.1	基本的参考文献和引用	75
4.4.3	公用编号的多行公式	47	6.1.2	BibTeX 数据库	76
4.5	数组和矩阵	47	6.1.3	BibTeX 样式	77
4.6	公式中的间距	48	6.1.4	使用 BibTeX 排版参考文献	77
4.7	数学符号的字体控制	49	6.1.5	natbib 宏包	78
4.7.1	数学字母字体	49	6.1.6	biblatex 宏包	79
4.7.2	加粗的数学符号	50	6.2	索引和 makeindex 工具	81
4.7.3	数学符号的尺寸	50	6.2.1	使用 makeindex 工具的 方法	81
4.8	定理环境	51	6.2.2	索引项的写法	82
4.8.1	L ^A T _E X 原始的定理环境	51	6.3	使用颜色	82
4.8.2	amsthm 宏包	51	6.3.1	颜色的表达方式	83
4.8.3	证明环境和证毕符号	52	6.3.2	带颜色的文本和盒子	84
4.9	符号表	54	6.4	使用超链接	84
4.9.1	L ^A T _E X 普通符号	54	6.4.1	hyperref 宏包	84
4.9.2	\mathcal{AMS} 符号	58	6.4.2	超链接	85
第五章	排版样式设定	61	6.4.3	PDF 书签	86
5.1	字体和字号	61	6.4.4	PDF 文档属性	86
5.1.1	字体样式	61	第七章	绘图功能	87
5.1.2	字号	61	7.1	绘图语言简介	87
5.1.3	选用字体宏包	63	7.2	TikZ 绘图语言	87
5.1.4	字体编码	63			
5.1.5	使用 fontspec 宏包更改 字体	63			

7.2.1	TikZ 坐标和路径	88	8.4	L ^A T _E X 可定制的一些命令和参数	101
7.2.2	TikZ 绘图命令和参数	90			
7.2.3	TikZ 文字结点	92	附录 A 安装 T_EX 发行版		103
7.2.4	在 TikZ 中使用循环	94	A.1	T _E X 发行版简介	103
			A.1.1	安装发行版	103
第八章	自定义 L^AT_EX 命令和功能	95	A.2	安装和更新宏包	104
8.1	自定义命令和环境	95	A.2.1	手动安装宏包	105
8.1.1	定义新命令	95	附录 B 排除错误、寻求帮助		107
8.1.2	定义环境	96	B.1	L ^A T _E X 错误	107
8.1.3	xparse 宏包简介	96	B.2	查找帮助文档	109
8.2	编写自己的宏包和文档类	98	B.3	常用宏包简介	110
8.2.1	编写简单的宏包	98	B.3.1	文字、公式和符号	110
8.2.2	在宏包中调用其它宏包	99	B.3.2	排版元素	110
8.2.3	编写自己的文档类	99	B.3.3	图表和浮动体	111
8.3	计数器	99	B.3.4	修改版式	111
8.3.1	定义和修改计数器	99	参考文献		113
8.3.2	计数器的输出格式	100	GNU Free Documentation License		115
8.3.3	L ^A T _E X 中的计数器	100			

源代码示例列表

1.1	L ^A T _E X 的一个最简单的源代码示例。	3
1.2	在 L ^A T _E X 中排版中文的最简源代码示例。	3
3.1	book 文档类的文档结构示例。	21
3.2	L ^A T _E X 默认的标题页示例和效果。	21
5.1	fancyhdr 宏包的使用方法示例。	74
6.1	利用 books.bib 生成参考文献的源代码 demo.tex。	78
6.2	应用 biblatex 的示例 egbibdata.bib 和 demo.tex。	80
7.1	TikZ 绘图示例源代码和效果。	94
8.1	宏包的一个最简示例。	99

第一章 L^AT_EX 的基本概念

欢迎使用 L^AT_EX! 本章开头用简短的篇幅介绍了 L^AT_EX 的来源, 然后介绍了 L^AT_EX 源代码的写法, 编译 L^AT_EX 源代码生成文档的方法, 以及理解接下来的章节的一些必要知识。

1.1 概述

1.1.1 T_EX

T_EX 是高德纳 (Donald E. Knuth) 为排版文字和数学公式而开发的软件 [6]。1977 年, 正在编写《计算机程序设计艺术》的高德纳意识到每况愈下的排版质量将影响其著作的发行, 为扭转这种状况, 他着手开发 T_EX, 发掘当时刚刚用于出版工业的数字印刷设备的潜力。1982 年, 高德纳发布 T_EX 排版引擎, 而后在 1989 年又为更好地支持 8-bit 字符和多语言排版而予以改进。T_EX 以其卓越的稳定性、跨平台能力和几乎没有 bug 的特性而著称。它的版本号不断趋近于 π , 当前为 3.141592653。

T_EX 读作“Tech”, 与汉字“泰赫”的发音相近, 其中“ch”的发音类似于“h”。T_EX 的拼写来自希腊词语 τεχνική (technique, 技术) 开头的几个字母, 在 ASCII 字符环境中写作 TeX。

1.1.2 L^AT_EX

L^AT_EX 是一种使用 T_EX 程序作为排版引擎的格式 (format), 可以粗略地将它理解成是对 T_EX 的一层封装。L^AT_EX 最初的设计目标是分离内容与格式, 以便作者能够专注于内容创作而非版式设计, 并能以此得到高质量排版的作品。L^AT_EX 起初由 Leslie Lamport 博士 [1] 开发, 目前由 L^AT_EX 工作组¹进行维护。

L^AT_EX 读作“Lah-tech”或者“Lay-tech”, 与汉字“拉泰赫”或“雷泰赫”的发音相近, 在 ASCII 字符环境写作 LaTeX。L^AT_EX 2_ε 是 L^AT_EX 的当前版本, 意思是超出了第二版, 但还远未达到第三版, 在 ASCII 字符环境写作 LaTeX2_ε。

1.1.3 L^AT_EX 的优缺点

经常有人喜欢对比 L^AT_EX 和以 Microsoft Office Word 为代表的“所见即所得” (What You See Is What You Get) 字处理工具。这种对比是没有意义的, 因为 T_EX 是一个排版引擎, L^AT_EX 是其封装, 而 Word 是字处理工具。二者的设计目标不一致, 也各自有自己的适用范围。

不过, 这里仍旧总结 L^AT_EX 的一些优点:

- 具有专业的排版输出能力, 产生的文档看上去就像“印刷品”一样。
- 具有方便而强大的数学公式排版能力, 无出其右者。

¹<https://www.latex-project.org>

- 绝大多数时候，用户只需专注于一些组织文档结构的基础命令，无需（或很少）操心文档的版面设计。
- 很容易生成复杂的专业排版元素，如脚注、交叉引用、参考文献、目录等。
- 强大的可扩展性。世界各地的人开发了数以千计的 L^AT_EX 宏包用于补充和扩展 L^AT_EX 的功能。一些常用宏包列在了本手册附录中的 B.3 小节。更多的宏包参考 *The L^AT_EX companion*[2]。
- 能够促使用户写出结构良好的文档——而这也是 L^AT_EX 存在的初衷。
- L^AT_EX 和 T_EX 及相关软件是跨平台、免费、开源的。无论用户使用的是 Windows, macOS (OS X), GNU/Linux 还是 FreeBSD 等操作系统，都能轻松获得和使用这一强大的排版工具，并且获得稳定的输出。

L^AT_EX 的缺点也是显而易见的：

- 入门门槛高。本手册的副标题叫做“111 分钟了解 L^AT_EX 2_ε”，实际上“111”是本手册正文部分（包括附录）的页数。如果真的以平均一页一分钟的速度看完了本手册，你只是粗窥门径而已，离学会它还很远。
- 不容易排查错误。L^AT_EX 作为一个依靠编写代码工作的排版工具，其使用的宏语言比 C++ 或 Python 等程序设计语言在错误排查方面困难得多。它虽然能够提示错误，但不提供调试的机制，有时错误提示还很难理解。
- 不容易定制样式。L^AT_EX 提供了一个基本上良好的样式，为了让用户不去关注样式而专注于文档结构。但如果想要改进 L^AT_EX 生成的文档样式则是十分困难的。
- 相比“所见即所得”的模式有一些不便，为了查看生成文档的效果，用户总要不停地编译。

1.1.4 命令行基础

L^AT_EX 和 T_EX 及相关软件大多仅提供了命令行接口，而不像 Word、Adobe InDesign 一样有图形用户界面。命令程序的结构往往比较简单，它们接受用户输入，读取相关文件，进行一些操作和运算后输出目标文件，有时还会将提示信息、运行结果显示在屏幕上。在 Windows 系统上，如需进入命令行，可在开始菜单中搜索“命令提示符”，也可在“运行”窗口中输入 cmd 打开；Linux 或 macOS 等 *nix² 系统中可搜索“Terminal”打开终端。部分系统也提供了一些快捷方式，具体请参考相关手册。

与常规软件类似，命令程序也都是可执行程序，在 Windows 上后缀名为 .exe，而在类 Unix 系统上则需要带有 x 权限。在大多数命令行环境中，系统会根据环境变量 PATH 中存储的路径来搜索可供执行的程序。因此在运行之前，需确保 L^AT_EX、T_EX 及相关程序所在路径已包含在 PATH 中。

在命令行中运行程序时，需要先输入程序名，其后可加一系列用空格分隔的参数，并按下 Enter 键执行。一般情况下，命令程序执行完毕会自行退出。若遇到错误或中断，可输入 Ctrl+C 以强制结束。

使用命令程序输入、输出文件时，需确保文件路径正确。通常需要先切换到文件所在目录，再执行有关程序。切换路径可以执行

```
cd <path>
```

²类 Unix 操作系统，包含 Linux、macOS (OS X)。

注意 $\langle path \rangle$ 中的多级目录在 Windows 系统上使用反斜线 \ 分隔，而在类 Unix 系统上使用正斜线 / 分隔。如果 $\langle path \rangle$ 中带有空格，则需加上引号 "。此外，在 Windows 系统上如果要切换到其他分区，还需加上 /d 选项，例如 `cd /d "C:\Program Files (x86)\"`。

许多用户会使用 TeXworks 或 TeXstudio 等编辑器来编写 L^AT_EX 文档。这些编辑器提供的编译功能，实际上只是对特定命令程序的封装，而并非魔法。

1.2 第一次使用 L^AT_EX

源代码 1.1 是一份最短的 L^AT_EX 源代码示例。

```
\documentclass{article}
\begin{document}
``Hello world!'' from \LaTeX.
\end{document}
```

源代码 1.1: L^AT_EX 的一个最简单的源代码示例。

这里首先介绍如何编译使用这份源代码，在后续小节中再介绍源代码的细节。你可以将这份源代码保存为 `helloworld.tex`，而后编译。具体来说：

- 如果使用 TeXworks 或 TeXstudio 等编辑器，你可以使用编辑器提供的“编译”按钮或者“排版”按钮。建议使用 pdfL^AT_EX 或 X_ƳL^AT_EX 作为默认的编译方式（不同编译方式的差别，见 1.7 节）。
- 如果使用命令行方式进行编译，则需打开 Windows 命令提示符或者 *nix 的终端，在源代码所在的目录下输入：

```
pdflatex helloworld
```

或者

```
xelatex helloworld
```

如果编译成功，可以在 `helloworld.tex` 所在目录看到生成的 `helloworld.pdf` 以及一些其它文件。

源代码 1.2 是在 L^AT_EX 排版中文的一个最简示例。编译的方式与上一份源代码相同，但需使用 X_ƳL^AT_EX 编译方式³。中文支持的详细内容见 2.2 节。

```
\documentclass{ctexart}
\begin{document}
“你好，世界！”来自 \LaTeX{} 的问候。
\end{document}
```

源代码 1.2: 在 L^AT_EX 中排版中文的最简源代码示例。

³注意两个问题：1. 文档保存为 UTF-8 编码；2. 某些发行版需要补充安装较多宏包（如 MiK_TE_X）。

1.3 L^AT_EX 命令和代码结构

L^AT_EX 的源代码为文本文件。这些文本除了文字本身，还包括各种命令，用在排版公式、划分文档结构、控制样式等等不同的地方。

1.3.1 L^AT_EX 命令和环境

L^AT_EX 中命令⁴以反斜线 \ 开头，为以下两种形式之一：

- 反斜线和后面的一串字母，如 \LaTeX。它们以任意非字母符号（空格、数字、标点等）为界限。
- 反斜线和后面的单个非字母符号，如 \\$。

要注意 L^AT_EX 命令是**对大小写敏感**的，比如输入 \LaTeX 命令可以生成错落有致的 L^AT_EX 字母组合，但输入 \Latex 或者 \LaTeX 什么都得不到，还会报错；它们与 \LaTeX 是不同的命令。

字母形式的 L^AT_EX 命令忽略其后的所有连续空格。如果要人为引入空格，需要在命令后面加一对花括号阻止其忽略空格⁵：

```
Shall we call ourselves
\TeX users

or \TeX{} users?
```

```
Shall we call ourselves TEXusers
or TEX users?
```

一些 L^AT_EX 命令可以接收一些参数，参数的内容会影响命令的效果。L^AT_EX 的参数分为可选参数和必选参数。可选参数以方括号 [和] 包裹；必选参数一般以花括号 { 和 } 包裹⁶。还有些命令可以带一个星号 *，带星号和不带星号的命令效果有一定差异。初次接触这些概念时，可以粗略地把星号看作一种特殊的可选参数。

L^AT_EX 中还包括**环境**，用以令一些效果在局部生效，或是生成特殊的文档元素。L^AT_EX 环境的用法为一对命令 \begin 和 \end：

```
\begin{environment name}[optional arguments]{mandatory arguments}
...
\end{environment name}
```

其中 *environment name* 为环境名，\begin 和 \end 中填写的环境名应当一致。类似命令，{*mandatory arguments*} 和 [*optional arguments*] 为环境所需的必选和可选参数。L^AT_EX 环境可能需要一个或多个必选/可选参数，也可能完全不需要参数。部分环境允许嵌套使用。

有些命令（如 \bfseries）会对其后所有内容产生作用。若要限制其作用范围，则需要使用**分组**。L^AT_EX 使用一对花括号 { 和 } 作为分组，在分组中使用的命令被限制在分组内，不会影响到分组外的内容⁷。上文提到的 L^AT_EX 环境隐含了一个分组，在环境中的命令被包裹在分组内。[5.1.1](#) 和 [5.1.2](#) 小节中介绍的修改字体和字号的命令用法，即属此类。

⁴也叫作控制序列（control sequence）。

⁵另外也可以在命令后面紧跟一个 _ 命令（反斜线加空格），代表插入一个间距。比如 \TeX_user 的输出效果就是 T_EX user。

⁶以单个字符作为命令的参数时，可以不加括号。例如，在数学环境下，\frac{1}{2} 和 \frac{1}{2} 的效果是一样的。

⁷个别命令在分组内仍然会产生全局作用，例如第 [8.3](#) 节介绍的 \setcounter 等命令。

1.3.2 L^AT_EX 源代码结构

L^AT_EX 源代码以一个 `\documentclass` 命令作为开头，它指定了文档使用的**文档类**。`document` 环境当中的内容是文档正文。

在 `\documentclass` 和 `\begin{document}` 之间的位置称为**导言区**。在导言区中常会使用 `\usepackage` 命令调用**宏包**，还会进行文档的全局设置。

```
\documentclass{...} % ... 为某文档类
% 导言区
\begin{document}
% 正文内容
\end{document}
% 此后内容会被忽略
```

1.4 L^AT_EX 宏包和文档类

本节将仔细解释在 1.3.2 小节中出现的宏包和文档类的概念以及详细用法。

1.4.1 文档类

文档类规定了 L^AT_EX 源代码所要生成的文档的性质——普通文章、书籍、演示文稿、个人简历等等。L^AT_EX 源代码的开头须用 `\documentclass` 指定文档类：

```
\documentclass[<options>]{<class-name>}
```

其中 *<class-name>* 为文档类的名称，如 L^AT_EX 提供的 `article`、`report`、`book`，在其基础上派生的一些文档类，如支持中文排版的 `ctexart`、`ctexrep`、`ctexbook`，或者有其它功能的一些文档类，如 `moderncv`、`beamer` 等。L^AT_EX 提供的基础文档类见表 1.1，其中前三个习惯上称为“标准文档类”。

表 1.1: L^AT_EX 提供的基础文档类

<code>article</code>	文章格式的文档类，广泛用于科技论文、报告、说明文档等。
<code>report</code>	长篇报告格式的文档类，具有章节结构，用于综述、长篇论文、简单的书籍等。
<code>book</code>	书籍文档类，包含章节结构和前言、正文、后记等结构。
<code>proc</code>	基于 <code>article</code> 文档类的一个简单的学术文档模板。
<code>slides</code>	幻灯格式的文档类，使用无衬线字体。
<code>minimal</code>	一个极其精简的文档类，只设定了纸张大小和基本字号，用作代码测试的最小工作示例 (Minimal Working Example)。

可选参数 *<options>* 为文档类指定选项，以全局地规定一些排版的参数，如字号、纸张大小、单双面等等。比如调用 `article` 文档类排版文章，指定纸张为 A4 大小，基本字号为 11pt，双面排版：

```
\documentclass[11pt,twoside,a4paper]{article}
```

L^AT_EX 的三个标准文档类可指定的选项包括：

10pt, 11pt, 12pt 指定文档的基本字号。默认为 10pt。

a4paper, letterpaper, ... 指定纸张大小，默认为美式信纸 letterpaper (8.5in × 11in, 大约相当于 21.6cm × 28.0cm)。可指定选项还包括 a5paper, b5paper, executivepaper 和 legalpaper。有关纸张大小的更多细节，请参考 5.4.1 小节。

twoside, oneside 指定单面/双面排版。双面排版时，奇偶页的页眉页脚、页边距不同。article 和 report 默认为 oneside，book 默认为 twoside。

onecolumn, twocolumn 指定单栏/双栏排版。默认为 onecolumn。

openright, openany 指定新的一章 \chapter 是在奇数页（右侧）开始，还是直接紧跟着上一页开始。report 默认为 openany，book 默认为 openright。对 article 无效。

landscape 指定横向排版。默认为纵向。

titlepage, notitlepage 指定标题命令 \maketitle 是否生成单独的标题页。article 默认为 notitlepage，report 和 book 默认为 titlepage。

fleqn 令行间公式左对齐。默认为居中对齐。

leqno 将公式编号放在左边。默认为右边。

draft, final 指定草稿/终稿模式。草稿模式下，断行不良（溢出）的地方会在行尾添加一个黑色方块；插图、超链接等功能也会受这一组选项影响，具体见后文。默认为 final。

1.4.2 宏包

在使用 L^AT_EX 时，时常需要依赖一些扩展来增强或补充 L^AT_EX 的功能，比如排版复杂的表格、插入图片、增加颜色甚至超链接等等。这些扩展称为**宏包**。调用宏包的方法非常类似调用文档类的方法：

```
\usepackage[<options>]{<package-name>}
```

\usepackage 可以一次性调用多个宏包，在 *<package-name>* 中用逗号隔开。这种用法一般不要指定选项⁸：

```
% 一次性调用三个排版表格常用的宏包
\usepackage{tabularx, makecell, multirow}
```

附录 B.3 汇总了常用的一些宏包。我们在手册接下来的章节中，也会穿插介绍一些最常用的宏包的使用方法。

在使用宏包和文档类之前，一定要首先确认它们是否安装在你的计算机中，否则 \usepackage 等命令会报错误。详见附录 A.2。

宏包（包括前面所说的文档类）可能定义了许多命令和环境，或者修改了 L^AT_EX 已有的命令和环境。它们的用法说明记在相应宏包和文档类的帮助文档。在 Windows 命令提示符或者 Linux 终端下输入命令可查阅相应文档：

```
texdoc <pkg-name>
```

其中 *<pkg-name>* 是宏包或者文档类的名称。更多获得帮助的方法见附录 B.2。

⁸使用多个宏包时指定选项，相当于给每个宏包指定同样的选项。如果有某个宏包不能识别指定的选项，则会出错。

1.5 L^AT_EX 用到的文件一览

除了源代码文件 `.tex` 以外，我们在使用 L^AT_EX 时还可能接触到各种格式的文件。本节简单介绍一下在使用 L^AT_EX 时能够经常见到的文件。

每个宏包和文档类都是带特定扩展名的文件，除此之外也有一些文件出现于 L^AT_EX 模板中：

`.sty` 宏包文件。宏包的名称与文件名一致。

`.cls` 文档类文件。文档类名称与文件名一致。

`.bib` Bib_TE_X 参考文献数据库文件。

`.bst` Bib_TE_X 用到的参考文献格式模板。详见 6.1.4 小节。

L^AT_EX 在编译过程中除了生成 `.dvi` 或 `.pdf` 格式的文档外⁹，还可能会生成相当多的辅助文件和日志。一些功能如交叉引用、参考文献、目录、索引等，需要先通过编译生成辅助文件，然后再次编译时读入辅助文件得到正确的结果，所以复杂的 L^AT_EX 源代码可能要编译多次：

`.log` 排版引擎生成的日志文件，供排查错误使用。

`.aux` L^AT_EX 生成的主辅助文件，记录交叉引用、目录、参考文献的引用等。

`.toc` L^AT_EX 生成的目录记录文件。

`.lof` L^AT_EX 生成的图片目录记录文件。

`.lot` L^AT_EX 生成的表格目录记录文件。

`.bbl` Bib_TE_X 生成的参考文献记录文件。

`.blg` Bib_TE_X 生成的日志文件。

`.idx` L^AT_EX 生成的供 `makeindex` 处理的索引记录文件。

`.ind` `makeindex` 处理 `.idx` 生成的用于排版的格式化索引文件。

`.ilg` `makeindex` 生成的日志文件。

`.out` `hyperref` 宏包生成的 PDF 书签记录文件。

1.6 文件的组织方式

当编写长篇文档时，例如当编写书籍、毕业论文时，单个源文件会使修改、校对变得十分困难。将源文件分割成若干个文件，例如将每章内容单独写在一个文件中，会大大简化修改和校对的工作。可参考源代码 3.1 的写法。

L^AT_EX 提供了命令 `\include` 用来在源代码里插入文件：

```
\include{<filename>}
```

`<filename>` 为文件名（不带 `.tex` 扩展名）¹⁰，如果和要编译的主文件不在一个目录中，则要加上相对或绝对路径，例如：

⁹X_YL^AT_EX 还可能生成 `.xdv` 文件。

¹⁰L^AT_EX 2020-10-01 版本之后允许添加扩展名。

```
\include{chapters/file} % 相对路径
\include{/home/Bob/file} % *nix (包含 Linux、macOS) 绝对路径
\include{D:/file} % Windows 绝对路径, 用正斜线
```

值得注意的是 `\include` 在读入 $\langle filename \rangle$ 之前会另起一页。有的时候我们并不需要这样, 而是用 `\input` 命令, 它纯粹是把文件里的内容插入:

```
\input{\langle filename \rangle}
```

当导言区内容较多时, 常常将其单独放置在一个 `.tex` 文件中, 再用 `\input` 命令插入。复杂的图、表、代码等也会用类似的手段处理。

L^AT_EX 还提供了一个 `\includeonly` 命令来组织文件, 用于导言区, 指定只载入某些文件。导言区使用了 `\includeonly` 后, 正文中不在其列表范围的 `\include` 命令不会起效:

```
\includeonly{\langle filename1 \rangle, \langle filename2 \rangle, ...}
```

需要注意的是, 使用 `\include` 和 `\input` 命令载入的文件名最好不要加空格和特殊字符, 也尽量避免使用中文名, 否则很可能会出错¹¹。

最后介绍一个实用的工具宏包 `syntonly`。加载这个宏包后, 在导言区使用 `\syntonly` 命令, 可令 L^AT_EX 编译后不生成 DVI 或者 PDF 文档, 只排查错误, 编译速度会快不少:

```
\usepackage{syntonly}
\syntonly
```

如果想生成文档, 则用 `%` 注释掉 `\syntonly` 命令即可。

1.7 L^AT_EX 和 T_EX 相关的术语和概念

在本章的最后有必要澄清几个概念:

引擎 全称为排版引擎, 是编译源代码并生成文档的程序, 如 pdfT_EX、X_YT_EX 等。有时也称为编译器。

格式 是定义了一组命令的代码集。L^AT_EX 就是最广泛应用的一个格式, 高德纳本人还编写了一个简单的 plain T_EX 格式, 没有定义诸如 `\documentclass` 和 `\section` 等等命令。

编译命令 是实际调用的、结合了引擎和格式的命令。如 `xelatex` 命令是结合 X_YT_EX 引擎和 L^AT_EX 格式的一个编译命令。

常见的引擎、格式和编译命令的关系总结于表 1.2。

`latex` 编译命令和 L^AT_EX 格式往往容易混淆, 在讨论关于 L^AT_EX 的时候需要明确。为避免混淆, 本手册中的 L^AT_EX 一律指的是**格式**, **编译命令**则用等宽字体 `latex` 表示。

在此介绍一下几个编译命令的基本特点:

latex 虽然名为 `latex` 命令, 底层调用的引擎其实是 pdfT_EX。该命令生成 dvi (Device Independent) 格式的文档, 用 `dvipdfmx` 命令可以将其转为 pdf。

pdflatex 底层调用的引擎也是 pdfT_EX, 可以直接生成 pdf 格式的文档。

¹¹L^AT_EX 在 2019 到 2020 年间做出了一系列更新, 文件名中包含操作系统所允许的字符时均不再出错。但保险起见, 仍然建议使用拉丁字母、数字、连字符等较为安全的字符组成文件名。

表 1.2: T_EX 引擎、格式及其对应的编译命令。

	文档格式	plain T _E X 格式	L ^A T _E X 格式
T _E X 引擎	DVI	tex	N/A
pdfT _E X 引擎	DVI	etex	latex
	PDF	pdftex	pdflatex
X _Y T _E X 引擎	PDF	xetex	xelatex
LuaT _E X 引擎	PDF	luatex	lualatex

xelatex 底层调用的引擎是 X_YT_EX，支持 UTF-8 编码和对 TrueType/OpenType 字体的调用。当前较为方便的中文排版解决方案基于 **xelatex**，详见 2.2 节。

lualatex 底层调用的引擎是 LuaT_EX¹²，这个引擎在 pdfT_EX 引擎基础上发展而来，除了支持 UTF-8 编码和对 TrueType/OpenType 字体的调用外，还支持通过 Lua 语言扩展 T_EX 的功能。**lualatex** 编译命令下的中文排版支持需要借助 **luatexja** 宏包。

¹²从 T_EX Live 2020 开始，**lualatex** 调用的引擎变为 LuaHB_T_EX。LuaHB_T_EX 是 LuaT_EX 与 HarfBuzz 库的结合，对复杂语言的支持更好。

第二章 用 L^AT_EX 排版文字

文字是排版的基础。本章主要介绍如何在 L^AT_EX 中输入各种文字符号，包括标点符号、连字符、重音等，以及控制文字断行和断页的方式。

本章简要介绍了在 L^AT_EX 中排版中文的方法。随着 L^AT_EX 和底层 T_EX 引擎的发展，旧方式（CCT、CJK 等）日渐退出舞台，x_el_atex 和 l_ua_la_te_x 编译命令配合 ct_ex 宏包/文档类的方式成为当前的主流中文排版支持方式。

2.1 语言文字和编码

L^AT_EX 源代码为文本文件，而文本文件的一个至关重要的性质是它的编码。在此用尽量短的篇幅介绍一下。

2.1.1 ASCII 编码

计算机的基本存储单位是字节（byte），每个字节为八位（8-bit），范围用十六进制写作 0x00–0xFF。ASCII（美国通用信息交换码）使用 0x00–0x7F 对文字编码，也就是 7-bit，覆盖了基本的拉丁字母、数字和符号，以及一些不可打印的控制字符（如换行符、制表符等）。

由于 T_EX 最初设计用于排版以英文为主的西文文档，ASCII 编码完全够用，因而早期版本的 T_EX 只支持 7-bit 和 ASCII 编码。排版扩展拉丁字符必须使用后文所述的各种符号和重音命令，如 Möbius 必须通过输入 M\ "obius 得到。

2.1.2 扩展编码

在 ASCII 之后，各种语言文字都发展了自己的编码，比如西欧语言的 Latin-1、日本的 Shift-JIS、中国大陆的 GB 2312—80 和 GBK 等。它们中的绝大多数都向下兼容 ASCII，因此无论是在哪种编码下，T_EX 以及 L^AT_EX 的命令和符号都能用。

T_EX 从 3.0 版开始支持 8-bit，能够处理编码在 0x80–0xFF 范围内的字符。西欧（拉丁字母）、俄语系（西里尔字母）等语言文字的编码方案大都利用了 0x80–0xFF 这个范围，处理起来较为方便。使用 latex 或 pdf_lat_ex 编译命令时，对源代码的编码处理由 inputenc 宏包支持。比如将源代码保存为 Latin-1 编码，并在导言区调用 inputenc 宏包并指定 latin1 选项后，Möbius 这样的词语就可以直接通过（用适当输入法）输入 Möbius 得到了。

用于汉字的 GBK 等编码是多字节编码，ASCII 字符为一个字节，汉字等非 ASCII 字符为两个字节，使用 latex 或 pdf_lat_ex 编译命令时需要借助一些宏包进行较为复杂的判断和处理。早期排版中文须使用 CJK 宏包，它是一个用于处理中、日、韩等东亚语言文字编码和字体配置的宏包。但 CJK 宏包的使用非常不方便，目前已不再推荐直接使用。

2.1.3 UTF-8 编码

Unicode 是一个多国字符的集合，覆盖了几近全球范围内的语言文字。UTF-8 是 Unicode 的一套编码方案，一个字符由一个到四个字节编码，其中单字节字符的编码与 ASCII 编码兼容。

现行版本的 L^AT_EX 使用 UTF-8 作为默认编码¹。将使用拉丁字母的文档保存为 UTF-8 编码后，可以用 `pdflatex` 直接编译，比如：

```
\documentclass{article}
\begin{document}
Français Portuguais Español Føroyskt
\end{document}
```

但是非拉丁字母仍然无法直接在 L^AT_EX 中使用，如西里尔字母（俄文）、希腊字母、阿拉伯字母以及东亚文字等。

较为现代的 T_EX 引擎，如 X_YT_EX 和 LuaT_EX，它们均原生支持 UTF-8 编码。使用 `xelatex` 和 `lualatex` 排版时，将源代码保存为 UTF-8 编码，并借助 `fontspec` 宏包（见 5.1.5 小节）调用适当的字体，原则上就可以在源代码中输入任意语言的文字。注意此时**不再适用 `inputenc` 宏包**。但一些复杂语言（如印地语、阿拉伯语等）的排版需要考虑到断词规则、文字方向、标点禁则等诸多细节，因此需要更多的宏包支持，如 `babel`、`polyglossia` 等，此处不再涉及。

2.2 排版中文

用 L^AT_EX 排版中文需要解决两方面问题，一方面是对中文字体的支持，另一方面是对中文排版中的一些细节的处理，包括在汉字之间控制断行、标点符号的禁则（如句号、逗号不允许出现在行首）、中英文之间插入间距等。CJK 宏包对中文字体的支持比较麻烦，已经不再推荐直接使用。X_YT_EX 和 LuaT_EX 除了直接支持 UTF-8 编码外，还支持直接调用 TrueType/OpenType 格式的字体。`xeCJK` 及 `luatexja` 宏包则在此基础上封装了对汉字排版细节的处理功能。

`ctex` 宏包和文档类²进一步封装了 CJK、`xeCJK`、`luatexja` 等宏包，使得用户在排版中文时不用再考虑排版引擎等细节。`ctex` 宏包本身用于配合各种文档类排版中文，而 `ctex` 文档类对 L^AT_EX 的标准文档类进行了封装，对一些排版根据中文排版习惯做了调整，包括 `ctexart`、`ctexrep`、`ctexbook` 等。`ctex` 宏包和文档类能够识别操作系统和 T_EX 发行版中安装的中文字体，因此基本无需额外配置即可排版中文文档。下面举一个使用 `ctex` 文档类排版中文的最简例子：

```
\documentclass{ctexart}
\begin{document}
在\LaTeX{}中排版中文。
汉字和English单词混排，通常不需要在中英文之间添加额外的空格。
当然，为了代码的可读性，加上汉字和 English 之间的空格也无妨。
汉字换行时不会引入多余的空格。
\end{document}
```

注意源代码须保存为 UTF-8 编码，并使用 `xelatex` 或 `lualatex` 命令编译。虽然 `ctex` 宏包和文档类保留了对 GBK 编码以及 `latex` 和 `pdflatex` 编译命令的兼容，但我们并不推荐这样做。

¹L^AT_EX 2018-04-01 版本之前，需要调用 `inputenc` 宏包并指定 `utf8` 选项才能使用 UTF-8 编码。

²C_T_EX 还经常用来指一个过时的 T_EX 发行版，注意与这里的 `ctex` 宏包和文档类区分。

2.3 L^AT_EX 中的字符

2.3.1 空格和分段

L^AT_EX 源代码中，空格键和 Tab 键输入的空白字符视为“空格”。连续的若干个空白字符视为一个空格。一行开头的空格忽略不计。

行末的换行符视为一个空格；但连续两个换行符，也就是空行，会将文字分段。多个空行被视为一个空行。也可以在行末使用 `\par` 命令分段。

```
Several spaces      equal one.
  Front spaces are ignored.

An empty line starts a new
paragraph.\par
A \verb|\par| command does
the same.
```

```
Several spaces equal one. Front spaces are
ignored.

An empty line starts a new paragraph.

A \par command does the same.
```

2.3.2 注释

L^AT_EX 用 % 字符作为注释。在这个字符之后直到行末，所有的字符都被忽略，行末的换行符也不引入空格。

```
This is an % short comment
% ---
% Long and organized
% comments
% ---
example: Comments do not bre%
ak a word.
```

```
This is an example:  Comments do not
break a word.
```

2.3.3 特殊字符

以下字符在 L^AT_EX 里有特殊用途，如 % 表示注释，\$, ^, _ 等用于排版数学公式，& 用于排版表格，等等。直接输入这些字符得不到对应的符号，还往往会出错：

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

如果想要输入以上符号，需要使用以下带反斜线形式输入，类似编程语言里的“转义”符号：

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

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

这些“转义”符号事实上是一些 L^AT_EX 命令。其中 `\^` 和 `\~` 两个命令需要一个参数，加一对花括号的写法相当于提供了空的参数，否则它们可能会将后面的字符作为参数，形成重音效果（详见 2.3.6 节）。`\\` 被直接定义成了手动换行的命令，输入反斜线就需要用 `\textbackslash`。

2.3.4 连字

西文排版中经常会出现连字（ligatures），常见的有 ff/fi/fl/ffi/ffl。

```
It's difficult to find \ldots\\
It's dif{f}{f}icult to f{f}ind \ldots
```

```
It's difficult to find ...
It's difficult to find ...
```

2.3.5 标点符号

中文的标点符号（绝大多数为非 ASCII 字符）使用中文输入法输入即可，一般不需要过多留意。而输入西文标点符号时，有不少地方需要留意。

引号

L^AT_EX 中单引号 ‘ 和 ’ 分别用 ``` 和 `'` 输入；双引号 “ 和 ” 分别用 ```` 和 `''` 输入（`"` 可以输入后双引号，但没有直接输入前双引号的字符，习惯上用 `'` 输入以和 ```` 更好地对应）。

```
``Please press the `x' key.``
```

```
“Please press the ‘x’ key.”
```

中文的引号 ‘ ’ 和 “ ” 与西文的引号实际上是同一组符号³，但由于中西文通常用不同的字体显示，它们的具体形状和宽度可能有所不同。在使用 `ctex` 宏包或文档类的情况下，中文引号可以通过输入法直接输入。

连字号和破折号

L^AT_EX 中有三种长度的“横线”可用：连字号（hyphen）、短破折号（en-dash）和长破折号（em-dash）。它们分别有不同的用途：连字号 - 用来组成复合词；短破折号 - 用来连接数字表示范围；长破折号 — 用来连接单词，语义上类似中文的破折号。

```
daughter-in-law, X-rated\\
pages 13--67\\
yes---or no?
```

```
daughter-in-law, X-rated
pages 13-67
yes—or no?
```

省略号

L^AT_EX 提供了 `\ldots` 命令表示省略号，相对于直接输入三个点的方式更为合理。`\dots` 与 `\ldots` 命令等效。

```
one, two, three, \ldots{} one hundred.
```

```
one, two, three, ... one hundred.
```

波浪号

我们在 2.3.3 小节中了解了 `\~` 命令，它可以用来输入波浪号，但位置靠顶端（`\~` 命令主要用作重音，参考下一小节）。西文中较少将波浪号作为标点符号使用，在中文环境中一般直接使用全角波浪号（~）。

2.3.6 拉丁文扩展与重音

L^AT_EX 支持用命令输入西欧语言中使用的各种拉丁文扩展字符，主要为带重音的字母：

```
H\^otel, na\"i ve, \'el\`eve,\\
sm\o rrebr\o d, !`Se\ norita!,\\
Sch\"onbrunner Schlo\ss{}
Stra\ss e
```

```
Hôtel, naïve, élève,
smørrebrød, ¡Se norita!,
Schönbrunner Schloß Straße
```

更多可用的符号和重音见表 2.1。注意与 4.3.7 小节的数学重音区分开来。

³它们在 Unicode 中共用了同一组码位。有人为了明确区分，会改用中文直角引号「」和『』。

表 2.1: L^AT_EX 文本中的重音和特殊字符

ò	\`o	ó	\'o	ô	\^o	õ	\~o
ō	\=o	ô	\.o	ö	\"o	ő	\r o
ǒ	\u o	Ǔ	\v o	ǔ	\H o	ǖ	\c o
ȝ	\d o	Ț	\b o	ôo	\t{oo}		
œ	\oe	Œ	\OE	æ	\ae	Æ	\AE
å	\aa	Å	\AA	ß	\ss		
ø	\o	Ø	\O	ł	\l	Ł	\L
ı	\i	Ј	\j	ı	!`	ı	?`

前四行实际上都是带一个参数的命令。`\`o` 也可以写作 `\`{o}`，以此类推。

2.3.7 其它符号

L^AT_EX 预定义了其它一些文本模式的符号，部分符号可参考表 4.4。

```
\P{} \S{} \dag{} \ddag{}
\copyright{} \pounds{}

\textasteriskcentered
\textperiodcentered
\textbullet

\textregistered{} \texttrademark
```

¶ § † ‡ © £

*•

® ™

更多的符号多由特定的宏包支持。参考文献 [14] 搜集了所有在 T_EX 发行版中可用的符号，使用时要留意每个符号所依赖的宏包。

2.3.8 L^AT_EX 标志

我们见到的所有错落有致的 L^AT_EX 标志都是由以下命令输入的：

T _E X	\TeX
L ^A T _E X	\LaTeX
L ^A T _E X 2 _ε	\LaTeXe

2.4 断行和断页

L^AT_EX 将文字段落在合适的位置进行断行，尽可能做到每行的疏密程度匀称，单词间距不会过宽或过窄。文字段落和公式、图表等内容从上到下顺序排布，并在合适的位置断页，分割成匀称的页面。在绝大多数时候，我们无需自己操心断行和断页。但偶尔会遇到需要手工调整的地方。

2.4.1 单词间距

在西文排版实践中，断行的位置优先选取在两个单词之间，也就是在源代码中输入的“空格”⁴。“空格”本身通常生成一个间距，它会根据行宽和上下文自动调整，文字密一些的地方，

⁴中文排版实现汉字间断行，则需要宏包（如 `xeCJK` 等）或特殊的排版引擎（如 `upLATEX`）的支持。

单词间距就略窄，反之略宽。

文字在单词间的“空格”处断行时，“空格”生成的间距随之舍去。我们可以使用字符 `~` 输入一个不会断行的空格（高德纳称之为 tie，“带子”），通常用在英文人名、图表名称等上下文环境：

```
Fig.~2a \\  
Donald~E. Knuth
```

```
Fig. 2a  
Donald E. Knuth
```

2.4.2 手动断行和断页

如果我们确实需要手动断行，可使用如下命令：

```
\[<length>] \*[<length>]  
\newline
```

它们有两点区别：一是 `\[<length>]` 可以带可选参数 `<length>`，用于在断行处向下增加垂直间距（见 5.3.5 小节），而 `\newline` 不带可选参数；二是 `\[<length>]` 也在表格、公式等地方用于换行，而 `\newline` 只用于文本段落中。带星号的 `\[<length>]` 表示禁止在断行处分页。

另外需要注意的是，使用 `\verb|\\|` 断行命令 `\\` 不会令内容另起一段，而是在段落中直接开始新的一行。

另外需要注意的是，使用 `\\` 断行命令不会令内容另起一段，而是在段落中直接开始新的一行。

断页的命令有两个：

```
\newpage  
\clearpage
```

通常情况下两个命令都起到另起一页的作用，区别在于：第一，在双栏排版模式中 `\newpage` 起到另起一栏的作用，`\clearpage` 则能够另起一页；第二，在涉及浮动体的排版上行为不同。后文的 3.9 节以及 5.4.3 小节会更详细地介绍相关内容。

有时候我们不满于 L^AT_EX 默认的断行和断页位置，需要进行微调，可以用以下命令告诉 L^AT_EX 哪些地方适合断行或断页，哪些地方不适合：

```
\linebreak[<n>] \nolinebreak[<n>]  
\pagebreak[<n>] \nopagebreak[<n>]
```

以上命令都带一个可选参数，用数字 `<n>` 代表适合/不适合的程度，取值范围为 0–4，不带可选参数时，缺省为 4。比如 `\linebreak` 或者 `\linebreak[4]` 意味着此处需要强行断行；`\nopagebreak` 或 `\nopagebreak[4]` 意味着禁止在此处断页。

以上命令适合给出优先考虑断行断页/禁止断行断页的位置，但不适合直接拿来断行或断页，使用 `\newline` 或 `\newpage` 等命令是更好的选择。因为 `\newline` 和 `\newpage` 会在断行/断页位置填充适当的间距，但 `\linebreak` 和 `\pagebreak` 不能，使用这些命令强行断行/断页可能会制造出糟糕的排版效果，并导致 L^AT_EX 报 Underfull \hbox 等警告。

使用 `\verb|\newline|` 断行的效果
`\newline`
与使用 `\verb|\linebreak|` 断行的效果
`\linebreak`
进行对比。

使用 `\newline` 断行的效果
与使用 `\linebreak` 断行的效果
进行对比。

2.4.3 断词

如果 \LaTeX 遇到了很长的英文单词，仅在单词之间的“空格”处断行无法生成疏密程度匀称的段落时，就会考虑从单词中间断开。对于绝大多数单词， \LaTeX 能够找到合适的断词位置，在断开的行尾加上连字符 -。

如果一些单词没能自动断词，我们可以在单词内手动使用 `\-` 命令指定断词的位置：

```
I think this is: su\per\cal\-%  
i\frag\i\lis\tic\ex\pi\-%  
al\i\do\cious.
```

```
I think this is: supercalifragilisticexpialido-  
cious.
```


第三章 文档元素

在知道了如何输入文字后，我们将在本章了解一个结构化的文档所依赖的各种元素——章节、目录、列表、图表、交叉引用、脚注等等。

3.1 章节和目录

3.1.1 章节标题

一篇结构化的、条理清晰文档一定是层次分明的，通过不同的命令分割为章、节、小节。三个标准文档类 `article`、`report` 和 `book`¹ 提供了划分章节的命令：

```
\chapter{<title>} \section{<title>} \subsection{<title>}
\subsubsection{<title>} \paragraph{<title>} \subparagraph{<title>}
```

其中 `\chapter` 只在 **report** 和 **book** 文档类有定义。这些命令生成章节标题，并能够自动编号。除此之外 `LaTeX` 还提供了 `\part` 命令，用来将整个文档分割为大的分块，但不影响 `\chapter` 或 `\section` 等的编号。

上述命令除了生成带编号的标题之外，还向目录中添加条目，并影响页眉页脚的内容（详见 5.5 节）。每个命令有两种变体：

- 带可选参数的变体：`\section[<short title>]{<title>}`
标题使用 `<title>` 参数，在目录和页眉页脚中使用 `<short title>` 参数；
- 带星号的变体：`\section*{<title>}`
标题不带编号，也不生成目录项和页眉页脚。

较低层次如 `\paragraph` 和 `\subparagraph` 即使不用带星号的变体，生成的标题默认也不带编号，事实上，除 `\part` 外：

- `article` 文档类带编号的层级为 `\section`、`\subsection`、`\subsubsection` 三级；
- `report` 和 `book` 文档类带编号的层级为 `\chapter`、`\section`、`\subsection` 三级。

对此的详细解释和调整方法见 8.3.3 小节。

`LaTeX` 及标准文档类并未提供为 `\section` 等章节命令定制格式的功能，这一功能由 `titlesec` 宏包提供。详情请参考宏包的帮助文档。

¹千万注意是**标准文档类**，其它文档类，如果不是从标准文档类衍生而来，很可能没有定义或只定义了一部分命令，如 `beamer` 或 `moderncv` 等。

3.1.2 目录

在 L^AT_EX 中生成目录非常容易，只需在合适的地方使用命令：

```
\tableofcontents
```

这个命令会生成单独的一章 (report/book) 或一节 (article)，标题默认为 “Contents”，可通过 8.4 节给出的方法定制标题。`\tableofcontents` 生成的章节默认不写入目录 (`\section*` 或 `\chapter*`)，可使用 `tocbibind` 等宏包修改设置。

正确生成目录项，一般需要编译两次源代码。

有时我们使用了 `\chapter*` 或 `\section*` 这样不生成目录项的章节标题命令，而又想手动生成该章节的目录项，可以在标题命令后面使用：

```
\addcontentsline{toc}{\langle level \rangle}{\langle title \rangle}
```

其中 `\langle level \rangle` 为章节层次 `chapter` 或 `section` 等，`\langle title \rangle` 为出现于目录项的章节标题。

`titletoc`、`tocloft` 等宏包提供了具体定制目录项格式的功能，详情请参考宏包的帮助文档。

3.1.3 文档结构的划分

所有标准文档类都提供了一个 `\appendix` 命令将正文和附录分开²，使用 `\appendix` 后，最高一级章节改为使用拉丁字母编号，从 A 开始。

`book` 文档类还提供了前言、正文、后记结构的划分命令：

`\frontmatter` 前言部分，页码使用小写罗马数字；其后的 `\chapter` 不编号。

`\mainmatter` 正文部分，页码使用阿拉伯数字，从 1 开始计数；其后的章节编号正常。

`\backmatter` 后记部分，页码格式不变，继续正常计数；其后的 `\chapter` 不编号。

以上三个命令还可和 `\appendix` 命令结合，生成有前言、正文、附录、后记四部分的文档。源代码 3.1 结合 1.6 节的 `\include` 命令和其它一些命令示意了一份完整的文档结构。

3.2 标题页

L^AT_EX 支持生成简单的标题页。首先需要给定标题和作者等信息：

```
\title{\langle title \rangle} \author{\langle author \rangle} \date{\langle date \rangle}
```

其中前两个命令是必须的 (不用 `\title` 会报错；不用 `\author` 会警告)，`\date` 命令可选。L^AT_EX 还提供了一个 `\today` 命令自动生成当前日期，`\date` 默认使用 `\today`。在 `\title`、`\author` 等命令内可以使用 `\thanks` 命令生成标题页的脚注，用 `\and` 隔开多个人名。

在信息给定后，就可以使用 `\maketitle` 命令生成一个简单的标题页了。源代码 3.2 给出了一个标题页的示例和大致效果。`article` 文档类的标题默认不单独成页，而 `report` 和 `book` 默认单独成页。可在 `\documentclass` 命令调用文档类时指定 `titlepage` 或 `notitlepage` 选项以修改默认的行为。

L^AT_EX 标准类还提供了一个简单的 `titlepage` 环境，生成不带页眉页脚的一页。用户可以在这个环境中使用各种排版元素自由发挥，生成自定义的标题页以替代 `\maketitle` 命令。甚至可以利用 `titlepage` 环境重新定义 `\maketitle`：

²有的参考文档可能使用 `\begin{appendix} ... \end{appendix}` 这样的写法，虽然有效，但并不规范，只要使用 `\appendix` 命令就够了。

```

\documentclass{book}

% 导言区，加载宏包和各项设置，包括参考文献、索引等
\usepackage{makeidx}      % 调用 makeidx 宏包，用来处理索引
\makeindex                % 开启索引的收集
\bibliographystyle{plain} % 指定参考文献样式为 plain

\begin{document}

\frontmatter              % 前言部分
\maketitle                % 标题页
\include{preface}         % 前言章节 preface.tex
\tableofcontents

\mainmatter               % 正文部分
\include{chapter1}        % 第一章 chapter1.tex
\include{chapter2}        % 第二章 chapter2.tex
...
\appendix                % 附录
\include{appendixA}       % 附录 A appendixA.tex
...

\backmatter              % 后记部分
\include{epilogue}        % 后记 epilogue.tex
\bibliography{books}      % 利用 BibTeX 工具从数据库文件 books.bib 生成参考文献
\printindex              % 利用 makeindex 工具生成索引

\end{document}

```

源代码 3.1: book 文档类的文档结构示例。

```

\title{Test title}
\author{ Mary\thanks{E-mail:*****@***.com}
  \and Ted\thanks{Corresponding author}
  \and Louis}
\date{\today}

```

Test title

Mary^{*}

Ted[†]

Louis

April 3, 2024

^{*}E-mail:*****@***.com

[†]Corresponding author

源代码 3.2: L^AT_EX 默认标题页示例和效果。

```
\renewcommand{\maketitle}{\begin{titlepage}
... % 用户自定义命令
\end{titlepage}}
```

事实上，为标准文档类指定了 `titlepage` 选项以后，使用 `\maketitle` 命令生成的标题页就是一个 `titlepage` 环境。

以上是 \LaTeX 标准文档类的标题页相关命令用法。在各种文档模板中经常有自定义的标题页，有可能需要除了 `\title` 和 `\author` 以外的命令给定信息，用法也可能与标准文档类的不一致（甚至有些模板可能没有定义 `titlepage` 等环境）。使用文档模板前一定要仔细阅读文档模板的帮助文档。

3.3 交叉引用

交叉引用是 \LaTeX 强大的自动排版功能的体现之一。在能够被交叉引用的地方，如章节、公式、图表、定理等位置使用 `\label` 命令：

```
\label{<label-name>}
```

之后可以在别处使用 `\ref` 或 `\pageref` 命令，分别生成交叉引用的编号和页码：

```
\ref{<label-name>} \pageref{<label-name>}
```

```
A reference to this subsection
\label{sec:this} looks like:
``see section~\ref{sec:this} on
page~\pageref{sec:this}.''
```

```
A reference to this subsection looks like:
“see section 3.3 on page 22.”
```

为了生成正确的交叉引用，一般也需要多次编译源代码。

`\label` 命令可用于记录各种类型的交叉引用，使用位置分别为：

章节标题 在章节标题命令 `\section` 等之后紧接着使用。

行间公式 单行公式在公式内任意位置使用；多行公式在每一行公式的任意位置使用。

有序列表 在 `enumerate` 环境的每个 `\item` 命令之后、下一个 `\item` 命令之前任意位置使用。

图表标题 在图表标题命令 `\caption` 之后紧接着使用。

定理环境 在定理环境内部任意位置使用。

在使用不记编号的命令形式（`\section*`、`\caption*`³、带可选参数的 `\item` 命令等）时不要使用 `\label` 命令，否则生成的引用编号不正确。

3.4 脚注和边注

使用 `\footnote` 命令可以在页面底部生成一个脚注：

```
\footnote{<footnote>}
```

假如我们输入以下文字和命令：

³需加载相关宏包，如 `caption`

“天地玄黄，宇宙洪荒。日月盈昃，辰宿列张。” \footnote{出自《千字文》。}

在正文中则为：“天地玄黄，宇宙洪荒。日月盈昃，辰宿列张。”⁴

有些情况下（比如在表格环境、各种盒子内）使用 \footnote 并不能正确生成脚注。我们可以分两步进行，先使用 \footnotemark 为脚注计数，再在合适的位置用 \footnotetext 生成脚注。比如：

```
\begin{tabular}{l}
\hline
“天地玄黄，宇宙洪荒。日月盈昃，辰宿列张。” \footnotemark \\
\hline
\end{tabular}
\footnotetext{表格里的名句出自《千字文》。}
```

效果为：

“天地玄黄，宇宙洪荒。日月盈昃，辰宿列张。”⁵

使用 \marginpar 命令可在边栏位置生成边注：

```
\marginpar[⟨left-margin⟩]{⟨right-margin⟩}
```

如果只给定了 ⟨right-margin⟩，那么边注在奇偶数页文字相同；如果同时给定了 ⟨left-margin⟩，则偶数页使用 ⟨left-margin⟩ 的文字。

例如以下代码：

```
\marginpar{\footnotesize 边注较窄，不要写过多文字，最好设置较小的字号。}
```

其效果见边栏。

边注较窄，不要写过多文字。最好设置较小的字号。

3.5 特殊环境

3.5.1 列表

L^AT_EX 提供了基本的有序和无序列表环境 `enumerate` 和 `itemize`，两者的用法很类似，都用 \item 标明每个列表项。`enumerate` 环境会自动对列表项编号。

```
\begin{enumerate}
\item ...
\end{enumerate}
```

其中 \item 可带一个可选参数，将有序列表的计数或者无序列表的符号替换成自定义的符号。列表可以嵌套使用，最多嵌套四层。

```
\begin{enumerate}
\item An item.
\begin{enumerate}
\item A nested item.\label{itref}
\item[*] A starred item.
\end{enumerate}
\item Reference(\ref{itref}).
\end{enumerate}
```

1. An item.
 - (a) A nested item.
 - * A starred item.
2. Reference([1a](#)).

⁴出自《千字文》。

⁵表格里的名句出自《千字文》。

```
\begin{itemize}
  \item An item.
  \begin{itemize}
    \item A nested item.
    \item[+] A 'plus' item.
    \item Another item.
  \end{itemize}
  \item Go back to upper level.
\end{itemize}
```

- An item.
 - A nested item.
 - + A 'plus' item.
 - Another item.
- Go back to upper level.

关键字环境 `description` 的用法与以上两者类似，不同的是 `\item` 后的可选参数用来写关键字，以粗体显示，一般是必填的：

```
\begin{description}
  \item[item title] ...
\end{description}
```

```
\begin{description}
  \item[Enumerate] Numbered list.
  \item[Itemize] Non-numbered list.
\end{description}
```

Enumerate Numbered list.
Itemize Non-numbered list.

各级无序列表的符号由命令 `\labelitemi` 到 `\labelitemiv` 定义，可以简单地重新定义它们：

```
\renewcommand{\labelitemi}{\ddag}
\renewcommand{\labelitemii}{\dag}
\begin{itemize}
  \item First item
  \begin{itemize}
    \item Subitem
    \item Subitem
  \end{itemize}
  \item Second item
\end{itemize}
```

‡ First item
 ‡ Subitem
 ‡ Subitem
 ‡ Second item

有序列表的符号由命令 `\labelenumi` 到 `\labelenumiv` 定义，重新定义这些命令需要用到 8.3 节的计数器相关命令：

```
\renewcommand{\labelenumi}{%
  {\Alph{enumi}>}}
\begin{enumerate}
  \item First item
  \item Second item
\end{enumerate}
```

A> First item
 B> Second item

默认的列表间距比较宽， \LaTeX 本身也未提供方便的定制功能，可用 `enumitem` 宏包定制各种列表间距。`enumitem` 宏包还提供了对列表标签、引用等的定制。有兴趣的读者可参考其帮助文档。

3.5.2 对齐环境

`center`、`flushleft` 和 `flushright` 环境分别用于生成居中、左对齐和右对齐的文本环境。

```
\begin{center} ... \end{center}
\begin{flushleft} ... \end{flushleft}
\begin{flushright} ... \end{flushright}
```

```
\begin{center}
Centered text using a
\verb|center| environment.
\end{center}
\begin{flushleft}
Left-aligned text using a
\verb|flushleft| environment.
\end{flushleft}
\begin{flushright}
Right-aligned text using a
\verb|flushright| environment.
\end{flushright}
```

Centered text using a `center` environment.

Left-aligned text using a `flushleft` environment.

Right-aligned text using a `flushright` environment.

除此之外，还可以用以下命令直接改变文字的对齐方式：

```
\centering \raggedright \raggedleft
```

```
\centering
Centered text paragraph.

\raggedright
Left-aligned text paragraph.

\raggedleft
Right-aligned text paragraph.
```

Centered text paragraph.

Left-aligned text paragraph.

Right-aligned text paragraph.

三个命令和对应的环境经常被误用，有直接用所谓 `\flushleft` 命令或者 `raggedright` 环境的，都是不甚严格的用法（即使它们可能有效）。有一点可以将两者区分开来：`center` 等环境会在上下文产生一个额外间距，而 `\centering` 等命令不产生，只是改变对齐方式。比如在浮动体环境 `table` 或 `figure` 内实现居中对齐，用 `\centering` 命令即可，没必要再用 `center` 环境。

3.5.3 引用环境

\LaTeX 提供了两种引用的环境：`quote` 用于引用较短的文字，首行不缩进；`quotation` 用于引用若干段文字，首行缩进。引用环境较一般文字有额外的左右缩进。

```
Francis Bacon says:
\begin{quote}
Knowledge is power.
\end{quote}
```

Francis Bacon says:

Knowledge is power.

《木兰诗》：

```
\begin{quotation}
```

万里赴戎机，关山度若飞。

朔气传金柝，寒光照铁衣。

将军百战死，壮士十年归。

归来见天子，天子坐明堂。

策勋十二转，赏赐百千强。……

```
\end{quotation}
```

《木兰诗》：

万里赴戎机，关山度若飞。

朔气传金柝，寒光照铁衣。将军

百战死，壮士十年归。

归来见天子，天子坐明堂。

策勋十二转，赏赐百千强。……

verse 用于排版诗歌，与 quotation 恰好相反，verse 是首行悬挂缩进的。

Rabindranath Tagore's short poem:

```
\begin{verse}
```

Beauty is truth's smile

when she beholds her own face in

a perfect mirror.

```
\end{verse}
```

Rabindranath Tagore's short poem:

Beauty is truth's smile when she

beholds her own face in a

perfect mirror.

3.5.4 摘要环境

摘要环境 abstract 默认只在标准文档类中的 article 和 report 文档类可用，一般用于紧跟 \maketitle 命令之后介绍文档的摘要。如果文档类指定了 titlepage 选项，则单独成页；反之，单栏排版时相当于一个居中的小标题加一个 quotation 环境，双栏排版时相当于 \section* 定义的一节。

3.5.5 代码环境

有时我们需要将一段代码原样转义输出，这就要用到代码环境 verbatim，它以等宽字体排版代码，回车和空格也分别起到换行和空位的作用；带星号的版本更进一步将空格显示成 “_”。

```
\begin{verbatim}
#include <iostream>
int main()
{
    std::cout << "Hello, world!"
               << std::endl;

    return 0;
}
\end{verbatim}
```

```
#include <iostream>
int main()
{
    std::cout << "Hello, world!"
               << std::endl;

    return 0;
}
```

```
\begin{verbatim*}
for (int i=0; i<4; ++i)
    printf("Number %d\n",i);
\end{verbatim*}
```

```
for_(int_i=0;_i<4;_++i)
    _printf("Number_%d\n",i);
```

要排版简短的代码或关键字，可使用 \verb 命令：

```
\verb<delim><code><delim>
```

$\langle delim \rangle$ 标明代码的分界位置，前后必须一致，除字母、空格或星号外，可任意选择使得不与代码本身冲突，习惯上使用 `|` 符号。

同 `verbatim` 环境，`\verb` 后也可以带一个星号，以显示空格：

```
\verb|\LaTeX| \\\
```

```
\verb+(a || b)+ \verb*+(a || b)+
```

```
\LaTeX
```

```
(a || b) (a_||_b)
```

`\verb` 命令对符号的处理比较复杂，一般不能用在其它命令的参数里，否则多半会出错。

`verbatim` 宏包优化了 `verbatim` 环境的内部命令，并提供了 `\verbatiminput` 命令用来直接读入文件生成代码环境。`fancyvrb` 宏包提供了可定制格式的 `Verbatim` 环境；`listings` 宏包更进一步，可生成关键字高亮的代码环境，支持各种程序设计语言的语法和关键字。详情请参考各自的帮助文档。

3.6 表格

\LaTeX 里排版表格不如 Word 等所见即所得的工具简便和自由，不过对于不太复杂的表格来讲，完全能够胜任。

排版表格最基本的 `tabular` 环境用法为：

```
\begin{tabular}[\langle align \rangle]{\langle column-spec \rangle}
\langle item1 \rangle & \langle item2 \rangle & ... \\
\hline
\langle item1 \rangle & \langle item2 \rangle & ... \\
\end{tabular}
```

其中 $\langle column-spec \rangle$ 是列格式标记，在接下来的内容将详细介绍；`&` 用来分隔单元格；`\\` 用来换行；`\hline` 用来在行与行之间绘制横线。

直接使用 `tabular` 环境的话，会和周围的文字混排。此时可用一个可选参数 $\langle align \rangle$ 控制垂直对齐：`t` 和 `b` 分别表示按表格顶部、底部对齐，其他参数或省略不写（默认）表示居中对齐。

```
\begin{tabular}{|c|}
center-\\ aligned \\
\end{tabular},
\begin{tabular}[t]{|c|}
top-\\ aligned \\
\end{tabular},
\begin{tabular}[b]{|c|}
bottom-\\ aligned \\
\end{tabular} tabulars.
```

center- aligned	,	top- aligned	,	bottom- aligned	tabulars.
--------------------	---	-----------------	---	--------------------	-----------

但是通常情况下 `tabular` 环境很少与文字直接混排，而是会放在 `table` 浮动体环境中，并用 `\caption` 命令加标题。

3.6.1 列格式

`tabular` 环境使用 $\langle column-spec \rangle$ 参数指定表格的列数以及每列的格式。基本的列格式见表 3.1。

表 3.1: L^AT_EX 表格列格式

列格式	说明
l/c/r	单元格内容左对齐/居中/右对齐，不折行
p{⟨width⟩}	单元格宽度固定为 ⟨width⟩，可自动折行
	绘制竖线
@{⟨string⟩}	自定义内容 ⟨string⟩

```
\begin{tabular}{lcr|p{6em}}
\hline
left & center & right & par box with
      & par box with fixed width\\
L    & C    & R    & P \\
\hline
\end{tabular}
```

left	center	right	par box with fixed width
L	C	R	P

表格中每行的单元格数目不能多于列格式里 l/c/r/p 的总数（可以少于这个总数），否则出错。

@ 格式可在单元格前后插入任意的文本，但同时它也消除了单元格前后额外添加的间距。@ 格式可以适当使用以充当“竖线”。特别地，@{} 可直接用来消除单元格前后的间距：

```
\begin{tabular}{@{} r@{:}lr @{}}
\hline
1 & 1 & one \\
11 & 3 & eleven \\
\hline
\end{tabular}
```

1:1	one
11:3	eleven

另外 L^AT_EX 还提供了简便的将格式参数重复的写法 *{⟨n⟩}{⟨column-spec⟩}，比如以下两种写法是等效的：

```
\begin{tabular}{|c|c|c|c|c|c|p{4em}|p{4em}|}
\begin{tabular}{|*{5}{c}|*{2}{p{4em}}|}
```

有时需要为整列修饰格式，比如整列改变为粗体，如果每个单元格都加上 \bfseries 命令会比较麻烦。array 宏包提供了辅助格式 > 和 <，用于给列格式前后加上修饰命令：

```
% \usepackage{array}
\begin{tabular}{>{\itshape}r<{*}l}
\hline
italic & normal \\
column & column \\
\hline
\end{tabular}
```

<i>italic</i> *	normal
<i>column</i> *	column

辅助格式甚至支持插入 \centering 等命令改变 p 列格式的对齐方式，一般还要加额外的命令 \arraybackslash 以免出错⁶：

⁶\centering 等对齐命令会破坏表格环境里 \\ 换行命令的定义，\arraybackslash 用来恢复之。如果不加 \arraybackslash 命令，也可以用 \tabularnewline 命令代替原来的 \\ 实现表格换行。


```
% \usepackage{array}
\begin{tabular}%
{>{\centering\arraybackslash}p{9em}}
\hline
Some center-aligned long text. \\
\hline
\end{tabular}
```

Some center-aligned long text.

`array` 宏包还提供了类似 `p` 格式的 `m` 格式和 `b` 格式，三者分别在垂直方向上靠顶端对齐、居中以及底端对齐。

```
% \usepackage{array}
\newcommand\txt{a b c d e f g h i}
\begin{tabular}{cp{2em}m{2em}b{2em}}
\hline
pos & \txt & \txt & \txt \\
\hline
\end{tabular}
```

			a b c
		a b c	d e f
pos	a b c	d e f	g h i
	d e f	g h i	
	g h i		

3.6.2 列宽

在控制列宽方面， \LaTeX 表格有着明显的不足：`l/c/r` 格式的列宽是由文字内容的自然宽度决定的，而 `p` 格式给定了列宽却不好控制对齐（可用 `array` 宏包的辅助格式），更何况列与列之间通常还有间距，所以直接生成给定总宽度的表格并不容易。

\LaTeX 本身提供了 `tabular*` 环境用来排版定宽表格，但是不太方便使用，比如要用到 `@` 格式插入额外命令，令单元格之间的间距为 `\fill`，但即使这样仍然有瑕疵：

```
\begin{tabular*}{14em}%
{@{\extracolsep{\fill}}|c|c|c|c|}
\hline
A & B & C & D \\ \hline
a & b & c & d \\ \hline
\end{tabular*}
```

A	B	C	D
a	b	c	d

`tabularx` 宏包为我们提供了方便的解决方案。它引入了一个 `X` 列格式，类似 `p` 列格式，不过会根据表格宽度自动计算列宽，多个 `X` 列格式平均分配列宽。`X` 列格式也可以用 `array` 里的辅助格式修饰对齐方式：

```
% \usepackage{array,tabularx}
\begin{tabularx}{14em}%
{!*{4}{>{\centering\arraybackslash}X|}}
\hline
A & B & C & D \\ \hline
a & b & c & d \\ \hline
\end{tabularx}
```

A	B	C	D
a	b	c	d

3.6.3 横线

我们已经在之前的例子见过许多次绘制表格线的 `\hline` 命令。另外 `\cline{<i>-<j>}` 用来绘制跨越部分单元格的横线：

```
\begin{tabular}{|c|c|c|}
\hline
4 & 9 & 2 \\ \cline{2-3}
3 & 5 & 7 \\ \cline{1-1}
8 & 1 & 6 \\ \hline
\end{tabular}
```

4	9	2
3	5	7
8	1	6

在科技论文排版中广泛应用的表格形式是三线表，形式干净简明。三线表由 `booktabs` 宏包支持，它提供了 `\toprule`、`\midrule` 和 `\bottomrule` 命令用以排版三线表的三条线，以及和 `\cline` 对应的 `\cmidrule`。除此之外，最好不要用其它横线以及竖线：

```
% \usepackage{booktabs}
\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

3.6.4 合并单元格

L^AT_EX 是一行一行排版表格的，横向合并单元格较为容易，由 `\multicolumn` 命令实现：

```
\multicolumn{⟨n⟩}{⟨column-spec⟩}{⟨item⟩}
```

其中 $\langle n \rangle$ 为要合并的列数， $\langle column-spec \rangle$ 为合并单元格后的列格式，只允许出现一个 `l/c/r` 或 `p` 格式。如果合并前的单元格前后带表格线 `|`，合并后的列格式也要带 `|` 以使得表格的竖线一致。

```
\begin{tabular}{|c|c|c|}
\hline
1 & 2 & Center \\ \hline
\multicolumn{2}{|c|}{3} & \\ \hline
\multicolumn{1}{r|}{4} & Right & \\ \hline
4 & \multicolumn{2}{c|}{C} \\ \hline
\end{tabular}
```

1	2	Center
3		Right
4	C	

上面的例子还体现了，形如 `\multicolumn{1}{⟨column-spec⟩}{⟨item⟩}` 的命令可以用来修改某一个单元格的列格式。

纵向合并单元格需要用到 `multirow` 宏包提供的 `\multirow` 命令：

```
\multirow{⟨n⟩}{⟨width⟩}{⟨item⟩}
```

$\langle width \rangle$ 为合并后单元格的宽度，可以填 `*` 以使用自然宽度。

我们看一个结合 `\cline`、`\multicolumn` 和 `\multirow` 命令的例子：

```
% \usepackage{multirow}
\begin{tabular}{ccc}
\hline
\multirow{2}{*}{Item} & & 
\multicolumn{2}{c}{Value} \\
\cline{2-3}
& First & Second \\
A & 1 & 2 \\
\hline
\end{tabular}
```

Item	Value	
	First	Second
A	1	2

3.6.5 嵌套表格

相对于合并单元格，拆分单元格对于 \LaTeX 来说并非易事。在单元格中嵌套一个小表格可以起到“拆分单元格”的效果。在以下的例子中，注意要用 `\multicolumn` 命令配合 `@{}` 格式把单元格的额外边距去掉，使得嵌套的表格线能和外层的表格线正确相连：

```
\begin{tabular}{|c|c|c|}
\hline
a & b & c \\
\hline
a & \multicolumn{1}{@{}c@{}}{
\begin{tabular}{c|c}
e & f \\
\hline
e & f
\end{tabular}
} & c \\
\hline
a & b & c \\
\hline
\end{tabular}
```

a	b	c
a	e	f
	e	f
a	b	c

如果不需要为“拆分的单元格”画线，并且只在垂直方向“拆分”的话，`makecell` 宏包提供的 `\makecell` 命令是一个简单的解决方案：

```
% \usepackage{makecell}
\begin{tabular}{|c|c|}
\hline
a & \makecell{d1 \\ d2} \\
\hline
b & c \\
\hline
\end{tabular}
```

a	d1 d2
b	c

3.6.6 行距控制

\LaTeX 生成的表格看起来通常比较紧凑。修改参数 `\arraystretch` 可以得到行距更加宽松的表格（相关命令参考 8.1.1 小节）：

```
\renewcommand\arraystretch{1.8}
\begin{tabular}{|c|}
\hline
Really loose \\
\hline
tabular rows. \\
\hline
\end{tabular}
```

Really loose
tabular rows.

另一种增加间距的办法是给换行命令 `\` 添加可选参数，在这一行下面加额外的间距，适用于在行间不加横线的表格：

```
\begin{tabular}{c}
\hline
Head lines \\[6pt]
tabular lines \\
tabular lines \\ \hline
\end{tabular}
```

Head lines
tabular lines
tabular lines

但是这种换行方式的存在导致了一个缺陷——从第二行开始，表格的首个单元格不能直接使用中括号 `[]`，否则 `\` 往往会将下一行的中括号当作自己的可选参数，因而出错。如果要使用中括号，应当放在花括号 `{}` 里面。或者也可以选择将换行命令写成 `\\[0pt]`。

3.7 图片

\LaTeX 本身不支持插图功能，需要由 `graphicx` 宏包辅助支持。

使用 `latex + dvipdfmx` 编译命令时，调用 `graphicx` 宏包时要指定 `dvipdfmx` 选项⁷；而使用 `pdflatex` 或 `xelatex` 命令编译时不需要。

读者可能听说过“ \LaTeX 只能插入 `.eps` 格式的图片，需要把 `.jpg` 转成 `.eps` 格式”的观点。 \LaTeX 发展到今天，这个观点早已过时。事实上不同编译命令支持的图片格式种类各异，见表 3.2。这个表格也能解答诸如“为什么 `.eps` 格式图片在 `pdflatex` 编译命令下出错”之类的问题。本表格也再一次说明，使用 `xelatex` 命令是我们最推荐的方式。

表 3.2: 各种编译方式支持的主流图片格式

格式	矢量图	位图
<code>latex + dvipdfmx</code>	<code>.eps</code>	N/A
<code>\</code> (调用 <code>bmpsize</code> 宏包)	<code>.eps</code> <code>.pdf</code>	<code>.jpg</code> <code>.png</code> <code>.bmp</code>
<code>pdflatex</code>	<code>.pdf</code>	<code>.jpg</code> <code>.png</code>
<code>\</code> (调用 <code>epstopdf</code> 宏包)	<code>.pdf</code> <code>.eps</code>	<code>.jpg</code> <code>.png</code>
<code>xelatex</code>	<code>.pdf</code> <code>.eps</code>	<code>.jpg</code> <code>.png</code> <code>.bmp</code>

注：在较新的 \TeX 发行版中，`latex + dvipdfmx` 和 `pdflatex` 命令可不依赖宏包，支持原来需要宏包扩展的图片格式（但 `pdflatex` 命令仍不支持 `.bmp` 格式的位图）。

在调用了 `graphicx` 宏包以后，就可以使用 `\includegraphics` 命令加载图片了：

```
\includegraphics[options]{filename}
```

其中 `<filename>` 为图片文件名，与 `\include` 命令的用法类似，文件名可能需要用相对路径或绝对路径表示（见 1.6 节）。图片文件的扩展名一般可不写。另外一定要注意，**文件名里既不要有空格（类似 `\include`），也不要有多余的英文点号**，否则宏包在解析文件名的过程中会出错。

另外 `graphicx` 宏包还提供了 `\graphicspath` 命令，用于声明一个或多个图片文件存放的目录，使用这些目录里的图片时可不用写路径：

⁷早期常使用 `latex + dvips` 组合命令，后者将 `.dvi` 文件转为 `.ps` 文件 (PostScript)，可进一步通过 `ps2pdf` 工具生成 PDF。`dvips` 和 `dvipdfmx` 在图形、颜色、超链接等功能的实现上有差别，而 \LaTeX 无法识别用户是用 `dvips` 还是 `dvipdfmx`，所以要指定选项（缺省为 `dvips`）。6.4 节中的 `hyperref` 宏包同理。

% 假设主要的图片放在 `figures` 子目录下, 标志放在 `logo` 子目录下
`\graphicspath{{figures/}{logo/}}`

在 `\includegraphics` 命令的可选参数 $\langle options \rangle$ 中可以使用 $\langle key \rangle = \langle value \rangle$ 的形式, 常用的参数如下:

表 3.3: `\includegraphics` 命令的可选参数

参数	含义
<code>width=\langle width \rangle</code>	将图片缩放到宽度为 $\langle width \rangle$
<code>height=\langle height \rangle</code>	将图片缩放到高度为 $\langle height \rangle$
<code>scale=\langle scale \rangle</code>	将图片相对于原尺寸缩放 $\langle scale \rangle$ 倍
<code>angle=\langle angle \rangle</code>	将图片逆时针旋转 $\langle angle \rangle$ 度

`graphicx` 宏包也支持 `draft/final` 选项。当 `graphicx` 宏包或文档类指定 `draft` 选项时, 图片将不会被实际插入, 取而代之的是一个包含文件名的与原图片等大的方框。

3.8 盒子

盒子是 \LaTeX 排版的基础单元, 虽然解释略有抽象: 每一行是一个盒子, 里面的文字从左到右依次排列; 每一页也是一个盒子, 各行文字从上到下依次排布……颇有一些活字印刷术的味道。

不管如何, \LaTeX 提供了一些命令让我们手动生成一些有特定用途的盒子。

3.8.1 水平盒子

生成水平盒子的命令如下:

```
\mbox{...}
\makebox[\langle width \rangle][\langle align \rangle]{...}
```

`\mbox` 生成一个基本的水平盒子, 内容只有一行, 不允许分段 (除非嵌套其它盒子, 比如后文的垂直盒子)。外表看上去, `\mbox` 的内容与正常的文本无二, 不过断行时文字不会从盒子里断开。

`\makebox` 更进一步, 可以加上可选参数用于控制盒子的宽度 $\langle width \rangle$, 以及内容的对齐方式 $\langle align \rangle$, 可选居中 `c` (默认值)、左对齐 `l`、右对齐 `r` 和分散对齐 `s`⁸。

```
| \mbox{Test some words.} | \\\
| \makebox[10em]{Test some words.} | \\\
| \makebox[10em][l]{Test some words.} | \\\
| \makebox[10em][r]{Test some words.} | \\\
| \makebox[10em][s]{Test some words.} |
```

```
| Test some words. |
|   Test some words.   |
| Test some words.   |
|       Test some words. |
| Test   some   words. |
```

3.8.2 带框的水平盒子

`\fbox` 和 `\framebox` 让我们可以为水平盒子添加边框。使用的语法与 `\mbox` 和 `\makebox` 一模一样:

⁸分散对齐方式强行拉开单词的间距, 往往会报 `Underfull \hbox` 的警告。

```
\fbox{...}
\framebox[⟨width⟩][⟨align⟩]{...}
```

```
\fbox{Test some words.}\
\framebox[10em][r]{Test some words.}
```

可以通过 `\setlength` 命令 (见 5.3.1 小节) 调节边框的宽度 `\fboxrule` 和内边距 `\fboxsep`:

```
\framebox[10em][r]{Test box}\[1ex]
\setlength{\fboxrule}{1.6pt}
\setlength{\fboxsep}{1em}
\framebox[10em][r]{Test box}
```

3.8.3 垂直盒子

如果需要排版一个文字可以换行的盒子, L^AT_EX 提供了两种方式:

```
\parbox[⟨align⟩][⟨height⟩][⟨inner-align⟩]{⟨width⟩}{...}
\begin{minipage}[⟨align⟩][⟨height⟩][⟨inner-align⟩]{⟨width⟩}
...
\end{minipage}
```

其中 `⟨align⟩` 为盒子和周围文字的对齐情况 (类似 `tabular` 环境); `⟨height⟩` 和 `⟨inner-align⟩` 设置盒子的高度和内容的对齐方式, 类似水平盒子 `\makebox` 的设置, 不过 `⟨inner-align⟩` 接受的参数是顶部 `t`、底部 `b`、居中 `c` 和分散对齐 `s`。

```
三字经: \parbox[t]{3em}%
{人之初 性本善 性相近 习相远}
\quad
千字文:
\begin{minipage}[b][8ex][t]{4em}
天地玄黄 宇宙洪荒
\end{minipage}
```

如果在 `minipage` 里使用 `\footnote` 命令, 生成的脚注会出现在盒子底部, 编号是独立的, 并且使用小写字母编号。这也是 `minipage` 环境之被称为“迷你页” (Mini-page) 的原因。而在 `\parbox` 里无法正常使用 `\footnote` 命令, 只能在盒子里使用 `\footnotemark`, 在盒子外使用 `\footnotetext`。

```
\fbox{\begin{minipage}{15em}%
这是一个垂直盒子的测试。
\footnote{脚注来自 minipage。}
\end{minipage}}
```

3.8.4 标尺盒子


`\rule` 命令用来画一个实心的矩形盒子, 也可适当调整以用来画线 (标尺):



```
\rule[⟨raise⟩]{⟨width⟩}{⟨height⟩}
```


Black \rule{12pt}{4pt} box.

Upper \rule[4pt]{6pt}{8pt} and lower \rule[-4pt]{6pt}{8pt} box.

A \rule[-.4pt]{3em}{.4pt} line.

Black  box.

Upper  and lower  box.

A  line.

3.9 浮动体

内容丰富的文章或者书籍往往包含许多图片和表格等内容。这些内容的尺寸往往太大，导致分页困难。L^AT_EX 为此引入了浮动体的机制，令大块的内容可以脱离上下文，放置在合适的位置。L^AT_EX 预定义了两类浮动体环境 `figure` 和 `table`。习惯上 `figure` 里放图片，`table` 里放表格，但并没有严格限制，可以在任何一个浮动体里放置文字、公式、表格、图片等等任意内容。以 `table` 环境的用法举例，`figure` 同理：

```
\begin{table}[<placement>]
...
\end{table}
```

`<placement>` 参数提供了一些符号用来表示浮动体允许排版的位置，如 `hbp` 允许浮动体排版在当前位置、底部或者单独成页。`table` 和 `figure` 浮动体的默认设置为 `tbp`。

表 3.4: 浮动体的位置参数

参数	含义
<code>h</code>	当前位置（代码所处的上下文）
<code>t</code>	顶部
<code>b</code>	底部
<code>p</code>	单独成页
<code>!</code>	在决定位置时忽视限制

注 1：排版位置的选取与参数里符号的顺序无关，L^AT_EX 总是以 `h-t-b-p` 的优先级顺序决定浮动体位置。也就是说 `[!htp]` 和 `[ph!t]` 没有区别。

注 2：限制包括浮动体个数（除单独成页外，默认每页不超过 3 个浮动体，其中顶部不超过 2 个，底部不超过 1 个）以及浮动体空间占页面的百分比（默认顶部不超过 70%，底部不超过 30%）。

双栏排版环境下，L^AT_EX 提供了 `table*` 和 `figure*` 环境用来排版跨栏的浮动体。它们的用法与 `table` 和 `figure` 一样，不同之处为双栏的 `<placement>` 参数只能用 `tp` 两个位置。

浮动体的位置选取受到先后顺序的限制。如果某个浮动体由于参数限制、空间限制等原因在当前页无法放置，就要推迟到之后处理，并使得之后的同类浮动体一并推迟。`\clearpage` 命令会在另起一页之前，先将所有推迟处理的浮动体排版成页，此时 `htbp` 等位置限制被完全忽略。

`float` 宏包为浮动体提供了 `H` 位置参数，不与 `htbp` 及 `!` 混用。使用 `H` 位置参数时，会取消浮动机制，将浮动体视为一般的盒子插入当前位置。这在一些特殊情况下很有用（如使用 `multicol` 宏包排版分栏内容的时候），但尺寸过大的浮动体可能使得分页比较困难。

3.9.1 浮动体的标题

图表等浮动体提供了 `\caption` 命令加标题，并且自动给浮动体编号：

```
\caption{...}
```

`\caption` 的用法非常类似于 `\section` 等命令，可以用带星号的命令 `\caption*`⁹ 生成不带编号的标题，也可以使用带可选参数的形式 `\caption[...]{...}`，使得在目录里使用短标题。`\caption` 命令之后还可以紧跟 `\label` 命令标记交叉引用。

`\caption` 生成的标题形如 “Figure 1: ...” (`figure` 环境) 或 “Table 1: ...” (`table` 环境)。可通过修改 `\figurename` 和 `\tablename` 的内容来修改标题的前缀（详见第 8.4 节）。标题样式的定制功能由 `caption` 宏包提供，详见该宏包的帮助文档，在此不作赘述。

`table` 和 `figure` 两种浮动体分别有各自的生成目录的命令：

```
\listoftables
```

```
\listoffigures
```

它们类似 `\tableofcontents` 生成单独的章节。

3.9.2 并排和子图表

我们时常有在一个浮动体里面放置多张图的用法。最简单的用法就是直接并排放置，也可以通过分段或者换行命令 `\\` 排版多行多列的图片。以下为示意代码，效果大致如图 3.1 所示。

```
\begin{figure}[htbp]
  \centering
  \includegraphics[width=...]{...}
  \quad
  \includegraphics[width=...]{...} \\[...pt]
  \includegraphics[width=...]{...}
  \caption{...}
\end{figure}
```



图 3.1: 并排放置图片的示意。

由于标题是横跨一行的，用 `\caption` 命令为每个图片单独生成标题就需要借助前文提到的 `\parbox` 或者 `minipage` 环境，将标题限制在盒子内。效果见图 3.2 和图 3.3。

```
\begin{figure}[htbp]
  \centering
  \begin{minipage}{...}
```

⁹需加载相关宏包，如 `caption`


```

\centering
\includegraphics[width=...]{...}
\caption{...}
\end{minipage}
\quad
\begin{minipage}{...}
\centering
\includegraphics[width=...]{...}
\caption{...}
\end{minipage}
\end{figure}

```



图 3.2: 并排图 1。

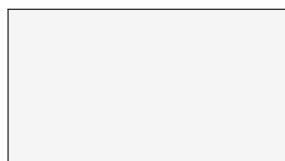


图 3.3: 并排图 2。

当我们需要更进一步, 给每个图片定义小标题时, 就要用到 `subcaption` 宏包的功能了。这里仅举一例, 效果见图 3.4a 和 3.4b。更详细的用法请参考 `subcaption` 宏包的帮助文档。

```

\begin{figure}[htbp]
\centering
\begin{subfigure}{...}
\centering
\includegraphics[width=...]{...}
\caption{...}
\end{subfigure}
\quad
\begin{subfigure}{...}
\centering
\includegraphics[width=...]{...}
\caption{...}
\end{subfigure}
\end{figure}

```



(a) 并排子图 1



(b) 并排子图 2

图 3.4: 使用 `subcaption` 宏包的 `subfigure` 环境排版子图。

`subcaption` 依赖上文提到过的 `caption` 宏包，因此也支持子图表标题样式的定制。并排子图表的功能也可通过 `subfig` 宏包的 `\subfloat` 命令实现，具体请参考宏包文档。

第四章 排版数学公式

准备好了！本章将见识到 \LaTeX 闻名的强项——排版数学公式。当然你得注意了，本章的内容只是一点皮毛，虽然对大多数人来说已经够用了，但是如果不能解决你的问题的话也不要大惊小怪，求助于搜索引擎或者有经验的人不失为一个好办法。

4.1 \mathcal{AMS} 宏集

在介绍数学公式排版之前，简单介绍一下 \mathcal{AMS} 宏集。 \mathcal{AMS} 宏集合是美国数学学会 (American Mathematical Society) 提供的对 \LaTeX 原生的数学公式排版的扩展，其核心是 `amsmath` 宏包，对多行公式的排版提供了有力的支持。此外，`amsfonts` 宏包以及基于它的 `amssymb` 宏包提供了丰富的数学符号；`amsthm` 宏包扩展了 \LaTeX 定理证明格式。

本章介绍的许多命令和环境依赖于 `amsmath` 宏包，这些命令和环境将以蓝色示意。以下示例都假定了导言区中写有

```
\usepackage{amsmath}
```

4.2 公式排版基础

4.2.1 行内和行间公式

数学公式有两种排版方式：其一是与文字混排，称为**行内公式**；其二是单独列为一行排版，称为**行间公式**。

行内公式由一对 `$` 符号包裹：

```
The Pythagorean theorem is  
$a^2 + b^2 = c^2$.
```

The Pythagorean theorem is $a^2 + b^2 = c^2$.

单独成行的**行间公式**在 \LaTeX 里由 `equation` 环境包裹。`equation` 环境为公式自动生成一个编号，这个编号可以用 `\label` 和 `\ref` 生成交叉引用，`amsmath` 的 `\eqref` 命令甚至为引用自动加上圆括号；还可以用 `\tag` 命令手动修改公式的编号，或者用 `\notag` 命令取消为公式编号（与之基本等效的命令是 `\nonumber`）。

```
The Pythagorean theorem is:  
\begin{equation}  
a^2 + b^2 = c^2 \label{pythagorean}  
\end{equation}  
Equation \eqref{pythagorean} is  
called 'Gougu theorem' in Chinese.
```

The Pythagorean theorem is:

$$a^2 + b^2 = c^2 \tag{4.1}$$

Equation (4.1) is called 'Gougu theorem' in Chinese.

```

It's wrong to say
\begin{equation}
1 + 1 = 3 \tag{dumb}
\end{equation}
or
\begin{equation}
1 + 1 = 4 \notag
\end{equation}

```

It's wrong to say

$$1 + 1 = 3 \quad (\text{dumb})$$

or

$$1 + 1 = 4$$

如果需要直接使用不带编号的行间公式，则将公式用命令 `\[` 和 `\]` 包裹¹，与之等效的是 `displaymath` 环境。有的人更喜欢 `equation*` 环境，体现了带星号和不带星号的环境之间的区别。

```

\begin{equation*}
a^2 + b^2 = c^2
\end{equation*}
For short:
\[ a^2 + b^2 = c^2 \]
Or if you like the long one:
\begin{displaymath}
a^2 + b^2 = c^2
\end{displaymath}

```

$$a^2 + b^2 = c^2$$

For short:

$$a^2 + b^2 = c^2$$

Or if you like the long one:

$$a^2 + b^2 = c^2$$

我们通过一个例子展示行内公式和行间公式的对比。为了与文字相适应，行内公式在排版大的公式元素（分式、巨算符等）时显得很“局促”：

```

In text:
$\lim_{n \to \infty} \sum_{k=1}^n \frac{1}{k^2} = \frac{\pi^2}{6}$.

In display:
\[
\lim_{n \to \infty} \sum_{k=1}^n \frac{1}{k^2} = \frac{\pi^2}{6}
\]

```

In text: $\lim_{n \rightarrow \infty} \sum_{k=1}^n \frac{1}{k^2} = \frac{\pi^2}{6}.$

In display:

$$\lim_{n \rightarrow \infty} \sum_{k=1}^n \frac{1}{k^2} = \frac{\pi^2}{6}$$

行间公式的对齐、编号位置等性质由文档类选项控制，文档类的 `fleqn` 选项令行间公式左对齐；`leqno` 选项令编号放在公式左边。

4.2.2 数学模式

当用户使用 `$` 开启行内公式输入，或是使用 `\[` 命令、`equation` 环境时， \LaTeX 就进入了**数学模式**。数学模式相比于文本模式有以下特点：

1. 数学模式中输入的空格被忽略。数学符号的间距默认由符号的性质（关系符号、运算符等）决定。需要人为引入间距时，使用 `\quad` 和 `\qquad` 等命令。详见 4.6 节。

¹ \LaTeX 原生排版行间公式的方法是用一对 `$$` 符号包裹，不过无法通过指定 `fleqn` 选项控制左对齐，与上下文之间的间距也不好调整，故不太推荐使用。

2. 不允许有空行(分段)。行间公式中也无法用 `\\` 命令手动换行。排版多行公式需要用到 4.4 节介绍的各种环境。
3. 所有的字母被当作数学公式中的变量处理, 字母间距与文本模式不一致, 也无法生成单词之间的空格。如果想在数学公式中输入正体的文本, 简单情况下可用 4.7.1 小节中提供的 `\mathrm` 命令。或者用 `amsmath` 提供的 `\text` 命令²。

```
% \usepackage{amssymb}
 $x^2 \geq 0$  \quad
\text{for \textbf{all}} }
 $x \in \mathbb{R}$ 
```

$$x^2 \geq 0 \quad \text{for all } x \in \mathbb{R}$$

4.3 数学符号

本节我们将接触到形形色色的数学符号, 它们是 \LaTeX 卓越的数学公式排版能力的基础。 \LaTeX 默认提供了常用的数学符号, `amssymb` 宏包提供了一些次常用的符号。大多数常用的数学符号都能在本章末尾的 4.9 节列出的各个表格里查到。更多符号可查阅参考文献 [14]。

4.3.1 一般符号

希腊字母符号的名称就是其英文名称, 如 α (`\alpha`)、 β (`\beta`) 等等。大写的希腊字母为首字母大写的命令, 如 Γ (`\Gamma`)、 Δ (`\Delta`) 等等。无穷大符号为 ∞ (`\infty`)。更多符号命令可参考表 4.5 和 4.14 等。

省略号有 \dots (`\dots`) 和 \cdots (`\cdots`) 两种形式。它们有各自合适的用途:

```
 $a_1, a_2, \dots, a_n$  \\
 $a_1 + a_2 + \cdots + a_n$ 
```

$$a_1, a_2, \dots, a_n$$

$$a_1 + a_2 + \cdots + a_n$$

`\ldots` 和 `\dots` 是完全等效的, 它们既能用在公式中, 也用来在文本里作为省略号 (详见 2.3.5 小节)。除此之外, 在矩阵中可能会用到竖排的 \vdots (`\vdots`) 和斜排的 \ddots (`\ddots`)。

4.3.2 指数、上下标和导数

在 \LaTeX 中用 `^` 和 `_` 标明上下标。注意上下标的内容 (子公式) 一般需要用花括号包裹, 否则上下标只对后面的一个符号起作用。

```
 $p^3_{ij}$  \quad \quad
 $m_{\mathrm{Knuth}}$  \quad \quad
 $\sum_{k=1}^3 k$  \\
 $a^{x+y} \neq a^{x+y}$  \quad \quad
 $e^{x^2} \neq e^{x^2}$ 
```

$$p^3_{ij} \quad m_{\mathrm{Knuth}} \quad \sum_{k=1}^3 k$$

$$a^{x+y} \neq a^{x+y} \quad e^{x^2} \neq e^{x^2}$$

导数符号 $'$ (`'`) 是一类特殊的上标, 可以适当连用表示多阶导数, 也可以在其后连用上标:

```
 $f(x) = x^2$  \quad  $f'(x)$ 
 $= 2x$  \quad  $f''(x) = 4$ 
```

$$f(x) = x^2 \quad f'(x) = 2x \quad f''(x) = 4$$

²`\text` 命令仅适合在公式中穿插少量文字。如果你的情况正好相反, 需要在许多文字中穿插使用公式, 则应该像正常的行内公式那样用, 而不是滥用 `\text` 命令。

4.3.3 分式和根式

分式使用 `\frac{分子}{分母}` 来书写。分式的大小在行间公式中是正常大小，而在行内被极度压缩。`amsmath` 提供了方便的命令 `\dfrac` 和 `\tfrac`，令用户能够在行内使用正常大小的分式，或是反过来。

```
In display style:
\[
3/8 \quad \frac{3}{8}
\quad \tfrac{3}{8}
\]
In text style:
$1\frac{1}{2}$~hours \quad
$1\dfrac{1}{2}$~hours
```

In display style:

$$3/8 \quad \frac{3}{8} \quad \tfrac{3}{8}$$

In text style: $1\frac{1}{2}$ hours $1\frac{1}{2}$ hours

一般的根式使用 `\sqrt{...}`；表示 n 次方根时写成 `\sqrt[n]{...}`。

```
$\sqrt{x}$ \Leftrightarrow x^{1/2}
\quad \sqrt[3]{2}
\quad \sqrt{x^2 + \sqrt{y}}$
```

$$\sqrt{x} \Leftrightarrow x^{1/2} \quad \sqrt[3]{2} \quad \sqrt{x^2 + \sqrt{y}}$$

特殊的分式形式，如二项式结构，由 `amsmath` 宏包的 `\binom` 命令生成：

```
Pascal's rule is
\[
\binom{n}{k} = \binom{n-1}{k}
+ \binom{n-1}{k-1}
\]
```

Pascal's rule is

$$\binom{n}{k} = \binom{n-1}{k} + \binom{n-1}{k-1}$$

4.3.4 关系符

\LaTeX 常见的关系符号除了可以直接输入的 $=$ ， $>$ ， $<$ ，其它符号用命令输入，常用的有不等号 \neq (`\ne`)、大于等于号 \geq (`\ge`) 和小于等于号 \leq (`\le`)³、约等号 \approx (`\approx`)、等价 \equiv (`\equiv`)、正比 \propto (`\propto`)、相似 \sim (`\sim`) 等等。更多符号命令可参考表 4.6 以及表 4.16。

\LaTeX 还提供了自定义二元关系符的命令 `\stackrel`，用于将一个符号叠加在原有的二元关系符之上：

```
\[
f_n(x) \stackrel{*}{\approx} 1
\]
```

$$f_n(x) \overset{*}{\approx} 1$$

4.3.5 算符

\LaTeX 中的算符大多数是二元算符，除了直接用键盘可以输入的 $+$ 、 $-$ 、 $*$ 、 $/$ ，其它符号用命令输入，常用的有乘号 \times (`\times`)、除号 \div (`\div`)、点乘 \cdot (`\cdot`)、加减号 \pm (`\pm`) / 干 (`\mp`) 等等。更多符号命令可参考表 4.7 以及表 4.17。

∇ (`\nabla`) 和 ∂ (`\partial`) 也是常用的算符，虽然它们不属于二元算符。

\LaTeX 将数学函数的名称作为一个算符排版，字体为直立字体。其中有一部分符号在上下位置可以书写一些内容作为条件，类似于后文所叙述的巨算符。

³倾斜的关系符号 \leq (`\leqslant`) 和 \geq (`\geqslant`) 由 `amssymb` 提供，见表 4.16。

表 4.1: L^AT_EX 作为算符的函数名称一览

不带上下限的算符				
<code>\sin</code>	<code>\arcsin</code>	<code>\sinh</code>	<code>\exp</code>	<code>\dim</code>
<code>\cos</code>	<code>\arccos</code>	<code>\cosh</code>	<code>\log</code>	<code>\ker</code>
<code>\tan</code>	<code>\arctan</code>	<code>\tanh</code>	<code>\lg</code>	<code>\hom</code>
<code>\cot</code>	<code>\arg</code>	<code>\coth</code>	<code>\ln</code>	<code>\deg</code>
<code>\sec</code>	<code>\csc</code>			
带上下限的算符				
<code>\lim</code>	<code>\limsup</code>	<code>\liminf</code>	<code>\sup</code>	<code>\inf</code>
<code>\min</code>	<code>\max</code>	<code>\det</code>	<code>\Pr</code>	<code>\gcd</code>

```
\[
  \lim_{x \rightarrow 0}
  \frac{\sin x}{x}=1
\]
```

$$\lim_{x \rightarrow 0} \frac{\sin x}{x} = 1$$

对于求模表达式, L^AT_EX 提供了 `\bmod` 和 `\pmod` 命令, 前者相当于一个二元运算符, 后者作为同余表达式的后缀:

```
$a\bmod b \\\
x\equiv a \pmod{b}$
```

$$a \bmod b \\ x \equiv a \pmod{b}$$

如果表 4.1 中的算符不够用的话, `amsmath` 允许用户在导言区用 `\DeclareMathOperator` 定义自己的算符, 其中带星号的命令定义带上下限的算符:

```
\DeclareMathOperator{\argh}{argh}
\DeclareMathOperator*{\nut}{Nut}
```

```
\[\argh 3 = \nut_{x=1} 4x\]
```

$$\argh 3 = \Nut_{x=1} 4x$$

4.3.6 巨算符

积分号 \int (`\int`)、求和号 \sum (`\sum`) 等符号称为**巨算符**。巨算符在行内公式和行间公式的大小和形状有区别。

```
In text:
$\sum_{i=1}^n \quad
\int_0^{\frac{\pi}{2}} \quad
\oint_0^{\frac{\pi}{2}} \quad
\prod_{\epsilon} $ \\\
In display:
\[\sum_{i=1}^n \quad
\int_0^{\frac{\pi}{2}} \quad
\oint_0^{\frac{\pi}{2}} \quad
\prod_{\epsilon} \]
```

In text: $\sum_{i=1}^n \quad \int_0^{\frac{\pi}{2}} \quad \oint_0^{\frac{\pi}{2}} \quad \prod_{\epsilon}$

In display:

$$\sum_{i=1}^n \quad \int_0^{\frac{\pi}{2}} \quad \oint_0^{\frac{\pi}{2}} \quad \prod_{\epsilon}$$

巨算符的上下标位置可由 `\limits` 和 `\nolimits` 调整, 前者令巨算符类似 `lim` 或求和算符 \sum , 上下标位于上下方; 后者令巨算符类似积分号, 上下标位于右上方和右下方。

```
In text:
 $\sum\limits_{i=1}^n \quad$ 
 $\int\limits_0^{\frac{\pi}{2}} \quad$ 
 $\prod\limits_{\epsilon} \quad$ 
In display:
 $\sum\limits_{i=1}^n \quad$ 
 $\int\limits_0^{\frac{\pi}{2}} \quad$ 
 $\prod\limits_{\epsilon} \quad$ 
```

In text: $\sum_{i=1}^n \int_0^{\frac{\pi}{2}} \prod_{\epsilon}$
In display:

$$\sum_{i=1}^n \int_0^{\frac{\pi}{2}} \prod_{\epsilon}$$

`amsmath` 宏包还提供了 `\substack`, 能够在下限位置书写多行表达式; `subarray` 环境更进一步, 令多行表达式可选择居中 (c) 或左对齐 (l):

```
% \usepackage{amssymb}
\[
\sum_{\substack{0 \leq i \leq n \\ j \in \mathbb{R}}}
P(i,j) = Q(n)
\]
\[
\sum_{\begin{subarray}{l} 0 \leq i \leq n \\ j \in \mathbb{R} \end{subarray}}
P(i,j) = Q(n)
\]
```

$$\sum_{\substack{0 \leq i \leq n \\ j \in \mathbb{R}}} P(i,j) = Q(n)$$

$$\sum_{\substack{0 \leq i \leq n \\ j \in \mathbb{R}}} P(i,j) = Q(n)$$

4.3.7 数学重音和上下括号

数学符号可以像文字一样加重音, 比如求导符号 \dot{r} (`\dot{r}`)、 \ddot{r} (`\ddot{r}`)、表示向量的箭头 \vec{r} (`\vec{r}`)、表示单位向量的符号 \hat{e} (`\hat{\mathbf{e}}`) 等, 详见表 4.9。使用时要注意重音符号的作用区域, 一般应当对某个符号而不是“符号加下标”使用重音:

```
 $\bar{x}_0 \quad \bar{x}_0$ 
 $\vec{x}_0 \quad \vec{x}_0$ 
 $\hat{\mathbf{e}}_x \quad \hat{\mathbf{e}}_x$ 
```

$\bar{x}_0 \quad \bar{x}_0$
 $\vec{x}_0 \quad \vec{x}_0$
 $\hat{\mathbf{e}}_x \quad \hat{\mathbf{e}}_x$

\LaTeX 也能为多个字符加重音, 包括直接画线的 `\overline` 和 `\underline` 命令 (可叠加使用)、宽重音符号 `\widehat`、表示向量的箭头 `\overrightarrow` 等。后两者详见表 4.9 和 4.11 等。

```
 $\overline{\overline{0.3}} =$ 
 $\underline{\underline{1/3}}$ 
 $\widehat{XY}$ 
 $\overrightarrow{AB}$ 
```

$\overline{\overline{0.3}} = 1/3$
 \widehat{XY}
 \overrightarrow{AB}

`\overbrace` 和 `\underbrace` 命令用来生成上/下括号, 各自可带一个上/下标公式。

$$\underbrace{\overbrace{(a+b+c)}^6}_{\overbrace{(d+e+f)}^7} \cdot \overbrace{(d+e+f)}^7$$

$$\underbrace{(a + b + c)}_6 \cdot \underbrace{(d + e + f)}_7 = 42$$

4.3.8 箭头

常用的箭头包括 `\rightarrow` (\rightarrow , 或 `\to`)、`\leftarrow` (\leftarrow , 或 `\gets`) 等。更多箭头详见表 4.10。

amsmath 的 `\xleftarrow` 和 `\xrightarrow` 命令提供了长度可以伸展的箭头，并且可以为箭头增加上下标：

$$\left[a \xrightarrow{x+y+z} b \right]$$

$$\left[c \xrightarrow[x < y]{a*b*c} d \right]$$

$$a \xleftarrow{x+y+z} b$$

$$c \xrightarrow[x \leq y]{a*b*c} d$$

4.3.9 括号和定界符

\LaTeX 提供了多种括号和定界符表示公式块的边界, 如小括号 $()$ 、中括号 $[]$ 、大括号 $\{\}$ ($\backslash\{$ 、 $\backslash\}$)、尖括号 $\langle \rangle$ (\backslashangle \backslashrangle) 等。更多的括号/定界符命令见表 4.12 和 4.13。

$$\{a,b,c\} \not= \{a,b,c\}$$

$$a, b, c \neq \{a, b, c\}$$

使用 `\left` 和 `\right` 命令可令括号（定界符）的大小可变，在行间公式中常用。L^AT_EX 会自动根据括号内的公式大小决定定界符大小。`\left` 和 `\right` 必须成对使用。需要使用单个定界符时，另一个定界符写成 `\left.` 或 `\right.`。

$$\left[1 + \left(\frac{1}{1-x^2}\right)^3 \frac{\partial f}{\partial t}\right]_{t=0}$$

$$1 + \left(\frac{1}{1 - x^2} \right)^3 \quad \left. \frac{\partial f}{\partial t} \right|_{t=0}$$

有时我们不满意于 L^AT_EX 为我们自动调节的定界符大小。这时我们还可以用 `\big`、`\bigg` 等命令生成固定大小的定界符。更常用的形式是类似 `\left` 的 `\bigl`、`\biggl` 等，以及类似 `\right` 的 `\bigr`、`\biggr` 等（`\bigl` 和 `\bigr` 不必成对出现）。

$$\begin{aligned} & \$\Bigl((x+1)(x-1)\Bigr)^{-2}\$\\ & \$\bigl(\ \Bigr(\ \bigr(\ \Bigl(\ \quad \\ & \bigr)\ \Bigr)\ \bigr)\ \Bigl)\ \quad \\ & \big|\ \Big|\ \bigg|\ \Bigg|\ \quad \\ & \big\downarrow\ \Big\downarrow \\ & \bigg\downarrow\ \Bigg\downarrow\$ \end{aligned}$$

$$\left((x+1)(x-1)\right)^2$$

使用 `\big` 和 `\bigg` 等命令的另外一个好处是：用 `\left` 和 `\right` 分界符包裹的公式块是不允许断行的（下文提到的 `array` 或者 `aligned` 等环境视为一个公式块），所以也不允许在多行公式里跨行使用，而 `\big` 和 `\bigg` 等命令不受限制。

4.4 多行公式

4.4.1 长公式折行

通常来讲应当避免写出超过一行而需要折行的长公式。如果一定要折行的话，习惯上优先在等号之前折行，其次在加号、减号之前，再次在乘号、除号之前。其它位置应当避免折行。

`amsmath` 宏包的 `multline` 环境提供了书写折行长公式的方便环境。它允许用 `\\` 折行，将公式编号放在最后一行。多行公式的首行左对齐，末行右对齐，其余行居中。

```
\begin{multline}
a + b + c + d + e + f
+ g + h + i \\
= j + k + l + m + n \\
= o + p + q + r + s \\
= t + u + v + x + z
\end{multline}
```

$$\begin{aligned}
 a + b + c + d + e + f + g + h + i \\
 &= j + k + l + m + n \\
 &= o + p + q + r + s \\
 &= t + u + v + x + z \quad (4.2)
 \end{aligned}$$

与表格不同的是，公式的最后一行不写 `\\`，如果写了，反倒会产生一个多余的空行。

类似 `equation*`，`multline*` 环境排版不带编号的折行长公式。

4.4.2 多行公式

更多的情况是，我们需要罗列一系列公式，并令其按照等号对齐。

读者可能阅读过其它手册或者资料，知道 \LaTeX 提供了 `eqnarray` 环境。它按照等号左边——等号——等号右边呈三列对齐，但等号周围的空隙过大，加上公式编号等一些 bug，目前已不推荐使用⁴。

目前最常用的是 `align` 环境，它将公式用 `&` 隔为两部分并对齐。分隔符通常放在等号左边：

```
\begin{align}
a &= b + c \\
&= d + e
\end{align}
```

$$a = b + c \quad (4.3)$$

$$= d + e \quad (4.4)$$

`align` 环境会给每行公式都编号。我们仍然可以用 `\notag` 去掉某行的编号。在以下的例子，为了对齐等号，我们将分隔符放在右侧，并且此时需要在等号后添加一对括号 `{}` 以产生正常的间距：

```
\begin{align}
a &={}&& b + c \\
&={}&& d + e + f + g + h + i \\
&&& + j + k + l \notag \\
&&& + m + n + o \\
&={}&& p + q + r + s
\end{align}
```

$$a = b + c \quad (4.5)$$

$$\begin{aligned}
 &= d + e + f + g + h + i + j + k + l \\
 &\quad + m + n + o
 \end{aligned} \quad (4.6)$$

$$= p + q + r + s \quad (4.7)$$

`align` 还能够对齐多组公式，除等号前的 `&` 之外，公式之间也用 `&` 分隔：

```
\begin{align}
a &= 1 && b = 2 && c = 3 \\
d &= -1 && e = -2 && f = -5
\end{align}
```

$$a = 1 \quad b = 2 \quad c = 3 \quad (4.8)$$

$$d = -1 \quad e = -2 \quad f = -5 \quad (4.9)$$

⁴Lars Madsen. *Avoid eqnarray!*, <https://tug.org/pracjourn/2006-4/madsen/madsen.pdf>

如果我们不需要按等号对齐，只需罗列数个公式，`gather` 将是一个很好用的环境：

```
\begin{gather}
a = b + c \\
d = e + f + g \\
h + i = j + k \notag \\
l + m = n
\end{gather}
```

$$a = b + c \quad (4.10)$$

$$d = e + f + g \quad (4.11)$$

$$h + i = j + k$$

$$l + m = n \quad (4.12)$$

`align` 和 `gather` 有对应的不带编号的版本 `align*` 和 `gather*`。

4.4.3 公用编号的多行公式

另一个常见的需求是将多个公式组在一起公用一个编号，编号位于公式的居中位置。为此，`amsmath` 宏包提供了诸如 `aligned`、`gathered` 等环境，与 `equation` 环境套用。以 `-ed` 结尾的环境用法与前一节不以 `-ed` 结尾的环境用法一一对应。我们仅以 `aligned` 举例：

```
\begin{equation}
\begin{aligned}
a &= b + c \\
d &= e + f + g \\
h + i &= j + k \\
l + m &= n
\end{aligned}
\end{equation}
```

$$\begin{aligned} a &= b + c \\ d &= e + f + g \\ h + i &= j + k \\ l + m &= n \end{aligned} \quad (4.13)$$

`split` 环境和 `aligned` 环境用法类似，也用于和 `equation` 环境套用，区别是 `split` 只能将每行的一个公式分两栏，`aligned` 允许每行多个公式多栏。

4.5 数组和矩阵

为了排版二维数组，`LaTeX` 提供了 `array` 环境，用法与 `tabular` 环境极为类似，也需要定义列格式，并用 `\\` 换行。数组可作为一个公式块，在外套用 `\left`、`\right` 等定界符：

```
\[ \mathbf{X} = \left(
\begin{array}{cccc}
x_{11} & x_{12} & \ldots & x_{1n} \\
x_{21} & x_{22} & \ldots & x_{2n} \\
\vdots & \vdots & \ddots & \vdots \\
x_{n1} & x_{n2} & \ldots & x_{nn}
\end{array}
\right) \]
```

$$\mathbf{X} = \begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1n} \\ x_{21} & x_{22} & \cdots & x_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{nn} \end{pmatrix}$$

值得注意的是，上一节末尾介绍的 `aligned` 等环境也可以用定界符包裹。

我们还可以利用空的定界符排版出这样的效果：

```
\[ |x| = \left\{
\begin{array}{rl}
-x & \text{if } x < 0, \\
0 & \text{if } x = 0, \\
x & \text{if } x > 0.
\end{array}
\right. \]
```

$$|x| = \begin{cases} -x & \text{if } x < 0, \\ 0 & \text{if } x = 0, \\ x & \text{if } x > 0. \end{cases}$$

不过上述例子可以用 `amsmath` 提供的 `cases` 环境更轻松地完成：

```
\[ |x| =
\begin{cases}
-x & \text{if } x < 0, \\
0 & \text{if } x = 0, \\
x & \text{if } x > 0.
\end{cases} \]
```

$$|x| = \begin{cases} -x & \text{if } x < 0, \\ 0 & \text{if } x = 0, \\ x & \text{if } x > 0. \end{cases}$$

我们当然也可以用 `array` 环境排版各种矩阵。`amsmath` 宏包还直接提供了多种排版矩阵的环境，包括不带定界符的 `matrix`，以及带各种定界符的矩阵 `pmatrix` (`()`)、`bmatrix` (`[]`)、`Bmatrix` (`\{ \}`)、`vmatrix` (`| |`)、`Vmatrix` (`\| \|`)。使用这些环境时，无需给定列格式⁵：

```
\[
\begin{matrix}
1 & 2 & \quad 3 & 4
\end{matrix} \quad \quad
\begin{bmatrix}
x_{11} & x_{12} & \ldots & x_{1n} \\
x_{21} & x_{22} & \ldots & x_{2n} \\
\vdots & \vdots & \ddots & \vdots \\
x_{n1} & x_{n2} & \ldots & x_{nn}
\end{bmatrix}
\]
```

$$\begin{matrix} 1 & 2 \\ 3 & 4 \end{matrix} \quad \begin{bmatrix} x_{11} & x_{12} & \dots & x_{1n} \\ x_{21} & x_{22} & \dots & x_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \dots & x_{nn} \end{bmatrix}$$

在矩阵中的元素里排版分式时，一来要用到 `\dfrac` 等命令，二来行与行之间有可能紧贴着，这时要用到 3.6.6 小节的方法来调节间距：

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

$$\mathbf{H} = \begin{bmatrix} \frac{\partial^2 f}{\partial x^2} & \frac{\partial^2 f}{\partial x \partial y} \\ \frac{\partial^2 f}{\partial x \partial y} & \frac{\partial^2 f}{\partial y^2} \end{bmatrix}$$

4.6 公式中的间距

前文提到过，绝大部分时候，数学公式中各元素的间距是根据符号类型自动生成的，需要我们手动调整的情况极少。我们已经认识了两个生成间距的命令 `\quad` 和 `\qquad`。在公式中我们可能用到的间距还包括 `\,`、`\:`、`\;`；以及负间距 `\!`⁶。文本中的 `_` 也能使用在数学公式中。

⁵事实上这些矩阵内部也是用 `array` 环境生成的，列格式默认为 `*{<n>}{c}`，`<n>` 默认为 10；`cases` 内部是一个列格式为 `@{}l@{\quad}l@{}` 的 `array` 环境。

⁶ \TeX 2020-10-01 版本之前，`\:`、`\;` 和 `\!` 只能用于数学环境。

无额外间距	aa	$\backslash,$	\mathfrak{u}	$a a$
$\backslash quad$	\quad	$\backslash:$	\mathfrak{u}	$a a$
$\backslash qquad$	$\quad\quad$	$\backslash;$	\mathfrak{u}	$a a$
$\backslash_$	\quad	$\backslash!$	$-\mathfrak{u}$	αa

一个常见的用途是修正积分的被积函数 $f(x)$ 和微元 dx 之间的距离。注意微元里的 d 用的是直立体：

```
\[
\int_a^b f(x)\mathrm{d}x
\qquad
\int_a^b f(x)\,\mathrm{d}x
\]
```

$$\int_a^b f(x)dx \quad \int_a^b f(x) \, dx$$

另一个用途是生成多重积分号。如果我们直接连写两个 \int ，之间的间距将会过宽，此时可以使用负间距 $\!$ 修正之。不过 `amsmath` 提供了更方便的多重积分号，如二重积分 \iint 、三重积分 \iiint 等。

```
\newcommand\diff{\,\mathrm{d}}
\begin{gather*}
\int\int f(x)g(y)
\diff x \diff y \!
\int\!\!\!\int f(x)g(y)
\diff x \diff y \!
\iint f(x)g(y) \diff x \diff y \!
\iint\quad\iiint\quad\idotsint
\end{gather*}
```

$$\begin{aligned}
&\int\int f(x)g(y) \, dx \, dy \\
&\iint f(x)g(y) \, dx \, dy \\
&\iint f(x)g(y) \, dx \, dy \\
&\iint \quad \iiint \quad \int \cdots \int
\end{aligned}$$

4.7 数学符号的字体控制

4.7.1 数学字母字体

\LaTeX 允许一部分数学符号切换字体，主要是拉丁字母、数字、大写希腊字母以及重音符号等。表 4.2 给出了切换字体的命令。某一些命令需要字体宏包的支持。

```
% \usepackage{amssymb}
$\mathcal{R}$ \quad $\mathfrak{R}$
\quad $\mathbb{R}$
\[\mathcal{L}
= -\frac{1}{4}F_{\mu\nu}F^{\mu\nu}\]
$\mathfrak{su}(2)$ and
$\mathfrak{so}(3)$ Lie algebra
```

\mathcal{R} \mathfrak{R} \mathbb{R}

$$\mathcal{L} = -\frac{1}{4}F_{\mu\nu}F^{\mu\nu}$$

$\mathfrak{su}(2)$ and $\mathfrak{so}(3)$ Lie algebra

一般来说，不同的数学字体往往带有不同的语义，如矩阵、向量等常会使用粗体或粗斜体，而数集常会使用 $\mathbb{}$ 表示。出于内容与格式分离以及方便书写的考虑，可以为它们定义新的命令。具体方法详见 8.1.1 小节。

如果需要为所有的数学符号切换字体，则需要直接调用数学字体宏包（见 5.1.3 小节）。在 `xelatex` 或者 `lualatex` 编译命令下，还可以使用基于 `fontspec` 宏包的 `unicode-math` 宏包配置 Unicode 数学字体，详见 5.1.7 小节。

表 4.2: 数学字母字体

示例	命令	依赖的宏包
$ABCDEabcde1234$	<code>\mathnormal{...}</code>	
$ABCD\!Eabcde1234$	<code>\mathrm{...}</code>	
$ABCDE\!abcde1234$	<code>\mathit{...}</code>	
$\mathbf{ABCDEabcde1234}$	<code>\mathbf{...}</code>	
$ABCD\!Eabcde1234$	<code>\mathsf{...}</code>	
$ABCDE\!abcde1234$	<code>\mathtt{...}</code>	
\mathcal{ABCDE}	<code>\mathcal{...}</code>	仅提供大写字母
$\mathcal{A}\mathcal{B}\mathcal{C}\mathcal{D}\mathcal{E}$	<code>\mathcal{...}</code>	eucal 仅提供大写字母
$\mathscr{A}\mathscr{B}\mathscr{C}\mathscr{D}\mathscr{E}$	<code>\mathscr{...}</code>	mathrsfs 仅提供大写字母
$\mathfrak{A}\mathfrak{B}\mathfrak{C}\mathfrak{D}\mathfrak{E}abcde1234$	<code>\mathfrak{...}</code>	amssymb 或 eufrak
$\mathbb{A}\mathbb{B}\mathbb{C}\mathbb{D}\mathbb{E}$	<code>\mathbb{...}</code>	amssymb 仅提供大写字母

4.7.2 加粗的数学符号

表 4.2 中的 `\mathbf` 命令只能获得直立、加粗的字母。如果想得到粗斜体⁷，可以使用 `ams-math` 宏包提供的 `\boldsymbol` 命令：

```
$\mu, M \quad \quad \quad \backslashboldsymbol{\mu}, \backslashboldsymbol{M}$
```

$\mu, M \quad \quad \mu, M$

也可以使用 `bm` 宏包提供的 `\bm` 命令：

```
% \usepackage{bm}
$\mu, M \quad \quad \quad \backslashbm{\mu}, \backslashbm{M}$
```

$\mu, M \quad \quad \mu, M$

在 \LaTeX 默认的数学字体中，一些符号本身并没有粗体版本，使用 `\boldsymbol` 也得不到粗体。此时 `\bm` 命令会生成“伪粗体”，尽管效果比较粗糙，但在某些时候也不失为一种解决方案。

4.7.3 数学符号的尺寸

数学符号按照符号排版的位置规定尺寸，从大到小包括行间公式尺寸、行内公式尺寸、上下标尺寸、次级上下标尺寸。除了字号有别之外，行间和行内公式尺寸下的巨算符也使用不一样的大小。 \LaTeX 为每个数学尺寸指定了一个切换的命令，见 4.3。

表 4.3: 数学符号尺寸

命令	尺寸	示例
<code>\displaystyle</code>	行间公式尺寸	$\sum a$
<code>\textstyle</code>	行内公式尺寸	$\sum a$
<code>\scriptstyle</code>	上下标尺寸	a
<code>\scriptscriptstyle</code>	次级上下标尺寸	a

我们通过以下示例对比行间公式和行内公式的区别。在分式中，分子分母默认为行内公式尺寸，示例中将分母切换到行间公式尺寸：

⁷国内使用粗斜体符号表示矢量，见 GB/T 3102.11—1993。

```
\[
r = \frac
{\sum_{i=1}^n (x_i - x)(y_i - y)}
{\displaystyle \left[
\sum_{i=1}^n (x_i - x)^2
\sum_{i=1}^n (y_i - y)^2
\right]^{1/2}}
\]
```

$$r = \frac{\sum_{i=1}^n (x_i - x)(y_i - y)}{\left[\sum_{i=1}^n (x_i - x)^2 \sum_{i=1}^n (y_i - y)^2 \right]^{1/2}}$$

4.8 定理环境

4.8.1 L^AT_EX 原始的定理环境

使用 L^AT_EX 排版数学和其他科技文档时，会接触到大量的定理、证明等内容。L^AT_EX 提供了一个基本的命令 `\newtheorem` 提供定理环境的定义：

```
\newtheorem{<theorem environment>}{<title>}[<section-level>]
\newtheorem{<theorem environment>}[<counter>]{<title>}
```

`<theorem environment>` 为定理环境的名称。原始的 L^AT_EX 里没有现成的定理环境，不加定义而直接使用很可能会出错。`<title>` 是定理环境的标题（“定理”，“公理”等）。

定理的序号由两个可选参数之一决定，它们不能同时使用：

- `<section level>` 为章节级别，如 `chapter`、`section` 等，定理序号成为章节的下一级序号；
- `<counter>` 为用 `\newcounter` 自定义的计数器名称（详见 8.3 节），定理序号由这个计数器管理。

如果两个可选参数都不用的话，则使用默认的和定理环境同名的计数器。

在以下示例代码中，我们定义了一个 `mythm` 环境，其序号设为 `section` 的下一级序号。注意 `mythm` 环境的可选参数以及 `\label` 的用法：

```
\newtheorem{mythm}{My Theorem}[section]
\begin{mythm}\label{thm:light}
The light speed in vacuum
is $299,792,458\,\mathrm{m/s}$ .
\end{mythm}
\begin{mythm}[Energy-momentum relation]
The relationship of energy,
momentum and mass is

$$E^2 = m_0^2 c^4 + p^2 c^2$$

where  $c$  is the light speed
described in theorem \ref{thm:light}.
\end{mythm}
```

My Theorem 4.8.1. *The light speed in vacuum is 299,792,458 m/s.*

My Theorem 4.8.2 (Energy-momentum relation). *The relationship of energy, momentum and mass is*

$$E^2 = m_0^2 c^4 + p^2 c^2$$

where c is the light speed described in theorem 4.8.1.

4.8.2 amsthm 宏包

L^AT_EX 默认的定理环境格式为粗体标签、斜体内容、定理名用小括号包裹。如果需要修改格式，则要依赖其它的宏包，如 `amsthm`、`ntheorem` 等等。本小节简单介绍一下 `amsthm` 的用法。

`amsthm` 提供了 `\theoremstyle` 命令支持定理格式的切换, 在用 `\newtheorem` 命令定义定理环境之前使用。`amsthm` 预定义了三种格式用于 `\theoremstyle`: `plain` 和 \LaTeX 原始的格式一致; `definition` 使用粗体标签、正体内容; `remark` 使用斜体标签、正体内容。

另外 `amsthm` 还支持用带星号的 `\newtheorem*` 定义不带序号的定理环境:

```
\theoremstyle{definition} \newtheorem{law}{Law}
\theoremstyle{plain} \newtheorem{jury}[law]{Jury}
\theoremstyle{remark} \newtheorem*{mar}{Margaret}
```

以上例子定义的 `jury` 环境与 `law` 环境共用编号, `mar` 环境不编号:

```
\begin{law}\label{law:box}
Don't hide in the witness box.
\end{law}
\begin{jury}[The Twelve]
It could be you! So beware and
see law~\ref{law:box}.\end{jury}
\begin{jury}
You will disregard the last
statement.\end{jury}
\begin{mar}No, No, No\end{mar}
\begin{mar}Denis!\end{mar}
```

Law 1. Don't hide in the witness box.

Jury 2 (The Twelve). *It could be you! So beware and see law 1.*

Jury 3. *You will disregard the last statement.*

Margaret. No, No, No

Margaret. Denis!

`amsthm` 还支持使用 `\newtheoremstyle` 命令自定义定理格式, 更为方便使用的是 `ntheorem` 宏包。感兴趣的读者可参阅它们的帮助文档。

4.8.3 证明环境和证毕符号

`amsthm` 还提供了一个 `proof` 环境用于排版定理的证明过程。`proof` 环境末尾自动加上一个 □ 证毕符号:

```
\begin{proof}
For simplicity, we use
\[
E=mc^2
\]
That's it.
\end{proof}
```

Proof. For simplicity, we use

$$E = mc^2$$

That's it. □

如果行末是一个不带编号的公式, □ 符号会另起一行, 这时可使用 `\qedhere` 命令将 □ 符号放在公式末尾:

```
\begin{proof}
For simplicity, we use
\[
E=mc^2 \qedhere
\]
\end{proof}
```

Proof. For simplicity, we use

$$E = mc^2 \quad \square$$

`\qedhere` 对于 `align*` 等环境也有效:


```
\begin{proof}
Assuming  $\gamma$ 
=  $1/\sqrt{1-v^2/c^2}$ , then
\begin{align*}
E &= \gamma m_0 c^2 \\
p &= \gamma m_0 v \quad \text{\qedhere}
\end{align*}
\end{proof}
```

Proof. Assuming $\gamma = 1/\sqrt{1 - v^2/c^2}$, then

$$E = \gamma m_0 c^2$$

$$p = \gamma m_0 v$$

□

在使用带编号的公式时，建议最好不要在公式末尾使用 `\qedhere` 命令。对带编号的公式使用 `\qedhere` 命令会使 □ 符号放在一个难看的位置，紧贴着公式：

```
\begin{proof}
For simplicity, we use
\begin{equation}
E=mc^2.\text{\qedhere}
\end{equation}
\end{proof}
```

Proof. For simplicity, we use

$$E = mc^2. \quad (4.14)$$

□

在 `align` 等环境中使用 `\qedhere` 命令会使 □ 盖掉公式的编号；使用 `equation` 嵌套 `aligned` 等环境时，`\qedhere` 命令会将 □ 直接放在公式后。这些位置都不太正常。

证毕符号 □ 本身被定义在命令 `\qedsymbol` 中，如果有使用实心符号作为证毕符号的需求，需要自行用 `\renewcommand` 命令修改（用法见 8.1.1 小节）⁸。我们可以利用在 3.8.4 小节介绍的标尺盒子来生成一个适当大小的“实心矩形”：

```
\renewcommand{\qedsymbol}{%
  {\rule{1ex}{1.5ex}}
\begin{proof}
For simplicity, we use
\[
E=mc^2 \text{\qedhere}
\]
\end{proof}
```

Proof. For simplicity, we use

$$E = mc^2$$

■

⁸注意，这个改法只对 **amsthm** 宏包适用。其它宏包如 `ntheorem` 等须参考帮助文档里提供的修改方法。

4.9 符号表

有几个注意事项：

1. 蓝色的命令依赖 `amsmath` 宏包 (非 `amssymb` 宏包)；
2. 带有角标^ℓ的符号命令依赖 `latexsym` 宏包。

4.9.1 L^AT_EX 普通符号

表 4.4: 文本/数学模式通用符号

这些符号可用于文本和数学模式。

{	\{	}	\}	\$	\\$	%	\%
†	\dag	§	\S	©	\copyright	...	\dots
‡	\ddag	¶	\P	£	\pounds		

表 4.5: 希腊字母

`\Alpha`, `\Beta` 等希腊字母符号不存在, 因为它们和拉丁字母 A,B 等一模一样; 小写字母里也不存在 `\omicron`, 直接用拉丁字母 *o* 代替。

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

表 4.6: 二元关系符

所有的二元关系符都可以加 `\not` 前缀得到相反意义的关系符, 例如 `\not=` 就得到不等号 (同 `\ne`)。

$<$	<code><</code>	$>$	<code>></code>	$=$	<code>=</code>
\leq	<code>\leq</code> or <code>\le</code>	\geq	<code>\geq</code> or <code>\ge</code>	\equiv	<code>\equiv</code>
\ll	<code>\ll</code>	\gg	<code>\gg</code>	\doteq	<code>\doteq</code>
\prec	<code>\prec</code>	\succ	<code>\succ</code>	\sim	<code>\sim</code>
\preceq	<code>\preceq</code>	\succeq	<code>\succeq</code>	\simeq	<code>\simeq</code>
\subset	<code>\subset</code>	\supset	<code>\supset</code>	\approx	<code>\approx</code>
\subseteq	<code>\subseteq</code>	\supseteq	<code>\supseteq</code>	\cong	<code>\cong</code>
\sqsubset^ℓ	<code>\sqsubset^\ell</code>	\sqsupset^ℓ	<code>\sqsupset^\ell</code>	\Join^ℓ	<code>\Join^\ell</code>
\sqsubseteq	<code>\sqsubseteq</code>	\sqsupseteq	<code>\sqsupseteq</code>	\bowtie	<code>\bowtie</code>
\in	<code>\in</code>	\ni, owns	<code>\ni, \text{owns}</code>	\propto	<code>\propto</code>
\vdash	<code>\vdash</code>	\dashv	<code>\dashv</code>	\models	<code>\models</code>
$ $	<code>\mid</code>	\parallel	<code>\parallel</code>	\perp	<code>\perp</code>
\smile	<code>\smile</code>	\frown	<code>\frown</code>	\asymp	<code>\asymp</code>
$:$	<code>:</code>	\notin	<code>\notin</code>	\neq or <code>\ne</code>	<code>\neq</code> or <code>\ne</code>

表 4.7: 二元运算符

$+$	<code>+</code>	$-$	<code>-</code>	
\pm	<code>\pm</code>	\mp	<code>\mp</code>	\triangleleft <code>\triangleleft</code>
\cdot	<code>\cdot</code>	\div	<code>\div</code>	\triangleright <code>\triangleright</code>
\times	<code>\times</code>	\setminus	<code>\setminus</code>	\star <code>\star</code>
\cup	<code>\cup</code>	\cap	<code>\cap</code>	$*$ <code>\ast</code>
\sqcup	<code>\sqcup</code>	\sqcap	<code>\sqcap</code>	\circ <code>\circ</code>
\vee, lor	<code>\vee, \text{lor}</code>	\wedge, land	<code>\wedge, \text{land}</code>	\bullet <code>\bullet</code>
\oplus	<code>\oplus</code>	\ominus	<code>\ominus</code>	\diamond <code>\diamond</code>
\odot	<code>\odot</code>	\oslash	<code>\oslash</code>	\uplus <code>\uplus</code>
\otimes	<code>\otimes</code>	\bigcirc	<code>\bigcirc</code>	\amalg <code>\amalg</code>
\bigtriangleup	<code>\bigtriangleup</code>	\bigtriangledown	<code>\bigtriangledown</code>	\dagger <code>\dagger</code>
\lhd^ℓ	<code>\lhd^\ell</code>	\rhd^ℓ	<code>\rhd^\ell</code>	\ddagger <code>\ddagger</code>
\unlhd^ℓ	<code>\unlhd^\ell</code>	\unrhd^ℓ	<code>\unrhd^\ell</code>	\wr <code>\wr</code>

表 4.8: 巨算符

Σ	\sum	<code>\sum</code>	\cup	\bigcup	<code>\bigcup</code>	\vee	\bigvee	<code>\bigvee</code>
\prod	\prod	<code>\prod</code>	\cap	\bigcap	<code>\bigcap</code>	\wedge	\bigwedge	<code>\bigwedge</code>
\coprod	\coprod	<code>\coprod</code>	\sqcup	\bigsqcup	<code>\bigsqcup</code>	\oplus	\bigoplus	<code>\bigoplus</code>
\int	\int	<code>\int</code>	\oint	\oint	<code>\oint</code>	\odot	\bigodot	<code>\bigodot</code>
\oplus	\oplus	<code>\bigoplus</code>	\otimes	\bigotimes	<code>\bigotimes</code>			
\iint	\iint	<code>\iint</code>	\iiint	\iiint	<code>\iiint</code>	\iiint	\iiint	<code>\iiint</code>
$\int \cdots \int$	$\int \cdots \int$	<code>\idotsint</code>						

表 4.9: 数学重音符号

最后一个 `\wideparen` 依赖 `yhmath` 宏包。

\hat{a}	<code>\hat{a}</code>	\check{a}	<code>\check{a}</code>	\tilde{a}	<code>\tilde{a}</code>
\acute{a}	<code>\acute{a}</code>	\grave{a}	<code>\grave{a}</code>	\breve{a}	<code>\breve{a}</code>
\bar{a}	<code>\bar{a}</code>	\vec{a}	<code>\vec{a}</code>	\mathring{a}	<code>\mathring{a}</code>
\dot{a}	<code>\dot{a}</code>	\ddot{a}	<code>\ddot{a}</code>	\dddot{a}	<code>\dddot{a}</code>
\ddot{a}	<code>\dddot{a}</code>				
\widehat{AAA}	<code>\widehat{AAA}</code>	\widetilde{AAA}	<code>\widetilde{AAA}</code>	\wideparen{AAA}	<code>\wideparen{AAA}</code>

表 4.10: 箭头

\leftarrow	<code>\leftarrow</code> or <code>\gets</code>	\longleftarrow	<code>\longleftarrow</code>
\rightarrow	<code>\rightarrow</code> or <code>\to</code>	\longrightarrow	<code>\longrightarrow</code>
\leftrightarrow	<code>\leftrightarrow</code>	\longleftrightarrow	<code>\longleftrightarrow</code>
\Lleftarrow	<code>\Lleftarrow</code>	\Longleftarrow	<code>\Longleftarrow</code>
\Rrightarrow	<code>\Rrightarrow</code>	\Longrightarrow	<code>\Longrightarrow</code>
\Leftrightarrow	<code>\Leftrightarrow</code>	\Longleftrightarrow	<code>\Longleftrightarrow</code>
\mapsto	<code>\mapsto</code>	\longmapsto	<code>\longmapsto</code>
\hookleftarrow	<code>\hookleftarrow</code>	\hookrightarrow	<code>\hookrightarrow</code>
\leftharpoonup	<code>\leftharpoonup</code>	\rightharpoonup	<code>\rightharpoonup</code>
\leftharpoondown	<code>\leftharpoondown</code>	\rightharpoondown	<code>\rightharpoondown</code>
\rightleftharpoons	<code>\rightleftharpoons</code>	\iff	<code>\iff</code>
\uparrow	<code>\uparrow</code>	\downarrow	<code>\downarrow</code>
\updownarrow	<code>\updownarrow</code>	\Uparrow	<code>\Uparrow</code>
\Downarrow	<code>\Downarrow</code>	\Updownarrow	<code>\Updownarrow</code>
\nearrow	<code>\nearrow</code>	\searrow	<code>\searrow</code>
\swarrow	<code>\swarrow</code>	\nwarrow	<code>\nwarrow</code>
\leadsto	<code>\leadsto</code> ^ℓ		

表 4.11: 作为重音的箭头符号

\overrightarrow{AB}	<code>\overrightarrow{AB}</code>	$\underline{\overrightarrow{AB}}$	<code>\underrightarrow{AB}</code>
\overleftarrow{AB}	<code>\overleftarrow{AB}</code>	$\underline{\overleftarrow{AB}}$	<code>\underleftarrow{AB}</code>
\overleftrightarrow{AB}	<code>\overleftrightarrow{AB}</code>	$\underline{\overleftrightarrow{AB}}$	<code>\underleftrightarrow{AB}</code>

表 4.12: 定界符

`amsmath` 还定义了 `\lvert`、`\rvert` 和 `\lVert`、`\rVert`，分别作为 `\vert` 和 `\Vert` 对应的开符号（左侧）和闭符号（右侧）的命令。

<code>(</code>	<code>(</code>	<code>)</code>	<code>)</code>	\uparrow	<code>\uparrow</code>	\downarrow	<code>\downarrow</code>
<code>[</code>	<code>[</code> or <code>\lbrack</code>	<code>]</code>	<code>]</code> or <code>\rbrack</code>	\Uparrow	<code>\Uparrow</code>	\Downarrow	<code>\Downarrow</code>
<code>{</code>	<code>\{</code> or <code>\lbrace</code>	<code>}</code>	<code>\}</code> or <code>\rbrace</code>	\Updownarrow	<code>\updownarrow</code>	\Updownarrow	<code>\Updownarrow</code>
<code> </code>	<code> </code> or <code>\vert</code>	<code> </code>	<code>\ </code> or <code>\Vert</code>	\lceil	<code>\lceil</code>	\rceil	<code>\rceil</code>
<code>\langle</code>	<code>\langle</code> or <code>\langle</code>	<code>\rangle</code>	<code>\rangle</code> or <code>\rangle</code>	\lfloor	<code>\lfloor</code>	\rfloor	<code>\rfloor</code>
<code>/</code>	<code>/</code>	<code>\</code>	<code>\</code> or <code>\backslash</code>				

表 4.13: 用于行间公式的大定界符

$\left(\left(\right.\right)$	<code>\lgroup</code>	$\right)$	<code>\rgroup</code>	$\left.\left.\right)$	<code>\lmoustache</code>
$\left \right $	<code>\arrowvert</code>	$\left.\right \right $	<code>\Arrowvert</code>	$\left \right $	<code>\bracevert</code>
$\left.\right)\right)$					<code>\rmoustache</code>

表 4.14: 其他符号

\dots	<code>\dots</code>	\cdots	<code>\cdots</code>	\vdots	<code>\vdots</code>	\ddots	<code>\ddots</code>
\hbar	<code>\hbar</code>	\imath	<code>\imath</code>	\jmath	<code>\jmath</code>	ℓ	<code>\ell</code>
\Re	<code>\Re</code>	\Im	<code>\Im</code>	\aleph	<code>\aleph</code>	\wp	<code>\wp</code>
\forall	<code>\forall</code>	\exists	<code>\exists</code>	\mho	<code>\mho</code>	∂	<code>\partial</code>
$'$	<code>'</code>	\prime	<code>\prime</code>	\emptyset	<code>\emptyset</code>	∞	<code>\infty</code>
∇	<code>\nabla</code>	\triangle	<code>\triangle</code>	\Box	<code>\Box</code>	\Diamond	<code>\Diamond</code>
\bot	<code>\bot</code>	\top	<code>\top</code>	\angle	<code>\angle</code>	\surd	<code>\surd</code>
\diamondsuit	<code>\diamondsuit</code>	\heartsuit	<code>\heartsuit</code>	\clubsuit	<code>\clubsuit</code>	\spadesuit	<code>\spadesuit</code>
\neg	<code>\neg</code> or <code>\lnot</code>	\flat	<code>\flat</code>	\natural	<code>\natural</code>	\sharp	<code>\sharp</code>

4.9.2 \mathcal{AMS} 符号

本小节所有符号依赖 `amssymb` 宏包。

表 4.15: \mathcal{AMS} 希腊字母和希伯来字母

\digamma	<code>\digamma</code>	\varkappa	<code>\varkappa</code>	\beth	<code>\beth</code>	\gimel	<code>\gimel</code>	\daleth	<code>\daleth</code>
------------	-----------------------	-------------	------------------------	---------	--------------------	----------	---------------------	-----------	----------------------

表 4.16: \mathcal{AMS} 二元关系符

\lessdot	<code>\lessdot</code>	\gtrdot	<code>\gtrdot</code>	\doteqdot	<code>\doteqdot</code>
\leqslant	<code>\leqslant</code>	\geqslant	<code>\geqslant</code>	\risingdotseq	<code>\risingdotseq</code>
\eqslantless	<code>\eqslantless</code>	\eqslantgtr	<code>\eqslantgtr</code>	\fallingdotseq	<code>\fallingdotseq</code>
\leqq	<code>\leqq</code>	\geqq	<code>\geqq</code>	\eqcirc	<code>\eqcirc</code>
\lll or \llless	<code>\lll</code> or <code>\llless</code>	\ggg	<code>\ggg</code>	\circeq	<code>\circeq</code>
\lesssim	<code>\lesssim</code>	\gtrsim	<code>\gtrsim</code>	\triangleq	<code>\triangleq</code>
\lessapprox	<code>\lessapprox</code>	\gtrapprox	<code>\gtrapprox</code>	\bumpeq	<code>\bumpeq</code>
\lessgtr	<code>\lessgtr</code>	\gtrless	<code>\gtrless</code>	\Bumpeq	<code>\Bumpeq</code>
\lesseqgtr	<code>\lesseqgtr</code>	\gtreqless	<code>\gtreqless</code>	\thicksim	<code>\thicksim</code>
\lesseqqgtr	<code>\lesseqqgtr</code>	\gtreqqless	<code>\gtreqqless</code>	\thickapprox	<code>\thickapprox</code>
\preccurlyeq	<code>\preccurlyeq</code>	\succcurlyeq	<code>\succcurlyeq</code>	\approxeq	<code>\approxeq</code>
\curlyeqprec	<code>\curlyeqprec</code>	\curlyeqsucc	<code>\curlyeqsucc</code>	\backsim	<code>\backsim</code>
\precsim	<code>\precsim</code>	\succsim	<code>\succsim</code>	\backsimeq	<code>\backsimeq</code>
\precapprox	<code>\precapprox</code>	\succapprox	<code>\succapprox</code>	\vDash	<code>\vDash</code>
\subseteq	<code>\subseteq</code>	\supseteq	<code>\supseteq</code>	\Vdash	<code>\Vdash</code>
\shortparallel	<code>\shortparallel</code>	\Supset	<code>\Supset</code>	\Vvdash	<code>\Vvdash</code>
\blacktriangleleft	<code>\blacktriangleleft</code>	\sqsupset	<code>\sqsupset</code>	\backepsilon	<code>\backepsilon</code>
\vartriangleright	<code>\vartriangleright</code>	\because	<code>\because</code>	\varpropto	<code>\varpropto</code>
\blacktriangleright	<code>\blacktriangleright</code>	\Subset	<code>\Subset</code>	\between	<code>\between</code>
\trianglerighteq	<code>\trianglerighteq</code>	\smallfrown	<code>\smallfrown</code>	\pitchfork	<code>\pitchfork</code>
\vartriangleleft	<code>\vartriangleleft</code>	\shortmid	<code>\shortmid</code>	\smallsmile	<code>\smallsmile</code>
\trianglelefteq	<code>\trianglelefteq</code>	\therefore	<code>\therefore</code>	\sqsubset	<code>\sqsubset</code>

表 4.17: \mathcal{AMS} 二元运算符

$\dot{+}$	<code>\dotplus</code>	\cdot	<code>\centerdot</code>		
\ltimes	<code>\ltimes</code>	\rtimes	<code>\rtimes</code>	\div	<code>\divideontimes</code>
\mathcal{U}	<code>\doublecup</code>	\mathcal{M}	<code>\doublecap</code>	\smallsetminus	<code>\smallsetminus</code>
\veebar	<code>\veebar</code>	$\bar{\wedge}$	<code>\barwedge</code>	$\overline{\wedge}$	<code>\doublebarwedge</code>
\boxplus	<code>\boxplus</code>	\boxminus	<code>\boxminus</code>	\ominus	<code>\circleddash</code>
\boxtimes	<code>\boxtimes</code>	\boxdot	<code>\boxdot</code>	\odot	<code>\circledcirc</code>
\intercal	<code>\intercal</code>	\circledast	<code>\circledast</code>	\times	<code>\rightthreetimes</code>
\curlyvee	<code>\curlyvee</code>	\curlywedge	<code>\curlywedge</code>	\times	<code>\leftthreetimes</code>

表 4.18: \mathcal{AMS} 箭头

\dashleftarrow	<code>\dashleftarrow</code>	\dashrightarrow	<code>\dashrightarrow</code>
\leftrightsquigarrow	<code>\leftrightsquigarrow</code>	\rightleftarrows	<code>\rightleftarrows</code>
\leftrightsquigarrow	<code>\leftrightsquigarrow</code>	\rightleftarrows	<code>\rightleftarrows</code>
\Lleftarrow	<code>\Lleftarrow</code>	\Rrightarrow	<code>\Rrightarrow</code>
\twoheadleftarrow	<code>\twoheadleftarrow</code>	\twoheadrightarrow	<code>\twoheadrightarrow</code>
\leftarrowtail	<code>\leftarrowtail</code>	\rightarrowtail	<code>\rightarrowtail</code>
\leftrightharpoons	<code>\leftrightharpoons</code>	\rightleftharpoons	<code>\rightleftharpoons</code>
\Lsh	<code>\Lsh</code>	\Rsh	<code>\Rsh</code>
\looparrowleft	<code>\looparrowleft</code>	\looparrowright	<code>\looparrowright</code>
\curvearrowleft	<code>\curvearrowleft</code>	\curvearrowright	<code>\curvearrowright</code>
\circlearrowleft	<code>\circlearrowleft</code>	\circlearrowright	<code>\circlearrowright</code>
\multimap	<code>\multimap</code>	\Uparrow	<code>\uparrows</code>
\downdownarrows	<code>\downdownarrows</code>	\Uparrow	<code>\upharpoonleft</code>
\upharpoonright	<code>\upharpoonright</code>	\Downarrow	<code>\downharpoonright</code>
\rightsquigarrow	<code>\rightsquigarrow</code>	\leftrightsquigarrow	<code>\leftrightsquigarrow</code>

表 4.19: \mathcal{AMS} 反义二元关系符和箭头

\nless	\ngtr	\varsubsetneqq
\lneq	\gneq	\varsupsetneqq
\nleq	\ngeq	\nsubseteq
\nleqslant	\ngeqslant	\nsupseteq
\lneqq	\gneqq	\nmid
\lvertneqq	\gvertneqq	\nparallel
\nleqq	\ngeqq	\nshortmid
\lnsim	\gnsim	\nshortparallel
\lnapprox	\gnapprox	\nsim
\nprec	\nsucc	\ncong
\npreceq	\nsucceq	\nvdash
\precneqq	\succneqq	\nvDash
\precnsim	\succnsim	\nVdash
\precnapprox	\succnapprox	\nVDash
\subsetneq	\supsetneq	\ntriangleleft
\varsubsetneq	\varsupsetneq	\ntriangleright
\nsubseteq	\nsupseteq	\ntrianglelefteq
\subsetneqq	\supsetneqq	\ntrianglerighteq
\nleftarrow	\rightarrow	\nleftrightarrow
\nLeftarrow	\nrightarrow	\nLeftrightarrow

表 4.20: \mathcal{AMS} 定界符

\ulcorner	\urcorner	\llcorner	\lrcorner
-------------	-------------	-------------	-------------

表 4.21: \mathcal{AMS} 其它符号

\hbar	\hslash	\Bbbk
\square	\blacksquare	\textcircled{S}
\vartriangle	\blacktriangle	\complement
\triangledown	\blacktriangledown	\Game
\lozenge	\blacklozenge	\bigstar
\angle	\measuredangle	\backprime
\diagup	\diagdown	\varnothing
\nexists	\Finv	\mho
\eth	\sphericalangle	

第五章 排版样式设定

至此你已经基本学会排版内容丰富的文档，标题、目录、章节、公式、列表、图片、表格等等应有尽有。但是你可能已经有点不甘心了，因为似乎你排版出来的文档是千篇一律的模样—— \LaTeX 默认的字体、单调的页眉页脚、不太令你满意的页边距，等等。本章的内容将带你一览如何修改 \LaTeX 的排版样式。

5.1 字体和字号

\LaTeX 根据文档的逻辑结构（章节、脚注等）来选择默认的字体样式以及字号。需要更改字体样式或字号的话，可以使用表 5.1 和表 5.2 中列出的命令。

```
{\small The small and  
\textbf{bold} Romans ruled}  
{\Large all of great big  
\itshape Italy}.
```

The small and **bold** Romans ruled all of
great big *Italy*.

\LaTeX 2_ϵ 相比于较早的 \LaTeX 版本（2.09 版或更早）在字体样式和字号的设定上有很大改进，令字体的各种属性相互独立（“正交”），用户可以改变字体的大小，而仍然保留字体原有的粗体或者斜体的特性。

5.1.1 字体样式

\LaTeX 提供了两组修改字体的命令，见表 5.1。其中诸如 `\bfseries` 形式的命令将会影响之后所有的字符，如果想要让它在局部生效，需要用花括号**分组**，也就是写成 `{\bfseries <some text>}` 这样的形式；对应的 `\textbf` 形式带一个参数，只改变参数内部的字体，更为常用。

在公式中，直接使用 `\textbf` 等命令不会起效，甚至报错。 \LaTeX 提供了修改数学字母样式的命令，如 `\mathbf` 等，详见 4.7.1 小节。

5.1.2 字号

字号命令实际大小依赖于所使用的文档类及其选项。表 5.3 列出了这些命令在标准文档类中的绝对大小，单位为 pt。

使用字号命令的时候，通常也需要用花括号进行分组，如同 `\rmfamily` 那样。

```
He likes {\LARGE large and  
\small small} letters.
```

He likes large and small letters.

\LaTeX 还提供了一个基础的命令 `\fontsize` 用于设定任意大小的字号：

```
\fontsize{<size>}{<base line-skip>}
```

表 5.1: 字体命令

<code>\rmfamily</code>	<code>\textrm{...}</code>	roman	衬线字体（罗马体）
<code>\sffamily</code>	<code>\textsf{...}</code>	sans serif	无衬线字体
<code>\ttfamily</code>	<code>\texttt{...}</code>	typewriter	等宽字体
<code>\mdseries</code>	<code>\textmd{...}</code>	medium	正常粗细（中等）
<code>\bfseries</code>	<code>\textbf{...}</code>	bold face	粗体
<code>\upshape</code>	<code>\textup{...}</code>	upright	直立体
<code>\itshape</code>	<code>\textit{...}</code>	<i>italic</i>	意大利斜体
<code>\slshape</code>	<code>\textsl{...}</code>	<i>slanted</i>	倾斜体
<code>\scshape</code>	<code>\textsc{...}</code>	SMALL CAPS	小型大写字母
<code>\em</code>	<code>\emph{...}</code>	<i>emphasized</i>	强调，默认斜体
<code>\normalfont</code>	<code>\textnormal{...}</code>	normal font	默认字体

表 5.2: 字号

<code>\tiny</code>	tiny font	<code>\Large</code>	larger font
<code>\scriptsize</code>	very small font	<code>\LARGE</code>	very large font
<code>\footnotesize</code>	quite small font	<code>\huge</code>	huge
<code>\small</code>	small font	<code>\Huge</code>	largest
<code>\normalsize</code>	normal font		
<code>\large</code>	large font		

表 5.3: 标准文档类中的字号大小

字号	10pt 选项（默认）	11pt 选项	12pt 选项
<code>\tiny</code>	5pt	6pt	6pt
<code>\scriptsize</code>	7pt	8pt	8pt
<code>\footnotesize</code>	8pt	9pt	10pt
<code>\small</code>	9pt	10pt	10.95pt
<code>\normalsize</code>	10pt	10.95pt	12pt
<code>\large</code>	12pt	12pt	14.4pt
<code>\Large</code>	14.4pt	14.4pt	17.28pt
<code>\LARGE</code>	17.28pt	17.28pt	20.74pt
<code>\huge</code>	20.74pt	20.74pt	24.88pt
<code>\Huge</code>	24.88pt	24.88pt	24.88pt

`\fontsize` 用到两个参数, $\langle size \rangle$ 为字号, $\langle base\ line\ skip \rangle$ 为基础行距。表 5.3 中的命令也都各自设定了与字号对应的基础行距, 大小为字号的 1.2 倍。如果不是在导言区, `\fontsize` 的设定需要 `\selectfont` 命令才能立即生效, 而表 5.2 的字号设定都是立即生效的。

5.1.3 选用字体宏包

至此已经介绍了如何改变字体样式如粗体、斜体等等, 以及如何改变字号, 但你依然用着 L^AT_EX 默认的那套、由高德纳设计制作的 Computer Modern 字体。有的人可能很喜欢 Times 和 Palatino, 或者更好看的字体。这些字体样式的自由设置在 L^AT_EX 里还不太容易。

幸好大部分时候, 许多字体宏包为我们完成了整套配置, 我们可以在调用宏包之后, 照常使用 `\bfseries` 或 `\ttfamily` 等我们熟悉的命令。表 5.4 列出了较为常用的字体宏包, 其中相当多的宏包还配置了数学字体, 或者文本、数学字体兼而有之。更多的字体配置参考 [16, 19]。

5.1.4 字体编码

字体编码对于 L^AT_EX 用户来讲是一个比较晦涩的概念。它规定了一个字体里包含的符号, 并将若干符号用 L^AT_EX 命令定义。注意字体编码与我们在 2.1.1 等小节叙述的 ASCII 编码等并非一一对应。

常见的正文字体编码有 OT1 和 T1 等。L^AT_EX 默认使用兼容 plain T_EX 的 OT1 编码, 使用起来有诸多限制: 高德纳在设计 Computer Modern 字体时认为一些符号, 如大于号、小于号等, 原则上都应该在公式里出现, 所以在正文字体里这些符号所在的位置被其它符号所占据 (OT1 字体编码、`\rmfamily` 和 `\sffamly` 字体族下, `<` 和 `>` 排版 `ı` 和 `ı` 两个倒立的标点符号, 正常的大于号和小于号可用命令 `\textgreater` 和 `\textless` 输入; `\ttfamily` 字体族下是正常的大于号和小于号)。扩展的 T1 字体编码则更加靠近 ASCII 文本编码, 不会出现上述的大于号、小于号的问题。

切换字体编码要用到 `fontenc` 宏包:

```
\usepackage[T1]{fontenc}
```

`fontenc` 宏包是用来配合传统的 L^AT_EX 字体的, 如表 5.4 中的一些传统字体宏包。如果使用 `xelatex` 编译方式, 并使用 `fontspec` 宏包调用 `ttf` 或 `otf` 格式字体, 就不要再使用 `fontenc` 宏包。使用表 5.4 中的字体宏包之前最好查看一下宏包的帮助文档, 了解使用方法和注意事项。

5.1.5 使用 `fontspec` 宏包更改字体

`xelatex` 和 `lualatex` 编译命令能够支持直接调用系统和 T_EX 发行版中的 `.ttf` 或 `.otf` 格式字体¹。相比于前文介绍的字体宏包, 我们有了更多自由修改字体的余地。

`xelatex` 和 `lualatex` 命令下支持用户调用字体的宏包是 `fontspec`。宏包提供了几个设置全局字体的命令, 设置 `\rmfamily` 等对应命令的默认字体²:

```
\setmainfont{\font name}[\font features]
\setsansfont{\font name}[\font features]
\setmonofont{\font name}[\font features]
```

¹Linux 下的 T_EX Live 为了令 X_ƎT_EX 使用 OpenType 字体, 需要额外的配置。详见附录 A。

²旧版本 `fontspec` 的命令把必选参数 $\langle font\ name \rangle$ 放在可选参数 $\langle font\ features \rangle$ 的后面。新版本目前兼容旧版本的使用法, 但推荐使用新版本的使用法。

表 5.4: 常见的 L^AT_EX 字体宏包

文本/数学字体搭配的宏包	
lmodern	Latin Modern 字体, 对 Computer Modern 字体的扩展
cmbright	仿 Computer Modern 风格的无衬线字体
euler	Euler 风格数学字体, 也出自于高德纳之手
ccfonts	Concrete 风格字体
txfonts	Times 风格的字体宏包
pxfonts	Palatino 风格的字体宏包
stix	Times 风格的字体宏包
newtxtext,newtxmath	txfonts 的改进版本, 分别设置文本和数学字体
newpxtext,newpxmath	pxfonts 的改进版本, 分别设置文本和数学字体
mathptmx	psnfss 字体宏集之一, Times 风格, 较为陈旧, 不推荐使用
mathpazo	psnfss 字体宏集之一, Palatino 风格, 较为陈旧, 不推荐使用
fourier	Fourier 风格数学字体, 配合 Utopia 正文字体
fouriernc	Fourier 风格数学字体, 配合 New Century Schoolbook 正文字体
arev	Arev 无衬线字体宏包, Vera Sans 风格
mathdesign	配合 Charter/Garamond/Utopia 正文字体的数学字体宏包
文本字体宏包	
以下字体包括传统的 L ^A T _E X 字体格式以及 TrueType/OpenType 格式。	
cm-unicode	Computer Modern 风格的 Unicode 字体, 支持多种西方语言
dejavu	DejaVu 开源字体
droid	Droid 开源字体
inconsolata	Inconsolata 开源等宽字体
libertine	Linux Libertine 和 Linux Biolium 开源字体
roboto	Roboto 开源无衬线字体
sourcesanspro	Source Sans Pro 开源无衬线字体
sourcecodepro	Source Code Pro 开源等宽字体
符号宏包	
mathabx	数学符号宏包之一
MnSymbol	数学符号宏包之一, 配合 Minion Pro 文本字体
fdsymbol	数学符号宏包之一
pifont	Zapf Dingbats 符号宏包

其中 $\langle font\ name \rangle$ 使用字体的文件名（带扩展名）或者字体的英文名称。 $\langle font\ features \rangle$ 用来手动配置对应的粗体或斜体，比如为 Windows 下的无衬线字体 Arial 配置粗体和斜体（通常情况下自动检测并设置对应的粗体和斜体，无需手动指定）：

```
\setsansfont{Arial}[BoldFont={Arial Bold}, ItalicFont={Arial Italic}]
```

$\langle font\ features \rangle$ 还能配置字体本身的各种特性，这里不再赘述，感兴趣的读者请参考 fontspec 宏包的帮助文档。

需要注意的是，fontspec 宏包会覆盖数学字体设置。需要调用表 5.4 中列出的一些数学字体宏包时，应当在调用 fontspec 宏包时指定 no-math 选项。fontspec 宏包可能被其它宏包或文档类（如 ctex 文档类）自动调用时，则在文档开头的 \documentclass 命令里指定 no-math 选项。

5.1.6 在 ctex 宏包或文档类中更改中文字体

前文已经介绍过的 ctex 宏包或文档类提供了和 fontspec 宏包非常类似的语法设置中文字体³：

```
\setCJKmainfont{\font name}[\font features]
\setCJKsansfont{\font name}[\font features]
\setCJKmonofont{\font name}[\font features]
```

由于中文字体少有对应的粗体或斜体， $\langle font\ features \rangle$ 里多用其他字体来配置，比如在 Windows 中设定基本字体为宋体，并设定对应的 BoldFont 为黑体，ItalicFont 为楷体：

```
\setCJKmainfont{SimSun}[BoldFont=SimHei, ItalicFont=KaiTi]
```

5.1.7 使用 unicode-math 宏包配置 Unicode 数学字体

Unicode 数学字体是一类 OpenType 字体，包含了 Unicode 字符集中的数学符号部分，字体中也设定了数学公式排版所需的一些参数。在 xelatex 或者 lualatex 编译命令下，借助 unicode-math 宏包可以调用 Unicode 数学字体配置数学公式的字体风格。

在导言区使用 \usepackage{unicode-math} 后，使用 \setmathfont 命令即可：

```
\setmathfont{\font name}[\font features]
```

绝大多数时候，只需要给定字体名称 $\langle font\ name \rangle$ 即可。由于篇幅所限，在此不介绍可选参数 $\langle font\ feature \rangle$ 涉及的配置，有兴趣的读者请参考宏包的帮助文档。Unicode 数学字体相比于正文字体的选择余地不多。表 5.5 给出了较为常用的 Unicode 数学字体。

unicode-math 宏包与传统数学字体、符号包并不兼容，但其本身已经提供了大量的符号和字体样式。实际上，4.7.1 和 4.7.2 小节中介绍的内容均已被 unicode-math 所涵盖，无需调用其他宏包就可以获得覆盖完整、风格较为统一的字体样式。具体使用方法请参考宏包帮助文档。

5.2 文字装饰和强调

强调文字的方法，或者是添加下划线等装饰物，或者是改变文字的字体。

L^AT_EX 定义了 \underline 命令用来为文字添加下划线：

An `\underline{underlined}` text.

An underlined text.

³使用 xelatex 编译时，这几个命令实际上由 xeCJK 宏包提供；使用 lualatex 编译时，则是由 ctex 宏包或文档类对 luatexja 宏包提供的类似命令进行额外封装。

表 5.5: 常用 Unicode 数学字体。

数学字体名称	配套正文字体名称	备注
开源字体，发布于 CTAN		
Latin Modern Math	Latin Modern	基于 Computer Modern 风格
STIX Math	STIX	Times 风格
XITS Math	XITS	基于 STIX, Times 风格，有粗体 XITS Math Bold 可用
TeX Gyre Pagella Math	TeX Gyre Pagella	Palatino 风格
TeX Gyre Termes Math	TeX Gyre Termes	Times 风格
TeX Gyre DejaVu Math	DejaVu Serif	DejaVu 风格
Libertinus Math	Libertinus	Linux Libertine 风格
Garamond Math	EB Garamond	Garamond 风格
Fira Math	Fira Sans	无衬线数学字体
商业字体		
Cambria Math	Cambria	微软 Office 预装的数学字体
Lucida Bright Math OT	Lucida Bright OT	须购买商业授权
Minion Math	Minion Pro	须购买商业授权

`\underline` 命令生成下划线的样式不够灵活，不同的单词可能生成高低各异的下划线，并且无法换行。`ulem` 宏包提供了更灵活的解决方案，它提供的 `\uline` 命令能够轻松生成自动换行的下划线：

An example of `\uline{some long and underlined words.}`

An example of some long and underlined words.

前一节介绍了 `\emph` 命令，它将文字变为斜体以示强调，而如果在已强调的文字中嵌套使用 `\emph` 命令，命令内则使用直立体文字：

Some `\emph{emphasized words, including \emph{double-emphasized} words}`, are shown here.

Some *emphasized words, including double-emphasized words*, are shown here.

5.3 段落格式和间距

5.3.1 长度和长度变量

在前面的一些章节，我们已经见到一些长度和长度变量的用法。本节首先统一介绍长度和长度变量。

长度的数值 $\langle length \rangle$ 由数字和单位组成。常用的单位见表 5.6。

在一些情况下还会用到可伸缩的“弹性长度”，如 `12pt plus 2pt minus 3pt` 表示基础长度为 `12pt`，可以伸展到 `14pt`，也可以收缩到 `9pt`。也可只定义 `plus` 或者 `minus` 的部分，如 `0pt plus 5pt`。

长度的数值还可以用长度变量本身或其倍数来表达，如 `2.5\parindent` 等。

\LaTeX 预定义了大量的长度变量用于控制版面格式。如页面宽度和高度、首行缩进、段落间距等。如果需要自定义长度变量，需使用如下命令：

```
\newlength{\langle length command \rangle}
```

表 5.6: T_EX/L^AT_EX 中的长度单位

pt	点 (point, 也译作“磅”), = 1/72.27 in
bp	大点 (big point), = 1/72 in
in	英寸, = 2.54 cm
cm	厘米
mm	毫米
em	大致相当于当前字号下大写字母 M 的宽度, 常用于设定水平距离
ex	大致相当于当前字号下小写字母 x 的高度, 常用于设定垂直距离
mu	数学单位 (math unit), = 1/18 em

长度变量可以用 `\setlength` 赋值, 或用 `\addtolength` 增加长度:

```
\setlength{\<length command>}{\<length>}
\addtolength{\<length command>}{\<length>}
```

5.3.2 行距

前文中我们提到过 `\fontsize` 命令可以为字号设定对应的行距, 但我们很少那么用。更常用的办法是在导言区使用 `\linespread` 命令。

```
\linespread{\<factor>}
```

其中 `\<factor>` 作用于基础行距而不是字号。缺省的基础行距是 1.2 倍字号大小 (参考 `\font-size` 命令), 因此使用 `\linespread{1.5}` 意味着最终行距为 1.8 倍的字号大小。

如果不是在导言区全局修改, 而想要局部地改变某个段落的行距, 需要用 `\selectfont` 命令使 `\linespread` 命令的改动立即生效:

```
{\linespread{2.0}\selectfont
The baseline skip is set to be
twice the normal baseline skip.
Pay attention to the \verb|\par|
command at the end. \par}
```

```
In comparison, after the
curly brace has been closed,
everything is back to normal.
```

```
The baseline skip is set to be twice the nor-
mal baseline skip. Pay attention to the
\par command at the end.
```

```
In comparison, after the curly brace has
been closed, everything is back to normal.
```

字号的改变是即时生效的, 而行距的改变直到文字分段时才生效。如果需要改变某一部分文字的行距, 那么不能简单地将文字包含在花括号内。注意下面两个例子中 `\par` 命令的位置, 包括上一个例子的写法 (`\par` 相当于分段, 见 2.3.1 小节):

```
{\Large Don't read this!
It is not true.
You can believe me!\par}
```

```
Don't read this! It is not true.
You can believe me!
```

```
{\Large This is not true either.
But remember I am a liar.}\par}
```

```
This is not true either. But
remember I am a liar.
```

5.3.3 段落格式

以下长度分别为段落的左缩进、右缩进和首行缩进：

```
\setlength{\leftskip}{\langle length \rangle}
\setlength{\rightskip}{\langle length \rangle}
\setlength{\parindent}{\langle length \rangle}
```

它们和设置行距的命令一样，在分段时生效。

控制段落缩进的命令为：

```
\indent
\noindent
```

L^AT_EX 默认在段落开始时缩进，长度为用上述命令设置的 `\parindent`。如果需要在某一段不缩进，可在段落开头使用 `\noindent` 命令。相反地，`\indent` 命令强制开启一段首行缩进的段落。在段落开头使用多个 `\indent` 命令可以累加缩进量。

L^AT_EX 还默认在 `\chapter`、`\section` 等章节标题命令之后的第一段不缩进⁴。如果不习惯这种设定，可以调用 `indentfirst` 宏包，令第一段的首行缩进照常。

段落间的垂直间距为 `\parskip`，如设置段落间距在 `0.8ex` 到 `1.5ex` 变动：

```
\setlength{\parskip}{1ex plus 0.5ex minus 0.2ex}
```

5.3.4 水平间距

L^AT_EX 默认将单词之间的“空格”转化为水平间距。如果需要在文中手动插入额外的水平间距，可使用 `\hspace` 命令：

```
This\hspace{1.5cm}is a space
of 1.5 cm.
```

```
This      is a space of 1.5 cm.
```

`\hspace` 命令生成的水平间距如果位于一行的开头或末尾，则有可能因为断行而被舍弃。可使用 `\hspace*` 命令代替 `\hspace` 命令得到不会因断行而消失的水平间距。

命令 `\stretch{\langle n \rangle}` 生成一个特殊弹性长度，参数 $\langle n \rangle$ 为权重。它的基础长度为 `0pt`，但可以无限延伸，直到占满可用的空间。如果同一行内出现多个 `\stretch{\langle n \rangle}`，这一行的所有可用空间将按每个 `\stretch` 命令给定的权重 $\langle n \rangle$ 进行分配。

命令 `\fill` 相当于 `\stretch{1}`⁵。

```
x\hspace{\stretch{1}}
x\hspace{\stretch{3}}
x\hspace{\fill}x
```

```
x      x      x      x
```

在正文中用 `\hspace` 命令生成水平间距时，往往使用 `em` 作为单位，生成的间距随字号大小而变。我们在数学公式中见过 `\quad` 和 `\qquad` 命令，它们也可以用于文本中，分别相当于 `\hspace{1em}` 和 `\hspace{2em}`：

⁴ctex 宏包和文档类默认按照中文习惯保持标题后第一段的首行缩进。

⁵注意不要用 `1.5\fill` 这样的用法，它生成的长度只有基础长度 `0pt` 而没有延伸部分。


```
{\Large big\hspace{1em}y}\\
{\Large big\quad y}\\
nor\hspace{2em}mal\\
nor\quad mal\\
{\tiny tin\hspace{1em}y}\\
{\tiny tin\quad y}
```

```
big y
big y
nor mal
nor mal
tin y
tin y
```

5.3.5 垂直间距

在页面中，段落、章节标题、行间公式、列表、浮动体等元素之间的间距是 L^AT_EX 预设的。比如 `\parskip`，默认设置为 0pt plus 1pt。

如果我们想要人为地增加段落之间的垂直间距，可以在两个段落之间的位置使用 `\vspace` 命令：

```
A paragraph.

\vspace{2ex}

Another paragraph.
```

```
A paragraph.

Another paragraph.
```

`\vspace` 命令生成的垂直间距在一页的顶端或底端可能被“吞掉”，类似 `\hspace` 在一行的开头和末尾那样。对应地，`\vspace*` 命令产生不会因断页而消失的垂直间距。`\vspace` 也可用 `\stretch` 设置无限延伸的垂直长度。

在段落内的两行之间增加垂直间距，一般通过给断行命令 `\\` 加可选参数，如 `\\[6pt]` 或 `*[6pt]`。`\vspace` 也可以在段落内使用，区别在于 `\vspace` 只引入垂直间距而不断行：

```
Use command \verb|\vspace{12pt}|
to add \vspace{12pt} some spaces
between lines in a paragraph.

Or you can use \verb|\\[12pt]|
to \\[12pt] add vertical space,
but it also breaks the line.
```

```
Use command \vspace{12pt} to add some
spaces between lines in a paragraph.

Or you can use \\[12pt] to
add vertical space, but it also breaks the
line.
```

另外 L^AT_EX 还提供了 `\bigskip`、`\medskip`、`\smallskip` 来增加预定义长度的垂直间距。

```
\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
```

5.4 页面和分栏

我们不妨回顾一下第一章介绍的文档类属性。L^AT_EX 允许用户通过为文档类指定选项来控制纸张的大小（见 1.4.1 小节），包括 `a4paper`、`letterpaper` 等等，并配合字号设置了适合的页边距。

控制页边距的参数由图 5.1 里给出的各种长度变量控制。可以用 `\setlength` 命令修改这些长度变量，以达到调节页面尺寸和边距的作用；反之也可以利用这些长度变量来决定排版内容的尺寸，如在 `tabularx` 环境或 `\includegraphics` 命令的参数里，设置图片或表格的宽度为 `0.8\textwidth`。



图 5.1: 本文档的页面参数示意图 (奇数页; 由 layout 宏包生成)。

页边距等比较直观的参数则必须间接设置。我们根据图 5.1 将各个方向的页边距计算公式给出（以奇数页为例）：

$$\begin{aligned}\langle left-margin \rangle &= 1\text{in} + \backslash hoffset + \backslash oddsidemargin \\ \langle right-margin \rangle &= \backslash paperwidth - \langle left-margin \rangle - \backslash textwidth \\ \langle top-margin \rangle &= 1\text{in} + \backslash voffset + \backslash topmargin + \backslash headheight + \backslash headsep \\ \langle bottom-margin \rangle &= \backslash paperheight - \langle top-margin \rangle - \backslash textheight\end{aligned}$$

如果需要设置合适的 $\langle left-margin \rangle$ 和 $\langle right-margin \rangle$ ，就要通过上述方程组把 $\backslash oddsidemargin$ 和 $\backslash textwidth$ 等参数解出来！

幸好 `geometry` 宏包提供了设置页边距等参数的简便方法，能够帮我们完成背后繁杂的计算。

5.4.1 利用 `geometry` 宏包设置页面参数

`geometry` 宏包的调用方式类似于 `graphicx`，在 `latex + dvipdfmx` 命令下需要指定选项 `dvipdfm` 或 `dvipdfmx`⁶；`pdflatex` 和 `xelatex` 编译命令下不需要。

既可以调用 `geometry` 宏包，然后用其提供的 `\geometry` 命令设置页面参数：

```
\usepackage{geometry}
\geometry{<geometry-settings>}
```

也可以直接在宏包选项中设置：

```
\usepackage[<geometry-settings>]{geometry}
```

其中 $\langle geometry-settings \rangle$ 多以 $\langle key \rangle = \langle value \rangle$ 的形式组织。

比如，符合 Microsoft Word 习惯的页面设定是 A4 纸张，上下边距 1 英寸，左右边距 1.25 英寸，于是我们可以通过如下两种等效的方式之一设定页边距：

```
\geometry{a4paper,left=1.25in,right=1.25in,top=1in,bottom=1in}
% or like this:
\geometry{a4paper,hmargin=1.25in,vmargin=1in}
```

又比如，需要设定周围的边距一致为 1.25 英寸，可以用更简单的语法：

```
\geometry{margin=1.25in}
```

对于书籍等双面文档，习惯上奇数页右边、偶数页左边留出较大的页边距，而靠近书脊一侧的奇数页左边、偶数页右边页边距较小。我们可以这样设定：

```
\geometry{inner=1in,outer=1.25in}
```

需要指出的是，通过 `geometry` 宏包设置的纸张大小是输出 PDF 文件的真实大小，而在文档类选项中设置的参数（见 1.4.1 小节）实际上只影响输出区域。`geometry` 宏包还能够修改页眉页脚高度、边注宽度等参数，并且能比较好地处理各参数之间的依赖关系。更详细的用法不再赘述，感兴趣的用户可查阅其帮助文档。

⁶早期版本的 `geometry` 宏包仅支持 `dvipdfm` 选项。

5.4.2 页面内容的垂直对齐

L^AT_EX 默认将页面内容在垂直方向分散对齐。对于有大量图表的文档，许多时候想要做到排版匀称的页面很困难，垂直分散对齐会造成某些页面的垂直间距过宽，还可能报大量的 Underfull \vbox 警告。L^AT_EX 还提供了另一种策略：将页面内容向顶部对齐，给底部留出高度不一的空白。

以下命令分别令页面在垂直方向向顶部对齐/分散对齐：

```
\raggedbottom
\flushbottom
```

5.4.3 分栏

L^AT_EX 支持简单的单栏或双栏排版。标准文档类的全局选项 onecolumn、twocolumn 可控制全文分单栏或双栏排版。L^AT_EX 也提供了切换单/双栏排版的命令：

```
\onecolumn
\twocolumn[\one-column top material]
```

\twocolumn 支持带一个可选参数，用于排版双栏之上的一部分单栏内容。

切换单/双栏排版时总是会另起一页 (\clearpage)。在双栏模式下使用 \newpage 会换栏而不是换页；\clearpage 则能够换页。

双栏排版时每一栏的宽度为 \columnwidth，它由 \textwidth 减去 \columnsep 的差除以 2 得到。两栏之间还有一道竖线，宽度为 \columnseprule，默认为零，也就是看不到竖线。

一个比较好用的分栏解决方案是 multicol，它提供了简单的 multicol 环境（注意不要写成 multicol 环境）自动产生分栏，如以下环境将内容分为 3 栏：

```
\begin{multicols}{3}
...
\end{multicols}
```

multicol 宏包能够在一页之中切换单栏/多栏，也能处理跨页的分栏，且各栏的高度分布平衡。但代价是在 multicol 环境中无法正常使用 table 和 figure 等浮动体环境，它会直接让浮动体丢失。multicol 环境中只能用跨栏的 table* 和 figure* 环境，或者用 float 宏包提供的 H 参数固定浮动体的位置。

5.5 页眉页脚

5.5.1 基本的页眉页脚样式

L^AT_EX 中提供了命令 \pagestyle 来修改页眉页脚的样式：

```
\pagestyle{<page-style>}
```

命令 \thispagestyle 只影响当页的页眉页脚样式：

```
\thispagestyle{<page-style>}
```

<page-style> 参数为样式的名称，在 L^AT_EX 里预定义了四类样式，见表 5.7。

其中 headings 的情况较为复杂：

表 5.7: L^AT_EX 预定义的页眉页脚样式

empty	页眉页脚为空
plain	页眉为空，页脚为页码。(article 和 report 文档类默认；book 文档类的每章第一页也为 plain 格式)
headings	页眉为章节标题和页码，页脚为空。(book 文档类默认)
myheadings	页眉为页码及 \markboth 和 \markright 命令手动指定的内容，页脚为空。

article 文档类，**twoside** 选项 偶数页为页码和节标题，奇数页为小节标题和页码；

article 文档类，**oneside** 选项 页眉为节标题和页码；

report 和 **book** 文档类，**twoside** 选项 偶数页为页码和章标题，奇数页为节标题和页码；

report 和 **book** 文档类，**oneside** 选项 页眉为章标题和页码。

\pagenumbering 命令令我们能够改变页眉页脚中的页码样式：

```
\pagenumbering{<style>}
```

<style> 为页码样式，默认为 arabic (阿拉伯数字)，还可修改为 roman (小写罗马数字)、Roman (大写罗马数字) 等。注意使用 \pagenumbering 命令后会将页码重置为 1。book 文档类的 \frontmatter 和 \mainmatter 内部就使用了 \pagenumbering 命令切换页码样式。对页码格式的详细说明见 8.3 节。

5.5.2 手动更改页眉页脚的内容

对于 headings 或者 myheadings 样式，L^AT_EX 允许用户使用命令手动修改页眉上面的内容，特别是因为使用了 \chapter* 等命令而无法自动生成页眉页脚的情况：

```
\markright{<right-mark>}
\markboth{<left-mark>}{<right-mark>}
```

在双面排版、headings 或 myheadings 页眉页脚样式下，<left-mark> 和 <right-mark> 的内容分别预期出现在左页（偶数页）和右页（奇数页）。事实上 \chapter 和 \section 等章节命令内部也使用 \markboth 或者 \markright 生成页眉。

L^AT_EX 默认将页眉的内容都转为大写字母。如果保持字母的大小写，可以尝试以下代码（\renewcommand 命令的用法详见 8.1.1 节）⁷：

```
\renewcommand\chaptermark[1]{%
  \markboth{Chapter \thechapter\quad #1}{}}
\renewcommand\sectionmark[1]{%
  \markright{\thesection\quad #1}}
```

其中 \thechapter、\thesection 等命令为章节计数器的数值（详见 8.3 节）。注意以上代码适用于 report 和 book 文档类；对于 article 文档类，与两个页眉相关的命令分别为 \sectionmark 和 \subsectionmark。

⁷但是这不能改变页眉的斜体样式 (\slshape)，斜体是定义在 headings 样式里的。如果不喜欢斜体，可在 \markboth 等命令的参数里先使用 \normalfont，再使用想要的字体样式命令，或直接尝试使用 fancyhdr 宏包。

5.5.3 fancyhdr 宏包

fancyhdr 宏包改善了页眉页脚样式的定义方式，允许我们将内容自由安置在页眉和页脚的左、中、右三个位置，还为页眉和页脚各加了一条横线。

fancyhdr 自定义了样式名称 fancy。使用 fancyhdr 宏包定义页眉页脚之前，通常先用 `\pagestyle{fancy}` 调用这个样式。在 fancyhdr 中定义页眉页脚的命令为：

```
\fancyhf[⟨position⟩]{...}
\fancyhead[⟨position⟩]{...}
\fancyfoot[⟨position⟩]{...}
```

其中 `⟨position⟩` 为 L (左) / C (中) / R (右) 以及与 O (奇数页) / E (偶数页) 字母的组合。`\fancyhf` 用于同时定义页眉和页脚，习惯上使用 `\fancyhf{}` 来清空页眉页脚的设置。

源代码 5.1 给出了 fancyhdr 基础用法的一个示例，效果为将章节标题放在和 headings 一致的位置，但使用加粗格式；页码都放在页脚正中；修改横线宽度，“去掉”页脚的横线。

```
% 在导言区使用此代码
\usepackage{fancyhdr}
\pagestyle{fancy}
\renewcommand{\chaptermark}[1]{\markboth{#1}{}}
\renewcommand{\sectionmark}[1]{\markright{\thesection\ #1}}
\fancyhf{}
\fancyfoot[C]{\bfseries\thepage}
\fancyhead[LO]{\bfseries\rightmark}
\fancyhead[RE]{\bfseries\leftmark}
\renewcommand{\headrulewidth}{0.4pt} % 注意不用 \setlength
\renewcommand{\footrulewidth}{0pt}
```

源代码 5.1: fancyhdr 宏包的使用方法示例。

fancyhdr 还支持用 `\fancypagestyle` 为自定义的页眉页脚样式命名，或者重新定义已有的样式如 plain 等：

```
% 自定义 myfancy 样式
\fancypagestyle{myfancy}{%
  \fancyhf{}
  \fancyhead{...}
  \fancyfoot{...}
}
% 使用样式
\pagestyle{myfancy}
```

更多用法请参考 fancyhdr 宏包的帮助文档。

第六章 特色工具和功能

本章介绍一些特色的 \LaTeX 辅助功能。前两个功能 \BibTeX 和 makeindex 依靠一些辅助程序自动生成参考文献、索引等；之后的使用颜色、超链接等则令我们生成美观易用的电子文档。

6.1 参考文献和 \BibTeX 工具

6.1.1 基本的参考文献和引用

\LaTeX 提供的参考文献和引用方式比较原始，需要用户自行书写参考文献列表（包括格式），因此较难直接使用。相关的命令我们只作最简单的介绍。

\LaTeX 提供了最基本的 \cite 命令用于在正文中引用参考文献：

```
\cite{<citation>}
```

$\langle citation \rangle$ 为引用的参考文献的标签，类似 \ref 里的参数； \cite 带一个可选参数，为引用的编号后加上额外的内容，如 $\text{\cite[page 22]{Paper2013}}$ 可能得到形如 [13, page 22] 这样的引用。

参考文献由 thebibliography 环境包裹。每条参考文献由 \bibitem 开头，其后是参考文献本身的内容：

```
\begin{thebibliography}{<widest label>}
  \bibitem[<item number>]{<citation>} ...
\end{thebibliography}
```

其中 $\langle citation \rangle$ 是 \cite 使用的文献标签， $\langle item number \rangle$ 自定义参考文献的序号，如果省略，则按自然排序给定序号。 $\langle widest label \rangle$ 用以限制参考文献序号的宽度，如 99 意味着不超过两位数字。通常设定为与参考文献的数目一致。

在 article 文档类， thebibliography 环境自动生成不带编号的一节，标题默认为 “References”；而在 report 或 book 文档类中，则会生成不带编号的一章，标题默认为 “Bibliography”。用户可通过 8.4 节给出的方法定制参考文献的标题。

以下为一个使用 thebibliography 排版参考文献的例子：

```
\documentclass{article}
\begin{document}
\section{Introduction}
Part I~\cite{germenTeX} has proposed that \ldots

\begin{thebibliography}{99}
```

```
\bibitem{germenTeX} H.~Partl: \emph{German \TeX},
  TUGboat Volume~9, Issue~1 (1988)
\end{thebibliography}
\end{document}
```

6.1.2 BibTeX 数据库

BibTeX 是最为流行的参考文献数据组织格式之一。它的出现让我们摆脱手写参考文献条目的麻烦。我们还可以通过参考文献样式的支持，让同一份 BibTeX 数据库生成不同样式的参考文献列表。

BibTeX 数据库以 .bib 作为扩展名，其内容是若干个文献条目，每个条目的格式为：

```
@<type>{<citation>,
  <key1> = {<value1>},
  <key2> = {<value2>},
  ...
}
```

其中 $\langle type \rangle$ 为文献的类别，如 `article` 为学术论文，`book` 为书籍，`incollection` 为论文集集中的某一篇文章，等等。 $\langle citation \rangle$ 为 `\cite` 命令使用的文献标签。在 $\langle citation \rangle$ 之后为条目里的各个字段，以 $\langle key \rangle = \{ \langle value \rangle \}$ 的形式组织。

我们在此简单列举学术论文里使用较多的 BibTeX 文献条目类别：

article 学术论文，必需字段有 `author`, `title`, `journal`, `year`；可选字段包括 `volume`, `number`, `pages`, `doi` 等；

book 书籍，必需字段有 `author/editor`, `title`, `publisher`, `year`；可选字段包括 `volume/number`, `series`, `address` 等；

incollection 论文集中的一篇，必需字段有 `author`, `title`, `booktitle`, `publisher`, `year`；可选字段包括 `editor`, `volume/number`, `chapter`, `pages`, `address` 等；

inbook 书中的一章，必需字段有 `author/editor`, `title`, `chapter/pages`, `publisher`, `year`；可选字段包括 `volume/number`, `series`, `address` 等。

例如 `article` 类别的参考文献数据条目写法如下：

```
@article{Alice13,
  title = {Demonstration of bibliography items},
  author = {Alice Axford and Bob Birkin and Charlie Copper and Danny Dannford},
  year = {2013},
  month = {Mar},
  journal = {Journal of \TeX perts},
  volume = {36},
  number = {7},
  pages = {114-120}}
```

所有类别的文献条目格式请参考 [CTAN://biblio/bibtex/base/btxdoc.pdf](http://ctan://biblio/bibtex/base/btxdoc.pdf)。

多数时候，我们无需自己手写 BibTeX 文献条目。从 Google Scholar 或者期刊/数据库的网站上都能够导出 BibTeX 文献条目，老牌的文献管理软件 EndNote 也支持生成 BibTeX 格式的数据库。开源软件 JabRef 甚至支持 BibTeX 文献条目的导入、导出和管理。

6.1.3 BibTeX 样式

参考文献的写法在不同文献里千差万别，包括作者、标题、年份等各项的顺序和字体样式、文献在列表中的排序规则等。BibTeX 用样式 (style) 来管理参考文献的写法。BibTeX 提供了几个预定义的样式，如 `plain`, `unsrt`, `alpha` 等。如果使用期刊模板的话，可能会提供自用的样式。样式文件以 `.bst` 为扩展名。

使用样式文件的方法是在源代码内（一般在导言区）使用 `\bibliographystyle` 命令：

```
\bibliographystyle{<bst-name>}
```

这里 `<bst-name>` 为 `.bst` 样式文件的名称，**不要带 `.bst` 扩展名**。

我们以 6.1.2 小节给出的数据条目为例，使用 `\bibliographystyle` 命令选择不同的参考文献样式，效果大致如表 6.1 所示。

表 6.1: BibTeX 样式的排版效果

plain

[1] Alice Axford, Bob Birkin, Charlie Copper, and Danny Dannford. Demonstration of bibliography items. *Journal of T_EXperts*, 36(7):114–120, Mar 2013.

alpha

[ABCD13] Alice Axford, Bob Birkin, Charlie Copper, and Danny Dannford. Demonstration of bibliography items. *Journal of T_EXperts*, 36(7):114–120, Mar 2013.

abbrv

[1] A. Axford, B. Birkin, C. Copper, and D. Dannford. Demonstration of bibliography items. *Journal of T_EXperts*, 36(7):114–120, Mar 2013.

amspain (*AMS* 文档类 `amsart` 等配套的样式)

[1] Alice Axford, Bob Birkin, Charlie Copper, and Danny Dannford, *Demonstration of bibliography items*, *Journal of T_EXperts* **36** (2013), no. 7, 114–120.

elsarticle-num (Elsevier 提供的 `elsarticle` 文档类配套的样式)

[1] A. Axford, B. Birkin, C. Copper, D. Dannford, Demonstration of bibliography items, *Journal of T_EXperts* 36 (7) (2013) 114–120.

IEEEtran (`IEEEtran` 模板文档类配套的样式)

[1] A. Axford, B. Birkin, C. Copper, and D. Dannford, “Demonstration of bibliography items,” *Journal of T_EXperts*, vol. 36, no. 7, pp. 114–120, Mar 2013.

gbt7714-numerical (GB/T 7714—2015 样式，由 `gbt7714` 宏包提供)

[1] 陈登原. 国史旧闻: 第 1 卷 [M]. 北京: 中华书局, 2000: 29.

6.1.4 使用 BibTeX 排版参考文献

现在我们来看如何利用 BibTeX 数据库生成参考文献和引用。

第一步：准备一份 BibTeX 数据库，假设数据库文件名为 `books.bib`，和 `LaTeX` 源代码一般位于同一个目录下。

第二步：在源代码中添加必要的命令。假设源代码名为 `demo.tex`（见源代码 6.1）。

1. 首先需要使用命令 `\bibliographystyle` 设定参考文献的格式。
2. 其次，在正文中引用参考文献。BibTeX 程序在生成参考文献列表的时候，通常只列出了 `\cite` 命令引用的那些。如果需要列出未被引用的文献，则需要 `\nocite{\langle citation \rangle}` 命令；而 `\nocite{*}` 则让所有未被引用的文献都列出。
3. 再次，在需要列出参考文献的位置，使用 `\bibliography` 命令代替 `thebibliography` 环境：

```
\bibliography{\langle bib-name \rangle}
```

其中 `\langle bib-name \rangle` 是 BibTeX 数据库的文件名，不要带 `.bib` 扩展名。

```
\documentclass{article}
\bibliographystyle{plain}
\begin{document}
\section{Some words}
Some excellent books, for example, \cite{citation1}
and \cite{citation2} \ldots

\bibliography{books}
\end{document}
```

源代码 6.1: 利用 `books.bib` 生成参考文献的源代码 `demo.tex`。

注意：`\bibliographystyle` 和 `\bibliography` 命令缺一不可，没有这两个命令，使用 BibTeX 生成参考文献列表的时候会报错。

第三步：写好了以上两个文件之后，我们就可以开始编译了。

1. 首先使用 `pdflatex` 或 `xelatex` 等命令编译 L^AT_EX 源代码 `demo.tex`；
2. 接下来用 `bibtex` 命令处理 `demo.aux` 辅助文件记录的参考文献格式、引用条目等信息。
`bibtex` 命令处理完毕后会生成 `demo.bbl` 文件，内容就是一个 `thebibliography` 环境；
3. 再使用 `pdflatex` 或 `xelatex` 等命令把源代码 `demo.tex` 编译两遍，读入参考文献并正确生成引用。

整个过程使用的命令如下（可以略去扩展名）：

```
xelatex demo
bibtex demo
xelatex demo
xelatex demo
```

使用 `latex + dvipdfmx` 命令编译时，则 `dvipdfmx` 命令放在最后，相当于先后使用 `latex`, `bibtex`, `latex`, `latex`, `dvipdfmx`。

6.1.5 natbib 宏包

时下许多学术期刊比较喜欢使用人名——年份的引用方式，形如 (*Axford et al.*, 2013)。`natbib` 宏包提供了对这种“自然”引用方式的处理。

除了 `\cite` 之外，`natbib` 宏包在正文中支持两种引用方式：

```
\citep{\citation}
\citett{\citation}
```

它们分别生成形如 (*Axford et al.*, 2013) 和 *Axford et al.* (2013) 的人名——年份引用。正确排版人名——年份引用还依赖于特定的 BibTeX 样式。natbib 提供了与 L^AT_EX 预定义样式相对应的几个样式, 包括 plainnat、abbrvnat 和 unsrtnat。学术论文模板是否支持 natbib, 需要参考其帮助文档。

natbib 宏包同样也支持数字引用, 并且支持将引用的序号压缩, 例如:

```
\usepackage[numbers,sort&compress]{natbib}
```

调用 natbib 宏包时指定以上选项后, 连续引用多篇文献时, 会生成形如 (3–7) 的引用而不是 (3, 4, 5, 6, 7)。

natbib 宏包还有更多选项和用法, 比如默认的引用是用小括号包裹的, 可指定 square 选项改为中括号; 再比如 \citep 命令也支持可选参数, 为引用前后都添加额外内容。这里不再赘述, 请参考 natbib 宏包的帮助文档。

6.1.6 biblatex 宏包

本节的末尾简单介绍一下基于 biblatex 宏包排版参考文献的方式。biblatex 宏包是一套基于 L^AT_EX 宏命令的参考文献解决方案, 提供了便捷的格式控制和强大的排序、分类、筛选、多文列表等功能。biblatex 宏包也因其对 UTF-8 和中文参考文献的良好支持, 被国内较多 L^AT_EX 模板采用。

基于 biblatex 宏包的方式与基于 BibTeX 的传统方式有一定区别, 下面从文档结构和命令、编译方式、样式选择等方面逐一介绍:

文档结构和 biblatex 相关命令

1. 首先是在导言区调用 biblatex 宏包。宏包支持以 $\langle key \rangle = \langle value \rangle$ 形式指定选项, 包括参考文献样式 style、参考文献著录排序的规则 sorting 等。
2. 接着在导言区使用 \addbibresource 命令为 biblatex 引入参考文献数据库。与基于 BibTeX 的传统方式不同的是, 这里需要写完整的文件名。
3. 在正文中使用 \cite 命令引用参考文献。除此之外还可以使用丰富的命令达到不同的引用效果, 如 \citeauthor 和 \citeyear 分别单独引用作者和年份, \textcite 和 \parencite 分别类似 natbib 宏包提供的 \citett 和 \citep 命令, 以及脚注式引用 \footcite 等。
4. 最后在需要排版参考文献的位置使用命令 \printbibliography。

编译方式

与基于 BibTeX 的传统方式不同的是, biblatex 宏包使用 biber 程序处理参考文献。因此上述文档的编译步骤为:

```
xelatex demo
biber demo
xelatex demo
xelatex demo
```

```
% File: egbibdata.bib
@book{caimin2006,
  title      = {UML 基础和 Rose 建模教程},
  address    = {北京},
  author     = {蔡敏 and 徐慧慧 and 黄柄强},
  publisher  = {人民邮电出版社},
  year       = {2006},
  month      = {1}
}

% File: demo.tex
\documentclass{ctexart}
% 使用符合 GB/T 7714-2015 规范的参考文献样式
\usepackage[style=gb7714-2015]{biblatex}
% 注意加 .bib 扩展名
\addbibresource{egbibdata.bib}

\begin{document}

见文献\cite{caimin2006}。

\printbibliography
\end{document}
```

源代码 6.2: 应用 biblatex 的示例 egbibdata.bib 和 demo.tex。

biblatex 样式和其它选项

biblatex 使用的参考文献样式分为著录样式 (bibliography style) 和引用样式 (citation style), 分别以 .bbx 和 .cbx 为扩展名。参考文献的样式在调用宏包时使用 style 选项指定, 或者使用 bibstyle 或 citestyle 分别指定:

```
% 同时调用 gb7714-2015.bbx 和 gb7714-2015.cbx
\usepackage[style=gb7714-2015]{biblatex}
% 著录样式调用 gb7714-2015.bbx, 引用样式调用 biblatex 宏包自带的 authoryear
\usepackage[bibstyle=gb7714-2015,citestyle=authoryear]{biblatex}
```

以下总结一些常用的参考文献样式, 除 biblatex 宏包自带的样式外, 许多样式以单独的宏包在发行版内发布。

authoryear biblatex 自带样式, 类似 natbib 默认的引用样式效果。

authortitle biblatex 自带样式, 采用作者-题名 (shorttitle 字段) 的引用样式。

verbose biblatex 自带样式, 引用样式中包含作者、题名、书名、页码等字段的信息。

alphabetic biblatex 自带样式, 著录样式与 BibTeX 的 alpha 样式类似。

trad-alpha biblatex-trad 样式包, 移植自 BibTeX 默认的 alpha 样式。另外还包括 trad-abbrev、trad-plain 和 trad-unsrt。

gb7714-2015 符合中文文献著录标准 GB/T 7714—2015 的样式, 著录按顺序编码排版。另外还包括按作者一年份顺序排版著录的样式 gb7714-2015ay。

caspervector 以中文文献著录标准 GB/T 7714—2015 为基础的一个样式。

ieee 兼容 IEEEtran 风格的样式, 著录按顺序编码排版。另外还包括按作者一年份顺序排版著录的样式 ieee-alphabetic。

6.2 索引和 makeindex 工具

书籍和大文档通常用索引来归纳关键词, 方便用户查阅。L^AT_EX 借助配套的 makeindex 程序完成对索引的排版。

6.2.1 使用 makeindex 工具的方法

要使用索引, 须经过这么几个步骤 (仍设源代码名为 demo.tex):

第一步, 在 L^AT_EX 源代码的导言区调用 makeidx 宏包, 并使用 \makeindex 命令开启索引的收集:

```
\usepackage{makeidx}
\makeindex
```

第二步, 在正文中需要索引的地方使用 \index 命令。 \index 命令的参数写法详见下一小节; 并在需要输出索引的地方 (如所有章节之后) 使用 \printindex 命令。

第三步, 编译过程:

1. 首先用 xelatex 等命令编译源代码 demo.tex。编译过程中产生索引记录文件 demo.idx;
2. 用 makeindex 程序处理 demo.idx, 生成用于排版的索引列表文件 demo.ind;
3. 再次编译源代码 demo.tex, 正确生成索引列表。

6.2.2 索引项的写法

添加索引项的命令为：

```
\index{⟨index entry⟩}
```

其中 $\langle index\ entry \rangle$ 为索引项，写法由表 6.2 汇总。其中 !、@ 和 | 为特殊符号，如果要向索引项直接输出这些符号，需要加前缀 "；而 " 需要输入两个引号 "" 才能输出到索引项。

表 6.2: 索引项的写法列表

| 举例 | 索引项 | 备注 |
|---|----------------------------|--------------------|
| 普通索引 | | |
| hello | hello, 1 | 普通索引 |
| 分级索引，以 ! 分隔，最多支持三级 | | |
| hello | hello, 1 | 一级索引 |
| hello!Peter | Peter, 3 | 二级索引 |
| hello!Peter!Jack | Jack, 3 | 三级索引 |
| 格式化索引，形式为 $\langle alpha \rangle @ \langle format \rangle$ | | |
| $\langle alpha \rangle$ 为纯字母，用来排序 | | |
| $\langle format \rangle$ 为索引的格式，可以包括 L ^A T _E X 代码和简单的公式 | | |
| Möbius@M\ "obius | Möbius, 2 | 输出重音 |
| alpha@\$\alpha\$ | α , 7 | 输出公式 |
| bold@\textbf{bold} | bold , 12 | 输出粗体 |
| 页码范围 | | |
| morning (morning) | morning, 6–7 | 范围索引的开头
范围索引的结尾 |
| 格式化索引页码 | | |
| Jenny textbf | Jenny, 3 | 调用 \textbf 加粗页码 |
| Joe see{Jenny} | Joe, <i>see</i> Jenny | 调用 \see 生成特殊形式 |
| Joe seealso{Jenny} | Joe, <i>see also</i> Jenny | 调用 \seealso 生成特殊形式 |

读者可以钻研一下以下给出的一个较为复杂的，结合多级索引、索引格式、页码格式等的用法示例。但在自己使用时，最好还是遵循“简单的就是最好的”原则，尽量使用表 6.2 中的写法。

```
Test index.
\index{Test@\textsf{"Test}|(textbf}
\index{Test@\textsf{"Test"!sub@"|sub"||see{Test}}
\newpage
Test index.
\index{Test@\textsf{"Test}|)textbf}
```

6.3 使用颜色

原始的 L^AT_EX 不支持使用各种颜色。color 宏包或者 xcolor 宏包提供了对颜色的支持，给 PDF 输出生成颜色的特殊指令。

6.3.1 颜色的表达方式

调用 `color` 或 `xcolor` 宏包后，我们就可以用如下命令切换颜色：

```
\color[color-mode]{code}
\color{color-name}
```

颜色的表达方式有两种，其一是使用色彩模型和色彩代码，代码用 0 ~ 1 的数字代表成分的比例。`color` 宏包支持 `rgb`、`cmlyk` 和 `gray` 模型，`xcolor` 支持更多的模型如 `hsb` 等。

```
\large\sffamily
{\color[gray]{0.6}
  60\% 灰色} \
{\color[rgb]{0,1,1}
  青色}
```

60% 灰色
青色

其二是直接使用名称代表颜色，前提是已经定义了颜色名称（没定义的话会报错）：

```
\large\sffamily
{\color{red} 红色} \
{\color{blue} 蓝色}
```

红色
蓝色

`color` 宏包仅定义了 8 种颜色名称，`xcolor` 补充了一些，总共有 19 种，见表 6.3。

表 6.3: `color` 和 `xcolor` 宏包可用的颜色名称

| 基本的 8 种颜色名称（后三种颜色为 CMYK 模型） | | | |
|--|---|---|--|
| black  | red  | green  | blue  |
| white  | cyan  | magenta  | yellow  |
| xcolor 额外可用的颜色名称： | | | |
| darkgray  | gray  | lightgray  | |
| brown  | olive  | orange  | lime  |
| purple  | teal  | violet  | pink  |

`xcolor` 还支持将颜色通过表达式混合或互补：

```
\large\sffamily
{\color{red!40} 40\% 红色} \
{\color{blue} 蓝色}
\color{blue!50!black} 蓝黑
\color{black} 黑色 \
{\color{-red} 红色的互补色}
```

40% 红色
蓝色 蓝黑 黑色
红色的互补色

我们还可以通过命令自定义颜色名称，注意这里的 `<color-mode>` 是必选参数：

```
\definecolor{<color-name>}{<color-mode>}{<code>}
```

如果调用 `color` 或 `xcolor` 宏包时指定 `dvipsnames` 选项，就有额外的 68 种颜色名称可用¹。`xcolor` 宏包还支持通过指定其它选项载入更多颜色名称。限于篇幅不展开介绍，详情请参考 `xcolor` 宏包的手册。

¹`color` 宏包实际要指定 `dvipsnames` 和 `usenames` 选项。

6.3.2 带颜色的文本和盒子

原始的 `\color` 命令类似于字体命令 `\bfseries`，它使之后排版的内容全部变成指定的颜色，所以直接使用时通常要加花括号分组。`color` 和 `xcolor` 宏包都定义了一些方便用户使用的带颜色元素。

输入带颜色的文本可以用类似 `\textbf` 的命令：

```
\textcolor[⟨color-mode⟩]{⟨code⟩}{⟨text⟩}
\textcolor{⟨color-name⟩}{⟨text⟩}
```

以下命令构造一个带背景色的盒子，`⟨material⟩` 为盒子中的内容：

```
\colorbox[⟨color-mode⟩]{⟨code⟩}{⟨material⟩}
\colorbox{⟨color-name⟩}{⟨material⟩}
```

以下命令构造一个带背景色和有色边框的盒子，`⟨fcode⟩` 或 `⟨fcolor-name⟩` 用于设置边框颜色：

```
\fcolorbox[⟨color-mode⟩]{⟨fcode⟩}{⟨code⟩}{⟨material⟩}
\fcolorbox{⟨fcolor-name⟩}{⟨color-name⟩}{⟨material⟩}
```

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

文字用红色强调
浅灰色背景
蓝色边框 + 文字，黄色背景

`\fcolorbox` 也可以像 3.8.2 小节里的 `\fbox` 那样调节 `\fboxrule` 和 `\fboxsep`；对于 `\colorbox`，调整 `\fboxsep` 是有效的。这里不再给出具体的示例。

6.4 使用超链接

PDF 文档格式是现今最流行的电子文档格式，而电子文档最实用的需求之一就是超链接功能。 \LaTeX 中实现这一功能的是 `hyperref` 宏包。

6.4.1 hyperref 宏包

`hyperref` 宏包涉及的链接遍布 \LaTeX 的每一个角落——目录、引用、脚注、索引、参考文献等等都被封装成超链接。但这也使得它与其它宏包发生冲突的可能性大大增加，虽然宏包已经尽力解决各方面的兼容性，但仍不能面面俱到。为减少可能的冲突，习惯上将 `hyperref` 宏包放在其它宏包之后调用。

与 `graphicx` 宏包类似，`latex + dvipdfmx` 命令下调用 `hyperref` 宏包时，需要指定选项 `dvipdfmx`；而 `pdflatex` 或 `xelatex` 命令下不需要。

`hyperref` 宏包提供了命令 `\hypersetup` 配置各种参数。参数也可以作为宏包选项，在调用宏包时指定：

```
\hypersetup{⟨option1⟩,⟨option2⟩=⟨value⟩,...}
\usepackage[⟨option1⟩,⟨option2⟩=⟨value⟩,...]{hyperref}
```

当选项值为 `true` 时，可以省略 “`=true`” 不写。可用的参数见表 6.4。

表 6.4: hyperref 宏包提供的参数设置

| 参数 | 默认值 | 含义 |
|--|--------------------|---|
| <code>draft=\langle true false \rangle</code> | <code>false</code> | 关闭所有超链接、书签等功能（也可以通过文档类选项指定） |
| <code>final=\langle true false \rangle</code> | <code>true</code> | 开启所有超链接、书签等功能（也可以通过文档类选项指定） |
| <code>colorlinks=\langle true false \rangle</code> | <code>false</code> | 设置为 <code>true</code> 为链接文字带颜色，反之加上带颜色的边框 |
| <code>hidelinks</code> | | 取消链接的颜色和边框 |
| <code>pdfborder=\{\langle n \rangle \langle n \rangle \langle n \rangle\}</code> | <code>0 0 1</code> | 超链接边框设置，设为 <code>0 0 0</code> 可取消边框 |
| <code>bookmarks=\langle true false \rangle^\dagger</code> | <code>true</code> | 是否生成书签 |
| <code>bookmarksopen=\langle true false \rangle</code> | <code>false</code> | 是否展开书签 |
| <code>bookmarksnumbered=\langle true false \rangle</code> | <code>false</code> | 书签是否带章节编号 |
| <code>pdftitle=\langle string \rangle</code> | 空 | 标题 |
| <code>pdfauthor=\langle string \rangle</code> | 空 | 作者 |
| <code>pdfsubject=\langle string \rangle</code> | 空 | 主题 |
| <code>pdfkeywords=\langle string \rangle</code> | 空 | 关键词 |
| <code>pdfstartview=\langle Fit FitH FitV \rangle</code> | <code>Fit</code> | 设置 PDF 页面以适合页面/适合宽度/适合高度等方式显示，默认为适合页面 |

[†] 该选项只能作为宏包选项，在调用宏包时指定。

6.4.2 超链接

hyperref 宏包提供了直接书写超链接的命令，用于在 PDF 中生成 URL：

```
\url{\langle url \rangle}
\nolinkurl{\langle url \rangle}
```

`\url` 和 `\nolinkurl` 都像抄录命令 `\verb` 一样输出一个 URL，区别是前者还为 URL 加上了超链接，后者没有。²在 `\url` 等命令的参数 `\langle url \rangle` 里，可直接输入如 `%`、`&` 这样的特殊符号。我们也可以像 HTML 中的超链接一样，把一段文字作为超链接：

```
\href{\langle url \rangle}{\langle text \rangle}
```

```
\url{https://wikipedia.org} \\\nolinkurl{https://wikipedia.org} \\\href{https://wikipedia.org}{Wiki}
```

```
https://wikipedia.org
https://wikipedia.org
Wiki
```

使用 hyperref 宏包后，文档中所有的引用、参考文献、索引等等都转换为超链接。用户也可对某个 `\label` 命令定义的标签 `\langle label \rangle` 作超链接（注意这里的 `\langle label \rangle` 虽然是可选参数的形式，但通常是必填的³）：

```
\hyperref[\langle label \rangle]{\langle text \rangle}
```

²注意，一些 PDF 阅读器会为 URL 文本自动加上超链接，这些超链接不是 L^AT_EX 生成的。
³不带可选参数的 `\hyperref` 命令带四个必选参数，前三个组成特殊格式的超链接，最后一个同 `\langle text \rangle`。

默认的超链接在文字外边加上一个带颜色的边框（在打印 PDF 时边框不会打印），可指定 `colorlinks` 参数修改为将文字本身加上颜色，或修改 `pdfborder` 参数调整边框宽度以“去掉”边框；`hidelinks` 参数则令超链接既不变色也不加边框。

```
\hypersetup{hidelinks}
% or:
\hypersetup{pdfborder={0 0 0}}
```

6.4.3 PDF 书签

`hyperref` 宏包另一个强大的功能是为 PDF 生成书签。对于章节命令 `\chapter`、`\section` 等，默认情况下会为 PDF 自动生成书签。和交叉引用、索引等类似，生成书签也需要多次编译源代码，第一次编译将书签记录写入 `.out` 文件，第二次编译才正确生成书签。

书签的一些属性见表 6.4。使用 CJK 宏包时，为了防止中文书签出现乱码，需要进行繁琐的设置；但在使用 `ctex` 宏包和文档类、且使用 `xelatex` 或 `lualatex` 编译的情况下，无需用户额外干预，即可正确生成中文书签。

`hyperref` 还提供了手动生成书签的命令：

```
\pdfbookmark[<level>]{<bookmark>}{<anchor>}
```

<bookmark> 为书签名称，*<anchor>* 为书签项使用的锚点（类似交叉引用的标签）。可选参数 *<level>* 为书签的层级，默认为 0。

章节命令里往往有 \LaTeX 命令甚至数学公式，而 PDF 书签是纯文本，对命令和公式的处理很困难，有出错的风险。`hyperref` 宏包已经为我们处理了许多常见命令，如 `\LaTeX` 和字体命令 `\textbf` 等，对于未被处理的命令或数学公式，就要在章节标题中使用如下命令，分别提供 \LaTeX 代码和 PDF 书签可用的纯文本：

```
\texorpdfstring{<LaTeX code>}{<PDF bookmark text>}
```

比如在章节名称里使用公式 $E = mc^2$ ，而书签则使用纯文本形式的 `E=mc^2`：

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

6.4.4 PDF 文档属性

`hyperref` 宏包还提供了一些参数用于改变 PDF 文档的属性，部分见表 6.4。

第七章 绘图功能

除了排版文字， \LaTeX 也支持用代码表示图形。不同的扩展已经极大地丰富了 \LaTeX 的图形功能，`TikZ` 就是其中之一。本章将带你了解一些基本的绘图功能。

一些特殊的绘图，如交换图、树状图甚至分子式和电路图也能够通过代码绘制，不过其复杂程度已经超出本手册范围，有兴趣的读者可以查阅一些帮助文档，或者在互联网寻求帮助。

7.1 绘图语言简介

\LaTeX 提供了原始的 `picture` 环境，能够绘制一些基本的图形如点、线、矩形、圆、Bézier 曲线等等，不过受制于 \LaTeX 本身，它的绘图功能极为有限，效果也不够美观。

当前较为流行的、用于 \LaTeX 的绘图宏包/程序主要有：

- PSTricks

以 PostScript 语法为基础的绘图宏包，具有优秀的绘图能力。它对老式的 `latex + dvips` 编译命令支持最好，而现在的几种编译命令下使用起来都不够方便。

- TikZ & pgf

德国的 Till Tantau 教授在开发著名的 \LaTeX 幻灯片文档类 `beamer` 时一并开发了绘图宏包 `pgf`，目的是令其能够在 `pdflatex` 或 `xelatex` 等不同的编译命令下都能使用。`TikZ` 是在 `pgf` 基础上封装的一个宏包，采用了类似 `METAPOST` 的语法，提供了方便的绘图命令，绘图能力不输 `PSTricks`。

- METAPOST & Asymptote

`METAPOST` 脱胎于高德纳为 \TeX 配套开发的字体生成程序 `METAFONT`，具有优秀的绘图能力，并能够调用 \TeX 引擎向图片中插入文字和公式。`Asymptote` 在 `METAPOST` 的基础上更进一步，具有一定的类似 C 语言的编程能力，支持三维图形的绘制。

它们作为独立的程序，通常的用法是将代码写在单独的文件里，编译生成图片供 \LaTeX 引用，也可以借助特殊的宏包在 \LaTeX 代码里直接使用。

本手册将介绍 `TikZ` 绘图宏包里最基本的部分。`TikZ` 还支持各种自定义的扩展，基于 `TikZ` 的专门用途的绘图宏包也不胜枚举，其复杂程度已远远超出入门手册的范围（`TikZ` 的帮助文档有上千页之厚）。对此感兴趣的读者需要自行查阅帮助文档，或者到互联网上参考现成的范例。

7.2 TikZ 绘图语言

在导言区调用 `tikz` 宏包，就可以用以下命令和环境使用 `TikZ` 的绘图功能了¹：

¹`latex + dvipdfmx` 编译方式要在 `tikz` 宏包之前调用 `graphicx` 宏包并指定 `dvipdfmx` 选项。

```

\tikz[...] <tikz code>;

\tikz[...] {\<tikz code 1>; <tikz code 2>; ...}

\begin{tikzpicture}[...]
<tikz code 1>;
<tikz code 2>;
...
\end{tikzpicture}

```

前一种用法为 `\tikz` 带单条绘图命令，以分号结束，一般用于在文字之间插入简单的图形；后两种用法较为常见，使用多条绘图命令，可以在 `figure` 等浮动体中使用。

7.2.1 TikZ 坐标和路径

TikZ 用直角坐标系或者极坐标系描述点的位置。

- 直角坐标下，点的位置写作 $(\langle x \rangle, \langle y \rangle)$ ，坐标 $\langle x \rangle$ 和 $\langle y \rangle$ 可以用 L^AT_EX 支持的任意单位表示，缺省为 `cm`；
- 极坐标下，点的位置写作 $(\langle \theta \rangle : \langle r \rangle)$ 。 θ 为极角，单位是度。

我们还可以为某个点命名：`\coordinate (A) at (<coordinate>)` 然后就可以使用 `(A)` 作为点的位置了。

```

\begin{tikzpicture}
\draw (0,0) -- (30:1);
\draw (1,0) -- (2,1);
\coordinate (S) at (0,1);
\draw (S) -- (1,1);
\end{tikzpicture}

```



坐标的表示形式还包括“垂足”形式：

```

\begin{tikzpicture}
\coordinate (S) at (2,2);
\draw[gray] (-1,2) -- (S);
\draw[gray] (2,-1) -- (S);
\draw[red] (0,0) -- (0,0 |- S);
\draw[blue] (0,0) -- (0,0 | - S);
\end{tikzpicture}

```



TikZ 最基本的路径为两点之间连线，如 $(\langle x_1 \rangle, \langle y_1 \rangle) -- (\langle x_2 \rangle, \langle y_2 \rangle)$ ，可以连用表示多个连线（折线）。连续使用连线时，可以使用 `cycle` 令路径回到起点，生成闭合的路径。

```

\begin{tikzpicture}
\draw (0,0) -- (1,1) -- (2,0) -- cycle;
\end{tikzpicture}

```



多条路径可用于同一条画图命令中，以空格分隔：

```
\begin{tikzpicture}
\draw (0,0) -- (0,1)
      (1,0) -- (1,1) -- (2,0) -- cycle;
\end{tikzpicture}
```



其它常用的路径还包括：

- 矩形、圆和椭圆：

```
\begin{tikzpicture}
\draw (0,0) rectangle (1.5,1);
\draw (2.5,0.5) circle [radius=0.5];
\draw (4.5,0.5) ellipse
      [x radius=1,y radius=0.5];
\end{tikzpicture}
```



- 直角、圆弧、椭圆弧：

```
\begin{tikzpicture}
\draw (0,0) |- (1,1);
\draw (1,0) -| (2,1);
\draw (4,0) arc (0:135:1);
\draw (6,0) arc (0:135:1 and 0.5);
\end{tikzpicture}
```



- 正弦、余弦曲线 (1/4 周期)：

```
\begin{tikzpicture}
\draw (0,0) sin (1,1);
\draw (0,1) sin (1,0);
\draw (2,1) cos (3,0);
\draw (2,0) cos (3,1);
\end{tikzpicture}
```



- 抛物线，用 bend 控制顶点：

```
\begin{tikzpicture}
\draw (0,0) parabola (1,2);
\draw (2,0) parabola
      bend (2.25,-0.25) (3,2);
\draw (4,0) parabola
      bend (4.75,2.25) (5,2);
\end{tikzpicture}
```



- 二次和三次 Bézier 曲线，分别使用一个和两个控制点：

```
\begin{tikzpicture}
\draw (0,0) .. controls
  (2,1) and (3,1) .. (3,0);
\draw (4,0) .. controls
  (5,1) .. (5,0);
\draw[help lines] (0,0)
  -- (2,1) -- (3,1) -- (3,0)
  (4,0) -- (5,1) -- (5,0);
\end{tikzpicture}
```



- 网格、函数图像，网格可用 `step` 参数控制网格大小，函数图像用 `domain` 参数控制定义域：

```
\begin{tikzpicture}
\draw[help lines,step=0.5]
  (-1,-1) grid (1,1);
\draw[->] (-1.5,0) -- (1.5,0);
\draw[->] (0,-1.5) -- (0,1.5);
\draw[domain=-1:1]
  plot(\x,{\x*\x*2 -1});
\end{tikzpicture}
```



7.2.2 TikZ 绘图命令和参数

除了 `\draw` 命令之外，TikZ 还提供了 `\fill` 命令用来填充图形，`\filldraw` 命令则同时填充和描边。除了矩形、圆等现成的闭合图形外，`\fill` 和 `\filldraw` 命令也能够填充人为构造的闭合路径。

```
\draw[...] <path>;
\fill[...] <path>;
\filldraw[...] <path>;
```

绘图参数可作为可选参数用在 `tikzpicture` 环境或 `\tikz` 命令时，参数会影响到所有具体的绘图命令；用在单个绘图命令 `\draw`、`\filldraw` 等时，只对这个命令起效。

TikZ 有数不清的绘图参数，这些参数令 TikZ 能够绘制丰富多彩的图像，同时也令 TikZ 难以精通。以下示例常用的一些绘图参数。

- `color/draw/fill=<color>` 为 `\draw` 或 `\fill` 等命令指定颜色。`draw` 和 `fill` 分别指定描边和填充的颜色，而 `color` 同时指定，也可以省略 `color=` 直接写颜色名称。

```
\begin{tikzpicture}[thick]
\draw[blue] (0,0) rectangle (1,1);
\filldraw[fill=yellow,draw=red]
  (2,0.5) circle [radius=0.5];
\end{tikzpicture}
```



- `thick=<length>/thin/semithick/...` 指定线条的粗细。

```
\begin{tikzpicture}
\draw[ultra thin] (0,0)--(0,2);
\draw[very thin] (0.5,0)--(0.5,2);
\draw[thin] (1,0)--(1,2);
\draw[semithick] (1.5,0)--(1.5,2);
\draw[thick] (2,0)--(2,2);
\draw[very thick] (2.5,0)--(2.5,2);
\draw[ultra thick] (3,0)--(3,2);
\end{tikzpicture}
```



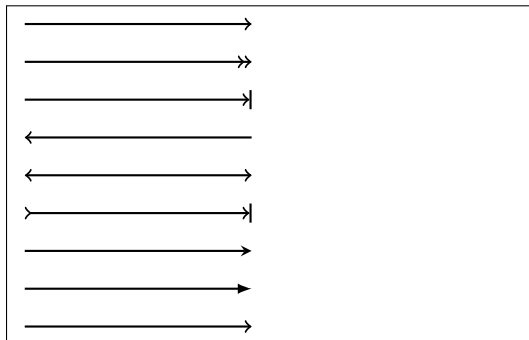
- `solid/dashed/dotted/dash dot/dash dot dot` 指定线条类型 (实线、虚线、点划线等)。与 `dashed` 对应地有 `densely dashed` 和 `loosely dashed`, 后三种类型同理。

```
\begin{tikzpicture}
\draw[dashed] (0,0) -- (0,2);
\draw[dotted] (0.5,0) -- (0.5,2);
\draw[dash dot] (1,0) -- (1,2);
\draw[dash dot dot] (1.5,0) -- (1.5,2);
\draw[densely dotted]
  (2,0) -- (3,2) -- (4,0) -- cycle;
\end{tikzpicture}
```



- `<arrow>-<arrow>` 指定线条首尾的箭头形式。复杂的箭头形式需要在导言区使用 `\usetikz-library {arrows.meta}`。

```
\begin{tikzpicture}[thick]
\draw[->] (0,4) -- (3,4);
\draw[->>] (0,3.5) -- (3,3.5);
\draw[->|] (0,3) -- (3,3);
\draw[<-] (0,2.5) -- (3,2.5);
\draw[<->] (0,2) -- (3,2);
\draw[->->|] (0,1.5) -- (3,1.5);
\draw[-stealth] (0,1) -- (3,1);
\draw[-latex] (0,0.5) -- (3,0.5);
\draw[-to] (0,0) -- (3,0);
\end{tikzpicture}
```



- `rounded corners[=radius]/sharp corners` 将路径转向处绘制成圆角/直角。可选参数 `<radius>` 控制圆角的半径。可以对某一段路径直接使用。

```
\begin{tikzpicture}
\draw[rounded corners]
  (0,0) rectangle (1,1);
\draw (2,0) -- (2,1)
  [rounded corners=.3cm]
  -- (3,1) -- (3.5,0)
  [sharp corners] -- cycle;
\end{tikzpicture}
```



- `scale/xshift/yshift/xslant/yslant/rotate` 设定图形的缩放、位移和旋转。

```
\begin{tikzpicture}
\draw[help lines](0,0) rectangle (1,1);
\draw[scale=1.5] (0,0) rectangle (1,1);
\draw[rotate=30] (0,0) rectangle (1,1);
\draw[help lines](2,0) rectangle (3,1);
\draw[yshift=4pt](2,0) rectangle (3,1);
\draw[help lines](4,0) rectangle (5,1);
\draw[xslant=0.4](4,0) rectangle (5,1);
\end{tikzpicture}
```



为了重复利用绘图参数，减少代码冗余，TikZ 引入了“样式”的概念，可以定义一个样式包含绘图参数，然后将样式作为一个参数用于绘图：

```
\begin{tikzpicture}
[myarrow/.style={blue,thick,->}]
\draw (0,0)--(0,1)--(2,1);
\draw[myarrow] (0,0)--(2,1);
\draw[myarrow,dotted]
(0,0)--(2,0)--(2,1);
\end{tikzpicture}
```



TikZ 还提供了 scope 环境，令绘图参数或样式在局部生效：

```
\begin{tikzpicture}
\draw (0,0) rectangle (2.5, 2.5);
\begin{scope}[thick,scale=0.5]
\draw (0,0) rectangle (2.5, 2.5);
\end{scope}
\end{tikzpicture}
```



7.2.3 TikZ 文字结点

TikZ 用 \node 命令绘制文字结点：

```
\node[options] (<name>) at (<coordinate>) {<text>;}
```

(<name>) 为结点命名，类似 \coordinate; at (<coordinate>) 指定结点的位置。这两者和前面的 <options> 都可以省略，只有 <text> 是必填的。

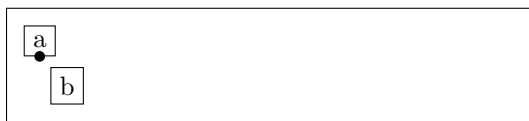
```
\begin{tikzpicture}
\node (A) at (0,0) {A};
\node (B) at (1,0) {B};
\node (C) at (60:1) {C};
\draw (A) -- (B) -- (C) -- (A);
\end{tikzpicture}
```



7.2.2 小节中的参数可用于 \node 命令的配置。除此之外，\node 还有一些特定的参数：

- anchor=<position> 令结点的某个角落 <position> 与 <coordinate> 对应。
- centered/above/below/left/right/above left/... [= <length>]
与 anchor 等效的选项。可选的 <length> 为结点相对于 <coordinate> 的距离。


```
\begin{tikzpicture}
\coordinate (A) at (1,1);
\fill (A) circle[radius=2pt];
\node[draw,anchor=south] at (A) {a};
\node[draw,below right=4pt] at (A) {b};
\end{tikzpicture}
```



- `shape=<shape>` 结点的形状，默认可用 `rectangle` 和 `circle`，可省略 `shape=` 直接写。在导言区使用命令 `\usetikzlibrary{shapes.geometric}` 可用更多的形状。
- `text=<color>` 结点文字的颜色。
- `node font={}` 结点文字的字体，形如 `\bfseries` 或 `\itshape` 等。

```
\begin{tikzpicture}
\node[circle,fill=blue,text=white,
node font={\bfseries}]
(A) at (0,0) {A node};
\node[rectangle,rounded corners,
draw=gray,
node font={\sffamily\slshape}]
(B) at (2,0) {B node};
\end{tikzpicture}
```



- `inner sep=<length>` / `outer sep=<length>` 结点边界向外和向内的额外距离。
- `minimum size=<length>` / `minimum height=<length>` / `minimum width=<length>` 结点的最小大小或最小高度/宽度。

`\node` 命令不仅为文字结点的位置命名，在 `\draw` 等命令中还可以使用某个结点的相对位置，以“东南西北”的方式命名：

```
\begin{tikzpicture}
\draw (0,0) circle[radius=1];
\fill (0,0) circle[radius=2pt];
\node[draw] (P) at (15:2) {center};
\draw[dotted] (0,0) -- (P.west);
\end{tikzpicture}
```



`\node` 命令的一种等效用法是在 `\draw` 等命令的路径中使用 `node`，不仅可以对某个位置标记结点，还能够对线标记：

```
\begin{tikzpicture}
\draw (2,1.5) node[above] {$A$}
-- node[above left] {$c$}
(0,0) node[left] {$B$}
-- node[below] {$a$}
(2.5,0) node[right] {$C$}
-- node[above right] {$b$}
cycle;
\end{tikzpicture}
```

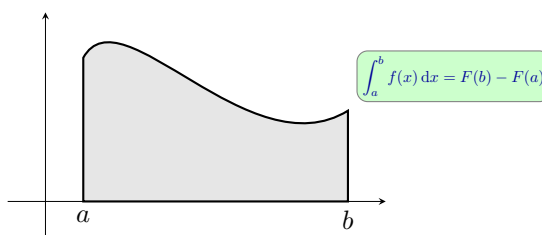


在此举一个较为复杂的例子，综合前面介绍过的各种路径、形状、文字结点和参数设置，见源代码 7.1。

```

\begin{tikzpicture}
\draw[-stealth,line width=0.2pt] (-0.5,0) -- (4.5,0);
\draw[-stealth,line width=0.2pt] (0,-0.5) -- (0,2.5);
\coordinate (a) at (0.5,1.9);
\coordinate (b) at (4,1.2);
\node[below] (a0) at (a |- 0,0) {$a$};
\node[below] (b0) at (b |- 0,0) {$b$};
\filldraw[fill=gray!20,draw,thick]
(a0) -- (a) .. controls (1,2.8) and (2.7,0.4) .. (b) -- (b0) -- cycle;
\node[above right,outer sep=0.2cm, rounded corners,
fill=green!20,draw=gray,text=blue!60!black,scale=0.6]
at (b) {$\displaystyle \int_a^b f(x) \mathrm{d}x = F(b) - F(a)$};
\end{tikzpicture}

```



源代码 7.1: TikZ 绘图示例源代码和效果。

7.2.4 在 TikZ 中使用循环

TikZ 通过 pgffor 功能宏包实现了简单的循环功能，语法为：

```
\foreach \a in {\list} {\commands}
```

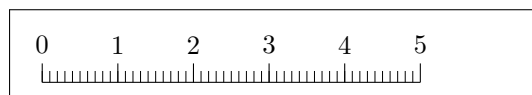
上述语法定义了 \a 为变量，在 {\commands} 中使用 \a 完成循环。

{\list} 可以直接将所有值写出来，如 1,2,3,4；也可以写成省略形式，如 1,2,...,10。

```

\begin{tikzpicture}
\draw (0,0)--(5,0);
\foreach \i in {0.0,0.1,...,5.0}
{
\draw[very thin]
(\i,0)--(\i,0.15);}
\foreach \I in {0,1,2,3,4,5}
{
\draw (\I,0)--(\I,0.25)
node[above] {\I};}
\end{tikzpicture}

```



\foreach 还可使用变量对参与循环：

```

\begin{tikzpicture}
\foreach \n/\t in
{0/\alpha,1/\beta,2/\gamma}
{
\node[circle,fill=lightgray,draw]
at (\n,0) {$\t$};}
\end{tikzpicture}

```



第八章 自定义 L^AT_EX 命令和功能

读到这一章之前，如果你确保掌握了前几章的知识并熟练运用，你已经能制作出内容和形式都相当丰富的文档了。但你可能还不会满足：我要如何制作一个简单但像样的毕业论文/书籍/简历模板，每次可以直接套用，而不是再在导言区写一堆代码？

本章的内容将有助于你实现这一个目标，让你能编写可重复利用的模块——宏包和文档类，并在其中自己定义命令和环境。不过作为入门手册，这些知识仍是不全面的。如果你不满足于此，需要参考更多资料，比如 [2, 10]。

8.1 自定义命令和环境

你也许已经意识到了，在本手册中介绍的所有命令都被包含在一个带颜色的矩形背景框中。我们并没有直接使用基础的 L^AT_EX 命令来实现这个效果，而是创建了一个**宏包**，并在其中定义了所需要的命令和环境。现在只需写成这样简单的形式：

```
\begin{command}
\cmd{dum}
\end{command}
```

```
\dum
```

这个例子中使用了一个新的环境 `command`。这个环境负责给命令代码加上一个带颜色的矩形背景框。同时还使用了一个命令：`\cmd`，这个命令负责输出命令的名字，包括前面的反斜线。

一旦想要修改命令代码的样式，比如更换颜色、加边框等等，可以通过改变 `command` 环境的定义来很容易地创建新的外观，而不是挨个修改每个命令示例。

8.1.1 定义新命令

使用如下命令可以定义你自己的命令：

```
\newcommand{\<name>}[\<num>]{\<definition>}
```

`\newcommand` 的基本用法需要两个必选参数，第一个参数 `<name>` 是要定义的命令名称（带反斜线），第二个参数 `<definition>` 是命令的具体定义。方括号里的参数 `<num>` 是可选的，用于指定新命令所需的参数数目（最多 9 个）。如果缺省可选参数，默认就是 0，也就是新定义的不带任何参数。

接下来的两个例子有助于理解。第一个例子定义了一个新的命令 `\tnss`。这个命令是本手册英文名称 “The Not So Short Introduction to L^AT_EX 2_ε” 的简写。如果需要在文档中多次使用本手册的名称，那么使用这个命令是一个非常方便的办法。

```
\newcommand{\tnss}{The not so Short
  Introduction to \LaTeXe}
This is ``\tnss'' \ldots{} ``\tnss''
```

```
This is “The not so Short Introduction to
LATEX 2ε” ... “The not so Short Introduction
to LATEX 2ε”
```

第二个例子演示了如何定义一个带参数的命令。在命令的定义中, 标记 #1 代表指定的参数。如果想使用多个参数, 可以依次使用 #2、……、#9 等标记。

```
\newcommand{\txsit}[1]{This is the
  \emph{#1} Short Introduction
  to \LaTeXe}
% in the document body:
\begin{itemize}
\item \txsit{not so}
\item \txsit{very}
\end{itemize}
```

- This is the *not so* Short Introduction to L^AT_EX 2_ε
- This is the *very* Short Introduction to L^AT_EX 2_ε

L^AT_EX 不允许使用 \newcommand 定义一个与现有命令重名的命令。如果需要修改命令定义的话, 使用 \renewcommand 命令。它使用与命令 \newcommand 相同的语法。

在某些情况之下, 使用 \providecommand 命令是一种比较理想的方案: 在命令未定义时, 它相当于 \newcommand; 在命令已定义时, 沿用已有的定义。

8.1.2 定义环境

与 \newcommand 命令类似, 可以用 \newenvironment 定义新的环境。它的语法如下所示:

```
\newenvironment{<name>}[<num>]{<before>}{<after>}
```

同样地, \newenvironment 命令有一个可选的参数。在 <before> 中的内容将在此环境包含的文本之前处理, 而在 <after> 中的内容将在遇到 \end{<name>} 命令时处理。

下面的例子演示了 \newenvironment 命令的用法:

```
\newenvironment{king}
{\rule{1ex}{1ex}%
  \hspace{\stretch{1}}}
{\hspace{\stretch{1}}%
  \rule{1ex}{1ex}}

\begin{king}
My humble subjects \ldots
\end{king}
```

■ My humble subjects ... ■

参数 <num> 的使用方式与 \newcommand 命令相同。L^AT_EX 还同样保证你不会不小心新建重名的环境。如果你确实希望改变一个现有的环境, 你可以使用命令 \renewenvironment, 它使用和命令 \newenvironment 相同的语法。

8.1.3 xparse 宏包简介

通过 \newcommand 和 \newenvironment 定义的命令或环境格式比较固定。如果需要定义带有多个可选参数、或者带星号的命令或环境, 可以使用 xparse 宏包¹。它提供了 \NewDocumentCommand 和 \NewDocumentEnvironment 等命令, 具体语法如下:

```
\NewDocumentCommand\<name>{\<arg spec>}{\<definition>}
\NewDocumentEnvironment{\<name>}{\<arg spec>}{\<before>}{\<after>}
```

¹L^AT_EX 2020-10-01 版本之后, xparse 宏包已集成在了格式之中, 不需要显式调用。

相比 `\newcommand` 和 `\newenvironment`, `xparse` 通过 $\{\langle arg spec \rangle\}$ 来指定参数的个数和格式。基本的参数格式见表 8.1。注意 $\{\langle arg spec \rangle\}$ 中的空格可以忽略。

表 8.1: `xparse` 参数格式

| 参数格式 | 说明 |
|---|--|
| <code>m</code> | 必选参数, 由 $\{\dots\}$ 给出 |
| <code>o</code> | 可选参数, 由 $[\dots]$ 给出; 未给出时返回 <code>-NoValue-</code> 标记 |
| <code>O{\langle default \rangle}</code> | 可选参数, 但在未给出时则返回默认值 $\langle default \rangle$ |
| <code>s</code> | 可选的星号 <code>*</code> |
| <code>+</code> | 修饰后面的 <code>m</code> 、 <code>o</code> 等, 表示允许在参数中分段 |

不同输入值在解析后的结果可以见表 8.2 中的示例。

表 8.2: `xparse` 参数示例

| 参数格式 | 输入值 | #1 | #2 | #3 |
|---|--------------------------------|---------------|-----------|---------|
| <code>m m</code> | $\{\text{foo}\}\{\text{bar}\}$ | foo | bar | |
| <code>o m</code> | $\{\text{foo}\}$ | -NoValue- | foo | |
| <code>o o m</code> | $[\text{foo}]\{\text{bar}\}$ | foo | -NoValue- | bar |
| <code>m O{\langle default \rangle}</code> | $\{\text{foo}\}$ | foo | default | |
| <code>m O{\langle default \rangle}</code> | $\{\text{foo}\}[\text{bar}]$ | foo | bar | |
| <code>m O{\langle default \rangle}</code> | $[\text{bar}]$ | 报错 | | |
| <code>s o m</code> | $*[\text{foo}]\{\text{bar}\}$ | \BooleanTrue | foo | bar |
| <code>s m O{\langle default \rangle}</code> | $\{\text{foo}\}$ | \BooleanFalse | foo | default |

`-NoValue-` 标记可以用 `\IfNoValueTF` 等命令来判断:

```
\IfNoValueTF{\langle argument \rangle}{\langle true code \rangle}{\langle false code \rangle}
\IfNoValueT{\langle argument \rangle}{\langle true code \rangle}
\IfNoValueF{\langle argument \rangle}{\langle false code \rangle}
```

举例如下:

```
% 百分号用于注释掉不必要的空格和换行符
\NewDocumentCommand\hello{om}
{
  \IfNoValueTF{#1}%
  {Hello, #2!}%
  {Hello, #1 and #2!}%
}
\hello{Alice}
\hello[Bob]{Alice}
```

Hello, Alice! Hello, Bob and Alice!

`\BooleanTrue` 和 `\BooleanFalse` 则可以用 `\IfBooleanTF` 等命令来判断:

```
\IfBooleanTF{<argument>}{<true code>}{<false code>}
\IfBooleanT{<argument>}{<true code>}
\IfBooleanF{<argument>}{<false code>}
```

举例如下：

```
\NewDocumentCommand\hereis{sm}
  {Here is \IfBooleanTF{#1}{an}{a} #2.}
\hereis{banana}
\hereis*{apple}
```

Here is a banana. Here is an apple.

需要注意的是，与命令不同，环境在定义时名字里面可以包含 *：

```
\NewDocumentEnvironment {mytabular} { o +m } {...} {...}
\NewDocumentEnvironment {mytabular*} { m o +m } {...} {...}
```

用 s 标记的 * 则应该放在 `\begin{<env>}` 的后面：

```
\NewDocumentEnvironment { envstar } { s }
  {\IfBooleanTF {#1} {star} {no star}} {}
\begin{envstar}*
\end{envstar}
```

与 `\renewcommand`、`\providecommand` 等命令类似，`xparse` 宏包也允许在命令或环境已有定义时做出相应的处理，具体见表 8.3。

表 8.3: `xparse` 提供的其他命令

| 定义命令 | 定义环境 | 已定义 | 未定义 |
|--------------------------------------|--|-------|------|
| <code>\NewDocumentCommand</code> | <code>\NewDocumentEnvironment</code> | 报错 | 给出定义 |
| <code>\RenewDocumentCommand</code> | <code>\RenewDocumentEnvironment</code> | 重新定义 | 报错 |
| <code>\ProvideDocumentCommand</code> | <code>\ProvideDocumentEnvironment</code> | 什么也不做 | 给出定义 |
| <code>\DeclareDocumentCommand</code> | <code>\DeclareDocumentEnvironment</code> | 重新定义 | 给出定义 |

8.2 编写自己的宏包和文档类

8.2.1 编写简单的宏包

如果定义了很多新的环境和命令，文档的导言区将变得很长，在这种情况下，可以建立一个新的 L^AT_EX 宏包来存放所有你自己定义的命令和环境，然后在文档中使用 `\usepackage` 命令来调用自定义的宏包。

写一个宏包的基本工作就是将原本在你的文档导言区里很长的内容拷贝到另一个文件中，这个文件需要以 `.sty` 作扩展名。你还需要加入一个宏包专用的命令：

```
\ProvidesPackage{<package name>}
```

```
% Demo Package by Tobias Oetiker
\ProvidesPackage{demopack}
\newcommand{\tnss}{The not so Short Introduction
    to \LaTeXe}
\newcommand{\txsit}[1]{The \emph{#1} Short
    Introduction to \LaTeXe}
\newenvironment{king}{\begin{quote}}{\end{quote}}
```

源代码 8.1: 宏包的一个最简示例。

这个命令应该放在你的宏包的最前面，并且一定要注意：*⟨package name⟩* 需要和宏包的文件名一致。`\ProvidesPackage` 让 \LaTeX 记录宏包的名称，从而在 `\usepackage` 命令再次调用同一个宏包的时候忽略之²。源代码 8.1 给出了一个小的宏包示例，其中包含了我们之前定义的一些命令。

8.2.2 在宏包中调用其它宏包

如果你想进一步把各种宏包的功能汇总到一个文件里，而不是在文档的导言区罗列一大堆宏包的话， \LaTeX 允许你在自己编写的宏包中调用其它宏包，命令为 `\RequirePackage`，用法和 `\usepackage` 一致：

```
\RequirePackage[⟨options⟩]{⟨package name⟩}
```

8.2.3 编写自己的文档类

当你更进一步，需要编写自己的文档类，如论文模板等，问题就稍稍麻烦了一些。首先，自己的文档类以 `.cls` 作扩展名，开头使用 `\ProvidesClass` 命令：

```
\ProvidesClass{⟨class name⟩}
```

当然了，*⟨class name⟩* 也需要和文档类的文件名一致。

但是有了上述命令和和你之前学到的 `\newcommand` 等，还并不能完成一个文档类的编写，因为诸如 `\chapter`、`\section` 等等许多常用的命令都是在文档类中定义的。事实上，许多时候我们只需要像调用宏包那样调用一个基本的文档类，省去许多不必要的麻烦。在你的文档类中调用其它文档类的命令是 `\LoadClass`，用法和 `\documentclass` 十分相像：

```
\LoadClass[⟨options⟩]{⟨package name⟩}
```

8.3 计数器

我们早就见识到了 \LaTeX 对文档元素自动计数的能力：章节符号、列表、图表……它们都是依靠 \LaTeX 提供的计数器功能完成的。

8.3.1 定义和修改计数器

定义一个计数器的方法为：

```
\newcounter{⟨counter name⟩}[⟨parent counter name⟩]
```

²但如果你以不同的选项多次引入宏包，则有可能会引起错误，见附录 B.1。

$\langle counter name \rangle$ 为计数器的名称。计数器可以有上下级的关系，可选参数 $\langle parent counter name \rangle$ 定义为 $\langle counter name \rangle$ 的上级计数器。

以下命令修改计数器的数值，`\setcounter` 将数值设为 $\langle number \rangle$ ；`\addtocounter` 将数值加上 $\langle number \rangle$ ；`\stepcounter` 将数值加一，并将所有下级计数器归零。

```
\setcounter{\langle counter name \rangle}{\langle number \rangle}
\addtocounter{\langle counter name \rangle}{\langle number \rangle}
\stepcounter{\langle counter name \rangle}
```

8.3.2 计数器的输出格式

计数器 $\langle counter \rangle$ 的输出格式由 `\the\langle counter \rangle` 表示，如我们在 5.5 一节见过的 `\thechapter` 等。这个值默认以阿拉伯数字形式输出，如果想改成其它形式，需要重定义 `\the\langle counter \rangle`，如将 `equation` 计数器的格式定义为大写字母：

```
\renewcommand\theequation{\Alph{equation}}
```

命令 `\Alph` 控制计数器 $\langle counter \rangle$ 的值以大写字母形式显示。下表列出所有可用于修改计数器格式的命令。注意：这些命令只能用于计数器，不能直接用于数字，如 `\roman{1}` 这样的命令会出错。

表 8.4: 计数器输出格式相关命令

| 命令 | 样式 | 范围 |
|------------------------|---------------------------------------|--------|
| <code>\arabic</code> | 阿拉伯数字（默认） | |
| <code>\alph</code> | 小写字母 | 限 0–26 |
| <code>\Alph</code> | 大写字母 | 限 0–26 |
| <code>\roman</code> | 小写罗马数字 | 限非负整数 |
| <code>\Roman</code> | 大写罗马数字 | 限非负整数 |
| <code>\fnsymbol</code> | 一系列符号，用于 <code>\thanks</code> 命令生成的脚注 | 限 0–9 |

注：`\fnsymbol` 使用的符号顺次为：* † ‡ § ¶ || ** †† ‡‡

计数器的输出格式还可以利用其它字符，甚至其它计数器的输出格式与之组合。如标准文档类里对 `\subsection` 相关的计数器的输出格式的定义相当于：

```
\renewcommand\thesubsection{\thesection.\arabic{subsection}}
```

8.3.3 L^AT_EX 中的计数器

- 所有章节命令 `\chapter`、`\section` 等分别对应计数器 `chapter`、`section` 等等，而且有上下级的关系。而计数器 `part` 是独立的。
- 有序列表 `enumerate` 的各级计数器为 `enumi`、`enumii`、`enumiii`、`enumiv`，也有上下级的关系。
- 图表浮动体的计数器就是 `table` 和 `figure`；公式的计数器为 `equation`。这些计数器在 `article` 文档类中是独立的，而在 `report` 和 `book` 中以 `chapter` 为上级计数器。
- 页码、脚注的计数器分别是 `page` 和 `footnote`。

我们可以利用前面介绍过的命令，修改计数器的样式以达到想要的效果，比如把页码修改成大写罗马数字，左右加横线，或是给脚注加上方括号：

```
\renewcommand\thepage{--\Roman{page}--}
\renewcommand\thefootnote{[\arabic{footnote}]}
```

我们在 5.5.1 小节中见过命令 `\pagenumbering`。它的内部机制是修改 `page` 计数器的格式 `\thepage`，并将计数器的值重置为 1。

最后介绍两个有用的计数器：

secnumdepth

L^AT_EX 标准文档类对章节划分了层级：

- 在 `article` 文档类里 `part` 为 0，`section` 为 1，依此类推；
- 在 `report` 和 `book` 文档类里 `part` 为 -1，`chapter` 为 0，`section` 为 1，等等。

`secnumdepth` 计数器控制章节编号的深度，如果章节的层级大于 `secnumdepth`，那么章节的标题、在目录和页眉页脚的标题都不编号（照常生成目录和页眉页脚），章节计数器也不计数。

可以用 `\setcounter` 命令设置 `secnumdepth` 为较大的数使得层级比较深的章节也编号，如设置为 4 令 `\paragraph` 也编号；或者设置一个较小的数以取消编号，如设置为 -1 令 `\chapter` 不编号。后者是生成不编号的章节的一个妙招，免去了手动使用 `\addcontentsline` 和 `\markboth` 的麻烦。

`secnumdepth` 计数器在 `article` 文档类里默认为 3（`subsubsection` 一级）；在 `report` 和 `book` 文档类里默认为 2（`subsection` 一级）。

tocdepth

`tocdepth` 计数器控制目录的深度，如果章节的层级大于 `tocdepth`，那么章节将不会自动写入目录项。默认值同 `secnumdepth`。

8.4 L^AT_EX 可定制的一些命令和参数

L^AT_EX 事实上有相当一些可以定制的命令和参数，不过对于修改样式或者开发宏包来说，这些定制项还远远不够。

对于用户来讲，容易定制的是这一些项目：

- 标题名称/前后缀等。表 8.5 列出了标准文档类里可定制的项目，表中所有的 L^AT_EX 命令都可以用 `\renewcommand` 来修改。
- 长度。前文在叙述各种排版元素时已经介绍过一些，现归纳于表 8.6。表中所有的长度命令可用 `\setlength` 来修改。大多数控制页面尺寸的长度参数在图 5.1 给出，此处不再赘述。

表 8.5: L^AT_EX 可定制的标题名称/前后缀

| 命令 | 默认值 | 含义 |
|------------------------------|-----------------|--|
| <code>\partname</code> | Part | <code>\part</code> 命令生成的标题前缀 |
| <code>\chaptername</code> | Chapter | <code>\chapter</code> 命令生成的标题前缀 |
| <code>\appendixname</code> | Appendix | 使用 <code>\appendix</code> 命令生成的附录部分的章标题前缀 |
| <code>\abstractname</code> | Abstract | 摘要环境 <code>abstract</code> 的标题名称 |
| <code>\contentsname</code> | Contents | <code>\tableofcontents</code> 命令生成的目录标题 |
| <code>\listfigurename</code> | List of Figures | <code>\listoffigures</code> 命令生成的插图目录标题 |
| <code>\listtablename</code> | List of Tables | <code>\listoftables</code> 命令生成的表格目录标题 |
| <code>\tablename</code> | Table | <code>table</code> 浮动体中 <code>\caption</code> 命令生成的标题前缀 |
| <code>\figurename</code> | Figure | <code>figure</code> 浮动体中 <code>\caption</code> 命令生成的标题前缀 |
| <code>\refname</code> | References | <code>thebibliography</code> 环境或 <code>\bibliography</code> 命令生成的参考文献标题 (<code>article</code> 文档类) |
| <code>\bibname</code> | Bibliography | <code>thebibliography</code> 环境或 <code>\bibliography</code> 命令生成的参考文献标题 (<code>report</code> 和 <code>book</code> 文档类) |
| <code>\indexname</code> | Index | <code>\printindex</code> 命令生成的索引标题 |

注：形如“第 X 章”和“第 X 部分”的中文章节标题不能直接由修改本表的命令得到，需要使用 `titlesec` 等宏包定制。如果使用 `ctex` 宏包或文档类，那么标题默认被修改成“第 X 章”和“第 X 部分”的形式，本表中的其它标题也修改为中文标题。详见 `ctex` 宏包的帮助文档。

表 8.6: L^AT_EX 可定制的长度参数

| 命令 | 默认值 | 含义 |
|--------------------------------|-------|---|
| <code>\fboxrule</code> | 0.4pt | <code>\fbox</code> 或 <code>\framebox</code> 等带框盒子的线宽 |
| <code>\fboxsep</code> | 3pt | <code>\fbox</code> 或 <code>\framebox</code> 等带框盒子的内边距 |
| <code>\arraycolsep</code> | 5pt | <code>array</code> 环境的表格项前后的间距 ^{注 1} |
| <code>\tabcolsep</code> | 6pt | <code>tabular</code> 环境的表格项前后的间距 ^{注 1} |
| <code>\arrayrulewidth</code> | 0.4pt | 表格线宽 |
| <code>\doublerulesep</code> | 2pt | 连续两根表格线之间的间距 |
| <code>\abovecaptionskip</code> | 10pt | <code>\caption</code> 命令上方的间距 ^{注 2} |
| <code>\belowcaptionskip</code> | 0pt | <code>\caption</code> 命令下方的间距 ^{注 2} |
| <code>\columnsep</code> | 10pt | 双栏排版下两栏的间距 |
| <code>\columnseprule</code> | 0pt | 双栏排版下两栏之间竖线的宽度 |

注 1: `\arraycolsep` 和 `\tabcolsep` 是每个表格项本身前后的间距（表格线前后无间距；@ 列格式会消除与前后表格项的间距）。两个表格项之间的间距相当于 `2\arraycolsep` 或 `2\tabcolsep`。

注 2: 在默认设置下，`\caption` 命令和位于它下方的图表之间无间距。宏包 `caption` 改善了这个问题。

附录 A 安装 T_EX 发行版

高德纳的 T_EX 程序开发于 20 世纪 80 年代，那时候电子计算机的运算能力有限，T_EX 还是大型服务器上的玩物。而如今个人计算机完全能够胜任排版的工作，并催生了用于个人计算机的工具集合——T_EX 发行版的发展。

本章会简单介绍如何安装 T_EX 发行版，以及保持发行版的内容紧跟最新。后者非常重要，因为 L^AT_EX 宏包是不断更新换代的。

A.1 T_EX 发行版简介

一个 **T_EX 发行版** 是 T_EX 排版引擎、支持排版的文件（基本格式、L^AT_EX 宏包、字体等）以及一些辅助工具的集合。各式各样的 T_EX 发行版经过十多年的发展，大浪淘沙，现今的两个主流发行版为：

- **T_EX Live**

T_EX Live 由类 UNIX 系统上的 teT_EX 发展并取而代之，最终成为跨平台的 T_EX 发行版。T_EX Live 自 2011 年起以年份作为发行版的版本号，保持了一年一更的频率。

MacT_EX 是 macOS (OS X) 系统下的一个定制化的 T_EX Live 版本，与 T_EX Live 同步更新。

- **MiK_TE_X**

MiK_TE_X 是主要用于 Windows 平台的一个稳定发展的 T_EX 发行版。在中国大陆曾经发行过“CT_EX 套装”，它是一个经过本地化配置的 MiK_TE_X，不过其配置较为过时，也不再有更新支持，使用起来可能有诸多问题，现已不推荐使用。

T_EX Live 和 MiK_TE_X 都集成了一个简单的 L^AT_EX 源代码编辑器 TeXworks (MacT_EX 则集成了类似的 TeXShop)。用户在安装完毕后，可直接使用 TeXworks 编写和编译 L^AT_EX 源代码。

A.1.1 安装发行版

T_EX Live

T_EX Live 的光盘镜像发布于 <https://www.tug.org/texlive/>¹。下载镜像到本地，挂载到虚拟光驱，或者用压缩工具解压后，在其根目录有几个用于安装的脚本：

- 用于 Windows 的批处理文件：
 - `install-tl-windows.bat` 双击启动图形界面安装程序，可以在图形安装界面的 Advanced 选项中定制安装；

¹Linux 发行版的软件源也提供 T_EX Live 的安装，不过不够完整，更新也不是很及时。建议直接从镜像安装。

- 在命令提示符中输入 `install-tl-windows.bat -no-gui` 启动文本界面安装程序。
- 用于 Linux 的 Perl 脚本 `install-tl` :
 - `install-tl` 启动文本界面安装程序;
 - `install-tl -gui` 启动图形界面安装程序。

另外也可以下载在线安装程序 `install-tl.zip`, 包含以上所有安装脚本。安装过程中会从 CTAN 软件源下载所有组件。

Linux 下 T_EX Live 安装完毕后, 一般还需要在 `root` 权限下进行以下操作, 使得 X_YL_AT_EX 能正确通过 `fontspec` 等宏包使用字体²:

1. 将 `texlive-fontconfig.conf` 文件复制到 `/etc/fonts/conf.d/09-texlive.conf`。
2. 运行 `fc-cache -fsv`。

MiK_TE_X

从 MiK_TE_X 官网 <https://www.miktex.org/> 下载名为 `basic-miktex-***.exe` 的 Windows 安装程序包。下载后直接双击打开, 按照程序的提示进行安装即可。

A.2 安装和更新宏包

T_EX Live 和 MiK_TE_X 分别提供了图形界面的宏包管理器 T_EX Live Manager 和 MiK_TE_X Package Manager, 用于安装、管理和更新宏包。一般情况下, 直接在图形界面的工具下按提示操作即可 (MiK_TE_X Package Manager 有普通权限和管理员权限的版本, 建议总是使用管理员权限的版本)。

两者也可以通过各自的命令行工具安装和更新宏包:

```
% TeX Live 命令行工具 tlmgr 的使用示例
% 安装/卸载宏包
tlmgr install <package-name>
tlmgr remove <package-name>
% 更新所有宏包 (包括 tlmgr 本身)
tlmgr update --all --self
% 列出所有可更新的宏包
tlmgr update --list
% 指定更新源地址
% <CTAN mirrors> 形如 https://mirrors.tuna.tsinghua.edu.cn/CTAN
tlmgr repository set <CTAN mirrors>/systems/texlive/tlnet
% 查看宏包信息, 加 --list 参数可列出宏包的所有文件
tlmgr info <package-name>

% MiKTeX 命令行工具 mpm 的使用示例
% 建议始终加 --admin 参数使用
% 安装/卸载宏包
mpm --admin --install <package-name>
```

²<https://www.tug.org/texlive/doc/texlive-zh-cn/texlive-zh-cn.pdf>, 可用 `texdoc texlive-zh-cn` 在本地打开。

```

mpm --admin --uninstall <package-name>
% 更新所有宏包
mpm --admin --update
% 列出所有可更新的宏包
mpm --admin --find-updates
% 指定更新源地址
mpm --admin --set-repository=<CTAN mirrors>/systems/win32/miktex/tm/packages
% 查看宏包信息
mpm --admin --print-package-info <package-name>

```

T_EX Live 默认安装所有宏包，而 MiK_TE_X 的安装程序只包含了 L^AT_EX 的一些基本宏包。从 T_EX Live 的光盘镜像和 MiK_TE_X 的安装包体积可见一斑。默认情况下，编译过程中如果遇到宏包未安装而报错的情况下，MiK_TE_X 会弹出一个对话框，让用户可以选择临时安装宏包，安装成功后继续编译。

A.2.1 手动安装宏包

如非万不得已，尽量不要手动安装宏包。绝大多数宏包都已打包到 T_EX Live 和 MiK_TE_X 两大发行版的安装源，可用宏包管理器安装。如果用户知道某个宏包的名称，但不确定是否在发行版中已打包，可在 CTAN 中搜索。

如果确实有手动安装宏包的需要，本小节的内容将有所帮助。在手动安装之前，有必要了解一下 T_EX 目录结构 (T_EX Directory Structure, TDS)。它是 T_EX 发行版中宏包、字体、帮助文档等文件的组织结构。TDS 有时也称为 TEXMF 树，取 T_EX+METAFONT 之意。

以 T_EX Live 为例，假设系统的 TEXMF 树根目录为 C:\texlive\2020\texmf-dist，其下有很多子目录，仅举几例：

tex/latex L^AT_EX 宏包。

doc/latex L^AT_EX 宏包的帮助文档。

source/latex L^AT_EX 宏包的源代码。

bibtex B_BL^AT_EX 工具相关文件，许多宏包配套的 B_BL^AT_EX 格式文件位于子目录 **bst** 中。

fonts/tfm T_EX 使用的字体文件，TFM 格式。

fonts/type1 PostScript 字体文件 (Type1)，PFB 格式。

fonts/opentype OpenType 格式的字体文件。

需要手动安装的宏包，一般已经按照上述目录结构打包完成。手动安装时，尽量不要拷贝到系统的 TEXMF 树，而是拷贝到发行版提供的用户 TEXMF 树，如 T_EX Live 的 C:\texlive\texmf-local。安装完成后，还需**刷新 T_EX 发行版的文件名数据库**，令新安装的宏包文件能够被系统找到。T_EX Live 用户须在 Windows 命令行或者 Linux 终端执行命令：

```

mktexlsr

```

MiK_TE_X 用户的命令为：

```

initexmf --update-fndb

```


附录 B 排除错误、寻求帮助

LaTeX 入门用户总会为两大问题头疼：我写的代码到底哪里出错了？如果想要实现某种用法该怎么办？本章首先总结了常见的 LaTeX 错误及应对的办法。

B.1 LaTeX 错误

当我们用排版引擎编译 LaTeX 代码时，命令行的窗口（终端）会显示大量信息（TeXworks 等编辑器会有一个区域显示这些信息）。当编译过程中出现错误时，信息将会停止在出错的地方，等待我们接下来的操作。

比如说我们有一个明显出错的例子：

```
\documentclass{article}
\begin{document}
Test \LaTeX{} and it's friends.
\end{document}
```

编译过程中遇到这个错误将会停顿下来，提示错误，并等待用户输入指令：

```
! Undefined control sequence.
1.3 Test \LaTeX
      {} and it's friends.
```

这种错误信息分两部分，前一部分提示了错误的信息，后一部分指出了错误发生的行号，以及通过错落的文字告知发生错误的命令所在位置。如上错误显示 \LaTeX 位置发生了错误，错误信息是“未定义的控制序列”，意思是 \LaTeX 是 TeX 编译器无法识别的一个命令，很显然是我们把 \LaTeX 的大小写写错了。

处理方式

出现错误时，编译过程将暂停，等待用户输入命令。用户可以直接敲回车跳过当前的错误，继续编译，相当于丢掉了写错的命令，将“Test and it's friends.”排版出来。但这个例子过于简单，有些复杂的代码中，有可能会由于一个小问题导致一连串的错误。此时可以选择按 **S/R/Q** 选择跳过接下来的所有错误，或者按 **X** 直接退出编译，将源代码中的错误修改后重新编译。

常见的 LaTeX 错误信息

我们在此总结一些经常发生、问题比较明确的 LaTeX 错误：

- **! Undefined control sequences.**

使用了未定义的命令。拼写错误是原因之一，如把 \LaTeX 写作 \Latex 这样。也有可能是没有调用某个宏包，但用了该宏包定义的命令。

- **! LaTeX error: Environment ... undefined.**

使用了未定义的环境。

- **! Missing \$ inserted.**

缺少数学环境的符号 \$。多由于将数学符号用在公式之外而导致。

- **Runaway argument?**

! Paragraph ended before ... was complete.

- **! File ended while scanning definition/use of ...**

这两个错误主要是由于漏写了包裹命令参数的花括号，导致识别参数时出现错误。许多编辑器的括号配对功能有助于检查和消除这类错误。

这类错误还有可能是由于前一次编译中断导致 .aux 等辅助文件不完整，再次编译读入不完整的文件产生错误。解决办法是删除辅助文件并重新编译。

- **! Extra alignment tab has been changed to \cr.**

- **! Misplaced \noalign.**

两个错误信息都与表格有关。

- 前者的字面意义是“一行中使用的列分隔符 & 太多”，有时可能确实是 & 的个数和列格式不匹配，但多数情况是漏掉了行尾的 \\ 命令。
- 后者常出现于漏掉了行尾的 \\ 命令而接着使用 \hline 命令画横线的时候。

- **! LaTeX Error: Lonely \item--perhaps a missing list environment.**

- **! LaTeX Error: Something's wrong--perhaps a missing \item.**

两个错误信息都与列表环境和 \item 命令有关。前者意味着在没有使用列表环境的情况下用了 \item；后者则相反，是在列表环境中漏了 \item。

- **! I can't find file `...'.**

- **! LaTeX Error: File `...' not found.**

两个错误都意味着缺少文件。

- 如果使用 \input 或者 \include 命令添加文件，出现上述错误的原因当然是文件不存在或者文件名不对；
- 如果错误提示里的文件名带 .sty 或者 .cls 扩展名，那么很显然，是因为**没有安装所需的宏包或文档类**。

- **! LaTeX Error: Missing \begin{document}.**

字面上是缺少 \begin{document}，实际上往往是由于在 \begin{document} 之前（导言区）输入了文字或某些命令。

- **! LaTeX Error: Can be used only in preamble.**

与上一条相反，由于将必须用于导言区的命令在 \begin{document} 之后使用而产生。

- **! LaTeX Error: \begin{...} on input line ... ended by \end{...}.**

环境首尾不匹配。比如 \begin{enumerate} 用了 \end{itemize} 结尾。或者也可能是由于漏写了 \begin 或者 \end 命令。

- **! LaTeX Error: Option clash for package `...`.**

以**不同选项**重复调用宏包造成冲突。有可能是因为其它宏包内部事先调用了这个宏包，用户再次带选项调用而导致冲突。解决问题的办法是去掉重复调用的宏包。如果宏包允许的话，尽量使用其定义的命令改变设置，减少宏包选项的使用。

- **! LaTeX Error: Command ... already defined,
or name \end... illegal ...**

使用 `\newcommand` 或 `\newenvironment` 定义已有的命令/环境时产生的错误。如果自己确实作了定义，可考虑用 `\renewcommand` 或 `\renewenvironment` 定义；如果是宏包定义的命令产生了这个错误，则属于隐性的宏包冲突。

相比之前的“Option clash”，隐性宏包冲突是更难以解决的问题，对各种宏包不熟悉的用户，尤其是使用模板的用户而言，往往难以下手。用户可尝试查找引起冲突的宏包的帮助文档。详尽的手册里通常会告知用户这个宏包应当在某个宏包的前面/后面调用，或者不能与某个宏包一起调用。如果是模板调用了大量宏包导致冲突，可联系模板的作者解决。

- **! LaTeX Error: Unknown option `...' for package `...'.**

调用宏包时指定了不能被其识别的选项。此时应该查找宏包的帮助文档来解决问题。

- **! Package `...' error: ...**

宏包或文档类自定义的错误，由于不正确地使用宏包里的命令而导致。此时应该查找宏包的帮助文档来解决问题。

B.2 查找帮助文档

无论是 $\text{T}_\text{E}\text{X}$ Live 还是 $\text{MiK}\text{T}_\text{E}\text{X}$ ，提供了一个命令行模式的程序 `texdoc`。比如对 5.5.3 小节的 `fancyhdr` 宏包感兴趣，这时在 Windows 命令提示符或者 Linux 终端输入以下命令，则会弹出宏包的帮助文档 `fancyhdr.pdf`：

```
texdoc fancyhdr
```

除了宏包的帮助文档外， $\text{T}_\text{E}\text{X}$ 发行版还包括了各类有用的文档，有一部分在参考文献中给出。

如果不熟悉命令行工具的话， $\text{T}_\text{E}\text{X}$ Live 提供了一个图形界面的程序 `TeXdoc GUI`。打开后，可以看到程序里的许多按钮，分别代表某一类的帮助文档。除此之外，点击 `File Search` 弹出搜索框，输入想要搜索的宏包和文件并按回车键，`TeXdoc GUI` 会弹出它搜索到的所有结果，可点击任意一项来打开文档。

当然对于初学者，有一个现实而棘手的问题：**某个命令到底是 $\text{L}^{\text{A}}\text{T}_\text{E}\text{X}$ 自有的，还是哪个宏包提供的？**很遗憾地说，除了通过慢慢积累、熟悉较多宏包之外，没有很方便的办法解决这个问题，因为 $\text{L}^{\text{A}}\text{T}_\text{E}\text{X}$ 的宏包实在太丰富了。本手册末尾的索引给出了所有在本手册见到的命令和环境，其中哪些命令和环境需要调用哪个宏包才能使用，一目了然。但是这个索引远远不够。

解决这个问题有几点可行的办法：

1. 查询一些综述性的资料，如总结所有 $\text{L}^{\text{A}}\text{T}_\text{E}\text{X}$ 自带命令的文档 [12]、 $\text{L}^{\text{A}}\text{T}_\text{E}\text{X}$ 符号大全 [14] 等；
2. 在互联网上搜索自己不清楚的命令；
3. 在论坛上提问求助有经验的人。

B.3 常用宏包简介

此处不包含设置字体或数学符号的宏包，它们已经在表 5.4 中列出。

B.3.1 文字、公式和符号

| | |
|--------------|--|
| amsmath | \mathcal{AMS} 数学公式扩展。 |
| mathtools | 数学公式扩展宏包，提供了公式编号定制和更多的符号、矩阵等。 |
| amsfonts | \mathcal{AMS} 扩展符号的基础字体支持。 |
| amssymb | 在 amsfonts 基础上将 \mathcal{AMS} 扩展符号定义成命令。 |
| bm | 提供将数学符号加粗的命令 <code>\bm</code> 。 |
| unicode-math | 使用 Unicode 数学字体。 |
| nicematrix | 排版复杂矩阵。 |
| siunitx | 以国际单位规范排版物理量的单位。 |
| mhchem | 排版化学式和方程式。 |
| tipa | 排版国际音标。 |

B.3.2 排版元素

| | |
|-------------|---|
| ulem | 提供排版可断行下划线的命令 <code>\uline</code> 以及其它装饰文字的命令。 |
| endnote | 排版尾注。 |
| marginnote | 改善的边注排版功能。 |
| multicol | 提供将内容自由分栏的 <code>multicols</code> 环境。 |
| multitoc | 生成多栏排版的目录。 |
| minitoc | 为章节生成独立的小目录。 |
| glossaries | 生成词汇表。 |
| verbatim | 对原始的 <code>verbatim</code> 环境的改善。提供了命令 <code>\verbatiminput</code> 调用源文件。 |
| fancyvrb | 提供了代码排版环境 <code>Verbatim</code> 以及对版式的自定义。 |
| listings | 提供了排版关键字高亮的代码环境 <code>lstlisting</code> 以及对版式的自定义。类似宏包有 <code>minted</code> 等。 |
| algorithmic | 一个简单的实现算法排版的宏包。如果要生成浮动体的话，需要搭配 <code>algorithm</code> 宏包使用。 |
| algorithm2e | 较为复杂的、可定制的计算排版宏包。类似宏包有 <code>algorithmicx</code> 等。 |
| amsthm | 定制定理环境。类似宏包包括 <code>theorem</code> 、 <code>ntheorem</code> 、 <code>thmtools</code> 等。 |
| mdframed | 排版可自动断页的带边框文字段落，提供边框样式的定制功能。 |
| tcolorbox | 以 <code>TikZ</code> 为基础提供排版样式丰富的彩色盒子的功能。 |

B.3.3 图表和浮动体

| | |
|------------|---|
| array | 对表格列格式的扩展。 |
| booktabs | 排版三线表。 |
| tabularx | 提供 tabularx 环境排版定宽表格，支持自动计算宽度的 X 列格式。 |
| arydshln | 支持排版虚线表格线。 |
| colortbl | 支持修改表格的行、列、单元格的顏色。 |
| multirow | 支持合并多行单元格。 |
| makecell | 支持在单元格里排版多行内容（嵌套一个单列的小表格）。 |
| diagbox | 排版斜线表头。 |
| longtable | 提供排版跨页长表格的 longtable 环境。 |
| ltxtable | 为跨页长表格提供 tabularx 的 X 列格式。 |
| tabularray | 排版复杂表格（基于 L ^A T _E X3 实现）。 |
| graphicx | 支持插图。 |
| bmpsize | latex + dvipdfmx 命令下支持 BMP/JPG/PNG 等格式的位图。 |
| epstopdf | pdf _l atex 命令下支持 EPS 格式的矢量图。 |
| wrapfig | 支持简单的文字在图片周围的绕排。 |
| caption | 控制浮动体标题的格式。类似宏包有 keyfloat 等。 |
| subcaption | 提供子图表和子标题的排版。类似宏包有 subfigure 和 subfig 等。 |
| bicaption | 生成双语浮动体标题。 |
| float | 为浮动体提供不浮动的 H 模式；提供自定义浮动体结构的功能。 |

B.3.4 修改版式

| | |
|-------------|----------------------------------|
| geometry | 修改页面尺寸、页边距、页眉页脚等参数。 |
| fancyhdr | 修改页眉页脚格式，令页眉页脚可以左对齐、居中、右对齐。 |
| titlesec | 修改章节标题 \chapter、\section 等的格式。 |
| titletoc | 修改目录中各条目的格式。类似宏包有 tocloft 等。 |
| tocbibind | 支持将目录、参考文献、索引本身写入目录项。 |
| footmisc | 修改脚注 \footnote 的格式。 |
| indentfirst | 令章节标题后的第一段首行缩进。 |
| enumerate | 提供简单的自定义标签格式的 enumerate 环境。 |
| enumitem | 修改列表环境 enumerate 和 itemize 等的格式。 |
| lettrine | 生成段落首字母大写的效果。 |

参考文献

§ L^AT_EX 经典书籍:

- [1] Leslie Lamport. *L^AT_EX: A Document Preparation System*, 2nd edition. Addison-Wesley, Reading, Massachusetts, 1994, ISBN 0-201-52983-1.
- [2] Frank Mittelbach, Michel Goossens, Johannes Braams, David Carlisle, Chris Rowley. *The L^AT_EX Companion*, 2nd edition. Addison-Wesley, Reading, Massachusetts, 2004, ISBN 0-201-36299-6.
- [3] Michel Goossens, Sebastian Rahtz and Frank Mittelbach. *The L^AT_EX Graphics Companion*, 2nd edition. Addison-Wesley, Reading, Massachusetts, 1997, ISBN 0-301-50892-0.
- [4] Helmut Kopka and Patrick W. Daly, *Guide to L^AT_EX*, 4th edition. Addison-Wesley, Reading, Massachusetts, 2004, ISBN 0-321-17385-6.
- [5] 刘海洋. *L^AT_EX 入门*. 电子工业出版社, 北京, 2013, ISBN 978-7-121-20208-7.

§ T_EX 经典书籍, 介绍底层的命令和排版方式, 有志深入学习 L^AT_EX 者或参与宏包开发者必读。有一些书籍已经开放网络资源:

- [6] Donald E. Knuth. *The T_EXbook*, Volume A of *Computers and Typesetting*, 2nd edition. Addison-Wesley, Reading, Massachusetts, 1984, ISBN 0-201-13448-9.
- [7] Paul Abrahams, Kathryn A. Hargreaves, Karl Berry. *T_EX for the impatient*, 2nd edition. Addison-Wesley, Reading, Massachusetts, 1984, ISBN 0-201-51375-7.
- [8] Victor Eijkhout. *T_EX by Topic, A T_EXnician's Reference*. Addison-Wesley, Reading, Massachusetts, 1992, ISBN 0-201-56882-9.

[CTAN://info/impatient/book.pdf](http://ctan.org/info/impatient/book.pdf) (texdoc impatient)

§ T_EX 发行版内置的资源 (除宏包帮助手册之外的一些文档), 可以在本地使用 texdoc 命令查找, 也可以在 CTAN 上找到:

- [9] L^AT_EX Project Team. *L^AT_EX 2_ε for authors*.
[CTAN://macros/latex/base/usrguide.pdf](http://ctan.org/macros/latex/base/usrguide.pdf) (texdoc usrguide)
- [10] L^AT_EX Project Team. *L^AT_EX 2_ε for class and package writers*.
[CTAN://macros/latex/base/clsguide.pdf](http://ctan.org/macros/latex/base/clsguide.pdf) (texdoc clsguide)
- [11] L^AT_EX Project Team. *L^AT_EX 2_ε font selection*.
[CTAN://macros/latex/base/fntguide.pdf](http://ctan.org/macros/latex/base/fntguide.pdf) (texdoc fntguide)

- [12] Karl Berry, Jim Hefferon, Vincent Belaïche. *L^AT_EX 2_ε: An unofficial reference manual*.
[CTAN://info/latex2e-help-texinfo/latex2e.pdf](https://ctan.org/info/latex2e-help-texinfo/latex2e.pdf) (texdoc latex2e)
- [13] Jürgen Fenn. *An essential guide to L^AT_EX 2_ε usage: Obsolete commands and packages* (English translation).
[CTAN://info/l2tabu/english/l2tabuen.pdf](https://ctan.org/info/l2tabu/english/l2tabuen.pdf) (texdoc l2tabuen)
- [14] Scott Pakin. *The Comprehensive L^AT_EX Symbol List*.
[CTAN://info/symbols/comprehensive/symbols-a4.pdf](https://ctan.org/info/symbols/comprehensive/symbols-a4.pdf) (texdoc symbols-a4)
- [15] Winston Chang. *L^AT_EX 2_ε Cheat Sheet*.
[CTAN://info/latexcheat/latexcheat/latexsheet.pdf](https://ctan.org/info/latexcheat/latexcheat/latexsheet.pdf) (texdoc latexcheat)

§ CTAN 上的其它网络资源:

- [16] Stephen G. Hartke. *A Survey of Free Math Fonts for T_EX and L^AT_EX*.
[CTAN://info/Free_Math_Font_Survey/survey.pdf](https://ctan.org/info/Free_Math_Font_Survey/survey.pdf)

§ 其它网络渠道可获得的资源:

- [17] The T_EX FAQ List. *The T_EX Frequently Asked Question List*.
<https://texfaq.org>
- [18] learnlatex.org. *Learn L^AT_EX online*.
<https://www.learnlatex.org>
- [19] Palle Jørgensen. *The L^AT_EX font catalogue*, a font catalogue of L^AT_EX font packages.
<https://tug.org/FontCatalogue/>
- [20] Indian T_EX users group. *L^AT_EX Tutorials: A primer*.
<https://www.tug.org/twg/mactex/tutorials/ltxprimer-1.0.pdf>
- [21] Overleaf. *Overleaf Documentation*.
<https://www.overleaf.com/learn>
- [22] 黄新刚. 雷太赫排版系统简介 (*L^AT_EX Notes*), 第二版.
<https://github.com/huangxg/lnotes/raw/master/lnotes2.pdf>
(旧版: texdoc latex-notes-zh-cn)

GNU Free Documentation License

Version 1.3, 3 November 2008

Copyright © 2000, 2001, 2002, 2007, 2008 Free Software Foundation, Inc. <<https://fsf.org/>>

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

0. Preamble

The purpose of this License is to make a manual, textbook, or other functional and useful document “free” in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondly, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of “copyleft”, which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The “**Document**”, below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as “**you**”. You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A “**Modified Version**” of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A “**Secondary Section**” is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document’s overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The “**Invariant Sections**” are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section

does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The “**Cover Texts**” are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A “**Transparent**” copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not “Transparent” is called “**Opaque**”.

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The “**Title Page**” means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, “Title Page” means the text near the most prominent appearance of the work’s title, preceding the beginning of the body of the text.

The “**publisher**” means any person or entity that distributes copies of the Document to the public.

A section “**Entitled XYZ**” means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as “**Acknowledgements**”, “**Dedications**”, “**Endorsements**”, or “**History**”.) To “**Preserve the Title**” of such a section when you modify the Document means that it remains a section “Entitled XYZ” according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document’s license notice requires Cover Texts, you must enclose the

copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

- A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.
- B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.
- C. State on the Title page the name of the publisher of the Modified Version, as the publisher.
- D. Preserve all the copyright notices of the Document.
- E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.
- F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.
- G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.
- H. Include an unaltered copy of this License.
- I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.

- J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the “History” section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.
- K. For any section Entitled “Acknowledgements” or “Dedications”, Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.
- L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.
- M. Delete any section Entitled “Endorsements”. Such a section may not be included in the Modified Version.
- N. Do not retitle any existing section to be Entitled “Endorsements” or to conflict in title with any Invariant Section.
- O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version’s license notice. These titles must be distinct from any other section titles.

You may add a section Entitled “Endorsements”, provided it contains nothing but endorsements of your Modified Version by various parties—for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled “History” in the various original documents, forming one section Entitled “History”; likewise combine any sections Entitled “Acknowledgements”, and any sections Entitled “Dedications”. You must delete all sections Entitled “Endorsements”.

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an “aggregate” if the copyright resulting from the compilation is not used to limit the legal rights of the compilation’s users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document’s Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled “Acknowledgements”, “Dedications”, or “History”, the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, or distribute it is void, and will automatically terminate your rights under this License.

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, receipt of a copy of some or all of the same material does not give you any rights to use it.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See <https://www.gnu.org/licenses/>.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License “or any later version” applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation. If the Document specifies that a proxy can decide which future versions of this License can be used, that proxy’s public statement of acceptance of a version permanently authorizes you to choose that version for the Document.

11. RELICENSING

“Massive Multiauthor Collaboration Site” (or “MMC Site”) means any World Wide Web server that publishes copyrightable works and also provides prominent facilities for anybody to edit those works. A public wiki that anybody can edit is an example of such a server. A “Massive Multiauthor Collaboration” (or “MMC”) contained in the site means any set of copyrightable works thus published on the MMC site.

“CC-BY-SA” means the Creative Commons Attribution-Share Alike 3.0 license published by Creative Commons Corporation, a not-for-profit corporation with a principal place of business in San Francisco, California, as well as future copyleft versions of that license published by that same organization.

“Incorporate” means to publish or republish a Document, in whole or in part, as part of another Document.

An MMC is “eligible for relicensing” if it is licensed under this License, and if all works that were first published under this License somewhere other than this MMC, and subsequently incorporated in whole or in part into the MMC, (1) had no cover texts or invariant sections, and (2) were thus incorporated prior to November 1, 2008.

The operator of an MMC Site may republish an MMC contained in the site under CC-BY-SA on the same site at any time before August 1, 2009, provided the MMC is eligible for relicensing.