简单粗暴 LaTeX

Kanglong Wu

本手册是wklchris-GitHub的 LaTeX-cn 项目

September 1, 2016

目录 | 0

序 1

第一稿序

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

1 - 我希望能够吸引初学者快速上手，解决手头的问题。因此去掉了枯燥的讲解和无穷无尽的宏包用法介绍，直接使用实例；

2 - 力求突出实用性。当然，也会提点一些可以深入学习的内容，读者可以自行查阅，或者阅读本手册中的扩展阅读章节（即带星号 * 的章节）。

3 - 本手册使用的编辑器为 TEXStudio，而非之前的商业软件 WinEdt. 这使得学习 LATEX 的门槛更低。当然了，你有权使用任何编辑器，我并不是说 TEXStudio 就是最好的。事实上在你熟练以后，使用记事本 + 命令行的方式生成文档都是没有问题的。

由于工作全部由我一人完成，限于视野，难免存在错漏之处。恳请读者指正。有任何使用中遇到的手册中无法解决的问题，欢迎向我提出。

Mail: wklchris@hotmail.com

Chris Wu
September 1, 2016 于湖北

2 LaTeX 基础

2.1 第一份文稿

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

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

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

之后你可以在编译窗口输入一篇小文档，并保存为 tex 文件进行测试：

```
\documentclass{article}
\usepackage[slantfont,boldfont]{xeCJK}
\setCJKmainfont[BoldFont=SimHei,ItalicFont=KaiTi]{SimSun}
\begin{document}
        Hello, world!
        你好，世界！
\end{document}
```

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

2.2 认识 LaTeX

2.2.1 命令与环境

LaTeX 中的**命令**通常是由一个反斜杠加上命令名称，再加上花括号内的参数构成的。比如：

```
\documentclass{article}
```

如果有一些选项是备选的，那么通常会在花括号前用方括号标出。比如：

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

在 LaTeX 中，还有一种重要的指令叫做**环境**。它定义了一整个区段的内容都受到环境（即其本身）的控制。这个区段是从 `\begin{environment}` 开始，到 `\end{environment}` 结束的。。比如：

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

同样的，环境也可以使用备选选项，只需要写在 begin 的后面就行了。

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

2.2.2 保留字符

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

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

它们的作用分别是：

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

以上除了反斜杠外，均能用前加反斜杠的行驶输出。即你只需要键入：

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

唯独反斜杠的输出比较头痛，你可以尝试：

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

另外需要说明的是，在中文格式下，波浪线~ 用来输出一个小空格，不能够直接输出。你可以通过命令`$\sim$`，或者仿上换成 char126 来产生一个。

2.2.3 导言区

任何一份 LaTeX 文档都应当包含以下结构：

```
1 \documentclass[options]{doc-class}
2 \begin{document}
3         ...
4 \end{document}
```

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

文档类的参数 doc-class 和可选选项 *options* 有以下取值：
在本文中，多数的文档类提及的均为 report/book 类。如果有 article 类将会特别指明。其余的文档类不予说明。本手册排版即使用了 book 类。

在导言区最常见的是**宏包**的加载工作，命令形如：`\usepackage{package}`。通俗地讲，宏包是指一系列已经制作好的功能“模块”，在你需要使用一些原生

表 2.1: 文档类和选项

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

LaTeX 不带有的功能时，只需要调用这些宏包就可以了。比如本文的代码就是利用 `listings` 宏包实现的。

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

2.2.4 错误的排查

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

- **Errors 错误**：严重的错误。一般地，编译若通过了，该项是零。
- **Warnings 警告**：一些不影响生成文档的瑕疵。
- **Bad Boxes 坏箱**[1]：指排版中出现的长度问题，比如长度超出（Overfull）等。后面的 Badness 表示错误的严重程度，程度越高数值越大。这类问题需要检查，排除 Badness 高的选项。

你可以向上翻阅控制台记录，来找到 Warning 开头的记录，或者 Overfull/Underfull 开头的记录。这些记录会指出你的问题出在哪一行 (比如 line 1-2) 或

[1]Box 是 LaTeX 中的一个特殊概念，具体将在这里进行讲解。

者在输出 pdf 的哪一页（比如 active [12]。注意，这个 12 表示页面显示页码或者计数器计数页码，而不是文件打印出来的真实页码）。此外你可以还需要了解：

- 值得指出的是，由于 LaTeX 的编译原理（第一次生成 aux 文件，第二次再引用它），目录想要合理显示**需要连续编译两次**。在连续编译两次后，你会发现一些 Warnings 会在第二次编译后消失。在 TeXStudio 中，你可以只单击一次“构建并查看”，它会检测到文章的变化并自动决定是否需要编译两次。
- 对于大型文档，寻找行号十分痛苦。你需要学会合理地拆分 tex 文件，参阅第2.13节的内容。

2.2.5 文件输出

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

.sty	宏包文件
.aux	用于储存交叉引用信息的文件。因此，在更新交叉引用（公式编号、大纲级别）后，需要编译两次才能正常显示。
.log	记录上次编译的信息。
.toc	目录文件。
.idx	如果文档中包含索引，该文件用于储存索引信息。
.lof	图形目录。
.lot	表格目录。

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

2.3 标点与强调

2.3.1 引号

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

英文下的引号嵌套需要借助\thinspace 命令分隔，比如：

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

“‘Max’ is here.”

中文下的单引号和双引号你可以用中文输入法直接输入。

2.3.2 破折、省略号与短横

英文的短横分为三种：

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

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

至于减号，也许你应该借助于数学环境来输入：$-$.

2.3.3 强调：粗与斜

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

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

2.3.4 下划线与删除线

LaTeX 原生提供了\underline命令简直烂的可以，建议你使用 ulem 宏包下的 uline 命令代替。ulem 宏包还提供了一系列有用的命令：

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

下划线
双下划线
虚下划线
点下划线
波浪线
删除线
斜删除线

需要注意的是，ulem 宏包重定义了\emph 命令，使得原来的加斜强调变成了下划线、原来的两次强调就取消强调变成了两次强调就双下划线，同时也支持换行文本。

2.3.5 其他

角度符号或者温度符号需要借助数学模式$...$ 输入：

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

30 ° 三角形
37 °C

然后是欧元符号，可能需要用到 textcomp 宏包，然后调用\texteuro命令就可以了。

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

再次是注音符号，比如 ô，也常用于拼音声调，参考注音符号表部分的附录内容。如果你想输入音标，请使用 tipa 宏包[2]，同样参考附录 A。

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

[2]tipa 需要在 amsmath 宏包之前加载，否则\! 相关的命令可能报错

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

XeLaTeX BibTeX

2.4 格式控制

首先了解一下 LaTeX 的长度单位：

pt point，磅。
pc pica。1pc=12pt，四号字大小
in inch，英寸。1in=72.27pt
bp bigpoint，大点。1bp=$\frac{1}{72}$in
cm centimeter，厘米。1cm=$\frac{1}{2.54}$in
mm millimeter，毫米。1mm=$\frac{1}{10}$cm
sp scaled point。TeX 的基本长度单位，1sp=$\frac{1}{65536}$pt
em 当前字号下，大写字母 M 的宽度。
ex 当前字号下，小写字母 x 的高度。

然后是几个常用的长度宏，更多的长度宏使用会在表格、分栏等章节提到。

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

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

2.4.1 换行与分段

通常的换行方法非常简单：LaTeX 会自动转行，然后在每一段的末尾，只需要输入两个回车即可完成分段。

在下划线一节的例子中已经给出了强制换行的方式，即两个反斜：\\. 不过这样做的缺点在于下一行段首缩进会消失。你可以用\par 来解决这个问题。

2.4.2 分页

用\newpage命令开始新的一页。

用\clearpage命令清空浮动体队列[3]，并开始新的一页。

用\cleardoublepage命令清空浮动体队列，并在偶数页上开始新的一页。

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

2.4.3 缩进、对齐与行距

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

[3]参见浮动体这一节的内容。

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

插入制表位、悬挂缩进、行距等复杂的调整参考这部分的内容。

2.5 字体与颜色

这一节只讨论行文中字体使用。数学环境内字体使用请参考这一节的内容。

2.5.1 字族、字系与字形

字体 (typeface) 的概念非常令人恼火，在电子化时代，基本上也都以字体 (font) 作为替代的称呼。宋体、黑体、楷体，这属于**字族**；对应到西文就是罗马体、等宽体等。加粗、加斜属于**字系和字形**。五号、小四属于**字号**。这三者大概可以并称**字体**[4]。

2.5.2 中西文“斜体”

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

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

2.5.3 原生字体命令

LaTeX 提供了基本的字体命令，包括表2.2中显示的内容。

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

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

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

更多自定义命令的语法请参考这一节。

然后就是字号的命令。行文会有一个默认的“标准”字号，比如你在 documentclass 的选项中设置的 12pt（如果你设置了的话）。LaTeX 给出了一系列“相对字号命令”，列出如表2.4。在平常使用中，小四为 12pt，五号为 10.5pt。

如果你想设置特殊的字号，使用：

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

[4]本文中的字族、字系等称呼难以找到统一标准，可能并不是准确的名称。

表 2.2: LaTeX 字体命令表

表 2.3: LaTeX 字体命令表

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

表 2.4: 相对字号命令表

表 2.5: 相对字号命令表

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

其中 font-size 填数字，以 pt 为单位；一般而言，line-height 填\baselineskip这个长度宏。

2.5.4 西文字体

在 XeLaTeX 编译下，一般使用 fontspec 宏包来选择西文字体。注意：fontspec 宏包可能会明显增加编译所需的时间。

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

```
  \lucida{This is Lucida Calligraphy}
```

2.5.5 中文字体

对于中文，需要参考 XeLaTeX 编译下的 xeCJK 宏包的使用。
比如在导言区：

```
\usepackage[slantfont,boldfont]{xeCJK}
  \xeCJKsetup{CJKMath=true}
  \setCJKmainfont[BoldFont=SimHei]{SimSun}
% 这里把SimHei直接写成中文“黑体”似乎也可以
```

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

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

上面提到的 xeCJKsetup 有下列可以定制的参数，下划线为默认值：

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

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

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

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

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

然后到`C:\list.txt` 中进行查看。

2.5.6 颜色

使用 xcolor 宏包来方便地调用颜色。比如本文中代码的蓝色：

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

2.6 引用与注释

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

2.6.1 标签和引用

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

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

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

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

其中 bookmarksopen 表示打开 pdf 时展开书签栏，bookmarknumbered 表示多级书签显示章节序号。colorlinks 表示超链接以彩色显示，默认为：linkcolor=red，anchorcolor=black，citecolor=green，urlcolor=magenta. 更多的内容请参考 hyperref 宏包帮助文档。

在加载了 hyperref 宏包后，可以使用的命令有：

```
% 文档内跳转
\hyperref[label-name]{print-text}
% 链接网站
\href{URL}{print-text}
\url{URL}
```

2.6.2 脚注

脚注是一种简单标注，使用方法是：

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

行文中切忌过多地使用脚注，它会分散读者的注意力。默认情况下脚注按章编号。脚注与正文的间距使用类似\setlength{\skip\footins}{0.5cm}的命令调节。

2.6.3 参考文献

参考文献主要使用的命令是\cite，与\label相似。通过 natbib 宏包的使用可以定制参考文献标号在文中的显示方式等格式，下面 natbib 宏包的选项含义为：

数字编号、排序且压缩、上标、外侧方括号，总体像这样：[1,3-5]。[5]

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

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

2.7 正式排版：封面、大纲与目录

2.7.1 封面

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

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

然后在 document 环境内第一行，写上：`\maketitle`，就能产生一个简易的封面。其中 title 和 author 是必须定义的，date 如果省略会自动以编译当天的日期为准，格式形如：January 1, 1970。如果你不想显示日期，可以写`\date{}`。

2.7.2 大纲与章节

LaTeX 中，将文档分为若干大纲级别。分别是：

\part: 部分。这个大纲不会打断 chapter 的编号。
\chapter: 章。基于 article 的文档类不含该大纲级别。
\section: 节。
\subsection: 次节。默认 report/book 文档类本级别及以下的大纲不进行编号，也不纳入目录。
\subsubsection: 小节。默认 article 文档类本级别及以下的大纲不进行编号，也不纳入目录。
\paragraph: 段。极少使用。
\subparagraph: 次段。极少使用。

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

对应的命令例如：`\section{第一节}`

以上各级别在 LaTeX 内部以“深度”参数作为标识。第一级别 part 的深度是 -1，以下级别深度分别是 $0, 1, \ldots$，类推。注意到由于 article 文档类缺少 chapter 大纲，因此 chapter 以下的深度在 book 和 report 文档类中会有所不同。

另外的一些使用技巧：

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

关于附录部分的大纲级别问题不在此讨论，请参考这一节。关于章节样式自定义的问题，则请看这里。

2.7.3 目录

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

目录的自定义需要借助 titletoc 宏包，参考这一节。

2.8 计数器与列表

2.8.1 计数器

LaTeX 中的自动编号都借助于内部的计数器来完成。包括：

章节 part, chapter, section, subsection, subsubsection, paragraph, subparagraph
编号列表 enumi, enumii, enumiii, enumiv
公式和图表 equation, figure, table
其他 page, footnote, mpfootnote[6]

用`\the`接上计数器名称的方式来调用计数器，比如`\thechapter`。如果只是想输出计数器的数值，可以指定数值的形式，如阿拉伯数字、大小写英文字母，或大小写罗马数字。常用的命令包括：

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

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

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

```
\renewcommand{\thechapter}{\Alph{chapter}}
\renewcommand{\thesection}
    {\thechapter-\arabic{section}}
```

2.8.2 列表

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

itemize 环境

例子：

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

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

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

enumerate 环境

例子：

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

1 - First
Foo Second
2 - Third

方括号的使用会打断编号，之后的编号顺次推移。更多的方法参考这一节。

description 环境

例子：

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

LaTeX Typesetting System.
wkl A Man.

默认的方括号内容会以加粗显示。更多的方法参考这一节。

2.9 浮动体与图表

2.9.1 浮动体

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

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

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

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

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

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

如果希望浮动体不要跨过 section，使用：

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

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

2.9.2 图片

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

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

其中可选参数指定了图片宽度为 0.8 倍该行文字宽。类似地可以指定 height(图片高)，scale(图片缩放倍数)，angle(图片逆时针旋转角度) 这样的命令。除 angle 外的命令不建议同时使用。

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

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

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

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

如果你希望行内图文混排，可参考 wrapfig 宏包：

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

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

2.9.3 表格

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

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

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

各参数的说明如下：

- 可选参数**对齐方式：**[t] 表示表格上端与所在行的网格线对齐。如果与它同一行的有文字的话，文字是与表格上端同高的。如果使用参数 [b]，就是下端同高。[c] 是中央同高。t=top, b=buttom, c=center.
- 必选参数**列格式：**用竖线符号“|”来表示竖直表线，连续两个“|”表示双竖直表线。最右边留空了，表示没有竖直表线。或者你可以使用“@{}”表示没有竖直表线。你也可以用“@{-}”这样的形式把竖直表线替换成“-”，具

体效果不再展示。而此处的 l、c、r 分别表示从左往右一共三列，分别**左对齐、居中对齐、右对齐**文字。在使用 l、c、r 时，表格宽度会自动调整。你可以用"p2em" 这样的命**指定某一列的宽度**，这时文字自动左对齐。注意：单元格中的文字默认向上水平表线对齐，即竖直居上。

- 在 tabular 环境内部，命令`\hline`来绘制水平表线。命令`\cline{i-j}`用于绘制横跨从 i 到 j 行的水平表线。两个连续的`\hline`命令可以画双线，但是双线之间相交时可能存在问题。
- 在 tabular 环境内部，命令"`&`" 用于把光标跳入该行下一列的单元格。每行的最后请使用两个反斜杠命令跳入下一行。命令`\hline`或`\cline`不能算作一行，因此它们后面没有附加换行命令。
- 在 tabular 环境内部，跨列命令`\multicolumn{number}{format}{text}`用于以 format 格式合并该行的 number 个单元格，并在合并后的单元格中写入文本 text。如果一行有了跨列命令，请注意相应地减少"`&`" 的数量。

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

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

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

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

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

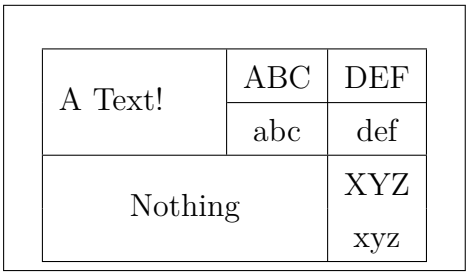
```
\newcolumntype{T}{>{$}c<{$}}
```

来一个 array 宏包的例子：

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

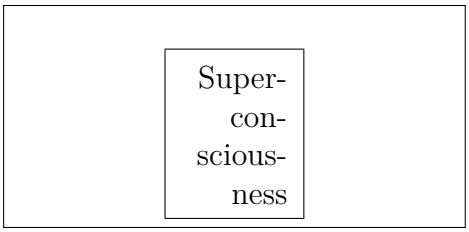
然后一个跨行跨列的例子。如果同时跨行跨列，必须把 multirow 命令放在 multicolumn 内部。有时候会用 multirow 和 multicolumn 作用于 1 行或 1 列，这可能是为了临时改变某个单元格的对齐方式。

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



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

```
% \usepackage{array}
\newcolumntype{P}[1]{>{#1\hspace{0pt}
  \arraybackslash}p{14mm}}
% 命令\arraybackslash用于自动换行
\begin{center}
\begin{tabular}{|P{\raggedleft}|}
  \hline Superconsciousness \\ \hline
\end{tabular}
\end{center}
```



一些其他的使用技巧：

1 - 快速输入多个格式相同的列：参数`|*{7}{c|}r|`，相当于 7 个居中列和 1 个居右列。

2 - 表格重音：原本的重音命令`\`, \'` 与`\=`，改为`\a`, \a'` 与`\a=`。

3 - 控制整个表格宽度：使用`\begin{tabular*}{width}[pos]{cols}`，比如你可以把 width 取值为`0.8\linewidth`之类。

2.9.4 浮动体外的图表标题

如果不使用浮动体，又想给图、表添加标题，请在导言区加上：

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

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

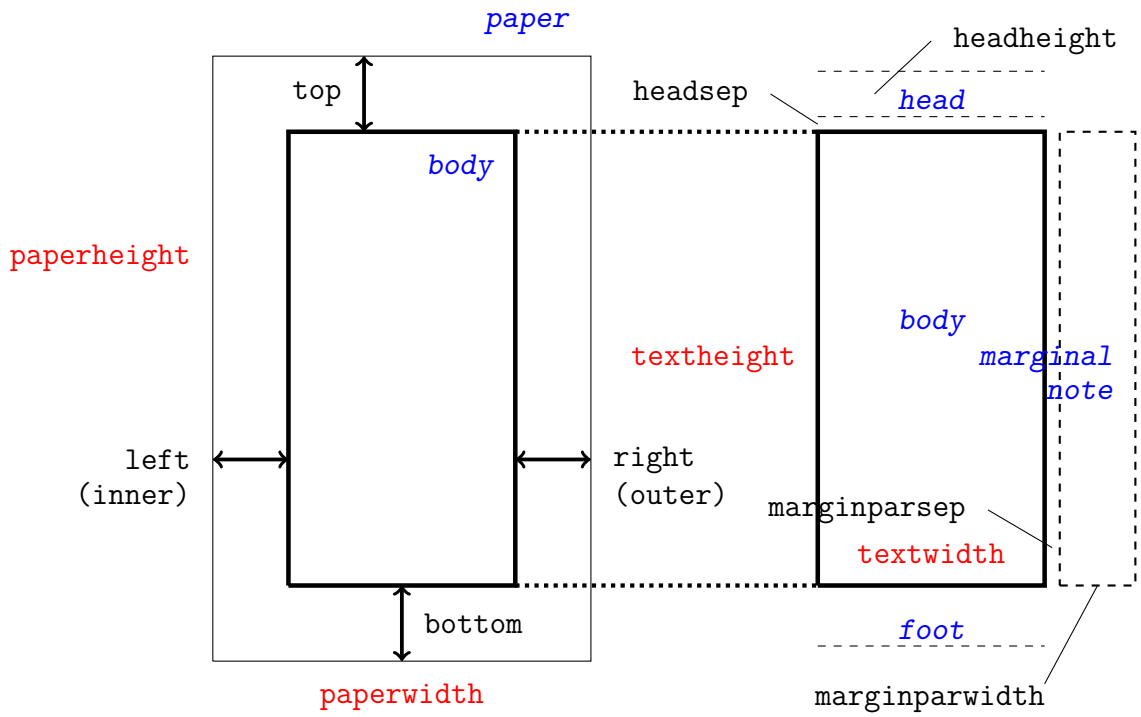


图 2.1: 页面构成示意图

2.10 页面设置

2.10.1 纸张、方向和边距

主要借助 geometry 宏包。先看一张页面构成，如图2.1：
geometry 宏包的具体的选项参数有：

- `paper=<papername>`：其中纸张尺寸有 [a0–a6, b0–b6, c0–c6]paper, ansi[a–e]paper, letterpaper, executivepaper, legalpaper.
- `papersize={<width>,<height>}`: 自定义尺寸。也可以单独对 paperwidth 或者 paperheigth 赋值。
- `landscape`: 切换到横向纸张。默认的是 portrait.

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

总文本区在默认状态下占纸张总尺寸的 0.7，由 scale=0.7 控制，你也可以分别用 `hscale` 和 `vscale` 指定宽和高的占比。用具体的长度定义也是可以的，使用 `(total)width` 和 `(total)height` 定义总文本区尺寸，或者用 `textwidth` 和 `textheight` 定义主文本区的尺寸[7]。或者直接用 `total={width,height}`，`body={width,height}` 定义。甚至你可以用 `lines=<num>` 行数指定 textheight。

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

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

页眉和页脚是位于 top 和 bottom 页边之内的文档元素。对于页眉和页脚的高度，分别使用 headheight/head, footskip/foot 参数指定。marginpar 用来指定侧页边的宽度。它们到主文本区的参数分别是 headsep, footnotesep, marginparsep. 你可以用 nohead, nofoot, nomarginpar 参数来清楚总文本区中的页眉，页脚和侧页边。

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

最后，这是一个小例子：

```
\usepackage[marginpar=3cm, includemp]{package}
```

2.10.2 页眉和页脚

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

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

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

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

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

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

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

fancyhdr 在定义双页文档时，采用了如下的默认设置：

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

上例中的\rightmark表示较低级别的信息，即当前页所在的 section，形式如“1.2 sectionname ”，对于 article 则是 subsection；而\leftmark表示较高级别的信息，即对应的 chapter，对于 article 则是 section. 命令\leftmark包含了页面上\markboth[8]下的最后一条命令的左参数，比如该页上出现了 section 1–2，那么 leftmark 就是“Section 2”；命令\rightmark则包含了页面上的第一个\markboth命令的右参数或者第一个\markright命令的唯一参数，比如可能是“Subsection 1.2”。

这听起来可能难以理解，但是 markboth 命令有两个参数，故有左右之分；而 markright 命令只有一个参数。你可以试着再去理解一下双页文档下的 fancyhdr 的默认设置。利用这一点来重定义 chaptermark, sectionmark, subsectionmark（仅其中两项）命令，举个例子:

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

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

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

2.11 抄录与代码环境

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

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

[8] \markboth是一个会被\chapter等命令调用的命令，默认右参数是空。注意，带星号的大纲不调用这一命令，你需要这样书写：\chapter*{This\markboth{This}{}}。

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

代码环境的输出，比如本文中带行号的代码块，参见这一节。

2.12 分栏

这部分内容使用`\documentclass[twocolumn]{...}`这个可选参数就可以实现。如果在同一页内需要分栏与单栏并存（比如摘要跨栏，正文分栏），或者想要分三栏，请参考 multicol 宏包。

2.13 文档拆分

文档拆分很简单，只需要把多个 tex 文件放在一起，然后再主文件中使用`\input{filename}`或者`\include{filename}`命令就可以了，带不带扩展名均可。两个命令区别在于`\include`命令将会很好地支持大纲编号。

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

2.14 西文排版及其他

2.14.1 连写

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

2.14.2 断词

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

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

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

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

I will show you this example: supercalifragilisticexpialidocious

如果你不想断词，比如电话号码，巧妙利用`\mbox`命令吧：

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

2.14.3 特殊符号

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

表 2.6: wasysym 宏包符号

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

2.14.4 其他

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

3 数学排版

3.1 行间与行内公式

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

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

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

另外一种公式排版方式是**行间公式**，也称行外公式，使用：

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

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

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

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

[1]还有一种 `$$...$$` 的写法，问题很多，绝不建议使用。

如果行间公式需要编号，使用 equation 环境[2]，还可以插入标签：

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

$$|\epsilon| > M \tag{3.1}$$

3.2 数学字体、字号与空格

3.2.1 空格

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

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

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

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

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

$$\int_0^1 x \ud{}x = \frac{1}{2}$$

其中`\ud` 命令是自定义的，这也是微分算子的正常定义：

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

3.2.2 数学字体

将字体转为正体使用`\mathrm`命令。如果需要保留行文中的空格，使用`\textrm`命令——这个与正文一致。但是，textrm 命令内的字号可能不会自适应，mathrm 则稳定得多。

例如自然对数的底数 e，在本文中就是这样定义的：

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

以下简单介绍几种数学字体。数学字体的总表参见表3.1。

数学粗体

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

[2]需要注意有一个已经淘汰的多行公式编号环境叫 `eqnarray`，请不要再使用。

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

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

空心粗体

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

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

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

3.3 基本命令

基本函数默认用正体书写，包括：

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

此外有一个叫\mathop的命令，可以把任何的数学字符串转换为数学对象，使其适用上标、下标等数学对象才能使用的命令。

3.3.1 上下标与虚位

用低划线和尖角符表示上标和下标，请仔细体会下述例子：

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

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

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

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

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

3.3.2 微分与积分

导数直接使用单引号'，积分使用\int 符号：

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

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

有时候需要更高级的微分或积分号，其中\ud命令在上文这里定义过：

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

效果：

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

其中的\dot 系的导数形式 LaTeX 只原生支持到二阶导数。后面的三阶、四阶需要 amsmath 宏包。\int 系的积分命令也是一样。而环形双重积分命令\varoiint需要 esint 宏包[3]。

\left. 或\right. 命令[4]只用于匹配，本身不输出任何内容。

3.3.3 分式、根式与堆叠

分式使用\frac 命令。或者 amsmath 宏包支持的\dfrac命令来强制获得行间公式大小的分数，\tfrac则强制获得行内公式大小的分数：

```
  $\sqrt{\frac{x}{y}}$ \\
  $x^{1/2} \qquad x^{1\frac{2}{3}}$
```

$$\sqrt{\frac{x}{y}}$$
$$x^{1/2} \qquad x^{1\frac{2}{3}}$$

根式：

```
  $\sqrt{2} \quad \surd$ \\
  $\sqrt[3]{\sqrt{2}+1}$
```

$$\sqrt{2} \quad \surd$$
$$\sqrt[3]{\sqrt{2}+1}$$

划线命令使用\underline 和\overline，水平括号使用 brace：

```
$\overline{m+n}$ \\
$\underbrace{a_1+\ldots+a_n}_{n}$
$\overbrace{a_1+\ldots+a_n}^{n}$
```

$$\overline{m+n}$$
$$\underbrace{a_1+\ldots+a_n}_{n} \quad \overbrace{a_1+\ldots+a_n}^{n}$$

事实上\overline 命令也存在问题，请比较：

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

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

其中 closure 是在导言区定义的

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

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

向量符号：

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

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

尖帽符号或者波浪符号：

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

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

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

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

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

还有一个很强大的堆叠放置命令\sideset：

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

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

去心邻域大概也属于堆叠符的一种？这样输出：

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

$\mathring{U}$

在下一次节：累加与累积中，还介绍了更多的堆叠命令。

3.3.4 累加与累积

使用\sum 和\prod 命令，效果如下：

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

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

有时需要复杂的堆叠方式，效果如下：

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

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

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

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

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

3.3.5 矩阵与省略号

矩阵的排版可以通过 array 环境和自适应定界符完成：

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

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

还有就是\cdots 命令。hmath 宏包还提供了一种省略号\adots：⋰，或许用得上呢？

LaTeX 原生支持自动添加定界符的形式，不过要放在数学环境内：

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

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

方括号和花括号使用\bmatrix 命令：

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

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

行列式使用\vmatrix 命令：

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

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

array 环境还有更多的用途，也可以像表格一样划线：

```
\[\begin{array}{c|c}
  a_{11} & a_{12} \\
  \hline
  a_{21} & a_{22}
  \end{array}\]
```

$$\begin{array}{c|c} a_{11} & a_{12} \\ \hline a_{21} & a_{22} \end{array}$$

3.3.6 分段函数与联立方程

array 也可以用来写分段函数：

```
\[y=\left\{\begin{array}{ll}
x+1, & x>0 \\
0,   & x=0 \\
x-1, & x<0
\end{array}\right.\]
```

$$y=\left\{\begin{array}{ll} x+1, & x>0 \\ 0, & x=0 \\ x-1, & x<0 \end{array}\right.$$

分段函数还能用 cases 环境，它自动生成一个花括号，也更紧凑：

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

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

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

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

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

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

3.3.7 多行公式及其编号

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

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

$$\begin{aligned} a^2 &= a\cdot a && (3.2)\\ &= a*a && (3.3)\\ &= a^2 && (3.4)\end{aligned}$$

LaTeX 中长公式不能自动换行[5]，请自行指定断行位置和缩进距离，就像上面一样。也许你不需要三个编号，你可以：

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

$$\begin{aligned} a^2 &= a\cdot a & b &= c\\ g &= a*a & d &> e > f\\ step &= a^2 & Z^3 && (3.5)\end{aligned}$$

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

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

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

If

$$y = 0$$
$$x < 0$$

then

$$z = x + y$$

当然，align 环境是用来分列对齐的。如果你仅仅想要所有行居中，使用 amsmath 宏包的 gather 环境即可。这是一个非常实用的环境，你也可以用 gather* 环境排版居中的、非编号的多行公式。

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

$$\begin{aligned} X &= 1+2+\cdots+n && (3.6)\\ Y &= 1 && (3.7)\end{aligned}$$

3.3.8 二项式与定理

二项式可能需要借助 amsmath 宏包的`\binom` 命令：

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

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

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

在使用这部分内容时，请加载 amsthm 宏包。

首先是定理环境格式的自定义。如同定义命令一样，在导言区加上：

\newtheorem{*name*}[*counter*]{*text*}[*section*]

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

来看一个例子。首先在导言区定义如下三个样式：

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

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

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

Law 3.3.1. Never believe easily.

Jury 3.3.2 (The 2nd)**.** *Never suspect too much.*

Margaret. Nothing else.

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

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

Proof. For an right triangle, we have:

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

3.4 数学符号与字体

3.4.1 数学字体

原生的数学字体命令：

表 3.1: 原生数学字体表

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

需要其他宏包支持的数学字体：

表 3.2: 宏包数学字体表

`\mathscr{ABCDabcde 1234}`	mathrsfs
\{$ABCDabcde1234$\}	
`\mathfrak{ABCDabcde 1234}`	amsfonts 或者 amssymb
\{$ABCDabcde1234$\}	
`\mathbb{ABCDabcde 1234}`	amsfonts 或者 amssymb
\{$ABCDabcde1234$\}	

3.4.2 定界符

表3.3给出了一些数学环境中使用的定界符。

表 3.3: 定界符

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

– 以下需要 amssymb 宏包 –

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

使用`\left` 或者`\right` 能够使定界符自适应式子的高度：

```
\[\left\{x+\frac{\left[\left|\frac
{y}{x}\right|^3-Z^{5\left(\frac{x}
{y}+z\right)}\right]}{y}\right\}\]
```

$$\left\{x+\frac{\left[\left|\frac{y}{x}\right|^3-Z^{5\left(\frac{x}{y}+z\right)}\right]}{y}\right\}$$

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

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

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

其中\vphantom 命令用于输出一个高度虚位，使得第二行的自适应定界符与第一行同等大小。

有时你可能希望手动指定定界符的尺寸，这时使用：

```
  $(\big(\Big(\bigg(\Bigg($ \\
  \[\bigl[\frac{x+y}{x^2}\bigr]\]
```

$$\big(\Big(\bigg(\Bigg($$
$$\Big[\frac{x+y}{x^2}\Big]$$

3.4.3 希腊字母

希腊字母表如表3.4所示。表中包含了小写希腊字母、大写希腊字母，其中部分希腊字母的输入方式与英文字母一致。

表 3.4: 希腊字母表

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

3.4.4 二元运算符

二元运算符包括常见的加减乘除，还有集合的交、并、补等运算。表3.5只列出常用的二元运算符，更多的请参考 symbols-a4 文档。

表 3.5: 二元运算符

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

3.4.5 二元关系符

二元关系符常常被用于判断两个数的大小关系，或者集合中的从属关系。表3.6和表3.7只列出常用的二元关系符，更多的请参考 symbols-a4 文档。

表 3.6: 二元关系符

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

表3.7中的二元关系符需要 amssymb 宏包。

表 3.7: amssymb 二元关系符

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

3.4.6 箭头

在表3.3中给出了几个箭头符号，但是不够全，这里给出总表如表3.8。

表 3.8: 箭头

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

依旧给出一个基于 amssymb 宏包的附表??。

表 3.9: amssymb 箭头

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

3.4.7 其他符号

最后是一些其他的难以归类的符号，也不全是数学领域会用到的，只不过它们可以在数学环境下输出出来，以及被 amssymb 宏包所支持。如表3.10和表3.11。

表 3.10: 其他符号

…	\dots	⋯	\cdots	⋮	\vdots	⋱	\ddots
∀	\forall	∃	\exists	ℜ	\Re	ℵ	\aleph
∠	\angle	∞	\infty	△	\triangle	∇	\nabla
ℏ	\hbar	𝜄	\imath	𝚥	\jmath	ℓ	\ell
♠	\spadesuit	♡	\heartsuit	♣	\clubsuit	♢	\diamondsuit
♭	\flat	♮	\natural	♯	\sharp		\

非数学符号：

£	\pounds	§	\S	©	\copyright	¶	\P
†	\dag	‡	\ddag	®	textregistered		

表 3.11: amssymb 其他符号

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

LaTeX 进阶 | 4

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

4.1 自定义命令与环境

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

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

现在来解释一下各个参数：

cmd: 新定义的命令，不能与现有命令重名。
args: 参数个数。
default: 首个参数，即 #1 的默认值。
def: 具体的定义内容。参数 1 以 #1 代替，参数 2 以 #2 代替，以此类推。

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

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

4.2 箱子：排版的基础

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

在 LaTeX 编译中，坏箱将记录在日志文件里。即使编译过程中产生了坏箱，编译也仍然会进行，并将坏箱强行输出到最终的文档。这就可能产生很多奇怪的情况，比如文字写到了页面之外。如果你是完美主义者，请你逐个排查坏箱；即使你不是，也请排查看起来非常不美观的那些坏箱。

4.2.1 无框箱子

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

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

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

无框小页的使用方法是 minipage 环境，参数类似\parbox：

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

4.2.2 加框箱子

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

```
\fbox{This is a frame box} \\
\begin{equation}\boxed{x^2=4}
\end{equation}
```

This is a frame box

$$\boxed{x^2 = 4} \tag{4.1}$$

加宽盒子的宽度、以及内容到盒子的距离可以自行定义。默认定义是：

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

加框小页使用 boxedminipage 环境，类似 minipage 环境。

4.2.3 竖直升降的箱子

命令\raisebox可以把文字提升或降低，它有两个参数：

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

4.2.4 段落箱子

段落箱子的强大之处在于它提供自动换行的功能，当然你需要指定宽度给它。

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

以及例子：

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

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

4.2.5 缩放箱子

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

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

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

4.2.6 标尺箱子

命令\rule[lift]{width}{height}能够画出一个黑色的矩形。你可以在单元格中使用 width, height 其一为 0 的该命令，作一个隐形的“支撑”来限定单元格的宽或高。例如：

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

text
2nd text

4.3 复杂距离

4.3.1 水平和竖直距离

长度单位参考这里介绍过的内容。水平距离命令有两种，一种禁止在此处断行，包括表4.1：

表 4.1: 禁止换行的水平距离

\thinspace 或\,	0.1667em	_ _
\negthinspace	-0.1667em	__
\enspace	0.5em	_ _
\nobreakspace 或 ~	空格	_ _

另一种允许换行，如表4.2：

表 4.2: 允许换行的水平距离

\quad	1em	_ _
\qquad	2em	_ _
\enskip	0.5em	_ _
\ (此处是一个空格)	空格	_ _

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

类似地使用\vspace和\vspace*命令，作为竖直距离的输出。

4.3.2 填充距离与弹性距离

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

弹性距离指以一定比例计算得到的多个空白，命令是\stretch。例子：

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

4.3.3 制表位与行距 *

制表位使用 tabbing 环境，需要指出，这是一个极其容易造成坏箱的环境。一个丑陋的例子：

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

```
也可以 \= 随时重设 \= 制表位 \\
这是 \> 新的 \> 一行
\end{tabbing}
```

效果：

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

几个要点：

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

4.3.4 悬挂缩进 *

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

如果需要对单独一段进行悬挂缩进，例如使用：

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

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

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

```
正文...

{\leftskip=3em\parindent=-1em
\indent 这是第一段。注意整体需要放在一组花括号内，且花括号前应当有空白行。第一段前需要加indent命令，最后一段的末尾需额外空一行，否则可能出现异常。

这是第二段。

\ldots

这是最后一段。别忘了空行。

}
```

正文...
　　这是第一段。注意整体需要放在一组花括号内，且花括号前应当有空白行。第一段前需要加 indent 命令，最后一段的末尾需额外空一行，否则可能出现异常。
　　这是第二段。
　　...
　　这是最后一段。别忘了空行。

[1]事实上这是一个排版诗歌的环境。

4.3.5 整段缩进 *

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

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

对于奇偶页处理，可以尝试赋值\leftskip等命令。

4.4 自定义章节样式

4.5 自定义目录样式

4.6 自定义图表

4.7 自定义编号列表

4.8 BibTeX 参考文献

4.9 附录、图表目录

4.10 编程代码输出环境

5 Tikz 绘图 *

其实 LaTeX 本身是有绘图功能的，可以绘制线段、斜率为整数的直线等基础功能。显然这是不能满足绘图需求的。因此在本手册中，对于 LaTeX 原生的绘图功能不再多做介绍，有兴趣的可以自行了解。本手册主要针对 Tikz 绘图。[1]

PGFPlots/Tikz 是两个 LaTeX 的绘图宏包，功能极其强大。其基本的结构为：

```
\documentclass{doc-class}
\usepackage{tikz}
\usetikzlibrary{...}
\begin{document}
  \begin{tikzpicture}
    ...
  \end{tikzpicture}
\end{document}
```

废话不多说，进入正题。

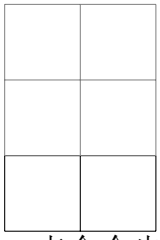
5.1 简单的实例

5.1.1 直线、网格与点

```
\begin{tikzpicture}
  \draw (0,0) -- (1,2);
\end{tikzpicture}
```

注意到 help lines 这个参数使网格绘制更有辅助线的效果，颜色更淡：

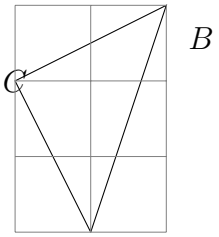
[1] 在 LaTeX 中其实还可以使用 PSTricks 宏包，是非常强力。XYpic 宏包则有时用来画小图。



```
\begin{tikzpicture}
  \draw (0,0) grid (2,1);
  \draw [help lines](0,1) grid (2,3);
\end{tikzpicture}
```

点命令也非常好理解。其中\coordinate 是真正将点的名字和坐标联系起来的命令（相当于“赋值”），而\node 命令仅是一个标注文本的作用。

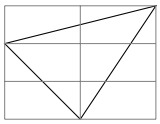
注意到第二行\node 命令，它使得字母 B 标注在 pB 点 315 度的位置。这样的效果往往优于其上一行的\node 命令的效果。注意：不要忘记\node 命令后面的花括号（即使它是空的）！




```
\begin{tikzpicture}
  \coordinate (pA) at (1,0);
  \coordinate (pB) at (2,3);
  \coordinate (pC) at (0,2);
  \node at (pC) {$C$};
  \node[label=315:$B$] at (pB){};
  \draw (pA) -- (pB) -- (pC) -- (pA);
  \draw [help lines](0,0) grid (2,3);
\end{tikzpicture}
```

5.1.2 拉伸

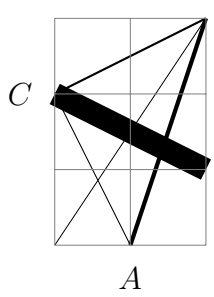
拉伸只需要在 tikzpicture 后添加 xscale/yscale/scale 的可选参数赋值即可。例如：



```
\begin{tikzpicture}[yscale=0.5]
  \coordinate (pA) at (1,0);
  \coordinate (pB) at (2,3);
  \coordinate (pC) at (0,2);
  \draw (pA) -- (pB) -- (pC) -- (pA);
  \draw [help lines](0,0) grid (2,3);
\end{tikzpicture}
```

5.1.3 线宽

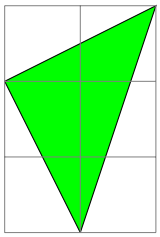
线宽可以在 tikzpicture 后使用 “[line width=5pt]” 之类的参数进行全局调整，也可以在每个 draw 指令下分别地进行调整：




```
\begin{tikzpicture}
  \coordinate (pA) at (1,0);
  \coordinate (pB) at (2,3);
  \coordinate (pC) at (0,2);
  \node[label=270:$A$] at (pA){};
  \node[label=0:$B$] at (pB){};
  \node[label=180:$C$] at (pC){};
  \draw[ultra thick] (pA) -- (pB);
  \draw[thick] (pB)-- (pC);
  \draw[thin] (pC)-- (pA);
  \draw[ultra thin] (pB) -- (0,0);
  \draw[line width=0.3cm] (pC) -- (2,1);
  \draw [help lines](0,0) grid (2,3);
\end{tikzpicture}
```

5.1.4 填色

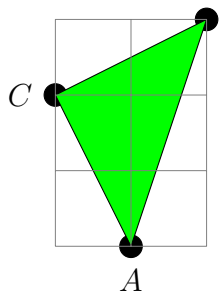
填色的逻辑非常简单，是指向绘制的**闭合**图形内部填色。本例中，draw 命令首尾相接地连接三个点，构成一个闭合图形——三角形，因此可以填色：



```
\begin{tikzpicture}
  \coordinate (pA) at (1,0);
  \coordinate (pB) at (2,3);
  \coordinate (pC) at (0,2);
  \draw[fill=green] (pA) -- (pB) -- (pC) --
     (pA);
  \draw[help lines] (0,0) grid (2,3);
\end{tikzpicture}
```

5.1.5 点样式

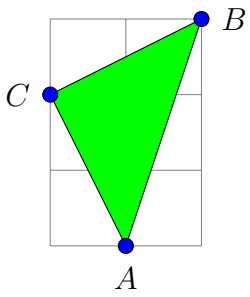
填色之后图形好看了许多，但是还不够。点标在图中似乎也不看清位置，不过这是有方法解决的：




```
\begin{tikzpicture}
  \coordinate (pA) at (1,0);
  \coordinate (pB) at (2,3);
  \coordinate (pC) at (0,2);
  \node[label=270:$A$,circle,draw,fill,inner
      sep=3pt] at (pA){};
  \node[label=0:$B$,circle,draw,fill,inner
     sep=3pt] at (pB){};
  \node[label=180:$C$,circle,draw,fill,inner
      sep=3pt] at (pC){};
  \draw[fill=green] (pA) -- (pB) -- (pC) --
     (pA);
  \draw[help lines] (0,0) grid (2,3);
\end{tikzpicture}
```

但是这一长串简直是太复杂了。大概能看懂 circle 是将点画成圆形，fill 表示填充，inner sep 表示点的大小。而 Tikz 给出的`\tikzstyle` 能够简化步骤。

还有一点就是，绿色的填充似乎位于点的上方？还是乖乖地调整语句顺序吧。

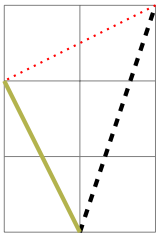



```
\begin{tikzpicture}
  \draw[help lines] (0,0) grid (2,3);
  \coordinate (pA) at (1,0);
  \coordinate (pB) at (2,3);
  \coordinate (pC) at (0,2);
  \draw[fill=green] (pA) -- (pB) -- (pC) --
     (pA);
  \tikzstyle{every node} = [circle,draw,fill
     =blue,inner sep=2pt];
  \node[label=270:$A$] at (pA){};
  \node[label=0:$B$] at (pB){};
  \node[label=180:$C$] at (pC){};
\end{tikzpicture}
```

注意：`\tikzstyle{every node}` 控制的是其下方代码的所有 node 的点样式，不影响其上方代码中的点。

5.1.6 线型和颜色

除了点的定义，线当然也可以定义。线型有 dashed/dotted 两种，颜色除了默认的也可以通过叹号的形式控制。

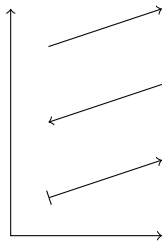


```
\begin{tikzpicture}
  \draw[help lines] (0,0) grid (2,3);
  \coordinate (pA) at (1,0);
  \coordinate (pB) at (2,3);
  \coordinate (pC) at (0,2);
  \draw[dashed, ultra thick] (pA) -- (pB);
  \draw[dotted, red, thick] (pB) -- (pC);
  \draw[blue!30!yellow, ultra thick] (pC) --
     (pA);
\end{tikzpicture}
```

主要的颜色包括[2]：red , green , yellow , blue , cyan , magenta , black , gray , darkgray , lightgray , brown , lime , olive , orange , pink , purple , teal , violet , 还有 white .

5.1.7 箭头

箭头也是需要经常绘制的内容。主要也是根据`\draw` 命令的参数控制：



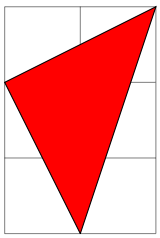
```
\begin{tikzpicture}
  \draw[->] (0.5,2.5) -- (2,3);
  \draw[<-] (0.5,1.5) -- (2,2);
  \draw[|->] (0.5,0.5)-- (2,1);
  \draw[<->] (0,3) -- (0,0) -- (2,0);
\end{tikzpicture}
```

[2]以下色块用`\tikz{\draw[`*color*`,line width=9] (0,0) -- (0.5,0);}` 绘制得到。

5.2 高效书写

5.2.1 变量

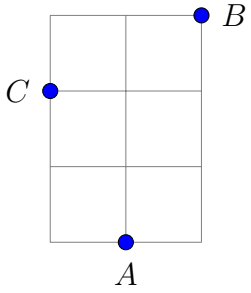
变量申请使用与 LaTeX 相同的语句：\newcommand. 注意：变量需要以反斜杠开头，并与现有命令不重复。



```
\begin{tikzpicture}
  \draw [help lines](0,0) grid (2,3);
  \newcommand{\aaa}{1};
  \newcommand{\bbb}{3};
  \newcommand{\ccc}{2};
  \coordinate (pA) at (\aaa,0);
  \coordinate (pB) at (\ccc,\bbb);
  \coordinate (pC) at (0,\ccc);
  \draw[fill=red] (pA) -- (pB) -- (pC) -- (
     pA);
\end{tikzpicture}
```

5.2.2 循环

但是同样的语句书写很多遍是很复杂的，好在 Tikz 提供了循环的实现方法：



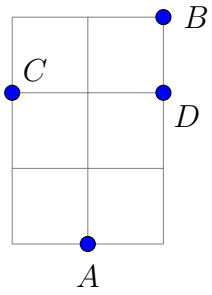

```
\begin{tikzpicture}
  \newcommand{\la}{1};
  \newcommand{\lb}{3};
  \newcommand{\lc}{2};
  \draw [help lines](0,0) grid (\lc,\lb);
  \coordinate (pA) at (\la,0);
  \coordinate (pB) at (\lc,\lb);
  \coordinate (pC) at (0,\lc);
  \tikzstyle{every node}=[circle, draw, fill
     =blue,inner sep=2pt];
  \foreach \x/\y in {A/270,B/0,C/180}{
    \node[label=\y:$\x$] at (p\x){};
  }
\end{tikzpicture}
```

注意到：甚至点的名称 pA, pB, pC 中的 A, B, C 也是可以通过 foreach 来指定的！

5.2.3 运算

注意：需要在导言区添加\usetikzlibrary{calc}。文中不再写出。

如果一个绘图工具仅仅能够绘图而不能做运算……它有什么用呢？如果作为科学排版系统下的绘图工具而不支持运算的话，Tikz 岂不是太弱了？




```
\begin{tikzpicture}
  \draw [help lines](0,0) grid (2,3);
  \coordinate (pA) at (1,0);
  \coordinate (pB) at (2,3);
  \coordinate (pC) at (0,2);
  \coordinate (pD) at ($(pB)+(0,-1)$);
  \tikzstyle{every node}=[circle, draw, fill
     =blue,inner sep=2pt];
  \foreach \x/\y in {A/270,B/0,C/45,D/315}{
    \node[label=\y:$\x$] at (p\x){};
  }
\end{tikzpicture}
```

附录 A

注音符号

表 A.1: 注音符号与特殊符号

样式 - 命令	样式 - 命令	样式 - 命令	样式 - 命令
ō - \=o	ó - \'o	ǒ - \v o	ò - \`o
ô - \^o	ö - \"o	ȯ - \.o	ő - \H o
ọ - \d o	ŏ - \u o	o̲ - \b o	o͡o - \t oo
õ — $\tilde{o}$		ô — $\hat{o}$	
-	-	-	-
ø - \o	Ø - \O	ɪ - \i	ȷ - \j
å - \aa	Å - \AA	æ - \ae	Æ - \AE
œ - \oe	Œ - \OE	¡ - !`	¿ - ?`

表 A.2: 国际音标输入表（部分）

样式 - 命令	样式 - 命令	样式 - 命令
ʣ - \textdzlig	ʃ - \textesh	ʧ - \textteshlig
ʤ - \textdyoghlig	ʌ - \textturnv	ə - \textschwa
ɡ - \textscriptg	θ - \texttheta	ʊ - \textupsilon
ɑ - \textscripta	ð - \dh	ɛ - \textepsilon
ɔ - \textopeno	ʒ - \textyogh	ŋ - \ng
重音	次重音	长音节
ˈ - \textprimstress	ˌ - \textsecstress	ː - \textlengthmark

注：*\dh* 命令在非 *CJK* 文档中有时编译会出现问题。