适用于 Mac OS 系统的第一份 LaTeX 学习文档

张好 / Hao ZHANG[1]

Saturday 20th January, 2018

[1]haozhang@me.com

Contents

1 LaTeX 软件的安装和使用

1.1 从 LaTeX 的官网安装适用于 Mac OS 的 MacTeX

LaTeX 是基于 ***LaTeX Project Public License (LPPL)*** 条款下的免费软件（**Figure 1**）。LaTeX 本身并不是一个独立的排版程序，而是基于 *Donald E. Knuth* 的 TeX 排版系统所运行的文档准备软件。TeX 发行版通常将运行 TeX 系统所需的所有部件捆绑在一起，并且通常会添加这两者的配置和维护程序。如今，LaTeX 及其上的许多软件包构成了任何 TeX 发行版的重要组成部分。

Figure 1: 通过 The LaTeXProject 官网下载 LaTeX 软件

- **TeX 发行版的安装**

  对于一个 TeX 和 LaTeX 的新手或者只是想简单安装 LaTeX 的用户而言，可以直接安装完整的 TeX 发行版。TeX 用户组（***TUG***）有一个值得注意的分布清单，可以自由选择完全安装或只安装最重要的部分。

- **四个 LaTeX 运行平台**

  不同平台的 LaTeX 发行版软件，可以分别适用于 Linux、Mac OS、Windows 以及 Online 这四种操作系统或平台（**Figure 2**）。包括支持 Linux 系统的 TeX Live；支持 Mac OS 系统的 MacTeX；支持 Windows 系统的 MiKTeX、proTeXt 或者 TeX Live 以及支持 Online 编辑的 Papeeria、Overleaf、ShareLaTeX、Datazar 和 LaTeX base。

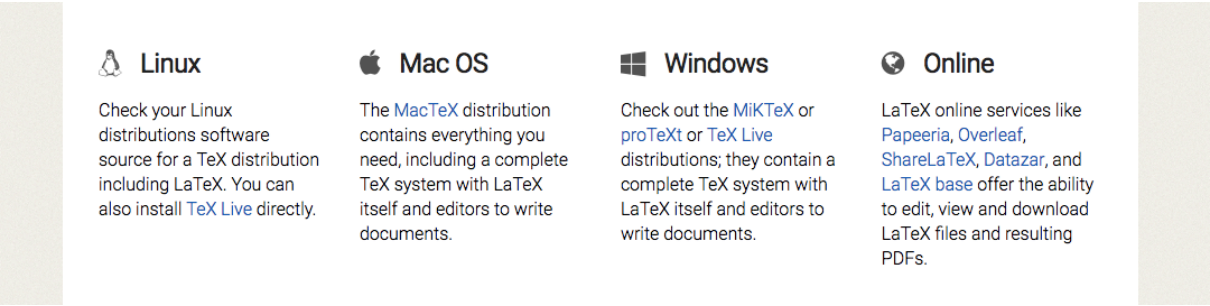

Figure 2: 支持四种不同平台的 LaTeX 发行版软件

1.2 安装适用于 Mac OS 系统的 Texmaker 和/或 TeXstudio

Texmaker 是一个自由且免费的 LaTeX 编辑器，支持 unicode，拼写检查，自动完成和代码折叠。它集成了一个集成的 pdf 显示屏，支持 synctex 并以连续模式显示。TeXstudio 是一个用于创建 LaTeX 文档的集成书写环境，与 Texmaker 相似，TeXstudio 也是一个自由且免费的 LaTeX 编

辑器。对于这两个 LaTeX 编辑器的选择，完全可以由用户的使用体验和使用习惯而定，两者各有优劣。**以下若无特殊说明，均以 Texmaker 为例。**

无论是使用 Texmaker 还是使用 TeXstudio，为了迎合中文的排版，通常所推荐的排版工具主要为 XeLaTeX。需要注意的是，TeX 和 pdfTeX 这两个排版工具是不直接支持 Unicode 字符的引擎，而 XeTeX 和 XeLaTeX 这两个引擎可以直接支持 Unicode 字符，因此对于中文排版的效果更为优秀。因此为了方便使用，可以将快速构建命令设置为“**XeLaTeX + 查看 PDF**”（**Figure 3**）。

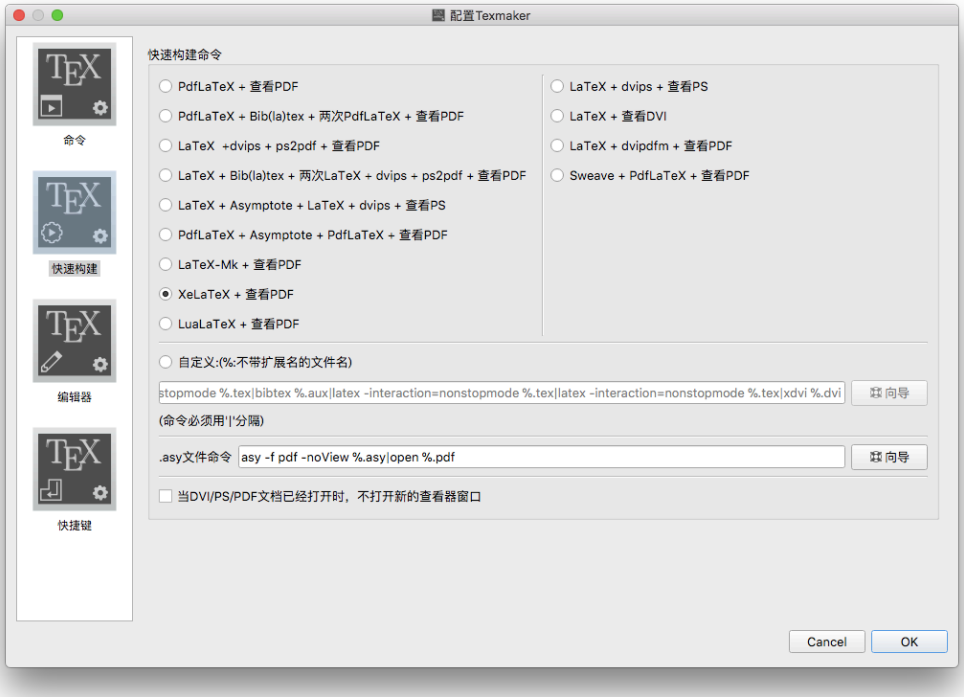

Figure 3: Texmaker 快速构建框中快速构建命令的设置

除外，为了方便观察 LaTeX 编辑器的编辑效果，可以将命令中的 PDF 的查看器设置为“内置常看器 + 嵌入”（**Figure 4**）。

2 了解 LaTeX 的第一步

2.1 编辑第一个 LaTeX 文档

1. 打开 Texmaker，检查编辑器框中编辑器字体编码是否为 UTF-8。若非 UTF-8，需先将其设置为 UTF-8 （**Figure 5**）。
2. 建立 一个新文档，将以下代码 复制 进入文档中， 保存 。

```
\documentclass{article}
\begin{document}
hello , world
\end{document}
```

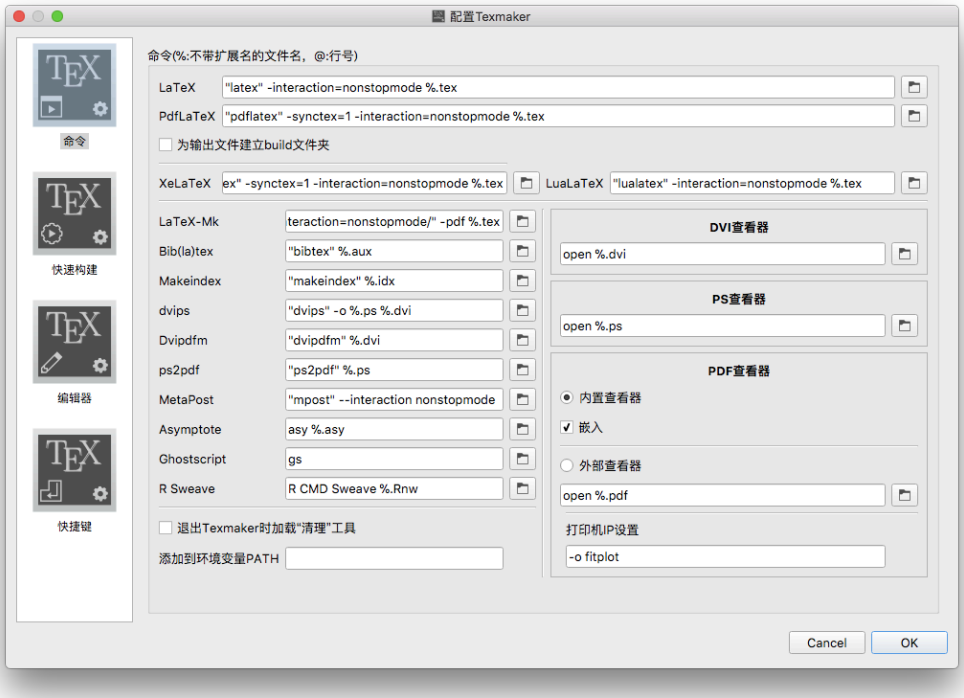

Figure 4: Texmaker 命令框中 PDF 内置查看器的设置

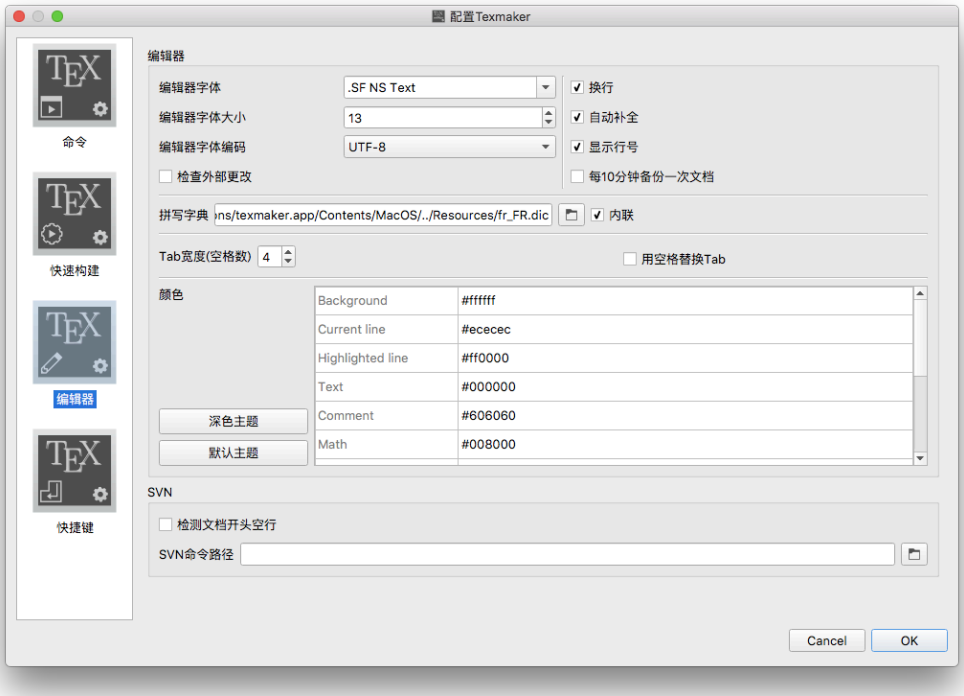

Figure 5: Texmaker 编辑器框中编辑器字体编码的设置

3. 运行 工具栏中的 快速构建 选项，观察编辑结果（**Figure 6**）。

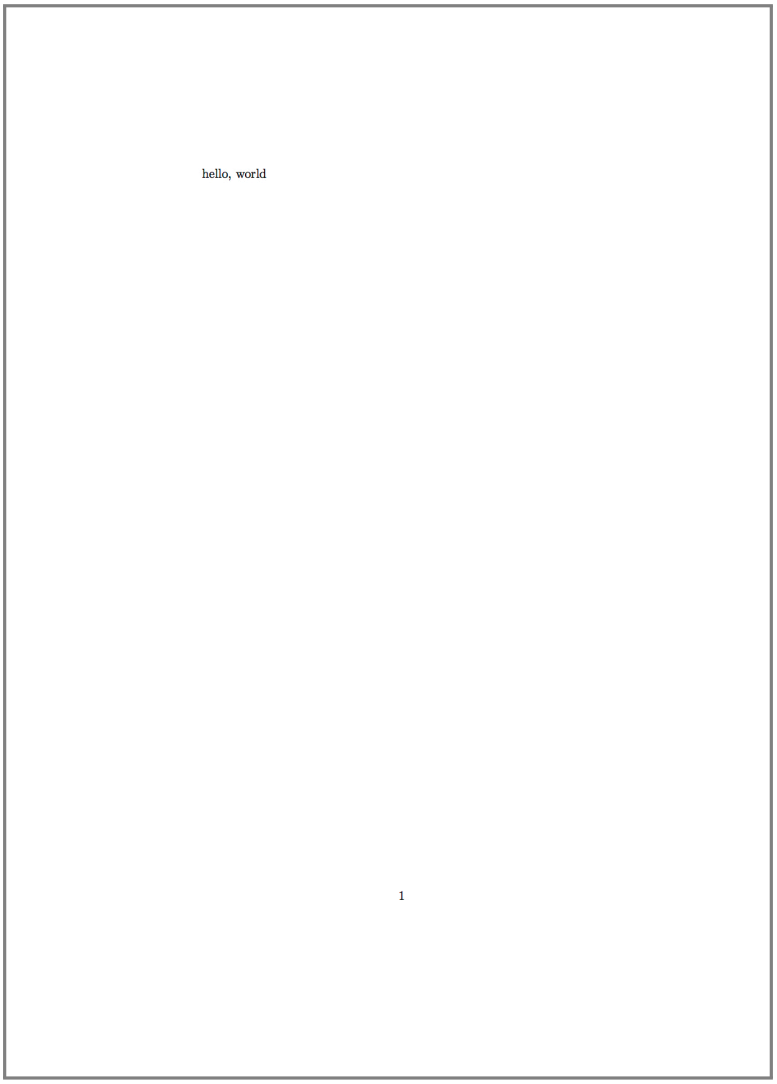

Figure 6: 第一个 LaTeX 文档的编辑效果

2.2 解密第一个 LaTeX 文档

2.2.1 LaTeX 的命令与环境

- **LaTeX 命令以反斜线 \ 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）作为分隔符。

  - 反斜线和后面的一个非字母符号，如 \$。它们无需分隔符。

- **LaTeX 的命令和参数：**

  - 大多数的 LaTeX 命令是带一个或多个参数，每个参数用花括号 { 和 } 包裹。

  - 除此之外，LaTeX 的花括号本身也起到分组的作用，可以将字体、格式等的更改限制在大括号范围之内。

- 有些命令带一个或多个可选参数，以方括号 [和] 包裹。
- 还有些命令在命令名称后可以带一个星号 * ，带星号和不带星号的命令效果有一定差异。

• **LaTeX 还引入了环境的用法，用以令一些效果在局部生效，或是生成特定的文档元素。LaTeX 环境的用法为一对命令 \begin 和 \end：**

```
\begin{<environment name>}{<arguments>}
…
\end{<environment name>}
```

- <environment name> 为环境名，\begin 和 \end 中填写的环境名应当一致。
- \begin 在 <environment name> 后可以带一个或多个参数，甚至可选参数。
- 环境允许嵌套使用。

2.2.2 LaTeX 源代码结构

1. LaTeX 源代码以一个 \documentclass 命令作为开头，它规定了文档使用的文档类：

```
\documentclass{ article / report / book / slides / beamer / lettre / memoir / seminar / …}
```

2. 接着 \documentclass 命令之后，可以用 \usepackage 命令调用宏包：

```
\usepackage{ geometry / setspace / hyperref / xeCJK / enumitem / listings / tcolorbox / …}
```

3. 最后，需要用以下一对命令来标记正文内容的开始位置和结束位置，从而将正文内容写入其中：

```
\begin{document}
…
\end{document}
```

注意：在 \documentclass 和 \begin{document} 之间的位置称为**导言区**，除了使用 \usepackage 调用宏包之外，一些对文档的全局设置命令也在这里使用。当然也可以视自己的需求，什么都不写，一个宏包都不调用。

3 了解 LaTeX 的第二步

3.1 给文档添加标题、作者和注释

1. 建立 一个新文档，将以下代码 复制 进入文档中， 保存 。

```
\documentclass{article}
\author{My Name}
\title{The Title}
\begin{document}
\maketitle
hello, world % This is comment
\end{document}
```

2. 运行 工具栏中的 快速构建 选项，观察编辑结果（**Figure 7**）。

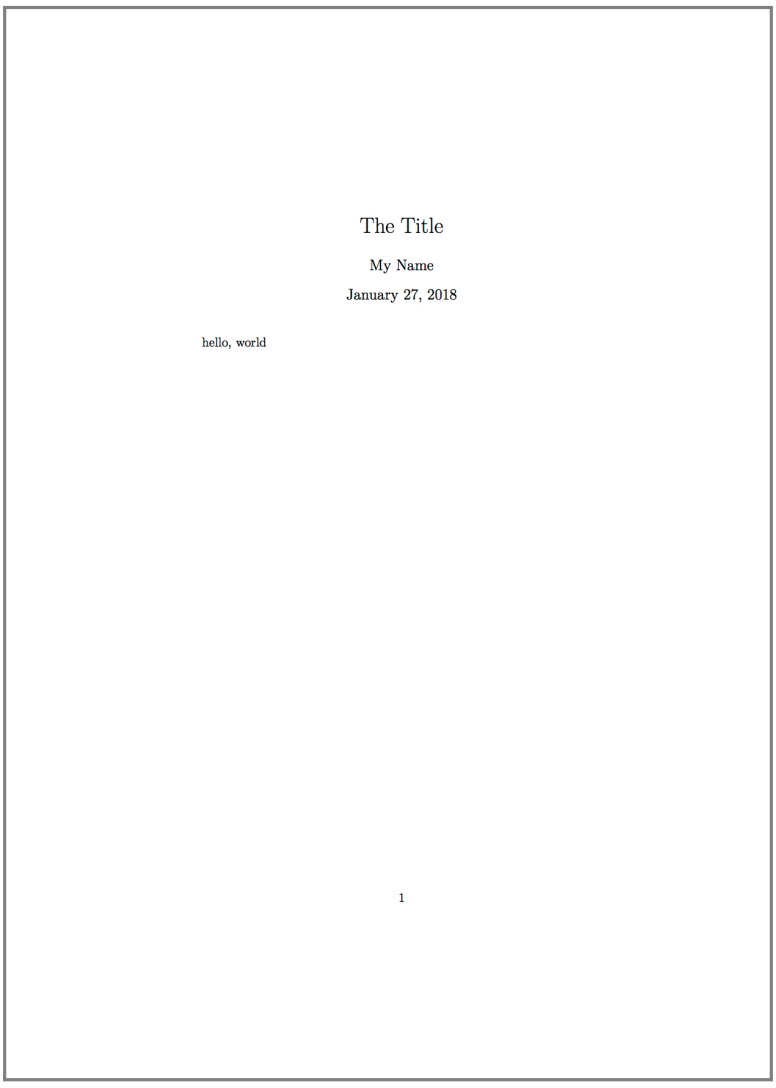

Figure 7: LaTeX 文档添加标题、作者和注释后的编辑效果

3.2 LaTeX 的标题页和注释的添加方法

3.2.1 LaTeX 的标题页

LaTeX 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{<title>}
\author{<author>}
\date{<date>}
```

其中前两个命令是必须的，\date 命令可选。LaTeX 还提供了一个 \today 命令自动生成当前日期，可用在 \date 的参数里，或者别的地方。

在 \title、\author 等命令内可以使用 \thanks 命令生成标题页的脚注，如：

```
\author{Mary\thanks{E-mail:*****@***.com}}
```

在信息给定后，就可以使用 \maketitle 生成一个简单的标题页了。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 命令中指定 titlepage / notitlepage 选项修改。

3.2.2 LaTeX 的注释

在科技著作的手稿中经常可以看到在边空里，在行间空白处，密密麻麻地写了很多文字，其中有些内容是遗漏补充，需要加入正文，有些则是注释，如对文稿中某些论述的说明、出处或考证等，这些注释内容通常不进入正文，专供作者备忘。在 LaTeX 源文件中，可在任何位置使用注释标记，将上述这些注释内容完整地保留下来，以备作者查阅，而在编译后的 PDF 文件中还看不到这些注释内容。

在写作或者修改论文时，有时会将某些语句、段落或者图表公式等全部或部分删除，可事后又觉得不妥，但很难恢复，只好重新再写。在 LaTeX 中，可以利用注释的方法将这些需要删除的内容或可能会用到的资料保存下来，以备不时之需。科研论文要经过反复推敲，多次修改，注释功能非常实用。

LaTeX 中的 % 符号是注释符，它表示其右侧的文字是对左侧命令或文本的说明；在编译源文件时，LaTeX 将忽略注释符及其右侧的所有字符。

4 了解 LaTeX 的第三步

4.1 结构化 LaTeX 文档的实现

1. 建立 一个新文档，将以下代码 复制 进入文档中， 保存 。

```
\documentclass{article}
\title{Hello World}
\begin{document}
\maketitle
\section{Hello China} China is in East Asia.
\subsection{Hello Beijing} Beijing is the capital of China.
\subsubsection{Hello Dongcheng District}
\paragraph{Tian'anmen Square}is in the center of Beijing
\subparagraph{Chairman Mao} is in the center of Tian'anmen Square
\subsection{Hello Guangzhou}
\paragraph{Sun Yat-sen University} is the best university in Guangzhou.
\end{document}
```

2. 运行 工具栏中的 快速构建 选项，观察编辑结果（**Figure 8**）。

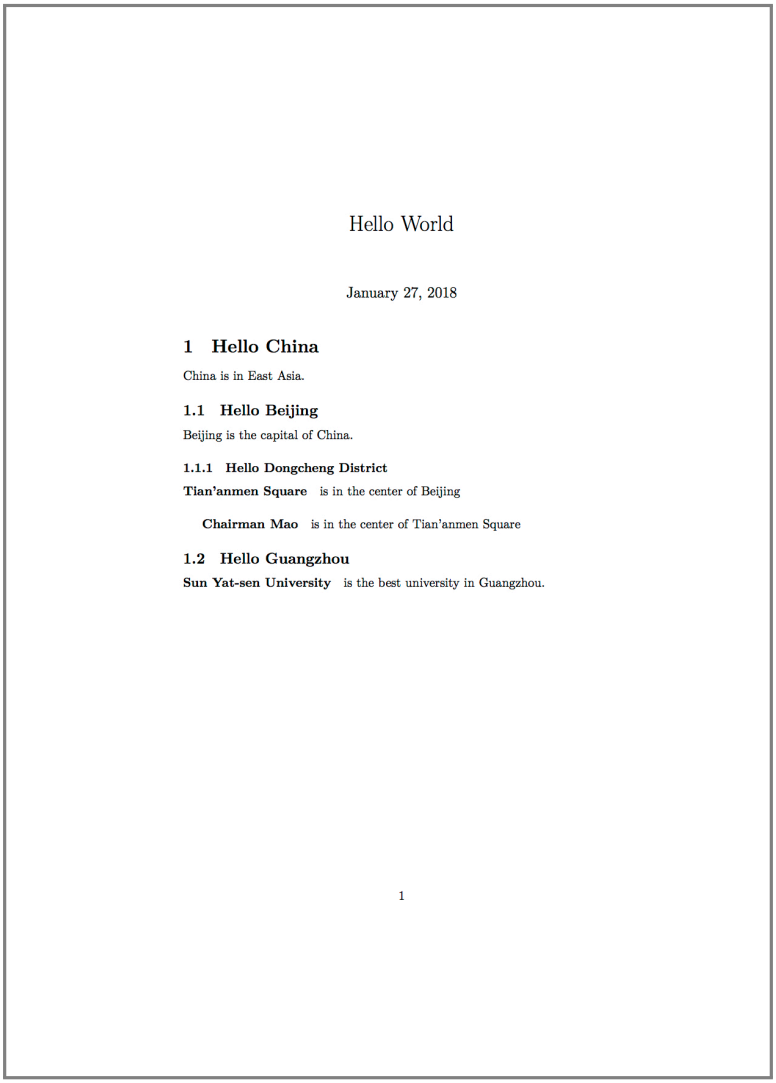

Figure 8: 结构化 LaTeX 文档的编辑效果

4.2 区分 LaTeX 章节和段落的方法

- **一篇结构化且条理清晰的文档一定是层次分明的，通过不同的命令分割为章、节、小节：**

  - LaTeX 的三个标准文档类 article、report 和 book 提供了一系列命令，用以划分章节、生成章节标题并自动编号：

```
\section{<title>}  \subsection{<title>}  \subsubsection{<title>}  \paragraph{<title>}  \subpara-
graph{<title>}
```

  - \part 命令用以将整个文档分割为大的分块，但不影响 \section 等的编号：

```
\part{<title>}
```

- book 和 report 提供了章一级的结构：

```
\chapter{<title>}
```

- **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。每个命令有两种变体：**

  - 带可选参数的变体：

```
\section[<short title>]{<title>}
此时：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>}
此时：标题不带编号，也不生成目录项和页眉页脚。
```

注意：LaTeX 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。

5 了解 LaTeX 的第四步

5.1 实现在 LaTeX 中生成目录

1. 建立 一个新文档，将以下代码 复制 进入文档中， 保存 。

```
\documentclass{article}
\begin{document}
\tableofcontents
\section{Hello China} China is in East Asia.
\subsection{Hello Beijing} Beijing is the capital of China.
\subsubsection{Hello Dongcheng District}
\paragraph{Hello Tian'anmen Square}is in the center of Beijing
\subparagraph{Hello Chairman Mao} is in the center of Tian'anmen Square
\end{document}
```

2. 运行 工具栏中的 快速构建 选项，观察编辑结果（**Figure 9**）。

5.2 向 LaTeX 文档加入目录的方法

在 LaTeX 中生成目录非常容易，只需在合适的地方使用命令：

```
\tableofcontents
```

正确生成目录项，一般需要多次编译源代码。可以使用 \chapter* 或 \section* 这样不生成目录项的命令。

如果之后又想手动生成该章节的目录，可以在标题命令后面使用：

```
\addcontentsline{toc}{<level>}{<title>}
```

其中 <level> 为章节层次 chapter 或 section 等，<title> 为需要生成目录项的章节标题。

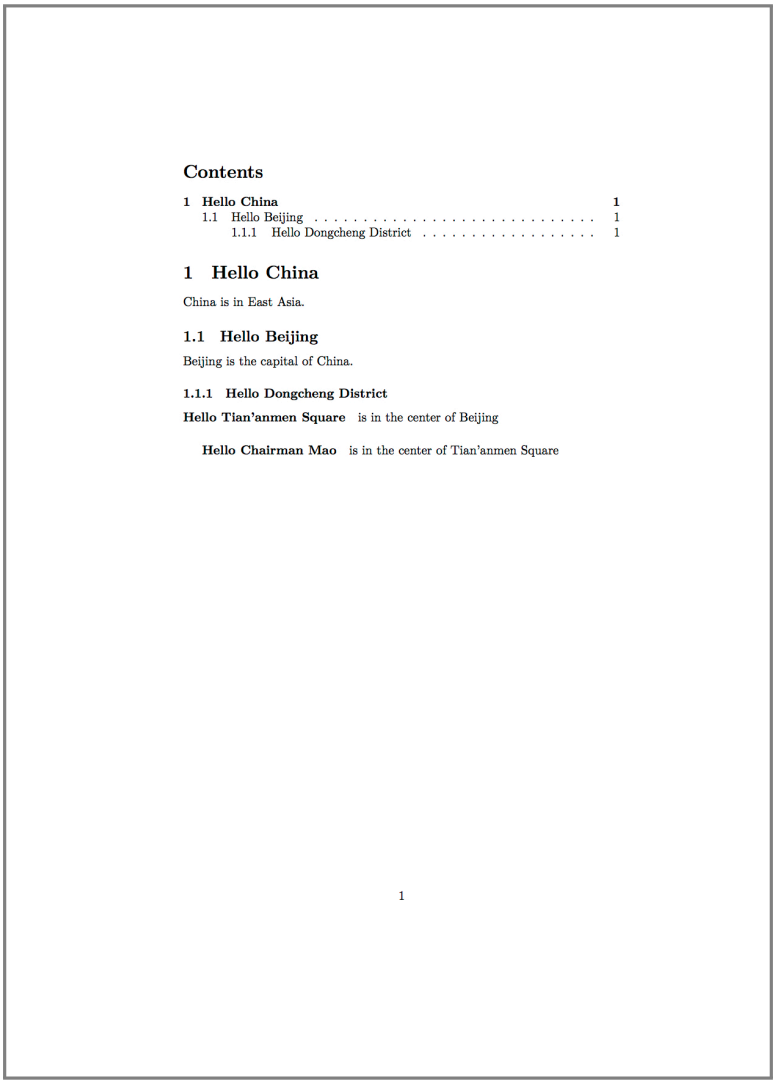

Figure 9: LaTeX 文档添加目录后的编辑效果

6 了解 LaTeX 的第五步

6.1 LaTeX 文档换行的实现

1. 建立 一个新文档，将以下代码 复制 进入文档中， 保存。

```
\documentclass{article}
\begin{document}
Beijing is
the capital
of China.

New York is

the capital

of America.

Amsterdam is \\ the capital \\
of Netherlands.
\end{document}
```

2. 运行 工具栏中的 快速构建 选项，观察编辑结果（**Figure 10**）。

6.2 令 LaTeX 文档立即换行的方法

源文件经过编译后生成 PDF 格式文件，其行宽是由导言中版面尺寸命令或版面设置宏包设定的。因此，系统会根据所设定的版心宽度，即文本行的宽度，自动进行换行，必要时还可按照所设定的断词规则，在换行处自动进行断词处理，所以通常无须干预系统的自动换行工作。但有时根据某些情况或是某种环境要求，还是需要人为进行换行或者禁止换行处理。

如果由于某种原因需要在某处中断排版另起一行，可在该处插入换行命令：

```
\\  \\*  \\[高度]  \newline
```

系统将在此处结束当前行的排版并新起一行。其中：

- 命令 \\ 表示在此换行。
- 而命令 * 表示在此换行，但是不能在此换页。
- 命令 \\[高度] 表示在此换行，并且在当前行与下一行之间增加一段高度为此处所要求的高度的垂直空白，高度通常为正值，也可根据需要取负值，该命令多用于调节表格数据行之间的间隔，或多行公式之间的间隔。
- 命令 \newline 只能用于段落模式，而其他 3 种命令还可在某些数学环境中使用。

注意：LaTeX 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。每行开头的空格忽略不计。

注意：行末的回车视为 一个空格；但连续两个回车，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。

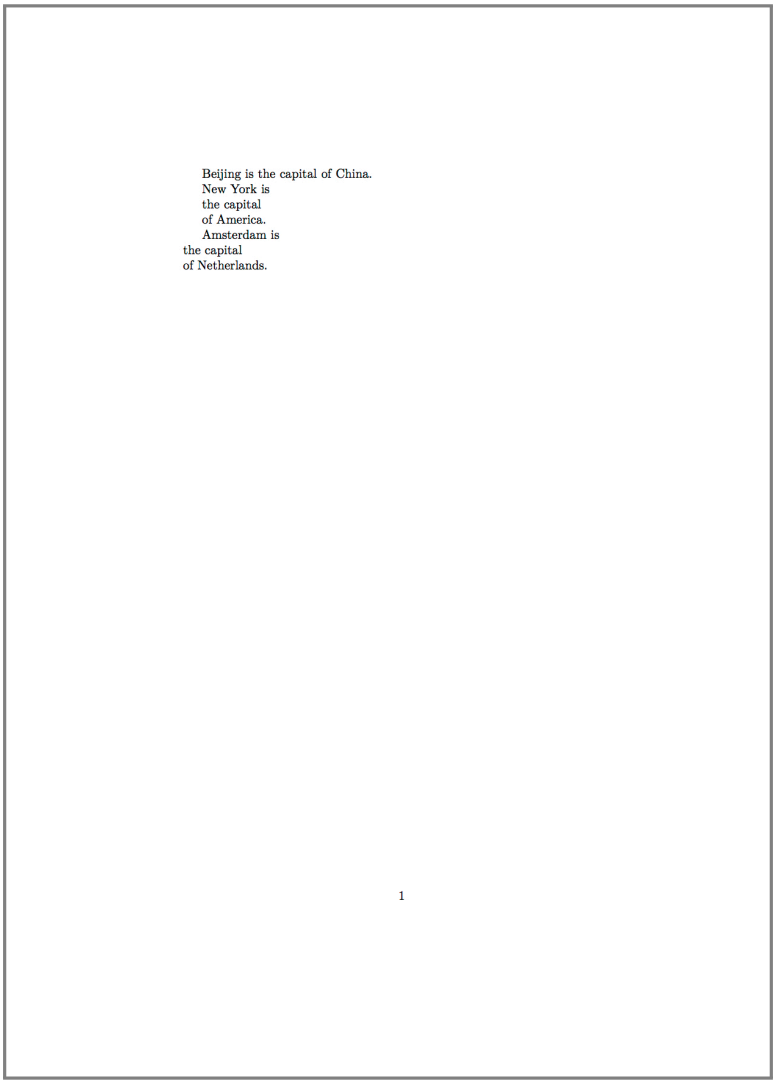

Figure 10: 不同 LaTeX 文档换行方法的编辑效果

7 了解 LaTeX 的第六步

7.1 使用 LaTeX 实现数学公式的编辑

1. 建立 一个新文档，将以下代码 复制 进入文档中， 保存 。

```
\documentclass{article}
\usepackage{amsmath}
\usepackage{amssymb}
\begin{document}
The Newton's second law is F=ma.

The Newton's second law is $F=ma$.

The Newton's second law is
$$F=ma$$

The Newton's second law is
\[F=ma\]

Greek Letters $\eta$ and $\mu$

Fraction $\frac{a}{b}$

Power $a^b$

Subscript $a_b$

Derivate $\frac{\partial y}{\partial t} $

Vector $\vec{n}$

Bold $\mathbf{n}$

To time differential $\dot{F}$

Matrix (lcr here means left, center or right for each column)
\[
\left[
\begin{array}{lcr}
a1 & b22 & c333 \\
d444 & e555555 & f6
\end{array}
\right]
\]

Equations(here \& is the symbol for aligning different rows)
\begin{align}
a+b&=c \\
d&=e+f+g
\end{align}

```

```
\[
\left\{
\begin{aligned}
&a+b=c \\
&d=e+f+g
\end{aligned}
\right.
\]

\end{document}
```

2. 运行 工具栏中的 快速构建 选项，观察编辑结果（**Figure 11**）。

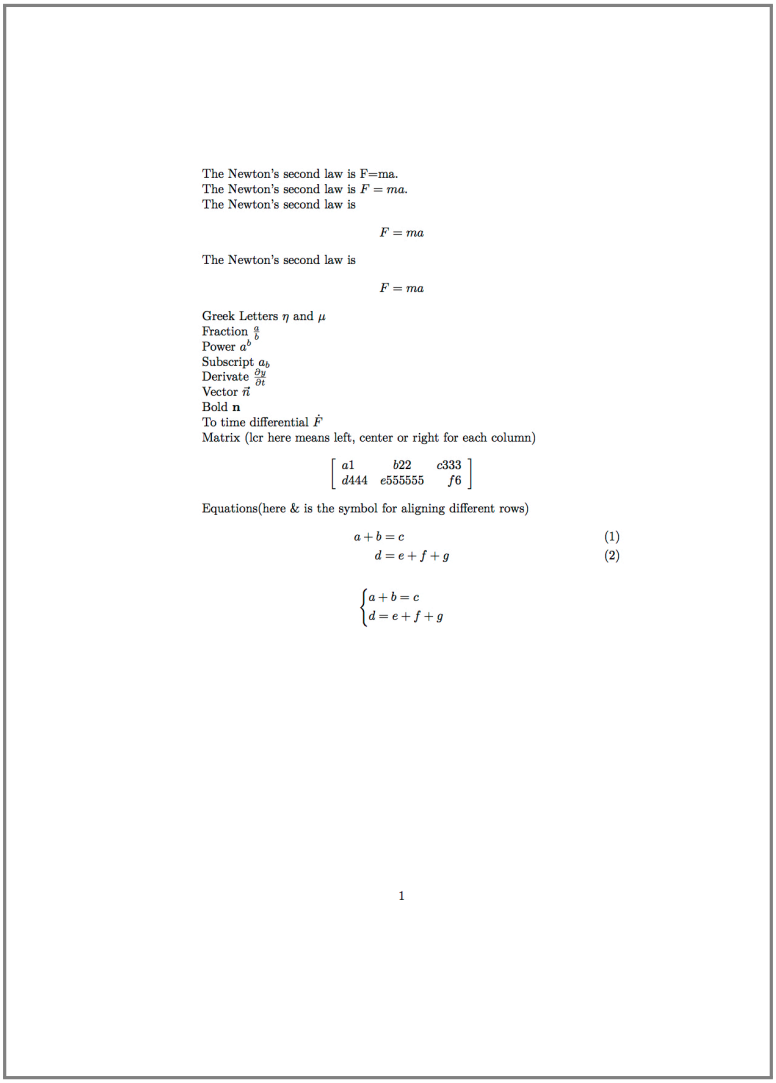

Figure 11: LaTeX 文档适用于各种简单或者复杂的数学公式

7.2 $\mathcal{AMS}$-LaTeX 宏集

$\mathcal{AMS}$-LaTeX 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。

7.2.1 公式排版基础

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。

行内公式

LaTeX 可以将数学公式与文字混合排版，此方式称为**行内公式**。

执行代码

```
Add $a$ squared and $b$ squared to get $c$ squared. Or, using a more mathematical approach: $a^2 + b^2 = c^2$
```

执行效果

Add a squared and b squared to get c squared. Or, using a more mathematical approach: $a^2+b^2 = c^2$

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）。

行间公式

LaTeX 也可以将数学公式以单独列为一行的方式来排版，此方式称为**行间公式**。

执行代码

```
Add $a$ squared and $b$ squared to get $c$ squared
\begin{equation}
a^2 + b^2 = c^2
\end{equation}
Einstein says
\begin{equation}
E = mc^2 \label{clever}
\end{equation}
This is a reference to \eqref{clever}.
```

执行效果

Add a squared and b squared to get c squared

$$a^2 + b^2 = c^2 \tag{1}$$

Einstein says

$$E = mc^2 \tag{2}$$

This is a reference to (2).

执行代码

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

执行效果

It's wrong to say

$$1 + 1 = 3 \tag{dumb}$$

or

$$1 + 1 = 4$$

除了可以为每个公式手动取消编号，LaTeX 提供了一对命令 \[和 \] 用于生成不带编号的行间公式，与之等效的是 displaymath 环境和 equation* 环境。行间公式是否有带编号，体现了带星号的 equation* 环境和不带星号的 equation 环境之间的区别。

执行代码

```
Again\ldots
\begin{equation*}
a^2 + b^2 = c^2
\end{equation*}
or you can type less for the same effect: \[ a^2 + b^2 = c^2 \]
or if you like the long one:
\begin{displaymath}
a^2 + b^2 = c^2
\end{displaymath}
```

执行效果

Again…

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

or you can type less for the same effect:

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

or if you like the long one:

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

TeX 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。

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。

注意：对比 LaTeX 的行内公式与行间公式，由于行内公式为了与文字相适应，其在排版大的公式元素（分式、巨算符等）时显得很“局促”。

7.2.2 数学模式和文本

当使用 $ 开启行内公式输入，或是使用 \[命令、equation 环境时，便进入了所谓的**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全部被忽略。数学符号的间隙默认完全由符号的性质（关系符号、运算符等）决定。需要人为引入空隙时，使用 \quad 和 \qquad 等命令。
2. **不允许有空行（分段）**。每个公式（每组多行公式）自成一个段落。
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 \mathrm 命令。\amsmath 提供了更加方便的 \text 命令。

执行代码

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

执行效果

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

注意：\text 命令仅适合在公式中穿插少量文字。如果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，而不是滥用 \text 命令。

7.2.3 一般数学符号与上下标

- 一般符号
  - 小写希腊字母符：α \alpha、β \beta、η \eta、μ \mu
  - 大写希腊字母符（首字母大写）：Γ \Gamma、Δ \Delta
  - 无穷大符号：∞ \infty
  - 普通省略号，具有两种形式。需要注意的是，这两种形式的省略号有各自合适的用途。另外，\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。

执行代码

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

执行效果

$a_1, a_2, \dots, a_n$

$a_1 + a_2 + \cdots + a_n$

- 矩阵中会用到的竖排或斜排省略号：⋮ \vdots、⋱ \ddots

- **指数、上下标和导数**

- LaTeX 用 ^ 和 _ 标明上、下标。需要注意的是，上、下标的内容（子公式）一般需要用花括号包裹，否则上、下标只对后面的第一个符号起作用。

执行代码

```
$p^3_{ij} \qquad m_\mathrm{Knuth}\qquad \sum_{k=1}^3 k $\\[5pt]
$a^x+y \neq a^{x+y}\qquad e^{x^2} \neq {e^x}^2$
```

执行效果

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

$a^x + y \neq a^{x+y}$ $e^{x^2} \neq e^{x2}$

- 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。

执行代码

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

执行效果

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

7.2.4 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行间公式，或是反过来。

执行代码

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

执行效果

In display style:

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

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

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

执行代码

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

执行效果

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

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

执行代码

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

执行效果

Pascal's rule is

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

7.2.5 关系符与算符

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、约等号 $\approx$ \approx、等价 $\equiv$ \equiv、正比 $\propto$ \propto、相似 $\sim$ \sim 等等。此外，倾斜的关系符号 $\leqslant$ \leqslant 和 $\geqslant$ \geqslant 由 amssymb 提供。

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。

执行代码

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

执行效果

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

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、除号 ÷ \div、点乘 · \cdot、加减号 ± \pm / ∓ \mp 等等。

∇ \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，类似于巨算符。

执行代码

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

执行效果

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

对于求模表达式，LaTeX 提供了 \pmod 和 \bmod 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。

执行代码

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

执行效果

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

此外，amsmath 允许用户用 \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。

导言区

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

执行代码

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

执行效果

$$\operatorname{argh} 3 = \operatorname*{Nut}_{x=1} 4x$$

7.2.6 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。

执行代码

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

执行效果

In text: $\sum_{i=1}^{n} \quad \int_{0}^{\frac{\pi}{2}} \quad \oint_{0}^{\frac{\pi}{2}} \quad \prod_{\epsilon}$

In display:

$$\sum_{i=1}^{n} \quad \int_{0}^{\frac{\pi}{2}} \quad \oint_{0}^{\frac{\pi}{2}} \quad \prod_{\epsilon}$$

巨算符的上下标用作其上下限。**行间公式**中，积分号默认将上下限放在右上角和右下角，求和号默认在上下方；**行内公式**一律默认在右上角和右下角。可以在巨算符后使用 \limits 手动令上下限显示在上下方，\nolimits 则相反。

执行代码

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

执行效果

In text: $\sum\limits_{i=1}^{n} \quad \int\limits_{0}^{\frac{\pi}{2}} \quad \prod\limits_{\epsilon}$
In display:

$$\sum\nolimits_{i=1}^{n} \quad \int\limits_{0}^{\frac{\pi}{2}} \quad \prod\nolimits_{\epsilon}$$

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}。

执行代码

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

执行效果

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

$$\sum_{\begin{subarray}{l} 0\le i\le n \\ j\in \mathbb{R} \end{subarray}} P(i,j) = Q(n)$$

7.2.7 数学重音、箭头和上下括号

数学符号可以像文字一样加重音，比如对时间求导的符号 $\dot{r}$ \dotr、$\ddot{r}$ \ddotr、表示向量的箭头 $\vec{r}$ \vecr、表示欧式空间单位向量的 $\hat{\mathbf{e}}$ \hat{\mathbf{e}} 等。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。

执行代码

```
$\bar{x_0} \quad \bar{x}_0$\\[5pt]
$\vec{x_0} \quad \vec{x}_0$\\[5pt]
$\hat{\mathbf{e}_x} \quad \hat{\mathbf{e}}_x$
```

执行效果

$\bar{x_0} \quad \bar{x}_0$

$\vec{x_0} \quad \vec{x}_0$

$\hat{\mathbf{e}_x} \quad \hat{\mathbf{e}}_x$

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。

执行代码

```
$0.\overline{3} = \underline{\underline{1/3}}$ \\[5pt]
$\hat{XY} \qquad \widehat{XY}$\\[5pt]
$\vec{AB} \qquad \overrightarrow{AB}$
```

执行效果

$0.\overline{3} = \underline{\underline{1/3}}$

$\hat{XY} \qquad \widehat{XY}$

$\vec{AB} \qquad \overrightarrow{AB}$

此外，除了作为上下标之外，箭头还用于表示过程。amsmath 的 \xleftarrow 和 \xrightarrow 命令可以为箭头增加上下标。

执行代码

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

执行效果

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

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

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

执行代码

```
$\underbrace{\overbrace{a+b+c}^6 \cdot \overbrace{d+e+f}^7 }_\text{meaning of life} = 42$
```

执行效果

$$\underbrace{\overbrace{a+b+c}^{6} \cdot \overbrace{d+e+f}^{7}}_{\text{meaning of life}} = 42$$

7.2.8 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。除小括号 `()`、中括号 `[]` 之外，其余都是 LaTeX 命令，包括大括号 `\{ \}`。

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。LaTeX 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.`。

执行代码

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

执行效果

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

当不满意于 LaTeX 自动调节的定界符大小时，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` 等但是，`\bigl` 和 `\bigr` 不必成对出现。

执行代码

```
$\Bigl((x+1)(x-1)\Bigr)^{2}$\\
$\bigl( \Bigl( \biggl( \Biggl( \quad \bigr\} \Bigr\} \biggr\} \Biggr\} \quad \big\| \Big\| \bigg\| \Bigg\|
\quad \big\Downarrow \Big\Downarrow \bigg\Downarrow \Bigg\Downarrow$
```

执行效果

$$\Bigl((x+1)(x-1)\Bigr)^{2}$$

$$\bigl(\Bigl(\biggl(\Biggl(\quad \bigr\} \Bigr\} \biggr\} \Biggr\} \quad \big\| \Big\| \bigg\| \Bigg\| \quad \big\Downarrow \Big\Downarrow \bigg\Downarrow \Bigg\Downarrow$$

7.2.9 长公式折行

用 LaTeX 编写长公式时，通常应当避免写出需要折行的长公式。如果一定要折行的话，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。另外，类似于 equation*，multline* 环境排版不带编号的折行长公式。

执行代码

```
\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}
```

执行效果

$$a + b + c + d + e + f + g + h + i$$
$$= j + k + l + m + n$$
$$= o + p + q + r + s$$
$$= t + u + v + x + z \quad (3)$$

注意：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。

7.2.10 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。

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：

执行代码

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

执行效果

$$a = b + c \quad (4)$$
$$= d + e \quad (5)$$

align 环境会给每行公式都编号。但仍然可以用 \notag 去掉某行的编号。为了对齐其他二元算符，可以将分隔符放在等号右边，这时需要给等号后添加一对括号 {} 以产生正常的间距。

执行代码

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

执行效果

$$
\begin{aligned}
a &= b + c && (6)\\
&= d + e + f + g + h + i + j + k + l \\
&\quad + m + n + o && (7)\\
&= p + q + r + s && (8)
\end{aligned}
$$

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。

执行代码

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

执行效果

$$
\begin{aligned}
a &= 1 & b &= 2 & c &= 3 && (9)\\
d &= -1 & e &= -2 & f &= -5 && (10)
\end{aligned}
$$

如果不需要按等号对齐，只需罗列数个公式，gather 将是一个很好用的环境。而 align 和 gather 有对应的不带编号的版本 align* 和 gather*。

执行代码

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

执行效果

$$a = b + c \tag{11}$$
$$d = e + f + g \tag{12}$$
$$h + i = j + k$$
$$l + m = n \tag{13}$$

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 结尾的环境用法与之前不以 -ed 结尾的环境用法一一对应。

执行代码

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

执行效果

$$\begin{aligned} a &= b + c \\ d &= e + f + g \\ h + i &= j + k \\ l + m &= n \end{aligned} \tag{14}$$

另外，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。

7.2.11 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。数组可作为一个子公式，在外套用 \left、\right 等定界符。

执行代码

```
\[
\mathbf{X} = \left(
\begin{array}{ccc}
x_1 & x_2 & \ldots \\
x_3 & x_4 & \ldots \\
\vdots & \vdots & \ddots
\end{array} \right)
\]
```

执行效果

$$\mathbf{X} = \left(\begin{array}{ccc} x_1 & x_2 & \ldots \\ x_3 & x_4 & \ldots \\ \vdots & \vdots & \ddots \end{array} \right)$$

值得注意的是，aligned 等环境也可以用定界符包裹，还可以利用空的定界符进行排版。

执行代码

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

执行效果

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

不过，采用 amsmath 提供的 cases 环境，能够更轻松地完成上述排版操作。

执行代码

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

执行效果

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

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。

执行代码

```
\[
\begin{matrix}
1 & 2 \\
3 & 4
\end{matrix} \qquad
\begin{bmatrix}
p_{11} & p_{12} & \ldots & p_{1n} \\
p_{21} & p_{22} & \ldots & p_{2n} \\
\vdots & \vdots & \ddots & \vdots \\
p_{m1} & p_{m2} & \ldots & p_{mn}
\end{bmatrix}
\]
```

执行效果

$$\begin{matrix} 1 & 2 \\ 3 & 4 \end{matrix} \qquad \begin{bmatrix} p_{11} & p_{12} & \ldots & p_{1n} \\ p_{21} & p_{22} & \ldots & p_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ p_{m1} & p_{m2} & \ldots & p_{mn} \end{bmatrix}$$

8 了解 LaTeX 的第七步

8.1 向 LaTeX 文档插入图片

1. 选择一张图片想要插入 LaTeX 文本中的文件，先将其 转换 成 .eps 格式，将其命名为 figure1.eps ，再 放置 在该 LaTeX 文件的同级目录下。
2. 建立 一个新文档，将以下代码 复制 进入文档中， 保存 。

```
\documentclass{article}
\usepackage{graphicx}
\begin{document}
\includegraphics[width=4.00in,height=3.00in]{figure1.eps}
\end{document}
```

3. 运行 工具栏中的 快速构建 选项，观察编辑结果（**Figure 12**）。

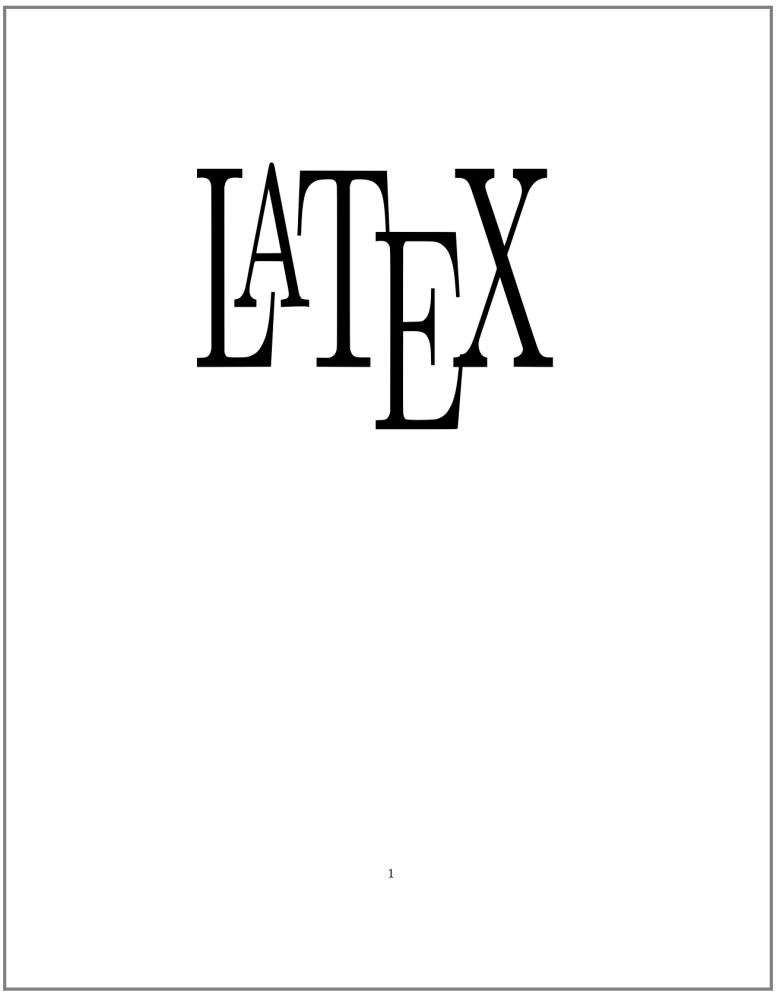

Figure 12: LaTeX 文档插入图片后的效果

8.2 支持插图功能的 graphicx 宏包

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 宏包时要给定 dvipdfmx 选项；而使用 pdflatex 或 xelatex 命令编译时不需要。那是因为早期常使用 latex + dvips 组合命令，后者将 .dvi 文件转为 .ps（PostScript）文件，可进一步通过 ps2pdf 工具生成 .pdf 文件。由于 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，所以要给定选项（缺省为 dvips），因此再调用 graphicx 宏包时要选择 dvipdfmx 项。具有超链接功能的 hyperref 宏包的调用方式与此相同。

过去，LaTeX 文档只能插入 .eps 格式的图片，因此需要把 .jpg 格式的图片转成 .eps 格式。但其实，LaTeX 发展到今天，对插入图片格式的支持已不仅局限于 .eps 格式。事实上不同的编译命令支持的图片格式范围各异（**Figure 13**）。不难发现，xelatex 命令所支持的图片格式种类最多且无需调用其他宏包，故最为推荐。

格式	矢量图	位图
latex + dvipdfmx	.eps	n/a
└（调用 bmpsize 宏包）	.eps .pdf	.jpg .png .bmp
pdflatex	.pdf	.jpg .png
└（调用 epstopdf 宏包）	.pdf .eps	.jpg .png
xelatex	.pdf .eps	.jpg .png .bmp

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 命令仍不支持除 .jpg 和 .png 以外的位图）。

Figure 13: LaTeX 各种编译方式支持的主流图片格式

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