

# L<sup>A</sup>T<sub>E</sub>X 使用简介

---

作者	XiaoCY
版本	1.0
完成日期	2020-02-04
最后修改	2020-02-04

# 目录

<b>1</b>	<b>写在前面</b>	<b>1</b>
<b>2</b>	<b>基础介绍</b>	<b>1</b>
2.1	文档组成 . . . . .	1
2.2	使用符号 . . . . .	1
2.3	设置字体 . . . . .	2
2.4	定义命令 . . . . .	3
2.5	环境介绍 . . . . .	3
<b>3</b>	<b>编写文档</b>	<b>4</b>
3.1	页面设置 . . . . .	4
3.2	添加目录 . . . . .	5
3.3	段落格式 . . . . .	6
3.4	定理环境 . . . . .	6
3.5	添加列表 . . . . .	7
3.6	使用颜色 . . . . .	8
3.7	使用图表 . . . . .	9
3.8	数学公式 . . . . .	12
3.9	插入代码 . . . . .	12
3.10	插入引用 . . . . .	13
<b>4</b>	<b>写在后面</b>	<b>15</b>
4.1	常用宏包 . . . . .	15
4.2	使用模板 . . . . .	15

## 表格

1	特殊符号及其说明 . . . . .	2
2	字体的坐标 . . . . .	2
3	文字强调 . . . . .	3
4	CT <sub>E</sub> X 文档常用选项 . . . . .	5
5	页码数字格式 . . . . .	5
6	定理类环境格式 . . . . .	7
7	彩色表格的常用命令 . . . . .	11

## 1 写在前面

L<sup>A</sup>T<sub>E</sub>X 是一种区别于 Word 的排版软件，其格式由各种命令、环境控制，很容易做到内容与格式分离，这是它区别于 Word 的重要一点。这篇文章主要是本人学习 L<sup>A</sup>T<sub>E</sub>X 的记录<sup>[1]</sup>，记录将常用的基本操作以便后期查阅，同时也为感兴趣的小伙伴们做一个粗略的介绍。

本人学习时安装的发行版为 T<sub>E</sub>X Live，对于不知道如何入门的小伙伴们，我同样推荐这个发行版，它避免了很多繁琐的配置。编译器以 X<sub>Y</sub>L<sup>A</sup>T<sub>E</sub>X 为主，编辑器可采用 TeXstudio。

## 2 基础介绍

### 2.1 文档组成

L<sup>A</sup>T<sub>E</sub>X 文档的格式通常是以后缀.tex 结尾的文本文件，为了正常使用中文和 Unicode 的特殊符号，务必将文本以 UTF-8 编码进行保存。

L<sup>A</sup>T<sub>E</sub>X 文档可分为导言区和正文区。导言区在命令`\begin{document}`之前，通常对文档的性质做一些设置，也可以自定义一些命令。导言区之后为正文区，是文档的主要内容。

在导言区内，我们首先需要声明文档类，本文采用了命令`\documentclass{ctexart}`来声明 C<sub>T</sub><sub>E</sub>X 文档类。此外，很多优秀的宏包可以辅助我们对文档格式进行控制，为了使用这些宏包，我们需要在导言区使用命令`\usepackage{<宏包名>}`。（如无特殊说明，本文的命令示意中以尖括号连同内容表示命令，使用时不需要添加尖括号。）

在正文区内，我们只需要输入文档的正文即可。编辑器内部的各种换行不会引起排版后文档的换行，中文文档的多余空格也不会出现在文档中出现。

文字换行可用双斜线`\\`实现，而换段则采用空行的形式，或者使用命令`\par`。tex 文件内的多个空行或多个`\par`不会在正文引起多余的空行。换行和换段的区别是：当设置了段落首行缩进时，换行不会引入缩进。

### 2.2 使用符号

正文中绝大多数符号都可以由键盘直接输入，如 @。但是有一些符号在 L<sup>A</sup>T<sub>E</sub>X 中具有特殊的作用，因而不能直接在正文中使用。通常情况下我们可以在这些符号前面加上反斜线`\`，但是也有个别例外。这些特殊符号的作用和使用方法如表1 所示。

除此之外，符号-在 L<sup>A</sup>T<sub>E</sub>X 正文中有很多用途：在数学模式下它是减号，如  $3-1=2$ ；单独使用时它是英文连字符，如 good-looking；两个连用（--）时用来表示数字范围，如 1-10；三个连用（---）时是破折号—比如这里。

为了在正文中使用空格，可以采用反斜线和空格组合实现。此外，有一种称为幻影的神奇空格，它由命令`\phantom{<内容>}`生成，空格的长度与<内容>所占据的长度相同。

表 1: 特殊符号及其说明

符号	作用	输入方式
~	不可打断的空格	<code>\~{}</code>
#	用于宏定义	<code>\#</code>
\$	数学模式	<code>\\$</code>
%	注释符	<code>\%</code>
^	上标	<code>\^{}</code>
&	用于表格对齐	<code>\&amp;</code>
{ }	用于分组	<code>\{ \}</code>
_	数学下标	<code>\_</code>
\	宏命令和转义符	<code>\textbackslash</code>

表 2: 字体的坐标

字体族	带参数的命令	申明命令	效果
罗马	<code>\textrm{&lt;text&gt;}</code>	<code>\rmfamily</code>	Roman font family
无衬线	<code>\textsf{&lt;text&gt;}</code>	<code>\sffamily</code>	Sans serif font family
打字机	<code>\texttt{&lt;text&gt;}</code>	<code>\ttfamily</code>	Typewriter font family
字体形状	带参数的命令	申明命令	效果
直立	<code>\textup{&lt;text&gt;}</code>	<code>\upshape</code>	Upright shape
意大利	<code>\textit{&lt;text&gt;}</code>	<code>\itshape</code>	<i>Italic shape</i>
倾斜	<code>\textsl{&lt;text&gt;}</code>	<code>\slshape</code>	<i>Slanted shape</i>
小型大写	<code>\textsc{&lt;text&gt;}</code>	<code>\scshape</code>	SMALL CAPITALS SHAPE
字体系列	带参数的命令	申明命令	效果
中等	<code>\textmd{&lt;text&gt;}</code>	<code>\mdseries</code>	Medium series
加宽加粗	<code>\textbf{&lt;text&gt;}</code>	<code>\bfseries</code>	<b>Bold extended series</b>

## 2.3 设置字体

字体具有五种不同的性质，在  $\text{\LaTeX}$  中一起决定了文字最终的输出效果。字号（font size）是指文字的大小，常常被独立出来，看作不同于字体的单独的性质；字体编码（font encoding）指字体包含的符号，一般情况下不直接进行设定。使用最多的是其他的三个性质：字体族（font family）、字体形状（font shape）、字体系列（font series）<sup>①</sup>。字体的这三个性质也常称为字体的坐标。

$\text{\LaTeX}$  提供了各种命令来对字体进行修改。命令和效果如表 2 所示。

中文字体没有英文字体那么复杂的成套的变体，各个字体之间一般都是独立的。因此，中文字体一般只用不同的字体族进行区分。`ctex` 宏包和文档类（如 `ctexart`）预定义了 Windows 常用的四种字体族：宋体、黑体、楷书、仿宋。为了方便使用，`ctex` 宏包和文档类提供了简化命令：`\songti` 宋体，`\heiti` 黑体，`\kaishu` 楷书，`\fangsong` 仿宋。

`\zihao{<字号>}` 可以用来修改字体的大小，例如 `\zihao{4}` 将字号设置为四号，字号前

<sup>①</sup>通常指字体的粗细和宽度

表 3: 文字强调

命令	效果
<code>\CJKunderdot{&lt;文字&gt;}</code>	强调文字
<code>\CJKunderline{&lt;文字&gt;}</code>	强调文字
<code>\CJKunderdblline{&lt;文字&gt;}</code>	强调文字
<code>\CJKunderwave{&lt;文字&gt;}</code>	强调文字
<code>\CJKsout{&lt;文字&gt;}</code>	强调文字
<code>\CJKxout{&lt;文字&gt;}</code>	强调文字

添加负号如`\zihao{-4}`则表示小四号。

除了修改字体和字号外, `ctex` 宏包和文档类还增加了各种文字强调的方式, 命令和效果如表3 所示。

## 2.4 定义命令

LaTeX 中的命令又称为宏, 它们都以反斜线开头, 一般格式为:

无参数: `\command`

有  $n$  个参数: `\command{<arg1>}{<arg2>}\dots{<argn>}`

有可选参数: `\command[<opt>]{<arg1>}{<arg2>}\dots{<argn>}`

例如, 本文档在生成目录时就采用了命令`\tableofcontents`。

用户可以在导言区自定义命令, 定义命令的基本格式为:

```
\newcommand{<命令名>}[<参数个数>][<首参数默认值>]{<定义>}
```

其中, 命令名应当符合要求, 以反斜线开头; 当定义无参数命令时, 参数个数和首参数默认值可连同中括号省略; 当不给定默认值时, 首参数默认值可连同中括号省略; 当给定多个参数时, 命令的定义中用井号加数字表示参数, 如`#2`表示第二个参数。

例如我们可以定义一个命令来简化四元数乘法, 按顺序输入两个四元数, 输出四元数的乘法。定义为:

```
\newcommand{\quadprod}[2] % 可换行定义命令内容
{\ensuremath{\mathfrak{\#1} \otimes \mathfrak{\#2}}}
```

这样就定义了一个新命令`\quadprod`。当使用`\quadprod{q_b^a}{q_c^b}`时, 编译后将输出  $q_b^a \otimes q_c^b$ 。

除了用`\newcommand`构造命令外, 还可以使用`\renewcommand`和`\providecommand`。它们的用法相同。区别在于, `\renewcommand`用于覆盖之前存在的同名命令; `\providecommand`则会检测是否存在同名命令, 若存在, 则保留之前的用法, 新定义不生效。

## 2.5 环境介绍

与命令相似, 环境也可分为有参数的环境和无参数的环境。有参数环境的一般格式为:

```
\begin{<环境名>}[<可选参数>]{<其他必要参数>}
  <环境内容>
\end{<环境名>}
```

一个环境就是一个分组，它限定了一些命令的作用范围。除了使用环境，也可以用成对的花括号 `{}` 直接产生一个分组。

我们给出一个分组的例子：为了将局部字体修改为黑体，只需要`{\heiti 像这样}`构造分组，就能得到像这样的局部黑体内容，而不会引起后面字体的改变。

## 3 编写文档

### 3.1 页面设置

个人认为，页边距是使用  $\text{\LaTeX}$  编写文档时要考虑的第一个问题。因为如果最后修改页边距，一些所谓的盒子的内容有可能会超出边距，从而使文档需要重复调整。如果提前设置好边距，就能随时编译预览时发现问题。

$\text{\LaTeX}$  对各种距离的定义较为复杂，为了能够快速修改，可以使用 `geometry` 宏包，并在导言区使用`\geometry`命令对尺寸进行设置。

如本文采用的设置为：

```
\usepackage{geometry}
\geometry{ % 设置整体页面格式
  a4paper,
  left = 2.1cm,
  right = 2.1cm,
  bottom = 2.1cm,
  top = 2.5cm
}
```

文档的第一页往往包含标题和作者等信息，也可以使用一个单独的封面。在标准文档类中，我们可以在正文中用`\maketitle`生成标题页。该命令需要在导言区提前申明各种信息，如`\title`、`\author`等。

除了使用预定义的标题页，我们也可以用 `titlepage` 环境自定义封面。该环境提供没有页码的单独一页，并使后面的内容页码从 1 开始计数。

文档可使用`\section`、`\subsection`、`\subsubsection`分别声明节、小节、小小节。 $\text{\LaTeX}$  会自动进行编号。当需要使用到附录时，可使用`\appendix`命令后，再分别使用`\section`声明各级附录。附录格式的修改可使用 `appendix` 宏包，此处不具体说明。

对于  $\text{\CTEX}$  文档类<sup>①</sup>，可以使用`\CTEXsetup`命令来设置各章节标题的格式：

<sup>①</sup>包括书籍 `ctexbook`，报告 `ctexrep`，文档 `ctexart`。

表 4: CTEX 文档常用选项

选项	使用说明	值的示例
name	< 前名 >, < 后名 >	{第, 节}
number	设置编号的格式	{\chinese{section}}
format	章节名和章节标题的全局格式	{\bfseries}
nameformat	章节名和编号的格式	同 format
numberformat	仅控制编号格式	同 format
titleformat	仅控制章节标题格式	同 format
aftername	章节名与标题之间的内容	{\hspace{2ex}}
beforeskip	章节标题前的段间距	{\vspace{2em}}
afterskip	章节标题后的段间距	同 beforeskip
indent	章节标题的缩进长度	同 aftername

```
\CTEXsetup[<选项1>=<值1>,<选项2>=<值2>...]{<对象类型>}
```

对于 ctexart 文档, 常用的对象类型包括 section、subsection 和 subsubsection 等。常用的选项如表4 所示。

### 3.2 添加目录

目录是最基本的自动化工具。L<sup>A</sup>T<sub>E</sub>X 会自动收集章节命令所定义的各章节标题, 用命令\tableofcontents即可输出。类似地, 命令\listoffigures和\listoftables 会分别收集 figure 和 table 中\caption命令的图表标题, 产生图表的目录。

目录部分的页码常用大写的罗马数字表示, 这可以使用命令\pagenumbering{<格式>}进行修改, 其中格式允许的值如表5 所示。

在目录部分结束之后, 常换页进入正文部分, 并将页码计数重置为 1, 格式设置为阿拉伯数字, 可采用以下命令:

```
\clearpage           % 换页
\pagenumbering{arabic} % 设置页码格式
\setcounter{page}{1}  % 正文重新开始于 1
```

表 5: 页码数字格式

格式	说明
arabic	阿拉伯数字
roman	小写的罗马数字
Roman	大写的罗马数字
alph	小写的字符形式
Alph	大写的字符形式



### 3.3 段落格式

在 C<sub>T</sub><sub>E</sub>X 文档类中已经预设了正确的首行缩进，如果需要在某一段临时取消缩进，可以使用 `\noindent` 命令；相反，为了在本来没有缩进的环境中临时设置缩进，可使用 `\indent` 命令。这两个命令的作用范围均为一个段落，因而不需要再进行分组加以限定。

除了段落的首行缩进，另一个关于分段的重要参数是段与段之间的垂直距离。这个距离由变量 `\parskip` 控制，可以采用命令 `\setlength{\parskip}{<length>}` 进行修改。这时应当注意，该命令会对后续的段落均产生影响。因此当只修改某些段落的段间距时，需要用花括号限制 `\setlength` 的作用范围。

在一些情况下，我们需要在文档内插入一段引文。`quote` 环境常用来引用小段文字，该环境下没有首行缩进，且左右边距比正常文本稍微大一些，且增加了环境前后段的间距。比如下面给出一个使用这种环境的例子。

学而时习之，不亦说乎？

而对于大段文字的引用，通常需要引入首行缩进，这时可采用 `quotation` 环境，如下面一段文字。

小车正穿行在落基山脉蜿蜒曲折的盘山公路上。克里朵夫·李维静静地望着窗外，发现每当车子即将行驶到无路的关头，路边都会出现一块交通指示牌：“前方转弯！”或“注意！急转弯”。而拐过每一道弯之后，前方照例又是一片柳暗花明、豁然开朗。

山路弯弯、峰回路转，“前方转弯”几个大字一次次地冲击着他的眼球，也渐渐叩开了他的心扉：原来，不是路已到了尽头，而是该转弯了。路在脚下，更在心中，心随路转，心路常宽。学会转弯也是人生的智慧，因为挫折往往是转折，危机同时是转机。

### 3.4 定理环境

定理环境实际上是一类环境，在使用前需要在导言区进行定义：

```
\newtheorem{thm}{定理}
```

这样我们就得到了一个名为 `thm` 的定理类环境，它在使用时会自动产生形如“定理 1”的提示。用同样的方法我们还可以定义引理、公理等。有时候我们希望定理的编号包含章节号，可在定义时增加参数即可：

```
\newtheorem{thm}{定理}[section]
```

定理环境还可以有一个可选参数，即定理的名字。下面给出这个环境的使用示例：

**定理 3.1 (勾股定理)** 直角三角形斜边的平方等于两直角边的平方和。

`ntheorem` 宏包扩充了定理类环境的格式，在导言区使用 `\theoremstyle{<格式>}` 可以方便地选择格式。可用的预定义格式如表6 所示。

表 6: 定理类环境格式

格式	说明
plain	默认格式
break	定理头换行
marginbreak	编号在页边, 定理头换行
changebreak	定理头编号在前文字在后, 换行
change	定理头编号在前文字在后, 不换行
margin	编号在页边, 定理头不换行
nonumberplain	同 plain 格式, 没有编号
nonumberbreak	同 break 格式, 没有编号
empty	没有编号和定理名, 只输出可选参数

该宏包具有很多设置命令和辅助功能, 比如在使用该宏包时添加[thmmarks]选项后, 可以使用`\theoremsymbol`命令在定理类环境末尾添加符号, 这对定义证明环境表示证毕符号非常有用:

```
% 导言区
\usepackage[thmmarks]{ntheorem}
{ % 利用分组使设置只对该分组内的定理类有效
  \theoremstyle{nonumberplain}
  \theoremheaderfont{\bfseries}
  \theorembodyfont{\normalfont}
  \theoremsymbol{\ensuremath{\Box}}
  \newtheorem{proof}{证明}
}
```

这样, 使用该定理类将会出现下面的效果:

**证明** 当  $x > 0$  时, 将  $e^x$  进行泰勒展开, 有

$$e^x = \sum_{k=0}^{+\infty} \frac{x^k}{k!} > \sum_{k=0}^n \frac{x^k}{k!} = P_n(x)$$

证毕。

□

### 3.5 添加列表

L<sup>A</sup>T<sub>E</sub>X 标准文档类提供了三种列表环境: 带编号的 `enumerate` 环境、不编号的 `itemize` 环境和使用关键字的 `description` 环境。在列表环境内部使用`\item`命令开始一个新的列表项, 它可以带一个可选参数表示手动编号或关键字。

这三种列表环境可以嵌套使用 (最多四层), L<sup>A</sup>T<sub>E</sub>X 会自动处理不同层次之间的缩进编号, 例如:

1. 使用 `enumerate` 环境直接用 `\item` 产生一个列表项
- 1b. 使用 `\item[1b.]` 测试手动编号
2. 这里恢复正常的使用方法，并引入一次 `itemize` 嵌套
  - 嵌套的第一层默认编号
    - † 使用 `\item[\dag]` 修改符号
  - 恢复到默认设置

### 3.6 使用颜色

可以利用 `xcolor` 宏包在文档中使用颜色。基本的命令包括声明式命令 `\color{<颜色>}` 和带参数的命令 `\textcolor{<颜色>}{<文字>}`。前者对后面的内容都生效，而后者仅将参数部分的文字按指定的颜色输出。例如：

```
{ % 使用分组，避免对后文上色
  \bfseries\color{red}
  这是一段夹杂\textcolor{blue}{蓝色字符}的红色段落。
}
```

**这是一段夹杂蓝色字符的红色段落。**

除了对文字上色，我们也能够对文字的“背景”进行上色。这用到所谓的“盒子”的功能，基本语法如下：

```
\pagecolor{<页面颜色>}
\colorbox{<盒子颜色>}{<文字>}
\fcolorbox{<线框颜色>}{<盒子颜色>}{<文字>}
```

为了直接使用颜色名如 `blue` 来表示颜色，可以在使用 `xcolor` 宏包时添加 `dvipsnames` 选项来扩充预定义的颜色名。除此之外，我们还有两种方式来定义新的颜色：

```
\definecolor{<色彩名>}{<模型>}{<分量值>}
\colorlet{<色彩名>}{<色彩标记>}
```

其中，色彩名是指新定义的颜色名字。常用的颜色模型有：`gray`（灰度）、`rgb`（红绿蓝）和 `cmymk`（印刷四分色）。`xcolor` 提供了用已有颜色生成新颜色的色彩标记方法，常用的有：

```
半色调： <颜色>!<百分数>
混合色： <颜色>!<百分数>!<颜色>
互补色： -<颜色>
```

下面给出一些实例供参考：

```
\begin{quote}
  \bfseries
  \textcolor{Blue!80}{80\%的蓝色} \\
  \textcolor{Red!60!Cyan}{60\%的红色加40\%青色} \\
  \colorbox{Green}{\textcolor{-Green}{绿色背景与互补色文字}}
\end{quote}
```

产生的效果为：

80% 的蓝色

60% 的红色加 40% 的青色

绿色背景与互补色文字

### 3.7 使用图表

L<sup>A</sup>T<sub>E</sub>X 支持直接使用代码绘图，但这通常非常复杂。更常用的操作是利用其他软件绘图，然后将图片插入到文档之中。graphicx 宏包提供了 `\includegraphics` 命令来实现插图。通常情况下，我们会把插图单独放在 figure 环境中。例如，图 1 采用了以下代码：

```
\begin{figure}[hbp]
  \centering
  \includegraphics[width=0.5\textwidth]{figure/mywife1.jpg}
  \caption{某春雨的超电磁炮}
  \label{fig:railgun}
\end{figure}
```



图 1: 某春雨的超电磁炮

从这个代码就可以看出，进入环境之后首先设置了在环境中生效的居中命令；通常情况下图片的名字在图片下面，所以先用`\includegraphics`命令插入图片，并通过可选参数设置图片的宽度<sup>①</sup>；再由`\caption`生成标题；考虑到需要使用的交叉引用，接着用`\label`打上标签。应当注意的是，为了产生正确的交叉引用，标签应当在`\caption`命令之后。

同样地，为了使用表格，可以使用在 `table` 环境中嵌套 `tabular` 环境。除此接触的小伙伴们可能会好奇，为什么制作表格还需要两层嵌套，而且这两个环境从含义上看来并没有很大的区别。我们可以这样认为：`tabular` 环境用来制作表格，而 `table` 环境用来指明表格在文档中的位置。

例如，表 5 的代码如下：

```
\begin{table}[hbp]
  \centering
  \caption{页码数字格式}
  \label{tab:pagenum}
  \begin{tabular}{ll}
    \toprule
    格式 & 说明 \\
    \midrule
    arabic & 阿拉伯数字 \\
    roman & 小写的罗马数字 \\
    Roman & 大写的罗马数字 \\
    alph & 小写的字符形式 \\
    Alph & 大写的字符形式 \\
    \bottomrule
  \end{tabular}
\end{table}
```

首先我们来分析这个代码：我们在最外围申明了表格所处的 `table` 环境；然后设置该环境下内容居中显示；由于表格的标题一般在表格上方，所以先生成标题并打上标签；剩下的一大块便是由 `tabular` 环境构造的表格。

然后我们讨论下 `tabular` 环境的使用方法：该环境有一个参数用来指明列格式，常用的列格式包括：

- `l` — 本列左对齐。
- `c` — 本列居中。
- `r` — 本列右对齐。
- `p{<列宽>}` — 指定列宽并允许自动换行。

<sup>①</sup>常用选项有：宽度 `width`、高度 `height` 和缩放比例 `scale`

- | — 画一条竖线，不占据表项计数。
- @{<内容>} — 任意添加内容，不占表项计数。
- \*{<计数>}{<列格式说明>} — 将给定列格式按计数重复多次

在表内，采用符号 & 指明对齐位置，并使用 \\ 进行换行。表格中可以使用 \hline 绘制横线，或使用 \cline{<开始>-<结束>} 指定横线的范围。本文引用了 booktabs 宏包产生三线表的横线，可使用命令 \toprule、\midrule、\bottomrule 分别绘制顶部横线、中间横线和底部横线。为了像 \cline 那样绘制指定位置的横线，该宏包还提供了 \cmidrule 命令。表格中绘制与单元格等高的竖线可用 \vline 命令。

在不单独指定列宽的情况下，tabular 环境可以根据内容自动改变表格宽度，这在绝大多数情况下是非常好用的。然而有时候我们希望指定表格的总宽度，这就需要 tabularx 宏包提供的 tabularx 环境。该环境在指定列格式之前必须指定表格的总宽度。同时，该宏包提供了一个特殊的列格式 X，它能够根据表的内容自动改变列宽，也可以和其他列格式一起使用。

由于 X 格式默认是左对齐的，如果需要居中则需要 \centering 进行设置。然而这个命令会影响单元格内的自动换行，需要加上命令 \arraybackslash 进行恢复。如果这种格式比较常用，我们可以利用下面的命令自定义一个新的列样式。

```
\newcolumntype{Y}{>{\centering\arraybackslash}X}
```

有时表格需要用到单元格合并，对于列合并，可以直接使用下面的代码实现。

```
\multicolumn{<列数>}{<列格式>}{<内容>}
```

列合并的项数可以为 1，常用于修改指定单元格的格式。表格的行合并相对复杂，需要利用 multirow 环境的同名命令，该命令的用法如下：

```
\multirow{<行数>}{<列宽>}{<内容>}
\multirow{<行数>}*{<内容>}
```

前一种格式会在内容达到指定列宽后自动换行，后一种格式会自动修改列宽，更为常用。

如果在使用颜色宏包 xcolor 时添加 table 选项，就可以在表格中使用颜色。使用方法如表 7 所示。

表 7: 彩色表格的常用命令

命令	作用	常用位置
\columncolor{<颜色>}	改变列的颜色	>{<...>} 列格式说明符
\rowcolor{<颜色>}	改变行的颜色	表格某一行的开头
\cellcolor{<颜色>}	改变单元格颜色	单元格内
\rowcolors{<起始行>}{<奇数色>}{<偶数色>}	产生颜色交错的表格	tabular 等环境之前

以上介绍了插图和制表的基本命令，现在我们来讨论下 figure 和 table 环境。细心的小伙伴一定发现上面两个例子在使用环境时均添加了[hbp]的可选参数，这是什么意思呢？

对于其他的环境，例如插入代码所使用的 lstlisting 环境，它在排版时保持了前后文相对关系不变，因而是固定的。而 figure 环境和 table 环境都是所谓的浮动体，它会根据当前页面的内容自动设置位置，从而避免页面出现大片的空白或图表超出页面范围。这两个环境的可选参数代表环境内容可以出现的位置，它们分别代表：

- **h** (here)，浮动体可以放置在代码所在的上下文位置。
- **t** (top)，浮动体可以放置在页面顶部，这可以是当前页面或下一页，当前页面排版时可能会出现在实际代码之前。
- **b** (bottom)，浮动体可以放置在页面底部。
- **p** (page) 一个或多个浮动体放置在单独的页面。

浮动体的允许位置可以由 h、t、b、p 任意组合，组合顺序不影响排版效果。figure 和 table 的默认选项是[tpb]。浮动体其他的一些预设值可能会限制浮动体内容出现的位置，可以在选项中使用！来临时取消预设值，如将可选参数设为[!htb]。

如果浮动体的自动排版达不到理想的效果，可以用 float 宏包对其进行设置。该宏包提供了一个新的位置选项 H，它只能单独使用，其功能是取消环境的浮动。

### 3.8 数学公式

$\LaTeX$  相关宏包提供了很好的数学支持，这些宏包包括：amsmath、amssymb、amsfonts。

### 3.9 插入代码

$\LaTeX$  输入特殊符号时不太便利，然而有时候我们必须经常性地使用特殊符号，例如在排版计算机程序源代码的时候。此时需要使用抄录功能。

使用命令\verb<符号><抄录内容><符号> 可以将抄录内容原封不动地输出到正文中，两端的符号可以任意给定，但是要确保相同。例如\verb|\TeX|采用“|”作为边界。

在插入大段代码时，可以使用 listings 宏包提供的 lstlisting 环境。但是直接使用往往得不到很好的效果，通常我们需要在导言区使用\lstset进行设置。例如本文的设置如下：

```
% 导言区
\usepackage{listings}
\usepackage{listings}
\lstset{ % 代码环境整体设置
    basicstyle = \ttfamily,
    keywordstyle = \bfseries,
    commentstyle = \rmfamily\upshape,
    stringstyle = \ttfamily\slshape,
```

```

    tabsize = 4,
    backgroundcolor = \color{lightgray},    % 需先引入颜色宏包
    frame = single,
    language = TeX                          % 设置默认语言
}

```

需要说明的是，`lstlisting` 环境对缩进是敏感的。这就是说，从`\begin{lstlisting}`开始的缩进都将进入正文。该环境还可以使用可选参数，对环境进行临时修改。为了使代码根据不同语言实现高亮，可以在使用该环境时添加可选参数 `language` 进行设置，例如上面的例子就引入了参数`[language=TeX]`。

### 3.10 插入引用

最简单的引用便是使用`\footnote{<脚注内容>}`产生脚注，例如这个位置<sup>①</sup>。脚注是自动编号的，也可以使用可选参数修改编号，但是这不改变原来脚注的编号。

在例如表格等环境中，`\footnote`命令往往不能直接使用。这时我们可以把脚注标记和脚注内容分开<sup>②</sup>，这两条命令分别为`\footnotemark`和`\footnotetext{<脚注内容>}`。

LaTeX 默认脚注格式不太好看，我们可以在导言区进行以下设置进行优化。

```

\usepackage[perpage]{footmisc}    % 脚注 每页 清零
\usepackage{pifont}               % 优化 带圈的 数字
\renewcommand{\thefootnote}{\ding{\numexpr171+\value{footnote}}}

```

交叉引用可以通过一个符号标签引用文档中某个对象的编号、页码、或标题等信息，而不必知道这个对象具体在什么位置。我们需要两个步骤来实现交叉引用：定义标签和引用标签。

定义标签是在合适的位置给一个带参数的对象添加标签，命令为`\label{<标签>}`。标签的名字最好是简洁而有用的名字，不能包含特殊字符。标签的位置常在`\section`命令之后；或 `table` 和 `figure` 环境中`\caption`命令之后；也可以在公式之后。

创建标签后，可分别使用 `\ref`、`\pageref`、`\nameref` 引用标签对象的计数、所在页码和名称。对于公式的引用，通常会给编码加上英文括号，好在  $\mathcal{M}\mathcal{S}$  的宏包在支持数学的基础上增加了命令`\eqref`专门用于公式引用，所以公式的引用会略有不同。

例如，我们使用下面的代码进行交叉引用的测试：

```

% 前面已使用 \label{tab:ctexopt} 定义标签
表 \ref{tab:ctexopt} 在第 \pageref{tab:ctexopt} 页，
名字是 \nameref{tab:ctexopt}。

```

输出为：表 4 在第 5 页，名字是 CTeX 文档常用选项。

<sup>①</sup>这是一个脚注的例子

<sup>②</sup>这又是一个脚注



细心的朋友对比后会发现，`\tableofcontents`生成的目录不是超级链接，而且编译生成的 pdf 文档中不会出现标签。为了解决这个问题，只需要使用 `hyperref` 宏包。或者在申明文档类时加上可选项，如下

```
% 导言区
\documentclass[hyperref,UTF8]{ctexart}
```

同时，该宏包还提供了命令 `\url{<URL>}`和`\href{<URL>}{<文字>}` 产生超级链接。比如，这里我可以班门弄斧一下 我的博客。

除了上面这些引用，科技类文档中最重要的引用就是文献引用了。这类引用首先需要有一个数据库文件，其通常以 bib 为后缀。该文件由多项参考文献构成，这里给出其中的一个例子：

```
@article{canuto_embedded_2018,
title = {Embedded model control: {Reconciling}
        modern control theory and error-based control design},
volume = {16},
issn = {2198-0942},
shorttitle = {Embedded model control},
url = {https://doi.org/10.1007/s11768-018-8130-1},
doi = {10.1007/s11768-018-8130-1},
language = {en},
number = {4},
urldate = {2019-07-16},
journal = {Control Theory and Technology},
author = {Canuto, Enrico and Novara, Carlo and Colangelo, Luigi},
month = nov,
year = {2018},
keywords = {embedded model control, disturbance rejection,
            error loop, error-based design, Modern control theory},
pages = {261--283}
}
```

通常情况下 bib 文件不需要我们手动编写，文献检索网站和各种文献管理软件一般都支持这种 BibTeX 格式的导出。

文献的引用需要以下三个步骤：

1. 使用`\bibliographystyle`命令设定参考文献的格式，这通常在导言区完成。基本的文献格式有：`plain` — 按作者、日期、标题排序；`unsrt` — 不排序但保持引用次序；`alpha` — 使用一种三字母缩写的方式编号并按作者排序；`abbrv` — 与 `plain` 基本相同，但是定义了一些缩写。

2. 在正文中使用`\cite`命令引用所需要的文献，当引用多个文献时用逗号隔开。
3. 使用`\bibliography`命令指明要使用的文献数据库，即包含文献信息的 bib 文件。同时， $\text{\LaTeX}$  会在这个命令的位置插入参考文献列表。

例如，使用以下命令进行文献引用：

Canuto 教授提出了一种模型嵌入控制`\cite{canuto_embedded_2018}`。

其效果为：Canuto 教授提出了一种模型嵌入控制<sup>[2]</sup>。参考文献列表见本文末尾。

默认的文样式不能满足所有出版社的要求，这时我们可以使用 `natbib` 宏包进行修改。该宏包同时定义三种专用的格式：`plainnat`、`abbrvnat`、`unstrnat`。

`natbib` 宏包提供了`\setcitestyle`命令来设置引用命令的输出格式，在参数中可以设置以下选项：

- 选择引用模式：`authoryear` 代表作者年代模式，如 [Canuto,2018]；`numbers` 表示数字序号模式，如 [3]；`super` 表示数字上标模式，如 <sup>[64]</sup>。
- 括号：`round` 圆括号，`square` 方括号，或是用`open={<左括号>}`和`close={<右括号>}` 分别进行设置。
- 多个引用之间的标点：`semicolon` 分号，`comma` 逗号，或是用`citesep={<符号>}`进行设置。
- 作者与年代之间的符号：`aysep={<符号>}`。
- 同一作者的几个年代间的符号`yysep={<符号>}`。
- 在引用命令可选参数的说明文字前的符号`notesep={<符号>}`。

简单起见，我们可以在使用宏包时添加选项来快速修改格式，如下：

```
\usepackage[super,square]{natbib}
```

## 4 写在后面

### 4.1 宏包汇总

### 4.2 使用模板

## 参考文献

[1] 刘海洋.  $\text{\LaTeX}$  入门. 2013.

[2] Enrico Canuto, Carlo Novara, and Luigi Colangelo. Embedded model control: Reconciling modern control theory and error-based control design. *Control Theory and Technology*, 16(4):261–283, November 2018. ISSN 2198-0942. doi: 10.1007/s11768-018-8130-1. URL <https://doi.org/10.1007/s11768-018-8130-1>.