目录

LaTeX 学习手册

1 LaTeX 基础

1.1 LaTeX 文档结构

1. 文档结构：以下是一份最为简单的 LaTeX 文档：

```
\documentclass{article}
\begin{document}
Hello, \LaTeX
\end{document}
```

Hello, LaTeX

2. 命令：在 LaTeX 中，命令也称控制序列 (control sequence)，以一个反斜杠加上命令名构成。开头的命令 `\documentclass` 指定了使用的文档类，花括号 `{}` 内的内容是命令的参数，`article` 表示文章格式的文档类，除此之外可用的参数还有表示报告的文档类 `report` 和表示书籍的文档类 `book` 等。

   如果命令的参数只有一个字符，则花括号可省略，但需要用空格区分命令与参数。

   有些命令还存在可选参数，可选参数通常会在花括号前用方括号 `[]` 表示，例如：

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

   多数命令只在原地产生效果，但有些命令则会影响作用域内后面的所有内容，这种命令又称为声明 (declaration)。

   `\begin` 和 `\end` 一对命令定义了一个环境，环境表示命令或内容的作用范围。环境如果有备选或额外参数，只需在 `\begin` 中表示。`document` 环境当中的内容是文档正文，在此环境外书写的内容可能不会出现在文档中。

   语句 `\begin{document}` 之前的内容称为**导言区**，导言区可以留空，也可以编写文档所需要的信息与工具。

   `\LaTeX` 命令用于输出一个特殊的符号 LaTeX，像这样表示符号的命令还有很多。

3. 注释：以百分号 `%` 开头的部分是行注释。

   注释有时放在行尾，用于取消换行产生的一个多余的空格。

   如果要单独表示百分号，需要使用反斜杠转义 `\%`

4. 单位：LaTeX 中常用的衡量长度的单位有：

   - pt：磅 (point)
   - pc：四号字 (pica) 1 pc = 12 pt
   - in：英寸 (inch) 1 in = 72.27 pt
   - bp：大点 (bigpoint) 1 bp = $\frac{1}{72}$ in
   - cm：厘米 (centimeter) 1 cm = $\frac{1}{2.54}$ in
   - mm：毫米 (millimeter)
   - sp：TeX 的基本长度单位 scaled point，1 sp = $\frac{1}{65536}$ 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

5. 宏包：宏包的作用是扩展或调整 LaTeX 的排版功能。一个宏包往往能提供更多的命令、环境，或为内置的命令/环境添加更多功能。

   在导言区使用 `\usepackage{}` 即可引入宏包，然后便可以使用宏包提供的功能。部分宏包在引入时可以通过命令的可选参数调节需要引入的功能。

   例如，引入 PKG ctex 宏包可以在文档中排版中文字符，引入 PKG amssymb 宏包可以使用命令 `\bigstar` 表示一个填充五角星符号 ★

1.2 特殊符号

1. 空格：在 LaTeX 中，字符间的空白会自动调整。

   不带参数的命令后面的空格，要用空的花括号对加上空格来表示，否则空格会被忽略，例如：

`\LaTeX without \{\}`	LaTeXwithout {}
`\LaTeX{} with \{\}`	LaTeX with {}

2. 引号：英文单引号分左右引号。左单引号用重音符 `` ` `` 表示，右单引号用普通引号 `'` 表示，左双引号用连续两个重音符 ``` `` ``` 表示，右双引号用连续两个单引号 `''` 表示。

   英文下的引号嵌套需要借助 `\thinspace` 命令分隔，例如：

   ```
   ``\thinspace`single' quotes''
   ```
   “‘single’ quotes”

   中文引号可以直接输入。

3. 短横：英文短横有 3 种：

   - 连字符：用一个短横 `-` 表示，如 clear-cut ；
   - 数字起止符：用两个短横 `--` 表示，如 page 1–2 ；
   - 破折号：用三个短横 `---` 表示，如 Look—It's a dash 。

   中文破折号可以直接输入。

4. 省略号：英文省略号用 `\ldots` 符号表示，效果为 …。中文省略号可以直接输入。

5. 保留字符：以下字符在 LaTeX 中具有特殊含义，不能直接作为文档中的一个字符：

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

   除反斜杠外，其余字符均能用反斜杠的形式转义输出：

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

   反斜杠 `\` 可以使用以下方式得到：

   ```
   \textbackslash $\backslash$
   ```

1.3 文字样式

1. 粗体与斜体：广义的斜体命令是 `\textit{}` ，粗体命令是 `\textbf{}` 。`\emph{}` 命令用于强调文本，对西文字母而言就是变为斜体。例如：

   ```
   \emph{text} and \textbf{text}
   ```
   text and **text**

2. 字体样式：可以通过以下声明修改字体的字族、字系、字形效果，这三种类型声明相互独立，可以组合使用：

字族	`\rmfamily`	罗马字族 Roman
	`\sffamily`	无衬线字族 Sans Serif
	`\ttfamily`	罗马字族 Typewriter
字系	`\bfseries`	粗体 **Bold Font**
	`\mdseries`	中粗体 Middle
字形	`\upshape`	竖直 Upshape
	`\slshape`	斜体 *Slant*
	`\itshape`	意大利体 *Italic*
	`\scshape`	小号大写体 SmallCap

   但如果不存在这样的字体设计，某些效果可能不会生效。

   这些声明会影响之后的所有文本，如果只是改变局部字体样式，可以使用 `\text..` 这类命令，例如 `\textbf{}` 。

3. 字号：在 `\documentclass` 的可选参数可以指定正文字号大小，参见对可选参数的第一次介绍。可以通过以下声明相对地改变字号大小：

```
\tiny    \scriptsize   \footnotesize
\small   \normalsize   \large
\Large   \LARGE        \huge
\Huge
```

4. 基本下划线：原生的 下划线命令 是 `\underline`，上划线命令 是 `\overline`，它们均需要在公式环境 (参见 2.1 节) 中使用。

5. 更好的下划线：PKG ulem 宏包提供了更好的下划线命令，包括：

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

这些命令可以直接使用。

ulem 宏包修改了 `\emph` 命令的效果，会给强调文本加下划线，可能不是想要的效果，可以通过宏包的 `[normalem]` 选项取消该效果。

1.4 段落处理

1. 文本对齐：有左中右三种对齐方式，如下表：

对齐方式	声明形式	环境形式	参数形式
居中对齐	`\centering`	`center` 环境	`\centerline{}`
左对齐	`\raggedright`	`flushleft` 环境	`\leftline{}`
右对齐	`\raggedleft`	`flushright` 环境	`\rightline{}`

声明形式的对齐会影响接下来所有文本的对齐方式，环境形式只影响环境内的对齐方式，参数形式只影响参数内文本的对齐方式。

2. 换行：LaTeX 中的多个空格会被视为一个，多个换行符也会被视为一个。使用两个回车可用于在正文中换行。使用两个反斜杠 `\\` 用于强制换行。

3. 段落与缩进：使用 `\par` 可以生成一个带缩进的新段。

4. 换页：使用 `\newpage` 开始新的一页。

2 文档排版

2.1 页面设置

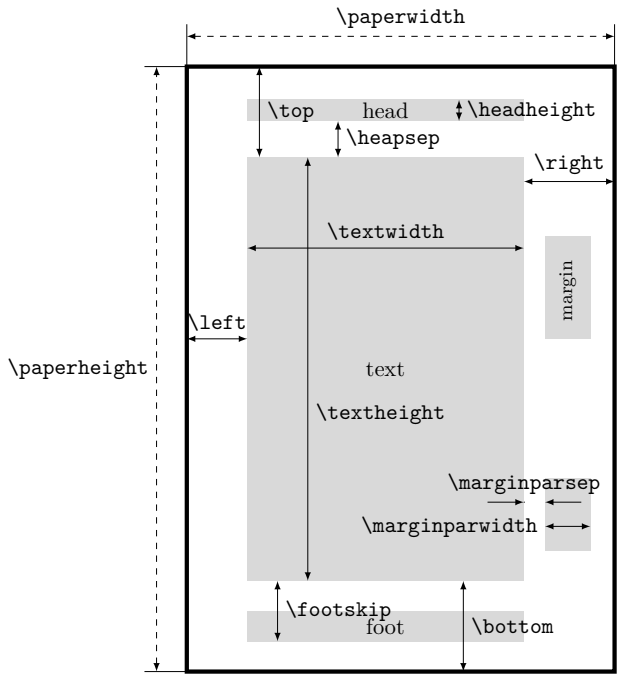
1. 页面元素尺寸设置：

geometry 宏包提供了设置页面所需工具，使用宏包提供的 `\geometry{}` 命令可以方便地设置页面尺寸。一份文档具有的典型页面元素及其尺寸如右图所示。

通过在命令中将这些尺寸修改为新值即可。例如，本文档采用的设置为：

```
\usepackage{geometry}
\geometry{paper=a4paper,
          left=0.8cm,right=0.8cm,
          top=1.5cm,bottom=1.2cm}
```

这些设置也可以放在引入宏包时的可选参数内，即使用 `\usepackage[...options...]{geometry}` 的形式。



2. 页面尺寸设置：geometry 宏包提供了各种设置页面的工具，调整 `paper=`*name* 选项可以设置纸张尺寸名，例如 `a4paper`，`ansibpaper`，`letterpaper`，`executivepaper`，`legalpaper` 等。也可以通过 `papersize={`*width*`,`*height*`}` 调整特定的纸张宽高，或用 `paperwidth=`*size* 和 `paperheight=`*size* 选项单独调整宽或高。

   使用 `landscape` 选项可以设置纸张为横向，默认是 `portrait` 纵向。

3. 理解页面的元素：一个页面可以包含页眉 (head)、页脚 (foot)、侧边栏 (margin note) 和文本主体 (body) 这几个元素组成。这几个元素可以单独或共同构成总文本区 (total body)。

   使用 `scale=`*scalar* 选项可以控制总文本区占纸张总尺寸的比例，默认为 0.7 。总文本区默认设置为 `includehead` 表示包含页眉，可以使用 `include` 或 `ignore` 加上 `foot` 、`head` 、`headfoot` 、`mp` 、`all` 构成的选项自由调节总文本区的组成，并使用 `(total)?width=` 和 `(total)?height=` 以及 `total={`*width*`,`*height*`}` 调节总文本区的尺寸。

   文本主体的尺寸由 `textwidth=` 和 `textheight=` 以及 `body={`*width*`,`*height*`}` 调节。

   页面边距 `top left bottom right` 都是相对文本主体的边距。书籍相关的文档类一般都提供了 `twoside` 选项，此时纸张两面相对书脊位置是相反的，因此有 `inner=` 和 `outer=` 左右边距的意义。可以用 `hmargin=` 和 `vmargin=` 指定水平和竖直的两边距，或者用 `hmarginratio=` 指定水平左右或内外边距之比（默认为单页 1.0 和双页 0.67）以及 `vmarginratio=` 指定竖直上下边距之比。使用 `hcentering` 、`vcentering` 或 `centering` 可以使文本主体居中，从而使边距相等。在文档的左侧或内侧，可以使用 `bindingoffset=` 预留装订线宽度。

   页眉的高度使用 `head(height)?=` 参数指定，侧边栏的宽度由 `marginparwidth` 指定。页脚的高度主要由内容决定，也可以通过 `foot(skip)?=` 设置最大高度。它们离文本主体的间距分别由参数 `headsep` 、`footnotesep` 、`marginparsep` 指定。通过 `nohead` 、`nofoot` 、`nomarginpar` 可以清除对应部分的所有尺寸，但不会删除内容。

   总之，如果想查看页面元素的位置，可以通过 `showframe` 选项给所有元素加上边框，便于检查尺寸。

4. 页面样式：使用 `\pagestyle{}` 可以设置页面样式。页面样式主要影响页眉和页脚。默认可用的页面样式有：

样式	效果
`empty`	无页眉页脚
`plain`	无页眉，页脚仅中央显示页码
`headings`	无页眉，页脚包含页码和章节名
`myheadings`	无页眉，页脚包含页码和自定义信息

使用 `\thispagestyle{}` 可以仅设置某一页样式。例如，可以将封面页设置为 `empty` 样式。

5. 自定义页眉页脚：PKG fancyhdr 宏包提供了 fancy 页面样式，可以自定义页眉页脚内容。使用 lcr 和 head 或 foot 组成的各种命令如 \rhead{} 可以自定义页眉页脚的左中右内容。

   PKG fancyhdr 宏包还可以修改页眉线和页脚线宽，需要通过 \renewcommand{\headrulewidth}{0.4pt} 这种重定义命令的形式设置。

   通过命令 \thepage 可以获取当前的页码数。以下是一个综合示例：

```
\usepackage{fancyhdr}
\pagestyle{fancy}
\lhead{Chapter 2}
% ...
\cfoot{page \thepage}
\rfoot{$e^{\pi i}+1=0$}
\renewcommand{\footrulewidth}{0.4pt}
```

Chapter 2 \chead{} LaTeX

这里是正文部分

\lfoot{} page 1 $e^{\pi i}+1=0$

   使用 \fancyhead[]{} 和 \fancyfoot[]{} 可以解决书籍双页环境下的位置问题，该命令可选参数可以使用 EO 和 LCR 的组合表示位置，前者表示左页或右叶位置，后者表示左中右位置。

6. 多栏排版：

   使用文档类提供的 twocolumn 可选参数项可以使文章变成两栏，默认为 onecolumn 一栏。多栏文档的 \newpage 用于换栏，而 \clearpage 才是换页。

   排版过程中也可以使用 \twocolumn 或 onecolumn 声明来切换单双栏，同时执行换页、清空浮动队列（即将图表等内容输出）。如果双栏命令带上了可选参数，则可选参数的内容将作为双栏上方的跨栏文本，一般用作双栏前的标题。这两个声明可以放在导言区或当前分组环境内。

   多栏下栏之间的间隔距离由 \columnsep 给出；栏宽间隔和文本主题的宽度确定，由 \columnwidth 给出。

   如果要局部分栏或使用任意多栏，可以使用 PKG multicol 宏包提供的 multicols 环境，它的完整环境参数为 {multicols}{n}[*preface*][*skip*] 。一个额外的必选参数指定了分栏数，可以为 2–10 。可选参数 *preface* 指定多栏上方的跨栏文本（标题），另一个可选参数 *skip* 表示多栏环境的预留高度，如果当前文本区的内容小于该值，则系统将换页并从下一页开始排版它。但这个参数不影响多栏环境的实际高度。

   在 multicols 环境下，可以使用 \columnbreak 命令强制换栏。

   multicols 环境与文档类自带的双栏选项的区别在于，它追求最小的高度，因此左右两栏内容是等高的，左右两栏会自动调节内容。而文档类的双栏只有左栏排满后才转到右栏排版，两侧内容往往不相等。可以在 multicols 环境开始命令之前使用命令 \raggedcolumns 使每栏文本的底部可以不对齐，防止多栏的段落间距出问题。

7. 多栏排版的分隔线：多栏排版下栏之间的分隔线宽度由 \columnseprule=*size* 给出，它的默认值是 0pt ，因此默认不显示分隔线。

   使用宏包 PKG multicolrule 可以设置各种样式的多栏分隔线。使用声明 SetMCRule {line-style=...,width=...} 可以修改分隔线，选项 line-style 改变分隔线样式，可用值有 dots 、circles 、dotted 、dash-dot 、dashed 等；选项 width= 修改线宽，可以是 thin 、thick 、2pt 等。

8. 侧边栏与边注：多栏排版有专用的命令，侧边栏一般用于编写简短的注解。

使用 `\marginpar{}` 可以在对应位置添加边注，参数为边注内容。边注的位置由系统自动确定：

- 单面排版时，内容放在右侧；
- 双面排版时，outer 一侧（远离书脊一侧）；
- 两栏排版时，内容放在远离分隔线一侧。

如果为该命令提供可选参数，则若边注放在左侧，则显示可选参数的内容；否

这是一个边注，它的起始位置与当前行对齐，高度会自动适应

这个边注放在右边，它显示必选参数的内容

1

这个边注放在左边，它显示可选参数的内容

则放在右侧则显示必选参数的内容。

单面排版时，边注永远放在右侧，因此可选参数被忽略。

2

以上书籍文档类的效果，书籍的第一页通常都在右侧，它与双栏不同。

9. 脚注：

使用 `footnote {}` 命令可以为对应的位置添加脚注 [1]，对应的位置会自动添加一个脚注标号。使用命令的可选参数可以修改脚注序号 [3] 。

[1]脚注上方会自动添加脚注线

[3]例如，这个脚注命令的可选参数是 [3]

3 数学公式

3.1 引入公式

1. 行内公式：行内公式将嵌入到文本行中，公式垂直距离不会过高。行内公式有 3 种引入方式：

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

   例如：

`$ a + b = c $`	$a+b=c$

2. 行间公式：行间公式将单独出现并居中，垂直距离将适应公式内容。行间公式有 3 种引入方式：

   - `$$ ... $$`
   - `\[ ... \]`
   - `\begin{displaymath} ... \end{displaymath}`

   例如：

`\[ \sum_{i=1}^{n}x_i \]`	$$\sum_{i=1}^{n}x_i$$

如果用行内公式表达它，受行高限制呈现的效果稍有不同，接下来给出示例：

3. 公式尺寸：公式在不同位置会呈现不同尺寸，如果要让公式强制排版为特定的尺寸，可以使用以下几个声明：

命令	尺寸	示例
\displaystyle	行间公式尺寸	$\displaystyle\sum_{i=1}^{n} x_i$
\textstyle	行内公式尺寸	$\textstyle\sum_{i=1}^{n} x_i$
\scriptstyle	上下标公式尺寸	$\scriptstyle\sum_{i=1}^{n} x_i$
\scriptscriptstyle	次上下标尺寸	$\scriptscriptstyle\sum_{i=1}^{n} x_i$

4. 下标与上标：下标使用 _ 符号，上标使用 ^ 符号，例如 `a_n` a_n 或 `e^x` e^x

   如果上下标内不止一个字符，这些字符需要用花括号 {} 定界，例如 `x^{10}` x^{10} 。

5. 数学符号：在数学公式内的字母将会变为斜体。一些基本函数如 `\sin` 需要使用命令表示，从而用正体表示字母：

```
function \sin x
```

$$function \sin x$$

3.2 基本公式排版

1. 基本符号：加号 + $+$ 减号 - $-$ 和等号 = $=$ 直接使用对应字符创建，其余四则运算符号需要使用命令创建：乘号 `\times` $\times$ 除号 `\div` $\div$ 不等号 `\neq` $\neq$

2. 分式：使用命令 `\frac{}{}` ，两个参数分别表示分子和分母，例如：

```
\[ \frac{e^x}{\frac a b} \]
```

$$\frac{e^x}{\frac{a}{b}}$$

   同样，如果分子或分母内不止一个字符，需要用花括号定界。

3. 导数：直接使用单引号 ' 来表示：

```
$ f'(x) = a^x \ln x $
```

$$f'(x) = a^x \ln x$$

4. 三种带上下限的特殊符号：求和 `\sum` $\sum$ 、求积 `\prod` $\prod$ 、积分 `\int` $\int$ ，它们均使用下标 _ 符号和上标 ^ 符号来表示其上下限，例如：

```
\sum_{i=1}^{N} a_i
```

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

5. 根号：使用 `\sqrt` 命令，配合可选参数可以得到不同次数的根式：

```
\sqrt{9} \sqrt[3]{x}
```

$$\sqrt{9} \quad \sqrt[3]{x}$$

6. 极限：用 `\lim` 表示极限运算符，使用下标的形式表示底下的趋近关系，其中使用 `\to` 命令表示趋近的箭头，例如：

```
\[ \lim_{x \to \infty}f(x) \]
```

$$\lim_{x \to \infty} f(x)$$

3.3 公式的细节

1. 空格：源文件中在数学公式中的空格会被忽略。可以通过以下几种命令向公式中添加空格：

命令	空格大小	命令	空格大小
\,	3/18 空格	\:	4/18 空格
\;	5/18 空格	\!	−3/18 空格
\quad	1 空格	\qquad	2 空格
\␣	9/18 空格	(反斜杠加空格)	

负数尺寸的空格会拉进两个字符的距离。

2. 数学运算符与数学关系符：数学运算符和数学关系符区别在于两侧的间距。例如 `$a+b$` 表现为 $a+b$，但 `$+b$` 表现为 $+b$。前者的符号 + 是数学运算符，而后者的符号 + 是正号符号。数学运算符相比普通符号，左右两侧间距更大。数学关系符类似，且两侧间距比运算符略大，如 `$a<b$` 中的关系符表现为 $a<b$。

数学模式中花括号 {} 内的公式将独立考虑符号关系，可以与两侧符号隔离，使之不成为运算符或关系符，从而取消间距，如 `$a{+}b$` 表现为 $a{+}b$。

普通符号两侧没有预留间距，命令 `\mathbin{}` 与 `\mathrel{}` 则能分别把参数转换为二元运算符和二元关系符，并正确设置两侧的空距，在对齐时用处较大。

3. 公式与标点符号：公式中常用的标点符号有 `,;`，它们与普通符号的区别为只与右侧符号有间距。普通的冒号 `:` 得到的是数学关系符，而命令 `\colon` 得到的才是标点符号中的冒号，且间距左小右大，参见 `$a:b$` $a:b$ 与 `$a colon b$` $a\colon b$ 的区别。

命令 `\mathpunct{}` 能把参数作为标点符号处理，并正确设置左无右有的间距。

4. 操作符与上下标：数学操作符的上下标作为行间公式时，将会显示在正下方，而不是右上下方，例如 `\min_i` 会显示为 $\min\limits_i$，`\mathop{}` 命令可以将参数转换为操作符，例如 `x_n^2` x_n^2 与 `\mathop{x}_n^2` $\mathop{x}_n^2$ 的区别

`\limits` 命令用在行内公式中，跟随在任意数学操作符后，可以将它的上下标显示在数学操作符的正上下方，并且保持行内公式的紧凑性，例如:

```
$\int\limits_a^b f(x)\df{}x$
```

$$\int\limits_a^b f(x)\mathrm{d}x$$

5. 堆叠上下标：`\substack{}` 命令需要 `amsmath` 宏包支持，用于将多行符号堆叠为一个上下标，参数中可以使用两个下划线 `\\` 换行，例如：

```
\[
  \lim_{\substack{x \to x_0 \\
                  y \to y_0}} f(x, y)
\]
```

$$\lim_{\substack{x \to x_0 \\ y \to y_0}} f(x, y)$$

6. 定界符：以下展示了一些常用的定界符：

命令	符号	命令	符号	命令	符号	命令	符号
(	(	)	)				
\lbrace or {	{	\rbrace or }	}	\lbrack or [	[	\rbrack or]	]
\lfloor	⌊	\rfloor	⌋	\lceil	⌈	\rceil	⌉
\langle	⟨	\rangle	⟩	\vert or |	|	\Vert or \\|	‖

7. 定界符高度：使用 `\big` 等一系列命令及其与 `r` 和 `l` 的组合可以产生具有不同大小的括号：

```
$$ ( \big( \Big(
    \frac{ax+b}{\ln\,x}
  \bigg) \Bigg) \Biggr) $$
```

$$ (\big(\Big(\frac{ax+b}{\ln\,x} \bigg) \Bigg) \Biggr) $$

使用 `\left`、`\right` 以及 `\middle` 能使定界符自适应公式的高度。`\left` 和 `\right` 必须成对出现以限定范围。单个点号表示的 `\left.` 和 `\right.` 仅用于配对以限定范围，不输出任何符号。

大于号 `<` 和小于号 `>` 也可以组合这些命令得到不同大小的尖括号。

8. 分式的各种表现：基本的 `\frac` 会在不同位置表现出不同的尺寸，`\tfrac` 和 `\dfrac` 分别创建 text 和 display 样式的分式，例如 `\dfrac{1}{1+x}` 在行内也显示为 $\dfrac{1}{1+x}$，这会撑起行高。

9. 连分式：使用 `\cfrac` 创建，以下是分别使用 `\frac`、`\dfrac` 和 `\cfrac` 创建的嵌套分式效果：

$$a_0 + \frac{1}{a_1 + \frac{1}{a_2 + \frac{1}{a_3}}} \qquad a_0 + \cfrac{1}{a_1 + \cfrac{1}{a_2 + \cfrac{1}{a_3}}} \qquad a_0 + \cfrac{1}{a_1 + \cfrac{1}{a_2 + \cfrac{1}{a_3}}}$$

3.4 公式字体与字符

1. 粗体：数学环境中的粗体使用 amsmath 宏包提供的 `\boldsymbol{}` 命令，例如：

```
$\boldsymbol{y} = k \boldsymbol{x}$
```

$\boldsymbol{y} = k \boldsymbol{x}$

2. 正体：数学公式内的字体默认为斜体，使用命令 `\mathrm{}` 可以在公式内创建正体，例如 `$\mathrm{argmin}$` 效果为 argmin

3. 原生数学字体：下表列出了原生的数学字体：

命令	效果
`\mathrm{ABCDabcd1234}`	$\mathrm{ABCDabcd1234}$
`\mathit{ABCDabcd1234}`	$\mathit{ABCDabcd1234}$
`\mathnormal{ABCDabcd1234}`	$\mathnormal{ABCDabcd1234}$
`\mathcal{ABCDabcd1234}`	$\mathcal{ABCD}\dashv\lfloor\rfloor\lceil\infty\in\ni\triangle$

4. 其它数学字体：其它一些宏包支持的数学字体对 `ABCDabcd1234` 的排版效果为：

命令	效果	所需宏包
`\mathbb`	$\mathbb{ABCDabcd1234}$	`amsfonts/amssymb`
`\mathfrak`	$\mathfrak{ABCDabcd1234}$	`amsfonts/amssymb`
`\mathscr`	$\mathscr{ABCD}$	`mathrsfs`

5. 希腊字母：使用对应字母名表示，例如 `\alpha` α，首字母大写表示大写希腊字母，例如 `\Omega` Ω。有些希腊字母前面加上 var 表示花写，例如 `\varphi` φ 相比于 `\phi` ϕ。

6. 正体小写希腊字母：宏包 PKG txfonts 以 up 结尾的命令可以得到正体的小写希腊字母，如 `\thetaup` $\thetaup$ 相较于斜体的 `\theta` θ 。

   宏包 PKG txfonts 可能会改变其它数学符号甚至正文的字体样式，可以替换为宏包 upgreek ，它以 up 开头的命令也可以得到正体小写希腊字母，如 `\uppi` $\uppi$ ，且不会有副作用。宏包 upgreek 有三个互斥的可选项：Euler、Symbol 和 Symbolsmallscale ，分别加载不同的字体或字号。

4 复合公式

4.1 矩阵

1. 基本矩阵：使用 array 环境可以得到表格一样的横竖对齐，可以用于对齐公式或排版矩阵。这些具有对齐的环境一般都使用 `\\` 切换到下一行，同一行内用 `&` 切换到下一列：

```
\[ \begin{array}{ccc}
    x_{11} & ax + b & x_{13} \\
    x_{21} & e^x & x_{23}
\end{array} \]
```

$$\begin{array}{ccc} x_{11} & ax + b & x_{13} \\ x_{21} & e^x & x_{23} \end{array}$$

   额外的参数指明了每一列的水平对齐方式都是居中对齐 (center)，还可以使用 `l` 指定左对齐以及使用 `r` 指定右对齐。

2. 通用矩阵：宏包 PKG amsmath 提供了通用的 `matrix` 矩阵环境，无需手动指明对齐方式，其余用法一致：

```
\[\begin{matrix}
  1 & 0 \\ 0 & 1
\end{matrix}\]
```

$$\begin{matrix} 1 & 0 \\ 0 & 1 \end{matrix}$$

3. 矩阵的边界符：可以用左右高度自适应的定界符为矩阵加上边界符。宏包 PKG amsmath 以 `pbBvV` 开头的各种 `matrix` 环境可以为矩阵两侧加上各种边界符，以下是按顺序展示的各个边界符效果：

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix} \quad \begin{bmatrix} a & b \\ c & d \end{bmatrix} \quad \begin{Bmatrix} a & b \\ c & d \end{Bmatrix} \quad \begin{vmatrix} a & b \\ c & d \end{vmatrix} \quad \begin{Vmatrix} a & b \\ c & d \end{Vmatrix}$$

4. 行内小矩阵：宏包 PKG amsmath 提供了 `smallmatrix` 环境，可以得到行内公式的小矩阵，加上自适应的左右括号后为 $\left(\begin{smallmatrix} a & b \\ c & d \end{smallmatrix}\right)$ 效果。

5. 矩阵的边界符与对齐：宏包 PKG mathtools 提供了各种带星的 `matrix*` 环境，在拥有边界符的同时，可以通过可选参数手动指定列对齐：

```
\[ \begin{pmatrix*}[r]
    150 & -450 \\ 10 & 15
\end{pmatrix*} \]
```

$$\begin{pmatrix*}[r] 150 & -450 \\ 10 & 15 \end{pmatrix*}$$

4.2 并列公式

1. 分段函数与公式并列：使用 `cases` 环境，会自动生成一个比 `\left` 更紧凑的花括号，例如：

```
\[ y=\begin{cases}
    x & (x \le 1) \\
    2x-1 & (x>1)
\end{cases}\]
```

$$y=\begin{cases}x & (x\le 1)\\ 2x-1 & (x>1)\end{cases}$$

cases 环境行列可像矩阵一样对齐，但最多只允许包含两列。

2. 并列公式中的公式样式：cases 环境下的公式默认为 text 样式，使用 PKG mathtools 宏包的 `dcases` 环境可以得到 display 样式的内容。以下是两者的比较：

$$\begin{cases}\int_0^x \frac{\mathrm{d}x}{x+1} & x>0\\ x+1 & x\le 0\end{cases}\qquad \begin{cases}\displaystyle\int_0^x \frac{\mathrm{d}x}{x+1} & x>0\\ x+1 & x\le 0\end{cases}$$

3. 左并列与右括号：使用 PKG mathtools 宏包的 `rcases` 环境可以得到右花括号环境，效果为：

```
\[ \begin{rcases}
    a > 0 \\
    x_1=2,x_2=5
\end{rcases} y=x^2-7x+10 \]
```

$$\begin{rcases}a>0\\ x_1=2,x_2=5\end{rcases}y=x^2-7x+10$$

4. 并列公式和并列文本：使用带星号的 `dcases*` 环境可以使并列环境的第二列条件不是公式而是文本，例如：

```
\[a_n =\begin{dcases*}
    2^n & n is odd\\
    n^2 & n is even
\end{dcases*} \]
```

$$a_n=\begin{cases}2^n & \text{n is odd}\\ n^2 & \text{n is even}\end{cases}$$

4.3 公式与编号

1. 带编号的公式：普通的行间公式不带编号，使用 `equation` 环境可以创建带编号的公式：

```
\begin{equation}
    a^2 + b^2 = c^2
\end{equation}
```

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

equation 环境自带公式环境，可以直接输入公式。

2. 连续编号的公式：PKG amsmath 宏包的 `gather` 环境可以给多行公式编号，环境内使用两个连续的反斜杠 `\\` 换行，例如：

```
\begin{gather}
    (a+b)^2=a^2+2ab+b^2\\
    (a-b)^2=a^2-2ab+b^2
\end{gather}
```

$$(a+b)^2=a^2+2ab+b^2 \tag{2}$$
$$(a-b)^2=a^2-2ab+b^2 \tag{3}$$

3. 多行的行间公式：普通的行间公式无法这样换行，带星号的 `gather*` 环境可以创建多行的行间公式，且均不编号。

4. 公式与子编号：amsmath 宏包使用 `subequations` 环境可以使环境内的公式编号变为子编号，例如：

```
\begin{subequations} \begin{gather}
    \sin^2 x=\frac{1-\cos 2x}{2}\\
    \cos^2 x=\frac{1+\cos 2x}{2}
\end{gather} \end{subequations}
```

$$\sin^2 x = \frac{1-\cos 2x}{2} \tag{4a}$$
$$\cos^2 x = \frac{1+\cos 2x}{2} \tag{4b}$$

5. 自定义编号行为：在行尾使用命令 `\notag` 可以取消该行的编号；在行尾使用命令 `\tag{}` 可以手动指定公式的编号，参数内为编号内容，且该编号会自动添加圆括号；带星号的命令 `\tag*{}` 同样可以手动指定编号，且不会自动添加圆括号。例如：

```
\begin{gather}
    a+b                  \\
    b+c \notag           \\
    c+d \tag{ii}         \\
    d+e \tag*{$\star$}   \\
\end{gather}
```

$$a+b \tag{5}$$
$$b+c$$
$$c+d \tag{ii}$$
$$d+e \tag{$\star$}$$

4.4 对齐公式

1. 公式对齐：使用 amsmath 宏包的 `align` 环境，通过 `&` 符号确定对齐位置：

```
\begin{align} a(b+c)
    &= ab + ac \\
    &= ac + ab \\
    &= a(c+b) \end{align}
```

$$a(b+c) = ab + ac \tag{6}$$
$$= ac + ab \tag{7}$$
$$= a(c+b) \tag{8}$$

`align` 环境实质是奇数列居右、偶数列居左的表格，因此不用像 `array` 环境需要给出列的数目和对齐参数，并且可以通过空列改变列的左右对齐方式。

`align` 环境会给每一行公式编号，带星的 `align*` 环境会取消所有编号。

2. 对齐与空白：amsmath 宏包的 `align` 环境列间会产生较大的空白，换用 `alignat` 环境可以得到紧凑的对齐。该环境需要指定一个数字作为参数，代表右左对齐的对数，计算方法为 ≥ 每行最大的 `&` 数量加 1 后再除以 2 。

使用带星号的 `alignat*` 环境可以不带编号。

例如，以下对齐：

```
\begin{alignat*}{3}
    x+&2&&y=-&5\\
     -& &&y= &7
\end{alignat*}
```

$$x+2y = -5$$
$$-\ y = \ \ 7$$

换用 `align` 环境会产生大量的空白，效果一言难尽，但可以从中看出 `alignat` 环境的参数数为 `3` ：

$$x+2 \quad y=- \quad 5$$
$$- \quad y= \quad 7$$

3. 公式块环境：gather、align 和 alignat 环境必定占据一整行，公式块环境 gathered 、aligned 和 alignated 环境只占公式实际宽度，因此同一行中可以编写其它内容。但是公式块环境需要置于其它数学环境中，且不带任何编号。

4. 对齐与编号：将公式块环境嵌套在 equation 环境中可以使多行公式只有一个居中的编号，例如：

```
\begin{equation} \begin{aligned}
    e^x&=1+x+\frac{1}{2!}x^2+\cdots
            +\frac{1}{n!}x^n+\cdots\\
    &=\sum_{n=0}^{\infty}\frac{x^n}{n!}
\end{aligned} \end{equation}
```

$$\begin{aligned} e^x &= 1 + x + \frac{1}{2!}x^2 + \cdots + \frac{1}{n!}x^n + \cdots \\ &= \sum_{n=0}^{\infty} \frac{x^n}{n!} \end{aligned} \tag{9}$$

5. 左中右对齐：PKG amsmath 宏包使用 `multline` 环境得到第一行左对齐、中间的行居中对齐、最后一行右对齐的公式环境：

```
\begin{multline}
    a_1 + b_1 \\
    a_2 + b_2 \\
    a_3 + b_3 \\
    a_4 + b_4
\end{multline}
```

$$\begin{multline} a_1 + b_1 \\ a_2 + b_2 \\ a_3 + b_3 \\ a_4 + b_4 \end{multline} \tag{10}$$

使用带星号的 `multline*` 环境可以去除最后一行产生的编号。

5 更多符号速查

5.1 基础数学

1. 二元运算符：

命令	符号	命令	符号
\times	$\times$	\div	$\div$
\pm	$\pm$	\mp	$\mp$
\cdot	$\cdot$		
\ast	$\ast$	\star	$\star$
\circ	$\circ$	\bigcirc	$\bigcirc$
\oplus	$\oplus$	\ominus	$\ominus$
\otimes	$\otimes$	\oslash	$\oslash$
\odot	$\odot$	\bullet	$\bullet$
\vee	$\vee$	\wedge	$\wedge$
\bigvee	$\bigvee$	\bigwedge	$\bigwedge$

2. 二元关系运算符：

命令	符号	命令	符号
\le	$\le$	\ge	$\ge$
\ll	$\ll$	\gg	$\gg$
\equiv	$\equiv$	\neq	$\neq$
\sim	$\sim$	\approx	$\approx$
\simeq	$\simeq$	\cong	$\cong$

3. 集合与关系：

命令	符号	命令	符号
\cup	$\cup$	\cap	$\cap$
\bigcup	$\bigcup$	\bigcap	$\bigcap$
\subset	$\subset$	\supset	$\supset$
\subseteq	$\subseteq$	\supseteq	$\supseteq$
\in	$\in$	\ni	$\ni$
\notin	$\notin$	\emptyset	$\emptyset$
\forall	$\forall$	\exists	$\exists$

4. 平面几何：

命令	符号	命令	符号
\angle	$\angle$	^\circ	$^\circ$
\perp	$\perp$	\parallel	$\parallel$

5. 微积分：

命令	符号	命令	符号	命令	符号
\partial	∂	\nabla	∇		
\infty	∞				

6. PKG amssymb 宏包提供的更多符号：

命令	符号	命令	符号
\because	∵	\therefore	∴
\leqslant	⩽	\geqslant	⩾
\nless	≮	\ngtr	≯
\lessdot	⋖	\gtrdot	⋗
\subseteqq	⫅	\supseteqq	⫆
\subsetneqq	⫋	\supsetneqq	⫌

5.2 点与箭头

1. 各种箭头：

命令	符号	命令	符号
\leftarrow	←	\rightarrow	→
\Leftarrow	⇐	\Rightarrow	⇒
\leftrightarrow	↔	\Leftrightarrow	⇔
\longleftarrow	⟵	\longrightarrow	⟶
\Longleftarrow	⟸	\Longrightarrow	⟹
\longleftrightarrow	⟷	\Longleftrightarrow	⟺
\mapsto	↦	\longmapsto	⟼
\nwarrow	↖	\nearrow	↗
\swarrow	↙	\searrow	↘
\leftharpoonup	↼	\rightharpoonup	⇀
\leftharpoondown	↽	\rightharpoondown	⇁
\rightleftharpoons	⇌		

2. amssymb 宏包提供的一些额外的箭头：

命令	符号	命令	符号
\leftrightarrows	⇆	\rightleftarrows	⇄
\nleftarrow	↚	\nrightarrow	↛
\nLeftarrow	⇍	\nRightarrow	⇏
\nleftrightarrow	↮	\nLeftrightarrow	⇎
\dashleftarrow	⇠	\dashrightarrow	⇢
\circlearrowleft	↺	\circlearrowright	↻

3. 各种点：

命令	符号	命令	符号
.	.	\cdot	·
\dots	…	\cdots	⋯
\vdots	⋮	\ddots	⋱

6 进阶内容

6.1 调节空白距离

1. 定长水平空白：以下几个水平空白在公式环境外也能产生相同的空白效果：

```
\, \: \; \! \quad \qquad
```

这些命令置于段首和段尾无效。

2. 自定义水平空白：`\hspace{}` 用于在两个字符间产生高度为零的水平空白，参数为产生空白的水平距离。水平距离可以为负值，表示拉近间距。

该命令置于段首有效，置于段尾无效。如果命令位于换行处（左右字符输出后位于行尾和行首）将失效。

带星号的 `\hspace*{}` 作用基本相同，但它在换行处也可以工作，一定能在其左右字符之间生成空白。

3. 虚位水平空白：`\hphantom{}` 同样在两个字符之间产生空白，但空白宽度为参数中内容的水平宽度。换句话说它为参数的内容“预留水平空白”。该命令置于段首和段尾无效。

4. 弹性水平空白：`\hfill` 命令用于生成弹性水平空白，用于将当前行剩余的空间填满。如果当前行已经充满内容则无效。

`\hfill` 的这些衍生命令可以使用具体的形状而不是空白填充：

命令	效果
\dotfill	用点线填充
\hrulefill	用水平线段填充
\downbracefill	用开口向下的花括号填充
\upbracefill	用开口向上的花括号填充
\leftarrowfill	用向左的箭头填充
\rightarrowfill	用向右的箭头填充

效果示例：

```
chapter1.3 \dotfill filling \\
left \leftarrowfill mid
     \rightarrowfill right
```

chapter1.3 ..... filling
left ←———— mid ————→ right

5. 自定义竖直空白：`\vspace{}` 用于在两段间产生宽度为零的竖直空白，参数为产生空白的竖直距离，负值表示拉近间距。

   该命令置于页首或页尾无效。

   带星号的 `\hspace*{}` 作用基本相同，但置于页首或页尾仍有效，一定能在其上下内容之间生成空白。

6. 定长竖直空白：`\smallskip` 生成一段高度为 3^{+1}_{-1} 的可伸缩的垂直空白，输出时系统会自动选择范围内的一个值。类似地，`\medskip` 生成一段高度为 6^{+2}_{-2} 的可伸缩的垂直空白；`\bigskip` 生成一段高度为 12^{+4}_{-4} 的可伸缩的垂直空白。

7. 虚位竖直空白：`\vphantom{}` 在两段之间产生空白，但空白高度为参数中内容的竖直高度。它为参数的内容“预留竖直空白”。

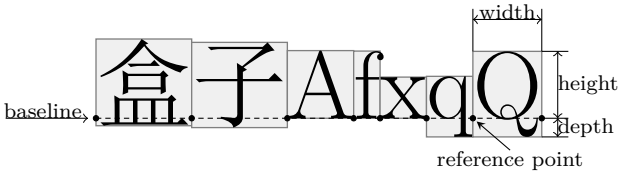
   该命令用在数学环境中，可主动调节 `\left \right` 的自适应大小。

8. 弹性竖直空白：`\vfill` 命令将当前页面剩余的垂直空间填满。该命令置于页首无效。

6.2 盒子

1. 盒子的概念：盒子 (box) 是 LaTeX 排版的基础。所有字符、符号甚至图表均被视为大小不一的矩形盒子。盒子是排版的最小单位，通过计算并调整盒子的位置可以使内容正确排布，从而获得合适的排版效果。

2. 盒子模型：每个盒子都是一个矩形区域，一个盒子模型有如下主要参数：



- 宽度 (width)：盒子的左边线与右边线的距离；
- 基线 (baseline)：基线是盒子竖直对齐的参照，当一系列盒子水平排列时，它们的基线总是对齐；
- 高度 (height)：基线与盒子的上边线的距离；
- 深度 (depth)：基线与盒子的下边线的距离；
- 参照点 (reference point)：基线与盒子的左边线的交点，也是盒子水平对齐的参照。当一系列盒子竖直排列时，它们的参照点总是对齐。

高度与深度之和是盒子的总高度。

3. 生成盒子：`\mbox{}` 用于生成无边框盒子，参数为各种内容，也可以为空。空盒子不占宽度，可用其做水平空白命令的挡板。

4. 生成边框盒子：`\fbox{}` 用于生成边框盒子，如 `\fbox{box}` 表现为 box

5. 盒子的尺寸调整：声明 `\fboxrule=`*size* 和 `\fboxsep=`*size* 用于设定盒子边框线的粗细（默认值是 0.4pt；当尺寸设置为 0 时，则没有边框线）和设定盒子中内容与边框线的距离（即盒子内边距，默认值是 3pt。当尺寸设置为 0 时，则边框线紧贴内容），这两个可以放在导言区，从而对整篇文档起作用。

6. 定宽盒子：`\makebox[][]{}` 用于生成一个固定宽度的无框盒子，第一个可选参数为宽度，第二个可选参数为对齐方式。对齐方式有四种 `lrcs` ，分别表示左对齐、右对齐、居中对齐（默认）和两端对齐。

7. 定宽边框盒子：`\framebox` 命令相当于在 `\makebox` 生成的盒子的四周添加边框，参数用法相同。

8. 虚线盒子：PKG dashbox 宏包提供的 `\dbox{}` 用于生成虚线边框盒子，以及 `\dashbox` 用于生成定宽虚线盒子。如 `\dbox{box}` 表现为 box

   声明 `\dashlength=`*size* 和 `\dashdash=`*size* 分别设置虚线盒子虚部的长度和实部的长度。

9. 盒子的溢出和不足：对于一个宽度或高度固定的盒子，在正常排版时内容应当位于盒子内部，如果内容超出了盒子的范围，会发生内容的溢出 (overfull)。溢出会发生内容的越位，可能造成覆盖或丢失等问题，例如：

```
\framebox[10ex][l]{overfull box} covered some content
```

overfull box covered some content

   如果盒子的宽度太大且内容不足，而盒子又必须被填满，此时就会发生内容的不足 (underfull)。这经常发生于两端对齐的情况，此时内容间就会强行添加额外的空白来填充宽度。

10. 升降盒子：命令 `\raisebox{`*raise*`}[`*height*`][`*depth*`]{`*text*`}` 可以使盒子对象在垂直方向上升或下降。

    必选参数 *raise* 决定了内容在垂直方向上移动的距离；可选参数 *height* 和 *depth* 决定了移动后盒子的实际高度与深度。如果忽略可选参数，移动后盒子的尺寸将自动适应内容；如果指定了可选参数，则盒子的实际尺寸与内容无关，可能造成溢出。

    升降盒子实际上是盒子的嵌套，它用一个新的盒子套住现有内容，让现有内容在其中上下移动，新盒子的尺寸由可选参数决定。例如：

```
\fbox{\raisebox{18pt}{\dbox{box}}}
\fbox{\raisebox{-14pt}{\dbox{box}}}
\fbox{\raisebox{14pt}[18pt]{\dbox{box}}}
\fbox{box}
\fbox{\raisebox{20pt}[16pt]{\dbox{box}}}
\fbox{\raisebox{-18pt}[16pt][10pt]{\dbox{box}}}
```

box box box box box box

11. 零尺寸盒子：当用盒子命令生成了一个零尺寸的盒子以后，将它放在任意位置均不会影响其它内容的排版，且移动的基准点为原先盒子的参考点，可以在不破坏文档布局的情况下调整元素。

```
\raisebox{10pt}[0pt][0pt]{ \hspace{-30pt}
    \makebox[0pt]{zero box} } and ...
```

zero box and following content

12. 行盒子：使用以上命令生成的盒子均是行盒子，它们的特点是只占一行，如果内容宽度超出了盒子或文档宽度也不会自动换行，而是会溢出。但是行盒子的高度和深度会随着内容自动调整。

```
\framebox[6em]{ \dbox{
  This is a {\LARGE long} long
  \raisebox{-10pt}{long} long line }}
```

This is a long long long long line

13. 段落盒子：使用 `\parbox[position][height][inner-pos]{width}{text}` 可以得到段落盒子，段落盒子的内容将根据宽度自动做断行处理。

    必选参数 *width* 决定段落盒子的宽度。可选参数 *position* 决定段落盒子的竖直对齐方式 `tcb`；*height* 可以手动指定内容的高度；*inner-pos* 决定盒子中内容的水平对齐方式 `tcbs` 。例如：

```
\fbox{\LARGE parbox:}
\fbox{\parbox[b]{14em}{a parbox is ...}}
```

parbox: a parbox is a box whose contents are created in paragraph mode

14. 小页环境：使用 `minipage` 可以得到小页环境。小页环境和段落盒子差不多，且参数的用法都基本一致。小页环境更适合包含大量图表等内容时使用。

15. 缩放盒子：需要 PKG graphicx 宏包支持，命令 `\scalebox{h-scale}[v-scale]{text}` 可以缩放一个盒子，水平缩放系数 *h-scale* 为必选参数，竖直缩放系数 *v-scale* 为可选参数。缩放系数设为 1 表示不缩放，设为负值表示反向缩放，例如：

```
\makebox[0pt][l]{mirror$\Omega$}%
\scalebox{1}[-1]{mirror$\Omega$}
```

mirrorΩ

16. 变形盒子：需要 PKG graphicx 宏包支持，命令 `\resizebox{h-length}{v-length}{text}` 和带星号的命令 `\resizebox*{h-length}{v-length}{text}` 可以变形盒子。必选参数 *h-scale* 代表变形后的宽度；另一个必选参数 *v-scale* 对普通的 `\resizebox` 来说代表变形后的高度，对带星号的 `\resizebox*` 来说代表变形后连同深度的总高度。

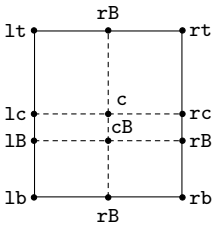
    在变形参数内可以使用单个感叹号 `!` 表示保持原有宽高不变，也可以使用负值按相反方向缩放。

17. 旋转盒子：需要 PKG graphicx 宏包支持，命令 `\rotatebox[origin=...,x=...,y=...,units=...]{angle}{text}` 可以旋转一个盒子，必选参数 *angle* 指定旋转角度，顺时针为负。

    可选参数中的 *origin* 决定旋转中心，可用值如左图所示。

    `tcBb` 分别位于盒子顶部、中心线、基线和底部。

    也可以换用可选参数中的 *x,y*，根据盒子参考点通过坐标的方式确定旋转中心。



    可选参数中的 *units* 确定旋转角的单位，默认为角度。设置 `units=-360` 表示以度为单位且旋转正向为顺时针；设置 `units=6.283185` 表示以弧度为单位且旋转正向为逆时针等。

```
$\pi$ \rotatebox{45}{$\pi$} \rotatebox{90}{$\pi$}
\makebox[0pt]{\rotatebox[origin=c]{30}{\dag}}\,%
\makebox[0pt]{\rotatebox[origin=c]{-30}{\dag}}
```

π π π †

18. 花哨盒子：PKG fancybox 宏包提供了一系列带有花哨边框盒子的命令，包括 `\shadowbox{}` 生成 阴影盒子，`\doublebox{}` 生成 双线盒子，`\ovalbox{}` 生成 圆角盒子，`\Ovalbox{}` 生成 粗圆角盒子。

    圆角盒子可以使用声明 `\cornersize{}` 和 `\cornersize*{}` 修改圆角直径，前者的参数是相对盒子宽或高的比例，后者的参数是绝对长度。例如，声明 `\cornersize{1}` 会使后续的圆角盒子变成 半圆盒子。

19. 矩形块：`\rule[]{}{}` 命令可以创建一个填充的矩形块一样的符号，两个必选参数是矩形块的宽和高，可选参数是矩形块竖直升降的距离。

    例如，`\rule{10pt}{12pt}` 创建的矩形块为 ■，`\rule[-2pt]{3em}{0.5pt}` 创建的矩形块为 _____

6.3 颜色

1. 简单的文本颜色：xcolor 宏包提供了创建颜色的工具。`\color{}` 是一个声明，它能使块中接下来所有文本（包括公式）变成参数的颜色，例如声明 `\color{blue}` 会使接下来所有文本变为蓝色。

2. 页面颜色：使用 `\pagecolor{}` 会更改当前及其之后页面的颜色，在某一页使用 `\nopagecolor` 可以暂时去除该页面施加的颜色效果。

   这是一个使用 `\pagecolor{black}` 和 `\color{white}` 的页面局部效果。

3. 颜色盒子：`\fcolorbox{}{}{}` 可以创建一个颜色盒子，为文本添加带有颜色的边框和背景，三个参数分别是边框颜色、背景色和文本内容。

   这是一段具有 blue 边框、yellow 背景的文本。

4. 局部颜色：使用 `\textcolor{}{}` 可以创建局部的颜色，两个参数分别为颜色名和影响的文本。它常用于公式环境中，例如：

```
\[ \lim_{\textcolor{red}{\Delta}\to 0}
(1+\textcolor{red}{\Delta})^\frac
{1}{\textcolor{red}{\Delta}} =e\]
```

$$\lim_{\Delta\to 0}(1+\Delta)^{\frac{1}{\Delta}}=e$$

5. 颜色名:可以直接使用颜色名指代颜色，xcolor 宏包可以直接使用的颜色名约有 20 个。使用宏包的 `[dvipsnames]` 参数可以额外使用约 70 个颜色名，例如 BrickRed 和 Periwinkle 这样的颜色名。

   进一步使用宏包的 `[x11names]` 参数可以加载 300 多种颜色名，例如 Aquamarine3 和 Goldenrod2 这样的颜色名。

6. 自定义颜色：在导言区使用 `\definecolor{}{}{}` 可以自定义颜色，三个参数分别是自定义的颜色名、颜色格式、颜色值。颜色值的形式取决于颜色格式，可用的颜色格式有：

   - `RGB`：红绿蓝混色，每种色值取值范围为 0–255，用逗号分隔
   - `rgb`：同上，但每种色值取值范围为 0–1.0
   - `cmyk`：彩印标准颜色，每种色值取值范围为 0–1.0
   - `gray`：灰度颜色，取值范围为 0–1.0
   - `HTML`：六位十六进制 RGB 颜色值
   - `wave`：对应波长的颜色，取值为 363–814

   对于如下定义：

```
\definecolor{FrozenBlue}{RGB}{157,234,242}
\definecolor{CandleRed}{HTML}{F57267}
```

   这是 FrozenBlue 和 CandleRed 的呈现效果。

   如果 `\definecolor` 定义的颜色名和现有颜色名重复，会覆盖之前定义的颜色名。`\providecolor` 命令用法类似，但发现重复定义后会放弃新的定义。

7. 自定义混合颜色：`\colorlet{}{}` 提供了通过混合颜色来定义颜色的一种形式，被混合颜色和混合比例均以感叹号 `!` 结尾，例如：

```
\colorlet{PurplePink}{red!60!blue!40!}
```

   会使用 60% 红色和 40% 蓝色混合出 PurplePink 颜色。

7 图表

7.1 图片

1. LaTeX 中的图片：使用插图有两种途径，一是插入外部绘制的图片，二是使用 LaTeX 代码直接在文档中画图。

2. 插入图片：需要 graphicx 宏包支持，使用 `\includegraphics` 命令插图。可选参数调整图片属性，必选参数指明图形的路径：

```
Hello, \includegraphics
[height=0.5cm]{./LaTeX.png}
```

Hello, 

支持的图形格式包括 PDF、PNG、JPG、EPS 等。

插入的图形就是一个有内容的矩形盒子，在正文中排版效果和一个很大的字符类似。

3. 图片环境：除了一些很小的标志图形，很少把插图直接夹在文字中。通常把图形放在一个可以变动相对位置的浮动环境中，使用 `figure` 环境：

```
\begin{figure}[ht]
    \centering
    \includegraphics{./LaTeX.png}
    \caption{\LaTeX symbol}
\end{figure}
```

图 1: LaTeXsymbol

`figure` 环境有可选参数 `[ht]` ，表示浮动体可以出现在环境周围的文本所在处 (here) 和一页的顶部 (top)。`figure` 环境内部相当于默认没有缩进的段落。

`\caption` 命令给插图加上自动编号和标题。

7.2 表格

1. 基本表格结构：表格一般都直接用 LaTeX 命令绘制。制作表格需要确定表格的行、列对齐模式和表格线，由 `tabular` 环境完成：

```
\begin{tabular}{|rrr|}
    \hline
        angle & $\sin$ & $\cos$ \\
    \hline
            0 &   0   & 1 \\
    $\pi/2$ &   1   & 0 \\
    \hline
\end{tabular}
```

angle	sin	cos
0	0	1
$\pi/2$	1	0

`tabular` 环境有一个参数，声明了表格中列的模式：`|rrr|` 表示表格有三列，都是右对齐 (right)，在第一列前面和第三列后面各有一条垂直的表格线。在 `tabular` 环境内部，行与行之间用换行命令 `\\` 隔开，每行内部的表项则用符号 `&` 隔开。表格中的横线则是用命令 `\hline` 产生的。

2. 表格的对齐：基本的对齐参数 `lcr` 表示左中右对齐，表格宽度会自动调整，但不会自动换行，可能超出页面宽度。可以用 `p{}` 形式指定某一列的宽度，参数内为宽度值，这时自动左对齐。

可以用可选参数表示文本垂直对齐方式，`tcb` 分别表示顶端对齐、垂直居中、底端对齐，默认顶端对齐。

3. 表格线：在必选参数内连续使用两个竖线 `||` 表示绘制一条竖直双线，使用 `@{}` 在每行对应绘制的竖直表格线替换为参数里的符号，参数为空表示不画竖直表格线。

   命令 `\hline` 用于绘制一根水平表线，而命令 `\cline{c1-c2}` 仅绘制从 c1 列到 c2 列的水平表线。这两个命令后面不用加换行符，因为它算当前行的顶部。两个连续的 `\hline` 可以画水平双线，但是这种方式制造的双线与竖直表线的相交效果不好。

4. 竖向表线与拆分单元格：在单元格内使用 `\vline` 命令仅画出等于所在行行高的竖直线。该命令常常用于横向拆分单元格。

5. 表格的高级对齐：加载 PKG array 宏包可以使表格使用更多对齐效果，例如 `m{}` 在指定某列宽度时，也会使该行的其余部分垂直居中；`b{}` 在指定某列宽度时，也会使该行的其余部分底端对齐。例如：

align	this column uses p{}	align	this column uses m{}	align	this column uses b{}

6. 列格式：PKG array 宏包还提供了 `>{}` 和 `<{}`，分别用在 lcrpmb 参数前后，可以使该列每个单元格开头、结尾都套上参数内的命令或声明。例如以下表格参数：

```
{|>{\bfseries}c|
  >{\raggedleft\arraybackslash}p{6cm}|}
```

ohhh	It's a very very very long loong looong looooong looooong line

   使第一列单元格使用加粗字系，第二列单元格左边不平齐（就是右对齐）。其中 `\arraybackslash` 是为避免行末的 \\ 出现异常。

7. 自定义列格式：可以在导言区通过自定义列格式命令 `\newcolumntype{}{}` 将复杂的 lcrpmb 与 >{} 和 <{} 组合成一个格式，第一个参数是新的格式名（必须是单字母），第二个参数是组合格式。例如，在导言区定义：

```
\newcolumntype{f}{>{\(}c<{\)}}
```

   则表格环境的参数中为 `f` 的列处于行内数学环境 \(... \) 并居中。

8. 小数对齐的列格式：宏包 PKG siunitx 提供了小数对齐的列格式，在导言区使用以下命令可以保留两位小数点：

```
\sisetup{
  round-mode=places,
  round-precision=2,
}
```

   然后便可以使用 `S` 格式表示该列需要对齐小数点，效果为：

```
\begin{tabular}{|c|S|}
    \hline
    A & 1.234 \\
    B & 12.34 \\
    C & 123.4 \\
    D & 12 \\ \hline
\end{tabular}
```

A	1.23
B	12.34
C	123.40
D	12

9. 表格与浮动：`tabular` 环境得到的也是一个比较大的盒子。一般也放在浮动环境 `table` 中，参数与大体的使用格式也与 figure 环境类似：

```
\begin{table}[H]
  % tabular
\end{table}
```

可选参数 `[H]` 表示不浮动 (Here)。该选项是由 PKG float 宏包提供的特殊功能。

10. 跨行列的单元格：需要 PKG multirow 宏包支持，分别使用 `\multirow{}{}{}` 和 `\multicolumn{}{}{}` 命令跨行和跨列。三个参数中：

    - 第一个参数表示跨越（总共包含）的行列数；
    - 第二个参数同 `tabular` 环境的参数（对齐和画线），也可以使用 `*` 表示自动适应宽度；
    - 最后一个参数是单元格内填入的文本。

    跨行列后，被跨位置的单元格仍应正常表示且内容留空，它们会被覆盖以实现跨行列。

```
\begin{tabular}{cc}
  \hline
  \multirow{2}{*}{multirow} & row1-2 \\
                            & row2-2 \\
  \hline
\end{tabular}
```

multirow	row1-2
	row2-2

同时跨行跨列注意嵌套关系 `\multirow` 命令放在 `\multicolumn` 内部。

一个非常复杂的综合示例：

```
\begin{tabular}{|c|c|c|c|c|}
  \hline
  \multirow{2}{*}{multi-row} &
  \multicolumn{2}{c|}{multi-col} &
  \multicolumn{2}{c|}{
    \multirow{2}{*}{multi-row\&col}}\\
  \cline{2-3}
    & col-1 & col-2 &
    \multicolumn{2}{c|}{} \\
  \hline
  item1&item2&item3&item4&item5\\
  \hline
\end{tabular}
```

multi-row	multi-col		multi-row&col	
	col-1	col-2		
item1	item2	item3	item4	item5

`\cline{2-3}` 用于在 multi-col 下画一条第 2 栏位到第 3 栏位的边框线。空的 `\multicolumn` 用于修改同时跨行列单元格的边框线问题。

11. 拆分单元格：拆分单元格的需求可以通过表格嵌套实现。

12. 单元格换行：宏包 PKG makecell 提供了在单元格内换行方式，使用 `\makecell{}` 命令可以在参数内使用 \\ 方便地换行：

```
\begin{tabular}{|c|c|c|}
    \hline col1 &
    \makecell{line1 \\ line2} &
    col3 \\ \hline
\end{tabular}
```

col1	row1 row2	col3

还可以配合可选参数使用 `[lcrtb]` 之一指定单元格对齐方式。

13. 水平表线宽：PKG makecell 宏包还提供了 `\Xhline{}` 和 `\Xcline{}{}` 命令，可以通过最后一个额外的参数指定水平表线宽。以下是一个模仿三线表的示例：

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

X	Y	
	Left	Right
a	A	C
b	B	D

14. 分割表头：PKG diagbox 宏包提供了分割表头的命令 `\diagbox` 。该命令支持两个或三个参数，分别表示将表头分割成两部分或三部分，例如：

```
\begin{tabular}{c|c}
  \hline \diagbox{X}{data}{Y} & $y$ \\
  \hline $x$ & 10 \\ \hline
\end{tabular}
```

data / Y X	y
x	10