

标题页

Ryan* Fan†

2020 年 5 月 4 日

摘要

一般用于紧跟\maketitle 命令之后介绍文档的摘要
中文 L^AT_EX 排版。

1 用 L^AT_EX 排版文字

分段

换行\\

冒号 “please press the ‘x’ key.”

连字符-用来组成复合词,

短破折号-用来连接数字表示范围,

长破折号—用来连接单词

省略号...和...

波浪号~

强调文字, 但是无法换行, ulem 宏包解决了这一问题, 它提供的 uline 命令能够轻松生成自动换行的下划线。emph 命令用来将文字变为斜体以示强调。如果在本身已经用 emph 命令强调的文字内部嵌套使用 emph 命令, 内部则使用直立体文字。

在合适的位置插入一个不会断行的空格 Fig. 1, Ryan Fan

断行

可以带可选参数 *length*, 用于在换行处向下增加垂直间距
或者 newline 命令, 不用带参数

*注脚

†注脚

断页，在在双栏排版中只起到另起一栏的作用

断词 I think this is: supercalifragilisticexpialidocious. And I think
this is: supercalifragilisticexpialidocious.

目录

1 用 L^AT_EX 排版文字	1
2 目录和页眉页脚	4
2.1 子章节	4
2.1.1 子子章节	4
标题不带编号	4
 第一部分 分块	 4
3 交叉引用	4
4 脚注和边注	4
5 特殊环境	5
5.1 列表	5
5.2 对齐环境	5
5.3 引用环境	6
5.4 代码环境	6
5.5 表格	6
5.5.1 列表格	6
5.5.2 横线	7
5.5.3 合并单元格	8
5.5.4 嵌套表格	8
5.5.5 行距控制	8
5.6 图片	9
5.7 盒子	10
5.7.1 水平盒子	10
5.7.2 带框的水平盒子	10
5.7.3 垂直盒子	10
5.7.4 标尺盒子	11
5.8 浮动体	11
5.8.1 浮动体的标题	12
5.8.2 并排和子图表	12
6 数学公式	13
6.1 公式排版基础	13
6.1.1 行内和行间公式	13
6.2 数学模式	14
6.3 数学符号	14
6.3.1 一般符号	14
6.3.2 指数、上下标和导数	16

6.3.3	分式和根式	16
6.3.4	关系符	17
6.3.5	算符	18
6.3.6	巨算符	19
6.3.7	数学重音和上下括号	20
6.3.8	箭头	21
6.3.9	括号和定界符	22
6.4	多行公式	22
A	附录	23

2 章节

2.1 子章节

2.1.1 子子章节

段落

子段落

标题不带编号

第一部分 分块

3 交叉引用

A reference to this subsection looks like: “see section 3 on page 4.”

4 脚注和边注

“天地玄黄，宇宙洪荒。日月盈昃，辰宿列张。”¹

有些情况下 (比如在表格环境、各种盒子内) 使用 footnote 并不能正确生成脚注。我们以分两步进行, 先使用 footnotemark 为脚注计数, 再在合适的位置用 footnotetext 生成脚注。²

边注较窄, 不要写过多文字, 最好设置较小的字号。

¹出自《千字文》。

²出自《千字文》。

5 特殊环境

5.1 列表

有序列表

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

无序列表

- An item.
 - A nested item.
 - + A ‘plus’ item. + A ‘plus’ item.
 - Another item. –Another item.
- Go back to upper level.

关键字环境

Enumerate Numbered list.

Itemize Non-numbered list.

重定义无序列表的符号

- † First item
 - ‡ Subitem
 - ‡ Subitem
- † Second item

重定义有序列表的符号

- A> First item
- B> Second item

5.2 对齐环境

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

Centered text using a `center` environment.

Left-aligned text using a `flushleft` environment.

Right-aligned text using a `flushright` environment.

还可以用以下命令直接改变文字的对齐方式: Centered text paragraph.

Left-aligned text paragraph.

Right-aligned text paragraph.

`center` 等环境会在上下文产生一个额外间距, 而 `\centering` 等命令不产生, 只是改变对齐方式。

比如在浮动体环境 `table` 或 `figure` 内实现居中对齐, 用 `\centering` 命令即可, 没必要再用 `center` 环境。

5.3 引用环境

quote 用于引用较短的文字, 首行不缩进

Francis Bacon says:

Knowledge is power.

quotation 用于引用若干段文字, 首行缩进

《木兰诗》:

万里赴戎机, 关山度若飞。朔气传金柝, 寒光照铁衣。将军百战死, 壮士十年归。

归来见天子, 天子坐明堂。策勋十二转, 赏赐百千强。

.....

5.4 代码环境

```
#include <iostream>
int main()
{
    std::cout << "Hello, world!"
                << std::endl;
    return 0;
}
```

```
for(i=0; i<4; i++)
    printf("Number %d\n", i);
```

要排版简短的代码或关键字 `\verb <delim><code><delim>`

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

`\LaTeX`

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

5.5 表格

5.5.1 列表格

`tabular` 环境使用 `<column - spec>` 参数指定表格的列数以及每列的格式。

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

@ 格式可在单元格前后插入任意的文本, 但同时它也消除了单元格前后额外添加的间距。

1:1 one

11:3 eleven

格式参数重复

one	two	three	four	five	Hello!	Hello!
1	2	3	4	5	L ^A T _E X	L ^A T _E X
					hello!	L ^A T _E X

辅助格式 > 和 <, 用于给列格式前后加上修饰命令

*italic** normal

*column** column

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

\centering 等对齐命令会破坏表格环

境里 \\换行命令的定

义, \arraybackslash 用来恢复之。如

果不加 \arraybackslash 命令, 也可

以用 \tabularnewline 命令代替原来

的 \\实现表格换行。

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

A	B	C	D
a	b	c	d

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

A	B	C	D
a	b	c	d

5.5.2 横线

\cline{<i>j</i>} 用来绘制跨越部分单元格的横线:

4	9	2
3	5	7
8	1	6

三线表由 booktabs 宏包支持, 它提供了 \toprule、\midrule 和 \bottomrule 命令用以排版三线表的三条线, 以及和 \cline 对应的 \cmidrule。除此之外, 最好不要用其它横线以及竖线:

Numbers			
	1	2	3
Alphabet	A	B	C
Roman	I	II	III

5.5.3 合并单元格

横向合并单元格较为容易, 由

`\multicolumn{<n>}{<column-spec>}{<item>}` 命令实现:

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

形如 `\multicolumn{1}{<column-spec>}{<item>}` 的命令可以用来修改某一个单元格的列格式。

1	2	Center
3		Right
4	C	

纵向合并单元格需要用到 `multirow` 宏包提供的

`\multirow{<n>}{<width>}{<item>}` 命令:

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

Item	Value	
	First	Second
A	1	2

5.5.4 嵌套表格

在单元格中嵌套一个小表格可以起到“拆分单元格”的效果。

注意要用 `\multicolumn` 命令配合 `@{}` 格式把单元格的额外边距去掉, 使得嵌套的表格线能和外层的表格线正确相连:

a	b	c
a	e	f
	e	f
a	b	c

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

a	d1
	d2
b	c

5.5.5 行距控制

\LaTeX 生成的表格看起来通常比较紧凑。修改参数 `\arraystretch` 可以得到行距更加宽松的表格:

Really loose
tabular rows.

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

Head lines
tabular lines
tabular lines

5.6 图片

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

使用 `\includegraphics[options]{filename}` 命令加载图片了:

其中 `filename` 为图片文件名, 文件名有时需要使用相对路径或绝对路径。

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

`\includegraphics` 命令的可选参数 `options` 支持 `key=value` 形式赋值, 常用的参数如下:

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





5.7 盒子

5.7.1 水平盒子

`\mbox{...}`

`\makebox[⟨width⟩][⟨align⟩]{...}`

`\mbox` 生成一个基本的水平盒子, 内容只有一行, 不允许分段 (除非嵌套其它盒子) `\makebox` 更进一步, 可以加上可选参数用于控制盒子的宽度 `⟨width⟩`, 以及内容的对齐方式 `⟨align⟩`, 可选居中 `c`(默认值)、左对齐 `l`、右对齐 `r` 和分散对齐 `s`

|Test some words.|

| Test some words. |

|Test some words. |

| Test some words. |

|Test some words. |

5.7.2 带框的水平盒子

`\fbox` 和 `\framebox` 让我们可以为水平盒子添加边框。

Test some words.

Test some words.

可以通过 `\setlength` 命令调节边框的宽度 `\fboxrule` 和内边距 `\fboxsep`:

Test box.

Test box.

5.7.3 垂直盒子

排版一个文字可以换行的盒子:

```

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

```

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

天地玄黄
宇宙洪荒

三字经: 人之初 千字文:

性本善

性相近

习相远

如果在 `minipage` 里使用 `\footnote` 命令, 生成的脚注会出现在盒子底部, 编号是独立的, 并且使用小写字母编号。而在 `\parbox` 里无法正常使用 `\footnote` 命令, 只能在盒子里使用 `\footnotemark`, 在盒子外使用 `\footnotetext`。

这是一个垂直盒子的测试。³

这是一个垂直盒子的测试。^a

^a脚脚来自 `minipage`。

5.7.4 标尺盒子

`\rule [⟨raise⟩]{⟨width⟩}{⟨height⟩}` 命令用来画一个实心的矩形盒子, 也可适当调整以用来画线 (标尺):

Black  box.

Upper  and lower  box.

A `_` line.

5.8 浮动体

\LaTeX 预定义了两类浮动体环境 `figure` 和 `table`。习惯上 `figure` 里放图片, `table` 里放表格, 但并没有严格限制, 可以在任何一个浮动体里放置文字、公式、表格、图片等等任意内容。

```

\begin{table}[⟨placement⟩]

```

...

```

\end{table}

```

$[⟨placement⟩]$ 参数提供了一些符号用来表示浮动体允许排版的位置, 如

`hbp` 允许浮动体排版在当前位置、底部或者单独成页。 `table` 和 `figure` 浮动

³脚脚来自 `fbox`

体的默认设置为 tbp。

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

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

`\clearpage` 命令会在另起一页之前, 先将所有推迟处理的浮动体排版成页, 此时 htpb 等位置限制被完全忽略。

float 宏包为浮动体提供了 H 位置参数, 不与 htpb 及! 混用。使用 H 位置参数时, 会取消浮动机制, 将浮动体视为一般的盒子插入当前位置。

5.8.1 浮动体的标题

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

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

可通过修改 `\figurename` 和 `\tablename` 的内容来修改标题的前缀。标题样式的定制功能由 caption 宏包提供。

table 和 figure 两种浮动体分别有各自的生成目录的命令:

`\listoftables`

`\listoffigures`

5.8.2 并列和子图表



图 1: 图片标题

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



图 2: 并排图 1



图 3: 并排图 2

给每个图片定义小标题时, 就要用到 `subfig` 宏包的功能



(a)



(b)

图 4: 使用 `subfig` 宏包的 `\subfloat` 命令排版子图。

6 数学公式

`amsmath` 宏包, 对多行公式的排版提供了有力的支持, `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。

6.1 公式排版基础

6.1.1 行内和行间公式

行内公式由一对 `$` 符号包裹:

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

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 `\qref` 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。

The Pythagorean theorem is:

$$a^2 + b^2 = c^2 \quad (1)$$

Equation (1) is called ‘Gougu theorem’ in Chinese.

It’s wrong to say

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

or

$$1 + 1 = 4$$

直接使用不带编号的行间公式:

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

For short:

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

Or if you like the long one:

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

行内公式和行间公式的对比:

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

In display:

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

6.2 数学模式

1. 数学模式中输入的空格被忽略。
2. 不允许有空行 (分段)。
3. 所有的字母被当作数学公式中的变量处理, 想在数学公式中输入正体的文本, 可以用 `\mathrm` 或者 `amsmath` 提供的 `\text` 命令。

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

6.3 数学符号

6.3.1 一般符号

表 1: 文本/数学模式通用符号

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

表 2: 希腊字母

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

表 3: 其它符号

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

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

$$a_1 + a_2 + \dots + a_n$$

6.3.2 指数、上下标和导数

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

$$p_{ij}^3 \quad m_{\text{Kunth}} \quad \sum_{k=1}^3 k$$

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

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

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

6.3.3 分式和根式

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

In display style:

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

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

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

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

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

Pascal's rule is

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

6.3.4 关系符

表 4: 二元关系符

<	<	>	>	=	=
≡	\equiv	≤	\leq or \le	≥	\geq or \ge
≪	\ll	≫	\gg	≐	\doteq
⋈	\prec	⋉	\succ	≈	\sim
⋊	\preceq	⋋	\succeq	≈	\simeq
⊂	\subset	⊃	\supset	≈	\approx
⊆	\subseteq	⊇	\supseteq	≅	\cong
⊆	\sqsubseteq	⊇	\sqsupseteq	⋈	\bowtie
∈	\in	∋	\ni , \owns	∝	\propto
⊢	\vdash	⊣	\dashv	⊨	\models
	\mid	∥	\parallel	⊥	\perp
⌣	\smile	⌢	\frown	⋈	\asymp
:	:	∉	\notin	≠	\neq or \ne

以下命令依赖 latexsym 宏包

⊆	\sqsubset	⊇	\sqsupset	⋈	\Join
---	-----------	---	-----------	---	-------

以下命令依赖 amssymb 宏包

⋈	\lessdot	⋉	\ltrdot	≐	\doteqdot
≍	\leqslant	≎	\geqslant	≐	\risingdotseq
≍	\eqslantless	≎	\eqslantgtr	≐	\fallingdotseq
≍	\leqq	≍	\geqq	≐	\eqcirc
≐	\circeq	≍	\ggg	≍	\lll or \llless
≈	\lesssim	≈	\gtrsim	△	\triangle
≈	\lessapprox	≈	\gtrapprox	≐	\bumpeq
≍	\lessgtr	≍	\gtrless	≐	\Bumpeq
≍	\lesseqgtr	≍	\gtreqless	≈	\thicksim
≍	\lesseqqgtr	≍	\gtreqqlless	≈	\thickapprox
≈	\preccurlyeq	≈	\succcurlyeq	≈	\approxeq
≈	\curlyeqprec	≈	\curlyeqsucc	≈	\backsimeq
≈	\precsim	≈	\succsim	≈	\backsimeq
≈	\precapprox	≈	\succapprox	⊨	\vDash
≈	\subseteqq	≈	\supseteqq	⊨	\Vdash
≈	\shortparallel	≈	\Supset	⊨	\Vvdash
▲	\blacktriangleleft	□	\sqsupset	∋	\backepsilon
▷	\vartriangleright	∴	\because	∝	\varpropto
▶	\blacktriangleright	⊆	\Subset	∅	\between
▷	\trianglerighteq	⌢	\smallfrown	⌢	\pitchfork
△	\vartriangleleft		\shortmid	⌣	\smallsmile
△	\trianglelefteq	∴	\therefore	⊆	\sqsubset

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

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

6.3.5 算符

表 5: 二元运算符

+	+	-	-	±	<code>\pm</code>
≠	<code>\mp</code>	◁	<code>\triangleleft</code>	·	<code>\cdot</code>
÷	<code>\div</code>	▷	<code>\triangleright</code>	×	<code>\times</code>
\	<code>\setminus</code>	*	<code>\star</code>	∪	<code>\cup</code>
∩	<code>\cap</code>	*	<code>\ast</code>	⊔	<code>\sqcup</code>
∩	<code>\sqcap</code>	○	<code>\circ</code>	∨	<code>\vee</code>
∨	<code>\lor</code>	∧	<code>\wedge</code>	∧	<code>\land</code>
•	<code>\bullet</code>	⊕	<code>\oplus</code>	⊖	<code>\ominus</code>
◇	<code>\diamond</code>	⊙	<code>\odot</code>	⊘	<code>\oslash</code>
⊕	<code>\uplus</code>	⊗	<code>\otimes</code>	○	<code>\bigcirc</code>
∏	<code>\amalg</code>	△	<code>\bigtriangleup</code>	▽	<code>\bigtriangledown</code>
†	<code>\dagger</code>	‡	<code>\ddagger</code>	‡	<code>\wr</code>
以下命令依赖 <code>latexsym</code> 宏包					
◁	<code>\lhd</code>	▷	<code>\rhd</code>		
◁	<code>\unlhd</code>	▷	<code>\unrhd</code>		
以下命令依赖 <code>amssymb</code> 宏包					
+	<code>\dotplus</code>	•	<code>\centerdot</code>		
×	<code>\ltimes</code>	×	<code>\rtimes</code>	※	<code>\divideontimes</code>
∪	<code>\doublecup</code>	∩	<code>\doublecap</code>	∖	<code>\smallsetminus</code>
∨	<code>\veebar</code>	∧	<code>\barwedge</code>	∧	<code>\doublebarwedge</code>
⊕	<code>\boxplus</code>	⊖	<code>\boxminus</code>	⊖	<code>\circleddash</code>
⊗	<code>\boxtimes</code>	⊙	<code>\boxdot</code>	⊙	<code>\circledcirc</code>
⋈	<code>\intercal</code>	⊕	<code>\circledast</code>	⋈	<code>\rightthreetimes</code>
∨	<code>\curlyvee</code>	∧	<code>\curlywedge</code>	∧	<code>\listoffigures</code>

表 6: L^AT_EX 作为算符的函数名称

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

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

对于求模表达式, L^AT_EX 提供了 `\pmod` 和 `\bmod` 命令:

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

amsmath 允许用户在导言区用 `\DeclareMathOperator` 在定义自己的算符, 其中带星号的命令定义带上下限的算符:

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

6.3.6 巨算符

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

In display:

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

巨算符的上下标位置可由 `\limits` 和 `\nolimits` 控制

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

In display:

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

amsmath 宏包还提供了 `\substack`, 能够在下限位置书写多行表达式:

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

subarray 环境更进一步, 令多行表达式可选择居中 (c) 或左对齐 (l):

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

6.3.7 数学重音和上下括号

表 7: 数学重音符号

\hat{a}	<code>\hat{a}</code>	\check{a}	<code>\check{a}</code>	\tilde{a}	<code>\tilde{a}</code>
\acute{a}	<code>\acute{a}</code>	\grave{a}	<code>\grave{a}</code>	\breve{a}	<code>\breve{a}</code>
\bar{a}	<code>\bar{a}</code>	\vec{a}	<code>\vec{a}</code>	\mathring{a}	<code>\mathring{a}</code>
\dot{a}	<code>\dot{a}</code>	\ddot{a}	<code>\ddot{a}</code>		
\widehat{AAA}	<code>\widehat{AAA}</code>	\widetilde{AAA}	<code>\widetilde{AAA}</code>		
以下命令依赖 <code>amsmath</code> 宏包					
\ddot{a}	<code>\ddot{a}</code>	\dddot{a}	<code>\dddot{a}</code>		
以下命令依赖 <code>yhmath</code> 宏包					
\wideparen{AAA}	<code>\wideparen{AAA}</code>				

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:

$$\begin{aligned} \bar{x}_0 & \quad \bar{x}_0 \\ \vec{x}_0 & \quad \vec{x}_0 \\ \hat{e}_x & \quad \hat{e}_x \end{aligned}$$

L^AT_EX 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。

表 8: 作为重音的箭头符号

\overrightarrow{AB}	<code>\overrightarrow{AB}</code>	\overleftarrow{AB}	<code>\overleftarrow{AB}</code>
以下命令依赖 <code>amsmath</code> 宏包			
\underleftarrow{AB}	<code>\underleftarrow{AB}</code>	\underrightarrow{AB}	<code>\underrightarrow{AB}</code>
\overleftrightarrow{AB}	<code>\overleftrightarrow{AB}</code>	\underleftrightarrow{AB}	<code>\underleftrightarrow{AB}</code>

$$\begin{aligned} 0.\bar{3} & = \underline{\underline{1/3}} \\ \hat{XY} & \quad \widehat{XY} \end{aligned}$$

$$\overrightarrow{AB} \quad \overleftarrow{AB}$$

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

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

6.3.8 箭头

表 9: 箭头

\leftarrow	<code>\leftarrow</code> or <code>\gets</code>	\longleftarrow	<code>\longleftarrow</code>
\rightarrow	<code>\rightarrow</code> or <code>\to</code>	\longrightarrow	<code>\longrightarrow</code>
\leftrightarrow	<code>\leftrightarrow</code>	\longleftrightarrow	<code>\longleftrightarrow</code>
\Leftarrow	<code>\Leftarrow</code>	\Lleftarrow	<code>\Lleftarrow</code>
\Rightarrow	<code>\Rightarrow</code>	\Rrightarrow	<code>\Rrightarrow</code>
\Leftrightarrow	<code>\Leftrightarrow</code>	\Leftrightarrow	<code>\Leftrightarrow</code>
\mapsto	<code>\mapsto</code>	\longmapsto	<code>\longmapsto</code>
\hookrightarrow	<code>\hookrightarrow</code>	\hookrightarrow	<code>\hookrightarrow</code>
\lhookrightarrow	<code>\lhookrightarrow</code>	\lhookrightarrow	<code>\lhookrightarrow</code>
\leftharpoonup	<code>\leftharpoonup</code>	\rightharpoonup	<code>\rightharpoonup</code>
\leftharpoondown	<code>\leftharpoondown</code>	\rightharpoondown	<code>\rightharpoondown</code>
\rightleftharpoons	<code>\rightleftharpoons</code>	\iff	<code>\iff</code>
\uparrow	<code>\uparrow</code>	\downarrow	<code>\downarrow</code>
\Uparrow	<code>\Uparrow</code>	\Updownarrow	<code>\Updownarrow</code>
\Downarrow	<code>\Downarrow</code>	\Downarrow	<code>\Downarrow</code>
\nearrow	<code>\nearrow</code>	\searrow	<code>\searrow</code>
\swarrow	<code>\swarrow</code>	\nwarrow	<code>\nwarrow</code>

以下命令依赖 `latexsym` 宏包

\leadsto `\leadsto`

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

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

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

6.3.9 括号和定界符

表 10: 定界符

(())	↑	\uparrow
[\lbrack or [] \rbrack or]	↓	\downarrow
{ \lbrace or \{ } \rbrace or \}	↕	\updownarrow
< \langle > \rangle	↕	\Updownarrow
\or	⇓	\Downarrow
/ / \ \backslash	⇕	\Uparrow
⌊ \lfloor ⌋ \rfloor	⌈	\rceil
⌈ \lceil		

表 11: 用于行间公式的大定界符

{ \lgroup	\arrowvert
} \rgroup	\Arrowvert
{ \lmoustache	! \bracevert
} \rmoustache	

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

使用 `\left` 和 `\right` 命令可令括号 (定界符) 的大小可变, 在行间公式中常用。L^AT_EX 会自动根据括号内的公式大小决定定界符大小。

`\left` 和 `\right` 必须成对使用。需要使用单个定界符时, 另一个定界符写成 `\left.` 或 `\right.`

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

还可以用 `\big`、`\bigg` 等命令生成固定大小的定界符。

更常用的形式是类似 `\left` 的 `\bigl`、`\biggl` 等, 以及类似 `\right` 的 `\bigr`、`\biggr` 等 (`\bigl` 和 `\bigr` 不必成对出现)。

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

$$\left(\left(\left(\left(\right) \right) \right) \right) \quad \left| \right| \left| \right| \left| \right| \left| \right| \quad \left\downarrow \left\downarrow \left\downarrow \left\downarrow \left\downarrow \right\downarrow$$

6.4 多行公式

A 附录