

第五章 数学公式

数学是 $\text{T}_\text{E}\text{X}$ 的灵魂。就是由于排版数学公式是那么得复杂，在通常的打字机上根本无法进行，Donald Knuth 才开发这个文本格式化系统。另一方面， $\text{L}^{\text{A}}\text{T}_\text{E}\text{X}$ 的灵魂是文档设计。不但如此，所有 $\text{T}_\text{E}\text{X}$ 的强大数学排版功能 $\text{L}^{\text{A}}\text{T}_\text{E}\text{X}$ 也都具备，而且提供了相当好的组合。我们在本章所讲的绝大部分（并不是全部）内容都适用于 $\text{T}_\text{E}\text{X}$ ，因此要把它们两者区分开，有时是很困难的。我们因此经常指出命令和环境是属于 $\text{L}^{\text{A}}\text{T}_\text{E}\text{X}$ 的，即使它也同样适用于 $\text{T}_\text{E}\text{X}$ 。

数学公式是通过输入特殊的描述性文本来生成的。这就意味着必须告诉 $\text{L}^{\text{A}}\text{T}_\text{E}\text{X}$ 要把下面的文本解释成一个数学公式，而且也能告诉它数学公式已结束，返回正常的文本状态。数学文本的处理是通过切换进入数学模式（1.5.3 节）实现的。数学环境就是为了这个目标而引进的。

§5.1 数学环境

数学公式可以出现在一个文本行中，如 $(a+b)^2 = a^2 + 2ab + b^2$ ，也可以与正文分开，如

$$\int_0^\infty f(x) dx \approx \sum_{i=1}^n w_i e^{x_i} f(x_i)$$

这两种形式分别被称为 正文公式 和 显示公式。

正文公式或者方程是用如下环境生成的：

`\begin{math}` 公式文本 `\end{math}`

由于正文公式通常都很短，有的时候只有一个字符，因此也可以用一个更方便的版本，即 `\(公式文本 \)`。如果还嫌它长，那可以用更短的形式 `$ 公式文本 $`。所有这三种形式是等价的，虽然内部具有某些差别，如 `\(` 是脆弱的，而 `$` 就比较牢靠。

公式的内容 公式文本 是由数学构造组成，我们将在下面几节中介绍这些构造。

显示公式或方程是用下面的环境生成的：

`\begin{displaymath}` 公式文本 `\end{displaymath}`

`\begin{equation}` 公式文本 `\end{equation}`

这两种环境的差别在于 `equation` 环境会自动给公式加上一个顺序的公式编号。`displaymath` 环境也可以用方便形式 `\[公式文本 \]` 给出。

在默认方式下，显示公式是水平居中的，而且如果有公式编号的话，编号会显示在右页边。通过选择文档类选项 `fleqn`（3.1.1 节），公式就会左对齐，而且可以具有一个能调整的缩进。这个选项在整篇文档中有效，而缩进量可以用命令 `\setlength{\mathindent}{缩进量}` 来改变，这里的 缩进量 就

是所要定义的长度。除此之外，文档类选项 `leqno` 会使得整篇文档中公式编号显示在左边界。

最后提一下，可以用如下环境创建多行公式：

```
\begin{eqnarray} 公式文本 \end{eqnarray}
```

```
\begin{eqnarray*} 公式文本 \end{eqnarray*}
```

这里的标准形式会给每行公式都加上一个顺序的公式编号，而 `*`- 形式则没有公式编号。

§5.2 数学公式的主要组成

§5.2.1 常量与变量

出现在公式中的数字称为常量，而简单变量只由一个字母表示。在绝大多数的数学排版中是用罗马字样显示常量，用斜体显示变量。 \LaTeX 在数学模式中也是自动遵守这个规则。在源文本中为了使作者容易读而加的空格都被忽略。在常量、变量和类似于 $+$, $-$, $=$ 这样的运算符之间的距离是由 \LaTeX 自动处理的。例如 `$z=2a+3y$` 和 `z = 2 a + 3 y $` 都生成 $z = 2a + 3y$ 。

在键盘上存在对应字符的数学符号有：

$+ - = < > / : ! ' | [] ()$

它们都可以直接用在数学公式中。大括号 $\{ \}$ 用来表示公式的逻辑组合，因此不能作为可直接显示的字符。为了在公式中显示大括号，就必须如通常文本中那样用命令 `\{` 和 `\}`。

$$M(s) < M(t) < |M| = m \quad \$M(s) < M(t) < |M| = m$$$

$$y'' = c\{f[y', y(x)] + g(x)\} \quad \$y'' = c\{f[y', y(x)] + g(x)\}$$$

准备工作：创建一个新的 \LaTeX 文件，名称为 `math.tex`，它只是由 `\documentclass{article}`，`\begin{document}` 和 `\end{document}` 命令组成。

练习 5.1: 用你自己的练习文件生成下面的文本：‘The derivative of the indirect function $f[g(x)]$ is $\{f[g(x)]\}' = f'[g(x)]g'(x)$. For the second derivate of the product of $f(x)$ and $g(x)$ one has $[f(x)g(x)]'' = f''(x)g(x) + 2f'(x)g'(x) + f(x)g''(x)$.’

注意：高阶导数是用多个 $'$ 符号生成的：`y'''` 的结果为 y''' 。

§5.2.2 指数和指标

在数学公式中经常可以见到指数和指标，即把字符相对于公式的主要文本所在行进行提升或下降，并以小号字体显示出来。虽然它们的数学意义可能不同，上标和下标在印刷上分别与指数和指标是完全一样的。甚至指数本

身还可以有指数或指标，依此类推。它们只是提升或下降操作的多次运用而已。

L^AT_EX和 T_EX以一种简单的方式，提供了以适当字体尺寸创建任意指数和指标组合的方法：字符命令 `^` 把紧接下来的字符做为指数（提升），而字符命令 `_` 把紧接下来的字符做为指标（下降）。

$$x^2 \quad x^{\text{2}} \quad a_n \quad a_{\text{n}} \quad x_i^n \quad x^{\text{n}}_{\text{i}}$$

当指数和指标一起出现时，它们的顺序是无关紧要的。上面最后那个例子也可以用 `x_i^n` 来生成。

如果指数或指标的内容不只一个字符，那么就必须用大括号 `{ }` 把这组符号包围起来：

$$x^{2n} \quad x^{\{2n\}} \quad x_{2y} \quad x_{\{2y\}} \quad A_{i,j,k}^{-n!2} \quad A_{\{i,j,k\}}^{\{-n!2\}}$$

也可以在指数和指标中多次进行提升或下降操作：

$$x^{y^2} \quad x^{\{y^2\}} \quad x^{y_1} \quad x^{\{y_1\}} \\ A_{j_{n,m}^{x_i^2}} \quad A^{\{x_i^2\}}_{\{j^{2n}\}_{\{n,m\}}}$$

注意：提升和下降命令 `^` 和 `_` 只能用在数学模式中。

§5.2.3 分数

比较小的分数，尤其是正文公式中的分数最好用斜杠字符 `/` 表示，如 `$(n+m)/2$` 表示 $(n+m)/2$ 。对于较复杂的分数，命令

`\frac{分子}{分母}`

可以用来把 `分子` 放在 `分母` 上面，而且中间有一适当长度的水平线。

$$\frac{1}{x+y} \quad \backslash[\backslashfrac{1}{x+y} \backslash] \\ \frac{a^2-b^2}{a+b} = a-b \quad \backslash[\backslashfrac{a^2-b^2}{a+b} = a-b \backslash]$$

分数也可以嵌套至任何层次：

$$\frac{\frac{a}{x-y} + \frac{b}{x+y}}{1 + \frac{a-b}{a+b}} \quad \backslash[\backslashfrac{\backslashfrac{a}{x-y} + \backslashfrac{b}{x+y}}{1 + \backslashfrac{a-b}{a+b}} \backslash]$$

L^AT_EX把分数中的分数有较小的字样显示出来。在 5.5.2 节中介绍了当 L^AT_EX的选择不很好时，如何修改这种自动字样尺寸。

§5.2.4 方根

方根是用如下命令显示的：

`\sqrt[开方数]{参数}`

例如：`$\sqrt[3]{8}=2$` 生成 $\sqrt[3]{8}=2$ 。如果不写可省参数 `开方数`，就会生成平方根：`\sqrt{a}` 得到 \sqrt{a} 。

方根符号的尺寸和长度是自动与 `参数` 大小匹配的：

$$\sqrt{x^2+y^2+2xy} = x+y \quad \sqrt{x^2+y^2+2xy} = x+y, \text{ 或者}$$

$$\sqrt[n]{\frac{x^n - y^n}{1 + u^{2n}}} \quad \backslash[\ \sqrt[n]{\frac{x^n - y^n}{1 + u^{2n}}} \backslash]$$

方根也可以嵌套至任何层次:

$$\sqrt[3]{-q + \sqrt{q^2 + p^3}} \quad \backslash[\ \sqrt[3]{-q + \sqrt{q^2 + p^3}} \backslash]$$

§5.2.5 求和与积分

求和与积分符号是用命令 `\sum` 与 `\int` 生成的, 根据其所在的是正文公式还是显示公式, 它们可以有两个不同的大小。

求和与积分通常都有上下限。这可以用指数和指标命令 `^` 和 `-` 来得到。上下限的位置也要看其所在的是正文公式还是显示公式而定。

在一个正文公式中 `\sum_{i=1}^n` 和 `\int_a^b` 的结果是 $\sum_{i=1}^n$ 和 \int_a^b , 而在显示公式中它们的形状如下面左边所示:

$$\sum_{i=1}^n \int_a^b \quad \text{而有些作者喜欢把积分的上下限也放在积分号的上方和下方, 就如同求和符号中那样。这可以在积分符号后面紧接 } \backslashlimits \text{ 命令来得到: } \int_{x=0}^{x=1}$$

在求和与积分符号前后的其它公式文本会与它们适当对齐的。

$$2 \sum_{i=1}^n a_i \int_a^b f_i(x) g_i(x) dx \quad \backslash[\ 2 \sum_{i=1}^n a_i \int_a^b f_i(x) g_i(x) \backslash, dx \backslash]$$

在类似于 $\int y dx$ 和 $\int f(z) dz$ 这样的积分中, 微分运算符 dx 和 dz 应与前面的被积函数之间有很小的空档。只是输入一个空格, 并不能实现这个目标, 因为在数学模式中空格都被忽略: `\int y dx` 的结果为 $\int y dx$ 。我们可以用在 3.5.1 节中提到的小间距命令 `\,` 来做到这一点。因此 `\int y \,, dx` 和 `\int f(z) \,, dz` 会得到所希望的结果 $\int y dx$ 和 $\int f(z) dz$ 。在 5.5.1 节中给出了更多的可用于数学公式中的间距命令。

§5.2.6 连续点 — 省略号

公式中有时会包含一串点 \dots , 表示 等等。若只是简单地在一行中输入三个句号会得到不是想要的结果: \dots , 即点靠得太近了。因此 \LaTeX 提供了几条命令:

`\ldots` ... 偏下的点 `\cdots` ... 中间点

`\vdots` \vdots 竖直点 `\ddots` \ddots 对角点

以生成正确间距的点。最好地说明前两条命令差别的例子是: a_0, a_1, \dots, a_n 和 $a_0 + a_1 + \dots + a_n$, 它们分别是用 `a_0, a_1, \ldots, a_n` 和 `$a_0 + a_1 + \cdots + a_n$` 生成的。

`\ldots` 命令也可以用在普通的文本模式中, 而其余三条命令则只能用在数学模式中。在文本模式中, `\dots` 命令可以代替 `\ldots`, 它们的作用一

样。

练习 5.2: 生成下面的结果:

The reduced cubic equation $y^3 + 3py + 2q = 0$ has one real and two complex solution when $D = q^2 + p^3 > 0$. These are given by Cardan's formula as

$$y_1 = u + v, \quad y_2 = -\frac{u+v}{2} + \frac{i}{2}\sqrt{3}(u-v), \quad y_3 = -\frac{u+v}{2} - \frac{i}{2}\sqrt{3}(u-v)$$

where

$$u = \sqrt[3]{-q + \sqrt{q^2 + p^3}}, \quad v = \sqrt[3]{-q - \sqrt{q^2 + p^3}}$$

注意: 在显示公式中不同部分之间的空档是用空档命令 `\quad` 和 `\qquad` 得到的。

练习 5.3: 选择文档类选项 `fleqn`, 并把定义 `\setlength{\mathindent}{2cm}` 放在导言中。重新排版上面的三个 y 公式, 每个都利用 `equation` 环境单独做为一个显示公式, 而不是用这里的 `displaymath` 或 `\[...\]` 括号。

练习 5.4: 生成下列的文本:

Each of the measurements $x_1 < x_2 < \cdots < x_r$ occurs p_1, p_2, \dots, p_r times. The mean value and standard deviation are then

$$x = \frac{1}{n} \sum_{i=1}^r p_i x_i, \quad s = \sqrt{\frac{1}{n} \sum_{i=1}^r p_i (x_i - x)^2}$$

where $n = p_1 + p_2 + \cdots + p_r$.

练习 5.5: 虽然下面这个公式看起来非常复杂, 但得到它不会有任何困难:

$$\int \frac{\sqrt{(ax+b)^3}}{x} dx = \frac{2\sqrt{(ax+b)^3}}{3} + 2b\sqrt{ax+b} + b^2 \int \frac{dx}{x\sqrt{ax+b}}$$

同样生成公式 $\int_{-1}^8 (dx/\sqrt[3]{x}) = \frac{3}{2}(8^{2/3} + 1^{2/3}) = 15/2$ 。

§5.3 数学符号

在数学文本中有相当多的符号, 其中只有很少一部分可以直接从键盘上输入得到。L^AT_EX 提供了通常使用的几乎所有可以想像得到的数学符号。可以用符号名称前缀命令字符 `\` 来得到。而它们的名称就是来自于其数学含义。

§5.3.1 希腊字母

小写字母

α	<code>\alpha</code>	θ	<code>\theta</code>	o	<code>o</code>	τ	<code>\tau</code>
β	<code>\beta</code>	ϑ	<code>\vartheta</code>	π	<code>\pi</code>	v	<code>\upsilon</code>
γ	<code>\gamma</code>	ι	<code>\iota</code>	ϖ	<code>\varpi</code>	ϕ	<code>\phi</code>
δ	<code>\delta</code>	κ	<code>\kappa</code>	ρ	<code>\rho</code>	φ	<code>\varphi</code>
ϵ	<code>\epsilon</code>	λ	<code>\lambda</code>	ϱ	<code>\varrho</code>	χ	<code>\chi</code>
ε	<code>\varepsilon</code>	μ	<code>\mu</code>	σ	<code>\sigma</code>	ψ	<code>\psi</code>
ζ	<code>\zeta</code>	ν	<code>\nu</code>	ς	<code>\varsigma</code>	ω	<code>\omega</code>
η	<code>\eta</code>	ξ	<code>\xi</code>				

大写字母

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>		

要得到希腊字母，只需要在字母名称前面加上命令字符`\`就可以了。大写字母的区别在于名称的第一个字母是大写的。没有列在上面清单的希腊字母一定是与某一个拉丁字母一样。例如，大写的 ρ 与拉丁字母的P一样，因此就没有特殊符号。

在数学公式中的大写希腊字母通常用的是罗马（直立）字样。如果需要斜体字样，可以用数学字体样式命令`\mathnormal`^[2ε]来得到：

`\mathnormal{\Gamma\Pi\Phi}` 生成 $\Gamma\Pi\Phi$ 。

（这条命令取代了用在 \LaTeX 2.09中的数学字样声明`\mit`^[2.09]，原来的用法是`\mit \Gamma\Pi\Phi`。）

希腊字母只能用在数学模式中。如果要用在普通文本中，那么必须把命令用`$...$`括起来。

§5.3.2 花体字母

在数学公式中也可以用26个花体字母：

$\mathcal{A}, \mathcal{B}, \mathcal{C}, \mathcal{D}, \mathcal{E}, \mathcal{F}, \mathcal{G}, \mathcal{H}, \mathcal{I}, \mathcal{J}, \mathcal{K}, \mathcal{L}, \mathcal{M}, \mathcal{N}, \mathcal{O}, \mathcal{P}, \mathcal{Q}, \mathcal{R}, \mathcal{S}, \mathcal{T}, \mathcal{U}, \mathcal{V}, \mathcal{W}, \mathcal{X}, \mathcal{Y}, \mathcal{Z}$

调用方法是用数学字体样式命令`\mathcal`^[2ε]：

`\mathcal{A,B,C,...,Z}`

（也可以用等价的声明`\cal`^[2.09]）。

§5.3.3 二元运算符

数学中通常把两个量用一个符号组合起来，生成一个新的量，这个符号称为二元运算符。可以用做二元运算的符号有：

\pm	<code>\pm</code>	\cap	<code>\cap</code>	\circ	<code>\circ</code>	\bigcirc	<code>\bigcirc</code>
\mp	<code>\mp</code>	\cup	<code>\cup</code>	\bullet	<code>\bullet</code>	\square	<code>\Box</code>
\times	<code>\times</code>	\uplus	<code>\uplus</code>	\diamond	<code>\diamond</code>	\Diamond	<code>\Diamond</code>
\div	<code>\div</code>	\sqcap	<code>\sqcap</code>	\triangleleft	<code>\lhd</code>	\triangleup	<code>\bigtriangleup</code>
\cdot	<code>\cdot</code>	\sqcup	<code>\sqcup</code>	\triangleright	<code>\rhd</code>	\triangledown	<code>\bigtriangledown</code>
$*$	<code>\ast</code>	\vee	<code>\vee</code>	\triangleleft	<code>\unlhd</code>	\triangleleft	<code>\triangleleft</code>
\star	<code>\star</code>	\wedge	<code>\wedge</code>	\triangleright	<code>\unrhd</code>	\triangleright	<code>\triangleright</code>
\dagger	<code>\dagger</code>	\oplus	<code>\oplus</code>	\oslash	<code>\oslash</code>	\setminus	<code>\setminus</code>
\ddagger	<code>\ddagger</code>	\ominus	<code>\ominus</code>	\odot	<code>\odot</code>	\wr	<code>\wr</code>
\amalg	<code>\amalg</code>	\otimes	<code>\otimes</code>				

注意：上面以及后面列表中名称加下划线的符号只能用在 \LaTeX 2_ϵ 中，而且需要调用软件包 `latexsym` (8.8.3 节)。

§5.3.4 关系运算符及其否定

当要比较两个数学量时，就要用关系运算符把它们连接起来。用于各种比较中不同类型关系运算符有：

\leq	<code>\le</code>	\leq	<code>\leq</code>	\geq	<code>\ge</code>	\geq	<code>\geq</code>	\neq	<code>\neq</code>	\sim	<code>\sim</code>
\ll	<code>\ll</code>	\gg	<code>\gg</code>	\doteq	<code>\doteq</code>	\simeq	<code>\simeq</code>				
\subset	<code>\subset</code>	\supset	<code>\supset</code>	\approx	<code>\approx</code>	\asymp	<code>\asymp</code>				
\subseteq	<code>\subseteq</code>	\supseteq	<code>\supseteq</code>	\cong	<code>\cong</code>	\smile	<code>\smile</code>				
\sqsubset	<code>\sqsubset</code>	\sqsupset	<code>\sqsupset</code>	\equiv	<code>\equiv</code>	\frown	<code>\frown</code>				
\sqsubseteq	<code>\sqsubseteq</code>	\sqsupseteq	<code>\sqsupseteq</code>	\propto	<code>\propto</code>	\bowtie	<code>\bowtie</code>				
\in	<code>\in</code>	\ni	<code>\ni</code>	\prec	<code>\prec</code>	\succ	<code>\succ</code>				
\vdash	<code>\vdash</code>	\dashv	<code>\dashv</code>	\preceq	<code>\preceq</code>	\succeq	<code>\succeq</code>				
\models	<code>\models</code>	\perp	<code>\perp</code>	\parallel	<code>\parallel</code>	\mid	<code>\mid</code>				

在上面的符号中有几个可以不只用一个名称调用。例如， \leq 可以用 `\le` 或 `\leq` 生成。

而关系运算符的相反或者否定在数学中是用一个斜杠贯穿符号而得到的：如 $=$ 和 \neq 表示等于和不等于。而 \neq 有一个特殊的命令 `\neq`。但是我们可以上面的符号名称前面用 `\not`，使斜线贯穿该符号。因此 `\not\in` 得到 \notin 。这同样也适用于键盘字符，如 `\not=`, `\not>` 和 `\not<` 的结果为 \neq , \nless 和 \ngtr 。

这样下面的符号可以如此否定。注意最后两个符号 `\not\in` 和 `\notin` 并不是一样的： \notin 和 \notin 。后者比前者更好看些。

\nless	<code>\not<</code>	\ngtr	<code>\not></code>	\neq	<code>\not=</code>
\nleq	<code>\not\le</code>	\ngeq	<code>\not\ge</code>	\nequiv	<code>\not\equiv</code>
\nprec	<code>\not\prec</code>	\nsucc	<code>\not\succ</code>	\nsim	<code>\not\sim</code>
\npreceq	<code>\not\preceq</code>	\nsucceq	<code>\not\succeq</code>	\nsimeq	<code>\not\simeq</code>
\nsubset	<code>\not\subset</code>	\nsupset	<code>\not\supset</code>	\napprox	<code>\not\approx</code>
\nsubseteq	<code>\not\subseteq</code>	\nsupseteq	<code>\not\supseteq</code>	\ncong	<code>\not\cong</code>
\nsubsetseq	<code>\not\subsetseq</code>	\nsupseteq	<code>\not\supseteq</code>	\nasymp	<code>\not\asymp</code>
\notin	<code>\notin</code>	\notin	<code>\notin</code>		

§5.3.5 箭头与指针

在数学文稿中通常会有箭头符号，它也称为指针。可用的箭头符号有下面这些：

\leftarrow	<code>\leftarrow</code>	\gets	<code>\gets</code>	\longleftarrow	<code>\longleftarrow</code>	\uparrow	<code>\uparrow</code>
\Leftarrow	<code>\Leftarrow</code>	\Longleftarrow	<code>\Longleftarrow</code>	\Uparrow	<code>\Uparrow</code>		
\rightarrow	<code>\rightarrow</code>	\to	<code>\to</code>	\longrightarrow	<code>\longrightarrow</code>	\downarrow	<code>\downarrow</code>
\Rightarrow	<code>\Rightarrow</code>	\Longrightarrow	<code>\Longrightarrow</code>	\Downarrow	<code>\Downarrow</code>		
\leftrightarrow	<code>\leftrightarrow</code>	\longleftrightarrow	<code>\longleftrightarrow</code>	\updownarrow	<code>\updownarrow</code>		
\Leftrightarrow	<code>\Leftrightarrow</code>	\Longleftrightarrow	<code>\Longleftrightarrow</code>	\Updownarrow	<code>\Updownarrow</code>		
\mapsto	<code>\mapsto</code>	\longmapsto	<code>\longmapsto</code>	\nearrow	<code>\nearrow</code>		
\hookleftarrow	<code>\hookleftarrow</code>	\hookrightarrow	<code>\hookrightarrow</code>	\searrow	<code>\searrow</code>		
\leftharpoonup	<code>\leftharpoonup</code>	\rightharpoonup	<code>\rightharpoonup</code>	\swarrow	<code>\swarrow</code>		
\leftharpoondown	<code>\leftharpoondown</code>	\rightharpoondown	<code>\rightharpoondown</code>	\nwarrow	<code>\nwarrow</code>		
\rightleftharpoons	<code>\rightleftharpoons</code>	\leadsto	<code>\leadsto</code>				

这里符号 \rightarrow 和 \leftarrow 也可用名称 `\to` 和 `\gets` 来调用。而 `\Longleftrightarrow` 命令也可以用 `\iff` 来代替，但是后者 (\iff) 与前者 (\Longleftrightarrow) 相比，在两边要多一点空档。

§5.3.6 其它各类符号

在 \LaTeX 中无疑包括了在数学文本中所有可能出现的符号。然而，不但如此， \LaTeX 还提供了下面这些符号。（这里与前面的表格有些重复，之所以如此，就是为了使相关符号列在一起。）

\aleph	<code>\aleph</code>	\prime	<code>\prime</code>	\forall	<code>\forall</code>	\Box	<code>\Box</code>
\hbar	<code>\hbar</code>	\emptyset	<code>\emptyset</code>	\exists	<code>\exists</code>	\Diamond	<code>\Diamond</code>
\imath	<code>\imath</code>	∇	<code>\nabla</code>	\neg	<code>\neg</code>	\triangle	<code>\triangle</code>
\jmath	<code>\jmath</code>	\surd	<code>\surd</code>	\flat	<code>\flat</code>	\clubsuit	<code>\clubsuit</code>
ℓ	<code>\ell</code>	∂	<code>\partial</code>	\natural	<code>\natural</code>	\diamondsuit	<code>\diamondsuit</code>
\wp	<code>\wp</code>	\top	<code>\top</code>	\sharp	<code>\sharp</code>	\heartsuit	<code>\heartsuit</code>
\Re	<code>\Re</code>	\bot	<code>\bot</code>	\parallel	<code>\parallel</code>	\spadesuit	<code>\spadesuit</code>
\Im	<code>\Im</code>	\vdash	<code>\vdash</code>	\angle	<code>\angle</code>	\Join	<code>\Join</code>
\mho	<code>\mho</code>	\dashv	<code>\dashv</code>	\backslash	<code>\backslash</code>	∞	<code>\infty</code>

§5.3.7 具有两种尺寸的符号

根据所处为正文公式还是显示公式，下列符号会以不同的大小显示出来：

\sum	\sum	<code>\sum</code>	\cap	\cap	<code>\bigcap</code>	\odot	\odot	<code>\bigodot</code>
\int	\int	<code>\int</code>	\cup	\cup	<code>\bigcup</code>	\otimes	\otimes	<code>\bigotimes</code>
\oint	\oint	<code>\oint</code>	\sqcup	\sqcup	<code>\bigsqcup</code>	\oplus	\oplus	<code>\bigoplus</code>
\prod	\prod	<code>\prod</code>	\vee	\vee	<code>\bigvee</code>	\oplus	\oplus	<code>\biguplus</code>
\coprod	\coprod	<code>\coprod</code>	\wedge	\wedge	<code>\bigwedge</code>			

我们在 5.2.5 节中已介绍了符号 `\sum` 和 `\int`。在那里我们演示了如何给这两个符号加上下限；同样，用移位命令 `^` 和 `_` 也可以给上面所有符号加上下限。有些符号的上下限位置会视所处为正文公式还是显示公式而发生变化。正如在 5.2.5 节中指出的那样，如果上下限只是放在符号旁边时，可以用命令 `\limits` 可强迫上下限放在符号的上方和下方。类似地，当上下限的标准位置是上方和下方时，可以用相反命令 `\nolimits` 使得上下限只是位于符号的旁边。

$$\int_0^\infty \oint_0^\infty \quad \backslash [\ointint^{\infty}_0 \ointint\limits^{\infty}_0 \backslash]$$

$$\prod_{\nu=0}^n \prod_{\nu=0}^n \quad \backslash [\prod^{\infty}_{\nu=0} \prod\limits^{\infty}_{\nu=0} \backslash]$$

§5.3.8 函数名

在数学公式中普遍使用的标准是把用斜体显示变量，而用罗马字体显示函数名。如果我们在数学模式中只是简单地写出函数名 *sin* 或 *log*， \LaTeX 就会认为它们是变量 *s i n* 和 *l o g*，从而显示为 *sin* 和 *log*。为了告诉 \LaTeX 我们需要的是一个函数名，那就需要在函数名前面加上命令字符 `\`。 \LaTeX 接受下面这些函数：

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

在这些函数中有几个也可以在显示时带上（下）限。这只要在函数名后面接指标命令就可以了：如 `\lim_{x \rightarrow \infty}` 在正文模式中是 $\lim_{x \rightarrow \infty}$ ，而在显示模式中是 $\lim_{x \rightarrow \infty}$ 。

下面这些函数名可以用指标命令 `_` 加上一个下限：

<code>\det</code>	<code>\gcd</code>	<code>\inf</code>	<code>\lim</code>	<code>\liminf</code>	<code>\limsup</code>	<code>\max</code>	<code>\min</code>
-------------------	-------------------	-------------------	-------------------	----------------------	----------------------	-------------------	-------------------

`\Pr` `\sup`

最后要提一点, 函数命令 `\bmod` 和 `\pmod{参数}` 都生成函数 mod , 但却是两种形式:

`$ a \bmod b $` $a \bmod b$ 或者
`$ y \pmod{a+b} $` $y \pmod{a+b}$ 。

§5.3.9 数学重音

在数学模式中可以用下面这些数学重音:

`\hat{a}` `\breve{a}` `\grave{a}` `\bar{a}`
`\check{a}` `\acute{a}` `\tilde{a}` `\vec{a}`
`\dot{a}` `\ddot{a}` `\{a\}`

当要给字母 i 和 j 要重音时, 应该去掉它们的点。为此, 需要用符号 `\imath` 和 `\jmath` 代替直接字母输入, 如

`$\vec{\imath} + \tilde{\jmath}$`: $\vec{i} + \tilde{j}$

对于 `\hat` 和 `\tilde`, 还存在一种宽的版本, 名称为分别是 `\widehat` 和 `\widetilde`。这两种符号可以被放在一个公式上:

`\widehat{1-x} = -\widehat{y}` `$\widehat{1-x}=\widehat{-y}$`
`\widetilde{xyz}` `\widetilde{xyz}`

练习 5.6: 两个集合 \mathcal{A} 和 \mathcal{B} 的并就是所有至少在其中一个集合中的元素全体, 并 $\mathcal{A} \cup \mathcal{B}$ 表示。这种操作是可交换的, 即 $\mathcal{A} \cup \mathcal{B} = \mathcal{B} \cup \mathcal{A}$, 也是可结合的, 即 $(\mathcal{A} \cup \mathcal{B}) \cup \mathcal{C} = \mathcal{A} \cup (\mathcal{B} \cup \mathcal{C})$ 。如果 $\mathcal{A} \subseteq \mathcal{B}$, 那么 $\mathcal{A} \cup \mathcal{B} = \mathcal{B}$ 。而且有 $\mathcal{A} \cup \mathcal{A} = \mathcal{A}$, $\mathcal{A} \cup \{\emptyset\} = \mathcal{A}$ 和 $\mathcal{I} \cup \mathcal{A} = \mathcal{I}$ 。

练习 5.7: 应用 l'Hôpital 法则, 我们有:

$$\lim_{x \rightarrow 0} \frac{\ln \sin \pi x}{\ln \sin x} = \lim_{x \rightarrow 0} \frac{\frac{\cos \pi x}{\sin \pi x}}{\frac{\cos x}{\sin x}} = \lim_{x \rightarrow 0} \frac{\pi \tan x}{\tan \pi x} = \lim_{x \rightarrow 0} \frac{\pi / \cos^2 x}{\pi / \cos^2 \pi x} = \lim_{x \rightarrow 0} \frac{\cos^2 \pi x}{\cos^2 x} = 1$$

练习 5.8: Gamma 函数 $\Gamma(x)$ 的定义为

$$\Gamma(x) \equiv \lim_{n \rightarrow \infty} \prod_{\nu=0}^{n-1} \frac{n! n^{x-1}}{x + \nu} = \lim_{n \rightarrow \infty} \frac{n! n^{x-1}}{x(x+1)(x+2) \cdots (x+n-1)} \equiv \int_0^\infty e^{-t} t^{x-1} dt$$

这里的积分只有当 $x > 0$ 时才有意义 (第二 Euler 积分)。

练习 5.9: 从练习 5.3 中的文档类选项中去掉 `fleqn` 选项, 重新得到输出。

练习 5.10:

$$\alpha \vec{x} = \vec{x} \alpha, \quad \alpha \beta \vec{x} = \beta \alpha \vec{x}, \quad (\alpha + \beta) \vec{x} = \alpha \vec{x} + \beta \vec{x}, \quad \alpha(\vec{x} + \vec{y}) = \alpha \vec{x} + \alpha \vec{y},$$

$$\vec{x} \vec{y} = \vec{y} \vec{x}, \text{ 但是 } \vec{x} \times \vec{y} = -\vec{y} \times \vec{x}, \vec{x} \perp \vec{y} \text{ 时 } \vec{x} \vec{y} = 0, \vec{x} \parallel \vec{y} \text{ 时 } \vec{x} \times \vec{y} = 0。$$

练习 5.11: 生成下节中的公式 5.1 和 5.2。

§5.4 其它要素

在前几节中描述的数学要素对于构造下面这样复杂的数学公式已是绰绰有余了：

$$\lim_{x \rightarrow 0} \frac{\sqrt{1+x}-1}{x} = \lim_{x \rightarrow 0} \frac{(\sqrt{1+x}-1)(\sqrt{1+x}+1)}{x(\sqrt{1+x}+1)} = \lim_{x \rightarrow 0} \frac{1}{\sqrt{1+x}+1} = \frac{1}{2} \quad (5.1)$$

$$\frac{\partial^2 U}{\partial x^2} + \frac{\partial^2 U}{\partial y^2} = 0 \implies U_M = \frac{1}{4\pi} \oint_{\Sigma} \frac{1}{r} \frac{\partial U}{\partial n} ds - \frac{1}{4\pi} \oint_{\Sigma} \frac{\partial}{\partial n} \left(\frac{1}{r} \right) U ds \quad (5.2)$$

$$I(z) = \sin\left(\frac{\pi}{2}z^2\right) \sum_{n=0}^{\infty} \frac{(-1)^n \pi^{2n}}{1 \cdot 3 \cdots (4n+1)} z^{4n+1} - \cos\left(\frac{\pi}{2}z^2\right) \sum_{n=0}^{\infty} \frac{(-1)^n \pi^{2n+1}}{1 \cdot 3 \cdots (4n+3)} z^{4n+3} \quad (5.3)$$

按照从左到右的顺序看公式，那么构造出生成它们的文本应该是没有什么困难的。例如，最后一个公式就是用如下输入生成的：

```
\begin{equation}
I(z) = \sin( \frac{\pi}{2} z^2 ) \sum_{n=0}^{\infty}
\frac{ (-1)^n \pi^{2n} }{ 1 \cdot 3 \cdot \cdots (4n+1) } z^{4n+1}
- \cos( \frac{\pi}{2} z^2 ) \sum_{n=0}^{\infty}
\frac{ (-1)^n \pi^{2n+1} }{ 1 \cdot 3 \cdot \cdots (4n+3) } z^{4n+3}
\end{equation}
```

上面的公式不是用 `displaymath` 环境或者其简写形式 `\[...\]` 生成，而用的是 `equation` 环境，这样它就会自动给公式加上编号。在文档类 `book` 和 `report` 中，公式是在章内顺序编号的，如上所示，编号前面有章号，并且把它们一起放在小括号 () 内。而对于文档类 `article`，公式是相对于整篇文档编号的。

在默认状态下，公式编号右对齐，而且相对于公式竖直居中。如果在公式行上没有足够空间显示它，那就把它放在公式下一行的右边。如果选择了文档类选项 `leqno`，那么整篇文档的公式编号是左对齐的。

公式的自动编号就意味着作者在写作时并不一定知道编号到底是多少。在 8.3.1 节中介绍的 L^AT_EX 交叉索引系统讲解了如何引用章节编号 (8.3.3 节)，这也同样适用于公式编号。通过在 `equation` 环境中包含一条命令 `\label{引用名}`，我们可以在正文中用命令 `\ref{引用名}` 来显示还不知道的公式编号，这里的引用名是一个关键词，它是字母、数字和符号的任意组合。

再仔细地看一下公式 5.3，你会发现在 `cos()` 和 `sin()` 中的括号 () 应该再大点。而且这个公式长度恰好等于行宽，如果它再长一点儿，那就需要在某个恰当的地方把它断开，后面那部分要相对于前面一行适当地定位。到现在为止我们所学的数学要素还没有提供这方面的功能。

即使是类似于如何在公式中包含普通文本这样简单的事情，我们到现在也没讲。这一节的其它部分就是为了解决这些问题的。

最后要提一下，有时候 \TeX 选择的尺寸并不总能令人满意，如公式 5.2 的后一个积分中，如果显示的是 $\partial \frac{1}{r}$ ，就要比 $\partial_r \frac{1}{r}$ 好看得多。我们在 5.5 节中会讲解这一点和其它的格式辅助工具，如怎样调整公式两部分之间的水平距离。

§5.4.1 括号符号的尺寸自动调整

在数学中经常包含括号符号，通常是成对出现的，用来包围公式的某部分。当显示的时候，这些括号应该与被包围公式有相同的尺寸。 \LaTeX 提供了一对命令

$\backslash\text{left}$ 左括号 部分公式 $\backslash\text{right}$ 右括号

用来做到这一点。把命令 $\backslash\text{left}$ 就放在左括号符号的前面，而 $\backslash\text{right}$ 就放在右括号符号的前面。

$$\left[\int + \int \right]_{x=0}^{x=1}$$
 这里的一对括号 $[]$ 会根据被包围公式自动调整其尺寸，从而导致指数和指标也相应地长高或除低。

$\backslash\text{left}$ 和 $\backslash\text{right}$ 命令必须成对出现。每一个 $\backslash\text{left}$ 命令必须在后面某处有一个相应的 $\backslash\text{right}$ 命令。这种匹配也可以嵌套。第一个 $\backslash\text{left}$ 是与最后一个 $\backslash\text{right}$ 配对的；接下来的 $\backslash\text{left}$ 与倒数第二个 $\backslash\text{right}$ 配对，依次类推。在一个嵌套中必须有相同数目的 $\backslash\text{right}$ 和 $\backslash\text{left}$ 。

相对应的左括号和右括号符号可以任意组合，并不一定要是逻辑上的一对：

这种括号的配对当然是不寻常的，但是可以接受：

$$\vec{x} + \vec{y} + \vec{z} = \left(\begin{array}{l} a \\ b \end{array} \right) \left[\begin{array}{l} \backslash\text{vec}\{x\} + \backslash\text{vec}\{y\} + \backslash\text{vec}\{z\} = \\ \backslash\text{letf}(\dots \backslash\text{right}[\backslash \end{array} \right.$$

有的时候公式中只有左括号或右括号，而没有相配对的部分。然而即使是这种情形， $\backslash\text{left} \dots \backslash\text{right}$ 命令也必须成对出现，只是用 ‘.’ 来表示那个看不见的括号符号：

$$y = \left\{ \begin{array}{ll} -1 & : x < 0 \\ 0 & : x = 0 \\ +1 & : x > 0 \end{array} \right. \quad \backslash[\quad y = \backslash\text{left}\{ \backslash\begin{array} \quad \backslash\text{r@{\quad:\quad}l} \quad -1 \& x<0 \quad \backslash\quad 0 \& x=0 \quad \backslash\quad +1 \& x>0 \quad \backslash\text{end}\{array\} \backslash\text{right}. \quad \backslash]$$

上面例子中的 `array` 环境已在 4.8.1 节中介绍了，它生成数学模式中的表格。

$\backslash\text{left} \dots \backslash\text{right}$ 命令可以应用于下面这 22 种不同的符号上。它们是

(())	[\lfloor] \rfloor
[[]]	[\lceil] \rceil
{ \{ } }	< \angle > \rangle
\l	↑ uparrow ↗ \Uparrow
/ / \ \backslash	↓ downarrow ↘ \Downarrow
	↕ \updownarrow ⇕ \Updownarrow

例如, `\left| ... \right|` 就会生成两条根据所包围公式文本自动调整高度的竖线。

练习 5.12: 在公式 5.3 中, 用 $\cos\left(\frac{\pi}{2}z^2\right), \sin\left(\frac{\pi}{2}z^2\right)$ 代替 $\cos(\frac{\pi}{2}z^2), \sin(\frac{\pi}{2}z^2)$ 。

§5.4.2 公式中的普通文本

有时候需要在公式中包含一些普通文本, 例如单个的单词 `and`, `or`, `if` 等等。在这种情形中, 虽然处于数学模式中, 但我们需要切换回 LR 模式 (1.5.3 节和 4.7.1 节)。这可以通过在公式中执行命令 `\mbox{普通文本}`, 并结合水平间距命令如 `\quad` 和 `\hspace` 等等, 来做到这一点。例如:

$$X_n = X_k \quad \text{if and only if} \quad Y_n = Y_k \quad \text{and} \quad Z_n = Z_k$$

```
\[ X_n = X_k \quad\quad\quad\mbox{if and only if}\quad\quad\quad
Y_n = Y_k \quad\quad\quad\mbox{and}\quad\quad\quad Z_n = Z_k \quad\quad\quad\]
```

为了得到如上面例子中的那样在显示公式中的一串长普通文本, 最好的方法是把公式和文本分别放在自己的子段盒子或小页中, 并把它们用适当的竖直定位并排摆放。

另一方面, 如果要把文本字体的字母做为数学符号, 那就需要用 $\mathrm{L}^{\mathrm{A}}\mathrm{T}_{\mathrm{E}}\mathrm{X} 2_{\epsilon}$ 的数学字母命令来输入:

```
\mathrm{2\epsilon} \quad \mathtt{2\epsilon} \quad \mathbf{2\epsilon}
\mathsf{2\epsilon} \quad \mathit{2\epsilon} \quad \mathcal{2\epsilon}
```

我们在 5.3.2 节中已经用 `\mathcal` 命令显示花体字母。所有这些命令的作用方式是一样的: 给它们的参数取相应的字体。

$$\mathbf{B}^0(x) \quad \mathit{T}_j^i \quad \mathbf{B}^0(x) \quad \mathsf{T}^{i_j}$$

在 5.3.1 节中的 `\mathnormal` 命令也属于这个组。它与 `\mathit` 的差别在于它把自己的参数设成正常的数学斜体, 而后者则设成普通文本斜体。字母是一样的, 而间距则不一样。

$$\mathnormal{differ} \neq \mathit{differ}$$

所有的数学字母命令只是设置数学模式中的文本，因此这就意味着空格还是要像通常那样被忽略。而对于在 `\mbox` 中的文本则不会如此。

在 \LaTeX 2.09 中，即使在数学模式中，也可以通过在一个无名环境中利用两字母字体声明来达到相同的效果。上例中的前一个可以用 `\mathbf{B}^0(x)` 来得到。

§5.4.3 矩阵和域

$$\begin{array}{cccc} a_{11} & a_{12} & \cdots & a_{1n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{array}$$

类似于左边这样的结构是矩阵，行列式，方程组等等的基础。这里把它们统称为 域。

域是用 `array` 环境生成的，我们已在 4.8.1 节关于表格的内容中给出了它的语法和构造。`array` 环境生成数学模式中的表格，即把列条目看做公式文本。例如：

$$\begin{array}{cccccc} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n & = & b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n & = & b_2 \\ \cdots & & \cdots \\ a_{n1}x_1 + a_{n2}x_2 + \cdots + a_{nn}x_n & = & b_n \end{array}$$

```
\[ \begin{array}{*{3}{c@{\:+\:}}c@{\:=\:};c}
  a_{11}x_1 & a_{12}x_2 & \cdots & a_{1n}x_n & b_1 \\
  a_{21}x_1 & a_{22}x_2 & \cdots & a_{2n}x_n & b_2 \\
  \multicolumn{5}{c}{\dotfill} \\
  a_{n1}x_1 & a_{n2}x_2 & \cdots & a_{nn}x_n & b_n
\end{array} \]
```

这里重新复习一下表格的构造要素（4.8.1 节）：`@{文本}` 在相邻两列间插入 文本 的内容。在上面的例子中，这就是 `\:+\:` 和 `\:=\:`。`\:` 和 `\;` 命令现在还没有讲，但它们是用来生成数学模式中小的水平间距（5.5.1 节）。`*{3}{c@{\:+\:}}` 是列定义 `c@{\:+\:}` 的三次重复。`c` 规定了列文本是居中排列的。`\multicolumn{5}{c}` 把五列进行了合并，用一个居中条目代替。`\dotfill` 用点充满该列。上面的方程组也可以用下面这样的格式更简单地得到：

```
\begin{array}{c@{\:+\:}c@{\:+\:\cdots+}\;c@{\:=\:};c}
```

也可以把 `array` 环境嵌套任意层次：

```
\[ \left( \begin{array}{cc}
  x_{11} & x_{12} \\
  x_{21} & x_{22}
\end{array} \right) x
```

$$\left(\begin{array}{cc} x_{11} & x_{12} \\ x_{21} & x_{22} \end{array} \right) x$$

```
\end{array} \right) x
```

绝大多数域中的列条目都是居中排列 (c) 的。在上面的列中第一个条目还是一个域，它具有两个居中的列。其左右用可自动调整高度的竖线包围。

`array` 环境在结构上与垂直盒子是一样的。这就是说在包围它的环境中，它只是被做为单个字符一样处理，因此它也可以同其它符号和结构一起出现：

$$\sum_{p_1 < p_2 < \dots < p_{n-k}}^{(1,2,\dots,n)} \Delta \sum_{q_1 < q_2 < \dots < q_k} \begin{vmatrix} a_{q_1 q_1} & a_{q_1 q_2} & \dots & a_{q_1 q_k} \\ a_{q_2 q_1} & a_{q_2 q_2} & \dots & a_{q_2 q_k} \\ \dots & \dots & \dots & \dots \\ a_{q_k q_1} & a_{q_k q_2} & \dots & a_{q_k q_k} \end{vmatrix}$$

```
\[ \sum_{p_1 < p_2 < \cdots < p_{n-k}}^{\{1,2,\ldots,n\}}
\Delta_{\begin{array}{l}
p_1 p_2 \cdots p_{n-k} \\
\end{array}}
\sum_{q_1 < q_2 < \cdots < q_k} \left| \begin{array}{llcl}
a_{q_1 q_1} & a_{q_1 q_2} & \cdots & a_{q_1 q_k} \\
a_{q_2 q_1} & a_{q_2 q_2} & \cdots & a_{q_2 q_k} \\
\multicolumn{4}{c}{\dotfill} \\
a_{q_k q_1} & a_{q_k q_2} & \cdots & a_{q_k q_k}
\end{array} \right| \]
```

在这个例子中，`array` 环境用在 Δ 的指标。然而，这个指标相对于其它部分公式显得太大了。在 5.4.6 节中提供了更好地解决这种域指标的方法。

同所有的表格环境一样，也可以在 `array` 环境中包含可省的垂直定位参数 `b` 或 `t`。在 4.7.3 和 4.8.1 节中已描述了其语法和结果。只有当域不需要竖居中，而是相对于它的顶行或底行垂直定位时才需要这个参数值。

$$\begin{array}{c} a_1 \\ \vdots \\ a_n \end{array} - \begin{array}{c} u - v \\ 10 \\ u + v \end{array} \begin{array}{c} 10 \\ 12 \\ -120 \end{array}$$

```
\[ x - \begin{array}{c} u - v \\ 10 \\ u + v \end{array} \begin{array}{c} 10 \\ 12 \\ -120 \end{array} \]
```

我们建议读者借助于右边的生成文本，尝试推断如何构造各种不同的域。

练习 5.13: 方程组的解

$$F(x, y) = 0 \quad \text{and} \quad \begin{vmatrix} F''_{xx} & F''_{xy} & F'_x \\ F''_{yx} & F''_{yy} & F'_y \\ F'_x & F'_y & 0 \end{vmatrix} = 0$$

生成 $F(x, y) = 0$ 所有可能拐点的坐标。

注意：上面的显示公式是由两个小公式组成的，中间的单词 and 要在其前后加上 `\quad` 以插入额外的间距。可以不用 `\left|\dots\right|` 确定的自动调整高度的竖线包围 `array` 环境，而代之以格式参数值 `{|\dots|}`（4.8.1 节）也能达同样的目的。这样的结构在数学上称为行列式。

练习 5.14: 两方程为

$$\frac{x-x_1}{l_1} = \frac{y-y_1}{m_1} = \frac{z-z_1}{n_1} \quad \text{and} \quad \frac{x-x_2}{l_2} = \frac{y-y_2}{m_2} = \frac{z-z_2}{n_2}$$

的直线之间的最短距离是如下表达式：

$$\frac{\pm \begin{vmatrix} x_1-x_2 & y_1-y_2 & z_1-z_2 \\ l_1 & m_1 & n_1 \\ l_2 & m_2 & n_2 \end{vmatrix}}{\sqrt{\begin{vmatrix} l_1 & m_1 \\ l_2 & m_2 \end{vmatrix}^2 + \begin{vmatrix} m_1 & n_1 \\ m_2 & n_2 \end{vmatrix}^2 + \begin{vmatrix} n_1 & l_1 \\ n_2 & l_2 \end{vmatrix}^2}}$$

如果分母为零，那么这两条直线必相交。

注意：在上面表达式分母中根号下的三个行列式并不推荐使用 `{|cc|}` 形式的格式化参数值。这里应该用 `\left|\dots\right|`。试试这两种可能，并把结果做一对比。

练习 5.15: Laurent 展开：令 $c_n = \frac{1}{2\pi i} \oint (\zeta - a)^{-n-1} f(\zeta) d\zeta$ ，那么对任何函数 $f(z)$ 都有下面的表达式（ $n = 0, \pm 1, \pm 2, \dots$ ）：

$$f(z) = \sum_{n=-\infty}^{+\infty} c_n (z-a)^n = \begin{cases} c_0 + c_1(z-a) + c_2(z-a)^2 + \dots + c_n(z-a)^n + \dots \\ \qquad \qquad \qquad + c_{-1}(z-a)^{-1} + c_{-2}(z-a)^{-2} + \dots \\ \qquad \qquad \qquad + c_{-n}(z-a)^{-n} + \dots \end{cases}$$

提示：公式的右边可以用只有一列的 `array` 环境生成。那么它的格式参数值是什么呢？

§5.4.4 公式上下的直线

命令

`\overline{部分公式}` 和 `\underline{部分公式}`

可以用来在公式或其一部分的上面或下面画一直线。而且它们可以嵌套任意次：

$$\frac{\overline{\overline{a^2 + xy + \overline{z}}}}{\overline{a^2 + xy + \overline{z}}} + \overline{\overline{\overline{z}}}$$

命令 `\underline` 也可用在普通文本模式中以生成下划线，而 `\overline` 只能用在数学模式中。

与这两条命令完全类似地还有:

`\overbrace{部分公式}` 和 `\underbrace{部分公式}`

它们在 部分公式 的上方或下方放上一个水平的大括号:

$$\overbrace{a+b+c+d} \quad \backslash\overbrace{a+\underbrace{b+c}+d}$$

在显示公式中, 这些命令也可以有指数或指标。(升高的)指数被放在上括号的上方, 而(降低的)指标放在下括号的下方。

$$\overbrace{a+b+\cdots+y+z}^{123} \quad \backslash[\underbrace{a+\overbrace{b+\cdots+y}^{123}}_{\alpha\beta\gamma}+z]_{\{\alpha\beta\gamma\}} \quad \backslash[\underbrace{a+\overbrace{b+\cdots+y}^{123}}_{\alpha\beta\gamma}+z]_{\{\alpha\beta\gamma\}}$$

练习 5.16: 从 n 个元素中取 m 个的排列总数 (用符号 P_n^m 表示) 为

$$P_n^m = \prod_{i=0}^{m-1} (n-i) = \underbrace{n(n-1)(n-2)\cdots(n-m+1)}_{\text{total of } m \text{ factors}} = \frac{n!}{(n-m)!}$$

§5.4.5 堆积符号

命令

`\stackrel{上部符号}{下部符号}`

把上部符号 居中放在 下部符号 的上方, 这里放在上方的符号要以较小的字样显示:

$$\vec{x} \stackrel{\text{def}}{=} (x_1, \dots, x_n) \quad \$\vec{x} \stackrel{\text{def}}{=} (x_1, \dots, x_n) \$$$

$$A \stackrel{\alpha'}{\longrightarrow} B \stackrel{\beta'}{\longrightarrow} C \quad \$ A \stackrel{\alpha'}{\longrightarrow} B \stackrel{\beta'}{\longrightarrow} C \dots \$$$

利用数学字体尺寸命令 (5.5.2 节), 用这个命令可以构造新的符号。例如, 有些作者喜欢符号 \leq 是 \leq 样子, 而不是 \leq , 那么这可以利用 `<` 和 `=` 的组合 `\stackrel{<}{=}` 得到。如果这里不用命令 `\textstyle`, 那么结果是 \leq 。

§5.4.6 其它的 TeX 数学命令

TeX 数学命令 `\atop` 和 `\choose` 也是非常有用的, 可以应用于任一 L^AT_EX 文档中。(事实上, 所有 TeX 数学命令中除了 `\eqalign`, `\eqalignno` 和 `\eqaligno` 外, 都可以用于 L^AT_EX 文稿中。) 它们的语法是

{ 顶部公式 } `\atop` { 底部公式 }

{ 顶部公式 } `\choose` { 底部公式 }

这两条命令都生成一个类似于没有分数线的分数结构。对于 `\choose` 命令, 该结构被包围在小括号内 (在数学中称之为二项式系数)。

$$\binom{n+1}{k} = \binom{n}{k} + \binom{n}{k-1} \quad \backslash[\{n+1 \choose k\} = \{n \choose k\} + \{n \choose k-1\} \backslash]$$

$$\prod_{j \geq 0} \left(\sum_{k \geq 0} a_{jk} z^k \right) = \sum_{n \geq 0} z^n \left(\sum_{\substack{k_0, k_1, \dots \geq 0 \\ k_0 + k_1 + \dots = n}} a_{0k_0} a_{1k_1} \dots \right)$$

`\[\prod_{j \geq 0} \left(\sum_{k \geq 0} a_{jk} z^k \right) =`
`\sum_{n \geq 0} z^n \left(\sum_{k_0, k_1, \dots \geq 0 \atop k_0 + k_1 + \dots = n} a_{0k_0} a_{1k_1} \dots \right) \]`

利用 L^AT_EX 环境也可以生成类似的结构：

`\begin{array}{c} 顶部行 \\ 底部行 \end{array}` (atop)

`\left(\begin{array}{c} 顶部行 \\ 底部行 \end{array} \right)` (choose)

在 `array` 结构与相应 T_EX 命令之间的差别在于前者总是以普通正文公式的样式和尺寸显示，而后者会视所处公式的部分而改变尺寸。

比较如下：

$\Delta_{\substack{p_1 p_2 \dots p_{n-k} \\ p_1 p_2 \dots p_{n-k}}}$ 这里的指标域是用 `\atop` 生成的

$\Delta_{\substack{p_1 p_2 \dots p_{n-k} \\ p_1 p_2 \dots p_{n-k}}}$ 这里的指标是用 `array` 环境生成的

上面的 T_EX 命令也可以用来生成正文公式中的小矩阵，如 $\begin{pmatrix} a & b & c \\ l & m & n \end{pmatrix}$ 或者 $\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$ 。这里的第一个矩阵是用

`\left(\begin{smallmatrix} 1 & 0 \\ 0 & 1 \end{smallmatrix} \right)`

生成的，而第二个是用

`\left(\begin{smallmatrix} a & b \\ l & m \end{smallmatrix} \right)`

生成的。

§5.4.7 多行公式

所谓多行公式，就是一个公式长达几行，其中每行的关系符号（例如 = 或 ≤）是彼此竖直对齐的。为此，环境

`\begin{eqnarray}` 第一行 ... 第 *n* 行 `\end{eqnarray}`

`\begin{eqnarray*}` 第一行 ... 第 *n* 行 `\end{eqnarray*}`

被用来排版显示模式中的长达几行的公式或方程。方程或公式中的行与行之间用 `\\` 分开。每行的形式为

左边公式 & 中间公式 & 右边公式 `\\`

在显示的时候，左边公式在左边列中右对齐，而右边公式在右边列中左对齐，中间公式在中间列居中。列分隔符号 & 标志着公式的不同部分。通常中间公式就是单个的数学符号，即上面所说的关系运算符。因此每一行公式的样子如同 `\begin{array}{rcl} ... \end{array}` 环境中的行。

在 `array` 环境和 `eqnarray` 环境之间的差别是：后者排版为显示公式。这也就是说对于那些列在 5.3.7 节中的符号，会取它们的大尺寸形式，分数的分子和分母也是正常尺寸。另一方面，对于 `array` 环境而言，列条目被设为正文公式，会选取那些符号的小尺寸形式，分数的两部分也是以较小尺寸显示的。

在 `eqnarray` 环境的标准形式中，是会自动给公式加上有顺序的编号的，而在 `*` 形式中则没有编号。为了不给标准形式中某行公式加编号，可以在该行公式的结束符 `\\` 前面插入命令 `\nonumber`。

公式的编号可以在正文中用 `\ref{引用名}` 命令引用，这里的引用名就是要用 `\label{引用名}` 命令赋给某一行公式的关键词。详情请见 8.3.1 节。

例：

$$\begin{aligned}(x+y)(x-y) &= x^2 - xy + xy - y^2 \\ &= x^2 - y^2\end{aligned}\tag{5.4}$$

$$(x+y)^2 = x^2 + 2xy + y^2\tag{5.5}$$

```
\begin{eqnarray}
(x+y)(x-y) &= & x^2-xy+xy-y^2 \nonumber\\
&= & x^2 - y^2 \\
(x+y)^2 &= & x^2+2xy+y^2
\end{eqnarray}
```

$$\begin{aligned}x_n u_1 + \cdots + x_{n+t-1} u_t &= x_n u_1 + (a x_n + c) u_2 + \cdots \\ &\quad + (a^{t-1} x_n + c(a^{t-2} + \cdots + 1)) u_t \\ &= (u_1 + a u_2 + \cdots + a^{t-1} u_t) x_n + h(u_1, \dots, u_t)\end{aligned}$$

```
\begin{eqnarray*}
x_{nu_1} + \cdots + x_{\{n+t-1\}u_t} &= & x_{nu_1} + (a x_n + c) u_2 + \\
&& \cdots \\
&+ & \left(a^{\{t-1\}x_n} + c(a^{\{t-2\}} + \cdots + 1)\right) u_t \\
&= & (u_1 + a u_2 + \cdots + a^{\{t-1\}u_t}) x_n + h(u_1, \ldots, u_t)
\end{eqnarray*}
```

我们需要对后面这个例子做一些解释。在第二行上有一个自动调整尺寸的 `\left(... \right)` 对。这样的对只能位于一行中；即不能把用行结束符 `\\` 把它们分开！如果多行公式中有自动调整尺寸的括号，只可能出现在单行中。

如果配对括号必须位于不同行中，那么可以尝试用 `\left(... \right.` `\\ \left. ... \right)` 结构。在第一行，`\left(` 与看不见的括号 `\right.`

配对，而第二行则由 `\left.` 开始，它与闭括号 `\right)` 配对。然而，这种方法只有当两部分公式的高度大致相当时才会使两个自动调整尺寸的结果差不多。在 5.5.3 节中描述了当自动调整尺寸失败时如何手工选择括号的尺寸。

第二行开头的 `+` 也需要加以解释。在数学中 `+` 和 `-` 有两种意义：在两个数学量之间，扮演一种结合的角色（二元运算符），但只是在一个数学符号前，那它就是一个符号标志（正号或负号）。 \LaTeX 通过插入不同的间距来强调这种不同（例如，请看 $+b$ 和 $a+b$ 的差别）。`n`

$y = a + b + c + d$ 如果一个很长的公式被分成几行，其中有的行以
 $+e + f + g$ `+` 或 `-` 开头， \LaTeX 会认为它是一个标志符号，
 $+h + i + j$ 从而把它向下一个字符移近。

解决的方法是在该行开头处引进一个零宽度的看不见字符。这可以是一个空结构 `{}`。请比较上面公式中 `&& +e+f+g` 和 `&&{}+h+i+j` 的结果。

有的时候对于多行公式最好采用下列这种断行的方法：

$w + x + y + z =$ 即第二行及其后续各行并不是相对于第一行的
 $a + b + c + d + e + f +$ 等号对齐，而是相对于第一行缩进一点后
 $g + h + i + j + k + l$ 左对齐。

`\begin{eqnarray*}` 在第一行的命令 `\lefteqn{w+x+y+z =}` `\`
`\lefteqn{w+x+y+z =}` `\` 可以使得参数值的内容被显示出来，但是
`&& a+b+c+d+e+f+ \` \LaTeX 认为它的宽度为零。因此左边列只有
`&& g+h+i+j+k+l` 列间距，由它生成其余各行的左面缩进。
`\end{eqnarray*}`

可以通过在 `\lefteqn{...}` 和行结束符之间插入 `\hspace{深度}` 命令来改变缩进深度。正的 深度 会增加缩进，而负值则减少缩进。

练习 5.17: 下面的公式的分行为：

$$\begin{aligned} \arcsin x &= -\arcsin(-x) = \frac{\pi}{2} - \arccos x = \left[\arccos \sqrt{1-x^2} \right] \\ &= \arctan \frac{x}{\sqrt{1-x^2}} = \left[\operatorname{arccot} \frac{\sqrt{1-x^2}}{x} \right] \end{aligned} \quad (5.6)$$

$$\begin{aligned} f(x+h, y+k) &= f(x, y) + \left\{ \frac{\partial f(x, y)}{\partial x} h + \frac{\partial f(x, y)}{\partial y} k \right\} \\ &+ \frac{1}{2} \left\{ \frac{\partial^2 f(x, y)}{\partial x^2} h^2 + 2 \frac{\partial^2 f(x, y)}{\partial x \partial y} kh + \frac{\partial^2 f(x, y)}{\partial y^2} k^2 \right\} \\ &+ \frac{1}{6} \{ \cdots \} + \cdots + \frac{1}{n!} \{ \cdots \} + R_n \end{aligned} \quad (5.7)$$

关于可能出现的错误信息的注释：

对那些有很多组逻辑括号的长公式,尤其是嵌套很多次的情形,刚开始排版时总会有错误。原因就在于括号的顺序不对或者遗漏了配对。

如果在公式处理过程中 L^AT_EX 报告有错误,初学者经常无法理解错误信息(在第 9 章中详细介绍了各种错误信息),这时候就应该仔细地检查公式文本中括号配对。有些文本编辑器可以帮助用户搜索配对括号,这样可以简化这种操作。

如果这样还没有找到错误所在,那么用户就可以尝试当出现错误时按回车键,以继续处理下去。从这样所得的打印结果中就会找到错误所在。

练习 5.18: 在 `eqnarray` 环境中会在列分隔符号 `&` 位于的地方插入额外间距。当公式是用一个长求和表达中的 `+` 或 `-` 分行对齐时,这种间距就是不必要的。例如:

多项式展开 $y = f(x) = ax + bx^2 + cx^3 + dx^4 + ex^5 + fx^6 + \cdots$ ($a \neq 0$) 的逆函数开头几项为

$$\begin{aligned} x = \varphi(y) = & \frac{1}{a}y - \frac{b}{a^3}y^2 + \frac{1}{a^5}(2b^2 - ac)y^3 \\ & + \frac{1}{a^7}(5abc - c^2d - fb^3)y^4 \\ & + \frac{1}{a^9}(6a^2bd + 3a^2c^2 + 14b^4 - a^3e - 21ab^2c)y^5 \\ & + \frac{1}{a^{11}}(7a^3be + 7a^3cd + 84ab^3c - a^4f - \\ & 28a^2b^2d - 28a^2bc^2 - 43b^5)y^6 + \cdots \end{aligned}$$

选择 `\arraycolsep` (4.8.2 节) 的一个适当值,使得 `+` 或 `-` 与分点之间的距离相对于其它部分的公式尽可能得小。

§5.4.8 有框公式和并列公式

显示公式或方程也可以放在适当宽度的竖直盒子中,即 `\parbox` 命令或 `minipage` 环境中。在竖直盒子中,公式是水平居中的,或者根据所选择的文档类选项进行 `\mathindent` 缩进后左对齐。

竖直盒子可以如同单个字符一样,相对于另一个竖直盒子定位(4.7.3 和 4.7.7 节)。通过这种方法,我们可以使得显示公式或方程并列摆放。

$$\begin{array}{ll} \alpha = f(z) & (5.8) \\ \beta = f(z^2) & (5.9) \\ \gamma = f(z^3) & (5.10) \end{array} \quad \begin{array}{l} x = \alpha^2 - \beta^2 \\ y = 2\alpha\beta \end{array}$$

左面的公式放在宽度为 4cm 的 `\parbox` 中,而右边那个放在宽度为 2.5cm 的 `\parbox` 中,而这段文本则是在宽度为 4.5cm 的 `minipage` 中。

```
\parbox{4cm}{\begin{eqnarray} \alpha &=& f(z) \dots \end{eqnarray}}
\hfill \parbox{2.5cm}{\begin{eqnarray*}
x &=& \alpha^2 - \beta^2 \backslash y &=& 2\alpha\beta \end{eqnarray*}}
\hfill \begin{minipage}{4.5cm} 左面的公式 ... \end{minipage}
```

竖直盒子也可以用来解决公式编号位置不恰当的问题。在 `eqnarray` 环境中是为每行生成一个公式编号，不要编号的话则用命令 `\nonumber`。而为了给一组公式加上竖直居中的编号，如：

$$\begin{aligned} P(x) &= a_0 + a_1x + a_2x^2 + \dots + a_nx^n \\ P(-x) &= a_0 - a_1x + a_2x^2 - \dots + (-1)^n a_nx^n \end{aligned} \quad (5.11)$$

就可以用如下输入来得到：

```
\parbox{10cm}{\begin{eqnarray*} \dots \end{eqnarray*}} \hfill
\parbox{1cm}{\begin{eqnarray}\end{eqnarray}}
```

这里真正的方程是用 `eqnarray*` 环境生成的，放在宽度为 10cm 的竖直盒子中，其后接一个空的 `eqnarray` 环境，它是一个宽度为 1cm 的盒子，由它生成公式编号。这两个盒子都是沿着它们自己的中间竖直对齐的。

如果要用方框强调某公式，也不需要什么新的结构要素。只要把它们放在 `\fbox`（4.7.7 节）中就可以了。正文公式 $a+b$ 要加上方框，只要输入 `\fbox{$a+b$}` 就可以了。

要给显示公式加上方框，那就首先需要把它放在一个宽度足够的 `\parbox` 或 `minipage` 环境中，然后把它们放在一个 `\fbox` 中。（另外的方法见 5.5.6 节。）

$$\int_0^\infty f(x) dx \approx \sum_{i=1}^n w_i e^{x_i} f(x_i)$$

就是用如下输入生成的：

```
\fbox{\parbox{5cm}{\[\int_0^\infty f(x)\,dx \approx \dots\]}}
```

§5.4.9 化学方程式和数学公式中的黑体

数学中有时需要把单个字符或者部分公式设成黑体。要做到这一点，只需要利用在 5.4.2 节中提到的数学字样命令 `\mathbf`^[25] 就可以了：

`\mathbf{S^{-1}TS = db(\omega_1, \ldots, \omega_n) = \Lambda}` \$ 可以得到 $\mathbf{S^{-1}TS = db(\omega_1, \dots, \omega_n) = \Lambda}$ 。

在这个例子中，所有公式都被设为 `\mathbf` 的参数值，这样全部都是黑体。实际上只有数字、小写和大写的拉丁字母以及大写的希腊字母被 `\mathbf` 设成黑体。小写的希腊字母和其它数学符号仍然是普通数学字体。

如果只有一部分公式被设成黑体, 那么就必须把这部分做为 `\mathbf` 的参数值。

`\mathbf{2\sqrt{x}/y}=z` $2\sqrt{x}/y = z$

数学字体命令 `\boldmath` 会把所有字符设成黑体, 只有下列符号例外:

- 上升或下降的符号 (指数和指标)
- 字符 `+ : ; ! ? () []`
- 有两种尺寸的符号 (5.3.7 节)

而 `\boldmath` 命令不能位于数学模式中。必须在切换进入数学模式之前或者在子段盒子及小页环境中使用。相反的命令 `\unboldmath` 把数学字体重设回正常的字样。

`\boldmath \[\oint\limits_C V\,d\tau = \oint\limits_\Sigma \nabla \times V\,d\sigma \]` `\unboldmath`

即使在数学模式外面已经使用了 `\boldmath`, 也可以在内部用如下命令暂时性地关闭这种设置: `\mbox{\unboldmath$...$}`, 以把这部分的公式取成正常数学字体。

`\boldmath\(\mathbf{P} = m\mathbf{b}\)` `\unboldmath`

的结果为: $\mathbf{P} = m\mathbf{b}$ 。类似地, 在数学模式中也可用 `\mbox{\boldmath$...$}` 结构暂时性打开 `\boldmath` 设置: $W_r = \int \mathbf{M} d\varphi = r^2 m \omega^2 / 2$

`\(\mathbf{W}_r = \int \mathbf{M} d\varphi = \dots\)`

化学公式通常采用的是罗马字样, 而不是数学公式中的斜体。通过把公式作为字体命令 `\mathrm` 的参数值可以实现这一效果:

`\mathrm{Fe_2^{2+}Cr_2O_4}` $\text{Fe}_2^{2+}\text{Cr}_2\text{O}_4$

如果仔细观察, 你会发现 Cr 和 O 的指标不像 Fe 的那么低。7.3.4 节在 172 页上的例 5 中描述了如何简单地生成化学方程式, 而且不会有这种缺陷。

在 \LaTeX 2.09 中, 没有字体命令 `\mathbf` 和 `\mathrm`, 而是用如 5.4.2 节所示那样, 用 `\bf` 和 `\rm` 取代。

§5.5 数学公式的精调

至此我们给出的数学要素, 几乎可以排版普通文稿中所可能出现的各种公式, 前提条件是作者要遵守印刷数学公式时某种一般性的广为接受的规则。那些到现在为止还在用打字机打印文稿的作者可能会觉得由 \TeX 选择的字符间距太窄了。事实上, 关于数学排版, \TeX 要比绝大多数作者知道得都多。在采用下面这节中的格式化工具进行修改公式排版时, 作者应该首先去查看一些出版文献中类似公式的排版。否则, 就有可能这个 \LaTeX 文档中手工修改的公式最后还是被专业制版工人重新改回原来标准 \TeX 所排版出来的样子。

§5.5.1 水平间距

虽然 $\text{T}_\text{E}\text{X}$ 对数学排版的规则有深入的了解，但是我们不能期望它理解数学公式的意义。例如 $y\,dx$ 通常意味着变量 y 与微分算子 dx 的组合，这种组合的标志是两者之间有一个小间距。然而， $\text{T}_\text{E}\text{X}$ 会去掉 $y\,dx$ 中的空格，从而显示为 ydx ，即 y 、 d 和 x 三个量的乘积。对于这种情形， $\text{L}^\text{A}\text{T}_\text{E}\text{X}$ 就需要一些精调辅助。

在数学模式中可以用下面的命令插入小量水平间距：

$\backslash,$ 小间距 = 3/18 of a quad
 $\backslash:$ 中间距 = 4/18 of a quad
 $\backslash;$ 大间距 = 5/18 of a quad
 $\backslash!$ 负间距 = -3/18 of a quad

在下面的例子中，第三列就是没有插入水平间距命令的结果：

$\backslash\sqrt{2}\backslash,x\backslash$	$\sqrt{2}\,x$	$\sqrt{2}x$
$\backslash\sqrt{\backslash,\backslash\log x}\backslash$	$\sqrt{\log x}$	$\sqrt{\log x}$
$\backslash O\left(1/\sqrt{n}\right)\backslash$	$O(1/\sqrt{n})$	$O(1/\sqrt{n})$
$\backslash[0,1)\backslash$	$[0,1)$	$[0,1)$
$\backslash\log n\backslash,(\backslash\log\log n)^2\backslash$	$\log n(\log \log n)^2$	$\log n(\log \log n)^2$
$\backslash x^2\backslash!/2\backslash$	$x^2/2$	$x^2/2$
$\backslash n\backslash!\backslash\log n\backslash$	$n/\log n$	$n/\log n$
$\backslash\Gamma_2\backslash+\backslash\Delta^2\backslash$	$\Gamma_2 + \Delta^2$	$\Gamma_2 + \Delta^2$
$\backslash R_i\backslash^j\backslash_{\backslash!kl}\backslash$	$R_i^j{}_{kl}$	$R_i^j{}_{kl}$
$\backslash\int_0^x\int_0^y\backslash dF(u,v)\backslash$	$\int_0^x\int_0^y dF(u,v)$	$\int_0^x\int_0^y dF(u,v)$
$\backslash[\int\int\int\backslash!\backslash!\backslash!\backslash\int_D\backslash dx\backslash,dy\backslash]$	$\iiint_D dx\,dy$	$\int\int_D dx\,dy$

注意：在倒数第三个例子中的 $R_i\backslash^j$ ，在 R_i 的指标之后构造一个零宽度的不可见字符，用这个空字符接受后面的指数。这样结果就是 R_i^j ，而不是 $R_i\backslash^j$ 的 R_i^j 。

对于数学间距命令，并没有一个严格的普遍适用的规则。可以考虑的就是前面提到的微分算子，正文公式中的小根号后接一个变量，除号 /，以及多重积分号。上面的例子演示了许多适合的情形。

§5.5.2 在公式中选择字体尺寸

可以把各部分公式改变成 $\text{T}_\text{E}\text{X}$ 所选择的字体尺寸。首先我们要解释一下在数学模式中有哪些尺寸可以使用， $\text{T}_\text{E}\text{X}$ 的选择准则是什么。

在数学模式中选择四种字体尺寸，它们的实际尺寸是相对于文档类的基础字体尺寸的：

<code>\displaystyle</code>	D	显示公式的标准尺寸
<code>\textstyle</code>	T	正文公式的标准尺寸
<code>\scriptstyle</code>	S	上下标的标准尺寸
<code>\scriptscriptstyle</code>	SS	更低层上下标的标准尺寸

从现在开始我们就采用这里的符号缩写 D , T , S 和 SS 。当切换到数学模式时, 被激活的字体是 D (显示公式) 或 T (正文公式)。它们的差别就在于那些有两种尺寸的符号, 以及那些上下标相应的样子 (5.3.7 节)。大符号属于 D , 小符号属于 T 。

从这两个基础尺寸开始, 各部分数学要素被设置为其它尺寸。一旦为某个要素选定了另一种尺寸, 那么这个尺寸就在这个要素中起作用。

下面的表格说明的是选择规则:	如果被激活的尺寸是 D , 那么就为分数的两部分选择 T 。也就是说对于分子和分母, 被激活的尺寸是 T 。如果它们还包括分数, 那么它们的两部分就是 S 。如果被激活尺寸是 DS 或 T , 上标和下标 (指数和指标) 就是 S ; 在上下标中 S 是激活尺寸, 其中的分数或移位就是 SS 。			
激活	分数	上、		
尺寸	上	下	下标	
D	T	T	S	
T	S	S	S	
S	SS	SS	SS	
SS	SS	SS	SS	

TeX命令 `{ \atop }` 和 `{ \choose }` 当做分数对待。

在 `array` 环境中激活的字体尺寸是 T 。

最小的可应用数学字体尺寸是 SS 。一旦已达到了这个尺寸, 就不会再更小了, 因此所有更低层的上标和下标都是 SS 。

从上面的表格很容易看出

$$\backslash[a_0 + \frac{1}{a_1 + \frac{1}{a_2 + \frac{1}{a_3 + \frac{1}{a_4}}}}] \quad a_0 + \frac{1}{a_1 + \frac{1}{a_2 + \frac{1}{a_3 + \frac{1}{a_4}}}}$$

的结果一定是右边那样的形式。

这样的数学结构, 称为连分数, 通常是如下排版的:

$$a_0 + \frac{1}{a_1 + \frac{1}{a_2 + \frac{1}{a_3 + \frac{1}{a_4}}}} \quad \backslash[a_0 + \frac{1}{\displaystyle a_1 + \frac{1}{\displaystyle a_2 + \frac{1}{\displaystyle a_3 + \frac{1}{a_4}}}}]$$

通过在每部分中显式地给出字体尺寸, 我们可以确定被激活的尺寸, 而不是内部所选择的尺寸。在这个例子中, 每个分母选择的都是 D 。因此后面的分数就以最大尺寸显示出来。最后那个分数中的 `\displaystyle` 命令可以忽略。(为什么呢?)

上面的尺寸选择表格使得我们可以精确地预见到所处部分公式的数学字

体尺寸, 这样就可以确定是否有必要显式定义字体尺寸。如此选择的效果也可以从后续公式要素中计算出来。

下面的例子中在右边列出的是没有显式定义数学字体尺寸的结果。

$$\frac{\frac{a}{x-y} + \frac{b}{x+y}}{1 + \frac{a-b}{a+b}} \quad \frac{a}{x-y} + \frac{b}{x+y} \quad \frac{a}{x-y} + \frac{b}{x+y}$$

$$e^{-\frac{x_i - x_j}{n^i + n^j}} \quad e^{\frac{x_i - x_j}{n^i + n^j}} \quad e^{-\frac{x_i - x_j}{n^i + n^j}}$$

$$\left(\begin{array}{cc} \binom{ab}{cd} & \frac{e+f}{g-h} \\ 0 & \left| \begin{array}{c} ij \\ kl \end{array} \right| \end{array} \right) \quad \left(\begin{array}{cc} \binom{ab}{cd} & \frac{e+f}{g-h} \\ 0 & \left| \begin{array}{c} ij \\ kl \end{array} \right| \end{array} \right)$$

如果在文档中经常需要显示定义尺寸 — 例如在 `array` 环境中的每个条目 — 那么就可以通过在导言中加入下列指令以避免繁琐输入:

```
\newcommand{\D}{\displaystyle}\newcommand{\T}{\textstyle}...
```

用这种方法, 当需要改变尺寸时, 只要简单地输入 `\D` 或 `\T` 等等就可以了。

这一节中余下内容对实际应用显得不是很重要。上面所给出的数学字体选择规则只是一种简化后的结果。如果读者想知道所有的细节, 我们下面就给出完整的描述。

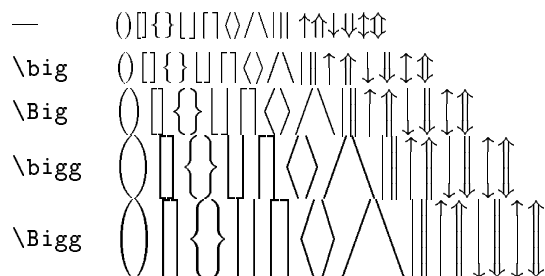
四种数学字体尺寸 D, T, S 和 SS 中	激活	分数	上	下
每一种都有一个修正版本 D', T', S'	尺寸	上	下	标
和 SS' 。差别就在于 D, T, S 和 SS 的	D	T	T'	S
上标(指数)要比 D', T', S' 和 SS' 的	D'	T'	T'	S'
稍高一点点。仔细比较一下分数线上	T	S	S'	S
下指数 2 的位置: $\frac{x^2}{x^2}$ 。在其它情形中	T'	S'	S'	S'
有撇和无撇的字体尺寸是一样的。右	S, SS	SS	SS'	SS
面给出了真正的选择规则。	S', SS'	SS'	SS'	SS'

当在分子或者上标中显式地指定字体尺寸时, 选取的是无撇形式; 而在分母和下标中则选取的是有撇字体。

§5.5.3 括号符号的手工尺寸调整

在 5.4.1 节中列出的 22 个括号符号当前缀 `\left ... \right` 命令时会自动根据被包围公式文本的高度调整其自身尺寸。然而, 也可以在括号命令

前面放上命令 `\big`, `\Big`, `\bigg` 或 `\Bigg` 来显式选择一个尺寸。



与 `\left ... \right` 对不同，显式括号尺寸命令并不需要包含在多行公式的一行中。开与闭括号可以处在不同的行上。这条规则也同样适用于下面段落中描述的命令。

也存在着另外两组名称为 `\bigl ... \Bigl` 和 `\bigr ... \Bigr` 的括号尺寸命令。这两组命令把后接的括号符号当做开或闭的括号处理。这些命令与标准命令之间的实际区别对于 \LaTeX 而言没有多大意义。

更进一步的括号尺寸命令是 `\bigm ... \Bigm`。对于这些命令，后接的括号符号如同关系运算符，在其前后

与相邻公式之间插入较大的水平间距。

	$[(a+b) (c+d)]$
<code>\[\big[(a+b) \big (c+d) \big] \]</code>	$[(a+b) (c+d)]$
<code>\[\bigr[(a+b) \bigr (c+d) \bigr] \]</code>	$[(a+b) (c+d)]$
<code>\[\bigm[(a+b) \bigm (c+d) \bigm] \]</code>	$[(a+b) (c+d)]$

§5.5.4 数学样式参数

下面列出的数学样式参数都由 \LaTeX 赋予了标准值。用户可随时用命令 `\setlength` 按通常的方式改变它们的值。

`\arraycolsep`

`array` 环境中列间距宽度的一半（也可见 4.8.2 节）。

`\jot` 在 `eqnarray` 和 `eqnarray*` 环境中插入多行公式两行之间的额外垂直距离。

`\mathindent`

当选择了文档类选项 `fleqn` 时数学公式的缩进量。

`\abovedisplayskip`

当显示公式的左边与左页边界的距离比前面文本行结尾到左页边界的距离还要小时插入在显示公式上方的垂直距离。这样的公式标记为太长。

`\belowdisplayskip`

插入在太长显示公式下方的垂直距离。

`\abovedisplayshortskip`

插入在太短显示公式上方的竖直距离。所谓太短公式就是公式的左边在前面文本行结尾的右边。

`\belowdisplayshortskip`

插入在太短显示公式下方的竖直距离。

`\topsep`

在选择文档类选项 `\fleqn` 时并不应用上述四种距离，这里要在显示公式的上方和下方插入 `\topsep`（见 4.4.2 节）。

上面除 `\jot` 外的所有参数都应该是橡皮长度（2.4.2 节）。

§5.5.5 更进一步的建议

有时候作者会希望实现用到现在为止描述的方法无法得到的公式水平或竖直对齐。这时可以考虑把公式放在水平或竖直盒子中，这样就可以随心所欲地定位了。

类似地，在 `array` 环境中结合显示尺寸声明以及来自于 4.8.1 节和 4.8.2 节中的表格构造和样式要素可以实现任意希望的水平或竖直对齐。

练习 5.19: 生成右边的连公数。

注意：与 127 页上的例子相比，这里的分子 1 都是左对齐的。

提示：还记得命令 `\hfill` 吗？

$$a_0 + \frac{1}{a_1 + \frac{1}{a_2 + \frac{1}{a_3 + \frac{1}{a_4}}}}$$

练习 5.20: 利用 `array` 环境生成下面这个方程组。

$$\begin{array}{ll} \sin 2\alpha = 2 \sin \alpha \cos \alpha, & \cos 2\alpha = \cos^2 \alpha - \sin^2 \alpha, \\ \sin 3\alpha = 3 \sin \alpha - 4 \sin^3 \alpha, & \cos 3\alpha = 3 \cos^3 \alpha - 3 \cos \alpha, \\ \sin 4\alpha = 8 \cos^3 \alpha \sin \alpha - 4 \cos \alpha \sin \alpha, & \cos 4\alpha = 8 \cos^4 \alpha - 8 \cos^2 \alpha + 1. \end{array}$$

提示：注意在 `array` 环境格式域中的 `@{\dots}` 可以用来插入两列间的水平间距和 / 或数学文本（见 5.4.3 节的第一个例子）。

练习 5.21: 用 `array` 环境得到下面的输出。

注意：如果能成功地排版这个数学表格的话，那么在定位公式或者其部分时不会再有任何困难了。

下面给出一些提示：

1. 定义一些缩写，如 `\D` 表示 `\displaystyle`，`\bm` 表示 `\boldmath`，甚至可以有 `\ba` 和 `\ea`，分别表示 `\begin{array}` 和 `\end{array}`。
2. 分步建立这个表格。首先处理标题，只有它样子可以时再继续下去。注意在 `array` 环境中的普通文本必须放在 `\mbox` 里。
3. 然后处理第一个公式行。这里第二列第一个元素是一个 `array` 环境，它与其它行的对齐方式是由定位参数值 `[t]` 确定的。在这个结构中不要忘记

记在必要时用 `\D` 激活字体尺寸。插入必要的支撑（4.7.6 节）以使得它离标题有适当的距离。而与第二行的距离可以用行结束命令 `\[. .]` 来定义。

4. 当成功地排版出这行时，下面的就没有什么困难了。
5. 在第三行公式中，第一列以及第二列的左边公式都是由 `array` 环境组成的。第三列由三个分数组成，其分母可以用 L^AT_EX 的 `array` 环境或者 T_EX 命令 `{... \atop ...}` 生成；而分数本身用另一个 `array` 环境分放在在两行上。
6. 最后一行公式的第二列和第三列也是 `array` 环境。有些部分的公式采用了 `\boldmath`。注意这个命令只能用在文本模式中，即在 `\mbox` 中。
7. 最后一行把最外层 `array` 环境的三列合并为一，其中的文本被放在一个适当宽度的子段盒子中：

```
\multicolumn{3}{|c|}{\parbox{...}{.....}}
```

Equations for the tangential plane and surface normal		
Equation for the surface	Tangential plane	Surface normal
$F(x, y, z) = 0$	$\frac{\partial F}{\partial x}(X - x) + \frac{\partial F}{\partial y}(Y - y) + \frac{\partial F}{\partial z}(Z - z) = 0$	$\frac{X - x}{\frac{\partial F}{\partial x}} = \frac{Y - y}{\frac{\partial F}{\partial y}} = \frac{Z - z}{\frac{\partial F}{\partial z}}$
$z = f(x, y)$	$Z - z = p(X - x) + q(Y - y)$	$\frac{X - x}{p} = \frac{Y - y}{q} = \frac{Z - z}{-1}$
$x = x(u, v)$ $y = y(u, v)$ $z = z(u, v)$	$\begin{vmatrix} X - x & Y - y & Z - z \\ \frac{\partial x}{\partial u} & \frac{\partial y}{\partial u} & \frac{\partial z}{\partial u} \\ \frac{\partial x}{\partial v} & \frac{\partial y}{\partial v} & \frac{\partial z}{\partial v} \end{vmatrix} = 0$	$\frac{X - x}{\begin{vmatrix} \frac{\partial y}{\partial u} & \frac{\partial z}{\partial u} \\ \frac{\partial y}{\partial v} & \frac{\partial z}{\partial v} \end{vmatrix}} = \frac{Y - y}{\begin{vmatrix} \frac{\partial z}{\partial u} & \frac{\partial x}{\partial u} \\ \frac{\partial z}{\partial v} & \frac{\partial x}{\partial v} \end{vmatrix}} = \frac{Z - z}{\begin{vmatrix} \frac{\partial x}{\partial u} & \frac{\partial y}{\partial u} \\ \frac{\partial x}{\partial v} & \frac{\partial y}{\partial v} \end{vmatrix}}$
$\mathbf{r} = \mathbf{r}(u, v)$	$(\mathbf{R} - \mathbf{r})(\mathbf{r}_1 \times \mathbf{r}_2) = 0$ <p>or $(\mathbf{R} - \mathbf{r})\mathbf{N} = 0$</p>	$\mathbf{R} = \mathbf{r} + \lambda(\mathbf{r}_1 \times \mathbf{r}_2)$ <p>or $\mathbf{R} = \mathbf{r} + \lambda\mathbf{N}$</p>
In this Table x, y, z and \mathbf{r} are the coordinates and the radius vector of a fixed point M on the curve; X, Y, Z and \mathbf{R} are the coordinates and radius vector of a point on the tangential plane or surface normal with reference to M ; furthermore $p = \frac{\partial z}{\partial x}$, $q = \frac{\partial z}{\partial y}$ and $\mathbf{r}_1 = \partial \mathbf{r} / \partial u$, $\mathbf{r}_2 = \partial \mathbf{r} / \partial v$.		

§5.5.6 有框的显示公式

在 5.4.8 节中我们给出了一种给显示公式加上方框的方法，即首先把它放在一个适当宽度的 `\parbox` 或 `minipage` 中，然后再把它们放在 `\fbox` 中以生成方框。这里的问题在于需要经常多次尝试才可能找到合理的宽度。

然而, 利用 5.5.2 节的数学字体尺寸命令还可以有另一种解决方法, 它不需要定义有框显示公式的宽度 (该方法是由德国 Kiel 大学的 Günter Green 提出的):

```
\begin{displaymath} 或 \begin{equation}
\fbbox{$ \displaystyle 公式文本 $}
\end{displaymath} 或 \end{equation}
```

在 124 面上的有框方程例子现在的样子为:

$$\boxed{\int_0^\infty f(x) dx \approx \sum_{i=1}^n w_i e^{x_i} f(x_i)} \quad (5.12)$$

生成文本为:

```
\begin{equation}
\fbbox{$ \displaystyle \int^{\infty}_0 f(x)\, dx \approx \dots $}
\end{equation}
```

由于这里用的是 `equation` 环境, 因此在最右边给公式自动加上了编号, 而方框只是围住了真正的公式。而用以前的方法, 公式编号也要放在一个盒子中。与 124 页上的例子相比, 现在方框与方程距离更近了。用户可以通过修改参数 `\fbboxsep` (4.7.8 节) 来改变这一距离。类似地, 方框的线粗也可以用 `\fbboxrule` 来设置。

采取同样的方法, 可以利用 `array` 环境, 得到有框的多行公式或方程组:

```
\begin{displaymath} 或 \begin{equation}
\fbbox{$ \begin{array}{rcl} 公式文本 \end{array} $}
\end{displaymath} 或 \end{equation}
```

由于在这一应用中数学字体尺寸命令被相当频繁地使用, 因此我们还是建议采用 128 页上的方法, 重定义命令名称。

§5.5.7 补遗

在这一章中列出的数学结构要素以及格式调整 (精调), 对于满足在数学写作方面即使是离奇需要也是足够了。在 5.5.5 节中给出的建议更使得所期望的定位和公式构造成为可能: 余下的问题就是如何积累丰富的经验, 充分利用所提供的各种功能了。

如果所需要的某一符号, \LaTeX 并没有提供, 那么我们可以尝试从已有的符号, 利用叠印、提升或降低等操作来构造出来。通过利用命令 `\newcommand`, 可以简化对这种组合符号的使用, 另外为了一般用途, 可以把它的定义存贮在一个文件中。

如果某个构造利用 \LaTeX 是无法达到的, 那么借助于 \TeX 总是可以的。

然而，我们要指出的是，超出 L^AT_EX的任何 T_EX结构要素都需要对 T_EX的工作机制有相当的了解，而大多数用户是做不到这一点的。

对于这种要求具有足够弹性的设计，一种比较合适，但是更乏味的解决方法就是利用下一章中关于构造图画命令集合。

如果作者坚持使用他自己的公式格式，那么当他把自己的文稿送到科技出版社时，他自己的格式有可能与国际标准相差太大，这样编辑们就可能把它重设成与 L^AT_EX类似的形式。

另一方面，对公式编号样式改变的要求通常就是如何对齐。在 7.3.4 节中给出了一个例子，说明如何做到这一点。可以把它做为一个样板。