第四章 谈谈数学问题

本章提要

在之前的两章中，我们以笔者大学毕业时的论文撰写过程为导引，逐一为大家介绍了`Markdown`的基本语法，以及这种轻量型标记语言在写作阶段的大部分应用。到目前为止，关于写作阶段的内容，我们就只剩下“如何用`Markdown`来描述数学问题”这一议题没有讨论了。之所以一直避而不谈这个议题，一方面是因为笔者的这篇论文主要论证的是一个网上书籍销售系统的构建过程，本身并没有涉及到数学问题。另一方面，更重要的是因为在`Markdown`文档中描述数学问题要涉及到$LaTeX$标记的运用，相对比较复杂，所以我们打算单独用一章的篇幅来讨论这个议题。

在这一章中，我们首先会介绍如何在`Markdown`文档中插入$LaTeX$标记，以呈现数学公式。然后，我们会具体介绍如何用$LaTeX$标记来描述基本四则运算、二项式方程、矩阵运算以及集合运算等数学问题。相信在学习完本章的内容之后，大家应该都能随心所欲地在`Markdown`文档中讨论数学问题了。

4.1 使用$LaTeX$标记

在`Markdown`中，数学问题的描述是通过引入$LaTeX$标记来完成的。由于$LaTeX$本身就是一门独立的标记语言，所以在介绍如何引入其相关标记之前，我们最好先对这门语言做一个简单的介绍。

4.1.1 $LaTeX$是什么?

事情得从TeX说起，众所周知，TeX是美国著名的计算机教授高德纳（Donald Ervin Knuth）[1]为撰写其伟大著作《计算机程序设计艺术》专门开发的一套排版系统。由于《计算机程序设计艺术》是一部讨论计算机算法的鸿篇巨作，其中涉及了大量的数学问题，所以为其开发的排版系统自然就在编辑复杂的数学公式方面具备了与生俱来的优势，这种先天优势使得它在数学、物理学和计算机科学等与数学表述密切相关的学术领域中非常流行，甚至很多人学习TeX就是为了使用它在数学领域中的强大表述能力。

但TeX的功能过于强大，它会要求我们在排版过程中精确描述到每一个细节，有时甚至是一个字母或标点也要照顾到。这对于大多数人来说，可能就学习成本太高，并且用起来太过繁琐和费时了。于是在上个世纪80年代，美国计算机科学家莱斯利·兰伯特（Leslie Lamport）[2]在TeX的基础上开发出了$LaTeX$这个新的排版系统（目前的版本为$LaTeX2e$）。从本质上来

说，该排版系统其实就是一组封装了TEX处理细节的宏，它可以让那些不了解排版和程序设计相关知识的人们也能使用TEX所提供的强大功能，并在几天甚至几个小时之内就能排版出具有专业质感的印刷品，而不必深陷于琐碎的排版细节中。

在中文支持方面，$LATEX$目前主要使用的`XeTeX`排版引擎，该引擎支持UTF-8编码和现代字体，可以直接使用本地计算机中安装的字体，这大大降低了$LATEX$的使用难度。

4.1.2 插入$LATEX$标记

正如上面所说的，$LATEX$本质上是一组封装了TEX处理细节的宏。所以我们接下来所使用的每一个标记实际上都是一个宏。$LATEX$的这些标记主要有两种形式，第一种形式是数学公式，而数学公式的编辑原本就是本章的主要议题，我们会在下一节中做详细介绍。在这里，我们要先来简单了解一下$LATEX$标记的命令形式，因为这种形式也会出现在复杂的公式中。这种形态的$LATEX$标记通常会以一个反斜杠开头，后面紧接着该命令的名称，这个名称既可能是个单一符号，也可能是一个字符串。命令名称之后还可以有一些指定参数，我们通常会用花括号将它们括起来（如果该参数只有一个字符，花括号可以省略）。除此之外，如果有可选参数，则用方括号括起来，具体如下：

- 无参数形式：`\command`。
- 有参数形式：`\command{参数1}{参数2}……{参数N}`。
- 带可选参数形式：`\command[可选参数]{参数1}{参数2}……{参数N}`。

例如，我们现在这里所显示的“$LATEX$”字样，使用的就是一个无参数形式的$LATEX$标记`\LaTeX`。再例如，我们之前所使用开方标记`\sqrt`，就需要有一个指定参数来表示被开方数。另外，我们还可以用该标记的可选参数来指定开方的次数。关于`\sqrt`标记的具体使用，我们在下一节中会详细介绍。

在`Markdown`中，引入$LATEX$标记的工作是靠`Mathjax`库[3]来完成的。和我们之前使用`Mermaid`库一样，`MathJax`也是一个跨浏览器的`JavaScript`程序库，它使用`MathML`、`LaTeX`和`ASCIIMathML`标记在Web浏览器中显示数学符号。当然，在使用这个库之前，必须要先确认我们所使用的`Markdown`编辑器是支持`Mathjax`的，譬如在VSCode编辑器中，我们需要安装Markdown+Math插件。

在设置好编辑环境之后，我们就可以在`Markdown`文档中插入$LATEX$标记了。具体来说，$LATEX$标记的引用方式主要有以下两种：

- **行内引用**：采用行内引用方式的$LATEX$标记通常与正常文本处于同一逻辑行中（软件界面导致的自动换行不算），具体语法就是*在待引用标记的前后各加一个美元符号*。例如，如果我们想显示“我正在使用$LATEX$来排版论文。”这句话，只需进行如下编码即可：

```
我正在使用$\LaTeX$来排版论文。
```

- **单独引用**：采用单独引用方式的$LaTeX$标记通常会独立于正常文本，自成一行，具体语法就是*在待引用标记的前后各加两个美元符号*。例如，如果我们想讨论勾股定理，就可以使用如下编码来描述该定理：

```
在平面上的一个直角三角形中，如果设直角三角形的两条直角边长度分别是a和b，斜边长度是c，那么按照勾股定理，这三边的关系如下：

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

这也就意味着：

$$ a = \sqrt{c^{2} - b^{2}} $$
$$ b = \sqrt{c^{2} - a^{2}} $$
$$ c = \sqrt{a^{2} + b^{2}} $$
```

  其渲染效果如下

> 在平面上的一个直角三角形中，如果设直角三角形的两条直角边长度分别是a和b，斜边长度是c，那么按照勾股定理，这三边的关系如下：
>
> $$a^2 + b^2 = c^2$$
>
> 这也就意味着：
>
> $$a = \sqrt{c^2 - b^2}$$
> $$b = \sqrt{c^2 - a^2}$$
> $$c = \sqrt{a^2 + b^2}$$

请暂时不用担心看不懂上面所使用的$LaTeX$标记，因为我们会在下一节中详细介绍这些标记及其使用方法。在这里，大家只需要对$LaTeX$标记的引用方式有清楚的认知就可以了。

4.2 编辑数学公式

数学公式的编辑一直以来都是我们在撰写科学类文章过程中所面临的最大麻烦之一，相信各位之前应该都尝试过各种在电子文档中输入数学公式的方法，但这些方法多数都不尽如人意，这恐怕也是驱使很多人最终下定决心来学习$LaTeX$的主要原因。毕竟，对数学公式进行编辑和排版本身就是TeX最重要的设计初衷之一，目前似乎也找不到比它更强大的数学公式编辑方法了。所以在接下来的内容中，我们就来详细、深入地介绍一下如何用$LaTeX$编辑数学公式。

4.2.1 初识公式

众所周知，数学公式与普通文本最大的不同就在于，数学公式中的每一个字符都有特定的含义，这些字符并不像普通文本那样只是单纯地横向或纵向排列，它们有一套属于自己的特殊结构和规则。例如，在上面勾股定理的推导公式中：

$$c = \sqrt{a^2 + b^2}$$

我们可以看到，a和b都由各自的平方号，外面还套着一个开根号，这些数学符号的排列既复杂又井然有序，在电子文档中实现这样的排列结构就是我们使用$LaTeX$编辑数学公式要完成的主要工作。

在$LaTeX$中，数学公式按照$LaTeX$表示引用方式的不同，也可以分为采用行内引用的**行内公式**和采用单独引用的**行间公式**两大类。譬如$f(x, y) = x^2 + y^2$，这就是一个行内公式，它们通常与普通文本同处于一个逻辑行内。而行间公式则就是我们在上面关于描述沟勾股定理的示例所看到的那种形式，它们通常独立于普通文本，自成一行。

在了解了数学公式的独特性和它们在文档中的引用形式之后，接下来就可以正式开始学习数学公式的编辑了。下面，让我们先从最基本的加减法开始。在`Markdown`中编写加减运算的公式很简单，就是在相应的数学表达式加上$LaTeX$标记，譬如，如果我们想编写$x = a + b - c$这样的公式，就只需要输入`$x=a+b-c$`即可（当然也可以使用行间公式的形式）。在这里，大家心里可能会有一个疑问：加减运算中使用的符号都是可以直接从键盘输入的，描述这类运算时似乎没有必要使用$LaTeX$标记吧？要想回答这个疑问，我们不妨先来看看使用$LaTeX$标记前后的区别：

如你所见，相同的表达式在使用了$LaTeX$标记之后，表达式中的每个字符都被设置了特定的字体。譬如，字母采用了倾斜的意大利字体，而数字和加号则采用的是直立的等宽字体，并且每个字符之间也都设置了相应的间距。正是这些排版样式将一个干巴巴的表达式变成了具有学术质感的数学公式。所以，即使没有用到任何特殊符号，我们也会建议读者尽量使用$LaTeX$标记来描述数学问题。

4.2.2 公式结构

正如我们之前所说，编辑数学公式并不是在简单地堆砌一堆数学符号，它们都是一些特定结构的组合。我们可以仔细回想一下自己迄今为止所见过的各种公式，看看是不是能从中提炼出几种特定的公式结构？这些结构中有哪些规则？下面我们就来回答一下这两个问题，用$\LaTeX$标记来描述一下这些公式结构。

4.2.3.1 上标与下标

在众多数学公式中，上标和下标无疑是两种最常见的公式结构了。其中，上标通常位于其目标符号的右上方，而下标则通常位于其目标符号的右下方，但在某些情况下，它们也会出现在目标符号的正上方和正下方，例如10^2、a_i、a_i^2、$\sum_{i=0}^{n}$等。

在$\LaTeX$中，上标用特殊字符"^"表示，下标用特殊字符"_"表示，上标和下标可以同时作用于同一个目标符号，并且它们的顺序是可以互换的，我们可以先写上标再写下标，也可以反过来写，两者互不影响。另外，如果上标或下标的内容的多于一个字符，就用括号将其括起来。下面来具体演示一下：

公式编码	渲染效果
`$10^2$`	10^2
`$2^{1024}$`	2^{1024}
`$a_i$`	a_i
`$a_{i+1}$`	a_{i+1}
`$a_i^2$`	a_i^2
`$a^2_i$`	a^2_i
`$A_{i,j}=2^{i+j}$`	$A_{i,j}=2^{i+j}$

除此之外，上标与下标之间还是可以相互嵌套的，譬如：

公式编码	渲染效果
`$2^{n_i}$`	2^{n_i}
`$2_{n^i}$`	2_{n^i}
`$2^{n^i}$`	2^{n^i}
`$2_{n_i}$`	2_{n_i}
`$2^{2^{2^{2}}}$`	$2^{2^{2^2}}$

在求和、积分等这类算子符号上，$LaTeX$基于排版因素的考虑，为避免文字过于拥挤或难看的行间距，对于行内公式和行间公式的上下标位置有不同的处理，譬如以求和式为例，其行内公式的上下标位于求和符号的右上角和右下角，而行间公式则位于求和符号的正上方和正下方。当然，我们也可以使用特定的$LaTeX$标记来指定上下标的位置。例如，如果想让求和式的行内公式采用与行间公式相同的上下标位置，可以对它使用`\limits`标记。下面，我们来具体演示一下求和式的上下标：

不止如此，我们还可以引用`amsmath`宏组[4]中的$LaTeX$标记，对上下标的位置进行更精确的指定。例如使用`\sideset`标记，可以在一个符号的四个角上各指定一个上下标。除此之外，我们还可以通过`\overset`或`\underset`标记来指定一般符号的上下标位于其正上方或正下方。下面，我们就来演示一下这些标记的使用：

公式编码	渲染效果
`$\sideset{^a_b}{} A$`	${}^a_b A$
`$\sideset{}{^c_d} A$`	${A}^c_d$
`$\sideset{^a_b}{^c_d} A$`	${}^a_b{A}^c_d$
`$\overset{*}{A}$`	$\overset{*}{A}$
`$\underset{*}{A}$`	$\underset{*}{A}$
`$\overset{*}{\underset{*}{A}}$`	$\overset{*}{\underset{*}{A}}$

对于`amsmath`宏组中其他与上下标相关的$\LaTeX$标记，读者可以自行参考相关的$\LaTeX$文档，这里就不再一一累述了。

4.2.3.2 线条与括号

我们在进行数学公式的推演和分析时，经常会需要在公式的上下方画上一些线或者花括号，以便标注一些信息，这也形成了一种独特的**标注结构**。在这一节中，我们就来介绍一下这种结构的公式编辑。先来看最简单的线条标注，在$\LaTeX$中，线条标注是通过`\overline`和`\underline`这两个标记来完成的。请注意，这两个标记不止可以作用在一般结构的数学公式上，它们可以与任意结构的公式进行嵌套组合。下面，我们来演示一下它们的用法：

公式编码	渲染效果
`$\overline{ab}=\overline{a}*\overline{b}$`	$\overline{ab}=\overline{a}*\overline{b}$
`$\underline{c}=\underline{a^2+b^3}$`	$\underline{c}=\underline{a^2+b^3}$
`$\underline{\overline{a^2}+\overline{b}^3}$`	$\underline{\overline{a^2}+\overline{b}^3}$
`$a^{\overline{2}}+b^{\underline{^3}}$`	$a^{\overline{2}}+b^{\underline{^3}}$

除此之外。我们还可以使用`amsmath`宏组中提供的六种带箭头的线条：

公式编码	渲染效果
`$\overrightarrow{a+b}$`	$\overrightarrow{a+b}$
`$\overleftarrow{a+b}$`	$\overleftarrow{a+b}$
`$\overleftrightarrow{a+b}$`	$\overleftrightarrow{a+b}$
`$\underrightarrow{a+b}$`	$\underrightarrow{a+b}$
`$\underleftarrow{a+b}$`	$\underleftarrow{a+b}$
`$\underleftrightarrow{a+b}$`	$\underleftrightarrow{a+b}$

学习完线条标注的方法之后，接下来我们来看看如何用花括号来进行标注。在$LaTeX$中，公式上下方的花括号标注是用`\overbrace`和`\underbrace`这两个标记来完成的，它们与之前线条标注之间最大的区别，就是花括号可以通过上标和下标来对公式进行文字标注，这在我们描述数学公式的分析过程中是非常有用的。下面我们就来演示一下这两个标记的用法：

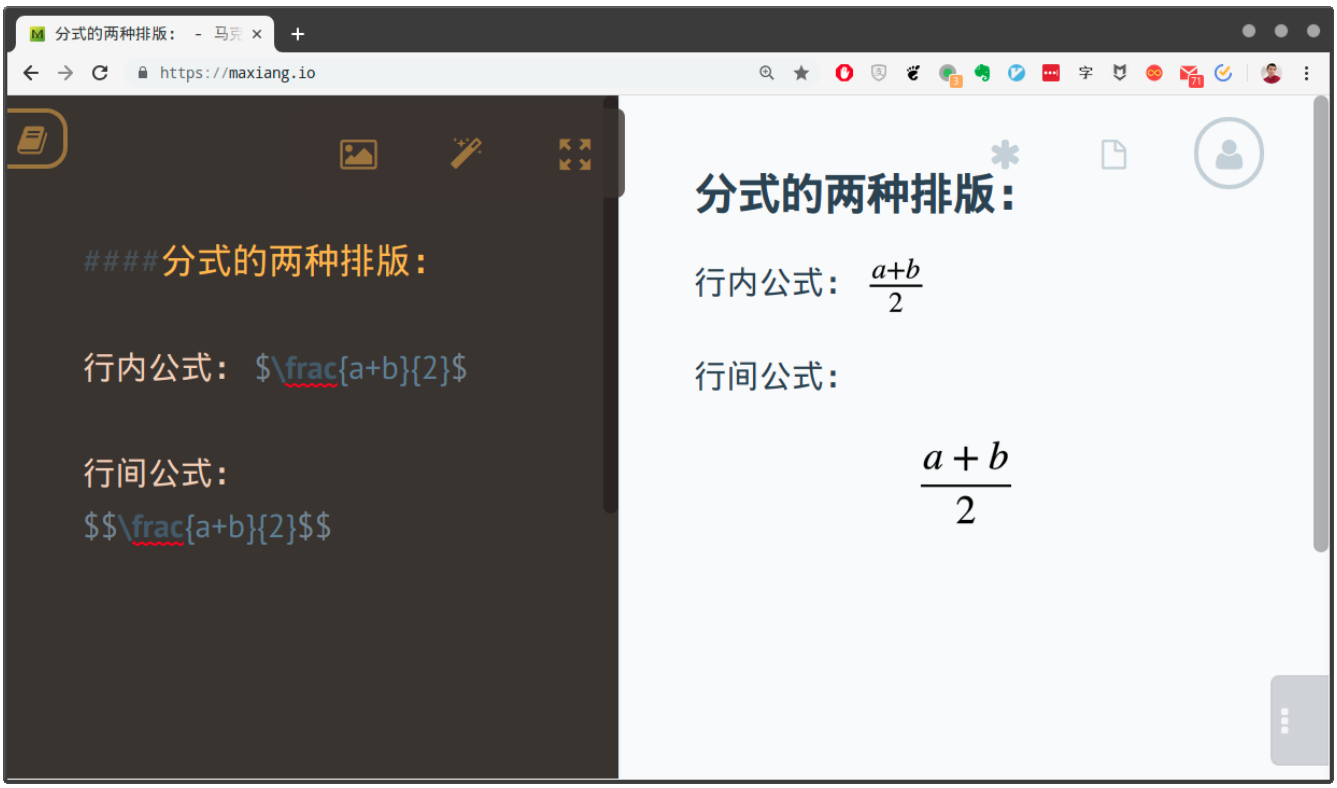
公式编码	渲染效果
`$a+c-\overbrace{b \times d}$`	$a+c-\overbrace{b \times d}$
`$a+c-\underbrace{b\times d}$`	$a+c-\underbrace{b\times d}$
`$a+c-\overbrace{b \times d}^{\text{乘法优先}}$`	$a+c-\overbrace{b \times d}^{\text{乘法优先}}$
`$a+c-\underbrace{b \times d}_{\text{乘法优先}}$`	$a+c-\underbrace{b \times d}_{\text{乘法优先}}$

4.2.3.3 分式与根式

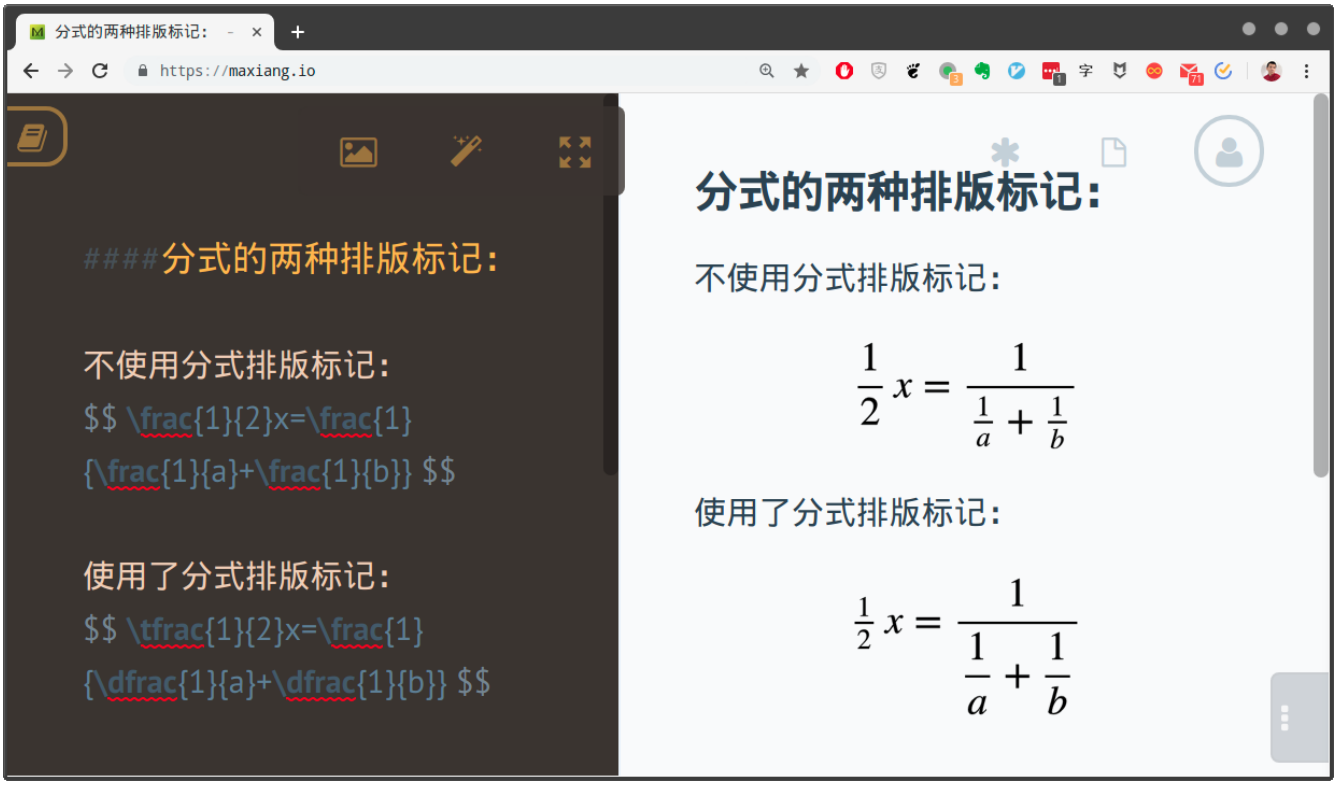
由于**分式**与**根式**都属于数学公式中极为常见的结构，并且它们在结构上也都相对简单，所以，我们打算将它们放在同一节中来介绍。先从**分式**开始，在$LaTeX$中，分式是通过`\frac`标记来实现的，其第一个参数为分子，第二个参数为分母。下面，我们来演示一下分式标记的具体使用：

公式编码	渲染效果
`$\frac{1}{2}$`	$\frac{1}{2}$
`$\frac{1}{a}$`	$\frac{1}{a}$
`$\frac{1}{a+b}$`	$\frac{1}{a+b}$
`$\frac{a+b}{2}$`	$\frac{a+b}{2}$

请注意，我们在表格中使用的都是分式的行内公式形态，如果拿它们对比一下分式的行间公式形态，就会发现其行内公式形态的分子和分母都使用了较小的字号，以配合普通文本的行高和行间距，以影响排版的整体质感，譬如：



当然，有时在行间公式中，我们也会想要让公式某一部分的字体小一点，采用行内公式形态，某一部分的字体大一点，采用行间公式形态。这时候，我们就可以使用`amsmath`宏组中提供的`\dfrac`（行间形态）和`\tfrac`（行内形态）来具体指定。譬如通过如下公式的比对，我们可以看到使用指定标记之后的效果：



除此之外，我们还会看到一些类似的、分上下两层的公式结构，譬如二项式系数，它使用的是`\binom`标记，用法与分式标记完全一致，这里就不一一累述了。下面，我们来看看**根式**的结构，在LaTeX中，根式是通过`\sqrt`标记来完成，该命令标记除了用于表示被开方数的指定参数外，还可以通过一个可选参数来表示开方的次数。我们来演示一下根式标记的具体使用：

公式编码	渲染效果
`$\sqrt{4}$`	$\sqrt{4}$
`$\sqrt[3]{8}$`	$\sqrt[3]{8}$
`$\sqrt[n]{a+b}$`	$\sqrt[n]{a+b}$
`$\sqrt[n]{\frac{1}{a+b}}$`	$\sqrt[n]{\frac{1}{a+b}}$

同样地。我们也可以使用`amsmath`宏组中提供的标记，对根式的排版样式做一些指定。例如，在一般状况下，根式的高度是随着其内容变化的，但当我们需要将几个根式并列出现在同一个公式中时，就会希望这些根式的高度是一致的，这时候就会需要用到`\vphantom`标记。我们可以来比对一下该标记使用之后的效果：

再例如，我们还可以通过`\uproot`和`\leftroot`这两个标记来调整开方次数所显示的位置，这两个命令标记的参数允许我们输入基于标准位置的偏移单位（正数代表上移或左移，负数代表下移或右移），以调整位置：

4.2.3.4 矩阵公式

现在我们来介绍最后一种公式结构：**矩阵**。在$\LaTeX$中，矩阵不同于一般的公式结构，它是一个独立的排版单元。因此在编辑矩阵时，我们需要先用单元定义标记`\begin`和`\end`来定义一个样式为`matrix`的独立单元。例如，下面是两个$2*2$的矩阵在执行加法运算：

```
$$
    \begin{matrix}
        a & b \\
        c & d \\
    \end{matrix}
    +
    \begin{matrix}
        a & b \\
        c & d \\
    \end{matrix}
$$
```

正如你所见，在上述矩阵的定义过程中，不同的列之间用符号“`&`”分隔，不同的行之间则用“`\\`”分割，其渲染效果如下：

$$\begin{matrix} a & b \\ c & d \end{matrix} + \begin{matrix} a & b \\ c & d \end{matrix}$$

但是，我们发现这样的矩阵在样式上实在太过简陋，尤其在当我们像上面一样，将多个矩阵并列写在同一个公式中时，它们似乎应该要有个边框会更好。所以，如今我们更常用的是`amsmath`宏组中提供的其他五种矩阵样式，下面我们就来看看这些矩阵样式与原生样式的对比：

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

这些标记的方法大同小异，只要将`\begin`和`\end`中的`matrix`参数改成相应的矩阵样式名称即可。比如，如果我们想为之前参与加法运算的矩阵加上一个中括号，就可以这样做：

```
$$
    \begin{bmatrix}
        a & b \\
        c & d \\
    \end{bmatrix}
    +
    \begin{bmatrix}
        a & b \\
        c & d \\
    \end{bmatrix}
$$
```

其渲染效果如下：

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix} + \begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

当然，如果某些矩阵过大，且无需将其中的元素全部输入，也可以在矩阵中使用`\cdots`、`\ddots`、`\vdots`这三个标记来输入一些省略号，以表示这些被省略的元素。例如：

```
$$
    \begin{bmatrix}
        1 & a_1 & a_1^2 & \cdots & a_1^n \\
        1 & a_2 & a_2^2 & \cdots & a_2^n \\
        \vdots & \vdots & \vdots & \ddots & \vdots \\
        1 & a_m & a_m^2 & \cdots & a_m^n \\
    \end{bmatrix}
$$
```

在这里，`\cdots`代表的是横向省略号、`\ddots`则代表了斜向省略号、而`\vdots`则代表纵向省略号，其渲染效果如下：

$$\begin{bmatrix} 1 & a_1 & a_1^2 & \cdots & a_1^n \\ 1 & a_2 & a_2^2 & \cdots & a_2^n \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & a_m & a_m^2 & \cdots & a_m^n \end{bmatrix}$$

在矩阵中0元素比较多的情况下，我们还可以使用嵌套矩阵的方法来编写一个分块矩阵，例如：

```
$$
    \begin{bmatrix}
        \begin{matrix} 1&1\\0&1 \end{matrix} & \Large{0} \\
        \Large{0} & \begin{matrix} 1&1\\0&1 \end{matrix} \\
    \end{bmatrix}
$$
```

如你所见，我们实际上是让内嵌矩阵成为了外层矩阵的元素，其渲染效果如下：

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

以上这些矩阵采用的都是行间公式的形态，但我们有时候也会在普通文本中编写一些简单的行内矩阵，这就需要用很小的字体来显示矩阵，这时候就可以采用`amsmath`宏组中提供的`smallmatrix`行内矩阵样式。当然，由于这种矩阵样式并没有设置括号，所以我们要手工为其添加一对括号，例如：

```
我们也可以用$\left[ \begin{smallmatrix} a & -b \\ b & a \end{smallmatrix} \right]$这样的矩阵来表示一个复数。
```

其渲染效果如下：

我们也可以用$\left[\begin{smallmatrix} a & -b \\ b & a \end{smallmatrix} \right]$这样的矩阵来表示一个复数。

4.2.3 数学公式中的符号

在掌握了以上几种公式结构之后，我们基本上能够构建起一个数学公式的主要骨干了。但是，数学公式具体所要计算的内容则还是需要其使用的具体数学符号来决定。在$\LaTeX$中，我们可以将数学公式中的符号按照作用分成三大类，它们分别用于标记字母与字体、运算与关系、分隔与连接这三种数学语义。在这一节中，我们就来介绍一下这三大类数学符号中比较常用的部分符号。如果读者想要知道全部的数学符号，还需要自行去查阅相关的$\TeX$文档。

4.2.3.1 字母与字体

让我们先从字母开始，数学公式中使用的字母主要为拉丁字母和希腊字母。其中，拉丁字母就是英文所用的字母，这26个字母（包括其大小写形态）都可以直接从键盘输入，不需要特定的标记来表示。但希腊字母的情况就要复杂一些了，除了部分字母与拉丁字母一致，可以直接用键盘输入外，其他就必须要用特定的$\LaTeX$标记来表示了，其字母表如下：

字母标记	显示效果	字母标记	显示效果	字符标记	显示效果	字符标记	显示效果
`\alpha`	α	`A`	A	`\beta`	β	`B`	B
`\gamma`	γ	`\Gamma`	Γ	`\delta`	δ	`\Delta`	Δ
`\epsilon`	ϵ	`E`	E	`\zeta`	ζ	`Z`	Z
`\eta`	η	`H`	H	`\theta`	θ	`\Theta`	Θ
`\iota`	ι	`I`	I	`\kappa`	κ	`K`	K
`\lambda`	λ	`\Lambda`	Λ	`\mu`	μ	`M`	M
`\nu`	ν	`N`	N	`\xi`	ξ	`\Xi`	Ξ
`o`	o	`O`	O	`\pi`	π	`\Pi`	Π
`\rho`	ρ	`P`	P	`\sigma`	σ	`\Sigma`	Σ
`\tau`	τ	`T`	T	`\upsilon`	υ	`\Upsilon`	Υ
`\phi`	ϕ	`\Phi`	Φ	`\chi`	χ	`X`	X
`\psi`	ψ	`\Psi`	Ψ	`\omega`	ω	`\Omega`	Ω

这其中，部分希腊字母还有变量专用形态，它们的$\LaTeX$标记通常以`\var-`开头，具体如下：

小写形态	大写形态	变量形态	显示效果
`\epsilon`	`E`	`\varepsilon`	$\epsilon \mid E \mid \varepsilon$
`\theta`	`\Theta`	`\vartheta`	$\theta \mid \Theta \mid \vartheta$
`\rho`	`P`	`\varrho`	$\rho \mid P \mid \varrho$
`\sigma`	`\Sigma`	`\varsigma`	$\sigma \mid \Sigma \mid \varsigma$
`\phi`	`\Phi`	`\varphi`	$\phi \mid \Phi \mid \varphi$

下面，我们再来看看这些字母使用的字体。在通常情况下，数学公式中的变量采用的是斜体的意大利体，常数采用的是直体的罗马体。当然，在特定情况下，我们也可以用以下$\LaTeX$标记来指定相关字母的字体：

字体标记	字体名称	显示效果	字体标记	字体名称	显示效果
`\rm`	罗马体	Sample	`\cal`	花体	$\mathcal{SAMPLE}$
`\it`	意大利体	*Sample*	`\Bbb`	黑板粗体	$\mathbb{SAMPLE}$
`\bf`	粗体	**Sample**	`\mit`	数学斜体	SAMPLE
`\sf`	等线体	Sample	`\scr`	手写体	$\mathscr{SAMPLE}$
`\tt`	打字机体	`Sample`			
`\frak`	旧德式字体	$\mathfrak{Sample}$			

指定字体的方式很简单，我们只需要在数学公式中以`{\字体{Sample}}`的形式使用上述字体标记，就可以将“Sample”这几个字符设置为指定字体了。除此之外，在必要情况下，我们还可以使用`\color`标记来指定这些字体的颜色，该标记的第一个参数为颜色的名称，第二个参数是被指定颜色的字符，譬如`$\color{black}{Sample}$`，就是将“Sample”这几个字符设置为`black`这个颜色（当然，这些颜色主要是面向Web浏览器的，对印刷品没有多大的意义）。在这里，可供我们选择的颜色如下：

颜色名称	显示效果	颜色名称	显示效果
black	$\color{black}{Sample}$	grey	$\color{grey}{Sample}$
silver	$\color{silver}{Sample}$	white	$\color{white}{Sample}$
maroon	$\color{maroon}{Sample}$	red	$\color{red}{Sample}$
yellow	$\color{yellow}{Sample}$	lime	$\color{lime}{Sample}$
olive	$\color{olive}{Sample}$	green	$\color{green}{Sample}$
teal	$\color{teal}{Sample}$	auqa	$\color{auqa}{Sample}$
blue	$\color{blue}{Sample}$	navy	$\color{navy}{Sample}$
purple	$\color{purple}{Sample}$	fuchsia	$\color{fuchsia}{Sample}$

4.2.3.2 运算符

在了解了数学公式中所用的字母之后，接下来就可以关注公式中的运算符了。在这些运算符中，除了加减法以及上一节中介绍的乘方、开方、分式、矩阵这些具有特定结构的运算符之外，其它各种运算的符号基本也都有特定的$\LaTeX$标记。下面，我们继续来介绍一些常见的数学符号及其用法。

对数运算

在数学中，对数运算是指数运算（即乘方运算）的逆运算，为最常见的数学运算之一，下面我们来演示一下如何用$LaTeX$标记来描述该运算，首先是该运算的运算符：

运算符标记	显示效果	运算符标记	显示效果	运算符标记	显示效果
`\log`	$\log$	`\lg`	$\lg$	`\ln`	$\ln$

接下来，我们就可以试着来编写一些常见的对数运算公式：

公式名称	公式编码	显示效果
自然对数	`$\ln a$`	$\ln a$
常用对数	`$\lg a$`	$\lg a$
和差公式	`$\log_{a} MN=log_{a} M+log_{a} N$`	$\log_a MN=log_aM+log_aN$
互换公式	`$M^{\log_{a} N}=N^{log_{a} M}$`	$M^{\log_a N}=N^{log_aM}$
换底公式	`$\log_{a} N=\frac{\log_{b}N}{log_{b} a}$`	$\log_a N=\frac{\log_b N}{log_ba}$

三角运算

三角形是平面几何的基础，它有着一套独特的数学表示方法。现在，我们要来看看如何用$LaTeX$标记来表述这套表示方法。同样的，首先是一些符号的标记：

运算符标记	显示效果	运算符标记	显示效果	运算符标记	显示效果
`30^\circ`	30°	`\bot`	$\bot$	`\angle A`	$\angle A$
`\sin`	$\sin$	`\cos`	$\cos$	`\tan`	$\tan$
`\csc`	$\csc$	`\sec`	$\sec$	`\cot`	$\cot$

接下来，我们用A、B、C来表示三角形的三条边，用α、β、γ来表示三角形的三个角，并以R为三角形外接圆的半径来示范一下三角形问题的描述：

公式名称	公式编码	显示效果
三角之和	`$\alpha+\beta+\gamma=180^\circ$`	$\alpha+\beta+\gamma=180^\circ$
正弦定理	`$\frac{A}{\sin\alpha}=\frac{B}{\sin\beta}=\frac{C}{\sin\gamma}=2R$`	$\frac{A}{\sin\alpha}=\frac{B}{\sin\beta}=\frac{C}{\sin\gamma}=2R$
余弦定理	`$C^2=A^2+B^2-2AB*\cos\alpha$`	$C^2=A^2+B^2-2AB*\cos\alpha$

比较运算

比较运算也是数学的一个重要分支，它在计算机程序设计领域中都有着重要的作用。我们在描述算法等问题时也都会需要表述比较运算。该运算除了`>`、`<`、`=`这些可直接从键盘输入的运算符之外，也有一些运算符是需要用$\LaTeX$标记来表示的：

运算符标记	显示效果	运算符标记	显示效果	运算符标记	显示效果
`\not<`	$\not<$	`\not>`	$\not>$	`\not=`	$\neq$
`\le`	$\leq$	`\ge`	$\geq$	`\approx`	$\approx$
`\equiv`	$\equiv$				

下面照例来做几个演示：

运算编码	显示效果
`if $a \not< b$, then $a \ge b$.`	if $a \not< b$, then $a \geq b$.
`if $a \not> b$, then $a \le b$.`	if $a \not> b$, then $a \leq b$.
`if $a \approx b$, then $a \not= b$.`	if $a \approx b$, then $a \neq b$.

集合运算

集合运算是离散数学的基础，在计算机科学领域有着非常重要的作用，我们在写计算机论文时少不了要描述这类运算，下面是一些集合运算符的$\LaTeX$标记：

运算符标记	显示效果	运算符标记	显示效果	运算符标记	显示效果
`\emptyset`	$\emptyset$	`\in`	$\in$	`\notin`	$\notin$
`\subset`	$\subset$	`\supset`	$\supset$	`\subseteq`	$\subseteq$
`\supseteq`	$\supseteq$	`\cap`	$\cap$	`\cup`	$\cup$
`\bigvee`	$\bigvee$	`\bigwedge`	$\bigwedge$	`\biguplus`	$\biguplus$
`\forall`	$\forall$	`\exists`	$\exists$	`\not\subset`	$\not\subset$

在这里，我们可以用上述标记来编写几条集合运算的基本法则，以作演示：

公式名称	公式编码	显示效果
同一律	`$A \cup \emptyset =A$`	$A \cup \emptyset =A$
交换律	`$A \cap B=B \cap A$`	$A \cap B=B \cap A$
结合律	`$(A \cup B) \cup C=A \cup (B \cup C)$`	$(A \cup B) \cup C=A \cup (B \cup C)$
分配律	`$(A \cap B) \cup C=(A \cup C ) \cap (B \cup C)$`	$(A \cap B) \cup C=(A \cup C) \cap (B \cup C)$

微积分运算

微积分是高等数学的入门学科，内容主要包括极限、微分学、积分学及其应用。下面是我们在描述微积分运算时会用到的一些$LaTeX$标记：

运算符标记	显示效果	运算符标记	显示效果	运算符标记	显示效果
`\int`	$\int$	`\iint`	$\iint$	`\iiint`	$\iiint$
`\iiiint`	$\iiiint$	`\oint`	$\oint$	`\prime`	$\prime$
`\lim`	$\lim$	`\infty`	∞	`\nabla`	∇

我们不妨在这里复习一下当年在大学一年级时写过的那些公式，只不过当时应该是手写为主，现在我们要用的是$LaTeX$标记：

公式名称	公式编码	显示效果
不定积分	`$\int f(x) {\rm{d} x}$`	$\int f(x) {\rm{d} x}$
定积分	`$\int^b_a f(x) {\rm{d} x}$`	$\int^b_a f(x) {\rm{d} x}$
微分方程	`$\frac{{\rm{d} x(t)}}{x(t)}=f(x) {\rm{d} x}$`	$\frac{{\rm{d} x(t)}}{x(t)}=f(x) {\rm{d} x}$
极限运算	`$\lim_{n \to \infty} \frac{1}{n(n-1)}$`	$\lim_{n \to \infty} \frac{1}{n(n-1)}$

当然，除上述运算外，我们在描述数学问题过程中还可能会用到如下这些运算符：

运算符标记	显示效果	运算符标记	显示效果	运算符标记	显示效果
`\pm`	$\pm$	`\times`	$\times$	`\div`	$\div$
`\mid`	$\mid$	`\nmid`	$\nmid$	`\cdot`	$\cdot$
`\circ`	$\circ$	`\ast`	$\ast$	`\bigodot`	$\bigodot$
`\bigotimes`	$\bigotimes$	`\bigoplus`	$\bigoplus$	`\sum`	$\sum$
`\prod`	$\prod$	`\coprod`	$\coprod$	`\backslash`	$\backslash$
`\because`	$\because$	`\therefore`	$\therefore$		

这些$\LaTeX$标记的使用与上述运算大同小异，这里就不一一示范了，读者可自行查阅相关文档。

4.2.3.3 定界符

在数学公式中，处理参与运算的字母变量、数字以及表明运算类型的运算符之外，以括号为代表的定界符也是一类非常重要的符号。众所周知，数学公式的编辑是离不开各种各样的括号的，我们需要用它们来凸显公式中的重点、对公式进行分组、甚至改变公式的运算顺序。在$\LaTeX$中，括号是由开符号和闭符号组成的，所以我们要分两个标记来表列$\LaTeX$所提供的括号种类，具体如下：

括号类型	开符号	闭符号	示例编码	显示效果
圆括号	`(`	`)`	`$a+(b-c)$`	$a+(b-c)$
方括号	`[`	`]`	`$a+[b-c]$`	$a+[b-c]$
花括号	`\lbrace`	`\rbrace`	`$a+\lbrace b-c \rbrace$`	$a+\{b-c\}$
尖括号	`\langle`	`\rangle`	`$a+\langle b-c \rangle$`	$a+\langle b-c\rangle$
向上取整	`\lceil`	`\rceil`	`$a+\lceil b-c \rceil$`	$a+\lceil b-c\rceil$
向下取整	`\lfloor`	`\rfloor`	`$a+\lfloor b-c \rfloor$`	$a+\lfloor b-c\rfloor$

由于数学结构的关系，我们有时候会希望括号的大小是可以随着公式结构变化的。为实现这个效果，我们需要在括号的开符号之前加上`\left`标记，闭符号之前加上`\right`标记。例如，对于下面这个公式：

> $$\alpha_x\alpha_y\left[\frac{1}{3}\left(x^2+y^{(2x-y)}\right)^2+xy\right]$$

我们需要在`Markdown`文档中输入如下编码：

```
$$
    \alpha_x \alpha_y \left[ \frac{1}{3} \left( x^2+y^{\left( 2x-y \right)} \right)^2 + xy \right]
$$
```

请注意，`\left`和`\right`这两个标记必须要位于同一逻辑行中，否则就无法配对，但它们所配对的定界符不一定是括号，甚至一个句点也可以的。当然，定界符也并非只有括号这种配对的符号，如下公式中的竖线和斜线也都属于定界符：

定界符	公式编码	显示效果
`/`	`$f(x) / x-1$`	$f(x)/x-1$
`\backslash`	`$f(x) \backslash x-1$`	$f(x)\backslash x-1$
`\vert`	`$f(x) \vert x-1$`	$f(x)\vert x-1$
`\Vert`	`$f(x) \Vert x-1$`	$f(x)\Vert x-1$

对这种不成对的单一定界符，我们也可以使用`\middle`标记令其根据公式的结构调整自身大小。例如，对于下面这个公式：

$$\Pr\left(X>\frac{1}{3}\middle|Y=0\right)=\int_0^1 p(t)\,\mathrm{d}t\Big/(\mathrm{N}^2+1)$$

我们只需要在`Markdown`文档中输入如下编码：

```
$$
   \Pr \left( X>\frac{1}{3} \middle\vert Y=0 \right) = \left. \int_0^1 p(t)\, \rm{d}t \middle/ (N^2+1) \right.
$$
```

4.2.4 多行公式

如果我们在撰写科学性文章时讨论的数学问题都是用行内公式，或者单行的行间公式就能表述的，那么即使用Office Word中的相关工具来编辑这些公式也不是一件多么让人不可接受的事情。但现实是残酷的，哪怕我们要演示一个开方运算的过程，其演算列表也少不了要连续写个五六行，更何况我们还会遇到长度足以需要换行的长公式，以及根据参数条件分组的条件公式。只有在编写这些公式时，我们才能真正体会到$\LaTeX$的强大。在这一节中，我们就来讨论一下多行公式的编辑。

和矩阵一样，多行公式在$LaTeX$中也被视为一个独立的排版单元。所以，我们首先需要用单元定义标记`\begin`和`\end`定义一个公式单元。同样的，在定义公式单元时也需要指定该单元的样式，我们最常用的公式样式有两种：

- 第一种是`gather`，在这种样式下，公式单元中的所有公式都会居中对齐，这也是普通的样式。例如，如果想撰写下面这样的一个三元线性方程组：

  > $$\begin{aligned}\alpha_{11}x+\alpha_{12}y+\alpha_{13}z&=A\\\alpha_{21}x+\alpha_{22}y+\alpha_{23}z&=B\\\alpha_{31}x+\alpha_{32}y+\alpha_{33}z&=C\end{aligned}$$

  我们就可以在`Markdown`文档中输入如下编码：

  ```
  $$
    \begin{gather}
        \alpha_{11}x+\alpha_{12}y+\alpha_{13}z=A \\
        \alpha_{21}x+\alpha_{22}y+\alpha_{23}z=B \\
        \alpha_{31}x+\alpha_{32}y+\alpha_{33}z=C \\
    \end{gather}
  $$
  ```

- 第二种是`align`，它会让公式单元中的所有公式按用`&`符指定的符号对齐。例如，如果想撰写下面这样的一个演算列表：

  > $$\begin{aligned}\sqrt{37}&=\sqrt{\frac{73^2-1}{12^2}}\\&=\sqrt{\frac{73^2}{12^2}\cdot\frac{73^2-1}{73^2}}\\&=\sqrt{\frac{73^2}{12^2}}\sqrt{\frac{73^2-1}{73^2}}\\&=\frac{73}{12}\sqrt{1-\frac{1}{73^2}}\\&\approx\frac{73}{12}\left(1-\frac{1}{2\cdot73^2}\right)\end{aligned}$$

  我们就需要使用`align`样式，令其按等号对齐，具体编码如下：

```
$$
  \begin{align}
      \sqrt{37} & = \sqrt{\frac{73^2-1}{12^2}} \\
      &= \sqrt{\frac{73^2}{12^2}\cdot\frac{73^2-1}{73^2}} \\
      &= \sqrt{\frac{73^2}{12^2}}\sqrt{\frac{73^2-1}{73^2}} \\
      &= \frac{73}{12}\sqrt{1 - \frac{1}{73^2}} \\
      &\approx \frac{73}{12}\left(1 - \frac{1}{2\cdot73^2}\right)
  \end{align}
$$
```

当然，我们还可以使用`flalign`和`alignat`这两种样式，使用方式大同小异，读者可以自行查阅相关文档。下面，我们来看看如何撰写按条件分组的公式。事实上，这种公式的写法就是在一个单行的公式中嵌套一个样式为`cases`的多行公式单元。例如，如果想撰写这样一个按奇偶数分组的公式：

$$
f(n) = \begin{cases} n/2, & \text{若}n\text{为偶数} \\ 3n+1, & \text{若}n\text{为奇数} \end{cases}
$$

我们就只需要在`Markdown`文档中输入如下编码即可：

```
$$
  f(n) =
      \begin{cases}
          n/2,  & \text{若$n$为偶数} \\
          3n+1, & \text{若$n$为奇数} \\
      \end{cases}
$$
```

同样的，`cases`只是分组公式单元最常见的一种样式，如果我们能引入`mathtools`宏组的话，还能使用`gathered`、`aligned`、`alignedat`、`multlined`等样式，它们的使用方式也大同小异，读者可以自行查阅相关文档，这里就不一一累述了。

在研究完如何罗列多个公式之后，让我们最后再来回头关心一下应该如何将单一公式分行显示。其实分行显示一个公式的方法也非常简单，就是先将公式单元定义为`gather`、`align`等样式，然后在编辑公式过程中，用`\\`来分行即可。例如，如果想撰写下面这样的一个公式：

$$
\begin{aligned}
a+b+c+d+e \\
+f+g+h+i \\
+j+k+l+m \\
+n+o+p+q \\
+r+s+t+u \\
+v+w+x+y+z
\end{aligned}
$$

我们就只需要在`Markdown`文档中输入如下编码即可：

```
$$
  \begin{align}
        a&+b+c+d+e \\
         &+f+g+h+i \\
         &+j+k+l+m \\
         &+n+o+p+q \\
         &+r+s+t+u \\
         &+v+w+x+y+z
  \end{align}
$$
```

本章小结

在这一章中，我们首先介绍了$\LaTeX$这门专用于排版的标记语言，包括它的前世今生，标记的基本语法形式，以及如何在`Markdown`文档中使用$\LaTeX$标记。然后，我们分公式结构、公式符号以及多行公式编辑三个主题详细而深入地介绍了如何在`Markdown`文档中表述数学问题。

到目前为止，`Markdown`在写作阶段的应用就基本介绍完了。在接下来的内容中，我们将介绍如何将`Markdown`文档转换成`docx`、`PDF`等主流的文档格式，或者制作成电子书或博客分享给读者，以及如何与合作者进行线上协作，同步修改，并在修改过程中完成文档的版本控制。

---

1. 注释：高德纳教授是现代计算机科学的先驱人物，他创立了算法分析理论，并在数个计算机理论分支上都做出了犹如基石一般的贡献，于1974年荣获图灵奖。↩

2. 注释：莱斯利·兰伯特是来自纽约的一位计算机科学家，$\LaTeX$排版系统的开发者，2013年荣获图灵奖。↩

3. 注释：官方网站：https://www.mathjax.org↩

4. 注释：amsmath宏组是LaTeX中最常用的组件之一，由美国数学协会（AMS）设计开发，它全面扩展了LaTeX的数学表述能力，目前已经成为了LaTeX中的必备组件。↩