

```
\newcommand{\BigFig}[1]{\parbox{12pt}{\Huge #1}}
```

GeoGebra 的 L^AT_EX 语法

GeoGebra of L^AT_EX

```
\begin{equation*}
```

```
(a_{k1})=\left(
```

```
\begin{matrix}
```

```
0 \ldots 0 & 1 & 0 \ldots 0 \\
```

```
& 0 \\
```

```
\BigZero & \cdots & \BigZero
```

```
& 0 \\
```

```
\end{matrix}
```

```
(right)
```

```
\quad
```

```
f(x)=
```

```
\begin{cases}
```

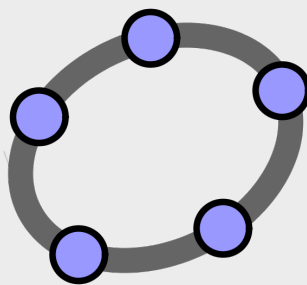
```
x^2,& \text{if } x < 0; \\
```

```
\alpha + x,& \text{if } 0 \leq x \leq 1; \\
```

```
x^2, & \text{otherwise.}
```

```
\end{cases}
```

```
\end{equation*}
```



GeoGebra Documentation Workshop.

Version: 1.01

Author: 段明

November , 2015 · KunMing

写在前面

发现大家对 GeoGebra ^① 这一软件的使用交流中，很多人对此软件所支持的 \LaTeX 这一语言产生了兴趣，而 \LaTeX 本身的功能过于庞大，学习时间过长，很多人需要的仅是其中输入数学公式这一部分，于是便萌生了写这个文档的念头。

此文档不是 GeoGebra 的入门教程，也不是 \LaTeX 的入门教程，其目的并不是让大家学会并掌握 \LaTeX 这一庞然大物，而仅仅只是针对在 GeoGebra 这一软件中数学公式的一些基础的、常用的输入方法进行介绍，希望能对大家的学习和工作起到微弱的作用并达到抛砖引玉的目的，以期待更多的高手能合力完善此文档。

此稿在初稿的基础上做了少量的修改，比如增加了字号设置，介绍了在“标题”选择中使用 \LaTeX 的方法，改进了示例的代码，修改了版面显示效果等内容。

段明 ^②

2015 年 11 月于昆明

^①官方网站：<http://www.geogebra.org/>

^②发现错误或者是需要改进的地方，请告诉我，邮箱：[surfand@126.com](mailto:surfan@126.com)，谢谢！

目 录

写在前面	II
第一章 TeX 简介	1
1.1 TeX 家族	1
1.1.1 L ^A TeX 简介	1
1.1.2 引擎: TeX	2
1.1.3 格式: L ^A TeX	2
1.1.4 宏包	3
1.1.5 驱动	3
1.1.6 小结	3
1.2 优点和缺点	3
1.2.1 选择 TeX 的理由	4
1.2.2 TeX 的缺陷	4
1.2.3 不选择 TeX 的理由	4
1.3 软件准备	5
第二章 几点说明	6
2.1 L ^A TeX 命令	6
2.1.1 命令区分大小写	6
2.1.2 命令的写法	6
2.1.3 环境	7
2.2 几个常用的命令	7
2.2.1 换行命令	7

2.2.2	横线和竖线	7
2.2.3	给公式加个框	7
2.2.4	撇号的输入	8
2.2.5	带圈字符的输入	8
2.3	特殊符号的输入	8
2.4	空白间距	9
2.5	字号的修改	9
2.6	行内公式和行间公式	10
2.6.1	行内公式	10
2.6.2	行间公式	10
2.7	如何使用 \LaTeX	11
2.7.1	在 GeoGebra 中使用 \LaTeX	11
2.7.2	“标题”中使用 \LaTeX	12
2.7.3	在其它软件中使用 \LaTeX 公式	12
第三章	数学公式	14
3.1	基本元素	14
3.1.1	希腊字母	14
3.1.2	上标和下标	15
3.1.3	虚位	15
3.1.4	分数和根号	16
3.1.5	二项式结构	16
3.1.6	运算符	17
3.1.7	箭头	19
3.1.8	分隔符	19
3.1.9	省略号	20
3.2	多行公式	21
3.2.1	长公式	21
3.2.2	公式组	21

3.2.3	分段函数	22
3.3	矩阵	23
3.3.1	无分隔符的矩阵	23
3.3.2	带分隔符的矩阵	23
3.3.3	较复杂的矩阵示例	24
3.4	注音和标注	27
3.5	常用符号	28
3.6	数学字体	28
3.7	标准数学函数	29
3.8	简单表格	29
3.9	一些示例	30
3.9.1	上下标综合应用	30
3.9.2	利用虚位来对齐等式	30
3.9.3	让人眼花的结构	31
3.9.4	和集合有关的公式	31
3.9.5	符号上下方的标注	32
3.9.6	背景色和文字颜色	32
3.9.7	将下标分行	33

第一章 $\text{T}_{\text{E}}\text{X}$ 简介

天下之事，
其得之不难，则失之必易；
其积之不久，则其发之必不宏。
--（明）王阳明

目标与要求

1. 了解 $\text{T}_{\text{E}}\text{X}$ 的产生及发展.
2. 了解 $\text{T}_{\text{E}}\text{X}$ 和 $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$ 的关系.
3. 了解常见发行版和编辑器.
4. 知道文本模式下 $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$ 的写法.

1.1 $\text{T}_{\text{E}}\text{X}$ 家族

1.1.1 $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$ 简介

$\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$ 是一种面向数学和其它科技文档的电子排版系统，我们通常据说的 $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$ 是一个总称，包括 $\text{T}_{\text{E}}\text{X}$ 、 $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$ 、 $\mathcal{A}\mathcal{M}\mathcal{S}\text{-L}_{\text{A}}\text{T}_{\text{E}}\text{X}$ 等，它们是免费软件，包括源程序也是免费的。

目前为止， $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$ 几乎在所有的计算机操作系统平台上得到实现，用 $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$ 编辑的源文件可在不同的平台之间自由交换，而得到的输出是完全相同的。

在文本模式下， $\text{T}_{\text{E}}\text{X}$ 和 $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$ 应该写作：“ TeX 和 LaTeX ”，在 GeoGebra 中可以用命令： $\backslash\text{LaTeX}$ 得到 $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$ 。

个人认为，要学好任何一门学科都应该学习其历史，这样才能更好的对所学知识加深理解，并有利于把所学知识融会贯通，所以在这里先非常简单的介绍一点点 $\text{T}_{\text{E}}\text{X}$ 方面的知识，如

果要了解更多的内容请自行搜索，网上有非常详细的介绍，包括其作者的一些传奇经历，这里所收集整理知识也主要来源于网络。

1.1.2 引擎： $\text{T}_\text{E}\text{X}$

1976 年，Donald Ervin Knuth (1938 –)^①为了排版自己的著作《The Art of Computer Programming》（计算机程序设计艺术）而开发的一个全新的排版引擎，这就是 $\text{T}_\text{E}\text{X}$ ，从这几个交错起伏的字母中便道出了“排版”二字的几分意味：精确、复杂、注重细节和品位，这里所说的引擎，是指能够实现断行、分页等操作的程序。

$\text{T}_\text{E}\text{X}$ 是一个相当稳定而几乎没有 bug 的系统，它的版本号趋近于 π ，每发布一个修正版，版本号就增加一位小数，当前版本是 2014 年 1 月更新的 3.14159265，在 $\text{T}_\text{E}\text{X}$ 四十多年的历史中，这仅仅是第九次更新，上次更新在 2008 年。

2004 年在 $\text{T}_\text{E}\text{X}$ 的基础上出现了一个新的引擎 $\text{X}_\text{Y}\text{T}_\text{E}\text{X}$ ，该引擎对中日韩等东亚字体的支持非常友好，现在是处理中文文档的首选，注意在使用 $\text{X}_\text{Y}\text{T}_\text{E}\text{X}$ 引擎时，字符编码请选择 UTF-8。

1.1.3 格式： $\text{L}_\text{A}\text{T}_\text{E}\text{X}$

基本的 $\text{T}_\text{E}\text{X}$ 系统只有 300 多个元命令 (primitive)，十分精悍，但很难读懂，只适于非正常人类，所以 Knuth 提供了一种格式 (format，宏命令的集合) 对 $\text{T}_\text{E}\text{X}$ 进行了封装，这就是 Plain $\text{T}_\text{E}\text{X}$ ，包含 600 多个宏命令，然而它还是不够高级。

1980 年代初期，斯坦福研究所 (SRI) 的 Leslie Lamport (1941 –)^②开发了一种新的格式，也就是 $\text{L}_\text{A}\text{T}_\text{E}\text{X}$ ，实际上 $\text{L}_\text{A}\text{T}_\text{E}\text{X}$ 是基于 plain $\text{T}_\text{E}\text{X}$ 定义的格式，它定义了许多新的控制命令并封装成一个可执行文件，这个可执行文件会把新定义的命令解释成为 $\text{T}_\text{E}\text{X}$ 的控制命令，并最终交由 $\text{T}_\text{E}\text{X}$ 引擎进行排版。1994 年在 2.09 版本的基础上发布了 $\text{L}_\text{A}\text{T}_\text{E}\text{X} 2_\epsilon$ ，这也是我们现在使用得最多的版本， $\text{L}_\text{A}\text{T}_\text{E}\text{X} 3$ 的开发也在进行中，只是正式版看起来遥遥无期。

^①Knuth 教授自己取了个中文名叫：高德纳，1960 年凯斯工学院数学学士，因为长得帅同时获赠硕士，1963 年加州理工数学博士，同年留校任教。1968 年跳槽到斯坦福，1974 年获图灵奖，1992 年退休，1995 年获冯·诺依曼奖。

^②1960 年 MIT 数学学士，布兰迪斯大学数学系 1963 年硕士，1972 年博士。1970 年加入麻省计算机同伙公司，1977 年跳槽到 SRI，1985 年跳到 DEC，2001 年跳到微软。2008 年获冯·诺依曼奖。

1.1.4 宏包

\LaTeX 出现之后, 在它的基础上出现了很多宏包 (package), 这些宏包丰富和加强了 \TeX 的功能, 比如提供数学功能的 `amsmath` 宏包, 制作幻灯片的 `beamer` 宏包, 用于绘图的 `tikz` 宏包等等。

使用 \LaTeX 的排版时, 许多功能需要在程序的开始部分 (导言区) 引入用到的宏包, 比如中文支持需要 `ctex` 宏包, 如数学字体 \mathcal{ABCXYZ} 需要 `mathrsfs` 宏包, 在 GeoGebra 中已经作过相应处理, 不需要引用宏包, 可以直接使用相应的命令。

1.1.5 驱动

Knuth 最初设计的 \TeX 只能用于施乐图形打印机, 后来有人将其输出改为设备无关的格式, 也就是 DVI, 发展到现在已经有了更加丰富的输出格式, 用户需要用驱动程序把它转换为自己想要的格式, 比如 `XeLaTeX` 命令可以将源文件编译成 PDF 格式的文本。

1.1.6 小结

数字排版有四个重要环节: 标记语言、页面描述语言、光栅图像处理器、输出设备。 \TeX 是最精确、最高级的面向专业排版的标记语言, \TeX 家族可以划分为四个层次: 引擎、格式、宏包、驱动。

我个人比较喜欢选择 `XYTeX` 引擎和 \LaTeX 格式^③。

1.2 优点和缺点

\TeX 本身的领域是专业排版 (即方正书版、InDesign 的领域), 但现在 $\text{\TeX}/\text{\LaTeX}$ 也被广泛用于生成电子文档甚至幻灯片等, \TeX 语言的数学部分偶尔也在其他一些地方使用 (如 GeoGebra 中 $\wedge_^$), 但是要注意 \TeX 并不适用于文书处理 (MS Office 的领域)。

在众多大侠的不断努力下, \LaTeX 现在同样能运用于物理、化学、乐谱、棋谱等领域的排版 (这需要用不同的宏包)。

\TeX 称为所想即所得的系统, 它和 Office 这一类的所见即所得的软件各有优缺点。

^③新的引擎 `LuaTeX` 和格式 `ConTeXt` 旧版可用, 新版正在开发中, 离真正可用还需一点时间, 故不推荐大家现在使用。

1.2.1 选择 $\text{T}_\text{E}\text{X}$ 的理由

- 免费、开源软件；
- 高质量、专业的排版效果；
- 数学公式尤其赏心悦目，是事实上的专业数学排版标准；
- 结构化，它的文档结构清晰；
- 跨平台，几乎可以支持于所有电脑硬件和操作系统平台；
-

1.2.2 $\text{T}_\text{E}\text{X}$ 的缺陷

相应的， $\text{T}_\text{E}\text{X}$ 由于其工作流程，设计原则，资源的缺乏，以及历史局限性等原因也存在一些缺陷：

- 语法不如 HTML 和 XML 严谨、清晰；
- 制作过程繁琐，不能直接或实时看到结果；
- 宏包鱼龙混杂，水准参差不齐，风格不够统一；
- 相对于商业软件，用户支持不够好，文档不完善。

实时显示

关于实时显示这一情况，在很多高手的不断努力下，目前得到了很大改善，其中比较有名的是 LyX 这一免费开源软件，很多同学喜欢用它来记数学笔记，值得注意的是，使用 LyX 时并不要求一定要会 $\text{L}^{\text{A}}\text{T}_\text{E}\text{X}$ ，直接安装这一软件就可以使用。

GeoGebra 中的 $\text{L}^{\text{A}}\text{T}_\text{E}\text{X}$ 也支持实时显示，效果很好，此外还有一些编辑器也在实时显示方面不断的努力，进步是巨大的。

1.2.3 不选择 $\text{T}_\text{E}\text{X}$ 的理由

- 需要较多的精力学习；
- 图文混合排版能力弱；

- 仅流行于数学、物理、计算机等领域；
- 中文期刊的支持较差；
-

1.3 软件准备

初学者面对上述那些引擎、格式、宏包、驱动等概念可能手足无措，看得一头雾水，所幸有志愿者把这些东西连同一些实用程序打包集成在一起，形成了一个发行版。

要学习和使用 $\text{T}_{\text{E}}\text{X}$ 需要事先在电脑上安装一个发行版^④，如果只是要输入数学公式，那么可以不用安装 $\text{T}_{\text{E}}\text{X}$ 的发行版，用 GeoGebra 这一软件中的 $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$ 来输入公式也基本够用了，并且能实时看到生成的公式。

发行版的基本作用是提供 $\text{T}_{\text{E}}\text{X}$ 后台处理机制和命令程序，用户还需要一个前台编辑器来编辑源文件，可以使用普通的文本编辑器，也可以使用专用的 $\text{T}_{\text{E}}\text{X}$ 编辑器，常见的发行版和编辑器见 表 1.1。

表 1.1: 常见发行版和编辑器

操作系统	发行版	编辑器
Windows	MikTeX	TeXstudio
Linux	teTeX	Kile
Mac OS	MacTeX	TeXShop
通用	TeX Live	TeXworks

这里只举例说明了一个编辑器，网上随便一搜索也能查询到几十个编辑器，各有特色，可自行选择使用。

在 Windows 平台下还有一个国人制作的发行版 $\text{C}_{\text{T}}\text{E}_{\text{X}}$ ，但是已经几年没有更新，以后估计也不会更新了，不建议使用（虽然这是带我真正入门的一个发行版），这里要和我们排版中文时用到的 `ctex` 宏包区分开来，虽然它们的名字是相同的。

^④如果要安装软件学习和使用 $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$ ，建议安装 TeX Live 发行版和 TeXstudio 编辑器，二者都是跨平台、免费开源的。

第二章 几点说明

学不博无以通其变，
思不精无以烛其微。
--（清）林佩琴

目标与要求

1. 了解行内公式和行间公式的区别.
2. 理解 \LaTeX 命令的书写规则.
3. 掌握换行的方法及字号的设置命令.
4. 掌握数学公式中输出空白间距的方法.

通常情况下 \LaTeX 有两种模式：文本模式和数学模式，而 GeoGebra 中的 \LaTeX 默认是数学模式，因此针对 GeoGebra 的特色，做如下一些说明。

2.1 \LaTeX 命令

2.1.1 命令区分大小写

\LaTeX 命令是大小写敏感的，在书写时请注意严格区分大小写，这样规定的目的有利于源文件在不同平台之间的移植。

2.1.2 命令的写法

命令以反斜线 (backslash) \backslash 开始，命令名只能由字母组成，命令名后的空格符、数字或任何非字母的字符都标志着该命令结束。



① 还有一种情况是，反斜线后面跟着一个特殊符号，例如换行命令、公式中的空白间距命令、或者是下面将要讲到的一些特殊符号等。

例如：`\sqrt[]{}{ }` 是一条命令，其中大括号 `{ }` 标识了命令的作用范围，表示这是一个整体，中括号 `[]` 标识的部分是可选参数，`\sqrt[3]{x + 5}` 可以得到 $\sqrt[3]{x+5}$ 这一根式。

2.1.3 环境

在 \LaTeX 中，由开始标志 `\begin{环境名}` 到结束标志 `\end{环境名}` 组成的一个代码块称为一个环境，环境名首尾相同，例如：写分段函数的 `cases` 环境、功能强大的 `array` 环境、表格环境 `tabular` 等，我们将在后面具体介绍。

很多复杂的内容都是用环境来完成的，环境可以嵌套。

2.2 几个常用的命令

2.2.1 换行命令

\LaTeX 不会根据回车符 (Enter) 进行换行，如果要想让文字或公式换行则需用双反斜线命令：`\\` 来控制^②。

2.2.2 横线和竖线

在一些数学公式或者是表格中，有时我们需要画横线和竖线，此时可以使用 `\hline` 命令和 `|` 管道符^③ 来完成，注意，这两个命令要在数学环境或者表格中才有作用。

2.2.3 给公式加个框

如果要给公式加一个边框，可以使用 `\boxed` 或 `\fbox` 命令，在 \LaTeX 中两个命令的使用环境是有区别的，在 `GeoGebra` 中使用方法完全相同。



① 此图标的命令是 `\textdbend`，`GeoGebra` 中支持这个命令。

② 可能有的人会问为什么不用 C 语言的 `\n` 呢，也许是因为 \TeX 的编程语言是 `Pascal` 吧。

③ 管道符就是键盘反斜线 `\` 上面的那个符号 `|`。

命令: `\boxed{E = mc^2}` `\fbox{P_n^m = \frac{n!}{(n-m)!}}`

效果:

$$E = mc^2$$

$$P_n^m = \frac{n!}{(n-m)!}$$

2.2.4 撇号的输入

公式中字母右上角的撇号用键盘上分号键后面的 ' 这个撇号输入, 两撇则输入两次撇号而不是双引号, 以此类推。

命令: `f'` `f''` `f'''`

效果: f' f'' f'''



如果要输入西文状态的单引号, 则应该用 ‘ (键盘上数字 1 前面的那个符号) 输入。

2.2.5 带圈字符的输入

一般情况下, 小标题的序号只希望占用一个字符的位置, 可以用下面的命令输入, 字母也可以换成数字。

命令: `\textcircled{a}` `\textcircled{c}` `\textcircled{e}`

效果: ① ③ ⑤

2.3 特殊符号的输入

文档中可以输入的文字符号大致可以分为: 普通字符、控制符、特殊符号、预定义字符串、注音符号等。

普通字符可以直接输入, 而有些字符 (例如 `$ ^ & _ { } ~` 等) 被用作特殊的控制符, 输入这些特殊符号时需要在前面加个 `\` 转义符, 而 `\` 本身则要用 `\backslash` 命令来输入, 因为 `\\` 被用作换行指令。

`\$` `\^` `\&` `_` `\{` `\}` `\~` `\backslash`


2.4 空白间距

用 \LaTeX 编辑公式时，插入在公式中的空格在编译时会被自动忽略，如果要在输出的结果中体现出空格的位置，可以使用 表 2.1 中的命令在公式中生成合适的空白间距，注意负间距命令 $\backslash!$ 可以用来减小间距。

表 2.1: 空白间距

命令	间距	命令	间距
$\backslash,$	$3/18\text{ em}$	\backslashquad	1 em
$\backslash:$	$4/18\text{ em}$	\backslashqqquad	2 em
$\backslash;$	$5/18\text{ em}$	$\backslash!$	$-3/18\text{ em}$

在 GeoGebra 文本对话框的 \LaTeX 数学式下拉列表框中提供的空格命令是 $\backslash;$ 这个，我们可以输入其它命令来产生不同的间距，这些命令可以连续使用而得到更大或更小的间隔。

 em 是一个相对单位，等于当前字体中的大写字母 M 的宽度，中文环境下相当于一个汉字的宽度。

2.5 字号的修改

GeoGebra 软件本身有多处地方可以改变文字的大小（即字号），但它们的作用范围都是整个对象，如果在使用 \LaTeX 编辑公式时，需要改变某一局部文字的大小，则可以使用 表 2.2 中的命令来改变文字的字号。

值得注意的是，表 2.2 中给出的字号是相对于基本字号定义为 10 pt 时的大小，如果基本字号的大小改变了，表中所给出的字号大小也会相应改变，这里只是给出一个参考值。大家可以从 GeoGebra 的选项菜单中修改字号来观察这些命令的显示效果。

在 GeoGebra 的选项菜单中设置的字号和用 $\backslash\text{normalsize}$ 这一命令设置的字号尺寸是相同的，即基本字号的大小。

 请用大括号把命令及需要改变字号的部分括起来，否则在 GeoGebra 中会把范围作用于命令之后的整行。命令： $\text{ABC}\{\backslash\text{huge ABC}\}\text{ABC}$ 得到 $\text{ABC}\text{ABC}\text{ABC}$ 。

表 2.2: 相对字号命令

命令	字号	命令	字号
<code>\tiny</code>	5 pt	<code>\scriptsize</code>	7 pt
<code>\footnotesize</code>	8 pt	<code>\small</code>	9 pt
<code>\normalsize</code>	10 pt	<code>\large</code>	12 pt
<code>\Large</code>	14.4 pt	<code>\LARGE</code>	17.28 pt
<code>\huge</code>	20.74 pt	<code>\Huge</code>	24.88 pt

相对字号的修改

`{\tiny Aa}` `{\scriptsize Aa}` `\dots` `{\LARGE Aa}` `{\huge Aa}` `{\Huge Aa}`

`Aa Aa Aa Aa Aa Aa Aa Aa Aa Aa`



中文书籍基本字号为 5 号字，约等于 11 pt。

2.6 行内公式和行间公式

\LaTeX 的数学模式有两种：行内模式 (inline) 和行间模式 (display)^④，行内模式输入的公式会被压缩，让其和同一行中的文字上下对齐，行间模式输入的公式不会进行压缩处理。

2.6.1 行内公式

行内公式的意思就是文字和公式处于一行中，在 GeoGebra 中，默认情况下输入的公式是行间公式，如果要使用行内公式，需要在公式两边加上定界符：`\(` 和 `\)`。

行内公式

行内公式：`\(\sum_{i=1}^n i = \frac{n(n+1)}{2}\)`。

行内公式： $\sum_{i=1}^n i = \frac{n(n+1)}{2}$ 。

2.6.2 行间公式

行间公式的意思就是在文字和公式混合排版时，公式独立占用一行，此时的公式比较漂亮，这也是 GeoGebra 的默认显示方式，GeoGebra 中省略了行间公式的定界符，直接输入公

^④有的书籍也会翻译为行内模式和特显模式、独立模式等，我个人喜欢称之为行内公式和行间公式，这样从字面上就很容易理解公式出现的位置。

式即可。

行间公式

行间公式: $\sum_{i=1}^n i = \frac{n(n+1)}{2}$.

行间公式:

$$\sum_{i=1}^n i = \frac{n(n+1)}{2}.$$



注意比较行内公式和行间公式上下标的位置及公式的大小。

2.7 如何使用 \LaTeX

在 GeoGebra 的文本对话框中, 不支持用鼠标进行复制粘贴操作, 可以用键盘 $\text{Ctrl}+\text{c}$ 和 $\text{Ctrl}+\text{v}$ 进行操作。

2.7.1 在 GeoGebra 中使用 \LaTeX

GeoGebra 中的 \LaTeX 命令主要是在文本对话框中使用, 具体步骤如下:

首先选择“文本”工具, 如 图 2.1, 接着在绘图区单击鼠标;

然后在弹出的文本对话框中, 勾选“ \LaTeX 数学式”, 如 图 2.2, 接下来就可以在编辑框中输入命令了, 预览框可以实时的看到输出结果。

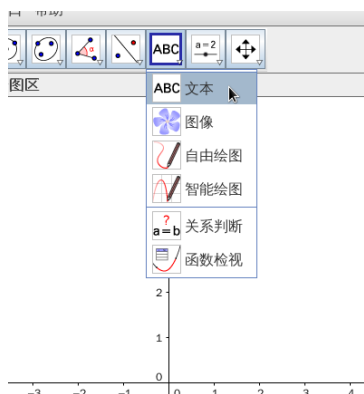


图 2.1: 选择选择文本工具

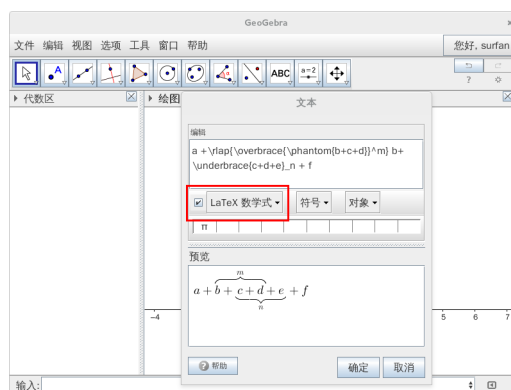


图 2.2: 勾选 \LaTeX 数学式



大家可以试一下 $\backslash\text{GeoGebra}$ 这一命令, 可以得到 $\text{\text{\text{GeoGebra}}}$ 。



在文本输入框中的“ \LaTeX 数学式”和“符号”两个下拉列表中提供了许多常用的符号和命令, 可以用鼠标选择使用, 这大大减少了记忆量。“对象”下拉列表有其它的作用。

2.7.2 “标题”中使用 L^AT_EX

L^AT_EX 除了在 GeoGebra 的文本对话框中使用外，还可以用来修改对象的标签，打开对象的属性对话框，在标题这一栏中输入需要的 L^AT_EX 代码，注意必须在代码前后增加 \$ 符号声明数学环境，然后把“显示标签”后面的内容改为“标题”，如图 2.3。

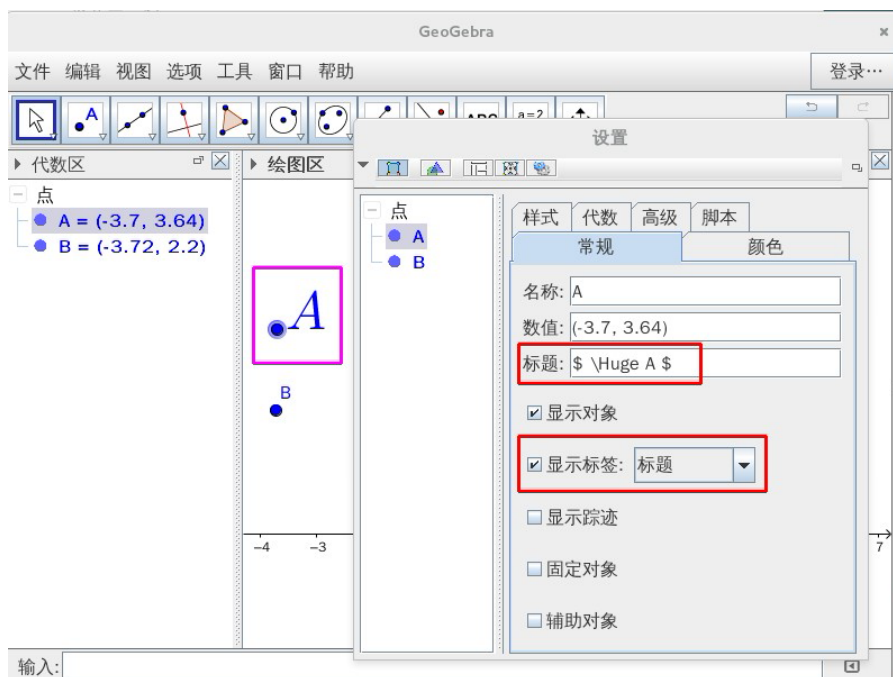


图 2.3: 在标题中使用 L^AT_EX

图 2.3 中点 B 的标签是原始大小，点 A 的标签是改变字号后的效果，在标题栏里支持能在 GeoGebra 中使用的大部分 L^AT_EX 代码。

2.7.3 在其它软件中使用 L^AT_EX 公式

如果想把用 L^AT_EX 生成的这些漂亮的公式用于别的不支持 L^AT_EX 语言的软件中，可以把生成的公式导出成图片^⑤，然后再把图片导入别的软件使用，具体方法如下：

右键选择公式

公式编辑好后，按住鼠标右键拖动选择需要导出的部分，如图 2.4。

^⑤在此感谢贵州的邹颖老师首先指出这一方法和所截图片。

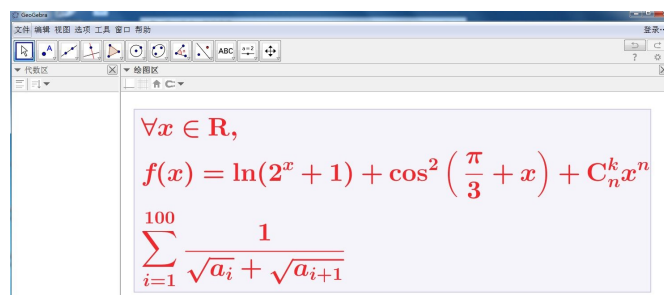


图 2.4: 右键选择公式

导出图片

单击“文件”菜单，选择“导出”--“图片”，导出图片时，选择 png 格式，并勾选“透明”这一选项（默认已经选中），如 图 2.5 这样可以得到占用空间小且背景透明的图片。

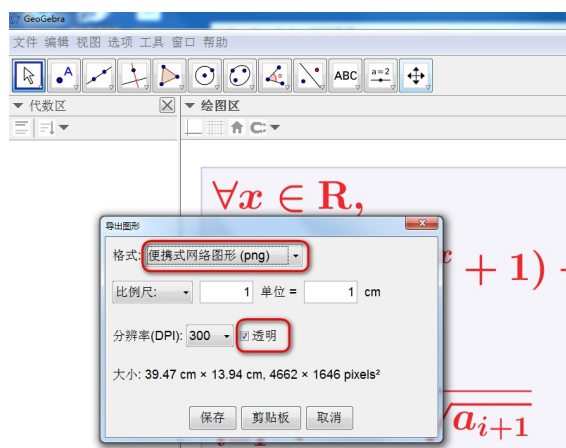


图 2.5: 导出图片

在其它软件中使用

把从 GeoGebra 中导出的图片插入其它软件，从 图 2.6 中可以看出公式背景是透明的。



图 2.6: 在其它软件中使用

第三章 数学公式

书不记，熟读可记；
义不精，细思可精。
惟有志不立，直是无著力处。
--（宋）朱熹

目标与要求

- 1. 掌握公式中基本元素的输入方法.
- 2. 掌握常见公式的输入方法.
- 3. 掌握多行公式的输入方法.
- 4. 熟练使用各种元素输入复杂公式.

3.1 基本元素

3.1.1 希腊字母

在 GeoGebra 的文本输入对话框的“符号”下拉列表中提供了全部的希腊字母和一些常用符号，但为了便于大家查询，所以这里还是把希腊字母的命令及符号用 表 3.1 列举出来。

当命令的首字母大写时，即为输出大写希腊字母，如：

大小写希腊字母举例

命令： \backslash Pi \backslash pi \backslash Phi \backslash phi \backslash Delta \backslash delta \backslash Sigma \backslash sigma

字母： Π π Φ ϕ Δ δ Σ σ



需要注意的是，有一些大写希腊字母的显示效果和大写英文字母完全相同，所以在 L^AT_EX 中没有专门的命令，但在 GeoGebra 中所有大写希腊字母都有对应的命令。

表 3.1: 希腊字母

命令	字母	命令	字母	命令	字母	命令	字母
<code>\alpha</code>	α	<code>\beta</code>	β	<code>\gamma</code>	γ	<code>\delta</code>	δ
<code>\epsilon</code>	ϵ	<code>\varepsilon</code>	ε	<code>\zeta</code>	ζ	<code>\eta</code>	η
<code>\theta</code>	θ	<code>\vartheta</code>	ϑ	<code>\iota</code>	ι	<code>\kappa</code>	κ
<code>\lambda</code>	λ	<code>\mu</code>	μ	<code>\nu</code>	ν	<code>\xi</code>	ξ
<code>\omicron</code>	\omicron	<code>\pi</code>	π	<code>\varpi</code>	ϖ	<code>\rho</code>	ρ
<code>\varrho</code>	ϱ	<code>\sigma</code>	σ	<code>\varsigma</code>	ς	<code>\tau</code>	τ
<code>\upsilon</code>	υ	<code>\phi</code>	ϕ	<code>\varphi</code>	φ	<code>\chi</code>	χ
<code>\psi</code>	ψ	<code>\omega</code>	ω				
<code>\Alpha</code>	\AA	<code>\Beta</code>	\AA	<code>\Gamma</code>	Γ	<code>\Delta</code>	Δ
<code>\Epsilon</code>	\AA	<code>\Zeta</code>	\AA	<code>\Eta</code>	\AA	<code>\Theta</code>	Θ
<code>\Iota</code>	\AA	<code>\Kappa</code>	\AA	<code>\Lambda</code>	Λ	<code>\Mu</code>	\AA
<code>\Nu</code>	\AA	<code>\Xi</code>	Ξ	<code>\Omicron</code>	\AA	<code>\Pi</code>	Π
<code>\Rho</code>	\AA	<code>\Sigma</code>	Σ	<code>\Tau</code>	\AA	<code>\Upsilon</code>	Υ
<code>\Phi</code>	Φ	<code>\Chi</code>	\AA	<code>\Psi</code>	Ψ	<code>\Omega</code>	Ω

3.1.2 上标和下标

指数或上标用 `^` 表示，下标用 `_` 表示，根号用 `\sqrt` 表示。上下标默认只作用于命令之后的一个字符，如果多于一个字符，需要用一对 `{ }` 括起来。

上标和下标

命令: `a_i` `x_{ij}^3` `x^{1/2}` `e^{\text{\AA}^2}\neq\{e^x\}^2` `\text{\AA}_{b}M_{c}^d`

效果: a_i x_{ij}^3 $x^{1/2}$ $e^{x^2} \neq \{e^x\}^2$ $\text{\AA}_{b}M_{c}^d$

3.1.3 虚位

当使用上标 `^` 和下标 `_` 时， \AA TeX 对文本的垂直对齐有时显得太过于自作多情，而有的时候我们需要让上下标占据一个虚位 (phantom)，使用 `\phantom` 命令可以给不在输出结果中显示的字符保留位置，具体的看下面例子。

虚位

命令: `\text{\AA}_{12}^6S` `\text{\AA}_{12}^{6}S` `\Gamma_{ij}^k`

`\Gamma_{ij}^{k}` `\Gamma_{ij}^{k}`

效果: \AA_{12}^6S \AA_{12}^6S Γ_{ij}^k Γ_{ij}^k Γ_{ij}^k

二项式结构

命令: $\mathrm{C}_n^k = \frac{n!}{k!(n-k)!}$

$\{n \text{ choose } k\} = \frac{n!}{k!(n-k)!}$

$\mathrm{binom}\{n\}\{k\} = \frac{n!}{k!(n-k)!}$

$\{n \text{ atopwithdelims}() k\} = \{n! \text{ above } 1\text{pt } k!(n-k)!\}$

效果: $C_n^k = \frac{n!}{k!(n-k)!} \quad \binom{n}{k} = \frac{n!}{k!(n-k)!} \quad \binom{n}{k} = \frac{n!}{k!(n-k)!} \quad \binom{n}{k} = \frac{n!}{k!(n-k)!}$



利用 $\text{atopwithdelims}()$ 还可以得到一些复杂的结构, above 命令是得到分数线, 1pt (数字 1) 是线条的粗细, 大家可以修改这一数值看看效果。

3.1.6 运算符

像 $+ - = < >$ 这些键盘上有的运算符可以直接输入, 更多的则需要使用命令来输入, 这里列举了一些常用符号, 在 GeoGebra 的文本输入对话框的“符号”下拉列表中提供了很多字母和符号, 可以直接用鼠标点击使用。

常用二元运算符

表 3.2 列出了常用的二元运算符。

表 3.2: 常用二元运算符

命令	符号	命令	符号	命令	符号	命令	符号
$\backslash times$	\times	$\backslash div$	\div	$\backslash pm$	\pm	$\backslash mp$	\mp
$\backslash cdot$	\cdot	$\backslash star$	\star	$\backslash bullet$	\bullet	$\backslash circ$	\circ
$\backslash oplus$	\oplus	$\backslash ominus$	\ominus	$\backslash otimes$	\otimes	$\backslash oslash$	\oslash
$\backslash bigoplus$	\bigoplus	$\backslash bigotimes$	\bigotimes	$\backslash odot$	\odot	$\backslash bigodot$	\bigodot
$\backslash vee$	\vee	$\backslash wedge$	\wedge	$\backslash bigvee$	\bigvee	$\backslash bigwedge$	\bigwedge
$\backslash cap$	\cap	$\backslash cup$	\cup	$\backslash bigcap$	\bigcap	$\backslash bigcup$	\bigcup
$\backslash Cap$	\cap	$\backslash Cup$	\cup	$\backslash uplus$	\uplus	$\backslash biguplus$	\biguplus

常用关系符

表 3.3 列出了常用的二元关系符, 更多符号请查阅相关文档。



很多命令可以在其前面加上 $\backslash not$ 来得到它们的否定形式, 如: $\backslash not \backslash equiv$ 得到 \neq 。

表 3.3: 常用二元关系符

命令	符号	命令	符号	命令	符号	命令	符号
<code>\leq</code>	\leq	<code>\geq</code>	\geq	<code>\nless</code>	\nless	<code>\ngtr</code>	\ngtr
<code>\leqslant</code>	\leqslant	<code>\geqslant</code>	\geqslant	<code>\neq</code>	\neq	<code>\equiv</code>	\equiv
<code>\ll</code>	\ll	<code>\gg</code>	\gg	<code>\lll</code>	\lll	<code>\ggg</code>	\ggg
<code>\approx</code>	\approx	<code>\cong</code>	\cong	<code>\mid</code>	\mid	<code>\nmid</code>	\nmid
<code>\in</code>	\in	<code>\notin</code>	\notin	<code>\ni</code>	\ni	<code>\not\ni</code>	$\not\ni$
<code>\subset</code>	\subset	<code>\supset</code>	\supset	<code>\subseteq</code>	\subseteq	<code>\supseteq</code>	\supseteq
<code>\subsetneq</code>	\subsetneq	<code>\supsetneq</code>	\supsetneq	<code>\subsetneqq</code>	\subsetneqq	<code>\supsetneqq</code>	\supsetneqq

大运算符

累加、累乘、极限、积分等大型运算符用 `\sum` `\prod` `\lim` `\int` `\oint` 等命令^②表示，如果要想定积分的上下标位于积分号的上下方，可以用 `\limits` 命令来标识上下标。

在 \LaTeX 中推荐用命令 `\prod` 来表示累乘，而不是用 π 的大写 `\Pi`，在 GeoGebra 中两个符号的大小有明显的差别。

大运算符

命令: `\sum_{i=1}^n i` `\prod_{i=1}^n` `\lim_{x \rightarrow 0} x^2`
`\int_a^b x^2 dx` `\int\limits_a^b x^2 dx` `\oint`

效果: $\sum_{i=1}^n i$ $\prod_{i=1}^n$ $\lim_{x \rightarrow 0} x^2$ $\int_a^b x^2 dx$ $\int_a^b x^2 dx$ \oint

追求完美的老师可能会觉得积分公式末尾的积分变量 dx 改成 $\mathrm{d}x$ 比较好看，另外积分函数和积分变量之间需要拉开一点距离，那么我们可以在公式中增加空白间距和改变数学字体来达到这一目的。

修改积分变量

命令: `\int_a^b x^2 dx` `\int_a^b x^2 \mathrm{d}x`

效果: $\int_a^b x^2 dx$ $\int_a^b x^2 \mathrm{d}x$

多重积分如果用多个 `\int` 来输入的话，积分号之间的距离会显得过宽，正确的方法是用 `\iint`、`\iiint`、`\idotsint` 等命令来输入，可以从下表中比较一下两种方法的差异。

^②在 GeoGebra 中，一直没有找到 \oint 这一符号是用哪个命令，望找到的老师告知。

多重积分

命令: `\int` `\int\int` `\int\int\int` `\int\int\int\int` `\int\cdots\int`
`\int` `\iint` `\iiint` `\iiiiint` `\idotsint`

效果: \int \iint \iiint \iiiiint $\int \cdots \int$

3.1.7 箭头

表 3.4 给出了一些箭头的输入方法, `\xleftarrow` 和 `\xrightarrow` 命令生成的箭头可以根据内容自动调整长度。

表 3.4: 箭头

命令	箭头	命令	箭头
<code>\leftarrow</code>	\leftarrow	<code>\rightarrow</code>	\rightarrow
<code>\longleftarrow</code>	\longleftarrow	<code>\longrightarrow</code>	\longrightarrow
<code>\leftrightarrow</code>	\leftrightarrow	<code>\longleftrightarrow</code>	\longleftrightarrow
<code>\leftrightsquigarrow</code>	\leftrightsquigarrow	<code>\rightleftarrows</code>	\rightleftarrows
<code>\Leftarrow</code>	\Leftarrow	<code>\Rightarrow</code>	\Rightarrow
<code>\nLeftarrow</code>	\nLeftarrow	<code>\nRightarrow</code>	\nRightarrow
<code>\Leftrightarrow</code>	\Leftrightarrow	<code>\Longleftarrow</code>	\Longleftarrow
<code>\Leftrightarrow</code>	\Leftrightarrow	<code>\Longrightarrow</code>	\Longrightarrow
<code>\Downarrow</code>	\Downarrow	<code>\Uparrow</code>	\Uparrow
<code>\nwarrow</code>	\nwarrow	<code>\nearrow</code>	\nearrow
<code>\swarrow</code>	\swarrow	<code>\searrow</code>	\searrow

命令: `\xleftarrow{x + y + z}` `\xrightarrow[x < y]{a * b * c}`

效果: $\xleftarrow{x+y+z}$ $\xrightarrow[x<y]{a*b*c}$

3.1.8 分隔符

括号用 `()` `[]` `\{ \}` `\langle \rangle` `\lVert \rVert` `\lfloor \rfloor` `\lceil \rceil` 等命令来表示, 在 GeoGebra 中的绝对值符号用键盘上的管道符 `|` 输入, 我们可以在这些分隔符前面加

`\big`、`\Big`、`\bigg`、`\Bigg` 这几个命令来调整它们的大小。

LaTeX 中原有的方法是在分隔符前面加 `\left` 和 `\right` 来自动调整大小，GeoGebra 中的“LaTeX 数学式”下拉列表中的“括号”就是采用的这种方法，但有的时候效果不佳，可以根据实际情况来选择不同的方法以便达到最佳的显示效果。

分隔符

命令：`\Bigg(\bigg(\Big(\big((x)\big)\Big)\bigg)\Bigg)`

`\Bigg[\bigg[\Big[\big[[x]\big]\Big]\bigg]\Bigg]`

`\Bigg\{\bigg\{\Big\{\big\{\{x\}\big\}\Big\}\bigg\}\Bigg\}`

效果： $\left(\left(\left(\left(x\right)\right)\right)\right)$ $\left[\left[\left[\left[x\right]\right]\right]\right]$ $\left\{\left\{\left\{\left\{\left\{x\right\}\right\}\right\}\right\}\right\}$

`\Bigg\langle\Big\langle\Big\langle x \rangle\Big\rangle\Bigg\rangle`

`\Bigg|\bigg|\Big|\big|| x |\big|\Big|\bigg|\Bigg|`

`\Bigg\lvert\Big\lvert\Big\lvert x \rvert\Big\rvert\Bigg\rvert`

`\Bigg\lfloor\Big\lfloor\Big\lfloor x \rfloor\Big\rfloor\Bigg\rfloor`

`\Bigg\lceil\Big\lceil\Big\lceil x \rceil\Big\rceil\Bigg\rceil`

效果： $\left\langle\left\langle\left\langle x \right\rangle\right\rangle\right\rangle$ $\left|\left|\left|x\right|\right|\right|$ $\left|\left|\left|x\right|\right|\right|$ $\left\lfloor\left\lfloor\left\lfloor x \right\rfloor\right\rfloor\right\rfloor$ $\left\lceil\left\lceil\left\lceil x \right\rceil\right\rceil\right\rceil$



为了让代码显示得简短一些，并没有把所有分隔符的情况都写出来，一般情况并不要求分隔符必须成对出现，甚至还可以不同大小的不同分隔符配合使用，请大家自行验证。

3.1.9 省略号

在数学环境中，有不同位置、不同方向的省略号，分别用不同的命令来实现，具体看下面的例子。

省略号

命令：`\dots` `\cdots` `\ldots` `\dotsb` `\vdots` `\ddots` `\iddots`

效果： \dots \cdots \ldots \dotsb \vdots \ddots \iddots



在 LaTeX 中用 `\dots` 命令得到的省略号会根据前后文的情况自动调整水平对齐的位置，

但在 GeoGebra 中 `\dots` 和 `\ldots` 命令的效果似乎没有区别。

3.2 多行公式

有的公式特别长一行放不下，需要手动为它们换行，或几个公式需要写成一组，还有些类似分段函数，需要给它加上一个左边的花括号，这时我们就要用到一些多行公式环境。

3.2.1 长公式

不须要对齐的长公式可以使用 `multline` 环境，需要对齐的长公式可以使用 `aligned` 环境，用 `\\` 来分行，并用 `&` 来标识对齐的位置。

请大家验证 表 3.5 中的两组代码，并比较显示效果的差异。

表 3.5: 长公式环境

不对齐	对齐
<code>\begin{multline}</code> $x = a+b+c+ \\ d+e+f+g$ <code>\end{multline}</code>	<code>\begin{aligned}</code> $x = &a+b+c+ \\ &d+e+f+g$ <code>\end{aligned}</code>

3.2.2 公式组

不需要对齐的公式组可以使用 `gather` 环境，需要对齐的公式组可以使用 `align` 环境，并使用 `&` 来标识对齐的位置，比较一下 表 3.6 中两组公式的显示效果。

表 3.6: 公式组环境

不对齐	对齐
<code>\begin{gather}</code> $x = a+b \\ \text{sum} = i+j+k+l$ <code>\end{gather}</code>	<code>\begin{align}</code> $x &= a+b \\ \text{sum} &= i+j+k+l$ <code>\end{align}</code>

不对齐（使用 `gather` 环境）

$$x = a + b \quad (3.1)$$

$$sum = i + j + k + l \quad (3.2)$$

对齐（使用 `align` 环境）

$$x = a + b \quad (3.3)$$

$$sum = i + j + k + l \quad (3.4)$$

3.2.3 分段函数

分段函数通常用 `cases` 环境来写分支公式，并使用 `&` 来标识对齐的位置。

表 3.7: 分段函数

<code>y = \begin{cases}</code>	<code>D(x)=\begin{cases}</code>
<code>-x, &x\leqslant \backslash</code>	<code>1, &\text{ if } x \in \mathbb{Q} \backslash</code>
<code>x, &x>0</code>	<code>0, &\text{ if } x \in \mathbb{R} \setminus \mathbb{Q}</code>
<code>\end{cases}</code>	<code>\end{cases}</code>

$$y = \begin{cases} -x, & x \leq 0 \\ x, & x > 0 \end{cases} \quad D(x) = \begin{cases} 1, & \text{if } x \in \mathbb{Q} \\ 0, & \text{if } x \in \mathbb{R} \setminus \mathbb{Q} \end{cases}$$



说明一下 表 3.7 中第二个例子的几个命令，`\text{ if }` 的作用是把 `if` 改为文本以正体显示，并且保留 `if` 前后的空格，`\mathbb` 命令的作用是把数学字体^③改为黑板体，命令 `\setminus` 的显示效果和 `\backslash` 的相同，但多用于公式中。

^③默认情况 \LaTeX 中的数学字体显示为斜体，关于数学字体在后面会专门介绍。

3.3 矩阵

行列式和矩阵的生成方法是相同的，这里统一用矩阵来说明，生成矩阵的环境有很多个，分别应用于不同分隔符的情况，这些环境中大部分都不支持自定义，支持自定义的是 `array` 这一环境，也是功能最强大的一个。



这些环境有一个共同的特点：都是用 `&` 来分隔一行中的元素，用 `\\` 来换行。

3.3.1 无分隔符的矩阵

可以用 `matrix` 环境生成没有分隔符也没有对齐参数（默认居中）的矩阵。

无分隔符的矩阵

```
\begin{matrix}
x_1 & x_2 & \dots \\
x_3 & x_4 & \dots \\
\vdots & \vdots & \ddots
\end{matrix}
```

```
x_1  x_2  ...
x_3  x_4  ...
⋮    ⋮    ⋱
```

3.3.2 带分隔符的矩阵

`pmatrix`、`bmatrix`、`Bmatrix`、`vmatrix`、`Vmatrix` 等环境可以在矩阵两边加上不同的分隔符，它们同样也没有对齐方式的参数。

带分隔符的矩阵

```
\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}
```

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

为了排版时显示得紧凑一些，这里把每一个矩阵写成了一行，大家也可以按照上例的样子写成多行的样式，得到的输出效果是相同的。

3.3.3 较复杂的矩阵示例

在上面所说的众多环境中，基本上不支持自定义功能，如果我们需要更加复杂的矩阵，比如要在矩阵内部画横线和竖线，甚至还有其他的一些要求，此时我们就要用 `array` 这一环境来达到目的。

较复杂的矩阵示例

```
(A,B) = \left(\begin{array}{cccc|c}
```

```
a & b & c & d & e \\
```

```
3 & 4 & 5 & 6 & -1 \\
```

```
k & f & i & -9 & 2 \\ \hline
```

```
s & x & L & R & E
```

```
\end{array}\right)_{m \times n}
```

$$(A,B) = \left(\begin{array}{cccc|c} a & b & c & d & e \\ 3 & 4 & 5 & 6 & -1 \\ k & f & i & -9 & 2 \\ \hline s & x & L & R & E \end{array} \right)_{m \times n}$$

`array` 环境具有非常强大的自定义功能，它提供了列对齐的参数，有三种方式：居左、居中、居右，分别用字母：`l`、`c`、`r` 表示，不同的列用符号 `&` 分隔，行用 `\\` 分隔，它还能在矩阵中根据需要来画横线和竖线^④。

在 `array` 环境生成的矩阵中，可以在对齐参数的中使用管道符 `|` 来生成竖线，在需要生成横线的地方使用 `\hline` 命令。

用 `array` 环境生成分段函数

之前我们是用 `cases` 环境来生成分段函数的，`array` 环境同样也能用来生成分段函数，使用起来更加灵活。

用 `array` 环境生成分段函数

```
\chi_A(x) = \left\{
  \begin{array}{ll}
    1, & \& x \in A \\
    0, & \& x \notin A
  \end{array}
\right.
```

$$\chi_A(x) = \begin{cases} 1, & x \in A \\ 0, & x \notin A \end{cases}$$



在使用 `array` 环境时，需要特别注意的是，环境左右两边可以不加分隔符（括号），加分隔符时左右两边的符号可以不相同，但是 `\left` 和 `\right` 这两个命令必须成对出现，如果不想把分隔符显示出来，那么可以用英文句号 `.` 来代替分隔符，请仔细观察上例中的右括号。

`array` 环境的嵌套

`array` 环境支持嵌套，请看下面的例子，注意 `@{}` 命令的作用是压缩列之间的空白间距，这个例子代码相对长了一点，但用到的知识点并不多，大家多试几次也就能熟练掌握了。

^④在 GeoGebra 中对齐方式的参数可以省略，默认为居中，但引用对齐参数的大括号 `{}` 不能省略。

array 环境的嵌套

```

\left(\begin{array}{c@{}c@{}c}

\begin{array}{|cc|}\hline

a_{11} & a_{12} \\

a_{21} & a_{22} \\\hline

\end{array} & \mathbf{0} & \mathbf{0} \\

\mathbf{0} & & \\

\begin{array}{|ccc|}\hline

b_{11} & b_{12} & b_{13} \\

b_{21} & b_{22} & b_{23} \\

b_{31} & b_{32} & b_{33} \\\hline

\end{array} & \mathbf{0} & \\

\mathbf{0} & \mathbf{0} & \\

\begin{array}{|cc|}\hline

c_{11} & c_{12} \\

c_{21} & c_{22} \\\hline

\end{array}

\end{array}\right)

```

$$\left(\begin{array}{ccc|ccc|cc} \boxed{a_{11} & a_{12}} & & & \mathbf{0} & & \mathbf{0} \\ \boxed{a_{21} & a_{22}} & & & & & \\ & & \boxed{b_{11} & b_{12} & b_{13}} & & \\ & \mathbf{0} & \boxed{b_{21} & b_{22} & b_{23}} & \mathbf{0} & \\ & & \boxed{b_{31} & b_{32} & b_{33}} & & \\ & \mathbf{0} & & \mathbf{0} & & \boxed{c_{11} & c_{12}} \\ & & & & & \boxed{c_{21} & c_{22}} \end{array} \right)$$

3.4 注音和标注

表 3.8 列出了一些数学注音符号，表 3.9 列出了一些长的标注符号。

表 3.8: 数学注音符号

命令	符号	命令	符号	命令	符号
<code>\bar{x}</code>	\bar{x}	<code>\acute{x}</code>	\acute{x}	<code>\mathring{x}</code>	\mathring{x}
<code>\vec{x}</code>	\vec{x}	<code>\grave{x}</code>	\grave{x}	<code>\dot{x}</code>	\dot{x}
<code>\hat{x}</code>	\hat{x}	<code>\tilde{x}</code>	\tilde{x}	<code>\ddot{x}</code>	\ddot{x}
<code>\check{x}</code>	\check{x}	<code>\breve{x}</code>	\breve{x}	<code>\ddd{x}</code>	$\overline{\overline{\overline{x}}}$

表 3.9: 长标注符号

命令	符号	命令	符号
<code>\overline{xxx}</code>	\overline{xxx}	<code>\overleftrightarrow{xxx}</code>	\overleftrightarrow{xxx}
<code>\underline{xxx}</code>	\underline{xxx}	<code>\underleftrightarrow{xxx}</code>	$\underleftrightarrow{xxx}$
<code>\overleftarrow{xxx}</code>	\overleftarrow{xxx}	<code>\overbrace{xxx}</code>	\overbrace{xxx}
<code>\underleftarrow{xxx}</code>	\underleftarrow{xxx}	<code>\underbrace{xxx}</code>	\underbrace{xxx}
<code>\overrightarrow{xxx}</code>	\overrightarrow{xxx}	<code>\widehat{xxx}</code>	\widehat{xxx}
<code>\underrightarrow{xxx}</code>	\underrightarrow{xxx}	<code>\widetilde{xxx}</code>	\widetilde{xxx}

向量的输入

向量可以通过在一个变量上方添加小箭头来指定，用 `\vec` 命令来输入即可，如果要表示一个从 A 到 B 的向量，则需要用到 `\overleftarrow` 和 `\overrightarrow` 这两个命令。

向量

命令：`\vec{a}` `\overleftarrow{BA}` `\overrightarrow{AB}`

效果： \vec{a} \overleftarrow{BA} \overrightarrow{AB}

带上下括号的公式

通常我们会遇到一些公式，需要用大括号对其进行注释说明，这类公式虽然遇到的不多，但是也非常实用，我们看下面的例子：

带上下括号的公式

命令: $a + \{\overbrace{b + \cdots}^{126}\} + z$

$\overset{m}{\overbrace{a + \underset{n}{\underbrace{(b + c)}}}}$

$\underbrace{a + \overbrace{b + \cdots + y}^{=t}} + z _ \{\text{total}\}$

$a + \rlap{\overbrace{}^m} b + \underbrace{c+d+e}_n + f$

效果: $a + \overbrace{b + \cdots}^{126} + z$ $a + \underbrace{(b + c)}_n$ $\underbrace{a + b + \cdots + y + z}_{\text{total}}$ $a + \overbrace{b + c + d + e}^m + f$

3.5 常用符号

除了前面讲到的那些符号外, 还有一些符号也是比较常用的, 表 3.10 列出了一部分。

表 3.10: 常用符号

命令	符号	命令	符号	命令	符号
<code>\because</code>	\because	<code>\therefore</code>	\therefore	<code>\perp</code>	\perp
<code>\infty</code>	∞	<code>\propto</code>	\propto	<code>\varnothing</code>	\varnothing
<code>\forall</code>	\forall	<code>\exists</code>	\exists	<code>\nabla</code>	∇
<code>\land</code>	\wedge	<code>\lor</code>	\vee	<code>\lnot</code>	\neg
<code>\odot</code>	\odot	<code>\triangle</code>	\triangle	<code>\square</code>	\square
<code>\angle</code>	\angle	<code>\measuredangle</code>	\measuredangle	<code>\sphericalangle</code>	\sphericalangle
<code>\parallel</code>	\parallel	<code>\nparallel</code>	\nparallel	<code>\slash</code>	\parallel
<code>\spadesuit</code>	\spadesuit	<code>\heartsuit</code>	\heartsuit	<code>\clubsuit</code>	\clubsuit
<code>\diamondsuit</code>	\diamondsuit	<code>\circledS</code>	\circledS	<code>\copyright</code>	\copyright
<code>\S</code>	\S	<code>\bigstar</code>	\bigstar	<code>\textperthousand</code>	\textperthousand
<code>\surd</code>	\surd	<code>\degree</code>	$^\circ$	<code>\celsius</code>	$^\circ\text{C}$



看了各种符号的输入, 大家可能已经发现, 要实现一种效果可能有不止一种方法, 比如要得到度这个符号, `180\degree` 和 `180^{\circ}` 这两个命令都会得到 180° 这一结果。

3.6 数学字体

我们在编辑数学公式时, 可能会根据需要选用不同的字体样式, 表 3.11 中给出了一些改变数学字体的命令及显示效果。需要注意的是, 有的命令只对大写字母有效。

表 3.11: 数学字体

命令	效果	命令	效果
缺省	<i>abcABC</i>	<code>\mathbf</code>	abcABC
<code>\mathrm</code>	abcABC	<code>\mathit</code>	<i>abcABC</i>
<code>\mathsf</code>	abcABC	<code>\mathbb</code>	ABCXYZ
<code>\mathtt</code>	abcABC	<code>\mathfrak</code>	abcABC
<code>\mathcal</code>	<i>ABCXYZ</i>	<code>\mathscr</code>	<i>ABCXYZ</i>

3.7 标准数学函数

数学函数一般用直立的 Roman 体排印，而普通字母一般用 Italic 字体，所以在输入函数的时候需要在函数名的前面加上 `\` 转义符，请参照 表 3.12。

表 3.12: 标准数学函数

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

3.8 简单表格

在 GeoGebra 中，可以用 L^AT_EX 来绘制一些简单的表格，`tabular` 环境提供了最简单的表格功能，它用 `\hline` 命令来表示横线，`|` 表示竖线，用 `&` 来分列，用 `\\` 来换行；每列可以采用居左、居中、居右等对齐方式，分别用字母 `l`、`c`、`r` 来表示。

简单表格

```
\begin{tabular}{|l|c|r|}\hline
```

```
居左 & 居中 & 居右 \\\hline
```

```
GGb & GGb & GGb \\\hline
```

```
GeoGebra & GeoGebra & GeoGebra \\\hline
```

```
TeX & TeX & TeX \\\hline
```

```
LaTeX & LaTeX & LaTeX \\\hline
```

```
\end{tabular}
```

居左	居中	居右
GGb	GGb	GGb
GeoGebra	GeoGebra	GeoGebra
TeX	TeX	TeX
LaTeX	LaTeX	LaTeX

3.9 一些示例

这里收集了一些常见公式样式的输入方法，以供参考。

3.9.1 上下标综合应用

$$e^{i \pi} + 1 = 0 \quad z = r \cdot e^{2 \pi i}$$

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

$$e^{i\pi} + 1 = 0 \quad z = r \cdot e^{2\pi i} \quad \lim_{n \rightarrow \infty} \sum_{k=1}^n \frac{1}{k^2} = \frac{\pi^2}{6}$$

3.9.2 利用虚位来对齐等式

利用虚位 `\phantom` 命令可以让递等式对齐，例子中第一行相当于虚拟了一个不显示出来的等号，然后用 `&` 来标识对齐的位置。

```

\begin{align}
& \phantom{=}(a+b)(a^2-ab+b^2) \\
& = a^3-a^2b+ab^2 + a^2b-ab^2+b^3 \\
& = a^3+b^3
\end{align}

```

$$\begin{aligned}
 & (a+b)(a^2-ab+b^2) \\
 & = a^3 - a^2b + ab^2 + a^2b - ab^2 + b^3 \\
 & = a^3 + b^3
 \end{aligned}$$

3.9.3 让人眼花的结构

写这个结构的目的是让大家知道复杂结构的公式并不难输入。

```

\sum_{i=1}^{\left[\frac{n}{2}\right]}
\binom{x_{i,i+1}^{i^2}}
{\left[\frac{i+3}{3}\right]}
\frac{\sqrt{\mu(i)^{\frac{3}{2}}(i^2-1)}}
{\frac{\sqrt{\mu(i)^{\frac{3}{2}}(i^2-1)}}{\left[\frac{i+3}{3}\right]}}
\sqrt[3]{\rho(i)-2} + \sqrt[3]{\rho(i)-1}

```

$$\sum_{i=1}^{\left[\frac{n}{2}\right]} \binom{x_{i,i+1}^{i^2}}{\left[\frac{i+3}{3}\right]} \frac{\sqrt{\mu(i)^{\frac{3}{2}}(i^2-1)}}{\sqrt[3]{\rho(i)-2} + \sqrt[3]{\rho(i)-1}}$$

3.9.4 和集合有关的公式

利用虚位命令可以把集合中的分割符变长一些，显得更美观，注意第二个公式的代码过长，在书写的时候分成了两行，请大家注意。

$$\left\{x \in \mathbf{R} \mid 0 < |x| < \frac{5}{3}\right\}$$

$$\left\{\phantom{\frac{5}{3}}x \in \mathbf{R} \mid 0 < |x| < \frac{5}{3}\right\}$$

$$\left. 0 < |x| < \frac{5}{3} \right\}$$

$$\left\{x \in \mathbf{R} \mid 0 < |x| < \frac{5}{3}\right\}$$

$$x \mapsto \{c \in C \mid c \leq x\}$$

$$A = \{x \in X \mid x \in X_i \text{ for some } i \in I\}$$

$$x \mapsto \{c \in C \mid c \leq x\} \quad A = \{x \in X \mid x \in X_i \text{ for some } i \in I\}$$


比较上面几个例子会发现，用管道符 `|` 和用 `\mid` 命令生成的分隔符是有区别的，管道符生成的分隔符和前后字符离得太近，但可以和 `\left` 和 `\right` 配合使用得到比较长的分隔符，与前后字符的间隔可以用空白间距命令来调整。

3.9.5 符号上下方的标注

如果要在数字或符号的上下方进行标注，可以使用 `\overset{...}{...}` `\underset{...}{...}` 命令，其作用是将前一括号中的内容置于后一括号的上方或下方。

`\thicksim` 命令是画波浪符号 \sim 此符号不能用键盘上感叹号前面的那个键输入。

$$u \overset{?}{+} v \underset{1}{\thicksim} w \overset{2}{\thicksim} z$$

$$f(x) \overset{\text{def}}{=} x^2 - 1$$

$$u \overset{?}{+} v \underset{1}{\thicksim} w \overset{2}{\thicksim} z \quad f(x) \overset{\text{def}}{=} x^2 - 1$$

3.9.6 背景色和文字颜色

有时需要给公式加一个背景色或者是修改某一字符的颜色，如果用 GeoGebra 本身的功能则作用范围是整个文本，缺少灵活性，要更加自由的设置可以使用 `\colorbox` `\textcolor` 命令来实现，具体的请看下面的例子。

背景色和前景色

```
\colorbox{yellow}{f(x)=\prod_{i=1}^n\left(i-\frac{1}{2i}\right)}
```

```
\textcolor{blue}{f(x)}=\int_1^{\infty}\textcolor{red}{x}\,\mathrm{d}x
```

$$f(x) = \prod_{i=1}^n \left(i - \frac{1}{2i}\right) \quad f(x) = \int_1^{\infty} x \, dx$$

3.9.7 将下标分行

有时一些大运算符的参数不止一个，为了公式的美观，我们可以把下标分成多行来写书，此方法同样适用于上标。

将下标分行

```
\sum_{\substack{0<i<n \\ 1<j<m}} P(i,j) =
```

```
\sum_{\substack{1 \in I \\ 1<j<m}} Q(i,j)
```

$$\sum_{\substack{0<i<n \\ 1<j<m}} P(i,j) = \sum_{\substack{1 \in I \\ 1<j<m}} Q(i,j)$$

跋

首先向一路披荆斩棘看到这里的读者表示祝贺，如果您认真阅读，动手完成了文中所有的例子，并琢磨了其中的意义，相信您已经可以用 GeoGebra 中的 \LaTeX 语言排版出非常漂亮的公式了，从此您也可以算得上半个 \LaTeX er 了，只要坚持下去，您会在数学领域中展现出更多的精彩。不管是在什么行业，要想成为高手，只需要勤学苦练就行了，希望您能在不断的练习中进行创新，发现更多的功能和技巧，并请分享出来。

很多读者没有坚持一直学习 \LaTeX 的原因之一可能是因为 \LaTeX 需要记忆的内容太多，所幸的是，在 GeoGebra 的文本对话框中，有两个下拉列表提供了很多常用命令和符号，只须用鼠标选择就能使用，这大大的方便了大家的使用。

费尽九牛二虎之力熬到本文档完成的时候，才发现自己需要学习的地方也非常之多，从前的一些想法很傻很天真。让我们搜练古今，博采沉奥，耐心等待更好的文档出现。