东北大学秦皇岛分校数学建模培训

LaTeX初步

数学与统计学院 刘建波

December 31, 2016

Contents

1 引言

1.1 TEX与 LATEX简介

电子排版系统的出现，给印刷出版业带来了一场革命，利用电子计算机及各种辅助设备，可以完成从文稿、 图表的录入、编辑、修改、组版，直至得到各种不同用途、不同质量的输出结果。利用电子排版系统， 可以减轻劳动强度，缩短出版周期。目前世界上有许多电子排版系统。这些系统各有特点，也各有自己的适用范围。 TEX（或 LATEX） 就是一种优秀的电子排版系统。

TEX（或 LATEX） 是一种排版系统，它非常适用于生成高印刷质量的科技（特别是数学类）文档， 这个系统同样适用于生成从简单的信件到完整书籍的所有其他种类的文档。而且这个系统使用方便、价格低廉， 且当排版论文、报告和书籍时，其排版质量毫不逊色于价格昂贵的大型系统。

TEX系统是由美国斯坦福大学的高德纳（Donald E. Knuth）教授研制的计算机排版软件系统。 高德纳教授既是著名的数学家，也是一名计算机专家，是享有盛誉的计算机程序设计系列专著 The Art of Computer Programming 的作者。按照他的计划，这套书总共六册，前三册出版以后，高德纳将修订的第二册的第二版交给出版社排版， 但是对收到的校样很不满意，因为这时出版社已经开始用计算机代替手工排版了，当时的字形和版面都很难看，每次校改都很麻烦。 为了以后出版方便，他放下手头的其它工作，着手开发设计一种高质量的计算机排版软件，花费了他大量的时间和精力以后， 成功研制了这套闻名于世的排版系统—TEX系统。随后，他又撰写了一整套的TEX手册，既讲TEX的使用方法，又讲设计原理。难能可贵的是， 高德纳没有利用TEX系统去发财致富，而是把源代码向用户无私地公开。

利用TEX系统编写手稿，除了正文内容以外，还需加入一些排版命令，与大型排版系统不同的是， 这些排版命令通常不是编辑加入的，而是由作者本人加入完成的。

TEX提供的排版命令功能强大，用户可以直接使用这些命令，也可以发挥创造力，利用已有的功能自行定义新的命令，以适合特定的需要。

TEX系统提供了300多条基本命令，其中大部分是键盘上没有的特殊符号的代码，功能虽然强大，但是使用不方便。 后来在这些基本命令的基础之上，又定义了600多条复合命令，即构成了名为 Plain TEX的软件包（宏包），现在我们提到的TEX就是 Plain TEX。

Plain TEX系统虽然比“原始系统”容易使用了，但是排版复杂的版面或

公式时，仍需要书写大量的命令， 还是不够方便，因此，国外一些研究人员利用TEX的宏定义功能，对系统进行了二次开发，产生了一些TEX系统的衍生版本，其中有美国数学会开发的$\mathcal{AMS}$-TEX和 Lamport 编写的LATEX。 前者的特点是容易排版复杂的数学公式，后者适合排版普通的文章和书籍。把这两种系统联合起来就更符合我们的需要，于是就出现了兼容于$\mathcal{AMS}$-TEX而又包含了LATEX优点的衍生版本，但是没有广泛流行。 但是LATEX由于在新版本（LATEX 2_ε）中可以加载amsmath宏包，基本包含了$\mathcal{AMS}$-TEX的功能，而大为流行，占据了TEX领域的重要地位。

我们这里介绍的主要就是LATEX 2_ε，我们所称的LATEX就是指LATEX 2_ε。

LATEX系统的一些特点：

LATEX系统的特色功能之一就是它的自动编号功能。文章、书籍的章、节、段落，以及公式、图表、文献、页码等均可自动进行编号，这给作者带来很大的方便。 增加、删除、改变一些带有编号的公式、章节等时，系统会自动改变编号；

LATEX系统可以自动生成目录页，可以生成索引附录；

LATEX系统可以建立链接，进行交叉引用；

等等。

后面，我们也用TEX泛指 Plain TEX，$\mathcal{AMS}$-TEX，LATEX，以及后面要介绍的TEX的衍生系统。

TEX系统受到科学工作者的喜爱，因为它能排版出复杂的图表和精美的公式，到目前为止，国内外公认的数学公式排版最好的仍然是这两个系统。 许多国际、国内专业学会的期刊杂志都欢迎、或要求用TEX排版后投稿，世界上许多一流的出版社如 Kluwer、Addison-Wesley、牛津大学出版社等也利用 TEX系统出版书籍和期刊。

1.2 能编排中文文档的TEX系统

TEX是面向西文的排版系统，不能直接用于中文，为了充分利用TEX系统的功能，国内的一些学者开发了几种中文预处理系统，只需要用这些预处理系统处理一下中文的稿件，就可以使用TEX系统了。通常，将中文预处理系统和TEX系统统称为中西文TEX排版系统，其中最有名的是CCT系统和天元系统。

另外，还有**CJK宏包**，CJK是Werner Lemberg开发的，能够处理中文、日文、韩文的宏包。与CCT系统和天元系统相比， CJK与LATEX的结合更为紧

密。

1.3 TEX的排版过程

1. 首先，编写或修改文稿（源文件），在文件中插入排版命令。通常用WinEdt编写，源文件扩展名用tex；

2. 用LATEX软件处理TEX文件，产生dvi文件，如果这一部分出现错误，就要回到第一步进行修改；

3. 在屏幕上显示或在打印设备上输出dvi文件，查看文件内容和排版效果，如果不满意还要回到第一步；

4. 完成上面几步，就可以打印输出，或者照排制版了。

2 基本文档排版

用于排版的源文件包含两部分：一部分是正文，也就是排版需要输出的内容；另一部分是排版控制命令， 用于控制版面的式样、字体、字形等格式，控制命令是用反斜线 “\”引导的字符串。

排版命令分为两种，一种是“控制字”，一种是“控制符”。

控制字由反斜线和一个或多个英文字母组成，区分大小写，可以用任何非字母的字符（如空格、数字、汉字、括号、标点等）作为控制字的结束。

控制符由反斜线和一个符号（不是字母）组成。

LaTeX忽略命令后面的空格，如果你希望在命令后面得到一个空格，那么可以在命令后面加上“{ }”和一个空格，或者加上一个特殊的空白距离命令。“{ }”将阻止LaTeX吞噬掉命令后面的空格。

2.1 基本格式

LaTeX需要所处理的源文件遵从一定的结构，每个LaTeX文档必须以如下的命令开始：

```
\documentclass[基本字体大小]{文档类型}
```

这个命令指定了你所写文档的基本字体大小和类型:

基本字体大小：10pt，11pt，12pt。这是一个可选项，默认是10pt。

文章类型：article，book，letter，report。

在此之后，你可以加入控制文档式样的命令，或者使用如下的命令来调入一些宏集，进而为LaTeX系统增添一些新的功能。

```
\usepackage{宏包}
```

当完成所有的设置后，你可利用如下的命令来开始你的文档：

```
\begin{document}
```

现在你可以输入你所希望排版的文本和所使用的一些LaTeX命令。在文档的最后键入下面的命令来告诉LaTeX你的文档到此结束，从而使LaTeX忽略文档在此命令之后的部分。

```
\end{document}
```

注：1. 在语句\documentclass[···]{···}与 \begin{document}之间可以放置控制全局的命令，例如，放置一些宏包，这个区域我们通常称为“导言区”。

2. 我们使用的

```
\begin{???}
??????????????
\end{???}
```

称为“？？？”环境，但是在一个源文件中有且仅有一个“document”环境。

例如：整个文档排版就是一个环境

```
\begin{document}
??????????????
\end{document}
```

3. 两个简单的例子：

Example 1：

```
\documentclass[11pt]{article}
\begin{document}
This is an example!
\end{document}
```

其输出结果为：This is an example!

Example 2：

```
\documentclass{article}
\usepackage{CJK}
\begin{document}
\begin{CJK}{GBK}{song}
这是一个中文的例子！
\end{CJK}
\end{document}
```

其输出的结果为：这是一个中文的例子！

CJK环境中的参数GBK表示使用的编码为扩展国标码GBK大字符，song表示使用宋字体。

2.2 特殊字符的输入

在利用LaTeX排版时，大部分字符都可以通过键盘直接输入。但是，有些

字符在LaTeX中有特殊的用途，如果排版需要输出时，可以按照下面的对应输入：

输出字符	#	$	%	{	}	~	_	^	\
输入序列	\#	\$	\%	\{	\}	\~	_	\^	$\backslash$

另外，LaTeX中的公式及数字一般用 $ $ 包含起来，使其处于数学模式。

2.3 TeX中的空格与换行

LaTeX将空格和制表符等空白字符视为相同的空白距离（space）。多个连续的空白字符等同为一个空白字符。 在LaTeX文件中，每行开始的空白字符将被忽略，而单个的回车符被视为一空格。LaTeX使用空行来结束段落，两行文本中的空行标志上一段落的结束和新段落的开始。如同空格一样，多个空行所起的作用和一个空行的作用是相同的。

如果想得到一个空格的距离，可以用“\␣”得到。

如果想得到一个空行，可以用回车两次（注意，这样就转入下一段了），或者使用命令

“\\，*，\newline.”

带星号的*除了强制分行外，还禁止在分行处分页。

2.4 TeX中的分段和分页

2.4.1 分段

写作文档的一般主旨在于向读者传递观点、信息或知识。如果这些观点组织得很好，读者将能更好地理解。如果排版风格反映了内容的逻辑和语义结构，读者就能看见和感觉到文章的这种脉络。

LaTeX与其他排版系统的不同在于你只要告诉它文档的逻辑和语义结构，它就按照文档类型和各种格式文件指定的“规则”导出文档的排版风格。

在LaTeX中段落是最重要的文档单位。我们之所以称之为“文档单位”，因为段落是反映一个连贯思想或观点的排版风格形式。在前面，我们学习了如何用 \\ 强行断行，通过在源文件中留一空行强行转段。所以，如果开始一个新思想，就另起一段，否则，只应使用断行。 如果还犹豫是否应转段，可以把文档想象为观点和思想的传递者。如果旧的思路还在继续，就不应转段。如果同一段中出现了全新的思路，就应该另起一段。

在源文件中的一个空行，即连续按两次回车就产生一个空行，相当于一个排版的分段命令，TEX排版输出时， 空行的上下方文稿会被排成两个段落，这两个段落之间并不出现空行。源文件中连续多个空行与一个空行的效果相同。

如果不愿意在源文件中用空行来控制分段，则可用分段命令“\par”。

2.4.2 分页

对于分页，通常情况下TEX会自动分页，无需我们控制。但是如果由于排版或内容需要必须分页时，可以利用强制分页命令

```
\newpage
```

如果这个命令出现在两个段落之间，则这前后两个段落就会被排版到前后两页，并且当上一段所在页面不满时，该页下部会出现空白。 如果该命令出现在一段内，则在排版输出时会在命令所在的位置把一段强行变成两段，然后再在这两段之间强行分页。

建议分页或建议不分页的命令是

```
\pagebreak[0-4]
```

和

```
\nopagebreak[0-4]
```

介于0和4之间的参数表示了建议的力度。

用 \pagebreak 分页和用 \newpage 分页的效果是有区别的。如果 \pagebreak 命令出现在两个段落之间，则这前后两个段落就会被排版到前后两页，但当上一段所在页面不满时，该页会自动增加段落间距充满这一页，下部不会出现空白。 如果该命令出现在一段内，则在排版输出时会在命令所在的位置完成当前行，然后再强行分页。

2.5 TEX中的水平和垂直间距

2.5.1 水平间距

在LATEX中，得到指定长度空白的命令为

```
\hspace{ 长度 }
```

或

\hspace*{ 长度 }

两种格式的命令，在通常情况下，是没有区别的。但是当排版结果恰好是在新行的开始插入了空白间距，则不带星号的命令插入的空白会被忽略掉， 而带星号的命令在任何情况下都会插入指定长度的空白间距。

LaTeX中的长度单位有如下几种

mm	毫米
cm	厘米, 1cm=10mm
in	英寸, 1in=2.54cm
pt	点, 1in=72.27pt
em	与当前字体的尺寸有关，相当于大写字母 M 的宽度
ex	与当前字体的尺寸有关，相当于小写字母 x 的宽度

也可以使用下列命令，得到一定的水平间距

命令	\quad	\,	\:	\;	\!	\qquad
距离	基本长	3/18	4/18	5/18	-3/18	2

2.5.2 垂直间距

在两行之间插入垂直间距，可以使用如下命令

\vspace{ 长度 }

或

\vspace*{ 长度 }

需要说明的是，这两个命令，都不起到分行的作用，即不会将命令前后的文档分成两行，只是将所在行和下一行的间距变大。

与水平间距一样，两种格式的命令，在通常情况下，是没有区别的。 但是当排版结果恰好是在新一页的开始插入了垂直间距，则不带星号的命令插入的间距会被忽略掉，而带星号的命令在任何情况下都会插入指定长度的垂直间距。

3 论文文档的排版

进行论文文档的排版，我们要在

\documentclass[基本字体大小]{文档类型}

中，“文档类型”选择为“article”。整篇文章的字体的基本尺寸，由你所希望的情况而决定选择哪种情形。

3.1 选择文字的字体和大小

LaTeX根据文档的逻辑结构（章节、脚注、……）来选择合适的字体和字体大小。在某些情况下，你可能会想要手工改变文档使用的字体及其大小。为了完成这个目的，可以使用下表中的命令。

\textrm{…}	roman	\textsf{…}	sans serif
\texttt{…}	`typerwriter`	\textmd{…}	medium
\textbf{…}	**bold face**	\textsl{…}	*slanted*
\textsc{…}	SMALL CAPS	\textup{…}	upright
\textit{…}	*Italic*	\textnormal{…}	document font

\tiny{…}	tiny font	\scriptsize{…}	very small font
\footnotesize{…}	quite small font	\small{…}	small font
\normalsize{…}	normal font	\large{…}	large font
\Large{…}	larger font	\LARGE{…}	very large font
\huge{…}	huge	\Huge{…}	largest

每个字体的实际尺寸是一个设计问题，并且它依赖于文档所使用的文档类。下表列出了这些字体变换命令在标准文档类中的绝对尺寸。

字体尺寸命令	基准字体10pt	基准字体11pt	基准字体12pt
\tiny	5	6	6
\scriptsize	7	8	8
\footnotesize	8	9	10
\small	9	10	11
\normalsize	10	11	12
\large	12	12	14
\Large	14	14	17
\LARGE	17	17	21
\huge	21	21	25
\Huge	25	25	25

LaTeX 2_ε的一个重要特征是字体的各种属性是相互独立的，这意味着你可以改变字体的大小而仍然保留字体原有的粗体或者斜体的特性。

Example 3：

```
\documentclass[11pt]{article}
\begin{document}
{\small The small and \textbf{bold} Romans ruled}
{\Large all of great big \textit{Italy}.}

He likes {\LARGE large and {\small small} letters}.
\end{document}
```

其输出结果为：

The small and **bold** Romans ruled all of great big *Italy*.
He likes large and small letters.

使用字体命令的时候，大括号{ }扮演了一个重要角色。它们被用于建立所谓的组。组限制了大多数LaTeX命令的作用范畴。

如果段落在字体的作用范畴中结束，那么字体尺寸命令还将改变段落中行的距离。因此用于分组的反向大括号} 不应该太早结束。

如果你希望改变整段甚至更多文本的字体，你可能应该使用变换字体的一些环境。这将使你从众多的大括号中解脱出来。例如：

Example 4：

```
\documentclass[11pt]{article}
\begin{document}
\begin{Large}
This is not true.But then again, what is these days \ldots
\end{Large}
\end{document}
```

其输出结果为：

This is not true.But then again, what is these days ...

3.2 标题

3.2.1 文章标题

LaTeX中的关于整篇文章标题由命令

```
\maketitle
```

产生。LaTeX用特定的字体字号居中打印标题内容，如果标题过长，会自动分行，也可以用分行命令 “\\” 人工分行。

标题的内容必须在调用 \maketitle 以前，由命令

```
\title{...},
\author{...}
\date{...}（可选，系统默认给出当前时间）
```

定义。在命令 \author 中，可以输入几个用 \and 命令分开的名字。例如：

```
\author{作者1 \\ 工作单位1 \\ 地址1
        \and 作者2 \\ 工作单位2 \\ 地址2}
```

这些命令列在导言区或正文均可。而 \maketitle 通常紧接 \begin{document} 之后，通常是文章的第一条命令。

如果不希望使用上述几个命令生成格式化的标题，而想随心所欲地排版标题页，可以使用下述环境：

```
\begin{titlepage}
标题页内容
\end{titlepage}
```

其中“标题页内容”可以含有排版命令和文本，因此可以排成任何你所想要的格式。这个环境结束时，在新的一页显示排好的标题页， 所以这个环境不需要 \maketitle 命令。对于article类型文档，也可以利用这个环境打印单独的标题页。

3.2.2 章节标题

为便于读者理解，应该把文档划分为章，节和子节。LaTeX用特殊的命令支持这个工作， 这些命令把节的标题作为参量。我们的任务是按正确次序使用它们。

对“article”风格的文档，有下列分节命令：

```
\section{...}          \paragraph{...}
\subsection{...}       \subparagraph{...}
\subsubsection{...}
```

因为 article 风格的文档不划分为章，所以很容易把它作为一章插入书籍文档中。节之间的间隔，节的序号和标题的字号由LaTeX自动设置。分节的两个命令有些特殊性：

篇命令\part不影响章的序号。

附录命令\appendix不带参量，只把章的序号改用为字母标记。

3.3 目录

LaTeX在文档编译的最后一个循环中，提取节的标题和页码以生成目录。 命令

```
\tableofcontents
```

在其出现的位置插入目录。为了得到正确的目录内容，一个新文档必须编译两次。有时还要编译三次。届时LaTeX会通知你。

上面列出的分节命令也以“带星”的形式出现。“带星”的命令通过在命令名称后加 $*$ 来实现。 它们生成的节标题既不出现于目录，也不带序号。例如，命令 \section{Help} 的“带星”形式为 \section*{Help}。

目录出现的标题，一般与输入的文本完全一致。有时这是不可能的，因为标题太长排不进目录。在这种情况下，目录的条目可由真实标题前的可选参量确定。

3.4 摘要

生成文章摘要的环境是

```
\begin{abstract}
摘要内容
\end{abstract}
```

在book类文档中没有这种摘要；在report类文档中，摘要位于单独一页并带有页码；在article类文档中，摘要与文章标题打印在同一页， 但是若选用了“titlepage”环境，摘要也是单独占一页的。

3.5 分栏

如果全文都是双列分栏格式，可以在 \documentclass 命令中加入参数 twocolumn。例如：

```
\documentclass[twocolumn]{文档类型}
```

如果要生成单双列混合的分栏格式，或者超过两列的分栏格式，则要用到 multicol 宏包的 multicols 环境。

```
\usepackage{multicol}
```

然后将需要多列的地方使用环境：

```
\begin{multicols}{<列数>}
多个分栏
\end{multicols}
```

3.6 交叉引用

在书籍、报告和论文中，需要对图、表和文本的特殊段落进行交叉引用。LaTeX提供了如下交叉引用命令:

```
\label{标记}
\ref{标记}
\pageref{标记}
```

其中“标记”可以是字母、数字和其它字符（特殊字符除外）的任意组合。

\label 是为了在被引用的位置设置一个标记。可以设置在节、子节、图、表、定理或公式等后面， \ref 命令就会显示出标记所在位置相应对象的编号。

例如，在 3.6 节中，我们介绍了交叉引用的使用。就是利用这两个命令完成的。

\pageref 命令排印 \label 输入处的页码。和章节标题一样，使用的序号是前面编译所产生。 例如：

A reference to this subsection looks like: “see section 3.6 on page 16.”
是由命令

```
A reference to this subsection \label{sec:this} looks like:
“see section~\ref{sec:this} on page~\pageref{sec:this}.”
```

生成的。

3.7 参考文献

在科技书籍和论文中，常常要列举大量的参考文献，在正文中通过文献的编号进行引用。我们希望，当添加或删除一些文献之后，LaTeX重新进行编号并相应地更改引用的编号。

参考文献是用下列环境生成的：

```
\begin{thebibliography}{编号样本}
   \bibitem[记号]{引用标志}文献条目
   \bibitem[记号]{引用标志}文献条目
   .....
   \bibitem[记号]{引用标志}文献条目
\end{thebibliography}
```

对于没有可选项“记号”的条目，编译后 \bibitem 就会生成一个用方括号括起来的编号，如“[1]、[2]”等，这些编号是按递增顺序逐个排列的，当增加和减少文献条目时，这些编号会自动改变。

“引用标志”不可省略，但它不会显示在排版结果中。在正文中，使用

```
\cite{引用标志1，引用标志2，··· }
```

引用相应的一些文献，在排版结果中，上述引用命令就变成了被方括号括起来的文献编号。“引用标志” 可以用字母、数字和除了逗号以外的符号组成。

由于TeX在第一次编译时，先遇到引用，最后才编译文献目录，因此第一次编译时，遇到引用时只能用问号“？”代替文献标号，还需要再编译一次才能去掉问号，变成正确的编号。

“编号样本”可以是一个数字，它的位数应是参考文献最大编号的位数，以使缩进的文本能够对齐。 如果使用了“记号”，该文献就不会被编号，在相当于编号的位置显示了记号本身，在正文中通过引用标志引用，排版后显示为给定的记号。 当记号的宽度大于编号的宽度时，应在“编号样本”处写最宽的记号。

4 数学公式的排版

这一章，我们将介绍LaTeX的强大功能之所在，数学公式和符号的排版。LaTeX最具竞争力的特点就是能够排版出精美的数学公式。

4.1 概述

文本模式和数学模式对排版的要求是不同的，最简单的例子就是文本模式将通常的字母排成正体，空格可以分隔单词，而在数学模式下，则将表示变量的字母排成斜体，空格通通被吃掉，LaTeX会自动安排各部分的间距。

LaTeX使用一种特殊的模式来排版数学符号和公式。为了标明源文件中的某段内容是数学公式，必须将该段的内容的两边加上特殊的标记，以“通知”LaTeX系统对标记的内容按数学模式进行排版处理。需要注意这种标记是“成对”出现的，前一个标记表示进入数学模式，后一个标记表示已经退出数学模式。前后标记的不匹配是造成LaTeX排版错误的常见原因。

根据数学公式出现的位置的不同，将公式分为两类：出现在一行文字内部的称为**行内公式**，也称为**正文公式**；出现在两行之间的称为**行间公式**，也称为**显示公式**。

4.2 行内公式

如果公式比较短，我们通常将其用行内公式编排。段落中的数学表达式应该置于 \(和 \)， $ 和 $ 或者位于如下环境中

```
\begin{math}
公式内容
\end{math}
```

例如：Add a squared and b squared to get c squared. Or, using a more mathematical approach: $c^2 = a^2 + b^2$.

就是由

```
Add $a$ squared and \(b\) squared to get $c$ squared.
Or, using a more mathematical approach:
\begin{math}
c^{2}=a^{2}+b^{2}.
\end{math}
```

所输出的内容。

4.3 行间公式

对于较大的数学式子，最好的方法是使用行间公式形式来排版。将它们放置于 \[和 \]， 或 $$ 和 $$ 之间，也可以置于如下环境中

```
\begin{displaymath}
公式内容
\end{displaymath}
```

或者

```
\begin{equation*}
公式内容
\end{equation*}
```

得到没有编号的公式；

也可以用

```
\begin{equation}
公式内容
\end{equation}
```

得到自动编号的公式， 或者用

```
$$公式内容 \eqno 编号$$
$$公式内容 \leqno 编号$$
```

得到手动编号的公式。

例如：由命令

```
\begin{equation} \label{eq:eps}
\epsilon > 0
\end{equation}
From (\ref{eq:eps}), we gather \ldots
```

可以生成

$$\epsilon > 0 \tag{1}$$

From (1), we gather ...

对于生成多行的公式，我们可以由

```
\begin{eqnarray*}
第一行公式 \\
第二行公式 \\
  。。。    \\
第m行公式
\end{eqnarray*}
```

得到没有编号的公式。

4.4 上标与下标

我们将上标和下标统称为**角标**，LaTeX系统会用较小的字体排版角标，并升高上标或降低下标的位置。 上下标的命令分别为

```
^ {上标内容 }
```

和

```
_ {下标内容 }
```

当角标就是单个字符时，可不用花括号。但是，当角标为多个字符组成时，就需要使用花括号。 这是因为，数学模式中的命令仅对其后面第一个字符起作用。所以，如果你希望某一命令作用于多个字符的话，那么你就必须将它们放置于括号中，例如

```
\begin{equation}
a^x+y \neq a^{x+y}.
\end{equation}
```

的输出结果就是

$$a^x + y \neq a^{x+y}. \tag{2}$$

另外，既有上标又有下标时，输入的次序是无关的。例如：

`x_0^2` 和 `x^2_0`

输出的效果一样。

大家注意下列输入

```
$a_1$ \qquad $x^{2}$ \qquad $e^{-\alpha t}$
\qquad $a^{3}_{ij}$\qquad $e^{x^2} \neq {e^x}^2$.
```

的输出结果：

a_1 x^2 $e^{-\alpha t}$ a_{ij}^3 $e^{x^2} \neq e^{x2}$.

4.5 分数和分式

当输入较短的分数或分式时，最简单的办法就是使用斜线，如输入

```
$(x+y)/2$
```

就产生$(x+y)/2$。但是，有时我们需要使用带有水平分数线的分式，这样，我们就需要使用命令

```
\frac{分子 }{分母}
```

当分子分母为单个字符时，可以不用花括号括起来。

Example:

$$a_0+\frac{1}{a_1+\frac{1}{a_2+\frac{1}{a_3+\frac{1}{a_4}}}} \tag{3}$$

就是由下面的命令得到的，

```
\begin{equation}
a_0+\frac{1}{a_1+\frac{1}{a_2+\frac{1}{a_3+\frac{1}{a_4}}}}
\end{equation}
```

4.6 根式

LaTeX中根式的命令是

```
开平方：\sqrt{被开方表达式}
开高次方：\sqrt[n]{被开方表达式}
```

其中 n 是开方次数。当被开方表达式是单个字符时，可以不用花括号。另外，根式可以嵌套。

如果不要根号上面的横线，则需使用命令

```
\surd
```

举几个简单的例子：

```
$\sqrt{x}$ \qquad $\sqrt{x^{2}+\sqrt{y}}$
\qquad $\sqrt[3]{2}$\qquad $\surd[x^2 + y^2]$
```

其输出结果为

$\sqrt{x}$ $\sqrt{x^2+\sqrt{y}}$ $\sqrt[3]{2}$ $\surd[x^2+y^2]$

4.7 求和和积分

产生求和号和积分号的命令分别是

```
\sum
```

和

```
\int
```

它们通常都带有上下限（与上下标的输入法相同），而且符号大小在行内公式和行间公式是不同的， 因此也被称为“具有两种尺寸的数学符号”，并且求和号的上下限的位置也有区别。例如：

```
在行内公式中的求和号与积分号：
$\sum_{i=1}^n a_i,\ \int_a^b f(x)dx$
在行间公式中的求和号与积分号：
$$\sum_{i=1}^n a_i,\ \int_a^b f(x)dx$$
```

其输出结果为

在行内公式中的求和号与积分号： $\sum_{i=1}^{n} a_i,\ \int_{a}^{b} f(x)dx$

在行间公式中的求和号与积分号：

$$\sum_{i=1}^{n} a_i,\ \int_{a}^{b} f(x)dx$$

在带有上下限的符号后面紧跟着写上 \limits，然后再写上上下限，就会使上下限的位置移到符号的头顶和脚下。 例如：

```
在行内公式中的求和号与积分号（注意上下限位置）：
$\sum\limits_{i=1}^n a_i,\ \int\limits_a^b f(x)dx$
```

其输出结果为

在行内公式中的求和号与积分号（注意上下限位置）： $\sum\limits_{i=1}^{n} a_i,\ \int\limits_{a}^{b} f(x)dx$

与 \limits 相对应的命令是 \nolimits，这个命令总是强制上下限位于符号的右侧。

4.8 极限和二项式公式

LaTeX中极限的命令为

```
\lim
```

这个命令与积分和求和号一样，也是具有两种长度，大小在行内公式和行间公式是不同的。 例如：

注意这两个极限的区别： $\lim_{n\rightarrow \infty}\frac{1}{n}=0$ 与

$$\lim_{n\rightarrow \infty}\frac{1}{n}=0.$$

这两个公式是由下面的命令输出的：

```
注意这两个极限的区别：
$\lim_{n\rightarrow \infty}\frac{1}{n}=0$
与
$$\lim_{n\rightarrow \infty}\frac{1}{n}=0.$$
```

排版二项系数或类似的结构可以使用命令

```
{上面内容 \choose 下边内容}
```

或

```
{上面内容 \atop 下边内容}
```

第二个命令与第一个命令的输出相同，只是没有括号。

例如：我们可以由命令

```
\begin{displaymath}
{n \choose k}\qquad {x \atop y+2}
\end{displaymath}
```

输出

$$\begin{displaymath}
{n \choose k}\qquad {x \atop y+2}
\end{displaymath}$$

4.9 矩阵、行列式和线性方程组

矩阵、行列式和线性方程组均需要利用“阵列”环境把一些元素排成横竖都对齐的矩阵阵列，其命令格式为

```
\begin{array}[竖向位置]{列的格式}
第一行\\
第二行\\
......
最后一行
\end{array}
```

其中“竖向位置”确定阵列的内容在竖直方向上与当前外部文本行的相对位置， 默认（即无当前选项时）阵列相对于外部基线居中放置，可取值为

“t”阵列顶部基线与外部基线对齐；

“b”阵列底部基线与外部基线对齐。

“竖向位置”通常我们就取默认值。

而“列的格式”往往需要我们指定，其取值有

“l”对应的列的内容靠左对齐；

“c”对应的列的内容居中对齐；

“r”对应的列的内容靠右对齐。

阵列中一行的各个列之间是用符号“&”分隔的，每一行都用 \\ 结束。 但是阵列不能用于文字模式，只能用于数学模式。也就是，阵列环境必须位于数学环境中， 或者使用 \(和 \)，\$ 和 \$ 括起来。 如果是排版矩阵或行列式，我们还需要用可以调整大小的圆括号或方括号与竖线括起来。例如

$$\mathbf{X} = \begin{pmatrix} x_{11} & x_{12} & \dots & x_{1n} \\ x_{21} & x_{22} & \dots & x_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \dots & x_{nn} \end{pmatrix}$$

就是由下面命令输出的

```
\begin{displaymath}
\mathbf{X} =
\left( \begin{array}{cccc}
x_{11} & x_{12} & \ldots & x_{1n}\\
x_{21} & x_{22} & \ldots & x_{2n}\\
\vdots & \vdots & \ddots & \vdots\\
x_{n1} & x_{n2} & \ldots & x_{nn}\\
\end{array} \right)
\end{displaymath}
```

array环境也可以使用“.”作为隐藏右分隔符来排版只有一个大分隔符的表达式。例如

$$y = \begin{cases} a & \text{if } d > c \\ b + x & \text{if } d = c \\ l & \text{otherwise} \end{cases}$$

4.10 多行公式

单行公式和多行公式均是指行间公式。我们前面已经介绍了单行公式的自动编号和不要编号环境和方法。 这一部分，我们主要介绍多行公式的处理。可以认为多行公式是“equation”环境与前面“array”环境的结合。

命令为

```
\begin{eqnarray}
第一行公式\\
第二行公式\\
。。。
第m行公式
\end{eqnarray}
```

得到自动编号的公式。这些自动编号的公式也可以用命令

```
\nonumber
```

去掉某个公式的编号。或使用没有编号的环境

```
\begin{eqnarray*}
第一行公式\\
第二行公式\\
。。。
第m行公式
\end{eqnarray*}
```

多行公式的环境“eqnarray”与阵列环境“array”是不同的，有如下区别：

1、“eqnarray”环境本身就是数学模式，所以不需要再加数学模式标记。但是“array”环境是一个只能用于数学模式的表格环境， 它本身不会进入数学模式，必须人为地将它标记出来；

2、“eqnarray”环境可以自动对每行公式进行编号，而“array”环境不能自动编号，必要时可以显示指定公式的编号。

例如

$$\begin{aligned} d(uv) &= (uv)'dx = (u'v + uv')dx \qquad (4)\\ &= v(u')dx + u(v')dx \\ &= vdu + udv \qquad (5)\end{aligned}$$

就是如下输入的：

```
\begin{eqnarray}
d(uv)&=&(uv)'dx=(u'v+uv')dx\\
     &=&v(u')dx+u(v')dx\nonumber\\
     &=&vdu+udv
\end{eqnarray}
```

我们注意到：“& 指定内容 &”起到了对齐的作用，可以使得各个不同的行，以某些特定内容对齐。 例如：

$$\begin{aligned} f(x) &= \cos x \\ f'(x) &= -\sin x \\ \int_0^x f(y)dy &= \sin x \end{aligned}$$

其输入为：

```
\begin{eqnarray*}
f(x) & = & \cos x \\
f'(x) & = & -\sin x \\
\int_0^x f(y)dy &= & \sin x
\end{eqnarray*}
```

5 图表的处理

5.1 表格的输出

“tabular” 环境能用来排印带有水平和铅直表线的漂亮表格。LaTeX自动确定每一列的宽度。 命令为

```
\begin{tabular}[竖向位置]{列的格式}
第一行\\
第二行\\
......
最后一行
\end{tabular}
```

其中“竖向位置”确定表格的内容在竖直方向上与当前外部文本行的相对位置， 默认（即无当前选项时）表格相对于外部基线居中放置，可取值为

“t”表格顶部基线与外部基线对齐；

“b”表格底部基线与外部基线对齐。

“竖向位置”通常我们就取默认值。

而“列的格式”往往需要我们指定，其取值有

“l”对应的列的内容靠左对齐；

“c”对应的列的内容居中对齐；

“r”对应的列的内容靠右对齐。

对应于边界或列间分割线的格式项有

“|”画单条竖线，

“||”画双条竖线（两条相隔很近的竖线）。

表格中一行的各个列之间是用符号“&”分隔的，每一行都用 “\\” 结束。 列的内容可以是空白的，但列的分隔符“&”是不能省略的，除非从某一列的开始其后各列都是空白且不画边界竖线， 该列后面的“&”才可以省略。

LaTeX把表格的每一列的内容当成一个组，即相当于用花括号括起来，因此放在列文本中的一些命令如字体字号的声明， 其作用将被局限于该行该列中。

下面介绍几个在表格中画横竖线的命令：

\hline：画一条与表格同宽的水平横线，它只能出现在一行的最前面或者行结束符 “\\” 的后面，它在刚结束的行下面画一条横线。连续两个这个命令，会画出间距很小的两条横线。

\cline{m-n}：从第m列的开始到第n列结束位置画一条水平横线，它只能出

现在行结束符 “\\” 的后面，它在刚结束的行下面画一条横线。可以多次使用这个命令。

\vline：画一条竖直线，其高度等于所在位置的行高。放在表格环境中开始的“列格式”中的竖线会画出贯穿整个表格的竖直线， 而这个命令只画出等于行高的竖直线。

注意下面表格输出的差别：

编号	姓名	籍贯
1	刘建波	唐山
2	王三	天津
3	李四	北京

编号	姓名	籍贯
1	刘建波	唐山
2	王三	天津
3	李四	北京

编号	姓名	籍贯
1	刘建波	唐山
2	王三	天津
3	李四	北京

编号	姓名	籍贯
1	刘建波	唐山
2	王三	天津
3	李四	北京

编号	姓名	籍贯
1	刘建波	唐山
2	王三	天津
3	李四	北京

编号	姓名	籍贯
1	刘建波	唐山
2	王三	天津
3	李四	北京

前两个的输入命令分别为：

```
\begin{tabular}{|c|c|c|}
  \hline
  % after \\: \hline or \cline{col1-col2} \cline{col3-col4} ...
  编号 & 姓名 & 籍贯 \\
  1 & 刘建波 & 唐山 \\
  2 & 王三  & 天津 \\
  3 & 李四  & 北京 \\
  \hline
\end{tabular}
```

```
\begin{tabular}{|c|c|c|}
  \hline
  % after \\: \hline or \cline{col1-col2} \cline{col3-col4} ...
  编号 & 姓名 & 籍贯 \\\hline
  1 & 刘建波 & 唐山 \\\hline
  2 & 王三  & 天津 \\\hline
  3 & 李四  & 北京 \\
  \hline
\end{tabular}
```

5.2 图形的输出

LaTeX通过“figure”和“table”环境提供了处理图像或者图形等浮动对象的基本能力。

有几种办法可以通过使用基本LaTeX命令或者LaTeX扩展宏包来产生实际的图形。但是大多数用户发现这些命令相当难以理解。 因此我们不打算在这个手册里深入介绍这些内容。如果需要这方面的详细信息，请参阅 [4]。

5.2.1 简单图形的绘制

LaTeX提供了若干绘图命令，可以用来绘制直线、矢量线、圆、矩形、圆角矩形（卵形线）、Bezier曲线等基本图形。 在LaTeX源文件中可以插入一些外部图形。用LaTeX命令画图时，对于线段的倾角和圆的直径都有一定的限制，如果使用天元软件的内嵌绘图功能，则可以画任意斜率的线段和任意半径的圆，并且有更强的绘图功能， 例如只要给出曲线的方程，就能画出二维或三维曲线的图形。

一、坐标系的绘制

绘图是需要坐标系的，在LaTeX中使用的是我们通常的右手坐标系，即水平轴为 x 轴，纵轴为 y 轴，且向右和向上规定为正向。 设置单位长度的命令为

```
\setlength{\unitlength}{长度}
```

习惯上令长度单位为1mm，用其它值也可以。图形中的所有长度都以单位长度为度量单位。默认值为1pt。 当一个图形画好以后，只要重置单位长度，就相当于对图形放大或缩小。

绘制的任何一个图形总是和一个坐标系联系在一起，坐标轴的原点可以放在任何地方，通常是放在图形区域的左下角。

在LaTeX中绘制图形需要的命令是绘图环境：

```
\begin{picture}(高度,宽度)(x坐标,y坐标)
绘图命令
\end{picture}
```

环境中的（高度，宽度）是图形占据的区域的宽度和高度，就是说，是LaTeX为插入图形所保留的区域的高度和宽度。但是， 如果绘制的图形大于这个尺寸，就有可能与其它文本重叠，如果实际尺寸远远小于这个区域，排版出来就会出现很大的空白， 所以尽可能使得实际尺寸与这个尺寸相当。

TeX把一切对象都看成一个盒子，图形也是一个盒子，这个盒子的宽度和高度就是绘图环境中指定的（高度，宽度）。 绘制任何图形都固连着一个坐标系。绘图环境中的（x坐标，y坐标）确定了坐标系在盒子中的位置，也就是确定了整个图形在盒子中的位置。 如果在绘图中省略了这个坐标，则默认值就是（0, 0）。当绘图完成后，如果改变这个坐标，就相当于将坐标系在盒子中做了平移。

确定图形元素在坐标系中位置的命令（定位命令）有

```
\put(x,y){图形元素}
```

和

```
\multiput(x,y)(Δx,Δy){个数}{图形元素}
```

其中“图形元素”可以是字符串或是后面要讲的能够产生图形的绘图命令。(x,y) 是图形基准点的坐标。 命令“multiput”产生指定个数的图形元素，这是通过把同样的图形元素进行平移产生的，平移向量是$(\Delta x,\Delta y)$。

二、基本绘图

1．直线和矢量线

使用LaTeX的绘图命令可以绘制任意长度的水平直线段、竖直直线段和某些特定斜率的斜线段。命令为

```
\line(Δx,Δy){长度}
```

对于横线和竖线，长度就是它们的实际长度（以 \unitlength 的值为单位长度），对于其他倾斜的直线段， 这个长度就是它们的坐标。直线的起点由定位命令 \put 或 \multiput 给出的坐标(x,y)所确定。 直线的方向数是$(\Delta x,\Delta y)$。对于方向数有如下限制：

- 必须是整数；
- 取值只能是0，±1，±2，±3，±4，±5，±6；
- Δx 和 Δy 不能有公因子。

绘制带有箭头的矢量线的命令是

\vector(Δx,Δy){长度}

其参数和直线段的命令基本一致，只是在线段的终点画上了箭头。但是， 矢量线的方向数取值只能是0，±1，±2，±3，±4。

指定直线或曲线粗细的命令分别为

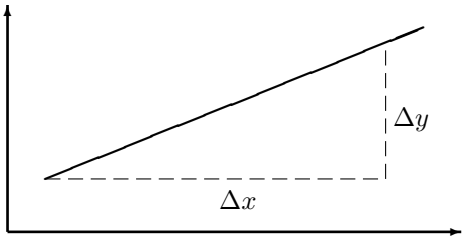
\thinlines　或　\thicklines

默认是细线。声明的作用直到遇到相反的作用为止，或遇到环境的结束。

对于横线和竖线，以及矢量线的直线部分，可以用下列声明指定线的粗细：

\linethickness{粗细}

参数值“粗细”是正的长度，必须带有单位，它与 \unitlength 的值无关。 但这个命令遇见粗线或细线的声明之后就失去了作用。



由下面的绘图命令集可以产生

```
\begin{center}
\setlength{\unitlength}{1mm}
\begin{picture}(60,30)
\linethickness{1pt}
\put(0,0){\vector(1,0){60}}
\put(0,0){\vector(0,1){30}}
\thicklines
\put(5,7){\line(5,2){50}}
\thinlines
\multiput(5,7)(3,0){15}{\line(1,0){2}}
\multiput(50,7.00)(0,3){6}{\line(0,1){2}}
\put(28,3){$\Delta x$}
\put(51,14){$\Delta y$}
\end{picture}
\end{center}
```

2. 圆

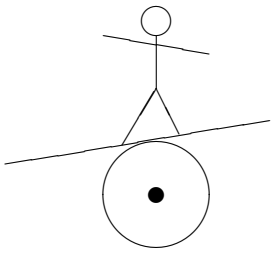
画圆的命令为

```
\circle{直径}
```

和

```
\circle*{直径}
```

其中第一个只画出了圆周，第二个命令画出的是实心圆。圆的圆周是由小圆弧组成的，由于事先设计好的小圆弧只有有限多种， 并不具有任意曲率，因此，LaTeX只能画指定直径的圆。规定，圆的直径不超过40pt（约14mm），实心圆直径不超过15pt（约5.3mm）。 画圆时，定位命令 \put 或 \multiput 给出的坐标(x,y)是圆心的坐标。 例如：下图命令和图形分别为



```
\setlength{\unitlength}{1mm}
\begin{center}
\begin{picture}(35,32)
\put(20,7){\circle*{2}}
\put(20,7){\circle{14}}
\put(0.00,11){\line(6,1){35}}
\put(15.5,13.6){\line(3,5){4.5}}
\put(20,21){\line(1,-2){3}}
\put(20,21){\line(0,1){7}}
\put(13,28){\line(6,-1){14}}
\put(20,30){\circle{4}}
\end{picture}
\end{center}
```

3. 图形中的盒子

在绘图环境中，能够产生无框、有框或虚线框的盒子。命令为

```
\makebox(宽,高)[位置]{文本}
\framebox(宽,高)[位置]{文本}
\dashbox{虚线长度}(宽,高)[位置]{文本}
```

其中(宽,高)是指矩形盒子的宽度和高度，均以 \unitlength 的值为单位。 可选项参数“位置”指文本在盒子中的位置，其值可取为：

t 文本水平方向居中，竖直方向靠在盒子顶部；

b 文本水平方向居中，竖直方向靠在盒子底部；

l 文本竖直方向居中，水平方向靠在盒子左边；

r 文本竖直方向居中，水平方向靠在盒子右边；

s 文本竖直方向居中，水平方向伸展充满整个盒子；

tl 文本位于盒子的左上角；

tr 文本位于盒子的右上角；

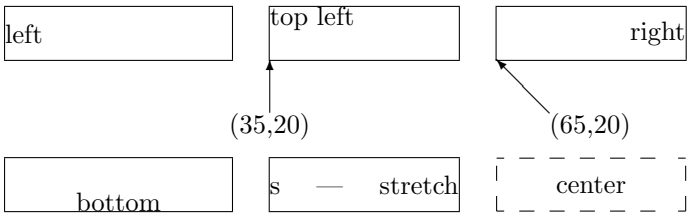
bl 文本位于盒子的左下角；

br 文本位于盒子的右下角。

如果省略了参数，则文本在水平和竖直方向均居中。命令中的“虚线长度”是组成虚线的小短线的长度， 两个小短线之间的空白也是这个长度，以 \unitlength 的值为单位。

例如下面的图形和命令集：

```
\setlength{\unitlength}{1mm}
\begin{center}
\begin{picture}(90,27)
\put(0,0){\framebox(30,7)[b]{bottom}}
\put(35,0){\framebox(25,7)[s]{s --- stretch}}
\put(65,0){\dashbox{2}(25,7){center}}
\put(0,20){\framebox(30,7)[l]{left}}
\put(35,20){\framebox(25,7)[lt]{top left}}
\put(65,20){\framebox(25,7)[r]{right}}
\put(35,13){\makebox(0,0)[t]{(35,20)}}
\put(35,13){\vector(0,1){7}}
\put(72,13){\makebox(0,0)[tl]{(65,20)}}
\put(72,13){\vector(-1,1){7}}
\end{picture}
\end{center}
```



4. 图形中的文本

在图形中插入文本的最简单方法就是在定位命令后面直接加上文本

```
\put(x,y){文本}
```

这样的文本不能太长，因为它不会自动换行。坐标(x,y)是行基本点的位置，也就是文本所占区域左下角的位置。

为了容易把文本放置在指定的位置，也常用前面所介绍的长度为零的盒子，即

```
\put(x,y){\makebox(0,0){文本}}
```

在文字模式或绘图环境中都可排版竖直堆叠的文本，命令为

```
\shortstack[位置]{一列文本}
```

“位置”的取值仍然为“l，r，c”，默认为“c”。例如

```
\setlength{\unitlength}{1mm}
\begin{center}
\begin{picture}(63,15)
\put(0,0){\shortstack[l]{这是\\ 左对齐\\ 堆叠文本}}
\put(20,0){\shortstack[r]{这是\\ 右对齐\\ 堆叠文本}}
\put(40,0){\shortstack{这是\\ 居中对齐\\ 堆叠文本}}
\put(60,0){\shortstack{A\\B\\C\\D}}
\end{picture}
\end{center}
```

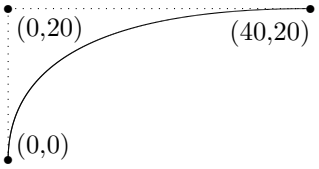
这是　　这是　　这是　　A
左对齐　　右对齐　　居中对齐　　B
堆叠文本　　堆叠文本　　堆叠文本　　C
D

5. 曲线

LaTeX中绘制曲线的命令是

\qbezier[数](x_1,y_1)(x_2,y_2)(x_3,y_3)

这是画一条从(x_1,y_1)到(x_3,y_3)的Bezier曲线，是用“数”+1个点画出来的。而(x_2,y_2)是控制曲线的弯曲程度的控制点：从该点到两个端点的连线是曲线的两条切线。例如：



```
\setlength{\unitlength}{1mm}
\begin{center}
\begin{picture}(41,21)(-0.5,-0.5)
\put(0,0){\circle*{1}}
\put(0,20){\circle*{1}}
\put(40,20){\circle*{1}}
\qbezier[20](0,0)(0,10)(0,20)
\qbezier[40](0,20)(20,20)(40,20)
\qbezier(0,0)(0,20)(40,20)
\put(1,0){\makebox(0,0)[bl]{(0,0)}}
\put(1,19){\makebox(0,0)[tl]{(0,20)}}
\put(40,18.5){\makebox(0,0)[tr]{(40,20)}}
\end{picture}\end{center}
```

其中的虚线也是用Bezier曲线命令画出的，只要使得命令中的三个点位于一条直线上，并且指定较少的点数就可以。 利用这种方法也可以画出任意斜率的直线，但只要指定的点足够多。

6. 使用amscd画交换图

首先要调用宏包“amscd”。首先观察下面的生成图和命令集

```
\begin{equation*}
\begin{CD}
A\prod B @>p_A>>A\\
@Vp_BVV        @AAfA\\
B          @<<g< Z
\end{CD}
\end{equation*}
```

$$\begin{array}{ccc} A\prod B & \xrightarrow{p_A} & A \\ {\scriptstyle p_B}\downarrow & & \uparrow{\scriptstyle f} \\ B & \underset{g}{\longleftarrow} & Z \end{array}$$

我们可以看出，@ >>>，@ <<<，@AAA，@VVV 分别表示向右、向左、向上、向下的箭头，而出现在第一个“<”或“>”后面的数学字符会用小字体打印在水平箭头的上面， 出现在第二个“<”或“>”后面的数学字符则会用小字体打印在水平箭头的下面。

类似地，出现在第一个“A”或“V”后面的数学字符会用小字体打印在竖直箭头的左面， 出现在第二个“A”或“V”后面的数学字符则会用小字体打印在竖直箭头的右面。

下面再看一个例子：

$$\begin{CD}
G @= G'\\
@. @|\\
@. H
\end{CD}$$

```
$$\begin{CD}
G@=G'\\
@.@|\\
@.H
\end{CD}$$
```

我们可以看到，命令@ =和@|能分别生成水平或竖直的两条平行线。此外，如果位于四角的某个对象不出现， 只要输入{ }或留空格就可以，但是如果某个箭头不出现，则要输入@.，不能只留空。

我们下面的另一个例子，还用到了LaTeX的幻影功能。

$$\begin{CD}
G @>\text{Clifford multiplication}>> H\\
@VfVV @AAgA\\
G' @<\phantom{\text{Clifford multiplication}}<\beta< H'
\end{CD}$$

```
$$\begin{CD}
G@>\text{Clifford multiplication}>> H\\
@VfVV   @AAgA\\
G'@<\phantom{Clifford multiplication}<\beta<H'
\end{CD}$$
```

其中我们看到了如何使用幻影功能，就是利用命令

```
\phantom{文本}
```

使得“文本”的内容只占用内容的空间，但不显示。

5.2.2 图形的插入

LaTeX中插入图形的宏包通常有“graphics”以及其扩展的“graphicx”，常用的还有“color”宏包。 利用这些宏包可以很容易地插入外部彩色或黑白的图形，还可对这些图形进行剪裁、放缩、翻转和旋转。

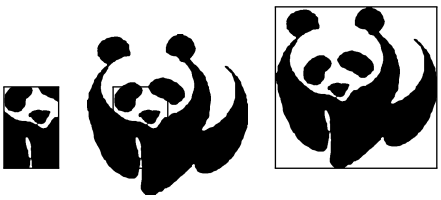
一、基本图形的插入

基本的插入图形的命令为

```
\includegraphics[左下角x,y][右上角x,y]{图形文件}
\includegraphics*[左下角x,y][右上角x,y]{图形文件}
```

其中带星号的命令插入剪裁后的图形，剪裁区域的左下角与右上角的坐标由命令中的可选项决定。如果省略了左下角的坐标， 则默认左下角坐标为（0,0）。不带星号的命令，是插入整个图形，剪裁范围的坐标仅仅用于确定图形占用的区域，当剪裁范围与图形边界盒子不相符时， 有可能插入的图形与周边文字重叠或出现较多空白。

```
\setlength{\unitlength}{1mm}
\setlength{\fboxsep}{0pt}
\begin{center}
\fbox{\includegraphics*[10,10][30,40]{pic/panda.eps}}\qquad
\fbox{\includegraphics[10,10][30,40]{pic/panda.eps}}\qquad
\fbox{\includegraphics{pic/panda.eps}}
\end{center}
```



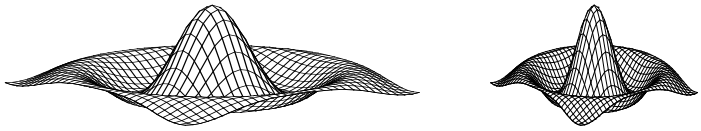
二、图形的放大和缩小

图形按照比例进行放缩的命令是

```
\scalebox{横向放缩因子}[竖向放缩因子]{插入的图形}
```

其中“竖向放缩因子”是可选项，若不用该项，则竖直方向和水平方向作相同的放缩，此时可保持图形的横竖比例不变。 放缩因子可以是负值，表示向反方向放缩。“插入的图形”可以是整个图形，也可以是经过剪裁以后的图形。 例如，下面的例子

```
\setlength{\unitlength}{1mm}
\begin{center}
\scalebox{0.4}[0.2]{\includegraphics{pic/pic.eps}}\quad
\scalebox{0.2}{\includegraphics{pic/pic.eps}}
\end{center}
```



利用这个命令我们还可以得到扁体字和瘦体字，例如：

```
\setlength{\unitlength}{1mm}
\begin{center}
\scalebox{2.0}[0.5]{中国}\quad
\scalebox{0.5}[2.0]{中国}
\end{center}
```

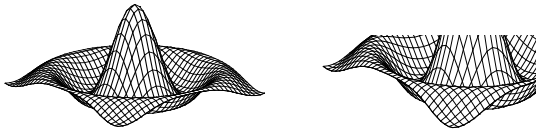
中国 中国

放缩到指定大小的命令是

```
\resizebox{宽度}{高度}{插入的图形}
```

其中“高度”“宽度”都要指定，但其中一个可以是{!}，这表示在一个方向上放缩到指定尺寸时， 仍保持图形原有的横竖比例不变。 例如：

```
\setlength{\unitlength}{1mm}
\begin{center}
\resizebox{4cm}{25mm}{\includegraphics{pic/pic.eps}}\quad
\resizebox{1.2in}{!}{\includegraphics*[100,430][400,600]
{pic/pic.eps}}
\end{center}
```



三、水平旋转和翻转

将图形水平翻转的命令是

```
\reflectbox{插入的图形}
```

例如（这是对前面图形的翻转，注意头的方向）：

```
\setlength{\unitlength}{1mm}
\begin{center}
\includegraphics{pic/panda.eps}\qquad
\reflectbox{\includegraphics{pic/panda.eps}}\\\vspace{2mm}
\includegraphics*[10,10][30,40]{pic/panda.eps}\qquad
\reflectbox{\includegraphics*[10,10][30,40]{pic/panda.eps}}
\end{center}
```

将图形进行旋转的命令是

```
\rotatebox{旋转角度}{插入的图形}
```

其“旋转角度”是以度为单位，逆时针方向为正。例如：

```
\setlength{\unitlength}{1mm}
\begin{center}
\includegraphics{pic/panda.eps}
\rotatebox{-30}{\includegraphics{pic/panda.eps}}
\rotatebox{30}{\includegraphics{pic/panda.eps}}
\rotatebox{180}{\reflectbox{\includegraphics
{pic/panda.eps}}}
\end{center}
```

另外，外部的图形也可以放在绘图环境中，例如：

```
\setlength{\unitlength}{1mm}
\begin{center}
\begin{picture}(75,40)
\put(0,30){这是一幅}
\put(18,0){\resizebox{!}{4cm}{\includegraphics
{pic/photo.eps}}}
\put(60,10){风景照片}
\end{picture}
\end{center}
```

这是一幅  风景照片

而且，上述命令的参数“插入的图形”可以换成普通文本或其它对象，例如：

```
\rotatebox{-15}{我是中国人}\quad
\rotatebox{15}{我是中国人}\quad
\rotatebox{90}{我是中国人}\quad
\rotatebox{180}{我是中国人}\quad
\rotatebox{270}{我是中国人}\quad
\rotatebox{360}{我是中国人}
```


References

[1] 陈志杰 等， *LaTeX入门与提高（第二版）*, 高等教育出版社，北京，2006。

[2] 邓建松 等， *LaTeX 2ε科技排版指南*, 科学出版社，北京，2001。

[3] Tobias Oetiker 等，*一份不太简短的 LaTeX 2ε介绍*，2003。

[4] 王磊，*LaTeX 2ε插图指南*，2000。