ＬＡＴＥＸ科技排版

目录

第一章 基础知识

1.1 LATEX 的产生与发展

1.1.1 TEX

TEX 是Donald E. Knuth 编写的一个以排版文章及数学公式为目标的计算机程序。1977 年，在意识到恶劣的排版质量正在影响自己的著作及文章后，Knuth开始编写TEX 排版系统引擎，探索当时开始进入出版工业的数字印刷设备的潜力，尤为希望能扭转排版质量下滑的这一趋势。我们现在使用的TEX 系统发布于1982 年，在1989 年又稍做改进，增加了对8 字节字符及多语言的支持。TEX以其卓越的稳定性、可在不同类型的电脑上运行以及几乎没有缺陷而著称。TEX的版本号不断趋近于 π，现在为3.141592。

TEX 发音为“Tech”，其中“ch” 和德语“Ach” 及苏格兰语“Loch”中的“ch” 类似。“ch” 源自希腊字母，希腊文中，X 是字母“ch” 或“chi”。TEX 同时也是希腊单词texnologia (technology) 的第一个音节。在ASCII 文本环境中，TEX 写作TeX。

1.1.2 LATEX

LATEX 是一个宏集，它使用一个预先定义好的专业版面，可以使作者们高质量的排版和打印他们的作品。LATEX 最初由Leslie Lamport 编写，它使用TEX 程序作为排版引擎。现在LATEX 由Frank Mittelbach 负责维护。

LATEX 的发音为“Lay-tech” 或“Lah-tech”。如果在ASCII 环境中引用LATEX，你可以输入LaTeX。LATEX 2 ε 的发音为“Lay-tech two e”，在ASCII 环境中写作LaTeX2e。

1.2 基础

1.2.1 作者、图书设计者和排版者

出版的第一步就是作者把打好字的手稿交给出版公司，然后由图书设计者来决定整个文档的布局（栏宽、字体、标题前后的间距、……）。图书设计者会把他的排版说明写进作者的手稿里，再交给排版者，由排版者根据这些说明来排版全书。

一个图书设计者要试图理解作者写作时的意图。他要根据手稿的内容和他自己的职业知识来决定章节标题、文献引用、例子及公式等等。

在一个LATEX 环境中，LATEX 充当了图书设计者的角色，而TEX 则是其排版者。但是LATEX“仅仅”是一个程序，因此它需要很多的指导。作者必须提供额外的信息，来描述其著作的逻辑结构。这些信息是以“LATEX 命令” 的形式写入文档中的。

这和大多数现代文字处理工具，如MS Word 及Corel WordPerfect 所采用的所见即所得(WYSIWYG) 的方式有很大区别。使用这些工具时，作者在向计算机中输入文档的同时，通过互动的方式确定文章的布局。作者可以从屏幕上看到作品的最终打印效果。而使用LATEX时，一般是不能在输入文档的同时看到最终的输出效果的，但是使用LATEX处理文档之后，便可以在屏幕上预览最终的输出效果。因此在真正打印文档之前还是可以做出改正的。

1.2.2 版面设计

排版设计是一门工艺。不熟练的作者认为书籍设计仅仅是个美学问题，因而经常会犯严重的格式错误“如果一份文档从艺术的角度看起来不错，那么它的设计就是成功的”。不过作为一份用来阅读而不是挂在画廊里的文档，可读性和可理解性远比漂亮的外观重要。例如：

- 必须选定字号和标题的序号，使读者能清楚的理解章节的结构。
- 每一行既要足够短以避免读者眼睛疲劳，又要足够长以维持页面的美观。

在使用所见即所得系统(WYSIWYG) 时，作者经常会写出一些看上去漂亮，但结构欠清晰或不连贯的文章来。LATEX 通过强制作者声明文档的逻辑结构，来避免这些排版格式错误。然后，LATEX 再根据文档的结构选择最合适的版面格式。

1.2.3 优势和不足

使用所见即所得(WYSIWYG) 的人和使用LATEX 的人遇到一起时，他们经常讨论的话题就是“相比一般文字处理软件，LATEX 的优势(advantages of LATEX)”或者不足。

LATEX 优于一般文字处理软件之处可归纳如下：

- 提供专业的版面设计，可以使一份文档看起来就像“印刷品”一样。
- 可以方便的排版数学公式。
- 用户只需要学一些声明文档逻辑结构的简单易懂的命令，而不必对文档的实际版面修修补补。
- 可以容易的生成像脚注、引用、目录和参考文献等很多复杂的结构。
- 很多不被基本LATEX支持的排版工作，可以由添加免费的宏包来完成。例如，支持在文件中插入PostScript 格式图像的宏包及排版符合各类准确标准的参考文献的宏包等。
- LATEX 鼓励作者按照合理的结构写作，因为LATEX 就是通过指明文档结构来进行排版工作的。
- TEX，作为LATEX 2 ε 的排版引擎，不仅免费，而且具有很高的可移植性，几乎可以在任何硬件平台上运行。

LATEX 也有一些不足之处。

- 没有原则的人不能使用LATEX 很好地工作……
- 尽管可以调节预先定义好的文档版面布局中的一些参数，但设计一个全新的版面还是很困难的，并会耗费大量时间。
- 很难用LATEX 来写结构不明、组织无序的文档。
- 即使有一个令人鼓舞的开端，你也可能无法完全掌握其精髓。

1.3 LATEX 源文件

LATEX 源文件为普通的ASCII 文件，你可以使用任何文本编辑器来创建。LATEX 源文件不仅包含了要排版的文本，而且也包含了告诉LATEX 如何排版这些文本内容的命令。

1.3.1 空白距离

空格和制表符等空白字符在LATEX 中被看作相同的空白距离(space)。**多个连续的空白字符等同于一个空白字符**。在句首的空白距离一般会被忽略，单个空行也被认为是一个“空白距离”。

两行文本间的空白行标志着上段的结束和下段的开始。多个空白行的

作用等同于一个空白行。下面便是一个例子。

```
It does not matter whether you
enter one or several       spaces
after a word.

An empty line starts a new
paragraph.
```

1.3.2 特殊字符

下面的这些字符是LATEX 中的保留字符(reserved characters)，它们或在LATEX 中有特殊的意义，或不一定存在于所有字库中。如果你直接在文本中输入这些字符，通常它们不会被输出，而且还会导致LATEX 做一些你不希望发生的事情。

$ % ^ & _ { } \

如你看到的，在这些字符前加上反斜线，它们就可以正常的输出到文档中。

\# \$ \% \^{} \& _ \{ \} \ {}

其他一些特殊符号可以由数学环境中的特殊命令或重音命令得到。反斜线\不能通过在其前面加另一个反斜线得到(\\);这是一个用来换行的命令。

1.3.3 LATEX 命令

LATEX 命令(commands) 是大小写敏感的，有以下两种格式：

- 以一个反斜线(backslash) \ 开始，命令名只由字母组成。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的结束。
- 由一个反斜线和非字母的字符组成。

LATEX 忽略命令之后的空白字符。如果你希望在命令后得到一个空格，可以在命令后加上{} 和一个空格，或加上一个特殊的空格命令。{} 将阻止LATEX 吃掉命令后的所有空格。

```
I read that Knuth divides the
people working with \TeX{} into
\TeX{}nicians and \TeX perts.\\
Today is \today.
```

有些命令需要一个参数(parameter)，该参数用花括号(curly braces) { } 括住并写在命令的后面。一些命令支持可选参数(optional parameters)，可选参数可用方括号(square brackets) [ ] 括住， 然后

写在命令的后面。下面的例子中使用了一些LATEX 命令，不要着急，后面将解释它们的含义。

```
You can \textsl{lean} on me!
Please, start a new line
right here!\newline
Thank you!
```

1.3.4 注释

当LATEX 处理一个源文件时，如果遇到一个百分号%，LATEX 将忽略%后的该行内容，换行符以及下一行前的空白字符。

我们可以据此在源文件中写一些注释，而且这些注释并不会出现在最后的排版结果中。

```
This is an % stupid
% Better: instructive <----
example: Supercal%
ifragilist%
icexpialidocious
```

符号% 也可以用来断开不能含有空白字符或换行符的较长输入内容。如果注释的内容较长，你可以使用verbatim 宏包提供的comment 环境。当然，在使用该环境前，你要在文档的导言区（后面将会解释其含义） 加上命令 \usepackage{verbatim}。

```
This is another
\begin{comment}
rather stupid,
but helpful
\end{comment}
example for embedding
comments in your document.
```

需要注意的是以上做法在数学环境等复杂环境中不起作用。

1.4 源文件的结构

当LATEX 2 ε 处理源文件时，它希望源文件遵从一定的结构(structure)。因此，每个源文件都要以如下命令开始

```
\documentclass{...}
```

这条命令指明了你所写的源文档的类型。然后，你就可以加入控制整篇文档样式的命令，或者载入一些为LATEX 增加新特性的宏包(package)。可以用如下命令载入一个宏包

```
\usepackage{...}
```

当完成所有的设置工作后，你可以用下面的命令开始文档的主体

```
\begin{document}
```

现在你就可以输入带有LATEX 命令的正文了。在文章末尾使用命令

```
\end{document}
```

来告诉LATEX 文档已经结束。LATEX 会忽略此命令后的所有内容。

图1.1 显示的是一个简单的LATEX 2 ε 文档的结构。一个较为复杂的源文件(input file) 结构如图1.2 所示。

```
\documentclass{article}
\begin{document}
Small is beautiful.
\end{document}
```

图1.1 - 一个简单的LATEX 源文件。

```
\documentclass[a4paper,11pt]{article}
% define the title
\author{H. Partl}
\title{Minimalism}
\begin{document}
% generates the title
\maketitle
% insert the table of contents
\tableofcontents
\section{Some Interesting Words}
Well, and here begins my lovely article.
\section{Good Bye World}
\ldots{} and here it ends.
\end{document}
```

图1.2 - article 类LATEX 源文件.

1.5 上机过程

1.5.1 一般操作过程

LATEX 本身没有图形用户界面或漂亮的按钮，它仅仅是一个处理你提供的源文件的程序。有些LATEX 安装版本提供了一个前端图形界面，你可以通过点击按钮来编译你的源文件。其他的一些系统上可能就要使用命令来编译源文件，下面演示的就是如何在一个基于文本的系统上让LATEX 编译你的源文件。需要注意：以下演示的前提是LATEX 已经正确的安装到了你的电脑中。

1. 创建并编辑你的源文件。源文件必须是普通的ASCII 格式。在Unix 系统下，所有的编辑器都可以创建这样的文件。在Windows 系统下，你必须确保文件以ASCII 或普通文本格式保存。当选取你源文件的文件名时，确保它的扩展名是.tex。

2. 运行LATEX 编译你的源文件。如果成功的话，你将会得到一个.dvi 文件。为了得到目录和所有的内部引用，可能要多次运行LATEX。当源文件中存在错误时，LATEX 会告诉你错误并停止处理源文件。输入ctrl-D 可以返回到命令行。

```
latex foo.tex
```

3. 现在可以通过几种方法来预览得到的DVI 文件。你可以使用下列命令将文件显示到屏幕上

```
yap foo.dvi
```

你也可以使用Ghostscript 将dvi 文件转换成PostScript 文件来打印或预览。

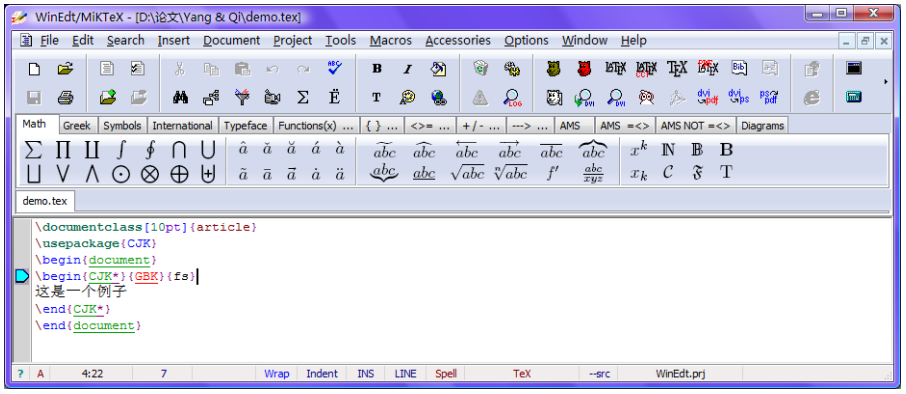
```
dvips -Pcmz foo.dvi -o foo.ps
```

如果你的LATEX 系统中带有dvipdf 工具的话，就可以直接将.dvi 文件转换成pdf 文件。

```
dvipdf foo.dvi
```

1.5.2 CTEX 操作

1. 运行WinEdit,并录入编辑源文件



2. 单击工具栏上的  按钮进行编译.

3. 单击工具栏上的  按钮进行预览.

1.6 文档布局

1.6.1 文档类

当LATEX 处理源文件时，首先需要知道的就是作者所要创建的文档类型。文档类型可由\documentclass 命令来指定。

\documentclass[options]{class}

class 指定想要的文档类型。

表1.1 给出了一些文档类型的解释。LATEX 2 ε 发行版中还提供了其他一些文档类，像信件和幻灯片等。通过options 参数可以定制文档类的属性。不同的选项之间须用逗号隔开。标准文档类的最常用选项如表1.2所示。

表1.1 文档类

article	排版科学期刊、演示文档、短报告、程序文档、邀请函……
proc	一个基于article的会议文集类。
minimal	非常小的文档类。只设置了页面尺寸和基本字体。主要用来查错。
report	排版多章节长报告、短篇书籍、博士论文……
book	排版书籍。
slides	排版幻灯片。该文档类使用大号sans serif字体。也可以选用FoilTEXa来得到相同的效果

表1.2 - 文档类选项。

10pt, 11pt, 12pt	设置文档中所使用的字体的大小。如果该项没有指定，默认使用10pt 字体。
a4paper, letterpaper, . . .	定义纸张的尺寸。缺省设置为letterpaper。此外，还可以使用a5paper, b5paper, executivepaper以及legalpaper。
fleqn	设置行间公式为左对齐，而不是居中对齐。
leqno	设置行间公式的编号为左对齐，而不是右对齐。
titlepage, notitlepage	指定是否在文档标题(document title)后另起一页。article文档类缺省设置为不开始新页，report和book 类则相反。
onecolumn, twocolumn	LATEX 以单栏(one column)或双栏(two column)的方式来排版文档。
twoside, oneside	report 类指定文档为双面或单面打印格式。article 和为单面(single sided)格式，book类缺省为双面(double sided)格式。注意该选项只是作用于文档样式，而不会通知打印机以双面格式打印文档。
landscape	将文档的打印输出布局设置为landscape模式。
openright, openany	决定新的一章仅在奇数页开始还是在下一页开始。在文档类型为article时该选项不起作用，因为该类中没有定义“章”(chapter)。report类默认在下一页开始新一章而book类的新一章总是在奇数页开始。

例子：一个LATEX 源文件以下面一行开始

\documentclass[11pt,twoside,a4paper]{article}

这条命令会引导LATEX 使用article 格式、11 磅大小的字体来排版该文档，并得到在A4 纸上双面打印的效果。

1.6.2 宏包

排版文档时，你可能会发现某些时候基本的LATEX 并不能解决你的问题。如果想插入图形(graphics)、彩色文本(coloured text) 或源代码到你的文档中，你就需要使用宏包来增强LATEX 的功能。可使用如下命令调用宏包

\usepackage[options]{package}

这里package 是宏包的名称，options 是用来激活宏包特殊功能的一组关键词。很多宏包随LATEX 基本发行版一起发布(见表1.3)，其他的则单独发

布。你可以在所安装的LATEX 系统中找到更多的宏包相关信息。

现代的TEX发行版包含了大量免费的宏包。

表1.3 - 随LATEX一起发行的宏包。

doc	排版LATEX 的说明文档
exscale	提供了按比例伸缩的数学扩展字体。
fontenc	指明使用哪种LATEX 字体编码(font encoding)。
ifthen	提供如下形式的命令‘if...then do...otherwise do...’
latexsym	提供LATEX 符号字体。
makeidx	提供排版索引的命令。
syntonly	编译文档而不生成dvi文件（常用于查错）。
inputenc	指明使用哪种输入编码，如ASCII,ISO Latin1, ISO Latin2, 437/850 IBM code pages, Apple Macintosh, Next, ANSI Windows 或用户自定义编码。

1.6.3 页面样式

LATEX 支持三种预定义的页眉/页脚(header/footer) 样式,称为页面样式(pagestyle)。如下命令

\pagestyle{style}

中的style参数确定了使用哪一种页面样式。表1.4 列出了预定义的页面样式。

表1.4 - LATEX预定义的页面样式。

plain	在页脚正中显示页码。这是页面样式的缺省设置。
headings	在页眉中显示章节名及页码，页脚空白。（本文即采用此样式）
empty	将页眉页脚都设为空白。

可以通过如下命令来改变当前页面的页面样式

\thispagestyle{style}

1.7 各类 LATEX 文件

使用LATEX 时，你可能很快发现自己置身于各种不同扩展名(extension) 或毫无线索的文件形成的迷宫之中。下面的列表解释了在使用LATEX 时可能遇到的文件类型。要注意的是，下表不是所有的扩展名列表:

.tex LATEX 或TEX 源文件。可以使用latex 命令编译。

.sty LATEX 宏包文件。可以使用\usepackage 命令将宏包文件载入到你的LATEX文档中。

.dtx 文档化TEX 文件。这是LATEX 宏包文件的主要发布格式。如果编译.dtx 文档，将会得到其中包含的LATEX 宏包文件的文档化宏代码。

.ins 对应.dtx 文件的安装文件。如果你从网上下载了一个LATEX 的宏包文件，其中一般会包含一个.dtx 文件和一个.ins 文件。使用LATEX 处理.ins 文件可以解开.dtx 文件。

.cls 定义文档外观形式的类文件，可以通过使用\documentclass 命令选取。

.fd 字体描述文件，可以告诉LATEX 有关新字体的信息。

下面这些文件是使用LATEX 处理源文件时产生的：

.dvi 设备无关文件。这是运行LATEX 编译的主要结果。你可以使用DVI 预览器预览其内容或使用dvips 或其他程序输出到打印机。

.log 记录了上次编译时的详细信息。

.toc 储存了所有的章节标题。下次编译时将读取该文件并生成目录。

.lof 和.toc 文件类似，可生成图形目录。

.lot 和.toc 文件类似，可生成表格目录。

.aux 用来向下次编译传递信息的辅助文件。主要储存交叉引用的相关信息。

.idx 如果文档中包含索引，LATEX 将使用该文件存储所有的索引词条。此文件需要使用makeindex 处理，详见第4.3 节。

.ind 处理过的.idx 文件。下次编译时将读入到你的文档中。

.ilg 和.log 文件类似，记录了makeindex 命令运行的详细信息。

1.8 中文支持

1.8.1 中文预处理系统

- CCT： 科学院张林波教授开发，在文档格式方面非常符合中文习惯.一个简单的老版本CCT 格式的例子是：

```
\documentclass{cctart}
\begin{document}
\kaishu 这是中文楷体字。
\end{document}
```

这个例子需要保存为.ctx 后缀的文件，然后用cct 命令进行预处理，生成同名.tex 文件。再用LATEX 编译，生成的DVI 文件需要patchdvi 进行处理后才能用DVI 浏览器进行查看或者用dvips 转换成PostScript 文件。假设文件名是test.ctx ，完整的编译过程是：

```
cct test
latex test
patchdvi -r600x600 -b test.dvi temp.dvi
del test.dvi
ren temp.dvi test.dvi
dvips test
```

新版的CCT 除了保留原来的处理方式以外,增加了两种新的处理方式。第一种是用TEX 的处理来代替原来的cct.exe 的预处理。这种方式的源文件和老的文件相同，但是不用再存成.ctx 为后缀的文件，也不用cct 命令进行预处理。除此之外，其余和老的处理方式基本相同。

第二种处理方式是采用CJK 的中文字库，需要在系统中安装好CJK 字库。与第一种方式的主要区别就在于去掉了patchdvi 处理DVI 文件的需要。使用上，是在\documentclass 命令中加上参数CJK 。具体的例子如下：

```
\documentclass[CJK]{cctart}
\begin{document}
\kaishu
这是中文楷体字。
\end{document}
```

这个例子可以象英文文档一样的编译得到正确的输出。

- TY： 华东师大肖刚、陈志杰等教授开发。

1.8.2 CJK

由德国 W. Lemberg 开发，可以同时处理中、日、韩三国文字。

在安装好CJK 的系统中，下面这个例子可以象英文文档一样的编译得到正确的输出。

```
\documentclass{article}
\usepackage{CJK}
\begin{document}
\begin{CJK*}{GBK}{kai}
这是中文楷体字。
\end{CJK*}
\end{document}
```

CJK 宏包有两种不同的处理方式。一个是

```
\begin{CJK}....
...
\end{CJK}
```

称为CJK 模式；另一个是

\begin{CJK*}....
...
\end{CJK*}
称为CJK* 模式。两个模式的区别在于CJK* 会忽略CJK 字符之间的空格，这是我们中文的习惯。而CJK 则使用英文的习惯，即词之间保留空格，当然如果空格多于一个，TEX 也会忽略多余的空格。

1.9 大型文档

当处理大型文档时，最好将文档分割成为几部分。LATEX 有两个命令可以帮助你完成这项工作。
\include{filename}

你可以使用该命令将名为filename.tex 的文档内容插入到当前文档中。需要注意的是，在处理插入的filename.tex 文档前，LATEX 会另起一页。

第二个命令只能在导言区使用。它可以让LATEX 仅读入某些\include 文件。
\includeonly{filename,filename,. . . }

这条命令在文档的导言区执行后，在所有的\include 命令中，只有文档名出现在\includeonly 的命令参数中的文档才会被导入。注意文档名和逗号之间不能有空格。

\include 命令会在新的一页上排版载入的文本。当使用\includeonly 命令时会很有帮助，因为即使一些载入的文本被忽略，分页处也不会发生变化。有些时候可能不希望在新的一页上排版载入的文本，这时可以使用命令
\input{filename}
\input 命令只是简单的载入指定的文本，没有其他限制。

如果想让LATEX 快速的检查文档中的错误，可以使用syntonly 宏包。它可以使LATEX 浏览整个文档，检查语法错误和使用的命令，但并不生成DVI 输出。在这种模式下，LATEX 运行速度很快，可以为你节省宝贵的时间。syntonly 宏包的使用非常简单：
\usepackage{syntonly}
\syntaxonly

如果想产生分页，只要注释掉第二行即可(在前面加上一个百分号%)。

第二章 文本排版

阅读了前一章之后，应该了解关于如何创建一个LATEX 文档的基本知识了。在这一章里，将补充其余部分，使你能够生成实际文档。

2.1 断行和分页

2.1.1 对齐段落

通常书籍是用等长的行来排版的。为了优化整个段落的内容,LATEX 在单词之间插入必要的断行点(line break) 和间隙。如果一行的单词排不下，LATEX 也会进行必要的断词。段落如何排版依赖于文档类别。通常，每一段的第一行有缩进，在两段之间没有额外的间隔。更多的内容请参考第6.3.2 节。

在特殊情形下，有必要命令LATEX 断行

\\ or \newline　　另起一行，而不另起一段。

*　　在强制断行后，还禁止分页。

\newpage　　另起一页。

\linebreak[n], \nolinebreak[n], \pagebreak[n], \nopagebreak[n]

上述命令的效果可以从它们的名称看出来。通过可选参量n，作者可以影响这些命令的效果。n可以取为0和4之间的数。如果命令的效果看起来非常差，把n取为小于4的数，可以让LATEX 在排版效果不佳的时候选择忽略这个命令。不要把这些“break” 命令与“new” 命令混淆。即使你给出了“break” 命令，LATEX 仍然试图对齐页面的右边界。

LATEX 总是尽可能产生最好的断行效果。如果断行无法达到LATEX 的高标准，就让这一行在段落的右侧溢出。然后在处理源文件的同时，报告溢出的消息(“overfull hbox”)。这最有可能发生在LATEX 找不到合适的地方断词的时候。你可以使用\sloppy 命令，告诉LATEX 降低一点儿标准。它通过增加单词之间的间隔，以防止出现过长的行，虽然最终的输出结果不是最优的。在这种情况下给出警告(“underfull hbox”)。在大多数情况下得到的结果看起来不会非常好。\fussy 命令把LATEX 恢复为缺省状态。

2.1.2 断词

必要时LATEX 就会断词。如果断词算法不能确定正确的断词点，可以使用如下命令告诉TEX 如何弥补这个缺憾。

命令

\hyphenation{word list}

使列于参量中的单词仅在注有“-” 的地方断词。命令的参量仅由正常字母构成的单词，或由LATEX 视为正常字母的符号组成。当断词命令出现时，根据正在使用的语言，断词的提示就已经被存好待选了。这意味着如果你在文档导言中设置了断词命令，它将影响英文的断词。如果断词命令置于\begin{document} 后面，而且你正使用比方babel 的国际语言支持宏包，那么断词提示在由babel 激活的语言中就处于活动状态。

下面的例子允许对“hyphenation” 和“Hyphenation” 进行断词， 却根本不允许“FORTRAN”, “Fortran” 和“fortran” 进行断词。在参量中不允许出现特殊的字符和符号。

例子：

\hyphenation{FORTRAN Hy-phen-a-tion}

命令\- 在单词中插入一个自主的断词点。它也就成为这个单词中允许出现的唯一断词点。对于包含特殊字符（例如：注音字符）的单词，这个命令是特别有用的，因为对于他们，LATEX 不会自动断词。

```
I think this is: su\-per\-cal\-%
i\-frag\-i\-lis\-tic\-ex\-pi\-%
al\-i\-do\-cious
```

命令

\mbox{text}

保证把几个单词排在同一行上。在任何情况下，这个命令把它的参量排在一起。

```
My phone number will change soon.
It will be \mbox{0116 291 2319}.
The parameter
\mbox{\emph{filename}} should
contain the name of the file.
```

命令\fbox 和\mbox 类似，此外它还能围绕内容画一个框。

2.2 内置字符串

在前面的例子中，你已经看到用来排版特殊文本字符串的一些非常简单的LATEX命令了。

命令	例子	描述
\today	July 17, 2008	今日日期
\TeX	TEX	你最喜爱的排版工具
\LaTeX	LATEX	游戏的名目
\LaTeXe	LATEX 2 ε	现在的化身

2.3 特殊字符和符号

2.3.1 引号

在LATEX 中，用两个`（重音）产生左引号，用两个’（直立引号）产生右引号。一个` 和一个’ 产生一个单引号。

```
``Please press the `x' key.''
```

当然这种实现机制不是最理想的，无论字体如何，它总是一个反向的勾号或者重音符(`) 当左引号，直立引号(') 当右引号。

2.3.2 破折号和连字号

LATEX 中有四种短划(dash) 标点符号。连续用不同数目的短划，可以得到其中的三种。第四个实际不是标点符号，它是数学中的减号：

```
daughter-in-law, X-rated\\
pages 13--67\\
yes---or no? \\
$0$, $1$ and $-1$
```

这些短划线是: ‘-’ 连字号(hyphen)，‘ - ’ 短破折号(en-dash)，‘—’ 长破折号(em-dash) 和‘−’ 减号(minus sign)。

2.3.3 波浪号(~)

波浪号经常和网址用在一起。它在LATEX 中，可用 \~ 产生，但其结果：~ 却不是你真正想要的。试一下这个：

```
http://www.rich.edu/\~{}bush \\
```

http://www.clever.edu/$\sim$demo

2.3.4 度的符号(°)

下面的例子演示了在LATEX 中如何排版度的符号(degree symbol)：

```
It's $-30\,^{\circ}\mathrm{C}$.
I will soon start to
    super-conduct.
```

textcomp 宏包里有另外一个度的符号\textcelsius。

2.3.5 省略号(...)

在打字机上,逗号(comma) 或句号(period) 占据的空间和其他字母相等。在书籍印刷中，这些字符仅占据一点儿空间，并且与前一个字母贴得非常紧。所以不能只键入三个点来输出“省略号” (ellipsis)，因为间隔划分得不对。有一个专门的命令输出省略号。它被称为\ldots

```
Not like this ... but like
this:\\ New York, Tokyo,
Budapest, \ldots
```

2.3.6 连字

一些字母组合不是简单键入一个个字母得到得的，而实际上用到了一些特殊符号。

效果应为ff fi fl ffi. . . 而不是ff fi fl ffi . . .
这就是所谓的连字(ligature)，在两个字母之间插入一个\mbox{}，可以 禁止连字。对于由两个词构成的单词，这可能是必要的。

```
Not shelfful\\
but shelf\mbox{}ful
Not shelfful
but shelfful
```

2.3.7 注音符号和特殊字符

LATEX 支持来自许多语言中的注音符号(accent) 和特殊字符(special character)。表2.2 就字母列出了所有的注音符号。对于其他字母也自然有效。在字母i 和j 上标一个注音符号，它的点儿必须去掉。这个可由\i 和\j 做到。

```
H\^otel, na\"\i ve, \'el\`eve,\\
sm\o rrebr\o d, !`Se\ norita!,\\
Sch\"onbrunner Schlo\ss{}
Stra\ss e
```

表2.2 - 注音符号和特殊字符。

ò	\`o	ó	\'o	ô	\^o	o	\ o
ō	\=o	ȯ	\.o	ö	\"o	ç	\c c
ŏ	\u o	ǒ	\v o	ő	\H o	ǫ	\c o
ọ	\d o	o͟	\b o	o͡o	\t oo		
œ	\oe	Œ	\OE	æ	\ae	Æ	\AE
å	\aa	Å	\AA				
ø	\o	Ø	\O	ł	\l	Ł	\L
ı	\i	ȷ	\j	☒	!`	☒	?`

2.4 单词间隔

为了使输出的右边界对齐，LATEX 在单词间插入不等的间隔。在句子的末尾插入的空间稍多一些，因为这使得文本更具可读性。LATEX 假定句子以句号、问号或惊叹号结尾。如果句号紧跟一个大写字母，它就不视为句子的结尾。因为一般在有缩写的地方，才出现句号紧跟大写字母的情况。

作者必须详细说明这些假设中的任何一个例外。空格前的反斜线符号产生一个不能伸长的空格。波浪字符‘~’ 也产生一个不能伸长的空格，并且禁止断行。句号前的命令\@ 说明这个句号是句子的末尾，即使它紧跟一个大写字母。

```
Mr. Smith was happy to see her\\
cf. Fig. 5\\
I like BASIC\@. What about you?
```

命令

\frenchspacing

能禁止在句号后插入额外的空白，它告诉LATEX 在句号后不要插入比正常字母更多的空白。除了参考文献，这在非英语语言中非常普遍。如果使用了\frenchspacing，命令\@ 就不必要了。

2.5 标题、章和节

为便于读者理解，应该把文档划分为章，节和子节。LATEX 用专门的命令支持这个工作，这些命令把节的标题作为参量。你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{...}　　\subsection{...}　　\subsubsection{...}
\paragraph{...}　　\subparagraph{...}

如果想把文档分成几个部分而且不影响章节编号，你可以使用

\part{...}

当你使用report 或者book 类的时候，可以用另外一个高层次的分节命令

\chapter{...}

因为article 类的文档不划分为章，所以很容易把它作为一章插入书籍中。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别：

- 命令\part 不影响章的序号。
- 命令\appendix 不带参量，只把章的序号改用为字母标记。

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(table of contents) 内容，一个新文档必须编译(“LATEXed”) 两次。有时还要编译第三次。

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，目录的条目可由实际标题前的可选参量确定。

```
\chapter[Title for the table of contents]{A long and especially
boring title, shown in the text}
```

整篇文档的标题(title) 由命令

\maketitle

产生。标题的内容必须在调用\maketitle 以前，由命令

\title{...}, \author{...} 和可选的\date{...}定义。在命令\author 的参量中，可以输入几个用\and 命令分开的名字。

除了上面解释的分节命令，LATEX 2ε 引进了其他三个命令用于book风格的文档。它们对划分出版物有用，也能如愿改变章的标题和页码：
\frontmatter 应接着命令\begin{document} 使用。它把页码更换为罗马数字，而且章节不计数。当你使用带星的分节命令(例如，\chapter*{Preface}) 时，这些章节就不会出现在目录里。
\mainmatter 应出现在书的第一章前面。它启用阿拉伯数字的页码计数器，并对页码重新计数。
\appendix 标志书中附录材料的开始。该命令后的各章序号改用字母标记。
\backmatter 应该插入与书中最后一部分内容的前面， 如参考文献和索引。在标准文档类型中，它对页面没有什么效果。

2.8 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(crossreferences)。LATEX 提供了如下交叉引用命令

\label{marker}, \ref{marker} 和\pageref{marker}

其中marker 是用户选择的标识符。如果在节、子节、图、表或定理后面输入\label 命令，LATEX 把\ref 替换为相应的序号。\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。

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

2.9 脚注

命令

\footnote{footnote text}

把脚注内容排印于当前页的页脚位置。脚注命令总是置于(put)其指向的单词或句子的后面。脚注是一个句子或句子的一部分，所以应用逗号或句号结尾。

```
Footnotes\footnote{This is a footnote.} are often used by people
using \LaTeX.
```

2.10 强调

如果文本是用打字机键入的， 用下划线来强调重要的单词。
\underline{text}

但是在印刷的书中，用一种斜体字体排印要强调的单词。LATEX 提供命令
\emph{text}
来强调文本。这些命令对其参量的实际作用效果依赖于它的上下文：

```
\emph{If you use emphasizing inside a piece of emphasized text, then
\LaTeX{} uses the \emph{normal} font for emphasizing.}
```

请注意要求LATEX 强调什么和要求它使用不同字体的不同效果:

```
\textit{You can also \emph{emphasize} text if it is set in italics,}
\textsf{in a \emph{sans-serif} font,} \texttt{or in
\emph{typewriter} style.}
```

2.11 环境

为了排版专用的文本，LATEX 定义了各种不同格式的环境(environment)：

\begin{environment} text \end{environment}

其中environment 是环境的名称。只要保持调用顺序，环境可以嵌套。

\begin{aaa}...\begin{bbb}...\end{bbb}...\end{aaa}

下面的章节对所有重要的环境都做了解释。

2.11.1 Itemize、Enumerate 和 Description

itemize 环境适用于简单的列表，enumerate 环境适用于有排列序号的列表，而description 环境用于带描述的列表。

```
\flushleft
\begin{enumerate}
\item You can mix the list environments to your taste:
 \begin{itemize}
\item But it might start to look silly.
\item[-] With a dash.
\end{itemize}
\item Therefore remember:
```

```
    \begin{description}
    \item[Stupid] things will not become smart because they are in a list.
    \item[Smart] things, though, can be presented beautifully in a list.
    \end{description}
    \end{enumerate}
```

2.11.2 左对齐、右对齐和居中

flushleft 和flushright 环境分别产生左对齐(left-aligned) 和右对齐(rightaligned)的段落。center 环境产生居中的文本。如果你不输入命令\\ 指定断行点， LATEX 将自行决定。

```
\begin{flushleft}
This text is\\ left-aligned. \LaTeX{} is not trying to make each line the same length.
\end{flushleft}
```

```
\begin{flushright}
This text is right-\\aligned. \LaTeX{} is not trying to make each line the same length.
\end{flushright}
```

```
\begin{center}
At the centre\\of the earth
\end{center}
```

2.11.3 引用、语录和韵文

quote 环境可以用于引文、语录和例子。

```
A typographical rule of thumb
for the line length is:
\begin{quote}
On average, no line should
be longer than 66 characters.
\end{quote}
This is why \LaTeX{} pages have
such large borders by default
and also why multicolumn print
is used in newspapers
```

有两个类似的环境：quotation 和verse 环境。quotation 环境用于超过几段的较长引用，因为它对段落进行缩进。verse 环境用于诗歌，在诗歌中断行很重要。在一行的末尾用\\ 断行，在每一段后留一空行。

```
I know only one English poem by
heart. It is about Humpty Dumpty.
\begin{flushleft}
\begin{verse}
Humpty Dumpty sat on a wall:\\
Humpty Dumpty had a great fall.\\
All the King's horses and all
the King's men\\
Couldn't put Humpty together
again.
\end{verse}
\end{flushleft}
```

2.11.4 摘要

科学出版物惯常以摘要开始，来给读者一个综述或者预期。LATEX 为此提供了abstract 环境。一般abstract 用于article 类文档。

```
\begin{abstract}
The abstract abstract.
\end{abstract}
```

2.11.5 原文打印

位于\begin{verbatim} 和\end{verbatim} 之间的文本将直接打印，

包括所有的断行和空白，就像在打字机上键入一样， 不执行任何LATEX 命令。在一个段落中，类似的功能可由

\verb+text+

完成。+ 仅是分隔符的一个例子。除了* 或空格，可以使用任意一个字符。

```
The \verb|\ldots| command \ldots
\begin{verbatim}
10 PRINT "HELLO WORLD ";
20 GOTO 10
\end{verbatim}
```

```
\begin{verbatim*}
the starred version of
the verbatim
environment emphasizes
the spaces in the text
\end{verbatim*}
```

带星的命令\verb 能以类似的方式使用：

\verb*|like this :-) |

verbatim 环境和\verb 命令不能在其他命令的参数中使用。

2.11.6 表格

tabular 环境能用来排版带有水平和垂直表线的漂亮表格(table)。LATEX 自动确定每一列的宽度。

命令 \begin{tabular}[pos]{table spec}

的参量table spec 定义了表格的格式。用一个l 产生左对齐的列，用一个r 产生右对齐的列，用一个c 产生居中的列； 用p*{width}* 产生相应宽度、包含自动断行文本的列； | 产生垂直表线。

如果一列里的文本太宽，LATEX 不会自动折行显示。使用p*{width}* 你可以定义如一般段落里折行效果的列。

参量pos 设定相对于环绕文本基线的垂直位置。使用字母t 、b 和c 来设定表格靠上、靠下或者居中放置。

在tabular 环境中，用& 跳入下一列，用\\ 开始新的一行， 用\hline 插入水平表线。用\cline*{j-i}* 可添加部分表线， 其中j 和i 分别表示表线的起始列和终止列的序号。

```
\begin{tabular}{|p{4.7cm}|}
\hline
Welcome to Boxy's paragraph.
We sincerely hope you'll
all enjoy the show.\\
\hline
\end{tabular}
```

```
\begin{tabular}{|r|l|}
\hline
7C0 & hexadecimal \\
3700 & octal \\ \cline{2-2}
11111000000 & binary \\
\hline \hline
1984 & decimal \\
\hline
\end{tabular}
```

表格的列分隔符可由@{...} 构造。这个命令去掉表列之间的间隔，代之为两个花括号间的内容。一个用途在于下面要解释的十进制数对齐问题。另一个可能应用在于用@{} 压缩表列右端空间。

```
\begin{tabular}{l}
\hline
leading space left and right\\
\hline
\end{tabular}
```

```
\begin{tabular}{@{} l @{}}
\hline
no leading space\\
\hline
\end{tabular}
```

由于没有内建机制使十进制数按小数点对齐18，我们可以使用两列“作弊”达到这个目的： 整数向右，小数向左对齐。\begin{tabular} 行中的命令@{.} 用一个“.” 取代了列间正常间隔，从而给出了按小数点列对齐的效果。不要忘记用列分隔符(&) 取代十进制小数点！使用命令 \multicolumn 可在数值“列”上放置一个列标签。

```
\begin{tabular}{c r @{.} l}
Pi expression &
\multicolumn{2}{c}{Value} \\
\hline
$\pi$ & 3&1416 \\
$\pi^{\pi}$ & 36&46 \\
$(\pi^{\pi})^{\pi}$ & 80662&7 \\
\end{tabular}
```

```
\begin{tabular}{|c|c|}
\hline
\multicolumn{2}{|c|}{Ene} \\
\hline
Mene & Muh! \\
\hline
\end{tabular}
```

用表格环境排印的材料总是呆在同一页上。如果要排印一个长表格，可以看一下supertabular 和longtabular 环境

2.12 浮动体

大多数出版物含有许多图片和表格。由于不能把它们分割在不同的页面上，所以需要专门的处理。如果一个图片或一个表格太大在当前页面排不下，一个解决办法就是每次新开一页。这个方法在页面上留下部分空白，效果看起来很差。

对于在当前排不下的任何一个图片或表格，其解决办法是把它们“浮动”到下一页，与此同时当前页面用正文文本填充。LATEX 提供了两个浮动体(floating bodies)环境；一个用于图片，一个用于表格。要充分发挥这两个环境的优越性，应该大致了解LATEX 处理浮动体的内在原理。但是浮动可能成为令人沮丧的主要原因，因为LATEX 总不把浮动体放在你想要的位置。

首先看一下供浮动使用的LATEX 命令：

包含在figure 环境或table 环境中的任何材料都将被视为浮动内容。两个浮动环境都支持可选参数

\begin{figure}[placement specifier] 或\begin{table}[. . .] 称为placement specifier，它由浮动许可放置参数写成的字符串组成。请见表2.9。这个参数用于告诉LATEX 浮动体可以被移放的位置。一个placement specifier由一串浮动体许可放置位置(float-placing permissions) 构成. 参见表2.9。

表2.9 - 浮动体放置许可。

Spec	浮动体许可放置位置……
h	here在文本的确切位置上，对于小的浮动体很有用。
t	在页面的顶部(top)
b	在页面的底部(bottom)
p	在一个只有浮动体的专门的页面(page)上。
!	忽略阻止浮动体放置的大多数内部参数a。

一个表格可以由如下命令，例如

\begin{table}[!hbp]

开始，placement specifier [!hbp] 允许LATEX 把表格就放当前页，或放在某页的底部(b)，或放在一个专门的浮动页上(p)，严格按照放置说明符放置即使看起来不好(!)。如果没有给定放置说明符,缺省值为[tbp]。LATEX 将按照作者提供的placement specifier ，安排它遇到的每一个浮动体。如果浮动体在当前页不能安排，就把它寄存在图片或表格等待队列中19。当新的一页开始的时候，LATEX 首先检查是否可能用等待队列中的浮动体填充一个专门的“浮动”页面。如果这不可能，就像对待刚在文本中出现的浮动体一样，处理等待队列中的第一个浮动体：LATEX 重新尝试按照其相应的放置说明符（除了不再可能的‘h’）来处理它。文本中出现的任何一个新浮动体寄存在相应的等待队列中。对于每一种浮动体，LATEX 保持它们出现的顺序。这就说明了为什么一个不能安排的图片把所有后来的图片都推到文档末尾的原因。所以：如果LATEX 没有像你期望的那样安排浮动体，那么经常是仅有一个浮动体堵塞了两个等待队列中的某一个。

仅给定单个placement specifiers 是允许的，但这会引起问题。如果在指定的位置安排不了，它就会成为障碍，堵住后续的浮动体。不要单独使用参数[h]，在LATEX 最近的版本中，它的效果太差了以至于被[ht] 自动替换。虽然对浮动体问题已经作了些说明，对table 和figure 环境还有些内容要交代。使用

\caption{caption text}

命令，可以给浮动体定义一个标题。序号和字符串“图”或“表”将由LATEX 自动添加。

两个命令

\listoffigures 和\listoftables

用起来和\tableofcontents 命令类似，分别排版一个图形目录和表格目录。在这些目录中，所有的标题都将重复。如果打算使用长标题，就必须准备一个能放进目录的，较短版本的标题。即在\caption 命令后面的括号内输入较短
版本的标题。

```
\caption[Short]{LLLLLooooooonnnnnggggg}
```

利用\label 和\ref，在文本中可以为浮动体创建交叉引用。

下面的例子画一个方形，并将它插入文档。如果想在完成的文档中为你打算嵌入的图片保留空间，你可以利用这个例子。

```
Figure \ref{white} is an example of Pop-Art.
\begin{figure}[!hbp]
\makebox[\textwidth]{\framebox[5cm]{\rule{0pt}{5cm}}}
\caption{Five by Five in Centimetres.\label{white}}
\end{figure}
```

在上面的例子中，为了把图片就放在当前位置(h)， LATEX 尝试得很辛苦(!)。如果这不可能，它将试图把图片安排在页面的底部(b)。如果不能将图片安排在当前页面，它将决定是否可能开一个浮动页面以放置这张图片或来自表格等待队列中的一些表格。如果没有足够的材料来填充一个专门浮动页面，LATEX 就开一个新页，像对文本中刚出现的图片一样，再一次处理这个图片。在一些情况下，可能需要使用命令
\clearpage 或者甚至是\cleardoublepage
它命令LATEX 立即放置等待队列中所有剩下的浮动体，并且开一新页。命令
\cleardoublepage 甚至会命令LATEX 新开奇数页面。

在本书的后面，将介绍如何在LATEX 2 ε 文档中插入PostScript 图形。

2.13 保护脆弱命令

作为命令（如\caption 或\section）参量的文本，可能在文档中出现多次（例如，在文档的目录和正文中）。当用于类似\section 的参量时，一些命令会失效。它们被称为脆弱命令(fragile commands)。\footnote 或\phantom 是脆弱命令的例子。这些脆弱命令需要的，正是保护。把\protect 命令放在它们前面，就能保护它们。

\protect 仅仅保护紧跟其右侧的命令，连它的参量也不惠及。在大多

数情形下，过多的\protect 并不碍事。

```
\section{I am considerate \protect\footnote{and protect my
footnotes}}
```

第三章 数学公式

现在你已经准备好了。那么在这一章里，让我们来着手于TEX 的强大之处：数学排版。但是，要提醒你的是，本章只是浅尝辄止。可对很多人来说，这里所讲述的内容已很受用，如果你在这里找不到你所需数学排版的解决方案的话，也请不要灰心。极有可能在*AMS*-LATEX中能找到针对你的问题的某个解决方案。

3.1 综述

LATEX 使用一种特有的模式来排版数学(mathematics) 公式。数学公式允许以行间形式排版在一个段落之中， 也可以以独立形式排版，此时段落可能会被拆开。处于段内的数学文本要放在\(与\) 之间，$ 与$ 之间，或者\begin{math} 与\end{math} 之间。

```
\TeX{} is pronounced as
\(\tau\epsilon\chi\).\\[6pt]
100 m$^{3}$ of water\\[6pt]
This comes from my
\begin{math}\heartsuit\end{math}
```

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

当你希望把自己的一些较长的数学方程或是公式单独的放在段落之外的时候，那么你最好显示(display) 它们，而不要拆开此段落。为此，你可以把它们放在\[与\] 之间，或者\begin{displaymath} 与\end{displaymath} 之间。

```
Add $a$ squared and $b$ squared to get $c$ squared. Or, using
a more mathematical approach:
\begin{displaymath}
c^{2}=a^{2}+b^{2}
\end{displaymath}
or you can type less with:
\[a+b=c\]
```

如果你希望LATEX 给你的方程编上号，你可以使用equation 环境。然后你就可以用\label 来给一个方程加上标签并在文中的某处用\ref 或amsmath 宏包中的\eqref 命令来引用它。

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

注意一下公式排版样式的不同，前者是行间式样，后者是显示式样

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

```
\begin{displaymath}
\lim_{n \to \infty}
\sum_{k=1}^n \frac{1}{k^2}
= \frac{\pi^2}{6}
\end{displaymath}
```

数学模式和**文本模式**都一些不同之处。例如，在数学模式中：

1.大多数的空格和断行没有任何意义，而且所有的空隙要么是从相应数学表达式中自然的生成，要么是用一些专门的命令来指定，如\,，\quad 或\qquad。

2.空白行是不允许的。每个公式只能为一段。

3.每一个字母都会被认为是一个变量名，且会相应被排版为此种样式。如果你想要在公式中排版普通的文本（直立字体和普通字距），那么你必须要把这些文本放在\textrm{...} 命令中（参阅第3.7 节）

```
\begin{equation}
\forall x \in \mathbf{R}:
\qquad x^{2} \geq 0
\end{equation}
```

```
\begin{equation}
x^{2} \geq 0\qquad
\textrm{for all }x\in\mathbf{R}
\end{equation}
```

数学家要使用空心粗体(“blackboard bold”)， 要包含此字体，得用到amsfonts 或是amssymb 宏包的\mathbb命令。上面的例子就变成

```
\begin{displaymath}
x^{2} \geq 0\qquad
\textrm{for all }x\in\mathbb{R}
\end{displaymath}
```

3.2 数学模式的群组

大部分数学模式的命令只对其后的一个字符有效，因此，如果你希望一个命令对多个字符起作用，你必须把它们放在一个群组中，使用花括号：{...}.

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

3.3 数学公式的基本元素

这一节将介绍数学排版中的最重要的一些命令。详细的数学排版符号的命令列表，可参阅第3.10节。

小写希腊字母(Greek letters) 的输入为\alpha、\beta、\gamma……，大写字母的输入为\Gamma、\Delta ……

```
$\lambda,\xi,\pi,\mu,\Phi,\Omega$
```

指数和下标可以使用^ 和_ 两个符号来指定。

```
$a_{1}$ \qquad $x^{2}$ \qquad
$e^{-\alpha t}$ \qquad
$a^{3}_{ij}$\\
$e^{x^2} \neq {e^x}^2$
```

平方根(square root) 输入用\sqrt；n 次根用\sqrt[n] 来得到。根号的大小由LATEX自动决定。如果仅仅需要根号，可以用\surd 得到。

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

命令\overline 和\underline 产生水平线,它们会被放在表达式的正上方或是正下方。

```
$\overline{m+n}$
```

命令\overbrace 和\underbrace 可以在一个表达式的上方或下方生成水平括号

```
$\underbrace{a+b+\cdots+z}_{26}$
```

为了给变量增加数学重音符号，如小箭头或是~(tilde)， 你可以使用表3.1 所列出的命令。覆盖多个字符的宽“帽子”和宽~号，可以由\widehat 和\widetilde 得到。’ 符号则给出了一个撇号(prime)。

```
\begin{displaymath}
y=x^{2}\qquad y'=2x\qquad y''=2
\end{displaymath}
```

向量可以通过在一个变量上方添加小箭头(arrow symbols) 来指定。为此，使用\vec 命令即可。\overrightarrow 和\overleftarrow 这两个命令可以用来表示一个从A 到B 的向量。

```
\begin{displaymath}
\vec a\quad\overrightarrow{AB}
\end{displaymath}
```

通常没有必要打出一个明显的点号来表明乘法运算；但是有时候也需要它来帮助读者分清一个公式。在这些情况下，你应该使用\cdot 命令。

```
\begin{displaymath}
v = {\sigma}_1 \cdot {\sigma}_2
{\tau}_1 \cdot {\tau}_2
\end{displaymath}
```

log 等类似的函数名通常是用直立字体,而不是如同变量一样用斜体,因此LATEX 提供了以下的命令来排版这些最重要的**函数名**:
\arccos \cos \csc \exp \ker \limsup \arcsin \cosh \deg \gcd \lg \ln \arctan \cot \det \hom \lim \log \arg \coth \dim \inf \liminf \max \sinh \sup \tan \tanh \min \Pr \sec \sin

```
\[\lim_{x \rightarrow 0}
     \frac{\sin x}{x}=1\]
```

取模函数(modulo function),有两个命令:\bmod 用于二元运算“$a \bmod b$”,而\pmod 则用于表达式如“$x \equiv a \pmod{b}$”。

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

一个上下的**分式**(fraction) 可用\frac{...}{...} 命令得到。而其倾斜形式如1/2,有时是更好的选择,因为对于简短的分子分母来说,这看上去更美观。

```
$1\frac{1}{2}$ hours
\begin{displaymath}
\frac{ x^{2} }{ k+1 }\qquad
x^{ \frac{2}{k+1} }\qquad
x^{ 1/2 }
     \end{displaymath}
```

排版**二项式系数**或类似的结构,你可以使用amsmath 宏包中的\binom 命令。

```
\begin{displaymath}
\binom{n}{k}\qquad\mathrm{C}_n^k
     \end{displaymath}
```

对于**二元关系**,有时候你需要到把符号互相堆积起来。\stackrel 命令会把其第一个参数中的符号以上标大小放在第二个上面,而第二个符号则以正常的位置摆放。

```
\begin{displaymath}
\int f_N(x) \stackrel{!}{=} 1
\end{displaymath}
```

积分号(integral operator) 可以用\int 产生,**求和号**(sum operator) 用\sum 命令， 而**乘积号**(product operator) 要用\prod 命令。上限和下限用^ 和_ 来指定，如同上标与下标一样。

```
\begin{displaymath}
\sum_{i=1}^{n} \qquad
\int_{0}^{\frac{\pi}{2}} \qquad
\prod_\epsilon
\end{displaymath}
```

为了更好的控制一个复杂表达式中指标的放置，amsmath 提供了两个额外的工具： \substack 命令和subarray 环境：

```
\begin{displaymath}
\sum_{\substack{0<i<n \\ 1<j<m}}
P(i,j) =\sum_{\begin{subarray}{l}
i\in I\\1<j<m
\end{subarray}} Q(i,j)
\end{displaymath}
```

TEX提供了各种各样的符号来得到括号(braces) 和其他定界符(delimiters)(如： $[\ h\ k\ l\ )$。圆括号和方括号可以由对应的键直接输入而花括号要用\{， 但是所有其它的定界符都要用一定的命令(如：\updownarrow) 生成。所有可用定界符的列表， 请查阅表3.7。

```
\begin{displaymath}
{a,b,c}\neq\{a,b,c\}
\end{displaymath}
```

如果你在某个左定界符前放一个\left 命令或是在某个右定界符前放一个\right 命令，TEX 将会自动决定这对定界符的大小。请注意，你必须为每个\left 命令配对相应的\right 命令,而且只有在左右定界符被排版在同一行时才会获得正确的大小尺寸。如果你不想使用任何右定界符，使用看不见的‘\right.’ 即可！

```
\begin{displaymath}
1 + \left( \frac{1}{ 1-x^{2} }
\right) ^3
\end{displaymath}
```

有些情况下，有必要手工指定一个数学定界符的正确尺寸，这可以使用\big，\Big，\bigg 和\Bigg 命令，大多数情况下你只需把它们放在定界符命令的前面。

```
$\Big( (x+1) (x-1) \Big) ^{2}$\\
$\big(\Big(\bigg(\Bigg($\quad
$\big\}\Big\}\bigg\}\Bigg\}$
\quad
$\big\|\Big\|\bigg\|\Bigg\|$
```

有很多命令可以实现在公式中插入三点列(three dots)。\ldots 得到在基线上的点列而\cdots 是上下居中的点列。另外，还有\vdots 命令产生竖直的点列，\ddots 产生对角线的点列。你可以在第3.5 节找到另外一个例子。

```
\begin{displaymath}
x_{1},\ldots,x_{n} \qquad
x_{1}+\cdots+x_{n}
\end{displaymath}
```

3.4 数学空格

如果公式内由TEX 选择的空格不令人满意，那么也可以通过插入一些特殊的空格控制命令来调整。有一些命令可以产生小空格：

\, 得到3/18 quad ( )

\: 得到4/18 quad ( )

\; 得到5/18 quad ( )。

转义的空格符\ 产生一个中等大小的空格，而\quad ( ) 和\qquad ( ) 产生大的空格。\quad 的大小与当前字体中字母‘M’ 的宽度有关。

\! 命令会产生一个-3/18 quad ( ) 的负空格。

```
\newcommand{\ud}{\mathrm{d}}
\begin{displaymath}
\int\!\!\!\int_{D} g(x,y)
\, \ud x\, \ud y
\end{displaymath}
instead of
\begin{displaymath}
\int\int_{D} g(x,y)\ud x \ud y
\end{displaymath}
```

请注意这里微分中的‘d’ 按惯例要设定成罗马字体。*AMS*-LATEX 为多重积分号之间空格的微调提供了另一种方法，即使用\iint,\iiint, \iiiint, 和\idotsint 命令。

加入amsmath 宏包后，上面的例子可以写成这样：

```
\newcommand{\ud}{\mathrm{d}}
\begin{displaymath}
\iint_{D} \, \ud x \, \ud y
\end{displaymath}
```

3.5 垂直取齐

要排版数组，使用array 环境。它的使用与tabular 环境有些类似。\\ 命令可用来断行。

```
\begin{displaymath}
\mathbf{X} =
\left( \begin{array}{ccc}
x_{11} & x_{12} & \ldots \\
x_{21} & x_{22} & \ldots \\
\vdots & \vdots & \ddots
\end{array} \right)
\end{displaymath}
```

$$\mathbf{X} = \left(\begin{array}{ccc} x_{11} & x_{12} & \ldots \\ x_{21} & x_{22} & \ldots \\ \vdots & \vdots & \ddots \end{array} \right)$$

array 环境也可以用来排版这样的表达式，表达式中使用一个“.”作为其隐藏的\right 定界符。

```
\begin{displaymath}
y = \left\{ \begin{array}{ll}
a & \textrm{if $d>c$}\\
b+x & \textrm{in the morning}\\
l & \textrm{all day long}
\end{array} \right.
\end{displaymath}
```

$$y = \begin{cases} a & \text{if } d > c \\ b + x & \text{in the morning} \\ l & \text{all day long} \end{cases}$$

就像在tabular 环境中一样，你也可以在array 环境中画线，如分隔矩阵中元素：

```
\begin{displaymath}
\left(\begin{array}{c|c}
1 & 2 \\
\hline
3 & 4
\end{array}\right)
\end{displaymath}
```

$$\left(\begin{array}{c|c} 1 & 2 \\ \hline 3 & 4 \end{array}\right)$$

对于跨行的长公式或是方程组(equation system)，你可以使用eqnarray 和eqnarray* 环境来替代equation 环境。在eqnarray 环境中每一行都有一个等式编号。eqnarray* 则不添加编号。

eqnarray 和eqnarray* 环境的用法与一个{rcl} 形式的3 列表格相类似，这里中间一列可以用来放等号，不等号， 或者是其他你选择的符号。\\ 命令可以断行。

```
\begin{eqnarray}
f(x) & = & \cos x \\
f'(x) & = & -\sin x \\
\int_{0}^{x} f(y)dy &
= & \sin x
\end{eqnarray}
```

注意，这里等号两边空白都有些大。\setlength\arraycolsep{2pt} 可以调小它。长等式不能被分成合适的小段。作者必须指定在哪里断且如何缩进。以下两种方法是最常用的。

```
{\setlength\arraycolsep{2pt}
\begin{eqnarray}
\sin x & = & x -\frac{x^{3}}{3!}
+\frac{x^{5}}{5!}-{}\nonumber\\
&& {}-\frac{x^{7}}{7!}+{}\cdots
\end{eqnarray}}
```

```
\begin{eqnarray}
\lefteqn{ \cos x = 1
-\frac{x^{2}}{2!} +{} }
\nonumber\\
& & {}+\frac{x^{4}}{4!}
-\frac{x^{6}}{6!}+{}\cdots
    \end{eqnarray}
```

\nonumber 命令告诉LATEX 不要给这个等式编号。

3.6 虚位

我们看不见虚位（phantom，也有幻影的意思），但是在许多人的头脑中它们依然占有一定的位置。LATEX 中也一样。我们可以使用它来实现一些有趣的小技巧。

当使用^ 和_ 时,LATEX 对文本的垂直对齐有时显得太过于自作多情。使用\phantom 命令你可以给不在最终输出中显示的字符保留位置。理解此意的最好方法是看下面的例子。

```
\begin{displaymath}
{}^{12}_{\phantom{1}6}\textrm{C}
\qquad \textrm{versus} \qquad
{}^{12}_{6}\textrm{C}
\end{displaymath}
```

```
\begin{displaymath}
\Gamma_{ij}^{\phantom{ij}k}
\qquad \textrm{versus} \qquad
\Gamma_{ij}^{k}
    \end{displaymath}
```

3.7 数学字体尺寸

在数学模式中，TEX 根据上下文选择字体大小。例如，上标会排版成较小的字体。如果你想要把等式的一部分排版成罗马字体，不要用\textrm 命令，只因\textrm 会暂时切换到文本模式， 而此时字体大小切换机制将不起作用。使用\mathrm 来保持字体大小切换机制的正常。但是要小心，\mathrm 只对较短的项有效。空格依然无效而且重音符号也不起作用。

```
\begin{equation}
2^{\textrm{nd}} \quad
2^{\mathrm{nd}}
\end{equation}
```

有时你仍需告诉LATEX 正确的字体大小。在数学模式中，可用以下四个命令来设定：
\displaystyle (123), \textstyle (123), \scriptstyle (123) and \scriptscriptstyle (123).
改变样式也会影响到上下限的显示方式。

```
\begin{displaymath}
\frac{\displaystyle
\sum_{i=1}^n(x_i-\overline x)
(y_i-\overline y)}
{\displaystyle\biggl[
\sum_{i=1}^n(x_i-\overline x)^2
\sum_{i=1}^n(y_i-\overline y)^2
\biggr]^{1/2}}
\end{displaymath}
```

这个例子中的括号要比\left[\right] 提供的括号更大些。\biggl 和\biggr命令分别对应于左和右括号。

3.8 定理、定律

当写数学文档时， 你可能需要一种方法来排版“引理”、“定义”、“公理”及其他类似的结构。

\newtheorem{name}[counter]{text}[section]
参量name 是用来标识“定理”的短关键字。而参数text 才是真正的“定

理”名，它会在最终的文档中被打印出来。方括号中是可选参量。两者都均用来指定“定理”的编号问题。使用counter参数来指定先前声明的“定理”的name。则此新的“定理”将与先前定理统一编号。section 参数让你来指定章节单元，而“定理”会按相应的章节层次来编号。

在你的文档的导言区执行\newtheorem 命令后，你就可以在文档中使用以下命令了。

```
\begin{name}[text]
This is my interesting theorem
\end{name}
```

amsthm 宏包提供了\newtheoremstyle{style} 命令,通过从三个预定义样式中选择其一来定义定理的外观，三个样式分别为：definition （标题粗体，内容罗马体）， plain （标题粗体，内容斜体）和remark （标题斜体，内容罗马体）。

理论上已经说够多了，下面我们联系一下实践，这个例子希望能够带走你的疑问并让你知道\newtheorem 环境其实比较复杂且不易理解。
首先定义定理环境:

```
\theoremstyle{definition} \newtheorem{law}{Law}
\theoremstyle{plain} \newtheorem{jury}[law]{Jury}
\theoremstyle{remark} \newtheorem*{marg}{Margaret}
\begin{law} \label{law:box}
Don't hide in the witness box
\end{law}
\begin{jury}[The Twelve]
It could be you! So beware and
see law \ref{law:box}\end{jury}
\begin{marg}No, No, No\end{marg}
```

“Jury” 定理与“Law” 定理共用了同一个计数器，因此它的编号与其他“Law” 定理的编号是顺序下来的。方括号中的参量用来指定定理的一个标题或是其他类似的内容。

```
\flushleft
\newtheorem{mur}{Murphy}[section]
\begin{mur}
If there are two or more
ways to do something, and
one of those ways can result
in a catastrophe, then
someone will do it.\end{mur}
```

“Murphy” 定理有一个与当前章节相联系的编号。你也可以使用其他的单元，如章(chapter) 或小节(subsection)。
amsthm 还提供了一个proof 环境。

```
\begin{proof}
Trivial, use
\[E=mc^2\]
\end{proof}
```

使用\qedhere 命令你可以移动“证毕”符。“证毕”符默认是在证明结束时单独放于一行。

```
\begin{proof}
Trivial, use \[E=mc^2 \qedhere\]
\end{proof}
```

3.9 粗体符号

在LATEX 中要得到粗体符号相当的不容易；这也许是故意设置的，以防业余水平的排版者过度的使用它们。字体变换命令\mathbf 可得到粗体字母，但是得到的是罗马体（直立的）而数学符号通常要求是斜体。还有一个\boldmath 命令，但是它只能用在数学模式之外。它不仅作用于字母也作用于符号。

```
\begin{displaymath}
\mu, M \qquad \mathbf{M} \qquad
\mbox{\boldmath $\mu, M$}
\end{displaymath}
```

请注意，逗号也成粗体了，这也许不是所需的。使用amsbsy 宏包（包含在amsmath 中）或tool 宏包集中的bm 将会便利许多，因为它们包含一个叫\boldsymbol 的命令。

```
\begin{displaymath}
\mu, M \qquad
\boldsymbol{\mu}, \boldsymbol{M}
\end{displaymath}
```

3.10 数学符号表

以下表格列出了数学模式中的所有常用符号。
要使用表3.11－3.15。必须在导言区先载入amssymb 宏包而且系统中安装了AMS 数学字体。
表3.1 － 数学模式重音符号。

$\hat{a}$	\hat{a}	$\check{a}$	\check{a}	$\tilde{a}$	\tilde{a}
$\grave{a}$	\grave{a}	$\dot{a}$	\dot{a}	$\ddot{a}$	\ddot{a}
$\bar{a}$	\bar{a}	$\vec{a}$	\vec{a}	$\widehat{A}$	\widehat{A}
$\acute{a}$	\acute{a}	$\breve{a}$	\breve{a}	$\widetilde{A}$	\widetilde{A}

表3.2 － 希腊字母。

α	\alpha	θ	\theta	o	o	υ	\upsilon
β	\beta	ϑ	\vartheta	π	\pi	ϕ	\phi
γ	\gamma	ι	\iota	ϖ	\varpi	φ	\varphi
δ	\delta	κ	\kappa	ρ	\rho	χ	\chi
ε	\epsilon	λ	\lambda	ϱ	\varrho	ψ	\psi
ε	\varepsilon	μ	\mu	σ	\sigma	ω	\omega
ζ	\zeta	ν	\nu	ς	\varsigma		
η	\eta	ξ	\xi	τ	\tau		
Γ	\Gamma	Λ	\Lambda	Σ	\Sigma	Ψ	\Psi
Δ	\Delta	Ξ	\Xi	Υ	\Upsilon	Ω	\Omega
Θ	\Theta	Π	\Pi	Φ	\Phi		

表 3.3 – 二元关系。

你可以在下列符号的相应命令前加上 `\not` 命令，而得到其否定形式。

$<$	`<`	$>$	`>`	$=$	`=`
$\leq$	`\leq` or `\le`	$\geq$	`\geq` or `\ge`	$\equiv$	`\equiv`
$\ll$	`\ll`	$\gg$	`\gg`	$\doteq$	`\doteq`
$\prec$	`\prec`	$\succ$	`\succ`	$\sim$	`\sim`
$\preceq$	`\preceq`	$\succeq$	`\succeq`	$\simeq$	`\simeq`
$\subset$	`\subset`	$\supset$	`\supset`	$\approx$	`\approx`
$\subseteq$	`\subseteq`	$\supseteq$	`\supseteq`	$\cong$	`\cong`
$\sqsubset$ [a]	`\sqsubset` [a]	$\sqsupset$ [a]	`\sqsupset` [a]	$\bowtie$ [a]	`\Join` [a]
$\sqsubseteq$	`\sqsubseteq`	$\sqsupseteq$	`\sqsupseteq`	$\bowtie$	`\bowtie`
$\in$	`\in`	$\ni$	`\ni` , `\owns`	$\propto$	`\propto`
$\vdash$	`\vdash`	$\dashv$	`\dashv`	$\models$	`\models`
$\mid$	`\mid`	$\parallel$	`\parallel`	$\perp$	`\perp`
$\smile$	`\smile`	$\frown$	`\frown`	$\asymp$	`\asymp`
$:$	`:`	$\notin$	`\notin`	$\neq$	`\neq` or `\ne`

[a] 使用 `latexsym` 宏包才能得到这个符号

表 3.4 – 二元运算符。

$+$	`+`	$-$	`-`		
$\pm$	`\pm`	$\mp$	`\mp`	$\triangleleft$	`\triangleleft`
$\cdot$	`\cdot`	$\div$	`\div`	$\triangleright$	`\triangleright`
$\times$	`\times`	$\setminus$	`\setminus`	$\star$	`\star`
$\cup$	`\cup`	$\cap$	`\cap`	$*$	`\ast`
$\sqcup$	`\sqcup`	$\sqcap$	`\sqcap`	$\circ$	`\circ`
$\vee$	`\vee` , `\lor`	$\wedge$	`\wedge` , `\land`	$\bullet$	`\bullet`
$\oplus$	`\oplus`	$\ominus$	`\ominus`	$\diamond$	`\diamond`
$\odot$	`\odot`	$\oslash$	`\oslash`	$\uplus$	`\uplus`
$\otimes$	`\otimes`	$\bigcirc$	`\bigcirc`	$\amalg$	`\amalg`
$\bigtriangleup$	`\bigtriangleup`	$\bigtriangledown$	`\bigtriangledown`	$\dagger$	`\dagger`
$\lhd$ [a]	`\lhd` [a]	$\rhd$ [a]	`\rhd` [a]	$\ddagger$	`\ddagger`
$\unlhd$ [a]	`\unlhd` [a]	$\unrhd$ [a]	`\unrhd` [a]	$\wr$	`\wr`

表 3.5 – “大”运算符。

$\sum$	`\sum`	$\bigcup$	`\bigcup`	$\bigvee$	`\bigvee`
$\prod$	`\prod`	$\bigcap$	`\bigcap`	$\bigwedge$	`\bigwedge`
$\coprod$	`\coprod`	$\bigsqcup$	`\bigsqcup`	$\biguplus$	`\biguplus`
$\int$	`\int`	$\oint$	`\oint`	$\bigodot$	`\bigodot`
$\bigoplus$	`\bigoplus`	$\bigotimes$	`\bigotimes`		

表 3.6 – 箭头。

$\leftarrow$	\leftarrow or \gets	$\longleftarrow$	\longleftarrow
$\rightarrow$	\rightarrow or \to	$\longrightarrow$	\longrightarrow
$\leftrightarrow$	\leftrightarrow	$\longleftrightarrow$	\longleftrightarrow
$\Leftarrow$	\Leftarrow	$\Longleftarrow$	\Longleftarrow
$\Rightarrow$	\Rightarrow	$\Longrightarrow$	\Longrightarrow
$\Leftrightarrow$	\Leftrightarrow	$\Longleftrightarrow$	\Longleftrightarrow
$\mapsto$	\mapsto	$\longmapsto$	\longmapsto
$\hookleftarrow$	\hookleftarrow	$\hookrightarrow$	\hookrightarrow
$\leftharpoonup$	\leftharpoonup	$\rightharpoonup$	\rightharpoonup
$\leftharpoondown$	\leftharpoondown	$\rightharpoondown$	\rightharpoondown
$\rightleftharpoons$	\rightleftharpoons	$\iff$	\iff (bigger spaces)
$\uparrow$	\uparrow	$\downarrow$	\downarrow
$\updownarrow$	\updownarrow	$\Uparrow$	\Uparrow
$\Downarrow$	\Downarrow	$\Updownarrow$	\Updownarrow
$\nearrow$	\nearrow	$\searrow$	\searrow
$\swarrow$	\swarrow	$\nwarrow$	\nwarrow
$\leadsto$ [a]	\leadsto		

[a] 使用 latexsym 宏包才能得到这个符号

表 3.7 – 定界符。

(	(	)	)	$\uparrow$	\uparrow
[	[or \lbrack	]	] or \rbrack	$\downarrow$	\downarrow
{	\{ or \lbrace	}	\} or \rbrace	$\updownarrow$	\updownarrow
$\langle$	\langle	$\rangle$	\rangle	$\vert$	| or \vert
$\lfloor$	\lfloor	$\rfloor$	\rfloor	$\lceil$	\lceil
/	/	\	\backslash	$\Updownarrow$	\Updownarrow
$\Uparrow$	\Uparrow	$\Downarrow$	\Downarrow	$\Vert$	\\| or \Vert
$\rceil$	\rceil				

表 3.8 – 大定界符。

$\lgroup$	\lgroup	$\rgroup$	\rgroup	$\lmoustache$	\lmoustache
$\arrowvert$	\arrowvert	$\Arrowvert$	\Arrowvert	$\bracevert$	\bracevert
$\rmoustache$	\rmoustache				

表 3.9 – 其他符号。

…	\dots	⋯	\cdots	⋮	\vdots	⋱	\ddots
$\hbar$	\hbar	$\imath$	\imath	$\jmath$	\jmath	ℓ	\ell
$\Re$	\Re	$\Im$	\Im	$\aleph$	\aleph	$\wp$	\wp
$\forall$	\forall	$\exists$	\exists	$\mho$ [a]	\mho [a]	∂	\partial
$'$	'	$\prime$	\prime	$\emptyset$	\emptyset	∞	\infty
∇	\nabla	$\triangle$	\triangle	$\Box$ [a]	\Box [a]	$\Diamond$ [a]	\Diamond [a]
$\bot$	\bot	$\top$	\top	$\angle$	\angle	$\surd$	\surd
$\diamondsuit$	\diamondsuit	$\heartsuit$	\heartsuit	$\clubsuit$	\clubsuit	$\spadesuit$	\spadesuit
$\neg$	\neg or \lnot	$\flat$	\flat	$\natural$	\natural	$\sharp$	\sharp

[a] 使用 latexsym 宏包才能得到这个符号

表 3.10 – 非数学符号。

也可以在文本模式中使用这些符号。

†	\dag	§	\S	©	\copyright	®	\textregistered
‡	\ddag	¶	\P	£	\pounds	%	\%

表 3.11 – AMS 定界符。

⌜	\ulcorner	⌝	\urcorner	⌞	\llcorner	⌟	\lrcorner
|	\lvert	|	\rvert	‖	\lVert	‖	\rVert

表 3.12 – AMS 希腊和希伯来字母。

\digamma	$\digamma$	\varkappa	$\varkappa$	\beth	ℶ	\gimel	ℷ	\daleth

表 3.13 – AMS 二元关系。

$\lessdot$	\lessdot	$\gtrdot$	\gtrdot	$\doteqdot$	\doteqdot
$\leqslant$	\leqslant	$\geqslant$	\geqslant	$\risingdotseq$	\risingdotseq
$\eqslantless$	\eqslantless	$\eqslantgtr$	\eqslantgtr	$\fallingdotseq$	\fallingdotseq
$\leqq$	\leqq	$\geqq$	\geqq	$\eqcirc$	\eqcirc
$\lll$	\lll or \llless	$\ggg$	\ggg	$\circeq$	\circeq
$\lesssim$	\lesssim	$\gtrsim$	\gtrsim	$\triangleq$	\triangleq
$\lessapprox$	\lessapprox	$\gtrapprox$	\gtrapprox	$\bumpeq$	\bumpeq
$\lessgtr$	\lessgtr	$\gtrless$	\gtrless	$\Bumpeq$	\Bumpeq
$\lesseqgtr$	\lesseqgtr	$\gtreqless$	\gtreqless	$\thicksim$	\thicksim
$\lesseqqgtr$	\lesseqqgtr	$\gtreqqless$	\gtreqqless	$\thickapprox$	\thickapprox
$\preccurlyeq$	\preccurlyeq	$\succcurlyeq$	\succcurlyeq	$\approxeq$	\approxeq
$\curlyeqprec$	\curlyeqprec	$\curlyeqsucc$	\curlyeqsucc	$\backsim$	\backsim
$\precsim$	\precsim	$\succsim$	\succsim	$\backsimeq$	\backsimeq
$\precapprox$	\precapprox	$\succapprox$	\succapprox	$\vDash$	\vDash
$\subseteqq$	\subseteqq	$\supseteqq$	\supseteqq	$\Vdash$	\Vdash
$\shortparallel$	\shortparallel	$\Supset$	\Supset	$\Vvdash$	\Vvdash
$\blacktriangleleft$	\blacktriangleleft	$\sqsupset$	\sqsupset	$\backepsilon$	\backepsilon
$\vartriangleright$	\vartriangleright	$\because$	\because	$\varpropto$	\varpropto
$\blacktriangleright$	\blacktriangleright	$\Subset$	\Subset	$\between$	\between
$\trianglerighteq$	\trianglerighteq	$\smallfrown$	\smallfrown	$\pitchfork$	\pitchfork
$\vartriangleleft$	\vartriangleleft	$\shortmid$	\shortmid	$\smallsmile$	\smallsmile
$\trianglelefteq$	\trianglelefteq	$\therefore$	\therefore	$\sqsubset$	\sqsubset

表 3.14 – AMS 箭头。

$\dashleftarrow$	\dashleftarrow	$\dashrightarrow$	\dashrightarrow
$\leftleftarrows$	\leftleftarrows	$\rightrightarrows$	\rightrightarrows
$\leftrightarrows$	\leftrightarrows	$\rightleftarrows$	\rightleftarrows
$\Lleftarrow$	\Lleftarrow	$\Rrightarrow$	\Rrightarrow
$\twoheadleftarrow$	\twoheadleftarrow	$\twoheadrightarrow$	\twoheadrightarrow
$\leftarrowtail$	\leftarrowtail	$\rightarrowtail$	\rightarrowtail
$\leftrightharpoons$	\leftrightharpoons	$\rightleftharpoons$	\rightleftharpoons
$\Lsh$	\Lsh	$\Rsh$	\Rsh
$\looparrowleft$	\looparrowleft	$\looparrowright$	\looparrowright
$\curvearrowleft$	\curvearrowleft	$\curvearrowright$	\curvearrowright
$\circlearrowleft$	\circlearrowleft	$\circlearrowright$	\circlearrowright
$\multimap$	\multimap	$\upuparrows$	\upuparrows
$\downdownarrows$	\downdownarrows	$\upharpoonleft$	\upharpoonleft
$\upharpoonright$	\upharpoonright	$\downharpoonright$	\downharpoonright
$\rightsquigarrow$	\rightsquigarrow	$\leftrightsquigarrow$	\leftrightsquigarrow

表 3.15 – AMS 二元否定关系符和箭头。

$\not<$	\nless	$\not>$	\ngtr	$\subsetneqq$	\varsubsetneqq
$\lneq$	\lneq	$\gneq$	\gneq	$\supsetneqq$	\varsupsetneqq
$\not\leq$	\nleq	$\not\geq$	\ngeq	$\not\subseteqq$	\nsubseteqq
$\not\leqslant$	\nleqslant	$\not\geqslant$	\ngeqslant	$\not\supseteqq$	\nsupseteqq
$\lneqq$	\lneqq	$\gneqq$	\gneqq	$\nmid$	\nmid
$\lvertneqq$	\lvertneqq	$\gvertneqq$	\gvertneqq	$\nparallel$	\nparallel
$\not\leqq$	\nleqq	$\not\geqq$	\ngeqq	$\nshortmid$	\nshortmid
$\lnsim$	\lnsim	$\gnsim$	\gnsim	$\nshortparallel$	\nshortparallel
$\lnapprox$	\lnapprox	$\gnapprox$	\gnapprox	$\not\sim$	\nsim
$\not\prec$	\nprec	$\not\succ$	\nsucc	$\not\cong$	\ncong
$\not\preceq$	\npreceq	$\not\succeq$	\nsucceq	$\nvdash$	\nvdash
$\precneqq$	\precneqq	$\succneqq$	\succneqq	$\nvDash$	\nvDash
$\precnsim$	\precnsim	$\succnsim$	\succnsim	$\nVdash$	\nVdash
$\precnapprox$	\precnapprox	$\succnapprox$	\succnapprox	$\nVDash$	\nVDash
$\not\subset$	\subsetneq	$\not\supset$	\supsetneq	$\ntriangleleft$	\ntriangleleft
$\varsubsetneq$	\varsubsetneq	$\varsupsetneq$	\varsupsetneq	$\ntriangleright$	\ntriangleright
$\not\subseteq$	\nsubseteq	$\not\supseteq$	\nsupseteq	$\ntrianglelefteq$	\ntriangleleftteq
$\subsetneqq$	\subsetneqq	$\supsetneqq$	\supsetneqq	$\ntrianglerighteq$	\ntrianglerighteq
$\nleftarrow$	\nleftarrow	$\nrightarrow$	\nrightarrow	$\nleftrightarrow$	\nleftrightarrow
$\nLeftarrow$	\nLeftarrow	$\nRightarrow$	\nRightarrow	$\nLeftrightarrow$	\nLeftrightarrow

表 3.16 – AMS 二元运算符。

$\dotplus$	\dotplus	$\centerdot$	\centerdot		
$\ltimes$	\ltimes	$\rtimes$	\rtimes	$\divideontimes$	\divideontimes
$\Cup$	\doublecup	$\Cap$	\doublecap	$\smallsetminus$	\smallsetminus
$\veebar$	\veebar	$\barwedge$	\barwedge	$\doublebarwedge$	\doublebarwedge
$\boxplus$	\boxplus	$\boxminus$	\boxminus	$\circleddash$	\circleddash
$\boxtimes$	\boxtimes	$\boxdot$	\boxdot	$\circledcirc$	\circledcirc
$\intercal$	\intercal	$\circledast$	\circledast	$\rightthreetimes$	\rightthreetimes
$\curlyvee$	\curlyvee	$\curlywedge$	\curlywedge	$\leftthreetimes$	\leftthreetimes

表 3.17 – AMS 其他符号。

$\hbar$	\hbar	$\hslash$	\hslash	$\Bbbk$	\Bbbk
$\square$	\square	$\blacksquare$	\blacksquare	Ⓢ	\circledS
$\vartriangle$	\vartriangle	$\blacktriangle$	\blacktriangle	$\complement$	\complement
$\triangledown$	\triangledown	$\blacktriangledown$	\blacktriangledown	$\Game$	\Game
$\lozenge$	\lozenge	$\blacklozenge$	\blacklozenge	$\bigstar$	\bigstar
$\angle$	\angle	$\measuredangle$	\measuredangle		
$\diagup$	\diagup	$\diagdown$	\diagdown	$\backprime$	\backprime
$\nexists$	\nexists	$\Finv$	\Finv	$\varnothing$	\varnothing
$\eth$	\eth	$\sphericalangle$	\sphericalangle	$\mho$	\mho

表 3.18 – 数学字母。

实例	命令	所需宏包
ABCDEabcde1234	\mathrm{ABCDE abcde 1234}	
ABCDEabcde1234	\mathit{ABCDE abcde 1234}	
$ABCDEabcde1234$	\mathnormal{ABCDE abcde 1234}	
$\mathcal{ABCDE}$	\mathcal{ABCDE abcde 1234}	
$\mathscr{ABCDE}$	\mathscr{ABCDE abcde 1234}	mathrsfs
$\mathfrak{ABCDEabcde1234}$	\mathfrak{ABCDE abcde 1234}	amsfonts or amssymb
$\mathbb{ABCDE}\nexists\neq\nvDash\nvdash$	\mathbb{ABCDE abcde 1234}	amsfonts or amssymb

第四章 专业功能

当你整理一个大型文档时， LATEX 的一些专门功能，例如自动生成索引、管理参考文献等等，会给你以很大的帮助。详细的关于LATEX 专业功能以及增强功能的描述可以在LATEX Manual 和The LATEX Companion 找到。

4.1 插入 EPS 图形

LATEX 通过figure 和table 环境提供了处理图像图形等浮动体的基本工具。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(graphics)，第五章中将会介绍其中的几种方法。

在文档中使用图形，一个相对容易的办法就是使用专门的软件包生成图形文件，然后将最终的图形文件插入到文档中。LATEX 的宏包提供了许多方法来完成这个工作。在这个手册里，我们只讨论Encapsulated PostScript (EPS) 图形文件的使用，因为它比较简单而且被广泛地使用。为了使用EPS 格式的图片，你必须有一个PostScript打印机来输出结果。

由D. P. Carlisle 制作的graphicx 宏包提供了一套很好的插图命令。它是一个叫作“graphics” 的宏包集中的一部分。

假设你使用的系统安装了PostScript 打印机和graphicx 宏包，那么你就可以通过下面的步骤把一幅图片加入你的文档中：

1. 用你的图形软件输出EPS 格式的文件4。
2. 在源文件的导言中加上下面的命令来载入graphicx 宏包。

```
\usepackage[driver]{graphicx}
```

这里driver 是你使用的“dvi 到postscript” 的转换程序。最常用的是dvips。因为TEX 中没有规定插入图形的标准，所以driver 的名字是必需的。知道了driver 的名字，graphicx 宏包就可以选择合适的方法在.dvi 文件中插入关于图形的信息。然后打印机理解这些信息并能正确的插入.eps 文件。

3. 使用命令

```
\includegraphics[key=value, . . . ]{file}
```

来把file 加入你的文档。可选的参数是一系列由逗号隔开的keys 和相应的(values)。keys 可以用来改变插图的宽度、高度以及旋转。表4.1 列出

了最重要的关键词。

表4.1 - graphicx 宏包使用的关键词。

width	把图形调整到指定的宽度
height	把图形调整到指定的高度
angle	逆时针旋转图形
scale	缩放图形

下面的示例代码可以帮助我们理解整个过程：

```
\begin{figure}
\centering
\includegraphics[angle=90,
width=0.5\textwidth]{test}
\caption{This is a test.}
\end{figure}
```

这段代码把文件test.eps 中的图形插入到文档里。首先图形被旋转90度，然后进行缩放使得图形的宽度等于标准段落宽度的0.5 倍。因为没有指定图形的高度,图形的高宽变化的比例是1.0,也就是保持原来的高宽比。高度和宽度参数也可以指定为绝对长度单位。详细的信息可以在表6.5 中找到。

4.2 参考文献

你可以通过thebibliography 环境来产生一个参考文献(bibliography)。每个参考文献的条目以如下的命令开头

```
\bibitem[label]{marker}
```

然后使用marker 在正文中引用这本书、这篇文章或者论文。

```
\cite{marker}
```

如果你不使用参数label，参考文献条目的编号是自动生成的。\begin{thebibliography} 命令后的参数设定了最大的编号宽度。在下面的例子中，{99} 告诉LATEX 参考文献条目的编号不会比数字99 更宽。

```
Partl \cite{pa} has
proposed that \ldots
\begin{thebibliography}{99}
\bibitem{pa} H. Partl:
\emph{German \TeX},
TUGboat Volume 9, Issue 1
(1988)
\end{thebibliography}
```

4.3 索引

许多书籍的一个非常有用的部分就是它们的索引(index) 了。使用LATEX 和辅助工具makeindex，我们能够很容易的生成索引。

为了使用LATEX 的索引功能，宏包makeidx 必须在导言部分被载入：
\usepackage{makeidx}
然后在导言中使用\makeindex激活索引命令。
索引的内容通过命令\index{key}指定，这里key 是索引项的关键词。你可以在需要被索引的地方加入这条命令。表4.2 举例解释了参量key 语法。

表 4.2 – 索引关键词语法示例。

示例	索引项	注释
\index{hello}	hello, 1	普通格式的索引项
\index{hello!Peter}	Peter, 3	‘hello’ 下的子项
\index{Sam@\textsl{Sam}}	Sam, 2	格式化的索引项
\index{Lin@\textbf{Lin}}	**Lin**, 7	同上
\index{Jenny|textbf}	Jenny, **3**	格式化的页码
\index{Joe|textit}	Joe, 5	同上
\index{ecole@\'ecole}	école, 4	重音标记

当LATEX 处理源文档时，每个\index 命令都会将适当的索引项和当前页码写入一个专门的文件中。这个文件的名字和LATEX 源文档相同，但具有不同的扩展名后缀(.idx)。这个.idx 文件需要用makeindex 程序来处理。

makeindex filename

makeindex 程序生成一个与源文件同名的序列化索引文件，这个文件使用.ind 为扩展名。当再次用LATEX 处理源文件时， 这个序列化索引文

件将被包含到源文件中\printindex命令出现的位置。

LATEX 2 ε 附带的宏包showidx 可以在正文的左边打印出索引项。这个功能在校对文档和索引项时十分有用。

请注意不谨慎地使用\index 命令，将会影响文档页版面布局。

```
My Word \index{Word}. As opposed
to Word\index{Word}. Note the
    position of the full stop.
```

4.4 定制页眉和页脚

Piet van Oostrum 编写的fancyhdr 宏包6， 提供了一些简单的命令使得我们可以定制文档的页眉和页脚。看一眼本页的顶部， 你就能发现这个宏包的用处。

定制页眉和页脚时一件棘手的事情就是得到每个页面所属的章节名称。LATEX通过两个步骤来完成这个任务。在定义页眉和页脚时，你可以使用\rightmark命令来代表当前的section 名，使用\leftmark 来代表当前的chapter 名。这两个命令的值将在处理chapter 或者section 命令时被重新赋值。

为了获得最大的灵活性， \chapter 等命令并不对\rightmark 和\leftmark直接进行重定义，而是间接地通过调用\chaptermark、\sectionmark 或者\subsectionmark 来重新定义\rightmark 和\leftmark。因此，只需要重新定义\chaptermark 命令， 就能修改页眉上显示的章名。图4.1 显示了如何配置fancyhdr 来得到和本文相似的页眉。

```
\documentclass{book}
\usepackage{fancyhdr}
\pagestyle{fancy}
with this we ensure that the chapter and section
headings are in lowercase.
\renewcommand{\chaptermark}[1]{%
\markboth{#1}{}}
\renewcommand{\sectionmark}[1]{%
\markright{\thesection\ #1}}
\fancyhf{} % delete current header and footer
\fancyhead[LE,RO]{\bfseries\thepage}
\fancyhead[LO]{\bfseries\rightmark}
\fancyhead[RE]{\bfseries\leftmark}
\renewcommand{\headrulewidth}{0.5pt}
\renewcommand{\footrulewidth}{0pt}
\addtolength{\headheight}{0.5pt} % space for the rule
\fancypagestyle{plain}{%
\fancyhead{} % get rid of headers on plain pages
\renewcommand{\headrulewidth}{0pt} % and the line
}
```

图4.1 - fancyhdr 设置实例。

4.5 Verbatim 宏包

在本文的前面部分你已经知道了verbatim 环境。在这一节中， 你将学会使用verbatim 宏包。verbatim 宏包重新实现了verbatim 环境，并解决了原来的verbatim 环境的一些限制。这本身并没有什么特别的，但 verbatim 宏包还实现了一些新增的功能， 这才是我在这里提到这个宏包的原因。verbatim 宏包提供了

\verbatiminput{filename}

命令，这个命令允许你把一个ASCII 码的文本文件包含到你的文档中来，就好像它们是在verbatim 环境中一样。

verbatim 宏包是‘tools’ 宏包集的一部分，大多数的系统中都预装了这个宏包集。

4.6 安装额外的宏包

大多数的LATEX 安装都带有大量预装的样式宏包，但更多的宏包可以在网上得到。在互联网寻找样式宏包的一个主要的地方就是CTAN (http://www.ctan.org/)。各种宏包的源文件，例如geometry， hyphenat 等等，一般来说都包含两个文件：一个扩展名为.ins，另一个扩展名为.dtx。此外，通常会有一个readme.txt 对宏包进行简要的说明。你应该先阅读这个文件。

无论如何，一旦你得到了宏包的源文件，你还要对它们进行处理使得 (a) 你的TEX 系统知道这个新的宏包， (b) 生成说明文档。下面是第一部分的步骤：

1. 对.ins 文件运行LATEX 命令。这将会产生一个.sty 文件。
2. 把.sty 文件移到LATEX 系统能找到的地方。一般是在../*localtexmf*/tex/latex子目录下（Windows 或者OS/2 用户应该改变斜线为反斜线）。
3. 刷新系统的文件名数据库。具体的命令取决于你使用的LATEX 系统： teTeX,fpTeX - texhash; web2c - maktexlsr; MikTeX - initexmf -updatefndb或者使用图形界面。

现在你可以从.dtx 文件生成说明文档：

1. 对.dtx 文件运行LATEX 命令。这会生成一个.dvi 文件。注意你可能需要多次运行LATEX 命令来正确处理交叉引用。
2. 检查一下LATEX 命令是否产生了.idx 文件。如果没发现这个文件，你就可以执行第5 步了。
3. 为了生成索引，敲入命令：

   makeindex -s gind.ist *name*（这里name 表示不带扩展名的主文件名）。
4. 再次对.dtx 文件运行LATEX 命令。
5. 最后一步但不是必需的，生成.ps 文件或者.pdf 文件以方便阅读。有时你会看见生成了一个.glo (glossary) 文件。在第4 步和第5 步之间运行下面的命令：

makeindex -s gglo.ist -o *name*.gls *name*.glo

确认在执行第5 步前最后对.dtx 文件运行一遍LATEX 命令。

4.7 使用 pdfLATEX

PDF 是一种超文本文档(hypertext) 格式。类似于网页，文档中的某

些词可以被超链接标记。它们链接到这个文档的另一个地方，甚至是另外一个文档。如果你点击这样一个超链接，你将转调到链接的目的地。这意味着在LATEX 格式的文档中所有\ref 和\pageref 出现的位置都变成超链接。此外，目录、索引和其它类似的结构变成超链接的集合。

现在大多数的网页都是用HTML 超文本标记语言编写。在写科技文档的时候，这种格式有两个严重的缺陷：

1. 数学公式在HTML 文档中通常都不被支持。尽管对此有一个标准，但是大多是现在使用的浏览器都不支持，或者缺少需要的字体。

2. 打印HTML 文档是比较容易的，但打印的结果会因系统平台和浏览器的不同而出现差异。这结果与我们在LATEX 世界中期待的高质量相差十万八千里。

有许多将LATEX 转为HTML 的尝试。其中一些可以说是相当成功的，它们能将一个标准的LATEX 源文件生成一个合格的网页。但是为了达到目的，需要去掉一些支持。当你开始使用LATEX 的复杂功能和扩展宏包时，那些尝试就行不通了。若希望即使是发不到网上也保留文档的高质量， 作者们就要使用PDF （便携式文档格式）， 它保留了文档的版式和允许超链接导航。现在大多数浏览器只要带上相应的插件都可以直接显示PDF 文档。

尽管几乎所有的操作系统都有DVI 和PS 格式的阅读器， 但你会发现人们更多地使用Acrobat Reader 和Xpdf 来阅读PDF 文档。因而提供PDF 格式的文档将使得潜在的读者更容易阅读。

4.7.1 发布到网上的 PDF 文档

有了H`an Th′ ˆe Th`anh 开发的pdf TEX 程序，使用LATEX源文件来创建PDF 文件将变得非常简单。一般的TeX 生成DVI，而pdf TEX 生成PDF。还有pdf LATEX 程序将LaTeX 源文件生成PDF。

现在大多数的TEX 发行版本都自动集成安装了pdf TEX 和pdf LATEX，例如：teTEX， fpTEX， MikTEX， TEXLive 和CMacTEX。

为了生成PDF 而不是DVI，只需要将命令latex file.tex 改成 pdflatex file.tex。在不是使用命令行的LATEX 系统中，你可以在TEX 控制中心找到一个专门的按钮。

在LATEX 中，你可以通过documentclass 的参数选项来定义页面的大小， 例如：a4paper 或letterpaper。这些也对pdf LATEX 有效，但是首先要让pdf TEX知道这种页面的实际大小来控制PDF 文件的页面大小。若你使用hyperref 宏包（参见第63 页），页面的大小将自动调整。否则，你需要手动的将下面两行加入到文档的导言区：

```
\pdfpagewidth=\paperwidth
```

\pdfpageheight=\paperheight
接下来的一节将深入介绍LATEX 和pdf LATEX 之间的不同。主要的不同有三个方面：采用的字体，包含图像的格式和超链接的手动配置。

4.7.2 字体

pdf LATEX 能处理所有的字体（PK 点阵、TrueType、PostScript type 1……），但是作为LATEX 通常的字体格式， PK 点阵字体在Acrobat Reader 下的显示效果非常差。为了获得文档更高的显示效果，最好使用PostScript Type 1 字体。高级的TEX 安装程序会自动设置好，你最好试一下，如果它正常运作，你就可以跳过这一节。PostScript Type 1 的Computer Modern 和AMSFonts 字体由Blue Sky Research 和Y&Y, Inc. 制作，后来版权属于美国数学学会(AMS)。这些字体在1997 年被开放，并且出现在大多数TEX 发行版中。

然而，如果你使用LATEX 来创建非英文的文档，你可能用到EC， LH 或CB 字体（关于OT1 字体的讨论可参见第20 页）。Vladimir Volovich 创建了cm-super字体包，包含全部EC/TC、EC Concrete、EC Bright 和LH 字体集。在CTAN:/fonts/ps-type1/cm-super 可以获得，也被集成进了TEXLive7 和MikTEX。由Apostolos Syropoulos 创建类似type 1 CB 的希腊字体可在CTAN:/tex-archive/fonts/greek/cb 获得。不幸的是，这些字体集跟Blue Sky/Y&Y 的Type 1 CM 字体的印刷质量不一样。LATEX 会自动提示，而且文档在屏幕的显示效果也不如Blue Sky/Y&Y type 1 CM 字体那么整洁。在高分辨率的输出设备下，它们生成的效果跟原来的EC/LH/CB 点阵字体一样。

如果你使用一种拉丁语系的语文来创建文档，你有其它一些选择。

- 使用aeguill 宏包，也叫Almost European Computer Modern with Guillemets。只需在导言区添加一行\usepackage{aeguill}，来启用AE 虚拟字体替代EC 字体。
- 或者使用mltex 宏包，但是它只在pdf TEX 设置了mltex 选项时有效。

AE 虚拟字体集，像MlTEX 系统，在CM 字体的字符基础上创建全部缺失的字母并按EC 顺序重新排列，就使得TEX 处理它的时候认为它有一个全部256 个字符的字体集。这样就可使用大部分系统中优质的type 1 格式的cm 字体。这套字体现在为T1 编码，在拉丁语系的欧洲语文中能很好的运作。唯一的不足就是创建的AE 字符不支持Acrobat Reader 的查找功能，因此你不能在PDF 文档中搜索那些带重音符号的单词。

对于俄文，一个类似的解决办法是使用C1 虚拟字体，这可以在ftp://ftp.vsu.ru/pub/tex/font-packs/c1fonts 上获得。这套字体集成了标准的Bluesky CM type 1 字体和Paradissa 与BaKoMa 的CMCYR type 1 字体，

这些都可以在CTAN 上找到。由于Paradissa 字体只包含俄文字母，C1 字体里就没有其他西里尔字符了。

另一种解决办法是使用其它PostScript type 1 字体。事实上，其中一些字体被包含在Acrobat Reader 的相应语言版本中。由于这些字体有不同的字符大小，页面上的文本格式将会改变。一般地，这类字体占得空间要比CM 字体大，因为CM 字体是一种非常节省空间的字体。而且文档可视的一致性也被破坏了，因为Times、Helvetica 和Courier 字体（这些是Acrobat 里基本的可替换字体）没有被设计来在单个文档中和平共处。

为了达到上述目的，有两套字体已经做好：pxfonts，它基于作为正文字体的Palatino；另外就是基于Times 的txfonts 宏包。只需要在文档的导言区加入下列几行就可以使用这些字体：

```
\usepackage[T1]{fontenc}
\usepackage{pxfonts}
```

注：当你编译的时候，在.log 文件中出现下面的信息：

Warning: pdftex (file eurmo10): Font eur... not found

这意味着文档使用的某些字体没有被找到。你必须解决这些问题，否则输出的PDF 文件可能不会显示缺失字符的页面。

上面讨论了如此多的字体问题，特别是缺乏与type 1 格式的CM 字体同样高质量的EC 字体一直困扰大家。最近一套新的被称为Latin Modern (LM) 的高质量字体已开发完成。这使得前面的担心都是多余的。如果你安装了最新的TEX， 你可能已经安装好这套字体，需要做的只是在你的文档的导言区添加

```
\usepackage{lmodern}
\usepackage[T1]{fontenc}
\usepackage{textcomp}
```

就可以创建支持所有拉丁字符的优质PDF 输出文件。

4.7.3 使用图形

通过graphicx 宏包，你可以在文档中插入图形。使用pdftex作为driver 的选项，这个宏包也可用于pdf LATEX：

```
\usepackage[pdftex]{color,graphicx}
```

上面这个例子中，我还包含了color 宏包，因此网页上彩色的文档会看起来更自然一些。

Encapsulated PostScript 格式的图形并不被Pdf LATEX 所支持。如果你不在命令\includegraphics 声明加载的文件的扩展名，宏包graphicx 将依赖于driver 选项的设置自动寻找一个合适的文件。对于pdftex，它支持的格式有.png，.pdf， .jpg 和.mps (MetaPost)，但不支持.eps。

一个解决这个问题的简单方法是通过在很多系统中可找到的epstopdf工具将你的EPS 文件转化为PDF 格式。对于矢量图（画），这时一个很好的解决办法，但对于位图（相片、扫描图）， 这并不是很理想，因为PDF 格式本来就支持包含PNG 和JPEG 图像。PNG 格式适合屏幕截图和其它一些只有较少颜色的图像。JPEG 是一种非常适合于相片的格式，而且占用空间少。

可能对于画一些特殊的几何图形这也不是很理想，幸好可以通过一些专门的命令语言来画图形，例如MetaPost， 它可以在大多数的TEX 发行版本中找到，并且也包含它的扩展手册。

4.7.4 超链接

hyperref 宏包将你的文档中的所有内部引用变成超链接。为此，一些特殊的设置是必须的，保证\usepackage[pdftex]{hyperref} 是你的文档导言区的最后一行命令。

有很多选项用于定义hyperref 宏包的效果：

- 或者在\usepackage[pdftex]{hyperref} 的pdftex 选项后用逗号隔开列出；
- 或者使用单独一行的命令\hypersetup{options}。

pdftex 是唯一必须的选项，其他参数用来改变hyperref 的默认效果7。下面的列表中默认值被写成upright 字体。

bookmarks (=true,*false*) 显示或隐藏书签栏

unicode (=false,*true*) 允许在Acrobat 书签栏使用非拉丁语系的字符

pdftoolbar (=true,*false*) 显示或隐藏Acrobat 的工具栏

pdfmenubar (=true,*false*) 显示或隐藏Acrobat 的菜单栏

pdffitwindow (=true,*false*) 调整显示PDF 的初始化放大倍率

pdftitle (=*{text}*) 定义Acrobat 显示的文档信息(Document Info)

pdfauthor (=*{text}*) PDF 文件作者名字

pdfnewwindow (=true,*false*) 定义当超链接到另一个文档时，是否打开一个新的窗口

colorlinks (=false,*true*) 链接有一个彩色框架(false) 或者链接文本的颜色设置(true)。链接的颜色通过下面的参数来控制（默认的颜色已列出）

linkcolor (=red) 内部链接的颜色(sections, pages, etc.),

citecolor (=green) 引用链接的颜色(bibliography)

filecolor (=magenta) 文件链接的颜色

urlcolor (=cyan) URL 链接的颜色(mail, web)

如果你觉得这些默认值合适，就直接使用

\usepackage[pdftex]{hyperref}

为了使书签列表打开和链接为彩色（=true 不需要写出来）：

```
\usepackage[pdftex,bookmarks,colorlinks]{hyperref}
```

当创建PDF 文档是用来打印的，使用彩色的链接并不是一件好事，因为彩色的链接将会被打印成灰色，从而难以阅读。你可以设置不打印彩色的框架：

```
\usepackage{hyperref}
\hypersetup{colorlinks=false}
```

或者将超链接变成黑色：

```
\usepackage{hyperref}
\hypersetup{colorlinks,%
            citecolor=black,%
        filecolor=black,%
        linkcolor=black,%
        urlcolor=black,%
        pdftex}
```

当你只想提供PDF 文件的文档信息(Document Info)：

```
    \usepackage[pdfauthor={Pierre Desproges},%
    pdftitle={Des femmes qui tombent},%
    pdftex]{hyperref}
```

除了自动的交叉引用的超链接，通过下面的命令可以嵌入明确的链接：

```
\href{url}{text}
```

代码

```
The \href{http://www.ctan.org}{CTAN} website.
```

生成的效果为“CTAN”；单击词“CTAN” 将把你带到CTAN 网站。

若链接的目的地不是一个URL ，而是一个本地文件，你可以使用\href 命令：

```
The complete document is \href{manual.pdf}{here}
```

生成的效果为“The complete document is **here**”。单击“here” 将打开文件manual.pdf （文件名跟依赖于当前文档的位置）。

若文章的作者希望读者可以容易地发邮件给她，只需要在文档的标题页的\author 命令中使用命令\href：

```
\author{Mary Oetiker $<$\href{mailto:mary@oetiker.ch}%
{mary@oetiker.ch}$>$
```

注意到这个链接不仅显示在链接中还显示在页面上。下面的链接

```
\href{mailto:mary@oetiker.ch}{Mary Oetiker}
```

在Acrobat 中可正常使用，但页面被打印时，邮箱地址将不可见。

4.7.5 链接的问题

当一个计数器被重新初始化后可能出现下面的信息：

```
! pdfTeX warning (ext4): destination with the same
identifier (name{page.1}) has been already used,
duplicate ignored
```

例如，在book 类的文档中使用命令\mainmatter 就可能出现上面的警告。它在书的第一章重设页码计数器为1，但是书的序言部分也有页码1，从而链接到“页码1”不再是唯一的，故出现了“duplicate has been ignored.”一个解决的办法是将plainpages=false 加入到hyperref 选项中。不幸的是这样只对页码计数器有效。或者冒险使用hypertexnames=false，但是 这会使得索引中的页码链接失效。

4.7.6 书签的问题

书签中的文本有时并不会像你想象中的那样显示,因为书签仅仅是“文本”，其中可使用的字符要比正常LATEX 文件的少得多。Hyperref 经常遇到这类问题而出现下面的警告：

```
Package hyperref Warning:
Token not allowed in a PDFDocEncoded string:
```

现在你可以通过为书签提供一个文本字符串来解决这个问题，用

```
\texorpdfstring{TEX text}{Bookmark Text}
```

来替换不正常显示的文本。

数学表达式也是这类问题的典型代表：

```
\section{\texorpdfstring{$E=mc^2$}%
{E=mc^2}}
```

而通常在书签中\section{$E=mc^2$} 显示为“E=mc2”。

颜色的改变也在书签显示中出问题：

```
\section{\textcolor{red}{Red !}}
```

显示的结果为“redRed!”。命令 \textcolor 被忽略。而它的参数(red)被输出。

如果你使用

```
\section{\texorpdfstring{\textcolor{red}{Red !}}{Red\ !}}
```

结果将更美观。

如果你写unicode （统一字符编码）类型的文档，就需添加宏包 hyperref

的选项unicode，这样你就可以在书签中使用unicode 字符。然后使用 \texorpdfstring 命令将让你有较大范围的字符供选择。

LATEX 和pdfLATEX 的源文件的兼容性

理想的话，你的文档用LATEX 和pdf LATEX 编译的效果应该一致， 这方面的问题主要是包含的图像。一个简单的解决办法是在命令\includegraphics 中不设置加载的图像的扩展名,它们会自动在当前目录寻找一个格式适合的文件，你需要做的是创建相应格式的图像文件。LATEX 会寻找.eps格式，而pdf LATEX 会尝试包含一个扩展名为.png，.pdf，.jpg 或.mps（按这个顺序）的文件。

若你想在PDF 格式的文件中使用不同的代码，只需要简单地在导言区加入宏包ifpdf。但前提是你已经安装了这个宏包，如果没有安装而你又使用MiKTEX 的话， 它将在你第一次使用的时候自动安装。这个宏包定义了一个特殊的命令\ifpdf，利用它你可很容易编写条件代码。例如，考虑到打印费用，用PostScript 格式仅使用黑白色，但在线查看的PDF 格式将是彩色的。

```
\RequirePackage{ifpdf} % running on pdfTeX?
\ifpdf
\documentclass[a4paper,12pt,pdftex]{book}
\else
\documentclass[a4paper,12pt,dvips]{book}
\fi
\ifpdf
\usepackage{lmodern}
\fi
\usepackage[bookmarks, % add hyperlinks
colorlinks,
plainpages=false]{hyperref}
\usepackage[T1]{fontenc}
\usepackage[latin1]{inputenc}
\usepackage[english]{babel}
\usepackage{graphicx}
...
```

在上面的例子中,在非PDF 格式中也包含了hyperref 宏包,这样\href 命令在所有情形下都有效，这也使得不用在每个情况下都使用条件声明。注意到当前的TEX 发行版本（例如TEXLive），通常的TEX 会根据文档类型的设置自动选择输出PDF 还是DVI。如果使用上面的代码，仍然可以使用pdflatex 命令来得到PDF 格式的输出或使用latex 得到DVI 格式。

4.8 创建演示文稿

你可以将你的科学工作成果通过黑板、透明片或者在你的笔记本电脑上直接使用演示文稿软件呈现。

pdf LATEX 和beamer 文档类允许你创建PDF 格式的演示文稿， 结果跟你用一天时间制作的PowerPoint 看上去差不多，但更便携因为Acrobat Reader 支持更多的系统平台。

beamer 文档类使用带参数的宏包graphicx、color 和hyperref 来适应屏幕阅读的演示文稿。

```
\documentclass[10pt]{beamer}
\mode<beamer>{%
\usetheme[hideothersubsections,
right,width=22mm]{Goettingen}
}
\title{Simple Presentation}
\author[D. Flipo]{Daniel Flipo}
\institute{U.S.T.L. \& GUTenberg}
\titlegraphic{\includegraphics[width=20mm]{USTL}}
\date{2005}
\begin{document}
\begin{frame}<handout:0>
\titlepage
\end{frame}
\section{ }
\begin{frame}
\frametitle{Things to do on a Sunday Afternoon}
\begin{block}{One could \ldots}
\begin{itemize}
\item walk the dog\dots \pause
\item read a book\pause
\item confuse a cat\pause
\end{itemize}
\end{block}
and many other things
\end{frame}
\end{document}
```

图4.2 - beamer 文档类的范例。

用PDFLATEX 编译图4.2 中的代码时，将得到一个PDF 文件，第一页为标题页，第二页有几个栏目，但当你单击你的演示文档时，一次显示一条栏目。

beamer 类创建PDF 文件的一个优点是直接生成可用的文档，而不像prosper需要先通过一个PostScript 步骤，也不像ppower4 宏包需要一个后加工处理才能生成演示文档。

用beamer 类，你可以用一个源文件生成几种版本。可以在源文件的中括弧中加入特定的选项来生成不同的版本。有下面几种版式：

- beamer PDF 屏幕阅读版本；
- trans 幻灯片版本；
- handout PDF 讲义版本。

默认的版本为beamer,你可以通过设置不同的全局选项来修改，例如：用\documentclass[10pt,handout]{beamer} 来生成讲义版本。

演示文稿外观依赖于你选择的主题。你可以选择beamer 类自带的一个主题， 也可以自己定义一个新的主题。详情请参见beamer 类的帮助文档beameruserguide.pdf。

让我们再来仔细分析图4.2 中的代码。

对于屏幕阅读版本的演示文稿\mode<beamer>,我们选择了Goettingen主题， 它将目录合成到导航面板。通过选项控制面板的大小（这个例子采用22mm），和确定面板的位置（正文右侧）。选项hideothersubsections 显示章节的标题， 但只显示当前章节的子节标题。对于\mode<trans> 和\mode<handout>的设置也是一样的，它们将出现在它们标准的版面上。

命令\title{}，\author{}，\institute{} 和\titlegraphic{} 定义标题页的内容。\title[]{} 和\author[]{} 的选项允许你定义显示在Goettingen主题的面板上的标题和作者名。

面板中的标题和子标题由frame 环境外面的命令\section{} 和\subsection{} 来创建。

屏幕底部的一些微型导航图标也可以让你浏览整个文档。它们的出现不依赖你选择的主题。

每张幻灯片或每版屏幕的内容放在frame 环境中。利用尖括弧(< 和>)里面的选项，用演示文档的一个版式来定义一个特殊的帧。在这个例子中，第一页不会由于参量<handout:0> 而显示为讲义模式。

除了幻灯片的标题页，强烈建议通过命令\frametitle{} 来重新设置每一张幻灯片的标题。如果需要，使用block 环境可以来定义子标题，在这个例子中也可体现出来。注意到章节命令\section{} 和\subsection{} 不在幻灯片上产生输出结果。

列表环境中的命令\pause 允许你一个接一个地显示列表栏目的内容。命令：\only、\uncover、\alt 和\temporal，可以让你获得其他的一些演示效果。很多情况下，你可以通过尖括弧中的内容来定制演示效果。

第五章　数学图形

大部分人使用LATEX排版文本。但是因为这种不面向内容和结构的写作方法太方便了， LATEX还提供了从文本描述生成图形输出的一种有局限性的方法。此外，大量的LATEX扩展被开发出来以克服这些限制。在本节中，我们将学习其中的一些。

5.1 概述

picture 环境可以在LATEX 里直接设计图形。一方面，这种方法有严重的局限性，比如线段的斜率和圆的半径只能在一个很小的范围内取值。另一方面， LATEX 2 ε 的picture 环境提供了\qbezier命令， “q” 表示“quadratic”。许多常用的曲线如圆、椭圆、或者悬链线都可以用二次B′ezier 曲线得到令人满意的近似，虽然这可能需要一些辛苦的数学准备。另外，如果有一种编程语言如Java 能用来生成LATEX源文档的\qbezier模块， picture 环境会更强大。

虽然直接在LATEX里设计图形的方法有严重的局限性而且通常比较繁琐， 但它还是很有用的。这份文档就是用它才变得体积很小，不需要插入额外的图片。

一些宏包,如epic 和eepic或者pstricks 可以排除picture 环境的局限，并大大地增强了LATEX的图形功能。

跟前两个宏包只是加强了picture 环境不同,pstricks 宏包有自己的绘图环境， pspicture。pstricks 的强大之处在于它广泛应用了PostScript。另外,许多宏包可以用来处理专门的问题。其一是XY-pic， 本章最后会讲到它。

5.2 picture 环境

5.2.1 基本命令

一个picture 环境1可以用下面两个命令中的一个来创建

\begin{picture}(*x*, *y*). . . \end{picture}

或者

\begin{picture}(*x*, *y*)(*x*0, *y*0). . . \end{picture}

数字*x*, *y*, *x*0, *y*0 是相对于\unitlength 而言的，任何时候（除了在picture环境之内以外），都可以使用命令如

\setlength{\unitlength}{1.2cm}

来改变。\unitlength 的默认值是1 pt。第一个数对，（*x*, *y*），在文档中为图形保留一个矩形的区域。可选的第二个数对，（*x*0, *y*0），为矩形左下角指派任意的坐标。

大多数的绘图命令是下面两种格式之一

\put(*x*, *y*){object}

或者

\multiput(*x*, *y*)(Δ*x*, Δ*y*){*n*}{object}

B′ ezier 曲线是一个例外。它们需要用命令

\qbezier(*x*1, *y*1)(*x*2, *y*2)(*x*3, *y*3)

来画。

5.2.2 线段

线段用命令

\put(*x*, *y*){\line(*x*1, *y*1){*length*}}

来画。命令\line 有两个参量：

1. 一个方向向量，
2. 一个长度。

方向向量需由以下整数构成

—6, —5, . . . , 5, 6,

而且它们需要互质（除1 以外，没有公约数），图形显示了第一象限中所有25个可能的斜率值。长度是相对于\unitlength 来说的。长度的参量当一个垂直线段时是垂直坐标，其他情况都是水平坐标。

```
\setlength{\unitlength}{5cm}
\begin{picture}(1,1)
\put(0,0){\line(0,1){1}}
\put(0,0){\line(1,0){1}}
\put(0,0){\line(5,6){.8333}}
\put(0,0){\line(6,1){1}}
\put(0,0){\line(6,5){1}}
\end{picture}
```

5.2.3 箭头

画箭头要用命令

\put(*x*, *y*){\vector(*x*1, *y*1){*length*}}

箭头的方向向量元素比线段的限制更严格，需由以下整数构成

−4, −3, . . . , 3, 4.

而且需要互质（除1 以外，没有公约数）。注意命令\thicklines 对指向左上方的两个箭头产生的效果。

```
\setlength{\unitlength}{0.75mm}
\begin{picture}(60,40)
\put(30,20){\vector(1,0){30}}
\put(30,20){\vector(4,1){20}}
\put(30,20){\vector(2,1){30}}
\thicklines
\put(30,20){\vector(-4,1){30}}
\put(30,20){\vector(-1,4){5}}
\thinlines
\put(30,20){\vector(-1,-1){5}}
\put(30,20){\vector(-1,-4){5}}
\end{picture}
```

5.2.4 圆

命令

\put(*x*, *y*){\circle{diameter}}

画了一个圆心在(*x*, *y*) 直径（不是半径）为diameter 的圆。picture 环境只允许直径最大是14 mm， 而且即使在这个限制之下， 也不是所有的直径都可获得。

命令\circle* 生成圆盘（填充的圆形）。

跟线段的情况一样，你可能需要其他宏包的帮助，比如eepic 或者pstricks。

picture 环境还有另外一个可能。如果你不怕麻烦的必要的计算（或者交给一个程序来处理），任意的圆和矩形都可以由二次B′ ezier 曲线拼成。

```
\setlength{\unitlength}{1mm}
\begin{picture}(60, 40)
\put(20,30){\circle{8}}
\put(20,30){\circle{32}}
\put(40,30){\circle{6}}
\put(40,30){\circle{14}}
\put(30,10){\circle*{4}}
\put(35,10){\circle*{5}}
\end{picture}
```

5.2.5 文本与公式

文本与公式可以使用\put 命令按照正常方式在picture 环境中使用。

```
\setlength{\unitlength}{0.8cm}
\begin{picture}(6,5)
\thicklines
\put(1,0.5){\line(2,1){3}}
\put(4,2){\line(-2,1){2}}
\put(2,3){\line(-2,-5){1}}
\put(0.7,0.3){$A$}
\put(4.05,1.9){$B$}
\put(1.7,2.95){$C$}
\put(3.1,2.5){$a$}
\put(1.3,1.7){$b$}
\put(2.5,1.05){$c$}
\put(0.3,4){$F=
\sqrt{s(s-a)(s-b)(s-c)}$}
\put(3.5,0.4){$\displaystyle
s:=\frac{a+b+c}{2}$}
\end{picture}
```

5.2.6 \multiput 与\linethickness

命令

\multiput(x, y)(Δx, Δy){n}{object}

有4 个参量：初始点，从一个对象到下一个的平移向量，对象的数目和要

绘制的对象。命令\linethickness 可作用于水平和垂直方向的线段，但不能作用于倾斜的线段和圆。然而，该命令可作用于二次B′ ezier 曲线。

```
\setlength{\unitlength}{2mm}
\begin{picture}(30,20)
\linethickness{0.075mm}
\multiput(0,0)(1,0){26}%
{\line(0,1){20}}
\end{picture}
```

5.2.7 椭圆

命令

\put(*x*, *y*){\oval(*w*, *h*)}

或

\put(*x*, *y*){\oval(*w*, *h*)[position]}

可以产生一个中心在(*x*, *y*) 处、宽为*w* 高为*h* 的椭圆。如本例所示，可选参量position 可以是b, t, l, r, 分别表示仅绘制椭圆的“下部”、“上部”、“左部”和“右部”，如例所示，这些参数可以进行组合。

以下两类命令可以控制线宽：一类为\linethickness{length}，另一类为\thinlines 与\thicklines。命令 \linethickness{length} 仅对水平和垂直直线(及二次B′ ezier 曲线)有作用，\thinlines 与\thicklines 则可以作用于倾斜的线段、圆和椭圆。

```
\setlength{\unitlength}{0.75cm}
\begin{picture}(6,4)
\linethickness{0.075mm}
\thicklines
\put(2,1){\oval(3,1.8)[tl]}
\put(4,1){\oval(3,1.8)[b]}
\put(4,3){\oval(3,1.8)[r]}
\put(3,1.5){\oval(1.8,0.4)}
\end{picture}
```

5.2.8 重复使用预定义的图形盒子

一个图形盒子可以使用命令\newsavebox{name}进行声明,然后使用命令\savebox{name}(width,height)[position]{content}
进行定义，最后使用命令\put(*x*, *y*)\usebox{name}进行任意次数的重复绘

制。

可选参数position 的作用是定义图形存放盒子的“锚点”。在本例中该参数被设置为bl，从而将锚点设置为图形存放盒子的左下角。其他的位置描述有t和r，分别表示“上”和“右”。

参量name 指明了LATEX 存储槽，揭示了其命令本质（在本例中指反斜线）。图形盒子可以嵌套：在本例中，\foldera 被用在了\folderb 的定义中。由于命令\line 在线段长度小于大约3 mm 的时候不能正常工作，所以必须使用命令\oval。

```
\setlength{\unitlength}{0.5mm}
\begin{picture}(120,168)
\newsavebox{\foldera}
\savebox{\foldera}
(40,32)[bl]{% definition
\multiput(0,0)(0,28){2}
{\line(1,0){40}}
\multiput(0,0)(40,0){2}
{\line(0,1){28}}
\put(1,28){\oval(2,2)[tl]}
\put(1,29){\line(1,0){5}}
\put(9,29){\oval(6,6)[tl]}
\put(9,32){\line(1,0){8}}
\put(17,29){\oval(6,6)[tr]}
\put(20,29){\line(1,0){19}}
\put(39,28){\oval(2,2)[tr]}
}
\newsavebox{\folderb}
\savebox{\folderb}
(40,32)[l]{% definition
\put(0,14){\line(1,0){8}}
\put(8,0){\usebox{\foldera}}
}
\put(34,26){\line(0,1){102}}
\put(14,128){\usebox{\foldera}}
\multiput(34,86)(0,-37){3}
{\usebox{\folderb}}
\end{picture}
```

5.2.9 二次 B′ ezier 曲线

令$P1=(x1, y1)$，$P2=(x2, y2)$ 和$m1$，$m2$ 分别表示一条二次B′ ezier 曲线的两个端点及其对应斜率。中间控制点$S=(x, y)$ 则由下述方程给出

$$\begin{cases} x &= \dfrac{m_2x_2 - m_1x_1 - (y_2 - y_1)}{m_2 - m_1}, \\ y &= y_i + m_i(x - x_i) \qquad (i = 1, 2). \end{cases} \tag{5.1}$$

\qbezier

```
\setlength{\unitlength}{0.8cm}
\begin{picture}(6,4)
\linethickness{0.075mm}
\multiput(0,0)(1,0){7}
{\line(0,1){4}}
\multiput(0,0)(0,1){5}
{\line(1,0){6}}
\thicklines
\put(0.5,0.5){\line(1,5){0.5}}
\put(1,3){\line(4,1){2}}
\qbezier(0.5,0.5)(1,3)(3,3.5)
\thinlines
\put(2.5,2){\line(2,-1){3}}
\put(5.5,0.5){\line(-1,5){0.5}}
\linethickness{1mm}
\qbezier(2.5,2)(5.5,0.5)(5,3)
\thinlines
\qbezier(4,2)(4,3)(3,3)
\qbezier(3,3)(2,3)(2,2)
\qbezier(2,2)(2,1)(3,1)
\qbezier(3,1)(4,1)(4,2)
\end{picture}
```

5.2.10 悬链线

悬链线$y = \cosh x - 1$ 对称的两半由二次B′ ezier 曲线分别近似地绘成。曲线的右半部分终止于点(2, 2.7622)，对应的斜率为$m = 3.6269$。再次使用公式(5.1)，我们可以计算中间控制点。计算结果为(1.2384, 0)

和(－1.2384, 0)。图中的十字为真正的悬链线上的点。误差小于百分之一，很难被发现。

\begin{picture}(4.3,3.6)(-2.5,-0.25)定义了方便的“数学”坐标：左下角（由黑色圆点标出）坐标是(－2.5, －0.25)。

```
\setlength{\unitlength}{1cm}
\begin {picture} (4.3, 3.6)
(-2.5,-0.25)
\put(-2,0){\vector(1,0){4.4}}
\put(2.45,-.05){$x$}
\put(0,0){\vector(0,1){3.2}}
\put(0,3.35){\makebox(0,0){$y$}}
\qbezier(0.0,0.0)(1.2384,0.0)
(2.0,2.7622)
\qbezier(0.0,0.0)(-1.2384,0.0)
(-2.0,2.7622)
\linethickness{.075mm}
\multiput(-2,0)(1,0){5}
{\line(0,1){3}}
\multiput(-2,0)(0,1){4}
{\line(1,0){4}}
\linethickness{.2mm}
\put( .3,.12763){\line(1,0){.4}}
\put(.5,-.07237){\line(0,1){.4}}
\put(-.7,.12763){\line(1,0){.4}}
\put(-.5,-.07237){\line(0,1){.4}}
\put(.8,.54308){\line(1,0){.4}}
\put(1,.34308){\line(0,1){.4}}
\put(-1.2,.54308){\line(1,0){.4}}
\put(-1,.34308){\line(0,1){.4}}
\put(1.3,1.35241){\line(1,0){.4}}
\put(1.5,1.15241){\line(0,1){.4}}
\put(-1.7,1.35241){\line(1,0){.4}}
\put(-1.5,1.15241){\line(0,1){.4}}
\put(-2.5,-0.25){\circle*{0.2}}
\end{picture}
```

5.2.11 坐标的相对性

公式(5.1) 给出了两条B′ ezier 曲线的控制点。正向分支由$P1$ = (0, 0), $m1$ = 1和$P2$ = (2, tanh 2), $m2$ = 1/ cosh2 2 确定。与前例相同，下例也定义了在数学上方便的坐标，左下角的坐标是(−3, −2)。

```
\setlength{\unitlength}{0.8cm}
\begin{picture}(6,4)(-3,-2)
\put(-2.5,0){\vector(1,0){5}}
\put(2.7,-0.1){$\chi$}
\put(0,-1.5){\vector(0,1){3}}
\multiput(-2.5,1)(0.4,0){13}
{\line(1,0){0.2}}
\multiput(-2.5,-1)(0.4,0){13}
{\line(1,0){0.2}}
\put(0.2,1.4)
{$\beta=v/c=\tanh\chi$}
\qbezier(0,0)(0.8853,0.8853)
(2,0.9640)
\qbezier(0,0)(-0.8853,-0.8853)
(-2,-0.9640)
\put(-3,-2){\circle*{0.2}}
\end{picture}
```

5.3 XY-pic

xy 是绘制流程图的专用宏包。要想使用它，只需在导言区加上：
\usepackage[options]{xy}
options 列出你需要载入的XY-pic 的选项。这些选项基本上被用于调试这个宏包的使用。建议你使用all，可以让LATEX 载入XY的所有命令。

XY-pic 流程图被绘制在一幅以矩阵定位的画布上，每一个流程图元素被放在矩阵的一个单元中：

```
\begin{displaymath}
\xymatrix{A & B \\
C & D }
\end{displaymath}
```

命令\xymatrix 必须置于数学模式中。这里，我们设定了一个两行两列的矩阵。为了画出流程，我们只需要使用命令\ar 增加带方向的箭头即可。

```
\begin{displaymath}
\xymatrix{ A \ar[r] & B \ar[d] \\
D \ar[u] & C \ar[l] }
\end{displaymath}
```

箭头命令要放在其出发的那个单元里。参量是箭头的方向（u：上， d：下， r：右以及l：左）.

```
\begin{displaymath}
\xymatrix{
A \ar[d] \ar[dr] \ar[r] & B \\
D & C }
\end{displaymath}
```

要画对角线，可以指定不只一个方向参量。实际上，你还可以重复同一个方向来得到更大的箭头。

```
\begin{displaymath}
\xymatrix{
A \ar[d] \ar[dr] \ar[drr] &&\\
B & C & D }
\end{displaymath}
```

我们还可以绘制一些更有趣的流程图，给箭头加上标签，只需要使用普通的上标和下标。

```
\begin{displaymath}
\xymatrix{
A \ar[r]^f \ar[d]_g &
B \ar[d]^{g'} \\
D \ar[r]_{f'} & C }
\end{displaymath}
```

如图所示，就像数学模式里一样使用上下标。唯一的区别在于：上标表示放在“箭头的上方”，下标表示放在“箭头的下方”。把文本放到箭头上可以用。

```
\begin{displaymath}
\xymatrix{
A \ar[r]|f \ar[d]|g &
B \ar[d]|{g'} \\
D \ar[r]|{f'} & C }
\end{displaymath}
```

绘制空心箭头的命令是\ar[...]|\hole。

某些情况下，需要区分不同类型的箭头。可以给它们标上标签，或者使用不同的外观来实现：

```
\shorthandoff{"}
\begin{displaymath}
\xymatrix{
\bullet\ar@{->}[rr] && \bullet\\
\bullet\ar@{.<}[rr] && \bullet\\
\bullet\ar@{~)}[rr] && \bullet\\
\bullet\ar@{=(}[rr] && \bullet\\
\bullet\ar@{~/}[rr] && \bullet\\
\bullet\ar@{^{(}->}[rr] &&
\bullet\\
\bullet\ar@2{->}[rr] && \bullet\\
\bullet\ar@3{->}[rr] && \bullet\\
\bullet\ar@{=+}[rr] && \bullet
}
\end{displaymath}
\shorthandon{"}
```

注意下面两幅流程图的区别：

```
\begin{displaymath}
\xymatrix{
\bullet \ar[r]
\ar@{.>}[r] &
\bullet
}
\end{displaymath}
```

```
\begin{displaymath}
\xymatrix{
\bullet \ar@/^/[r]
\ar@/_/@{.>}[r] &
\bullet
}
\end{displaymath}
```

两条斜线间的修饰元素决定了曲线应该如何被画出。XY-pi 提供了很多办法来改变曲线的形状。

第六章　定制 LATEX

到目前为止，运用你所学过的命令可以制作出能被绝大多数读者接受的文档。尽管这些文档看上去不够奇妙，但它们遵循了高质量排版的已有规则， 这些规则可以使得文档易读，同时看起来也非常舒适。

然而在一些情况下，LATEX 也许并没有提供适合你需要的命令或者环境， 或者现有的命令产生的输出和你想要的不同。本章中，将尝试给出一些新的技巧，运用这些技巧可以教会LATEX 玩一些新的把戏，也可以使得LATEX 产生与众不同的输出。

6.1 新建命令、环境和宏包

6.1.1 新建命令

为了建立你自己的命令，可以使用如下的命令：

\newcommand{name}[num]{definition}

基本上，这个命令有两个参量，第一个name 是你想要建立的命令的名称，第二个definition 是命令的定义。方括号里的参数num 是可选的， 用于指定新命令所需的参量数目（最多9 个）。如果不给出这个参数，默认就是0，也就是新建的命令不要任何参量。

接下来的两个例子有助你的理解。第一个例子定义了一个新的命令：\tnss。

这个命令是句子“The Not So Short Introduction to LATEX 2 ε” 的简写。如果你需要在文档中多次使用本书的名称，那么定义这个命令将是非常方便的。

```
\newcommand{\tnss}{The not
so Short Introduction to
\LaTeXe}
This is ``\tnss'' \ldots{}
``\tnss''
```

下一个例子演示了如何建立一个接受单一参数的命令。在命令的定义中，标记#1 将被你指定的参量所代替。如果你想使用多个参量，那么可以

依次使用#2、……、#9 等标记。

```
\newcommand{\txsit}[1]
{This is the \emph{#1} Short
Introduction to \LaTeXe}
% in the document body:
\begin{itemize}
\item \txsit{not so}
\item \txsit{very}
\end{itemize}
```

LATEX 不允许你新建一个与现有命令重名的命令。如果你确实需要这么做，有一个专门的命令用于处理这种情况：\renewcommand。它使用与命令\newcommand 相同的语法。

在某些情况之下，你可能会希望使用\providecommand 命令。它完成与\newcommand 命令相同的工作。但如果命令已经存在，LATEX 2 ε 将会悄悄忽略原有的那个。

处理LATEX 命令后尾随的空格有一些要注意的事项，参看第4 页可以获得更多这方面的信息。

6.1.2 新建环境

与\newcommand 命令类似，有一个命令用于建立新的环境。它的语法如下所示：\newenvironment{name}[num]{before}{after}

同样地，\newenvironment 命有一个可选的参量。在before 中的内容将在此环境包含的文本之前处理，而在after 中的内容将在遇到\end{name}命令时处理。

下面的例子演示了\newenvironment 命令的用法：

```
\newenvironment{king}
{\rule{1ex}{1ex}%
\hspace{\stretch{1}}}
{\hspace{\stretch{1}}%
\rule{1ex}{1ex}}
\begin{king}
My humble subjects \ldots
\end{king}
My humble subjects . . .
```

参量num 的使用方式与\newcommand 命令相同。LATEX 还同样保证你

不会不小心新建重名的环境。如果你确实希望改变一个现有的环境，你可以使用命令\renewenvironment,它使用和命令\newenvironment 相同的语法。

6.1.3 额外的空白间距

当创建新的环境时，你或许会为遇到额外的空白间距而烦扰，这些间距可能产生严重的后果。比如当你建立一个标题环境，既不要自身的缩进也不要紧接着的下一段缩进时，在begin 中加入命令\ignorespaces 会使新环境忽略执行begin 之后遇到的一切空白间距，而end 就需要耍个小花招,因为我们要等到环境结束后才开始处理。使用\ignorespacesafterend，LATEX 会在end 处理完毕后，产生一个\ignorespaces。

```
\newenvironment{simple}%
{\noindent}%
{\par\noindent}
\begin{simple}
See the space\\to the left.
\end{simple}
Same\\here.
```

```
\newenvironment{correct}%
{\noindent\ignorespaces}%
{\par\noindent%
\ignorespacesafterend}
\begin{correct}
No space\\to the left.
\end{correct}
Same\\here.
```

6.1.4 自建宏包

如果你定义了很多新的环境和命令，你的文档的导言部分将变得相当长，在这种情况下， 建立一个新的LATEX 宏包来存放所有你自己定义的命令和环境将是一个好的处理方式。然后你可以在文档中使用\usepackage 命令来引入自定义宏包。

```
% Demo Package by Tobias Oetiker
\ProvidesPackage{demopack}
\newcommand{\tnss}{The not so Short Introduction
to \LaTeXe}
\newcommand{\txsit}[1]{The \emph{#1} Short
Introduction to \LaTeXe}
\newenvironment{king}{\begin{quote}}{\end{quote}}
```

图6.1 - 宏包样例。

写一个宏包的基本工作就是将你原本很长的文档导言内容拷贝到另外一个的文件中去， 这个文件需要以.sty 结尾。你还加入一个专用的命令：

\ProvidesPackage{package name}

这个命令应该放在你的宏包的最前面。\ProvidesPackage 告诉LATEX 宏包的名称从而让LATEX 在你尝试两次引入同一个宏包的时候给出一个明显的错误信息，图6.1 给出了一个小的宏包示例，其中包含了我们之前定义的一些命令。

6.2 字体和字号

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，你可能会想要手动改变文档使用的字体及其大小。为了完成这个目的，你可以使用表6.1 和表6.2 中列出的那些命令。每个字体的实际大小是一个设计问题，并且它依赖于文档所使用的文档类及其选项。表6.3 列出了这些命令在标准文档类中的绝对pt 大小。

表6.1 - 字体。

\textrm{...}	roman	\textsf{...}	sans serif
\texttt{...}	typewriter		
\textmd{...}	medium	\textbf{...}	bold face
\textup{...}	upright	\textit{...}	italic
\textsl{...}	slanted	\textsc{...}	Small Caps
\emph{...}	emphasized	\textnormal{...}	document font

表6.2 - 字号。

\tiny	tiny font	\large	large font
\scriptsize	very small font	\Large	larger font
\footnotesize	quite small	\LARGE	very large font

	font		
\small	small font	\huge	huge
\normalsize	normal font	\Huge	largest

表6.3 - 标准文档类中的绝对pt 大小。

大小	10pt（默认）	11pt选项	12pt选项
\tiny	5pt	6pt	6pt
\scriptsize	7pt	8pt	8pt
\footnotesize	8pt	9pt	10pt
\small	9pt	10pt	11pt
\normalsize	10pt	11pt	12pt
\large	12pt	12pt	14pt
\Large	14pt	14pt	17pt
\LARGE	17pt	17pt	20pt
\huge	20pt	20pt	25pt
\Huge	25pt	25pt	25pt

```
{\small The small and
\textbf{bold} Romans ruled}
{\Large all of great big
    \textit{Italy}.}
```

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。

在**数学模式**中你可以使用字体变换命令来暂时退出**数学模式**，然后输入一些正常的文字。如果你希望改变数学公式本身所使用的字体,LATEX 提供了另外一套命令。参看表6.4。

表 6.4 – 数学字体。

\mathrm{...}	Roman Font
\mathbf{...}	**Boldface Font**
\mathsf{...}	Sans Serif Font
\mathtt{...}	Typewriter Font
\mathit{...}	*Italic Font*
\mathcal{...}	CALLIGRAPHIC FONT
\mathnormal{...}	*Normal Font*

使用字体命令的时候，大括号(curly braces) 扮演了一个重要角色。

它们被用于建立所谓的组(group)。组限制了大多数LATEX 命令的作用范围。

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

如果段落在字体的作用范围中结束，那么字号命令还将改变段落中行距。因此用于分组的反向大括号} 不应该太早出现。注意下面两个例子中\par 命令的位置。

```
{\Large Don't read this!
It is not true.
     You can believe me!\par}
```

```
{\Large This is not true either.
     But remember I am a liar.}\par
```

如果你希望改变整段甚至更多文本的字体，你可能应该使用字体变换命令的环境语法。

```
\begin{Large}
This is not true.
But then again, what is these
days \ldots
     \end{Large}
```

这将使你从一堆大括号中解脱出来。

6.3 间距

6.3.1 行距

如果你想在文档中使用更大的行距，你可以在导言中使用如下命令进行设定：\linespread{factor}

如\linespread{1.3} 产生1.5 倍行距，而\linespread{1.6} 则产生双倍行距。缺省情况下的行距为1。

注意\linespread 的效果相当夸张而且不适合出版工作。因此如果你很想改变行距可能会希望使用如下的命令：

```
\setlength{\baselineskip}{1.5\baselineskip}
{\setlength{\baselineskip}%
{1.5\baselineskip}
This paragraph is typeset with
the baseline skip set to 1.5 of
what it was before. Note the par
command at the end of the
paragraph.\par}
This paragraph has a clear
purpose, it shows that after the
curly brace has been closed,
     everything is back to normal.
```

6.3.2 段落格式

在LATEX 中，有两个参数可以影响段落的布局。在文档的导言部分，可以通过如下的定义来改变段落的布局。

\setlength{\parindent}{0pt}

\setlength{\parskip}{1ex plus 0.5ex minus 0.2ex}

这两个命令增加了段落间距，并将首行缩进设置为0。

例子中,长度设定中的plus 和minus 部分将使得TEX 按照指定大小压缩和伸展段落间距。为了使得段落正确的显示在页面之上，TEX 将在0.8ex到1.5ex 之间调整段落间距。

在欧洲大陆，段落通常用一些空白分隔并且一般首行不缩进。但是值得注意的是，这也会影响目录。目录的行距也会变得非常疏松。为了避免这种情况，你可能需要将上面的两个命令从导言中移到文档中\tableofcontents 以下适合的位置，或者根本不要使用这些，因为一般来说专业的书籍都是用缩进并且通常不用空白来分离段落。如果你想缩进一个本来没有缩进的段落，可以在段落的开始使用命令：

\indent

当然，这个命令只有在\parindent 不为零的情况下才有效果。

为了创建一个不缩进的段落，你可以在段落的开始部分使用命令：

\noindent

当文档以正文而不是章节命令开始的时候，这个命令会提供方便。

6.3.3 水平间距

LATEX 系统自动决定单词和句子之间的距离。为了增加水平距离， 使

用命令：
\hspace{length}

如果这个水平间距即使在行首或者行末也应该保持的话，用命令
\hspace*
代替\hspace。命令的length 参数在简单的情况下只是一个带有单位的数字。最重要的长度单位在表6.5 中列了出来。

```
This\hspace{1.5cm}is a space  of 1.5 cm.
```

表 6.5 – TEX 单位。

mm	millimetre $\approx 1/25$ inch	⊔
cm	centimetre = 10 mm	⊔——⊔
in	inch = 25.4 mm	⊔————⊔
pt	point $\approx 1/72$ inch $\approx \frac{1}{3}$ mm	‖
em	approx width of an ‘M’ in the current font	⊔⊔
ex	approx height of an ‘x’ in the current font	⊔

命令
\stretch{n}
将产生一个特殊的橡皮长度：一个能把行内剩余所有空隙填满的空白。如果两个\hspace{\stretch{n}} 命令位于同一行，那么它们将根据伸缩因子分配空间。

```
x\hspace{\stretch{1}}x\hspace{\stretch{3}}x
```

当在正文中使用水平间距的时候，相对于字号来调整间距大小会更有道理。这可以通过使用与文本有关的单位em 和ex 来实现：

```
{\Large{}big\hspace{1em}y}\\{\tiny{}tin\hspace{1em}y}
```

6.3.4 垂直间距

在段落、节、小节…… 之间的距离是由LATEX 系统自动决定的。如果必要的话，可以在两段之间增加额外的距离，使用的命令如下所示：
\vspace{length}
这个命令通常用于两个空行之间。如果这个额外的行距应该在于页的顶部和末尾也保留下来，那么使用这个命令的星号版本\vspace* 来代替

\vspace。
命令\stretch 和\pagebreak 结合使用可以在页的最后一行输出文本，也可以用来保证文本在页面上垂直居中。

```
Some text \ldots
\vspace{\stretch{1}}
This goes onto the last line of the page.\pagebreak
```

同一段或**同一个表格**中两行之间的距离可以用如下命令来指定：
\\[length]

使用命令\bigskip 和\smallskip 你可以获得一个预定义的垂直间距。

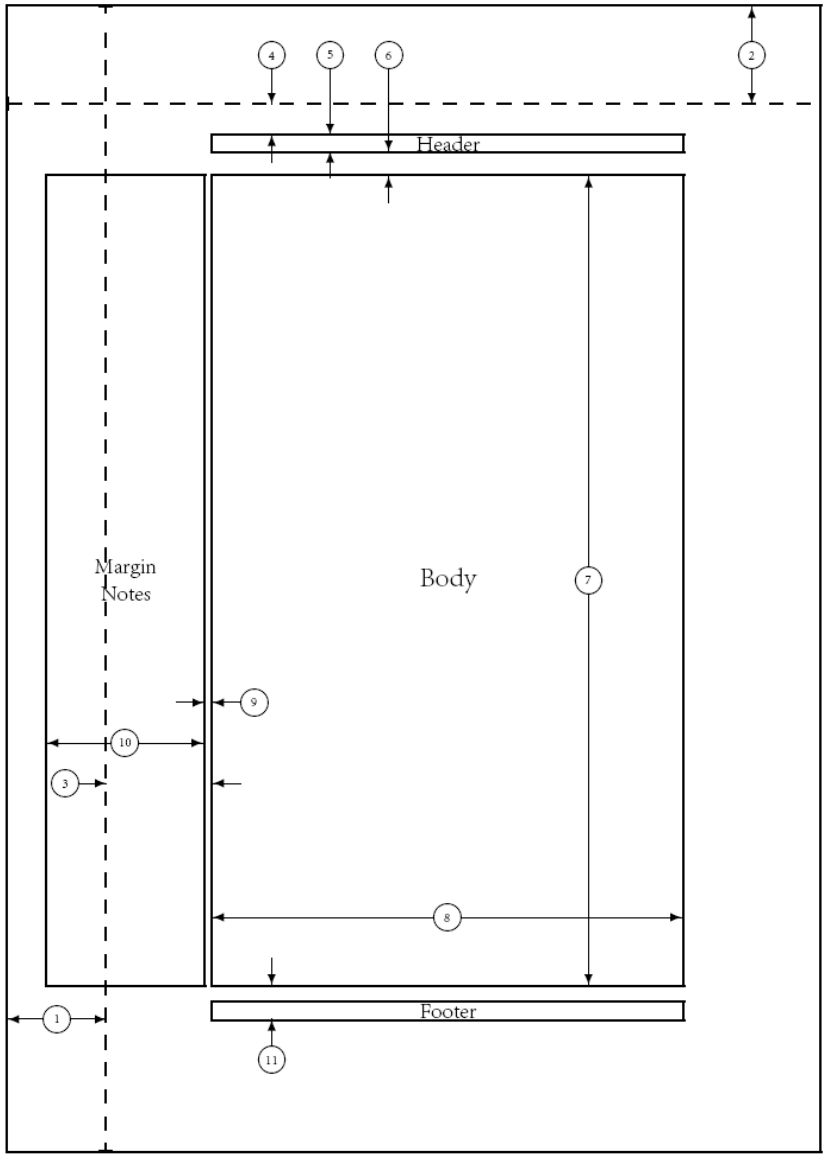
6.4 页面布局

LATEX 2 ε 允许你在\documentclass 命令中指定纸张尺寸(paper size)。然后它将自动的选择合适的页边距。但有些时候你可能不满意LATEX的预设值，这个时候你可以自己改变这些参数。图6.2 中显示了所有能改变的页面参数。

LATEX 提供了两个命令来改变这些参数。他们通常在文章的导言部分使用。
第一个命令给某些参数一个固定的值：
\setlength{parameter}{length}
第二个命令给某些参数增加一个长度：
\addtolength{parameter}{length}
第二个命令实际上比\setlength 命令更为实用,因为你可以相对于现有的设置来获得所需的结果。为了给文本的宽度增加1 厘米，将如下的命令放置到文档导言：

```
\addtolength{\hoffset}{-0.5cm}
\addtolength{\textwidth}{1cm}
```

这时候，你可能会想要看看calc 包，它允许你在\setlength 的参量中进行算术运算。它也可以运用到任何用数值作为函数参量的地方。



1	one inch + \hoffset	2	one inch + \voffset
3	\oddsidemargin = 28pt or \evensidemargin	4	\topmargin = 23pt
5	\headheight = 12pt	6	\headsep = 18pt
7	\textheight = 598pt	8	\textwidth = 345pt
9	\marginparsep = 7pt	10	\marginparwidth = 115pt
11	\footskip = 25pt		\marginparpush = 5pt (not shown)
	\hoffset = 0pt		\voffset = 0pt
	\paperwidth = 597pt		\paperheight = 845pt

图 6.2 – 页面布局参数。

6.5 更有趣的长度

只要可能，就应该避免在LATEX 文档中使用绝对长度。我更愿意通过页面中其他元素的宽度或高度来指定长度。比如一个图形，我指定 \textwidth 作为它的宽度从而使得图形恰好充满整个页面。

下面的三个命令允许你获得一个文本串的宽度、高度以及深度。

\settoheight{variable}{text}
\settodepth{variable}{text}
\settowidth{variable}{text}

下面的例子显示了如何应用这些命令：

```
\flushleft
\newenvironment{vardesc}[1]{%
\settowidth{\parindent}{#1:\ }
\makebox[0pt][r]{#1:\ }}{}
\begin{displaymath}
a^2+b^2=c^2
\end{displaymath}
\begin{vardesc}{Where}$a$,
$b$ -- are adjoin to the right
angle of a right-angled triangle.
$c$ -- is the hypotenuse of
the triangle and feels lonely.
$d$ -- finally does not show up
here at all. Isn't that puzzling?
    \end{vardesc}
```