XeLaTeX 及 WinEdt 6.0 入门指南

hy_haoyun [*]

版本号：v1.0991　　修改日期：2011/9/20

[*]E-mail: haoyun_tex@163.com

目录

1 写作目的及学习目标

简单的来说, 我写作本文的目的是为了使大家学会以 WinEdt 为编辑工具，使用 XeLaTeX 进行科技文章特别是数学文章的写作。所以，学习的目标当然是**迅速**学会使用 XeLaTeX。但具体的来说，就涉及到为什么为什么要学习 TeX，为什么选用 XeLaTeX 及 WinEdt，学习达到怎样的程度等等诸多问题，下面将会一一介绍。首先应该明白 XeLaTeX 的含义：TeX 是一个排版底层软件，它最初使用 Plain TeX 语言，LaTeX 是在 Plain TeX 的基础上开发出的一种更为简单的语言，XeTeX 是在 TeX 的基础上开发的新的排版底层软件，把语言及软件写在一起，叫做 XeLaTeX。好了，进入正题。

1.1 为什么使用 TeX

这其实是一个无需过多讨论的问题，网上有大量的内容讨论这个问题，也介绍了很多 TeX、LaTeX、PDFTeX、XeLaTeX、ConTeXt 等等 TeX（搞不清这些是什么？可以看看 [1]）的发展史以及高德纳的传奇故事，我在这里没有必要再次的重述，重述也仅仅是摘抄罢了。所以这里对 TeX 不再做任何的介绍，也不再做它和 MS Word 任何的优劣分析，需要的自己 Google。总之一句话，作为一个现代的，学数学的，非常有必要掌握 TeX，越早越好。

1.2 为什么使用 XeLaTeX 以及 WinEdt

TeX 已经发展了很多很多年了，至今仍在迅速发展。从**引擎（软件）**层面上看，一个简单的不完全的（甚至不准确的）发展线路是

$$\text{TeX}\to\text{e-TeX}\to\text{PDFTeX}\to\text{XeTeX(2004) \& LuaTeX(2006).}$$

从**语言**层面上看，发展思路（简单的甚至不准确的）为

$$\text{Plain TeX}\to\mathcal{AMS}\text{TeX}\to\text{LaTeX}\to\text{ConTeXt.}$$

现在我们要使用的是 XeLaTeX，也就是说，使用 LaTeX 的语言，用 XeTeX 排版。从上面的发展思路来看，这都不是最新的，但是是相对新的。但是基于以下的原因，我们使用 XeLaTeX。

- 从引擎上看，TeX 开发的时候就没有考虑到非英语使用者，所以起初是不支持非英语的。后来 TeX 发展了，越来越多的人开始使用它，

迫切需要解决非英语语言的输入输出问题，特别是中日韩等东亚语言。于是后来中科院张林波开发了 CCT 系统（1991 ？），德国人（竟然是德国人!）Werner Lemberg 开发出来了 CJK（Chinese、Japanese & Korean）宏包（1996 ？），华东师大陈志杰开发了 TY（天元）系统（2000 ？）。然而这些都没有从根本上解决中文的输入输出问题。最终，新的 XeTeX 从底层上支持 Unicode 编码，从而从根本上支持了中文。现在可以说，CCT 及 TY 都已经过时了（尽管 CCT 仍过支持 XeTeX 的尝试）。同时，由于市面上卖的书都是基于 CJK 宏包的，所以国内大部分人输入中文用的是 CJK 宏包，用 PDFTeX 编译。最新出版的《LaTeX 2ε完全学习手册》（2011）也鲜有介绍 XeTeX。所以有必要写一些东西介绍一下 XeTeX，尽管极不系统，甚至也没有太多的涉及 XeTeX 专有的东西。不过我们既然学习了，就应该学习较先进的。而 LuaTeX 尚处于发展之中，2012 年正式推出第一版。这样，本文介绍 XeTeX 用法，读者学习 XeTeX 用法都有充分理由了，即它是新的，旧的过时了，更新的还没有开发完善（不过，喜欢尝鲜的朋友可以自己试试）。**所以选用 XeTeX 底层**。

- 从语言上看，我们选择 LaTeX，而不是最新的 ConTeXt，这是由于新一代的 ConTeXt 尚在发展之中（旧一代已经可用，但优势不大），尽管它更为先进，但仍有很多不完善之处。而 LaTeX 经过了这么多年的发展，有很多方便的可用的宏包，一定程度上可以更好的实现一些功能。出于更完善的考虑，所以学习使用 LaTeX。而且，LaTeX 也仍在开发之中。事实上，现在已经从 LaTeX 2ε 时代逐渐向 LaTeX3 时代过渡了。新的 TeXLive 2010 发行版本已经包含了 LaTeX3，也就是说, 开发了好多年的 LaTeX3 已经准备开始投入使用了。不过，LaTeX3 还达不到可用的程度。基于以上考虑，即它是新的，更新的还没有开发好，**所以使用 LaTeX 语言**。

那么为什么选择 WinEdt 呢？你现在需要注意一点，WinEdt 不是 TeX，它仅仅是一款文本编辑器，可以理解为记事本的高级扩展版。利用它仅仅是为了我们的方便。使用 WinEdt 是因为 WinEdt 是 Windows 平台下使用最为广泛的 TeX 文本编辑器。另一个重要的原因是 CTeX 小组发行的 CTeX 套装（最新版本号：v2.9.0.152 2011/1/21）里面包含了最新的 WinEdt。WinEdt 用起来是很方便的，它有自动补全、一键编译、查看等等很多辅助功能。我自己使用 Linux 下的 Kile 软件，但是考虑到大部

分人使用 Windows，所以在这里介绍 Windows 平台下的 WinEdt，而不是 Linux 平台下的 Kile。当然还有很多很多的其他 TEX 文档编辑器，具体可以参考 [2–4]。不过请注意，WinEdt 并不是一个免费的软件，从一定程度上来说, 使用盗版软件是违法且不道德的。有人说它是一款共享软件（Shareware），可以免费使用，然而归根结底，共享软件有一定的试用期，过了试用期是需要付费的。通过技术手段延长使用期，恐怕不是太好。

1.3 如何学习及学到怎样的程度

学习 TEX 其实很简单，找一本教程按部就班实践一次即可。遇到问题到 Google 一下基本都可以解决。大家经常推荐的入门书是 [5, 6]。解决不了的可以到论坛上去问。不过一定要先自己 Google 找答案，实在找不到再去问，因为没有任何人有义务回答你由于懒惰而产生的疑问。不要养成遇到问题就去问的习惯。很多问题需要自己先尝试去解决。这样印象深刻并且更容易掌握。至于学到什么程度，我想看完本指南之后可以打出一个和本指南输出效果完全一样的文档就足够了。之后就要看自己的学习了。

事实上，我会在接下来的内容里介绍这篇文档里的每一个效果是如何实现的。边看文档边操作就学会了。这篇文档的写作思路也是流水账式的，并且相当啰嗦。啰嗦的结果就是这篇文章看起来是有些长，但是内容并不多。希望有耐心看下去。

1.4 还有问题

还有问题啊，Google 一下就行了。当然，可以发邮件来问我，不过我必然不能解答所有的问题，因为我也只是一个新手。我的邮箱是 haoyun_tex@163.com。**问问题的最好去处是 http://bbs.ctex.org**。这是中文 CTEX 小组的大本营，关于 TEX 的任何问题都可以到哪里去搜索或者咨询。不过一定要先自己搜索，再提问题！

2 TEX 系统的安装

至于 TEX 系统的安装，这里没有必要多说，就是安装一个软件而已。当然，TEX 还可以有其他的更高级的，更自定义的安装方式，可是这些就很复杂了。这里推荐下载 TEXLive 2010 或者 CTEX v2.9.0.152（如果有最新版的就下载最新版的），完全安装就行了。现在的硬盘空间那么大，应该

不在乎 2—3G 的空间。不过注意：安装的路径最好只更改盘符，不要更改其他路径，即最好安装到 X:/TeXLive/ 或 X:/CTEX/，其中 X 是盘符，不要更改除盘符以外的路径。

在这里你要明白，这里 TeXLive 以及 CTeX 只是各种 TeX 底层以及一些其他文件的打包。叫做 TeX 的**发行版本**，它们并不是什么新的软件或者语言。

等到学会使用 TeX 了，便可以根据自己的需求，自定义安装 TeX 了。另外，如果安装 TeXLive 2010，还需要自己安装 WinEdt，自己下载、破解。我不在这里鼓动大家去破解软件，前面说了，这是不道德且违法的（此处有争议），即使国情如此。另外还得配置一下 PDF 的阅读器 SumatraPDF。所以，还是安装更方便的 CTeX v2.9.0.152 好了。

3 最基本的内容:LaTeX 的基本格式等

到现在为止，假定你已经安装好了一个 TeX 发行版本以及 WinEdt（CTeX 套装自带）。

3.1 认识 WinEdt

现在打开 WInEdt，并且新建一个空白的文件（可别说不会新建……）。如 图 1 所示。

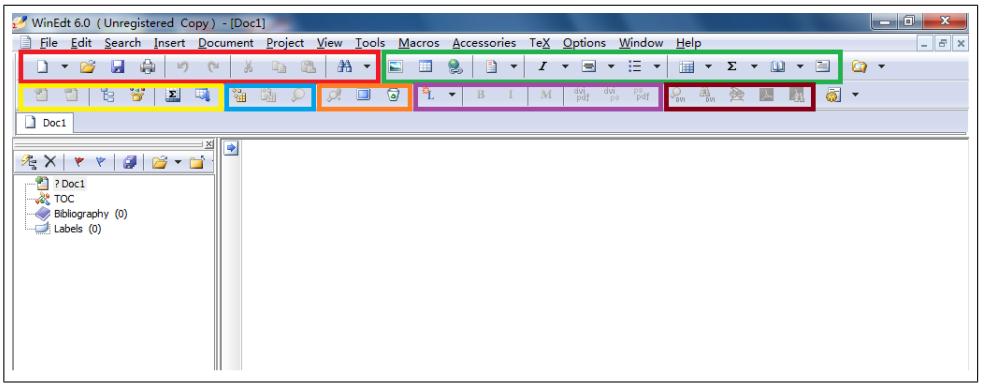

图 1 WinEdt 界面

工具栏上面有很多的按钮，这里不可能也没有必要解释每一个按钮到底有什么用。事实上，很多键是用不到的。同时，菜单栏也有极其多的菜单，同样不可能也没有必要详细的介绍每一个。感兴趣并且需要的话自己

想办法了解（按钮和菜单多足见功能之强大）。我只说一下我们需要用到的：

- 首先注意一下 **图 1** 中红色框中的几个键，这是极其通用的新建、打开、保存、打印、撤销、重做、剪切、复制、粘贴、查找等等。无需多言。
- 其次需要注意一下紫色框中的第一个，这个是最最常用的**编译**按钮，由于我们使用 XeLaTeX，所以现在点击第一个按钮旁边的箭头，选择 XeLaTeX （倒数第二个），以后我们称它为**编译**按钮。**编译**作为一个动词就是点这个按钮。紫色框中的 “**B**” 稍后也会用到。
- 深红色框中第 4 个，也就是有 Adobe Reader 标志的那一个。这个用于查看编译生成的 PDF 文件，以后称它为**查看**按钮。**查看**作为一个动词就是点这个按钮。

知道了这些就足够了. 现在输入一些东西. 不妨先输入

```
1 \documentclass{article}
```

现在，如果你觉得 WinEdt 的的字体有些小的话，你可以按照如下的方式去设置它：

> 菜单栏 Options——Options——左边 Font Schemes:font，Tabs，Caret...——Font——右边找到 FONT_SIZE=10，把 10 改的大一些，比如 12——右键单击左边的 Font——Load Script——成功。

接下来你可以开始编辑你的第一篇 TEX 文档了。

3.2 第一篇 TEX 文档

输入以下的内容。注意，一定要自己输，最好不要复制，如果不自己亲自输入，那么以后还得去复制，很难学会，只有自己输入了，才能更好的理解命令的含义。后面我会介绍每一个命令的含义。不过根据英文，可以大概理解它的大致含义了。

```
1 % !Mode::"TeX:UTF-8"
2 \documentclass{article}
3 \author{yourname}
```

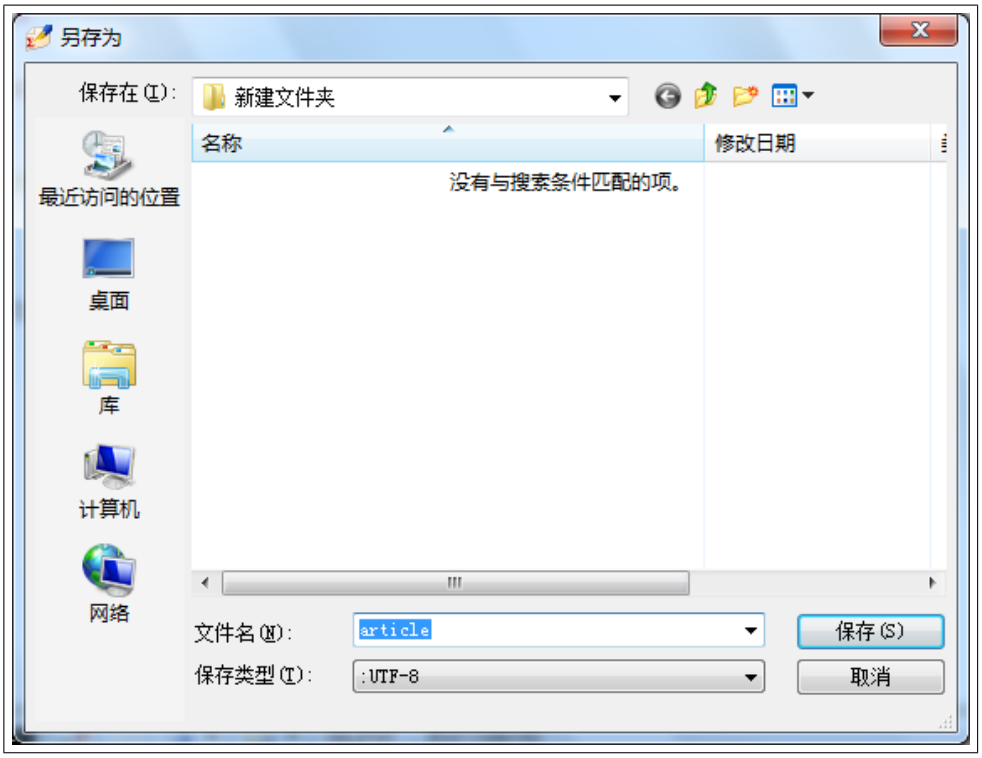

图 2 另存为对话框

```
4  \title{My First \LaTeX{} article}
5  \begin{document}
6    \maketitle
7    Wow! This is my FIRST \LaTeX{} Article!
8
9    Hello World!
10 \end{document}
```

然后保存这个文件，以后称它为**源文件**。这里有一点需要注意，点击保存后，会弹出另存为对话框，如 **图 2** 所示。在保存类型中一定一定一定

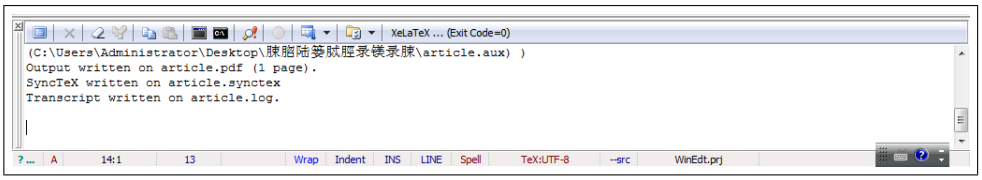

图 3 WinEdt Console

要选择 **UTF-8**（为什么是 UTF-8，其实前面解释过了，为了支持各国语言，XETEX 从底层支持 Unicode 编码，而 UTF-8 是 Unicode 的一种实现形式），并且强烈建议新建一个文件夹，把这个文件放在单独的一个文件夹里（原因稍后叙述），以后称这个目录为**工作目录**。保存完毕之后，单击前面说的 **图 1** 中紫色框中的第一个，即**编译**按钮，然后会发现 WinEdt 下面出现了一个小窗口（WinEdt Console），如 **图 3** 所示。这里面显示了编译的状态。如果你前面的操作没有错的话，会看到这个小窗口里最终会显示

```
.....
Output written on article.pdf (1 page).
SyncTeX written on article.synctex
Transcript written on article.log.
```

并且 **图 1** 中深红色框中第 4 个按钮，即**查看**按钮，变亮了（红色）. 点击它，会打开一个窗口，在这里你可以看到编译生成的文件，如 **图 4** 所示。

如果你没有看到输出的结果，在 WinEdt 下面的 Console 中显示了类似于下面的文字：

```
.....
! Paragraph ended before \begin was completed
<to be read again>
                \par
l.11

?
```

这就说明你上面的内容输错了。单击 Console 上面一栏的红色叉号，仔细检查后再进行编译。也许你现在没有遇到任何问题，但是将来必然会出现问题，除非你是神，或者你不再使用 TEX，所以如果你现在没有出错，为了看一下出错会发生什么，把你前面打的代码改错一下，比如删除一个 “}”，再编译，就会出现错误了。

如何检查错误呢？其实 WinEdt Console 里面已经告诉你大概是哪里出错了，比如上面 l.11 表示 11 行有问题，上面的文字说明了错误是 `! Paragraph ended before \begin was completed`。所以你只需要去检查 11 行就行了。11 行在哪里呢？总不能一行一行数吧，其实，观察一下 WinEdt 最下面一行的状态栏，是显示了行号的。在哪儿？自己找找看吧。另外，WinEdt 也是可以在代码的左边显示行号的。右键单击输入代码的区

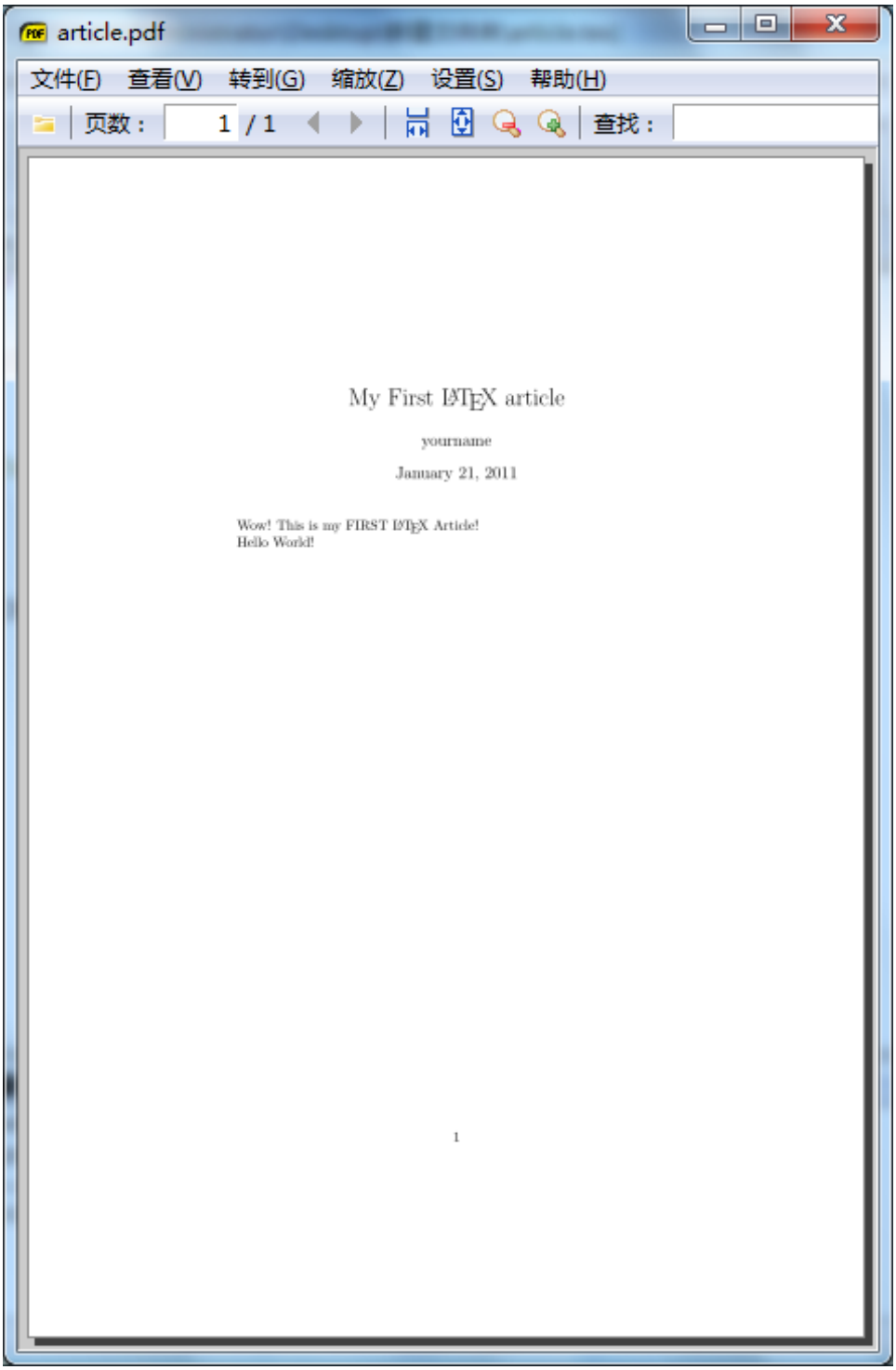
article.pdf
文件(F) 查看(V) 转到(G) 缩放(Z) 设置(S) 帮助(H)
页数： 1 / 1 查找：
My First LaTeX article
yourname
January 21, 2011
Wow! This is my FIRST LaTeX Article!
Hello World!
1

图 4 第一次编译的结果

域左边的空白处，选择 Show Line Numbers 就行了。

如果你已经完成了上面的操作，看到了生成的 PDF 文件，那么现在可以去看看**工作目录**，里面多了好多额外的文件，他们是什么呢？

```
article.aux
article.log
artilce.pdf
article.synctex
article.tex
```

你显然知道 article.tex 和 artilce.pdf 是什么，但另外的 3 个呢？不多说，自己 Google 吧。其实，还可能有很多其他的文件（aux、toc、bbl 等），现在你并没有很大的必要知道他们是什么。前面强烈建议把这个 article 文件保存到一个单独的文件夹里面，就是因为会有这么多文件生成，以免弄乱了，特别是喜欢把文件保存在桌面上的同学。

这些文件都是 TEX 编译过程中产生的，所以如果这些文件出错了，可能也会出现错误。这种情况往往发生在：第一次编译出错了，你找到了错误所在并改正了，但第二次编译刚刚开始就停下了。这时，你可以清除一下这些额外产生的文件，然后重新编译。有一个简单的方法来实现这个功能。看一看 **图 1** 中编译按钮旁边的那个垃圾箱的按钮。点点看，可以一键清除这些文件。

再回到刚才编译生成的 PDF 文件，点查看按钮，然后放大看一下，里面有**题目**：My First LATEX article，**作者**：yourname，还有**日期**：今天。另外还有两段话。注意是两段，不是两行。其实你应该可以理解刚才输入的是什么了，至少这两句话：

```
\author{yourname}
\title{My First \LaTeX{} article}
```

我想在这里，你最感兴趣的应该是 LATEX 这个符号了。它是内置的，我们一般写作可以 LaTeX。但是，通过 TEX，我们可以也应该更规范地写作 LATEX。为了满足你的好奇心，我现在告诉你我前面打过的所有类似于这样的符号该怎么输入，还有一些前面没有出现的。

首先，在`\documentclass{artile}` 下面插入三行

```
\usepackage{amsmath}
\usepackage{xltxtra}
\usepackage{mflogo,texnames}
```

然后再在 Hello World! 后面另起一行输入

```
\TeX{}, \LaTeXe{},
\XeTeX{}, \XeLaTeX{}, \AmSTeX{}, \AmS-\LaTeX{},
\BibTeX{}, \LuaTeX{}, Con\TeX{}t,
\METAFONT, \MF,          \MP
```

编译,**查看**，你会看到很多漂亮的 Logo. 像这样:

TeX, LaTeX 2_ε, XeTeX, XeLaTeX, $\mathcal{AMS}$-TeX, $\mathcal{AMS}$-LaTeX, BibTeX, LuaTeX, ConTeXt, METAFONT, METAFONT, METAPOST

3.3 LaTeX 的文档结构

到现在为止，你的源文件应该是这个样子的：

```
% !Mode::"TeX:UTF-8"

\documentclass{article}
\usepackage{amsmath}
\usepackage{xltxtra}
\usepackage{mflogo,texnames}

\author{yourname}
\title{My First \LaTeX{} article}

\begin{document}
  \maketitle
  Wow! This is my FIRST \LaTeX{} Article!

  Hello World!
  \XeTeX{}, \XeLaTeX{}, \AmSTeX{}, \AmS-\LaTeX{},
  \BibTeX{}, \LuaTeX{}, Con\TeX{}t,
  \METAFONT, \MF,          \MP
\end{document}
```

接下来对这个源文件做一个剖析。

第一行：

```
% !Mode::"TeX:UTF-8"
```

这句话的存在完全是由于 WinEdt 自身识别文档编码的能力不强所要求的。它为了使 WinEdt 能够识别你的这篇文章，正确显示所有的字符而不是乱码，也就是告诉 WinEdt，这是一篇 UTF-8 编码的 TeX 文档. 其实，TeX 是不识别这一句的，因为前面有一个 %，这是 TeX 注释的标记符，TeX 不执行注释的内容，如果你实在不想输入这句无用的话，那么在每次打开你的文件时，一定要注意选择文件类型为 UTF-8，类似于 **图 2** 所示的那样。这样，你就无法从目录里双击文件打开，而必须先打开 WinEdt，再从 WinEdt 里面打开你的文档了。所以多一句话可以减少一些麻烦。

这里要强调的是，应该学会利用注释。当我们修改一篇文档的时候，有些内容需要删除，不妨暂时把它**注释**掉，这样 TeX 便不再执行这一行。如果后悔了，只需再删除 % 就行了。这在查错的时候非常有用。

接下来的几行

```
\documentclass{article}
\usepackage{amsmath}
\usepackage{xltxtra}
\usepackage{mflogo,texnames}

\author{yourname}
\title{My First \LaTeX{} article}
```

即从 `\documentclass{article}` 到 `\begin{document}` 之前的部分，这一部分称为**导言部分**。导言部分里面定义文档的类型（documentclass），所使用的宏包（package）等等。也就是说这里放置这篇文档的**全局**性的设定，这里定义文档的一些通用的属性。所以现在这篇文档里的这几行字，除了括号里宏包的名字，的意思已经很清楚了，不再多做解释（除非你不认识上面的几个单词）。而需要强调的是格式，从上面的命令可以看出，LaTeX 的命令均以 `\` 开始，然后跟着命令的名字，`{ }` 里面是命令设置的内容。事实上有些命令还有一些可选的参数，放在 `[ ]` 里面，在后面的实践中将会遇到。

再接下来的内容:

```
\begin{document}
  \maketitle
  Wow! This is my FIRST \LaTeX{} Article!

  Hello World!
```

```
6   \XeTeX{}, \XeLaTeX{}, \AmSTeX{}, \AmS-\LaTeX{},
7   \BibTeX{}, \LuaTeX{}, Con\TeX{}t,
8   \METAFONT, \MF,          \MP
9 \end{document}
```

即包括 \begin{document} 和 \end{document} 以之间内容的部分，称之为**正文部分**。命令的英文也是容易理解的。其中 \maketitle 是生成标题的命令。正文的输入和使用 MS Word 没有什么区别，正常输入即可，不过在 MS Word 中，你可以直接看到效果，这里必须编译之后才能看到效果。

现在引入一个名词，叫做**环境**。凡是以 \begin{xxx} 开始以 \end{xxx} 结束的部分均称为**环境**。

需要注意的是，观察一下刚才编译生成的文件，只有两个段落，并且从第二段可以看出，首行是自动缩进的。为什么只有两段呢？TEX 分段的机制是什么呢？很容易观察出来，在 “Wow! This is my FIRST LATEX Article!” 之后空了一行，而那一堆 Logo 的命令之后没有空行，所以全连在一起了。由此，如果想要分段，只需要空一行就行了（两行呢? 自己试试），如果不空行的话 TEX 会认为是一段.

那么空格是如何处理的呢？从上面的代码中可以看出，\MF 和 \MP 之间有很多的空格, 但是 TEX 并没有识别它. 为什么呢？事实上，一篇正规的文档里是不允许出现非必要的空格的，即使是在 MS Word 里面。如果真要输入空格怎么办呢？试一试 \quad 命令，其实 TEX 里面定义了很多种的空格，自己 Google 吧！

3.4 WinEdt 的正、反向搜索及自动补全功能

选择 WinEdt 作为编辑器的一个好处就是它可以为我们提供很多的便利，例如正反向搜索和自动补全。下面一一介绍。

所谓**正向搜索**就是根据源码找 PDF 文件中的对应行（段落），要实现这个功能，你可以这样做，选中 WinEdt 中刚刚输入的 “Hello World!” 然后点击**查看**按钮右边的那个键。看看出现了什么情况。

所谓**反向搜索**就是根据 PDF 文件中的文字找源文件中的对应行（段），要实现这个功能，在打开的 PDF 文件上双击某一行，看看出现了什么情况。

自动补全就是让 WinEdt 为你补全你没有拼写完全的命令及单词，或者自动补全一个**环境**。如何实现呢？现在在刚才输的 \usepackage{amsmath}

和 \usepackage{xltxtra} 之间插入一行，输入 \usep，然后按 Ctrl+Enter，发生了什么？WInEdt 自动把它补全成了 \usepackage，这就是自动补全命令（单词）。现在你把它补全成这个样子

```
\usepackage{amssymb}
```

如何补全环境呢？把刚刚输入 \begin{document} 删了，尝试使用上面自动补全命令 (单词) 的方法输入 \begin{document}。然后再紧接着输入一个 }，发生了什么？WinEdt 自动把它补全成了

```
\begin{document}
  *
\end{document}
```

明白了吗？这是小技巧，以后使用过程中自然会熟悉的。那么现在把这个源文件重新整理一下，使他看起来像这个样子

```
% !Mode::"TeX:UTF-8"

\documentclass{article}
\usepackage{amsmath}
\usepackage{amssymb}

\author{yourname}
\title{My First \LaTeX{} article}

\begin{document}

  \maketitle

  Wow! This is my FIRST \LaTeX{} Article!

\end{document}
```

3.5 小结

现在，可以说你已经会使用 WinEdt 编写用 XeLaTeX 编译文档了。几个要点如下：

- 新建一个文件夹, 保存源文件为 UTF-8 格式;

- 点击**编译**按钮编译, 点击**查看**按钮查看;
- 文档分为两大部分,**导言**区和**正文**部分.
- 导言部分以定义文档类型`\documentclass`{article} 开始, 正文部分放在`\begin`{document} 和`\end`{document} 之间;
- 可以使用**正向搜索**以及**反向搜索**、**自动补全**.

好了，现在把你刚刚写好的文档整理一下（最好和上面的一样），进入下一个的章节！

4 章、节、段落

一篇文章要分为许多节，段落等等，像 Section 1，Section 1.2，Section 1.2.5 这样等等，如何实现呢？只需要像下面一样输入就好了。现在修改你的源文件，使它像这样（省略了了一些上一节展示过的内容）：

```
% !Mode::"TeX:UTF-8"
\documentclass{article}
... ...

\begin{document}
... ...
  \section{First Section}
  This is the first section
  \subsection{First subsection}
  I like the first subsection.
  \subsection{Second subsection}
  I don't like this subsection.
  \subsubsection{First subsubsection}
  \paragraph{1st paragraph}
  This is the first paragraph.
  \subparagraph{1st subparagraph}
  This is the first subparagraph.
  \subparagraph{2nd subparagraph}
  This is the second subparagraph.
  \section{Second Section}
\end{document}
```

编译并**查看**。试试反向搜索以及正向搜索。我想你应该知道怎么设置节了。那么什么是章呢？当然，应该是`\chapter{xxx}`, 另外还有`\part{xxx}`。然而，一篇文章（article）中是用不到章的，只有在书（book）等文档类中才用的到。现在学习 TEX 的目的是写文章，所以暂时就别管这些了。需要说明一下的是，前面已经说明了 TEX 对回车及空格有自己的处理机制，所以，为了使你的源文件看起来更加的有条理，你可以增加或者删除一些空行或者空格，以及注释。例如像这样

```
%=============First Section==================
\section{First Section}                        %
                                               %
This is first section.                         %
                                               %
... ...                                        %
%=============================================


%=============Second Section=================
\section{Second section}                       %
                                               %
This is the second section.                    %
                                               %
... ...                                        %
%=============================================

... ...
```

这里的段（paragraph）并不是通常意义上的自然段，可以成为**目**。只是当你为了突出某一段话（一个或多个自然段）的时候才可能用到它。分自然段的话只需要在两部分内容之间空一个（或多个）空行就行了。

无需小结，直接进入下一节。

5 数学

5.1 数学符号

还记得刚才在**导言部分**输入了一个命令：

```
\usepackage{amssymb}
```

它是什么意思呢？直接翻译就是 “\使用宏包{美国数学协会符号}”。所以，你应该先明白什么叫宏包，为什么要使用宏包，什么叫美国数学协会，什么叫符号！

事实上，前面说了，我们使用 LaTeX 做为我们的语言，就是为了充分利用它已经发展的很完善了，有大量的宏包可以使用，amssymb 就是其中之一。所谓宏包（package），就是一些重新定义了的命令的打包，以帮助你用更简单的命令实现复杂的功能。这样，宏包是什么和为什么使用宏包就貌似解释清楚了。接下来就是什么是美国数学协会（American Mathematics Society），这个嘛，自己 Google 吧，作为学数学的不知道这个就有些孤陋寡闻了。那什么是符号呢？这个需要解释吗？不过这里的符号侧重于指数学符号，而我们使用 amssymb 宏包的目的就是输入数学符号。更多的宏包将会在后面遇到。

通过上一节的学习，我们知道，正文是要放在**正文**环境里面的（\begin{document} 和 \end{document} 之间），加载（使用）宏包是要放在**导言部分**的（\begin{document} 之前的）。所以后面我可能不太强调把东西输在哪里，自己可以判断一下。

现在在 “Wow! This is my FIRST LaTeX Article!” 后面新建一**节**, 起名为 symbol （刚刚提到过如何分节），输入下面的内容，编译并查看。

```
1+1=2, $1+1=2$, I know 1+1=2, $I really know 1+1=2.$

This is in text mode, $This is in math mode,$ $This\ is\ in\ math\
    in\ mode.$
```

这里需要注意一下，在英文的输入习惯中，标点符号后面是需要加一个空格的，也就是说上面的每一个逗号后面都有一个空格。

输出的结果是大概是这个样子的:

1+1=2, $1+1=2$, I know 1+1=2, $Ireallyknow 1+1=2$.

This is in text mode, $Thisisinmathmode$, *This is in math in mode.*

仔细观察发现，放在美元符号（为啥不是人民币符号呢？）之间的无论数学内容还是文字内容都与不放在美元符号之间的不一样。

于是现在引入一个新的概念，叫做**数学模式（math mode）**。上面的放在两个美元构成的区域就叫做数学模式（也可以叫做数学环境）。在数学模式之中输入数学符号，在数学模式之外，称之为**文本模式（text mode）**

，输入文字。否则，在数学模式之中输入文字，全是连在一起的，并且是斜体（TEX 自动把它们当成了数学中的变量，所以用斜体），如果需要输入空格，需要用 “\␣”（前面3.3节最后提到自己 Google 空格的输入方法，你找到这个了吗？）。文本模式中输入数学符号看起来有些挤，更重要的是，文本模式中不可能输入更复杂的数学符号。最简单的例子，分数。除非你总喜欢把 $\frac{1}{x+1}$ 写作 $1/(x+1)$ 。

接下来需要明白的是另外两个概念，叫做**行内公式**和**行间公式**。没必要在这里定义它们，可以通过下面的例子认识它们。在刚才输入的内容后面再另起一段（就是空一行），输入以下内容，编译查看。

```
1 I know that you know $1+1=2$, but I know $2-1=1$, which you don't
    know. Now look at it $$2-1=1$$ I DO know more than you.
```

你会发现，放在双美元之间的 $2-1=1$ 另起一行居中了，然后紧接着是接下来的话。现在，你应该已经可以明白什么叫做行内公式和行间公式了。一个是在行的内部（inline equation），一个在两行之间（display equation）。

从 **表 1** 可以看出各种数学模式的标记符。

公式	行内公式	行间公式
方法一	$ ... $	$$ ... $$
方法二	\(... \)	\[... \]
方法三	\begin{math} ... \end{math}	\begin{displaymath} ... \end{displaymath}

表 1 数学模式

在这里做如下的推荐，行内公式使用 $ ... $，行间公式使用 \[... \]。

好了既然你现在知道该怎么输入符号了，现在就开始把常用的符号过一遍，这里不可能罗列所有的符号，也没有什么意义。可以参考

开始——**程序**——**ctex**——**help**——mathematics 和 symbol。

其实 WinEdt 也可以帮助你。点一下工具栏中 TeX GUI Symbols 按钮（**图 1** 中黄色方框中的倒数第二个）。另外，还有一篇很好很好的文档，叫做 Math Mode，见 [7]，介绍的相当的全面。还有两篇文章（事实上是一篇

文章，中文版和英文版）介绍了输入符号公式的一些细节内容，你可以在 [8] 中找到它们。下面开始常用的符号。

另起一段, 输入下面的内容, 编译查看.

```
$\frac{2011}{2012}, x_1,x_2,\ldots,x_n, a^2+b^2=c^2, x_1^2+x_2^2+\
    dots+x_n^2=r^{100}, \sqrt{x+1}, \sqrt[3]{x^2+1}$

$$\frac{2011}{2012}, x_1,x_2,\ldots,x_n, a^2+b^2=c^2, x_1^2+x_2^2+\
    dots+x_n^2=r^{100}, \sqrt{x+1}, \sqrt[3]{x^2+1}$$
```

效果如下

$\frac{2011}{2012}, x_1,x_2,\ldots,x_n, a^2+b^2=c^2, x_1^2+x_2^2+\cdots+x_n^2=r^{100}, \sqrt{x+1}, \sqrt[3]{x^2+1}$

$$\frac{2011}{2012}, x_1,x_2,\ldots,x_n, a^2+b^2=c^2, x_1^2+x_2^2+\cdots+x_n^2=r^{100}, \sqrt{x+1}, \sqrt[3]{x^2+1}$$

可以看出，同样的内容作为行间公式和行内公式显示的效果是不一样的，这完全是出于美观的考虑，使得行内公式的字体小一些，与文字有相同或相近的大小。另外，从上面的输入中你应该已经学会了输入**分式**、**根号**、**上标**、**下标**、**上下标**等等（当然你可能会想，上下标在左边该如何输入呢？自己到 Google 一下）。注意一下 r^{100} 怎么输入，100 是用花括号括起来的。如果没有花括号呢？试试看。

特别注意一点，前面说过，数学模式中式忽略空格的。另外，按照英文的习惯，以及美观的考虑，英文标点后面需要额外添加一个空格。然而，从上面的代码中可以看出，我没有这样做。原因是在这里只是为了展示一下如何输入各种符号。在实际的操作中，连续的公式应该这样输入：`$ a$,␣$b$,␣$c$`，而不是：`$a,␣b,␣c$`。另外，这样也有利于断行正确。

我们继续，另起一段输入下面的内容编译查看：

```
$\sin x, \cos x, \tan x, \arctan x, \sinh x, \cosh x, \max x, \min x
    , \ln x, \log x, \log_2 x.$
```

效果如下:

$\sin x, \cos x, \tan x, \arctan x, \sinh x, \cosh x, \max x, \min x, \ln x, \log x, \log_2 x.$

现在你应该知道该怎么输入**函数**了，为什呢函数名前面要加上一个 \ 呢？你会发现，上面编译效果为函数为正体，变量为斜体，这是为了区分变量与函数。当然你完全可以输入 `$sin x$`，但是效果呢？自己试试看。

继续另起一段输入以下内容并编译查看。

```
$a \in A, A \subset B, A \cap B, A \cup B, +\infty, \forall, \exists, f'(x), f''(x)$
```

效果如下：

$$a \in A, A \subset B, A \cap B, A \cup B, +\infty, \forall, \exists, f'(x), f''(x)$$

这是一些**集合**中常用的符号，以及一些**逻辑关系符**，等等。交和并的命令挺有意思，不是吗？不多说了，接下来是一些复杂的符号。

另起一段输入以下内容编译并查看：

```
$\lim_{n \to \infty} a_n = 1, \sum_{n=1}^{\infty} n = 5050, \int_{a}^{b}f(x) \mathrm{d}x = I$

$$\lim_{n \to \infty} a_n = 1, \sum_{n=1}^{\infty} n = 5050, \int_{a}^{b}f(x) \mathrm{d}x = I$$
```

效果如下：

$\lim_{n \to \infty} a_n = 1, \sum_{n=1}^{\infty} n = 5050, \int_{a}^{b}f(x) \mathrm{d}x = I$

$$\lim_{n \to \infty} a_n = 1, \sum_{n=1}^{\infty} n = 5050, \int_{a}^{b}f(x) \mathrm{d}x = I$$

我们又一次发现了行内公式于行间公式的差别，那么如果我非要使行内公式显示的和行间公式一样呢？给你两个关键词 `\displaystyle` 和 `\limits`，Google 一下吧（不过，不推荐在行内公式中使用 display 样式）！另外，从上面的公式之中，我们可以看出，复杂的公式仅仅是简单公式的组合。那么现在看看你会不会输入下面的两个式子呢？

$$\lim_{n \to +\infty} (1 + \frac{1}{n})^n = e, \int_{-\infty}^{+\infty} \frac{\sin x}{x} \mathrm{d}x = I.$$

试试看，如果你输入成功了，说明上面的东西你已经学会了。另外一个小问题是我输入了 `\mathrm{d}`，它是为了使 d 和 x 长的不一样，这在后面会介绍。这样其实是告诉 TEX，d 不是一个变量，它和 x 不一样。

再来些，另起一段，输入下面的内容，编译并查看：

```
$a \times b, c \div d, a<b , b=c, c \neq d, d > e, e \geq f, f \leq g, g \geqslant h, h \leqslant i$
```

效果如下:

$$a \times b, c \div d, a < b, b = c, c \neq d, d > e, e \geq f, f \leq g, g \geqslant h, h \leqslant i$$

这是一些简单的**四则运算**以及**大小关系**符号等。到此为止，你应该已经会输入一些数学符号了，但是还远远不够。上面的命令可能有些难记，其实想一想他们对应的英文，应该是很容易记忆的。

接下来的东西可能就需要记忆一些英文了。另起一段，输入下面的内容编译并查看：

```
1 $\alpha\beta\gamma\delta\epsilon\varepsilon\xi\pi\rho\sigma\eta\
    theta\phi\varphi\omega$
2
3 $\Gamma\Delta\Sigma\Phi$
```

效果是:

$$\alpha\beta\gamma\delta\epsilon\varepsilon\xi\pi\rho\sigma\eta\theta\phi\varphi\omega$$

$$\Gamma\Delta\Sigma\Phi$$

你可能已经发现了，输入大写的**希腊字母**只需要把首字母改成大写就行了，那么大写的 $\alpha\beta\varepsilon$ 呢？事实上，他们跟大写的英文字母 ABE 一模一样，所以 TEX 里面就没有定义这些字符。另外，需要注意一下的是，这里有两个字母有不同的写法，ϵ ε 和 ϕ φ。不同的人有不同的习惯，我们国内基本上都喜欢用后面的那个。

还有一些其他的东西，比如**向量**，**粗体**，**正体**，**空心字**什么的，试试下面的代码，另起一段输入

```
1 $|A|, \|A\|, \vec{a}, \overrightarrow{AB}, \tilde{x}, \widetilde{xyz
    }, \mathrm{sin}, \mathbb{RCZQ}, \mathbf{ABCD}$
```

效果是:

$$|A|, \|A\|, \vec{a}, \overrightarrow{AB}, \tilde{x}, \widetilde{xyz}, \mathrm{sin}, \mathbb{RCZQ}, \mathbf{ABCD}$$

我需要强调的是，还记得你刚才输入了命令 “\使用宏包{美国数学协会符号}” 吗。事实上，在使用这个宏包的同时，已经自动加载了另外的一个宏包，叫做 amsfonts。所以你不用再使用命令 “\使用宏包{美国数学协会字体}” 了（如果要用，实际命令是什么呢？放在那里呢？别告诉我你不会）。它定义了上面的空心字，如果没有这个宏包你是没办法使用这个字体

的。其实，这个宏包还定义了许多其他的东西，想知道吗？看一看我前面说的

开始——程序——ctex——help——mathematics

这篇文档吧。

下面开始一些更难的内容。比如分段函数该怎么输入呢？矩阵该怎么输入呢？一个大大的括号怎么输入呢？比如。像前面的公式

$$\lim_{n\to+\infty}(1+\frac{1}{n})^n=e$$

应该输入成

$$\lim_{n\to+\infty}\left(1+\frac{1}{n}\right)^n=e$$

该怎么办呢？

这样的一些内容将在下面为你解答，并且以这些内容结束这样的一个小节。

其实分段函数是基于矩阵的——一个只带半边花括号的矩阵。所以如果你知道怎么输入矩阵了，也就会输入分段函数了，也就会输入大括号了，同时，也应该会输入简单的表格了，不是吗？

先来一个简单的矩阵，另起一段，输入

```
\begin{equation}
  \left(
  \begin{array}{ccc}
  a_{11} & a_{12} & a_{13} \\
  a_{21} & a_{22} & a_{23} \\
  a_{31} & a_{32} & a_{33}
  \end{array}
  \right)
\end{equation}
```

你会得到

$$\left(\begin{array}{ccc}a_{11} & a_{12} & a_{13}\\ a_{21} & a_{22} & a_{23}\\ a_{31} & a_{32} & a_{33}\end{array}\right) \tag{1}$$

你看到了什么？矩阵的右边有一个编号 (1)。

为了告诉你这是怎么得到的，我一条一条地剖析如何得到这样的一个带编号的矩阵。

- `\begin{equation}` 和 `\end{equation}` 构成了一个**公式**环境。在这样的环境之中输入公式，是不需要再加美元符号的，并且默认是一个行间公式。它还带有编号，而且这个编号是自动的。自动编号的好处将在后面提及。
- 公式环境之中嵌套了一个 array 环境，它是用来输入矩阵的。
- `\left(` 和 `\right)`，是左括号和右括号。它们会根据括号内容自动变换大小。现在想想上面的

$$\lim_{n\to+\infty}\left(1+\frac{1}{n}\right)^n=e$$

  该怎么输入呢?
- 符号 & 是用来分隔列的, 即同一行不同列之间的元素要用 & 分隔开.
- {ccc} 是 array 环境的必选参数，即 center，center，center，当然还有 l 和 r 你可以自由组合。比如`{rcl}` 表示一共有三列，第一列向右对齐，第二列居中对齐，第三列靠左对齐。
- \\ 符号是强制换行的意思，上面的公式之中只在前两行的后面有\\，注意最后一行是没有的。这里就要提到一个前面没有提到的问题了，前面的一个问题是如何输入空格，那么 TEX 怎么输入一个回车呢？哦，原来是 \\. 这个命令的意思是强制换行，但是同空格一样，一篇规范的文档往往是用不到强制换行的，所有的回车都可以通过其他的排版设置来搞定，所以学习 TEX 不要总想着输入回车来换行。当然，你可以 Google 一下 TEX 到底该怎么输入回车，该怎么空行，你也可以自己做做实验。

通过上面的介绍，我想你应该会输入一个 4×4 的矩阵了（试试看吧），矩阵中的元素可以是前面输入讲到过的任何内容。但是请注意，前面说了，equation 环境已经构成了一个数学模式，所以你不需要再输入任何进入数学模式的标记符了。

再输入下面的代码（把上面的那个复制一遍，稍稍修改即可）

```
\begin{equation}
  \left\{
  \begin{array}{c||c|c}
  a_{11} & a_{12} & \\
  \hline
  a_{21} & & & a_{23} \\
   & a_{32} & a_{33}
  \end{array}
  \right)
\end{equation}
```

你会得到

$$\left\{\begin{array}{c||c|c} a_{11} & a_{12} & \\ \hline a_{21} & & a_{23} \\ & a_{32} & a_{33} \end{array}\right) \tag{2}$$

这是一个很奇怪的矩阵，需要注意一下几点：

- 花括号的输入是\{，自此引发的一系列问题是$，%，&，\ 等等该怎么输入。事实上，对于前三个可以在相应的符号前面加上一个 \，即 \$,\%,\&。但是第四个总不能还是加上 \ 吧，那样就变成强制换行了。实际上，它的命令是 \textbackslash，有些长。其实还有好多像这样的符号，被定义为了 TEX 语言的标记符，只能通过其他的方法输入。现在插入一个无关紧要的问题，双引号该怎么输入？左引号的命令是``（键盘上 1 旁边的那一个输入两次）, 右引号是''（回车旁边的哪一个输入两次）。现在你应该知道单引号是怎么输入了，试试吧。

- 矩阵中的元素是可以不输入的，但是 & 一定要有，并且与对齐的参数所定义的列数即 {ccc} 相呼应。事实上是不能多于对其参数所定义的列数。

- 这个例子中使用了 {c||c|c}，其实你已经看到效果了，两个 | 是两条竖线，一个是一条。横线的命令是 \hline。

- 左右两个括号是可以不一样的，甚至可以一个有一个没有，但是请注意，有了 \left 一定要有 \right。

下面的这个输入一个分段函数, 来结束本节内容。

另起一段，开始一个新的公式：

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

你会得到

$$\chi_A(x) = \begin{cases} 1, & x \in A \\ 0, & x \notin A \end{cases} \tag{3}$$

这就是充分利用矩阵的结果。我们得到了一个分段函数，里面涉及了 $\notin$ 的输入方式，类比一下，你还能得到什么呢？前面说了`\left` 和`\right` 一定要成对出现，但是我们确实不需要右边的括号，怎么办呢？`\right` 后面输入一个 “.” 就行了。

那么你一定还会问，更大的矩阵该怎么输入呢？分块矩阵该怎么输入呢？特别是分块矩阵中有空白，或者一个大大的 $\mathbf{0}$？这就是一个复杂的问题了，这里不再多说，刚开始学习 TEX 很少遇到这么复杂的东西，需要的话可以参考 [9]，这是一个相当复杂的例子，不必管它，除非真的用的到。

另外，需要注意的一点，矩阵中的元素在 TEX 看来都都是行内公式，前面已经提到了，行内公式和行间公式是不一样的，有时候行内公式看起来确实有一些 “小气” 了，前面说可以用 `\displaystyle` 命令来实现把行内公式写成行间公式的样子，但是如果一个很大的矩阵，总不能每一个元素都输上 `\displaystyle` 吧。这也可以通过其他方式实现。你可以参考一下 [10]。

5.2 行间公式

尽管上面的内容已经涉及了什么叫行内公式，什么叫行间公式，但是，上面的内容仅仅是一些符号的输入，并不是完整的公式。在这一小节里，

我们着重关注的是如何输入**行间公式**。其实，前面已经涉及到一个公式环境了，即 equation 环境，你可能会想到，我怎么一下子输入两个公式，并且都带编号呢？

直观的想，我有两种办法：

- 连续使用两个 equation 环境，就可以输入两个公式了，并且都带编号。但是问题出来了，如果两个式子我想要它们共用一个编号呢？想到了吗？对了，把两个公式输入成一个矩阵就行了。
- 在一个 equation 环境里面输完一个公式，然后输入强制换行\\。但是管用么？试试看。

其实，第一个方法尽管可以实现想要的效果，但是未免过于麻烦了一些。另外，如果我希望输入三个公式，公用一个编号，并且等号对齐，该怎么做呢？

回想一下，前面为了为入一些 Logo（还记得吗?），加载了一个宏包，做 amsmath，就是这一行

```
1 \usepackage{amsmath}
```

再直观的翻译一下，就是 “\使用宏包{美国数学协会数学}”，这个宏包为我们提供了一系列的数学环境，用来输入行间公式（当然远不止这些, 比如为了输入 $\mathcal{AMS}$ 其实已经用到了它）。比如这样的一个：

$$x = y \qquad X = Y \qquad a = b + c \tag{4}$$

$$x' = y' \qquad X' = Y' \qquad a' = b \tag{5}$$

$$x + x' = y + y' \qquad X + X' = Y + Y' \qquad a'b = c'b \tag{6}$$

再比如这样的一个:

$$\left.\begin{aligned} B' &= -\partial \times E, \\ E' &= \partial \times B - 4\pi j, \end{aligned}\right\} \quad \text{Maxwell' s equations}$$

在这里，我不会告诉你上面的内容到底该怎么输入，因为展开来说将会有无穷无尽的内容。在初学的过程中，你也完全可以避免排版这种复杂的东西。非得要排版出这样的东西的话，下面会告诉你如何自己学习。

这里要说的是，你可以想象得到的，所有你见过的东西，TEX 都是可以排版出来的。并且经过了多年的发展，LATEX 有大量的宏包可以使用，这

些宏包的使用给我们带来了无穷无尽的便利。当你在使用 TEX 的过程中，如果有东西不会打，一定要先 Google 一下，很可能会找到相应的宏包来处理这些内容。而且，所有的宏包都有对应的说明文档，里面有详细的说明。不过需要注意的是，说明文档一般都是英文的，如果你真的排斥英文，就真的很难进步了。

至于该如何找文档，其实只要你的电脑上安装了完整版的任何一个发行套装，文档就存在你的电脑上，在 WinEdt Console（前面已经说了它是什么，见 **图 3** ）里面找到  按钮，点一下，你会打开命令行窗口，在里面输入 texdoc␣xxx，回车，（其中 XXX 是你想要的宏包名），你就可以得到相应的文档了。或者，在 WinEdt 菜单栏点击**帮助** ——**LaTeX Doc**，输入宏包名，也可以查到文档。现在你可以尝试打开 amsmath 的说明文档了。另外你可以省事地直接参考 [11]。看了之后，就会学会很多行间公式的环境。

另外，如果你没有找到，最方便的方法便是上网去搜索了，一个网址是 www.ctan.org，怎么搜？自己看一看网页上的说明文字吧，还是那句话，如果排斥英文，你就很难进步了。

5.3 定理、证明

数学文档中常见的另外一类东西就是定义、引理、命题、定理、推论以及它们的证明，当然这些内容完全可以直接输入，就像你在 MS Word 中输入的那样。但是涉及的一个要点就是如何给定理编号。试想一下，手动地给仅仅几十个定义、引理、命题、定理、推论编号将会是多么复杂的一件事情，特别是有增添定理的时候。

在前面我们已经看到了 equation 环境可以自动编号。同样，定理环境也是可以自动编号的，下面需要加载一个宏包：amsthm。

在导言部分输入

```
1 \usepackage{amsthm}
```

什么意思呢？“\使用宏包{美国数学协会定理}”，我们一直在和美国数学协会（$\mathcal{AMS}$）打交道。它的文档在 [12]。

下面我们开始输入一些定理以及证明。在导言区输入下面的内容（在\usepackage{amsthm}之后的任何一行）

```
1 \newtheorem{definition}{Definition}
```

```
\newtheorem*{thmwn}{Thm}
\newtheorem{theorem}{Theorem!!!}[section]
\newtheorem{lemma}[theorem]{Lemma}
\newtheorem{proposition}{Proposition}[section]
\newtheorem{corollary}{Corollary}[theorem]
\newtheorem{example}{Example}
```

上面的命令看起来有一些复杂，因为它们的格式并不一样，先不管这些，稍后你就会明白了。

下面在正文部分开始新的一节, 像这样子的:

```
\section{section using amsthm}
\begin{definition}
  Definition is a environment for typing definition in \LaTeX{}.
\end{definition}

\begin{definition}[Theorem]
  A sentence is called a Theorem if and only if it satisfies \dots.
\end{definition}

\begin{thmwn}
  This is a Theorem without any numbers.
\end{thmwn}

\begin{lemma}
  This is a lemma.
\end{lemma}

\begin{theorem}
  This is the first theorem with automatic numbers.
\end{theorem}

\begin{proposition}
  This is a proposition.
\end{proposition}

\begin{proof}
  This is a brief proof
\end{proof}
```

```
29
30 \begin{theorem}[second one]
31   This is the second theorem with automatic numbers.
32 \end{theorem}
33
34 \begin{corollary}
35   This is a corollary.
36 \end{corollary}
37
38 \begin{proof}[My own Proof]
39   This is my own proof for the corollary.
40 \end{proof}
41
42 \begin{corollary}
43   This is also a lcorollary.
44 \end{corollary}
```

效果如 图 5 所示。

这一段的命令看起来比较长，但实际上很有规律，全都是一个一个的环境。为了明白上面命令的含义，有一下几个问题你应该注意到：

- 前面导言部分输入的命令定义了一些**定理环境**（当然可以不是定理，比如定义、推论什么的），使用的时候在正文中放在 `\begin{xxx}` 和 `\end{xxx}` 的 `{ }` 之中，即构成一个环境。环境的名字是 `\newtheorem` 后紧接着的花括号里面的东西（可以自己定义，但是不要于 TEX 已存在的命令名重复，特别是注意不要用 `\def`）。编译显示的结果是 `\newtheorem` 后面第二个花括号里面的名字（可以任意定义，也就是你在编译后的文稿中希望显示的内容）。**证明环境（proof）**不需要定义。

- 如果只有两个花括号定义的定理环境，编译的结果中的只有一个数字编号。并且这个编号系统贯穿整篇文档。就像 Definition。如果使用命令 `\newtheorem*{xxx}{xxxx}` 来定义定理环境，则不显示编号。

- 如果在第一个花括号后面有一个方括号，它的含义是该定理环境和方括号里面定义的环境使用一致的编号，不再重新编号。比如 `\newtheorem{lemma}[theorem]{Lemma}` 的意思是说 lemma 环境和 theorem 使用同一套编号，这从 图 5 可以看出来。

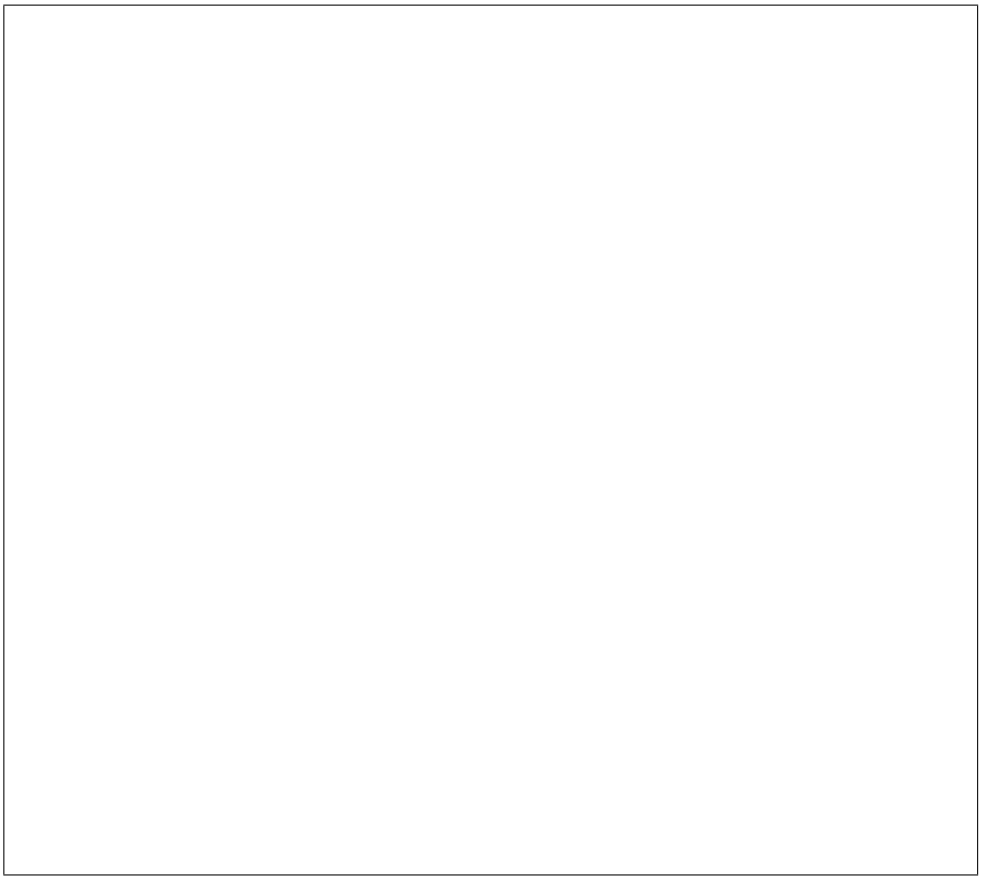

图 5 定理

- 如果在第二个花括号后面有一个方括号，它的含义是该定理从属于方括号里面的内容，编号方式为 “方括号内容的编号.此定理环境的编号” 就像上面的 **Theorem4.2**，4 的意思是第四节。这个方括号里面不仅可以放 section，也可以放入前面定义好的定理环境，就像 `\newtheorem{corollary}{Corollary}[theorem]`，它的意思是说推论是从属于定理的，所以编号为 **4.1.1**，多了一级。
- 如果在`\begin`xxx 的后面加上一个方括号, 可以产生一个注释, 编译后显示在定理后面. 这在一个定理有名字的时候经常用到. 在证明环境中也可以使用, 不过这时把 Proof 替换为方括号里的内容. 比如可以把 Proof 换成 Solution.

另外，为了输入定理，还可以用 `ntheorem` 宏包，自己查找一下说明文档看看吧。

5.4 小结

到现在为止，我们可以对上面的内容作一个总结了，事实上，这一节的内容是非常丰富的，该怎么总结一下呢？最好的方法是把现在的源文件和下面的对比一下。看看有没有设么遗漏的。然后看看都学到了什么。

你的文档大致应该是这个样子的。

```
% !Mode::"TeX:UTF-8"

%%%%%%%%%%%%%%%%%%%%%%这是导言部分的开始%%%%%%%%
%========== 导言部分声明文档的类型==================
\documentclass{article}

%==========导言部分可可以加载宏包==================
\usepackage{amsmath}
\usepackage{amssymb}
\usepackage{amsthm}

%==========导言部分配置宏包或定义命令以供全局使用=======
\newtheorem{definition}{Definition}
\newtheorem*{thmwn}{Theorem}
\newtheorem{theorem}{Theorem!!!}[section]
\newtheorem{lemma}[theorem]{Lemma}
\newtheorem{proposition}{Proposition}[section]
\newtheorem{corollary}{Corollary}[theorem]
\newtheorem{example}{Example}

%==========导言部分可以定义标题信息================
\author{yourname}
\title{My First \LaTeX{} article}

%%%%%%%%%%%%%%%%%%%%%%这是导言部分的结束%%%%%%%%%


%%%%%%%%%%%%%%%%%%%%%%这是正文部分的开始%%%%%%%%%
\begin{document}

%==========先生成标题============================
  \maketitle

%==========开始正文的输入========================
  Wow! This is my FIRST \LaTeX{} Article!

%==========这一段告诉你如何生成节标题等==========
  \section{First Section}
  This is the first  section
```

```
\subsection{First subsection}
I like the first subsection.
\subsection{Second subsection}
I don't like this subsection.
\subsubsection{First subsubsection}
\paragraph{1st paragraph}
This is the first paragraph.
\subparagraph{1st subparagraph}
This is the first subparagraph.
\subparagraph{2nd subparagraph}
This is the second subparagraph.
\section{Second Section}

%==========下面是如何输入数学符号====================
\section{Symbol}

1+1=2, $1+1=2$, I know 1+1=2, $I really know 1+1=2$

This is in text mode, $This is in math mode$, $This\ is\ in\ math\ in\ mode$

I know that you know $1+1=2$, but I know $2-1=1$, which you don't know. Now look at it $$2-1=1$$ I DO know more than you.

$\frac{2011}{2012}, x_1,x_2,\ldots,x_n, a^2+b^2=c^2, x_1^2+x_2^2+\dots+x_n^2=r^{100}, \sqrt{x+1}, \sqrt[3]{x^2+1}$

$$\frac{2011}{2012}, x_1,x_2,\ldots,x_n, a^2+b^2=c^2, x_1^2+x_2^2+\dots+x_n^2=r^{100}, \sqrt{x+1}, \sqrt[3]{x^2+1}$$

$\sin x, \cos x, \tan x, \arctan x, \sinh x, \cosh x, \max x, \min x, \ln x, \log x, \log_2 x.$

$a \in A, A \subset B, A \cap B, A \cup B, +\infty, \forall, \exists, f'(x), f''(x)$

$\lim_{n \to \infty} a_n = 1, \sum_{n=1}^{\infty} n = 5050, \int_{a}^{b}f(x) \mathrm{d}x = I$

$$\lim_{n \to \infty} a_n = 1, \sum_{n=1}^{\infty} n = 5050, \int_{a}^{b}f(x) \mathrm{d}x = I$$

$a \times b, c \div d, a<b , b=c, c > d, d \geq e, e \leq f, f \geqslant g, g \leqslant h$
$\alpha\beta\gamma\delta\epsilon\varepsilon\xi\pi\rho\sigma\eta\theta\phi\varphi\omega$

$\Gamma\Delta\Sigma\Phi$

$|A|, \|{A}, \vec{a}, \overrightarrow{AB}, \tilde{x}, \widetilde{xyz}, \mathrm{sin}, \mathbb{RCZQ}, \mathbf{A}$
```

```
81
82    \begin{equation}
83    \left(
84    \begin{array}{ccc}
85    a_{11} & a_{12} & a_{13} \\
86    a_{21} & a_{22} & a_{23} \\
87    a_{31} & a_{32} & a_{33}
88    \end{array}
89    \right)
90    \end{equation}
91
92    \begin{equation}
93      \left\{
94      \begin{array}{c||c|c}
95      a_{11} & a_{12} & \\
96      \hline
97      a_{21} & & a_{23} \\
98       & a_{32} & a_{33}
99      \end{array}
100     \right)
101   \end{equation}
102
103   \begin{equation}
104     \chi_A(x)=
105     \left\{
106     \begin{array}{ll}
107     1, & x \in A \\
108     0, & x \notin A
109     \end{array}
110     \right.
111   \end{equation}
112 %=========下面是如何输入定理及证明================
113   \section{section using amsthm}
114   \begin{definition}
115     Definition is a environment for typing definition in \LaTeX{}.
116   \end{definition}
117
118   \begin{definition}[Theorem]
119     A sentence is called a Theorem if and only if it satisfies \dots.
120   \end{definition}
121
122   \begin{thmwn}
123     This is a Theorem without any numbers.
124   \end{thmwn}
125
126   \begin{lemma}
127     This is a lemma.
128   \end{lemma}
```

```
129
130   \begin{theorem}
131     This is the first  theorem with automatic numbers.
132   \end{theorem}
133
134   \begin{proposition}
135     This is a proposition.
136   \end{proposition}
137
138   \begin{proof}
139     This is a brief proof
140   \end{proof}
141
142   \begin{theorem}[second one]
143     This is the second theorem with automatic numbers.
144   \end{theorem}
145
146   \begin{corollary}
147     This is a corollary.
148   \end{corollary}
149
150   \begin{proof}[My own Proof]
151     This is my own proof for the corollary.
152   \end{proof}
153
154   \begin{corollary}
155     This is also a lcorollary.
156   \end{corollary}
157
158 \end{document}
159 %%%%%%%%%%%%%%%%%%%%%%%%%这是正文部分的结束%%%%%%%%%%%%%
```

你可以对照一下，看看自己那些学会了，那些还没有学会，自己修改一些命令，编译并查看，看看能得到什么，会出现怎么样的错误，试着找出错误的原因并改正它。学习 TEX 的过程之中必然会经常遇到电脑报错，有时候是电脑错了，但绝大多数情况下，是你错了，仔细检查一下，改正错误。

最为一个概述，在这一节里面，你学会了

- 输入各种各样的数学符号，包括**分数，根号，上下标，函数，四则运算符，逻辑运算符，比较关系符，集合关系符，希腊字母，空心字母，粗体字母，矩阵**，矩阵的应用 (**分段函数**) 等等。
- 如何输入定义、引理、命题、定理、推论, 以及证明。

- 如何查找文档，通过文档自己学习 TEX（这才是最重要的!!!）。

现在你还会问一大堆问题，什么什么该怎么输，什么什么该怎么输，符号太多了，不可能也没必要罗列出来，遇到了自己查一查前面说过的那几篇文档就好了。

6 中文

你可能已经尝试过输入中文了，但是没能成功，对吗？

一开始就说了，TEX 本来不支持中文，我们选择 XETEX 的原因就是它从底层支持中文。但是为什么前面一直没有输入中文呢？事实上，为了输入中文，你需要先定义一下字体，这是很重要的。另外，中文文章中的段首缩进两个字符、标点符号使用习惯等等都和英文不太一样。所以，有必要加载一系列的宏包或者通过自己的定义来实现这些东西。这些宏包有多少呢？很多很多。

但是我们有一个简单的途径，下面就告诉你。

现在把`\documentclass{article}` 换成

```
1 \documentclass{ctexart}
```

ctexart 什么意思呢？是 ctex article。`ctex` 是中文 CTEX 小组开发的一个宏包，也就是说有人已经把该设置该加载的东西都提前为你准备好了，直接用就行了。现在试试看把 `\author{yourname}` 中换成你的中文名字，编译查看，是否显示中文了呢？

现在你可以尝试把你想要修改的东西都改成中文，比如输入一条中文的定理，中文的命题。

到此为止，你应该已经会输入中文了。甚至可以说，你已经可以完成一篇自己的文档了。试试看吧！

但是故事到这里还没有完，注意观察一下，ctexart 和 article 编译出来的文档看起来是不那么一样的, 如果我想要使用 article, 或者 amsart(没用过, 试试看. 就是 ams-article, 又见到 ams 了) 的文档类, 又要输入中文怎么办呢? 试试这个命令吧:

```
1 \usepackage{ctex}
```

应该知道把它放在哪儿吧？注意你现在的文档类应该是 article 或者 amsart。编译一下看看，是否能显示中文呢？

另外，你也可以使用 ctexcap 宏包，官方的说明文档上，它只支持 article，book，report 三个文档类，事实上，它能支持的更多（不过小心出错）。具体的区别你可以自己查看 ctex 的说明文档。

先别急，还有一点需要告诉你，现在仍然使用 ctexart 的文档类，注释掉\usepackage{amsmath} 这一行（怎么注释掉？前面提到过，命令前加上一个 %），编译，看看出现了什么？**出错了**。错误提示是：

```
1 Runaway argument?
2 \x@protect \[\protect \[ \@nil \@ifpackageloaded {amstex}{\def \
    @tempa \ETC.
3 ! Paragraph ended before \@tempa was complete.
4 <to be read again>
5                    \par
6 l.435 \newenvironment{proof}[1][\proofname]{\par
7
8 ?
```

你的文档哪里有 435 行呢？其实，这回不是你错了，是宏包出错了。LaTeX 语言有好多的宏包，是不同作者开发出来的，所以一起用难免会冲突，但常用的宏包冲突是极少的。现在告诉你这里为什么冲突了，你的文档使用的是 ctexart 文档类，相当于调用了 ctex 宏包，而 ctex 宏包又自动调用了 fontspec 宏包，而 fontspec 宏包和 amsthm 宏包有小冲突。现在你可能迷茫了。我怎么知道宏包冲不冲突呢？这基本上靠经验了，等到用熟练了，则然会判断错误的。不过还是放心，**宏包的冲突并不像你想想的那么多**，所以如果编译出错了，先看看是不是自己错了。

对于这个冲突，怎么解决呢？其实，把刚才注释掉的 \usepackage{amsmath} 还原就行了，也就是说，只要你预先加载了 amsthm 宏包，就可以避免这个冲突了。现在你可以做一个小实验，把文档类换成 article，注释掉 \usepackage{amsmath} 这一行，仅仅在 \usepackage{amsthm} 的前面加上 \usepackage{ctex} 编译查看，出错了，对吗？再把 \usepackage{ctex} 挪到 \usepackage{amsthm} 后面，编译查看，嘿嘿，通过了。

如果你希望自己配置中文的输入，你可以学习完9.1节之后，参考一下那里提到的几个宏包。

不多说了，进入下一节。

7 插图、表格以及交叉引用

7.1 插图

一个重要的问题是在书写数学文档的时候往往需要插图，事实上，在 TEX 里面插图是一件很简单的事情（想变得复杂的话也很复杂）。另外通过一些宏包，TEX 还具有画图功能。在这里只介绍插图。

首先，为了使插图更为方便，首先需要载入 graphicx（注意它比 graphics 先进，一些教程上提到用 graphics 宏包，你完全可以用 graphicx 替代）宏包：

```
\usepackage{graphicx}
```

要插图，就首先得有图，截屏得到一个，保存为 figure.jpg，或者找一张照片，或者用 MATLAB 画一个，保存为 figure.eps，当然推荐用后者，找到图片以后放到**工作目录**里面。这里提示一点，我们一般见到的图形都是**位图**，比如 jpg 格式的，但是，数学图形用**矢量图**来保存更好，比如 eps 格式的图形。好在那里可以自己 Google 一下，知道就算了。比如我现在用 MATLAB 画了一幅图，保存为了 figure.eps。

然后再正文部分输入:

```
\begin{figure}[h]
  \centering
  \includegraphics[width=0.5\textwidth,angle=60]{figure}
\end{figure}
```

编译并查看，你就会在你的文档里得到一个像 **图 6** 所示的图形。

还是分条告诉你上面命令的意思：

- 有一个新的环境，叫做 figure，它是一个**浮动体**。什么叫浮动体呢？也就是说它是漂浮不定的，你插入的图片不一定在你输入它的地方。TEX 会选择最合适的地方放置它。
- [h] 的含义是 here，它是 figure 环境的一个可选参数，如果不输入，默认是 [htbp]，也就是这儿、页面上部、页面下部、另起一页，TEX 会按照这顺序检测到底放在那里最美观。
- [h] 命令对 TEX 来说表示允许放置的位置是 here，同样，如果输入 [bp]，表示图片允许放置的位置是下面、另一页。但是 here 命令太强

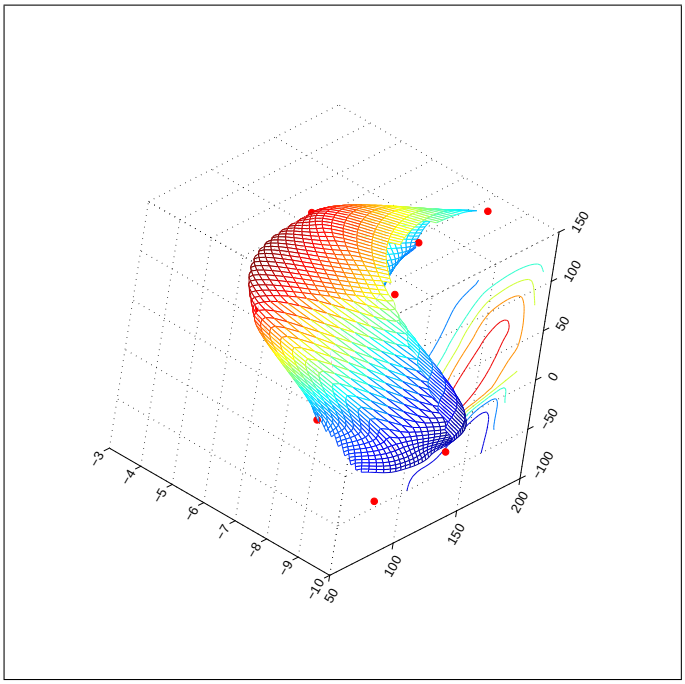

图 6

了，一般情况下，here 并放不下一张图片，或者并不“美观”。于是会造成，你输入了 [h]，TEX 并不一定把图形放在你输入它的地方。

- `\includegraphics[xxx]{xxx}` 是插图的命令，方括号里面是参数，用于控制图形打尺寸，旋转等等. 花括号里面是图像文件的名字，不需要扩展名，但是 XeTEX 支持的图像类型确实是有限的，常用的只有 jpg、eps 和 pdf，所以你不需要顾及太多。
- `\centering` 的意思是居中，即插图要居中，否则会靠左对齐。
- `\textwidth` 是文本宽度，到底什么是文本宽度呢？你可以从 **图 7** 中看出。

另外还有两个问题，如何给图片加标题且编号，如何插入一个不浮动的图形。

先回答第二个问题，很简单，把插图命令 `\includegraphics[xxx]{xxx}` 直接放在你想要的地方, 而不是单独的一个 figure 环境里面。

第一个问题也很简单，在插图命令后面加上`\caption{xxx}` 就行了，就像这样：

```
\begin{figure}[h]
  \includegraphics[width=0.5\textwidth,angle=60]{figure}
  \caption{这是标题}
\end{figure}
```

于是你便可以得到一个带标题的图形，而且有编号。如果你仅仅想有编号而没有标题的话，只需要空着\caption{xxx} 的括号里面的内容就行了。

等等，如果你你把一个\caption{xxx} 命令放在了一个不浮动的图形之后，就会**出错啦**。因为\caption{xxx} 命令只能输在一个浮动环境里面。但若你非要给一个不浮动的图形加标题，去看看 nonfloat 宏包吧！也可以参考 [13, 14]。另外 float 宏包提供了一个 [H] 选项，可以把图片排到 Here ！

如果希望更多地了解插图，可以参考刚刚提到的 [13]，这是英文版，中文翻译的是 [14]。

7.2 表格

现在开始输入表格，前面学习输入矩阵的时候就已经告诉你了，矩阵就相当于表格。但是有一点不知道你注意到没有，array 环境只能放在**数学模式**之中。我们知道，数学模式之中输入文字是有些麻烦的（前面说了该怎么输入，不会的话可以回顾一下前面的，也可以 Google 一下）。那么，有没有直接在**文本模式**中输入表格的方法呢？显然应该是有的。这个环境叫做 tabular。也就是说，你想输入一个表格，只需要这个样子：

```
\begin{tabular}{l|l}
Name & score \\
\hline
You & 100 \\
Me & 59
\end{tabular}
```

编译查看, 就得到了这个:

Name	score
You	100
Me	59

可以看出，它和 array 的输入方法是一模一样的。

另外一点重要的东西是，上面的插图有浮动的插图环境 figure，同样也有表格的浮动环境叫 table，放在 table 环境里的表格也可以很方便的添加标题并编号。例如像这个样子：

```
\begin{table}
\begin{tabular}{l|l}
Name & score \\
\hline
You & 100 \\
Me & 59
\end{tabular}
\caption{我的表格}
\end{table}
```

好了，基本内容说完了，可以进入下一部分。不过，可能这些内容对你来说太少了，你希望输入一个更复杂的表格，但是我在这里就不告诉你那么多了，希望了解的话可以看看 [15]。如果你希望对图片和表格的标题格式有所改进，可以使用 caption 宏包。具体就不再多说了。

7.3 交叉引用（cross-reference）

可能你之前没有听说过这个名词，但是你一定见到过它的身影，比如我这篇文章里面使用的 “如 **图 3** 所示，如公式(2)所示”。明白了吗？前面说给一大堆定理编号是一个头疼的事情，其实这个交叉引用更是一个令人头疼的事情，特别在修改文章的时候，如果编号不同步，将会造成引用的混乱。而 TEX 天生就可以解决这个简单的问题。

怎么做呢？找到你之前输入的一个放在 equation 环境中的公式，在公式的后面加上

```
\label{myequation}
```

myequation 可以是你希望的任何名字（英文），比如是这个样子的：

```
\begin{equation}
  \lim_{x \to 0}\frac{\sin x}{x}=1
  \label{myequation}
\end{equation}
```

然后你输入这样的一段话:

```
1 (\ref{myequation})式是一个很重要的极限.
```

编译**两遍**你看看发生了什么。`\ref{myequation}` 被自动替换成了 myequation 所标记的那个公式的编号。同样的道理，你可以给图片和表格以及定理和章节标题添加 label，也即是在你需要引用的东西的后面放上 `\label{xxx}` 命令，在后再你学要应用编号的地方使用 `\ref{xxx}` 即可。但是请注意，引用需要放在环境之内，而且图表的引用需要放在 `\caption` 命令的后面. 试试看吧！

另外一个需要注意的是，`\ref` 命令只会引用编号，即数字，不包括**图、表、定理、引理、公式**等文字内容。如果想要省些事儿，可以使用 `hyperref` 宏包提供的 `\autoref{xxx}` 进行引用，搜索一下它的说明文档吧。还有一个宏包，叫做 `cleveref`（最新的是 19/01/2011 修订的）, 你也可以试试看。

7.4 小结

在这一节里，你学会了以下的内容。

- 插入图片、表格. 插入图片是一条命令，插入表格是一个类似于 array 的环境。
- 认识了浮动体 figure 和 table。把插入图片和插入表格的命令放在相应的浮动环境之中，可以使得图片和表格放置在较为合理的位置，以免超出页边距。但有时也并不能把图片排到我们想要的位置。
- 给图片和表格加标题并引用。浮动环境中的图片和表格可以添加标题并且自动编号。注意`\label{xxx}` 命令一定要放在`\caption{xxx}` 的后面。另外可以使用 `caption` 宏包定制标题。
- 引用公式、定理、图片、表格等内容。先给相应的带编号的东西加上一个标签 `\label{xxx}`，然后再使用`\ref{xxx}` 引用相应的数字编号。

8 常用文档元素的实现: 摘要、参考文献 (GB/T 7714—2005) 等

请原谅我这一节讲的将不会那么详细，因为这一节里面的一些内容属于一些稍微高级的内容，刚入门的话不一定用的到。我着重介绍的是**参考文献**。

8.1 页面设置: 页眉、页脚、页码、双栏

设置页面的话，你应该首先了解一些页面布局。 **图 7** 展示了这个文档的页面布局。在你一开始使用 TEX 的时候或许已经注意到了，TEX 文档看起来很窄，而且，插图的时候你感觉明明可以放下一幅图，但 TEX 还是把图浮动到了另外的地方。这都是 Body 部分很小的原因。你当然可以改变它，也就是根据这幅图改变每一部分的宽度，得到你想要的效果。然而，这里并不推荐这样做，并且我不会告诉你修改的命令是什么。

如果你特别喜欢 MS Word 默认的页面布局，可以使用如下的命令：

```
1 \usepackage[top=2.54cm, bottom=2.54cm, left=3.18cm, right=3.18cm]{
    geometry}
```

把它放在导言部分，编译查看。你会发现，页面布局确实改变了。

要先实现双栏很容易，在文档类的命令处加上一个参数就行了，像这个样子：

```
1 \documentclass[twocolumn]{article}
```

这里想告诉你的一点是，文档类是可以加参数的，具体可以加那些参数，Google 一下吧。

另外，对于双栏排版有一个很纠结的问题。比如我现在在写一篇文章，使用双栏排版，但是我还有一张很宽的表格，或者一个很长的公式，不想断行，想要打通双栏，该怎么办呢？还有我希望题目摘要是单栏，正文是双栏，该怎么办呢？这都是可以解决的，可以试试 `multicol` 宏包等。

页眉、页脚、页码的设置可以利用 `facyhdr` 宏包，具体的用法可以见他的说明文档。下面举一个例子，你可以看一看。在导言部分输入下面的内容：

```
1 \usepackage{fancyhdr}
2 \pagestyle{fancy}
```

```
\lhead{abc}
\chead{\thesection}
\cfoot{\thepage}
```

上面的命令分别定义了页眉左边、中间以及页脚中间显示的内容，具体的用法可以参考它的文档，或者自己 Google。也可以懒省事的看看 [16]。这是一个中文的简要说明，看一下应该就会了。

如果想要得到 **M of N**（或者**第 M 页共 N 页**）这样的页码，其中 M 是当前页，N 是总共的页数，可以使用 `lastpage` 宏包。这一点在 `fancyhdr` 宏包的说明文档中也有提及。

8.2 列举

TEX 之中有多个列举的**环境**，itemize、enumerate 以及 description。他们的使用方法完全一样，就像下面所列出的

```
\begin{xxx}
  \item 这是第一条
  \item 这是第二条
  \item 这是第三条
\end{xxx}
```

明白了吗? 你可以分别将上面的 xxx 替换成对应的环境，赶快试试看都是怎样的效果。另外如果你希望对这个环境进行更精确地控制，可以参考这里 [17]。

8.3 摘要

摘要有专门的环境，叫做 abstract，这就是说，你只需要把摘要的内容放在 `\begin{abstract}` 和 `\end{abstract}` 之间就行了。

你可能在想，关键词、分类号、文献标识码等等东西该怎么输入，一般有些模板会提供相应的命令，否则的话，自己手敲吧。

The circle is at 1 inch from the top and left of the page. Dashed lines represent (`\hoffset + 1 inch`) and (`\voffset + 1 inch`) from the top and left of the page.

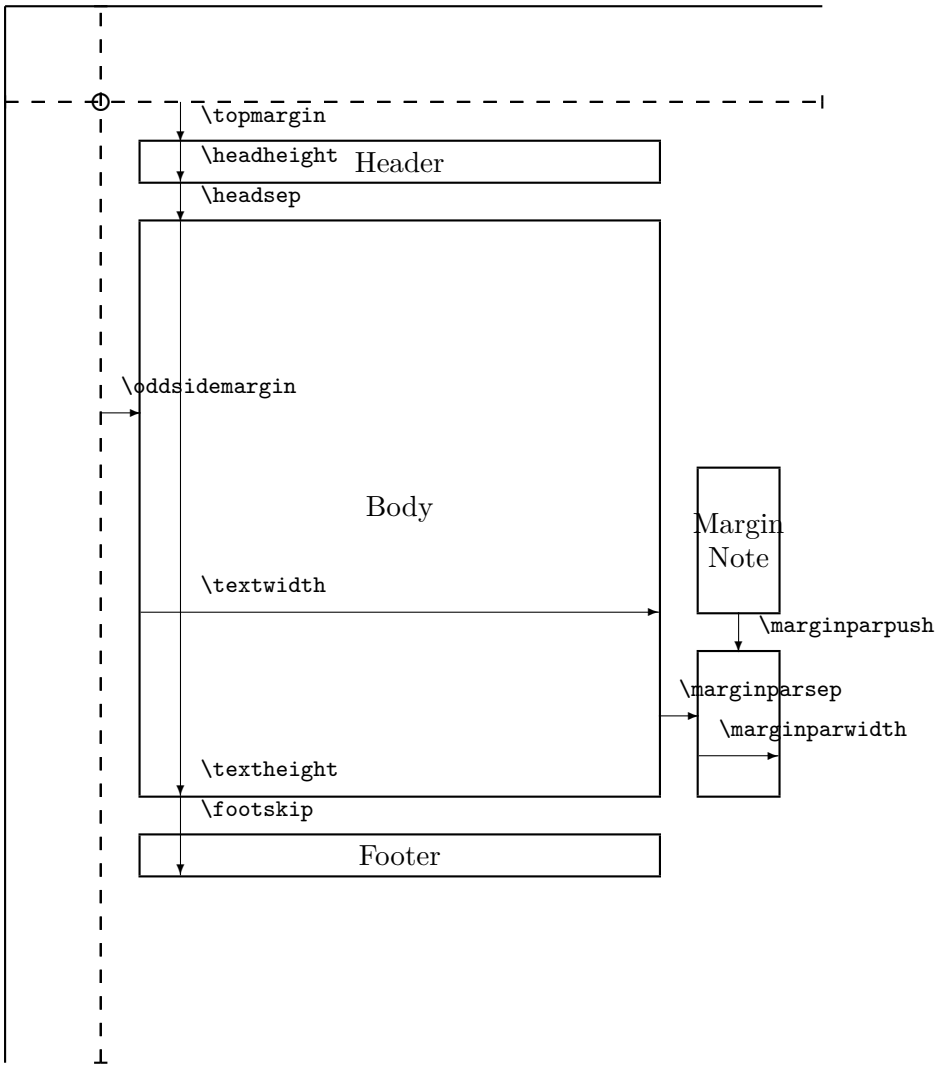

图 7 页面布局, 这里数值显示的是当前页面的页面布局, 不是默认值.

8.4 参考文献

这是这一节的关键内容，如何输入参考文献。首先你应该明白 GB/T 7714—2005 是个什么玩意儿。它是中华人民共和国国家标准中的《文后参考文献著录规则》。你可以上网搜一下它，快速浏览一遍，你会发现，它的要求特别特别的细致。能你已经觉得写参考文献是一个极为复杂的事情了，但是先别急，我在下面会详细地告诉你该怎么书写参考文献。

8.4.1 手工输入方法

首先，简单地来，如果你的参考文献只有一两条或者两三条，可以使用 thebibliography 环境手工输入。在你的文档末尾（当然得在正文环境以内了）输入下面的内容，编译**两遍**并查看（你现在可以想一下为什么是两遍，前面还有那些地方是两遍）。

```
1 As is stated in \cite{bibitem1} \dots
2 \begin{thebibliography}{9}
3   \bibitem{bibitem1} 大傻瓜.如何做一个合格的大傻瓜[M].傻瓜帝国:傻瓜出
      版社.2011.
4   \bibitem{bibitem2} 小傻瓜.如何成为一个大傻瓜[M].傻瓜帝国:傻瓜出版
      社.2011.
5 \end{thebibliography}
```

看到结果了吗？我还是分条叙述一下上面的要点：

- thebibliography 环境的使用方法和列举环境是类似的，但是列举环境中使用的是 `\item`，这里使用的是 `\bibitem`。
- {9} 的意思是最多可能有 9 条参考文献，这是 TEX 为了使序号能够对奇而要求的，因为一位数和两位数所占用的空间是不一样的。
- 每一个 `\bibitem` 后面还加了一个花括号，这相当于 `\label` 的作用，即给参考文献加上标签，方便引用。事实上，`\bibitem` 后面还可以加上一个方括号，也就是 `\bibitem[xxx]{xxx}` 这样的格式，你可以试试看。
- 引用参考文献的方法是 `\cite`，不是 `\ref`，并且 `\cite` 还引用了方括号，不仅是数字。

这样，其实你已经会输入参考文献了。但是当你面对大量的参考文献，这样的方法或许就有些麻烦了。

8.4.2 BibTeX 方法

下面要说的是一个高级的用法，在有大量的参考文献的时候使用起来特别方便。

还记得3.1节所说的那个 **B** 的按钮吗？就在编译按钮的右边。它是 BibTeX 的编译按钮。BibTeX 这个 Logo 还会输入吗？它是一个用于管理参考文献的系统。下面我要说的就是如何使用 BibTeX 来管理参考文献。

现在请新建一个空白文件，仍然保存为 UTF-8 格式，并且命名为 reference.bib，放在**工作目录**里面。

同样先输入一句没用的话，告诉 WinEdt 这是一个 UTF-8 编码的文档：

```
% !Mode::"BibTeX:UTF-8"
```

然后就可以让 WinEdt 帮助你做一些事情了。点击菜单栏：**Insert**——**BibTeX Items**。你会发现里面有很多的选项，先选择一个 article。然后现在你的文件就是这个样子的了：

```
% !Mode::"BibTeX:UTF-8"

@ARTICLE{*,
  AUTHOR =      {*},
  TITLE =       {*},
  JOURNAL =     {*},
  YEAR =        {*},
  volume =      {*},
  number =      {*},
  pages =       {*},
  month =       {*},
  note =        {*},
  abstract =    {*},
  keywords =    {*},
  source =      {*},
}
```

左边一列中，大写字母是必选项，小写字母是可填项，而 `@ARTICLE{` 后面的 `*` 的含义就像前面 `\label` 的内容，以及 `\bibitem` 后面花括号里面的内容，也是必填的。

现在不妨把必填项的 * 都替换成所需要的内容，选填项选填，不想填的话把对应的 * 删除，或者直接把该项删除（注意一下逗号），保存。类似于这样：

```
@ARTICLE{fool,
  AUTHOR =      {Little fool},
  TITLE =       {How to become a fool},
  JOURNAL =     {Research on the fool},
  YEAR =        {2011},
}
```

然后注释掉（删了也行）源文件中上一小节所输入的与参考文献有关的内容（选中，点右键，Insert Comment），再输入以下的内容：

```
\bibliographystyle{plain}
\nocite{*}\bibliography{reference}
```

然后按照按照如下顺序编译你的源文件。**注意是源文件**，不是 reference.bib：

XeLaTeX——BibTeX——XeLaTeX——XeLaTeX

BibTeX 的编译就是按一下那个 **B** 就好了。

现在查看一下，是否生成了参考文献呢？上面的几个命令第一个是定义参考文献的样式（Style），即使用 plain 样式。第三个是告诉 TeX 参考文献数据库是 reference.bib 文件。中间的那个是告诉 TeX 罗列出所有没有引用的文献，如果没有这句话的话，编译结果将不输出没有用 `\cite` 命令引用的文献。好了，你现在可以尝试**引用（cite）**一下刚才输入的参考文献了。

看下面的内容以前，你一定要确认上面的输入方法你已经成功了，不成功的话，仔细看看上面的每一步骤，然后再试。否则，后面的内容可能就有些难了。

也许聪明的你已经想到了，为了生成符合 GB/T 7714—2005 标准的参考文献，只需要定义一个参考文献样式（Style）就好了。事实上，这个工作已经有人做过了，见 [18]。所以你需要做的就是学会如何使用它。

那么你现在你需要做的是免费注册一个 bbs.ctex.org 的账号，然后下载 **GBT7714-2005.bst20100504.zip** 压缩包以及 **Winedt 的 Bib 模板.rar** 压缩包。当然你还可以下载两个说明文档。注意，这个论坛下载时无需积分的。事实上，你会经常用到这个论坛的。

第一个压缩包的文件目录如下：

```
GBT7714-2005.bst20100504
    |------ GBK
            |------ GBT7714-2005AYLang.bst
            |------ GBT7714-2005NLang.bst
    |------ UTF-8
            |------ GBT7714-2005AYLang-UTF8.bst
            |------ GBT7714-2005NLang-UTF8.bst
```

很显然，我们使用 UTF-8 编码，所以只有两个文件 GBT7714-2005AYLang-UTF8.bst 和 GBT7714-2005NLang-UTF8.bst 对我们有用。

先来看一看这两个文件的命名：

- 共同部分 GBT7714-2005 也就是国标的编号，也就是我们用的标准的名字。
- AY 是指 Author-Year，N 指 Number。分别对应 [大傻瓜,2011] 和 [1] 两种格式。
- UTF8 不用说。
- bst 是 bibtex-style 的缩写。其实，前面的使用过的 plain 用到了 plain.bst 文件。所以为了实现国标，确实是通过定义样式文件来完成的。你也可以编写自己的 bst 文件（当然, 这有些难度）。plain.bst 是任何 TEX 发行套装都带有的，但是刚才下载的那两个文件应该是任何发行套装都不包含的。所以如果使用它，仅仅安装了 TEX 的发行套装是不行的。如果需要用他们，你可以把它放在你的**工作目录**里面。当然你也可以它放在 TEX 的 TEXMF 和 LOCALTEXMF 里，他们是什么呢？可以到参考 [19]。简单起见，还是把它放在工作目录里面吧，这样无论把文件带到哪里都可以编译（因为换台电脑就不一定有这两个文件了）。

另外的一个压缩包的目录结构是这个样子的:

```
Bib
  |------ article.bib
  |------ book.bib
  |------ ......
  |------ ......
  |------ 使用说明
```

它们如何使用，自己看看**使用说明**吧（切记要备份原有的）。

下面我来告诉你如何编写符合 GB/T 7714-2005 标准的参考文献。两个准备工作：

- 把 GBT7714-2005AYLang-UTF8.bst 和 GBT7714-2005NLang-UTF8.bst 放到该放的位置（前面说过了）。
- 按照**使用说明**把模板放到该放的位置。

现在以 UTF-8 编码的方式打开刚才的 reference.bib 文件，注释掉（或删除）除了第一行（告诉 WinEdt 文件编码的那一句）以外的所有内容，**Insert——BibTeX Items——article**，你会在你的文件中看到以下的内容：

```
@ARTICLE{*,
  AUTHOR =     {*},
  normalauthor = {},
  TITLE =      {*},
  JOURNAL =    {*},
  YEAR =       {*},
... ...
... ...
  citedate =   {},
  url =        {},
  language =   {},
}
```

和刚才的不一样了，这就说明你的模板安装正确了。否则，再看看**使用说明**，看看哪里出错了。这次，凡是带有 * 号的都是必填的（如果同时出现 AUTHOR 和 EDITOR 两项（比如 Book 类），任选一个填写，删了另一个的 * 即可）。填写的方法和之前一模一样。特别注意别忘了 Cite_key，就是 `@ARTICLE{` 后面的那个 *。不明白每一项的含义的话自己 Google，最直接的方法是看 [18] 中的两个说明文档。现在需要注意的是，最后一行是 language 域，如果你的参考文献是中文的话，一定要在里面填上 Chinese（其实填什么都行，别空着），英文就不用了。

然后再插入一条文献，比如 Book 类的，填上必填的内容，最好和上面的不一样，一个是英文的，一个是中文的。填好以后保存。

然后打开你的源文件，把相应的行改成

参考文献

[大傻瓜 (2011)] 大傻瓜. 如何做一个合格的大傻瓜 [J]. 傻瓜研究, 2011.

[fool(2011)] FOOL L. How to become a fool[J]. Research on the fool, 2011.

图 8 参考文献

```
1 \bibliographystyle{GBT7714-2005NLang-UTF8}
```

然后按照前面所说的编译顺序进行编译。看看结果是什么。我得到了如 **图 8** 所示的参考文献，我插入了两条 article 类的参考文献，一个英文，一个中文。

注意，我这里使用了 ctexart 的文档类，如果你使用了 article 的文档类，显示结果稍稍有所不同，但是参考文献的格式应该是完全一样的。

到这里，我想你可能已经试用过 GBT7714-2005AYLang-UTF8 的参考文献样式了，但是你发现出错了，对吗？没试过的话赶紧试试看。

其实你需要一个宏包来帮助你，叫做 `natbib`。你可以在 [20] 看到它的一个非常简要的说明文档。

加载上它，可以使用下面的命令：

```
1 \usepackage{natbib}
```

然后再按正确的编译方法试试看。编译成功了吗？

细心的你可能已经发现了，在用 BibTeX 编译的过程中，WinEdt Console 中显示了很多的错误。而且在使用 GBT7714-2005NLang-UTF8 的时候，参考文献的格式显示的并不是前面所说的 Number 样式的，而是 Author-Year 样式的，为什么呢？

先说说错误时怎么回事，如果你仔细看了 [18] 的话, 会发现这个 bst 文件是一个测试版。它经过了反复的修改，尽管可以实现所有的样式效果了，但是源文件变得有些乱了，所以就出现了一些错误。如果你有兴趣修改的话可以试一试，帖子中的回复有人提到了如何修改。想告诉你的是，如果你得到你想要的输入效果了，任何错误提示都不是你的原因造成的，但是如果你没有得到你想要的效果，错误就在你自己了。

再说说样式显示不正确的问题。事实上，这两个 bst 文件都得结合 `natbib` 宏包使用才能显示正确的效果。

如果你使用 GBT7714-2005NLang-UTF8 样式，你需要这样

```
1 \usepackage[numbers,sort&compress]{natbib}
```

加载 natbib 宏包，就得到你所需要的样式，后面的那个参数自己看 [20] 吧。

如果你使用 GBT7714-2005AYLang-UTF8 样式，需要这样

```
1 \usepackage[authoryear,sort&compress]{natbib}
```

调用 natbib 宏包。由于默认参数是 authoryear，所以你可以省略这里的 authoryear。

通过这两种方式，便可以得到你想要的效果。就像我这篇文章里的参考文献一样。

在最后，推荐一个可用于管理参考文献的软件，可以使试试它：JabRef reference manager。哪里有，自己找找看吧。

我想到这里，参考文献的事情可以告一段落了，其实还有好多现成的 bst 文件，你可以看看 [21]。

8.5 小结

在这一节里，我主要介绍了参考文献的输入。其他的东西说的都不多，但是大都介绍了一些宏包，如果需要可以自己去查看文档。其实这一点在学习 TEX 的过程中是非常重要的，你应该学会如何去找到你想要的宏包，并且学会如何去使用它们。你可以按照这个方式去查找：首先使用 Google 去搜索，最好以这样的关键词：“xxx␣LaTeX” 去搜索，其中 xxx 是你想要实现的功能。一般情况下，google 会把你带到 ctex 论坛。如果你在某些地方看到了你想要的宏包的名称，就到 http://www.ctan.org 去搜索它的说明文档，通过看文档目录去寻找你想要实现功能所在的位置。然后去看一下它的例子，进行模仿。一般情况下，你是没有必要完整阅读一篇文档的。你可以在 [22] 看一看高手们都使用那些宏包。

刚开始学习 TEX 一定会遇到很多麻烦，因为使用它和使用 MS Word 太不一样了。不过切记，如果你非要实现各种各样花哨的排版效果，还是用 Word 吧。当你学习了 TEX 之后，你一定会发现，你在使用 MS Word 的时候，几乎就没有任何排版的概念。

在最后我解释一下为什么编译参考文献这么麻烦，需要四次编译。

事实上，第一次编译生成了一个 aux 文件，然后 BibTeX 利用它去编译 bib 文件，从而生成 bbl 文件。然后 XeLaTeX 利用 bbl 文件来排版参考文献，再一次编译是为了使交叉引用编译正确。前面为了使公式图表的引用正确也是这个原因，只是少了前两个环节。XeLaTeX 直接处理源文件便使得所用需要编号的内容编号正确，而第二遍编译是为了使引用的地方显示正确。

前面说到了 bbl 文件，你可以以 UTF-8 格式打开它。你会发现，里面的内容是一个完整的 thebibliography 环境，使用 BibTeX 的作用就是让电脑根据参考文献的要素自动生成符合要求格式的代码。如果你对自动生成的参考文献不甚满意，可以微调 bbl 文件里的内容。

9 杂项 (包括字体设置)

这一节里面我会处理一下在这篇文档中我使用过的，你见过的一些效果是如何实现的。

9.1 字体配置

先说说 XeTeX 的字体设置，我为什么把它放在杂项里面，是因为这个设置暂时可以不去理他，因为有了 `ctex` 宏包，你真的没有必要去设置字体。

XeTeX 的发明一方面是为了解决各国语言的输入问题，另外一方面就是充分利用已存在的各种字体。最早，TeX 用的都是高德纳自己设计的字母，当然，英文字母很少，所以可以这么做。各国语言那么多，而且系统里面已经安装了那么多字体，为什么不用一下呢？当然，这里在强调一点，绝大部分字体是有版权的（收费的）！另外，TeX 设计的初衷不是为了让你把一篇文章整的像广告一样花哨，当然它完全可以实现，但我们学数学的不需要。

为了设置字体，你先看一下 `X:\CTEX\MiKTeX\tex\latex\ctex\fontset` 文件夹里面的 `ctex-xecjk-winfonts.def` 文件, 用记事本或者 WinEdt 打开它 (找不到? 搜索一下)。看看里面的命令。

你应该看不懂这些命令，但是你会发现，它只用到了六种字体。也是我们最常用的一些字体，当然你可以修改它们，但最好先别这样做。我不会告诉你该如何设置字体，因为暂时没有任何必要，想了解字体的话，可以

先看一看专门研究字体的大侠 K M C 的入门级著作 about:fonts，参见 [23]。非常简短的一篇介绍，有了前面的基础，你可以迅速入门。

不过，我前面（6节）已经提到了，为了输入中文，你可以参考这里，我不能什么也不告诉。事实上，为了详细了解一下中文的配置，你需要参考三个文档，分别是 xeCJK、fontspec 和 ctex 宏包的说明。其中，第一个和第三个是中国人开发的，所以文档时中文的，看起来比较顺，只有第二个的文档时英文的。当你看完了这三个宏包的文档之后，你就可以理解中文的配置了。

9.2 超链接、网址

你可能已经注意到了，我在文章中添加了很多的超链接，而且目录、交叉引用都是超链接形式的。它是利用 hyperref 宏包实现的，我在前面提到过它，不是吗？输入网址也是用它。可以参考 [24, 25]。都是中文的文档。

9.3 插入代码

你一定会纳闷，在源文件里面输入的命令都让 TEX 给编译了，我是如何输入的呢？其实，我使用了 listings 宏包。它是一个专门排版代码的宏包，支持几十种语言代码的排版，并且可以实现代码加行号，代码加框，关键词高亮等功能。看看他的说明文档吧，其实网上有好多中文文献介绍它。

不过有一点纠结的地方，它对中文的支持不是很好，你可以试试找到你一个解决办法。

现在注意，不使用 listing 宏包也是可以在文章中插入代码的。LATEX 原生提供了 \verb 命令以及 verbatim 环境。可以用来输入代码。另外，还有一个 verbatim 宏包可供使用。

这里仅仅举一个例子说明，正文部分输入以下的内容并查看。

```
1 \verb|\begin{document}|
2
3 \begin{verbatim}
4   在这个环境中所有的命令都不起作用。\end{document}
5 \end{verbatim}
```

需要注意的是，我使用了 | 而不是 { } 作为分隔符。你还可以使用其他的字符，比如 ~, !, @, =, $, ^, & 等等。

9.4 还有...

还有一些，比如这篇文档的标题页下面的**脚注**是如何实现的。我的代码是：

```
\author{hy\_haoyun \thanks{E-mail: hy\_haoyun@bbs.ctex.org}}
\title{\Huge{\XeLaTeX{}及WinEdt 6.0入门指南}}
\date{\today}
```

看明白了吗?

过如何给正文中添加脚注，你得自己搜索了。

两外的一个东西是**目录**，其实目录的生成也是很简单的，只需要在你想要生成目录的地方输入

```
\tableofcontents
```

编译**两遍**就行了，这样就会自动为你生成目录。如果你需要对目录的格式进行设置，可以使用 `titletoc` 宏包。

如果你希望对章节标题进行设置，可以使用 `titlesec` 宏包。

还有哪些呢？我再想想，稍后补充。不过最直接的方法是，充分利用**反向搜索**功能，看看我是怎么实现各种效果的。

10 幻灯片（Slides, NOT PPT!）

下面就是一个崭新的、崭崭新的章节了。你可能会奇怪，TEX 还能做幻灯片？是的，它能，并且可以做的很专业，很漂亮。不过有一个概念你必须搞清楚，幻灯片不是 PPT，PPT 只是用 MS PowerPoint 做出的一种幻灯片。用 TEX 做出的不叫 PPT！你可以叫它幻灯片，或者 Slides，或者 Beamer，对了就是 Beamer。我们就是用 Beamer 来做幻灯片。

10.1 Beamer 是什么

Beamer 是一个 LATEX 文档类，就像 article 和 ctexart 一样。不过制作幻灯片的文档类不仅仅有 Beamer 一个，还有常见的 PDFScreen 和 prosper。国内大部分人用 PDFScreen，不过用的多并不代表它很好，现在 Beamer 应该才是王道。

如果你学会了上面的内容，你应该知道该怎么书写一篇 Beamer 文档，其实它也是分为导言部分以及正文部分，导言部分的内容和正文部分的内

容基本一样，正文部分的写法和 article 文档的写法也基本一样。这个一样是说：分章节（section,subsection）、输入行内行间公式、列举环境、插图、插入表格、目录等等都是一样的。不一样的是：

- Beamer 中图、表、定理默认是没有自动编号的（可以设置），为什么？任何看你做报告的人基本不可能记得起你给那个定理编上了号，哪个图片是图 4, 哪个表格是表 3。听众的注意力并不在编号上。
- Beamer 中可以通过 columns 环境实现分栏，原因很简单，如果没有这样一个简单的环境进行分栏，一张幻灯片那么大的空白利用率可能不高。
- Beamer 中不是自动分页的，你需要把你认为该放在一张幻灯片里的内容放在一个 frame 环境里（千万别太充实了）。并且可以给每一个 frame，也就是每一张幻灯片添加一个标题。

10.2 如何使用 Beamer

我其实本来不想多说如何使用 Beamer，实际上，网上有很多的文档介绍该如何用 beamer 制作幻灯片。不过有一点，网上绝大部分的方法都是基于 PDFLaTeX 以及 CJK 宏包的。不能说过时，但是我们可以更简单。直接加载 `ctex` 宏包，然后便能够像编译英文一样直接编译中文了。那么你需要注意的就是，你在看网上其它的文档时，完全可以忽略所有与 CJK 相关的内容。

我先举一个例子。新建一个文件，以 UTF-8 的编码保存到新的（旧的也行）工作目录下，起名为 beamer.tex（注意扩展名还是 tex）：

```
1  % !Mode::"TeX:UTF-8"
2
3  \documentclass{beamer}
4  \usepackage{ctex}
5
6
7  \title{我的Beamer}
8  \author{name}
9  \institute{my institute}
10
11 \begin{document}
```

```
12   \titlepage
13 \end{document}
```

编译（还是之前的 XeLaTeX 编译）并查看，是不是生成了一个带有中文的页面呢？如 图 9 所示。

图 9 Beamer 标题页面

解释一下上面的命令。没有见到过的有两个：`\institute` 和 `\titlepage`。前面一个是 Beamer 特有的一个标题信息，用于生成作者单位（多作者该怎么办？Google 下看）。后面的一个相当于 `\maketitle`。事实上, 完全可以使用 `\maketitle`，可以得到同样的效果。这里使用 `\titlepage` 的目的是为了告诉你，这里生成的标题是占一页的，叫做标题页。

接下来你可以编写你的 beamer 文稿了。下面仍然是举例子（此处省略了导言部分）：

```
1 \begin{document}
2 \titlepage
3
4 \section{第一节}
```

```
5   \begin{frame}
6    \frametitle{1.第一张幻灯片}
7    \[\exp{i \theta} = \cos \theta + i \sin \theta \]
8   \end{frame}
9
10  \begin{frame}
11   \frametitle{2.第二张幻灯片}
12   This is the second frame.
13  \end{frame}
14 \end{document}
```

这就是书写 beamer 的格式，把每一张幻灯片的内容放在一个 frame 环境里面。这个环境里面任何东西的输入方法和普通的文档没有区别。

我想可以结束了，在最后我列举几样东西，这是你需要掌握的，而我实在不想多言了。

- 如果你细心的话，你会发现，这里的数学字体和之前见到的不太一样。也许你更喜欢这个样子的，也许你不喜欢. 如果想要换回 TEX 标准的样式的话，在导言区添加命令 `\usefonttheme{professionalfonts}`。
- 你可以尝试在导言区放上 `\usetheme{Warsaw}` 命令，编译看一下效果。这是一个很常用的主题，你也可以使用其他的主题，有怎样的主题可用可以在 10.3 中提到的任何一篇文档里找到。
- 还有一个常用的 block 环境，你可以参考我将在后面列出的文档。

10.3 小结

这一节是在没有什么可以总结的, 不过我想可以给你推荐几个文档, 这样更有助于你制作一个漂亮的幻灯片.

- 首先是一个韩国人 Ki-Joo Kim 写的文档 Beamer v3.0 Guide，强烈推荐这一个，简单明了。英文文档见 [26]，中文翻译见 [27]，中文的可能需要是教育网才能访问，你也可以根据题目搜索一下，自己下载。
- 和第一篇一样，是直接用 Beamer 写的幻灯片，看起来很省事。Charles T. Batts 写的 [28]，不过没有中文版。

- beamer 自己的说明文档，前面已经告诉你怎么查找文档了，所以就不再多说了。

为了更好的学习 beamer，强烈建议看看前两篇文档，很短，一会就可以搞定，但收获一定是很大的。

11 后记

连续写了 5 天（2011-1-21—2011-1-25），再加上两天的修改（2011-1-26—2011-1-27），这篇文档终于完工了。尽管修改了一遍，但错误应该还有很多，但是应该不影响阅读了。稍后的错误我会在以后修订。暂时可以给它一个版本号，Version 1.0。希望大家在这个假期里抽出一段时间来学会它。并且使用它做出一个漂亮的幻灯片，开学后做报告。另外，祝大家假期愉快！

2011-1-27

2011-1-27 把这篇文档上传到了 bbs.ctex.org，受到了很多朋友特别是 LiYanrui 和 LeoLiu 的批评指正。所以又对这篇文档做了小小的修改。升级为 Version 1.09。

2011/2/1

2011-3-17，论坛上的一些朋友 silverriver、孤云残剑又发现了一些小错误，我也发现了几个，一并修改了。升级为 Version 1.099。

2011/3/17

2011-9-20，换邮箱为 haoyun_tex@163.com。稍微更改版本号为 Version 1.0991。

2011/9/20

参考文献

[1] CTEX.ORG. 关于 tex 版本的问题 [EB/OL].[2010/1/26]. http://bbs.ctex.org/viewthread.php?tid=49647.

[2] INSTANTON. 关于 TeX 编辑器 (1)[EB/OL].[2010/1/26]. http://blog.163.com/soft_share@126/blog/static/4298360320077284423611 5/.

[3] INSTANTON. 关于 TeX 编辑器 (2)[EB/OL].[2010/1/26]. http://blog.163.com/soft_share@126/blog/static/4298360320077285261749 3/.

[4] INSTANTON. 关于 TeX 编辑器 (3)[EB/OL].[2010/1/26]. http://blog.163.com/soft_share@126/blog/static/4298360320077301025393 90/.

[5] OETIKER T, PARTL H, HYNA I, et al. The Not So Short Introduction ti LaTeX 2_ε[M/OL]. http://www.ctan.org/tex-archive/info/lshort/english/lshort.pdf.

[6] 中文 TEX 学会. 一份不太简短的LaTeX 2_ε介绍 [T/OL]. http://www.ctan.org/tex-archive/info/lshort/chinese/lshort-zh-cn.pdf.

[7] VOß H. Math Mode[M/OL]. http://www.tex.ac.uk/tex-archive/info/math/voss/mathmode/Mathmode.pdf.

[8] CTEX.ORG. 如何精调文字与公式 [EB/OL].[2010/1/26]. http://bbs.ctex.org/viewthread.php?tid=28224.

[9] CHINATEX. LaTeX 技巧心得 291：怎样绘制分块矩阵的竖虚线 [EB/OL]. http://blog.sina.com.cn/s/blog_5e16f1770100h7uc.html.

[10] CHINATEX. LaTex 技巧心得 219：如何在矩阵中统一字体 [EB/OL]. http://blog.sina.com.cn/s/blog_5e16f1770100gvge.html.

[11] SOCIETY A M. User’s Guide for the amsmath Package[M/OL]. ftp://ftp.ams.org/pub/tex/doc/amsmath/amsldoc.pdf.

[12] SOCIETY A M. Using the amsthm Package[M/OL]. ftp://ftp.ams.org/pub/tex/doc/amsmath/amsldoc.pdf.

[13] RECKDAHL K. Using Imported Graphics in LaTeX 2ε[M/OL]. http://tex.loria.fr/graph-pack/epslatex.pdf.

[14] 王磊. LaTeX 2ε插图指南 [T/OL]. http://www.ctex.org/documents/latex/graphics/.

[15] 汤银才. LaTeX 中表格的制作 [M/OL]. http://70.40.198.72/latexstudio/books/LaTeXtable.pdf.

[16] CTEX.ORG. CTEX - 在线文档 - TeX/LaTeX 常用宏包 -fancyhdr[EB/OL].[2010/1/26]. http://www.ctex.org/documents/packages/layout/fancyhdr.htm.

[17] CTEXWIKI. LaTeX/列表 [EB/OL]. http://wiki.ctex.org/index.php/LaTeX/%E5%88%97%E8%A1%A8.

[18] CTEX.ORG. [中文处理] 发布 GBT7714-2005.bst version1 Beta 版 [EB/OL].[2010/1/26]. http://bbs.ctex.org/viewthread.php?tid=33591.

[19] COHOMO. LaTeX 之 TeX 系统目录结构 [EB/OL]. http://cohomo.blogbus.com/logs/7039075.html.

[20] MERKEL S. Reference sheet for natbib usage[EB/OL]. http://merkel.zoneo.net/Latex/natbib.php.

[21] TURNER K. BibTeX Style Examples[EB/OL]. http://www.cs.stir.ac.uk/~kjt/software/latex/showbst.html.

[22] CTEX.ORG. [TeX@China] 统计：我使用的宏包 [EB/OL].[2010/1/26]. http://bbs.ctex.org/viewthread.php?tid=49115.

[23] CTEX.ORG. [原创]XeTeX about:fonts[EB/OL]. http://bbs.ctex.org/viewthread.php?tid=43244.

[24] 王广民. hyperref 的应用 [M/OL]. http://math.ecnu.edu.cn/~latex/docs/packages/hyperref_chs.pdf.

[25] CTEXWIKI. Hyperref[EB/OL]. http://wiki.ctex.org/index.php/Hyperref.

[26] KIM K J. Beamer v3.0 Guide[M/OL]. http://faq.ktug.or.kr/wiki/uploads/beamer_guide.pdf.

[27] 黄旭华. Beamer v3.0 指南 [T/OL]. http://math.ecnu.edu.cn/~latex/beamer/beamer_guide_cn.pdf.

[28] BATTS C T. A Beamer Tutorial in Beamer[M/OL]. http://www.uncg.edu/cmp/reu/presentations/Charles%20Batts%20-%20Beamer%20Tutorial.pdf.