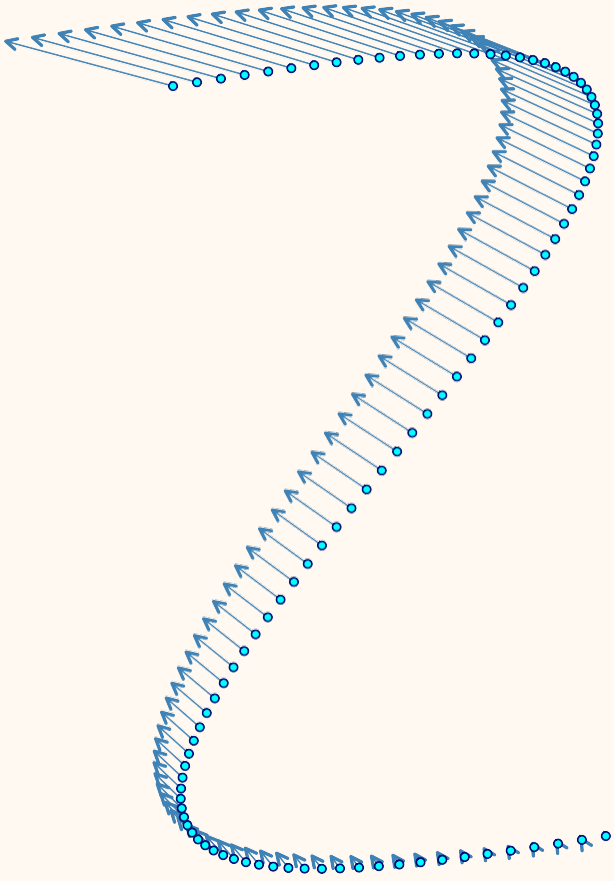
# 大家来学 LaTeX

Version 1.0

2004 年 3 月 8 日



By Edward G.J. Lee 李果正

Email ： edt1023@info.sayya.org

本教学文件之制作，部份接受下列计划补助：

行政院研考会委办 「政府机关数据文件交换之电子文件格式应用研究」

# 目 录

# 第 1 章

# 先来说一些故事

TEX 是 Donald E. Knuth[1] 教授的精心杰作，它是个功能非常强大的幕后排版系统，含有弹性很大，而且很低阶的排版语言。当初，是因为 Knuth 教授在写他的大着 TAOCP(*The Art of Computer Programming*) 时，发觉书商把他书中的数学式子排得太难看了，于是决定自行开发一个非常适合排数学式子的排版语言，这就是 TEX 系统的来由。

不仅仅是谈到 TEX 一定会提到 Knuth 教授，只要提到排版，没有人可以忽略他的 TEX 所带来的变革、影响，甚至 TEX 已经 20 几岁了，仍然深深影响着排版界及排版软件，可见这个排版软件真的是非同小可。

## 1.1 Knuth 教授的生平简介

| | |
|---|---|
| 1938.01.10 | 诞生。Milwaukee, Wisconsin; U.S. citizen。 |
| 1956 | 进入凯思工学院（Case Institute of Technology）学习物理。 |
| 1960 | 毕业后进入加州理工学院（California Institute of Technology）研究所，此时转向数学领域的研究方向。 |
| 1961.06.24 | 和 Nancy Jill Carter 结婚。他的中文名字是高德纳，他老婆名叫高精兰，老婆小他一岁。两个小孩，一男一女。中文名字是 1977 去中国大陆时取的。 |
| 1963 | 拿到 Ph.D.，并留校任教。 |
| 1968 | 开始任教于 Stanford 大学，信息科学系(Computer Science)。同年开始撰写名闻遐迩的 TAOCP(*The Art of Computer Programming*)。有人曾说，看了这部书，往后对写程序的话题都会变得谦虚。:) |
| 1977 | 不满意书商所印出的 TAOCP，因此，自行开发 TEX 排版系统，这可就影响了往后的排版、出版界，至今不坠。但也因此拖延了 TAOCP 第四册的完成时间。 |
| 1986 | 荣获 ACM 软件系统奖。 |

[1] http://www-cs-faculty.stanford.edu/~knuth/

他可说是著作等身，书籍、论文都有，他的任何著作有个奇怪的『副作用』，那就是任何 人发现书上的错误，都可以向他举发，并领取 $2.56（美金）！想试试看「手气」吗？台湾就有人领过。:-) 为什么是 $2.56？Knuth 教授的答案是：

> *“256 pennies is one hexadecimal dollar.”*

发现 TEX 系统的臭虫也是一样，这个奖金每年倍数增加，直至 $655.36($2^{16}$ pennies) 为止。

他也很推崇自由软件基金会 (Free Software Foundation)[2] 及 GNU/Linux，把一些希望都寄托于 GNU/Linux，尤其是 Unicode 环境，他希望 GNU/Linux 很快就能在网页上显示他的中文名字，而不必使用图档。其实是可以做到了，只是 Unicode 环境还不算普及罢了。他曾在 1996 年接受 Dr. Dobb’s Journal 访问时英雄惜英雄的公开表示，创导自由软件 (Free Software)[3] 的 Richard M. Stallman 是他心目中的英雄之一，他认为自由软件基金会这些人所做的贡献很不错，虽然和他的方式不一定一样，但许多贡献是互有认同的。

在发展 TEX 时就同时思考 WEB（这个词比目前使用的 WWW Web 还早使用），那是一种 literate programming 的程序方法。他认为目前已成熟的可以提出含有文件的程序方法，使写程序就像写文学作品（小说？）一般的艺术表现。后来也把他由 C 改写（和Silvio Levy 合作），名为 CWEB[4]。TEX 就是由 WEB 写成的，WEB 可视为 Pascal 语言的一个子集。

一个 literate 程序员可被视为文学作家、评论写作者、随笔作者……，程序的表现不仅仅是搬弄符号，而是展现自己的风格，当然也是指达成目的的风格、甚至程序中变量运用的风格。

这样一来就可以展现让人类较能理解的程序代码。使用形式及非形式的融合，而且两者间相辅相成，目的达成了，也让阅读的人就好像读文学作品般的去抓住作者的心，使程序创作提升至更高的（文学）艺术境界，而不再是死板板的 code 了。

Knuth 大师已于 1992 年自大学退休，但仍在 Stanford/Oxford 等大学有授课。目前正处于隐居的生活，他这么早退休的原因，就是因为 TAOCP 这部书，他估计大约要花 20 年来完成，因此目前的重点工作是完成他的 TAOCP（分成好几册，目前真正出版的只有三大册）。他认为 email[5] 会影响他的思路，因此，宁愿留住址，要和他联络就只好写信，传

[2] http://www.fsf.org
[3] http://www.gnu.org/philosophy/free-sw.html
[4] http://www-cs-faculty.stanford.edu/~knuth/cweb.html
[5] http://www-cs-faculty.stanford.edu/~knuth/email.html

真。给他的秘书的 email，是最后有时间才会去看，他曾公开的表明，这部书是他这一生中最重要的工作。

虽然 Knuth 教授写了许多严肃艰深的书籍、论文，但是他也是有风趣的一面，在 1996 年，*Mathematisch Centrum* (MC, 为庆祝五十周年庆改称为 *Centrum voor Wiskunde en Informatica*, CWI) 曾邀请他演讲，并知会荷兰的 TeX User Group(NTG)[6]，NTG 见机会难得，就邀请 Knuth 教授另开个 TeX 及 METAFONT 讨论会，并接受大家的提问，他说：『不对，我也是可以问你们问题的！』。而且，他还说：『这种问答的内容，很可能在不同场合重复过，所以，如果我对同一个问题，曾有过两种答案的话，你们必需取其平均值。』他的妙语如珠，在许多类似的场合常常引起哄堂大笑，但实际的内容却绝非泛泛之言。:-)

### 1.1.1 TAOCP(*The Art of Computer Programming*)

这可是算法的大着，请别折腾我，我只是心向往之，这部书我没有一本是看得懂的。:-)

1. 第一册，*Fundamental Algorithms*，1968 第一版，ISBN 0-201-89683-4。
2. 第二册，*Seminumerical Algorithms*，1969 第一版，ISBN 0-201-89684-2。
3. 第三册，*Sorting and Searching*，1973 第一版，ISBN 0-201-89685-0。
4. 第四册，*Combinatorial Algorithms*，尚未完成，可能会先出分册。
   (a) 分册 4A, *Enumeration and Backtracking*
   (b) 分册 4B, *Graph and Network Algorithms*
   (c) 分册 4C, *Optimization and Recursion*
5. 第五册，*Syntactic Algorithms*，计划 2010 年完成。
6. 第六册，the theory of context-free languages，书名可能会变更，尚未开始。
7. 第七册，Compiler techniques，书名可能会变更，尚未开始。

详细的目录大纲及修订版的信息，请参考网页上的数据：

http://www-cs-staff.stanford.edu/~knuth/taocp.html

## 1.2 那么，LaTeX 又是什么呢？

TeX是个很低阶的排版语言，如果排版时都要从这种低级语言来控制版面的话，那将会非常的烦复，所以，一些经常要用到的功能，都会先去定义好（称为宏，macro），这样

[6]http://www.ntg.nl，荷兰的 TeX User Group 算是相当活跃的。

排版时才会方便、快速，直接引用已定义好的宏里头的指令就可以了。

原始的 TeX 已经有了一组 macro，是 Knuth 教授所写的，那就是著名的 Plain TeX，但仍然不够方便、直观，于是 Leslie Lamport[7] 又写了另一组的 macro，称为 LaTeX，主要是把版面配置和文章内容，适度的分开处理，只要使用者选定了一种类别，整本书或整篇 文章的结构就是按照这个类别来安排版面，这样写文件的人只要专注于文章内容就可以了，版面配置就完全交给 TeX/LaTeX 去处理。

既然 LaTeX 只不过是 TeX 的一大组宏，那，当然原来的 TeX 的指令，大部份也是可以用在 LaTeX 文稿当中的。而且，LaTeX 并不是目前唯一的 TeX macro，其他如 eplain TeX, ConTeXt, TeXinfo 等都是 TeX macro，也都有他们自成一套的语法。目前的 LaTeX 由 LaTeX 3 Project[8] 所维护及发展。

如果谈到这里，你还是雾煞煞的话，模拟成 HTML markup 标记语言就能大概知道一些概念了，当然，这和 HTML 标记语言是可相提而无法并论的。如果连 HTML 也不熟悉，那也没关系啦！这章本来就是在说故事嘛！:-) 只要继续看后面的内容就行了。

## 1.3 一般人对 LaTeX 的迷思

这里引用 Peter Flynn 在他的 *A beginner's introduction to typesetting with LaTeX* [9] 一文中所提出来的六大迷思，并添加个人的一些看法：

- LaTeX 只能使用一种字型？
  当然不是，虽然 TeX 系统默认是使用当初 Knuth 教授所设计的 METAFONT，但目前 OpenType, TrueType, Adobe Type1 字型都可以用在 LaTeX 当中，只不过，安装字型的部份不是那么的直观就是了，但比起其他的排版系统，TeX/LaTeX 所能利用的字型种类，可以说是最多的。

- LaTeX 只能用于 Unix-like 的操作系统？
  Knuth 教授慷慨的把 TeX 的源代码开放出来[10]， 所以， 只要是有人在使

[7] Leslie Lamport 于 1985 在 CSL 任职时写了 LaTeX。目前任职于 DEC Systems Research Center 但已几乎没有参与 LaTeX 的后续发展了。

[8] http://www.latex-project.org/latex3.html

[9] ftp://ftp.dante.de/tex-archive/info/beginlatex/beginlatex.pdf

[10] 虽然有一些版权的规定，但是 Knuth 教授的本意，这些原始码是属于 Public Domain，只是，如果经过修改，不能再以 TeX 为名，要另外改一个名称，以免和原来的 TeX 搞混。TeX 这个商标的所有人是美国数学协会（American Mathematical Society, AMS）。

用的操作系统都可以移植过去，不必担心版权的问题。像 MS DOS, OS/2, MS Windows, Mac OS, Unix-like 系统都有 TeX 的移植版本，甚至是 PDA（e.g. the Sharp *Zaurus*）都有 TeX/LaTeX 的纵迹。可以说是走到哪儿，就可以用到哪儿，而且，文稿都是互通的，打印结果也相同。

- LaTeX 已经过时了？
  刚好相反，LaTeX Project[11] 及其他相关 packages 正稳定的研发当中，尤其和目前新一代的 SGML/XML/HTML 及数据库系统，都积极的想办法衔接起来。对于数学式子的排版，至今无人能出其右。有兴趣的话，可以参考 news://comp.text.tex 的流量，及其中 CTAN[12] 对于宏更新、上传的消息发布。

- LaTeX 没有所见即所得（WYSIWYG）？
  某种层面上而言， 是的。因为他本质上是幕后排版系统。但是， 所产生的 dvi/ps/pdf 文件，可以很精准的显示你所想要表达的内容，不晓得这算不算是「所见即所得」？另外，一些相关 GUI 配合的软件，也会打破幕后排版的定义，例如 LyX[13], TeXmacs[14] 等等。

- LaTeX 很难学？
  这个嘛！我只能说，有什么东西是很好学的？如果只是一般使用，而不是当个排版专家，甚至 TeX/LaTeX programmer，那么，几十分钟的说明，就可以「上工」了！剩下的只是一些细部调整的问题（不去调整，也绝对不会离谱）。相对于一般图形 化 Office 类软件，要真正把他的内容熟悉，恐怕也是不简单的。另外的问题，大概是幕前、幕后系统操作习惯的问题，甚至是一种第一印象了，就好像说到计算机，有 不少人的脑子里就会浮现 MS Windows 的图象一样。:-)

- LaTeX 是专为科学家或数学家而写的？
  的确，当初 Knuth 教授是为了表达精确、质量优美的数学式子而开发 TeX 的，但由于 TeX 的弹性，使得在其他的领域的使用者也争相使用，已经不是局限在学术界在使用了。尤其 XML 的兴起，需要一个适合的格式化工具（formatter）的配合，TeX/LaTeX 就刚好称职的做他排版专业的工作。

[11] http://www.latex-project.org/
[12] http://tug.ctan.org/
[13] http://www.lyx.org/
[14] http://www.texmacs.org/

## 1.4 本文的重点方向

TEX/LATEX 的相关议题：版面的配置、排版规范、字型技术、TEX the program 的改进、绘图技术、TEX macro 的撰写、pre/post 处理程序的撰写、出版流程· · · · · 等等，浩瀚无涯，可以说穷一辈子也可能研究不完，所以，各位在「陷入」这个领域之前，建议最好有个适当的范围，以免「愈陷愈深」终至无法自拔。

所以，本文的重点是放在「标准」LATEX 本身，其他相关的 package/macro 除非必要，尽量不提及。但是 LATEX 本身也不是十全十美的，所以，有需要的地方也需要一并提及外来的 macro，无论如何，重点是放在标准 LATEX 本身，请阅读本文的朋友注意一下这个方向。LATEX 本身就能解决的，就不假外求了，虽然会失去了一些弹性，但为免造成 package/macro 满天飞，打乱学习阵脚，刚开始也只好如此了！

而且，可能的情形下，会往一般用途的方向去介绍，而不仅仅专注在数理排版。数学式子虽是 TEX/LATEX 的拿手把戏，但不代表一般用途就不适合，相反的，现在有许多商业公司正把 TEX/LATEX 用于一般商业出版上。

# 第 2 章

# 行前准备

使用 TEX/LATEX 系统，刚开始，比较麻烦的是安装问题。不过，以现在的操作系统而言，几乎较流行的操作系统都有现成包好的 TEX 系统套件可以安装，例如 Un*x 系统的
teTEX、Windows 系统的 MiKTEX 及 fpTEX。另外，也有 TEX Live CD[1] 可以供下载、购买，这是 TUG(TEX User Group)[2] 制作的各种操作系统的可执行文件，使用上相当方便。

目前所谓的 TEX 套件，是把原来的 TEX 排版引擎本身，加上 LATEX 及其他相关的宏，再加上字型软件（METAFONT）、绘图程序（METAPOST）、字型文件· · · · · 等等，所组合成的一整个可实际运作的排版系统。因此，什么是 TEX？会因使用的场合不同而有不同的意义，一般纯指指令本身的时候，就单纯写成小写的 tex，此时所用的宏，默认就是 Knuth 教授所写的 plain TEX。写成 TEX 时，一般是指整个系统而言。这在 LATEX 宏亦同，latex 指的是指令，LATEX 指的是整个宏系统。

## 2.1 Unix-like 系统

一般 Unix-like 系统都是安装 teTEX 套件，凡是和 tetex 字样相关的 packages 都安装起来，目前 GNU/Linux 各种 distribution 及 FreeBSD 都有现成的 packages 供安装使用。如果是没有提供这个套件的操作系统，可能得自行编译了，原始码在：

http://www.tug.org/teTeX
ftp://cam.ctan.org/tex-archive/systems/unix/teTeX
ftp://tug.ctan.org/tex-archive/systems/unix/teTeX
ftp://dante.ctan.org/tex-archive/systems/unix/teTeX

[1] http://tug.org/texlive/

[2] http://www.tug.org/。加入会员的话，只要些许会费，则可以获得 TEX Live CD 及年报信息的寄发。

## 2.2 MS Windows 系统

最常使用的 free 版本，大概就是 MiKTEX 及 fpTEX，其中，后者等于是 Un*x 中的 teTEX 的 Windows 移植版本。

http://www.miktex.org/
http://www.fptex.org/

安装的话都自动化了，应该可以很方便的安装起来。

### 2.2.1 cygwin 环境

这是 Windows 系统下的一个 Un*x 环境（正确的说，是 Linux-like），有了这个环境就可以使用 Unix-like 的界面，也可以编译 Unix-like 中的程序，当然也就可以安装 Unix-like 系统的 teTEX 套件了，有人习惯了 Unix-like 的操作环境，但又常需要在 Windows 平台下作业，这是个不错的选择。

http://sources.redhat.com/cygwin
http://sources.redhat.com/cygwin/setup.exe

只要先下载 setup.exe 这个可执行文件，然后执行后按着指示就可以完成安装，当然，网络要联机。至于 teTEX 相关的套件，安装好 cygwin 就会安装，至于中文 CJK 套件，感谢
seventeen 的制作，请参考：

http://seventeen.mit.edu/blog/17/archives/000141.html

## 2.3 Mac OS X 系统

个人对 Mac OS X 并不熟悉，所以仅提供个人知道的 distribution。但 Mac OS X 亦可以安装 Un*x 系统上的 teTEX 系统，也可以在其上自行编译。

参考文件：
[文件]MacOS 10.2.4安装teTeX
http://www.rna.nl/tex.html
http://www.cs.wright.edu/~jslater/mac-tex/mac-tex-intro/mac-tex-intro.pdf
TEXShop
http://www.uoregon.edu/~koch/texshop/texshop.html
iTEXMac
http://itexmac.sourceforge.net/

## 2.4 商业维护的 TEX/LATEX 系统

上面所提到的都是 free 的版本，纵使是 TEX Live CD，虽也可以让非 TUG 会员订购，但他们的 iso 是可以自由下载，不一定要花钱买。当然，这里是鼓励大家加入会员或花点小钱购买，也算是赞助他们继续维护好质量的 TEX 系统。另外，亦有商业维护的 TEX/LATEX 系统，虽然要花钱，但功能上可能会比较符合特别的需要，而且有比较完整的售后服务方案：

| | | |
|---|---|---|
| TrueTEX | http://www.truetex.com/ | Windows |
| TurboTEX | http://www.truetex.com/ | Un*x, DOS, Windows |
| Y&Y TEX | http://www.YandY.com/ | Windows |
| pcTEX | http://www.pctex.com/ | Windows |
| VTEX | http://www.micropress-inc.com/ | Windows, Linux, OS/2 |
| Scientific Word | http://www.sciword.demon.co.uk | Windows |
| Textures | http://www.bluesky.com/ | Macintosh |
| OzTEX | http://www.trevorrow.com/oztex/ | Mac OS X |
| Scientific Assistant | http://www.advanced-science.com/ | Mac OS X |

## 2.5 选个顺手的编辑器

TEX/LATEX 本身并不内附编辑器，这和许多排版软件不一样，他只专注在排版的过程，原始文稿是如何产生的并不插手干涉，这样子的自由度很大，每个人都可以选用适合他自己的编辑器，但和目前一般的幕前排版系统比较的话，会让初接触的朋友不知所措，因为他不晓得要如何使用 TEX/LATEX 来「编辑」原始文稿！

当然，有些 TEX/LATEX 的发行版，干脆就弄了个编辑器，把编辑器和排版系统本身连接起来，这样是很方便，但有很大的可能又得花时间学习另一种不熟悉的编辑器操作。本 文的说明，不准备被编辑器给绑住，你爱用什么编辑器就用什么编辑器，让我们专心在排 版过程本身吧！

以下仅简单介绍和中文兼容的编辑器。当然，只要使用顺手的编辑器都可以拿来用。原则上，刚开始接触 LATEX，个人是建议从命令行开始了解起，等整个流程有个概念后再来使用一些方便的编辑器上的宏及按钮设定，不然，有时编辑器上的设定有问题时，会不知 道从何改起。至于编辑器的操作，请自行参考各编辑器的说明文件或网络上的教学文件， 这里就不多谈了。

### 2.5.1 Vim

这有各种平台的版本可以下载：

http://www.vim.org/

可以配合 vim-latex suite 来使用：

http://vim-latex.sourceforge.net/

### 2.5.2 GNU Emacs/XEmacs

这也是有各种平台的版本，也可以配合 AUCTEX[3] 来使用，相当方便，这也是 Knuth 教授本身所爱用的编辑器：

http://www.gnu.org/software/emacs/emacs.html
http://www.xemacs.org

### 2.5.3 NEdit

这也可以配合 AUCTEX 来使用，但只有 Un*x 系统的版本：

http://nedit.org

### 2.5.4 WinEdt

这是 shareware，只有 Windows 版本：

http://home.istar.ca/~winedt

### 2.5.5 UltraEdit

这也是 shareware，也是只有 Windows 版本。

http://www.ultraedit.com

[3] http://www.gnu.org/software/auctex/

### 2.5.6 Kile

这是很方便的 LaTeX 图形界面整合环境，还可叫出绘图软件来绘图，如果其他操作系统也有安装 Qt/KDE 的话，也是可以编译安装使用：

http://kile.sourceforge.net/

# 第 3 章

# TEX/LATEX 语法概说

这一章要谈的是，和一般的纯文本文稿及其他 markup 式文件系统在语言上的一般差异性。为了让观念上能够更清楚，以下所述主要是要在命令行执行的，致于编辑器上方便的按键及宏，这里就不多谈，一方面是每个人使用的编辑器不一样，二方面是要先把黑盒 子拿掉，整个处理流程才会有概念。

当然，由于完全还没有开始实际写文稿来测试，所以，这章是纸上谈兵，不必动手，用看 的就好。但，别急，我们会在第 4 章开始实际玩看看，请别忘了，到时要再回头来复习一下这些资料。

而且，前面已经说过，这篇文章主要是着眼于 LATEX 所附上的宏，一些其他方便的套件引用，将会在最后或另文再来谈。其实，不引用任何外来特殊套件，让 LATEX 本身去处理，最起码也就不会太离谱，要讲求美观、微调，个人是认为先把基础弄起来再说，有些 套件的复杂程度，会令人头疼，你是不是真有这个需要，值得考虑。而且，很多时候自认为不错的「微调」，其实常常会不合排版的惯例。TEX/LATEX 的语法，可以是很简单明白，也可以是相当的复杂，这是 TEX 系统本身的弹性所导致。

## 3.1 LATEX 文稿的处理流程

最简单的一句话，就是把编辑器编辑好的文稿（通常结尾是 .tex），利用 latex 这个指令去编译文稿就对了！

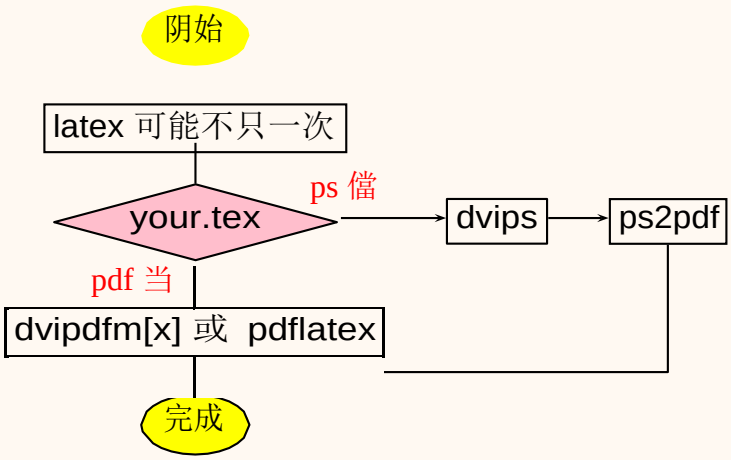
```
latex your.tex
```

需要注意的是，如果有索引，还要用 makeindex 执行一次，有参考文献，还需要 bibtex 处理一次，最后再使用 latex 再处理一至二次，也就是说视文稿的复杂程度，latex 可能需要执行好几次，这在往后碰到时会再提出来。另外，处理中文的话，需要其他前置处

理，这里暂时先以英文文稿来说明，中文的部份，只要加入中文环境及（或）改用能处理中文的预处理器就可以了。

这样经过 latex 处理后， 会产生一个 your.dvi 檔， 然后可以使用 dvips 来产生 PostScript 格式的档案。也可以使用 dvipdfm[x] 来产生 PDF 的格式，当然，也可以使用 ps2pdf 经由 PostScript 格式转换成 PDF 格式。另外，也可由 pdflatex 由 your.tex 直接编译成 PDF 格式的输出。

### 3.1.1 总结一下处理流程




## 3.2 LATEX 的特殊专用符号

在 TEX/LATEX 的世界，原始文稿都是纯文本档，任何一种编辑器都可以打开来编辑、观看。而排版指令通常是由反斜杠（\, backslash）所开头来引导。批注则是由百分号（%）来引导。例如，以编辑器编辑下列文字：

```
This is my first \LaTeX\ typesetting example.
```

编译后会变成以下的结果：

This is my first LATEX typesetting example.

其中的 \LaTeX 就是 LATEX 的一个指令，会显示 LATEX 这个特殊的图示。

由于，西方国家的语系，通常字母、符号的最大容量只有 256 个（$2^8$），因此，许多现有的符号必须拿来当做控制指令，才能符合排版的多样化需求。以下的符号，接触

TEX/LATEX 的朋友，可能都得时时留意，不要未经处理就直接写进文稿里头去了。

通常，编辑器的语法颜色会帮助判断语法是否正确，但不是都能完美无缺，有时还是会漏掉，这时别忘了查看一下 *.log 档案，例如：编译 your.tex 档的话，他的 log 檔就是 your.log。

| 符号 | 作用 | 文稿上使用 | LATEX 的替代指令 |
|---|---|---|---|
| \ | 下排版命令 | $\backslash$ | \textbackslash |
| % | 批注 | \% | NA |
| # | 定义宏 | \# | NA |
| ~ | 产生一个空白 | \~{} | \textasciitilde |
| $ | 进入（离开）数学模式 | \$ | \textdollar |
| _ | 数学模式中产生下标字 | _{} | \textunderscore |
| ^ | 数学模式中产生上标字 | \^{} | \textasciicircum |
| { | 标示命令的作用范围 | \{ | \textbraceleft |
| } | 标示命令的作用范围 | \} | \textbraceright |
| < | 数学模式中的小于符号 | $<$ | \textless |
| > | 数学模式中的大于符号 | $>$ | \textgreater |
| \| | OT1 编码，数学模式中才能正确显示 | $\|$ | \textbar |
| & | 表格中的分隔符 | \& | NA |

## 3.3 LATEX 排版上的一些规范或惯例

除了上面所谈到的特殊符号外，也有一些规范或惯例要遵守，有些是比较硬性的规定，有些则只是惯例，可能不同的国家、语言会有不同的惯例，暂时先把他当成是 LATEX 的游戏规则就成了。

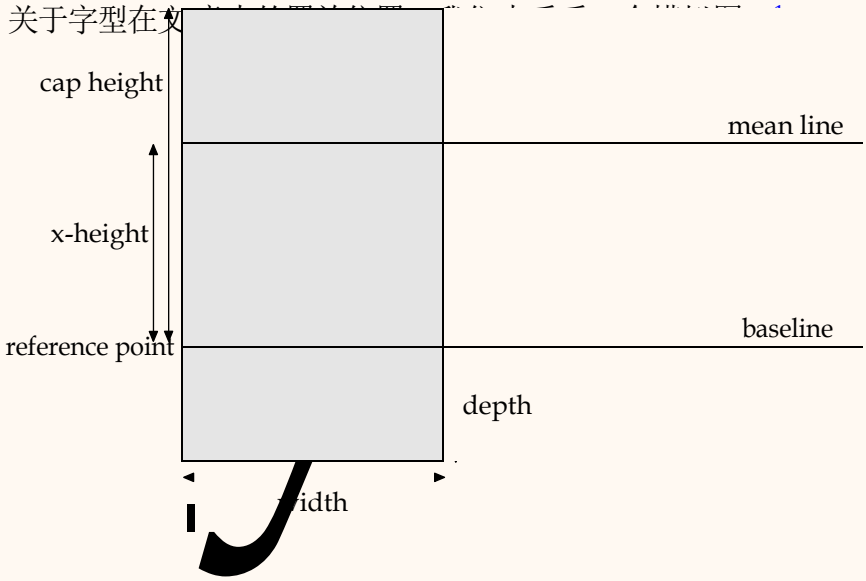
### 3.3.1 字型的相关术语

要谈排版上的规范、惯例前，我们得先认识一下字型的一些术语，以便往后文章中提到时 有个概念。通常我们每个字都是置放于一个假想的方框中，称为 em-square，同一个字型的同一个点数，每一个 em-square 大小都是相同的，实际上的字（glyph）要置放在这个 em-square 的什么位置，这是字型设计者的观点，所以，同样点数的不同字型，他的字的大小不一定会一样，因为我们是使用 em-square 的大小在比较的，而非实际的 glyph。

在文章中排列的时候，则是将 glyph 置于一个以假想参考点（reference point）为基准的一个假想在线，称为基线（baseline），大写字母除了 Q 以外，他们的底部都是置于基线

上的。但小写字母则不一定刚好座落在基在线，有些字的笔画可能超出基线以下，例如

y、j 等字母。关于字型在文




这个超出基线以下的长度，我们称之为深度（depth），以上的就称为字高（height），当然大小写的不同又分为大写字母的字高（cap height）及小写字母 x 的字高（x-height），由于这个例子里是调合字，所以每个字的宽度（width）不一定会一样，像打字机字族的则是等宽的字型。字高加上深度，我们就称之为 totalheight，大部份的情况，仅仅说
height 时是不包括 depth 的，而且通常指的是 cap height。

mean line 在一般比较少用到，通常是字型设计时才会用到，他是指小写字母去除上面突出的部份所连成的一个基线，这个 mean line 到 baseline 的距离，一般就称为 x-height， 当然就是小写字母 x 的高度，因此我们会有一个长度单位，称为 ex，指的就是这个x-height。

中文字的话比较特殊，他是以 em-square 的中心点来置放 glyph 的，在中英文混合时，中文字并不是刚好座落于基在线的，会超出基线下一点点，至于会超出多少，则和字型的设计有关，每种字型都有可能会不同。这也是为求排版上的一致性，字型可能都需要尽量使用同一套的各种字型的原因，否则就得经过微调，才能使整个字型表现上取得协调一致。

这些专门术语，往后提到一些指令的参数的描述时都会使用到，因此先熟悉一下，例如： 字型旋转时，跟据的就是以参考点（reference point）为准，沿延伸出的轴心来旋转的， 而一般所说的行距，指的是上下两 baseline 的距离。

[1]实际字型设计上的各部份专有名词及结构，当然不会是这么简略，这里的模拟图，只是暂时让字的一般置放有个粗略的概念。

### 3.3.2 一般性的游戏规则

1. LATEX 的指令都是大小写有别的，由 \ 开头，后接由字母组成的字符串或单一的非字母字符。其中由 [ ] 中括号括住的是选择性参数，可以省略，由 { } 大括号括住的是不能省略的参数，当然，LATEX 的指令不一定会有参数，但绝大部份都会有参数，只不过把他给省略使用默认值罢了。

2. LATEX 文稿中，空一个英文空白和空多个英文空白的作用是一样，LATEX 会认作一个英文空白。

3. 平常我们编辑纯文本档，按个 Enter 键，就代表换行，但实际排版出来，一行的宽度是按照排版版面的设定，也就是说，你在文稿中按 Enter，不代表排版后就是从这里断行，LATEX 会依一行应有的宽度经过整体计算后自动补成一整行后再来断行，而且会在中间自动补足一个空白。这在英文很自然，称为字（word）间空白，但中文则 不一样，在编辑器中编辑中文，随意按 Enter 的结果，会造成文章中的中文间出现空白。这会在本文中适当的时机，提出解决的方法。

4. 编辑器中，多按几次 Enter 就多空出几行，但在 LATEX 文稿里，多个空白行，和一个空白行是一样的作用，LATEX 会把他认作是一个空白行。而这个空白行，LATEX 同时也会认作是新段落的开始，所以 LATEX 是以空白行来分隔各个段落。

5. LATEX 预设每个章节的第一个段落的第一行是不内缩（noindent），从第二个段落开始才会内缩（indent）。当然，这是可以更改的，往后会再提及。

6. LATEX 的指令，是从反斜杠后第一个字母开始，到第一个非字母符号为止（包括空白、标点符号及数字）。因此：

   ```
   This is my first \LaTeX typesetting example.
   ```

   这样的话，实际结果，因为 \LaTeX 后的空白是属于指令的一部份，空白将不会被解释，这样会印成：

   ```
   This is my first LaTeXtypesetting example.
   ```

   这种结果，LATEX 和 typesetting 连在一起了。要避免的话，就要指定指令的作用范围，例如以下的大括号。或就真的加个空白，例如 \ ，LATEX 碰到 \ 就会形成完整的指令，其后的空白就会被真正解释为空白了：

   ```
   This is my first {\LaTeX} typesetting example.
   This is my first \LaTeX{} typesetting example.
   This is my first \LaTeX\ typesetting example.
   ```

所以，正常印出来应该是：

This is my first LaTeX typesetting example.

7. 批注符号（%），可以放在一行的任何地方，% 后的文字会被 LaTeX 忽略。所以，如果是放在一行的最尾端，那么 LaTeX 会自动插入的字间空白也将会被忽略。例如：

```
This is my fisrt \LaTeX\ document. Give \LaTeX\ a%
try.
```

这样一来，排版出来会变成：

This is my fisrt LaTeX document. Give LaTeX atry.

a 和 try 连在一起了！正常应该是：

This is my fisrt LaTeX document. Give LaTeX a try.

基于这个特性，我们可以应用在中文，也就是说在编辑器中，中文文章按 Enter 键换行时，尾端加个 %，这样一来 LaTeX 就不会插入英文字间空白，中文字就可以连成中间没有空白的一整行了，否则 LaTeX 在整篇文稿断句时，会自动在原换行处填入一个英文空白，因为，原始的 TEX/LATEX 是认不得中文的。

8. 中英文混合的时候，通常，英文字前后都会留个空白，以便和中文区隔开来，只是这个空白要多大，这就没有固定的惯例，通常留个英文空白也是可以，要讲究的话，等谈到中文排版相关议题时再来讨论，目前就养成习惯，英文单字前后留个英文空白。

### 3.3.3 针对标点符号的游戏规则

1. 中英文的引号不一样，这里请特别注意，许多人常常搞错。中、英文引号不管单双 都要分左右。英文的话，左边的引号是 grave accent，是键盘左上方 Esc 或 F1 下方有波形号的那一个键；右边的是 apostrophe，也就是键盘左边 Enter 键隔壁的那个键。双引号的情形是键入两次的左单引号及两次的右单引号，而不是用 " 这个一次完成两个点的 ditto marks。所以，实际上在键入文稿时是：

```
Please press an ‘Esc’ key.
Please press an ’Esc’ key. 这是错误示范！
“This sentence.”
"This sentence." 这是错误示范！
```

排版出来的情形是：

```
Please press an ‘Esc’ key.
Please press an ’Esc’ key. 这是错误示范！
“This sentence.”
”This sentence.” 这是错误示范！
```

中文的话，我们是使用中文全角的「、」及『、』，在中国大陆则已改用和英文相 同形状的全角符号，但这在中文直排时会出问题，因此，中文的单、双引号还是得 维持我们目前使用的。

2. LATEX 会在英文文章的一个句子结束和另一个句子开始的中间，自动调整成较大一点的空白，这可以增加文章的易读性。所谓一个句子结束，例如：句点（.）、问号（?）、惊叹号（!）及冒号（:），这当然是指英文的半角标点符号，不是中文的全形标点符号。你可以注意一下上面所举的例子，在 document. 和 Give 之间的空白会稍微大于其他英文单字间的空白。

   现在的问题是，如果这些标点符号后面不是另一个句子的开始的时候，LATEX 无法去判断这种情形，这时得由我们自己自行判断、处理了。例如英文缩写字：

```
I am Mr. Edward G.J. Lee, G.J. is a abbreviation of my name. I
am Mr.~Edward G.J. Lee, G.J. is a abbreviation of my name.  I
am Mr.\ Edward G.J. Lee, G.J. is a abbreviation of my  name.
```

   其中 Mr.\ Edward 的写法，和 Mr.~Edward 几乎是一样的，都是强迫插入一个比较小的正常单字间空白，差别在于后者也另外表示不可以从这里换行，通常用在人名 的时候，让他们不致中断，一般在人名的排版，包括他的头衔、职称，是不中断成 两行而分开的。而且整个文句较长的话，以后者较恰当，才不会因为断行被分成两 半，这个 ~ 符号也因此在 TEX 的专有名词，就称为 tie，把他们绑住的意思。排版出来的时候会变成：

```
I am Mr.  Edward G.J. Lee, G.J. is an abbreviation of my  name.
I am Mr. Edward G.J. Lee, G.J. is an abbreviation of my name.
```

   请放大去仔细比较一下结果就知道了。第二行的才是正确的，Mr. 和 Edward 之间的空白是正常单字间空白，比第一行的句子结束空白要小一点点。其他有使用到缩写 字的场合，例如：‘Dr.’、‘etc.’、‘e.g.’、‘i.e.’、‘vs.’、‘Fig.’、‘cf.’、‘Mrs.’，这些都不 是代表句子结束，所以，要插入一个正常空白。

   那 G.J. 后面为什么没有插入正常空白？那是因为，J 是大写的，这时 LATEX 不会去误认为是句子结束，通常句子结束时的句点前的那个字母是小写的。Well，有没有觉得有点道理？:-)

等等，事情还没有结束！Knuth 教授出了一道考题，如果句子的结束是 'Please see Appendix A.' 后面又还接有另一个句子。这时怎么办？由于，不会认为是句子结束，因此会插入正常空白，但这正是句子结束呀！请暂时先记得，使用 ...Appendix A\null.，或 ...Appendix A\@.。这个说来有点话长，有机会再来探讨，请记得 '\null' 和句点间是没有空白的。例如：

```
Please see Appendix A. We will be there soon.
Please see Appendix A\null. We will be there soon.
```

排版出来的结果将会是（差异不明显，请小心比较）：

Please see Appendix A. We will be there soon.
Please see Appendix A. We will be there soon.

如果，你现在阅读的是 HTML 格式文件，有些例子如果无法明显显示出来，请改阅览 PDF 版本。而且，如果你制作 PDF 格式时，字型没有内嵌（本文的英数字是嵌入 Computer Modern Type1 字型），差异可能将会更不明显。可试着使用 gv/gsview 去阅览，然后调整成 Landscape，把句子尾部拉到边缘的地方去就看得出来了。这在句子多的时候，这个空白也并非固定大小的，LATEX 会视文章结构的需要做细微的调整。

3. 删节号中文英也是不同，英文是三点，如果碰到句点的话，则是四点。中文的话是 六点，碰到中文句点很容易就分得清楚。但是英文这个三点，不是就打个三个句点 了事，这样的点太密集，可以使用 \ldots 或 \dots 指令，例如：

```
I'm not a good man ..., but a good husband .... 错 误 示 范 !
I'm not a good \ldots\ man \ldots, but a good husband \ldots.  I'm
not a good \dots\ man \dots, but a good husband \dots.
```

排版出的来结果是：

I'm not a good ... man ..., but a good husband .... 错误示范!
I'm not a good . . . man . . . , but a good husband . . . .
I'm not a good . . . man . . . , but a good husband . . . .

中文的删节号是由两个全角的三点所组成六点的，即：……，就是我们 Big-5 码的 0xa14b（U+2026），但由于 Unicode 尚有一个 MIDLINE HORIZONTAL ELLIPSIS(U+22EF)，因此，有些软件在解读上有可能会不一样，因为我们的字型， 大部分在制作标点符号时是置在中央的地方，不像中国大陆是置放在基线的地方， 而 Unicode 官方采用的样本字型，刚好是中国大陆的厂商所提供，这样一来有些软件工作者就认为我们的删节号应该是 U+22EF 了，很不幸的，我们的 Big-5 码并没有相对应的字码。

4. 破折号。在英文，相当于破折号的可能有三种：

- hyphen
  这是最短的 dash，通常就是键入 - 就行了，例如 father-in-low，这样会表现成 fater-in-low 。
- en-dash
  这是最常用的破折号，是键入两个 hyphen。例如 1991--2003 年，这会表现成 1991–2003 年。
- em-dash
  这是最长的 dash，由三个连续的 hyphen 组成，应该是最相近于我们中文所说的破折号。例如 I am---a good man. 会表现成 I am—a good man.。至于这个三个连续的 hyphen 前后是否要留空白，都有人使用，并没有硬性的惯例，但为了和中文的破折号配合（中文破折号前后，通常不留空白），个人通常是不留空白的。
- 真正的减号
  这应不能算是破折号，而是实际的减号或负号，这要进入数学模式，例如：负五，要写成 $-5$，然后表现出来是 −5。这也常常会有人搞错，不能直接键入一般的负号那个键来充数，这是因为 TEX/LATEX 的数学式子的用字和间隔处理，和一般内文不同的关系。
- 中文的破折号
  中文的破折号是占两个中文字位置的的一横线，长度和删节号相同。在中线位置的，定义上是有两种，en-dash 是 Big-5 0xa156，em-dash 是 Big-5 0xa158。但由于中文字间距的问题，有可能打出来的破折号中间会有一点空白[2]，例如——中文的 em-dash，这是－－中文的 en-dash。在论文中，破折号通常可以使用小括号或冒号代替。
- 中文的私名号及书名号
  中文的私名号，可以标明人名、地名，如孙逸仙；书名号（私名号的底线换成 波纹形状），可以用在书名，这些符号常造成排版上的困扰，常使用 《 》来取代书名号，私名号则无其他取代方法。在一般的自然及应用科学论文上通常不 使用这种旧式的私名号及书名号。

[2]这是可以调整的，也就是去除两个横线之间的字间距，这样就不会产生小空白了，中文删节号也有同样的情形，我们会在微调的部份再来讨论。

5. 避头点

   这可是排版的重要功能。英文的通常没有问题，LaTeX 会自动避开处理，中文就不一定了，LaTeX 可不认识中文，但通常中文相关程序及套件，多多少少都会处理，只不过，有时候偶尔可能会误判。那么，到底什么是避头点？底下列个表，大家就明白了，我列中文的，英文的就不列出来了，因为 LaTeX 会自动处理，不必我们担心。

| 标点符号 | 置放处 |
|---|---|
| ，。；、:」）》！？ | 不能置于行首 |
| 「（《 | 不能置于行尾 |
| 破折号及删节号 | 置于首尾皆可 |

   简单的说，除了破折号及删节号，没有开口的，不能置于最开头，开口向右的，不 能置于最右，开口向左的，不能置于最左。通常都会处理好，但校稿的时候要注意 一下误判的地方。

## 3.4 LaTeX 的文稿结构

### 3.4.1 环境（environment）

上一节所谈的都是指令，虽然也可以由大括号来定作用范围，但如果是一整段，甚至是一整篇文章都要作用时，那指令可能就不很适合了，因此，LaTeX 也有一种宏结构，称为环境（environment），主要是让作用范围能扩大至较大的范围。

所有的环境，都是起于 \begin{环境名称}，止于 \end{环境名称}，这两个指令之间的文稿都会被作用，而且，环境之内还可以套用其他不同的环境。

LaTeX 文稿的内文，其实就是包在一个 \begin{document} 和 \end{document} 这个 document 环境当中。

### 3.4.2 最简单的 LaTeX 的文稿结构

以下就是所有 LaTeX 必需具备的文稿大结构：

```
\documentclass{article}

  这里是 preamble 区
```

```
\begin{document}
    这里是本文区
\end{document}
```

\documentclass{article}，这是在告诉 LATEX 使用哪一种类别，我们目前使用的是 article 类别，关于类别会在第 6 章讨论。preamble 区，则是下一些会影响整个文稿的指令，及引用宏套件的地方，当然，完全不引用宏，也不使用影响全文的指令的话，preamble 区就是空白，不写任何东西。本文区，就是我们实际上写文章的地方。

现在也可以把前面所举的例子，放入本文区里头，preamble 区空白没关系，然后存盘，试着编译看看：

```
latex example.tex
dvips example.dvi          => 产生 ps 格式 example.ps
dvipdfm[x] example.dvi => 产 生 pdf格 式
pdflatex example.tex       => 直接由 .tex 产生 .pdf
```

真正的实例解说，会在下一章来进行，所以，这里暂时不会介绍有什么环境可以使用，先玩看看没有关系。由于还没谈到中文的问题，因此如果你想试试看，那暂时先使用英文，道理都是相通的。

### 3.4.3 preamble 区可以放些什么？

这里可以引用宏，而且会影响整篇文稿的指令，例如一些事先定义好的指令，想在整篇 文稿中使用，就可以置放在 preamble 区。

#### 1. 宏的引用

本文主要是标准 LATEX，但前面已提到，会有些宏套件不得不要引用，底下就来说明如格引用套件。这些套件都是一般 TEX 系统都会附上的。

指令及环境要如何开头都介绍过了，现在来看看引用宏要怎么开头。

```
\documentclass{article}
\usepackage{color}
\begin{document}
\textcolor{blue}{This is blue color.}
\end{document}
```

编译一下，看看结果是什么？这里使用的就是 color package，里头是由 TEX/LATEX macro 所写成一个宏套件。一般简单的我们就称为宏（macro），复杂一点的就称为宏套件（package），其实，里头都是一样的，只不过大小及有没有整理成一个系统的差别。

LATEX 里头有什么现成的套件可以使用，每个散布的 TEX 系统所收集的可能都会有所不同。大概没有人可以精通所有现存的 LATEX 宏套件，因为实在是太多了，不过，本文大概都会提到常用的宏套件。详细的宏套件的种种，会在第 7 章来说明。

2. 影响整篇文稿的指令

会影响整篇文稿的指令，通常也是放在 preamble 区，例如：

```
\linespread{1.36}
\parindent=0pt
```

\linespread 是在控制上下行的行距，这里就是将行距变成原来的 1.36 倍。至于什么是行距呢？就是这一行的基线（baseline）到下一行的基线的距离，通常英文文章不必去调整他的行距，但中文得适当加大行距以利阅读。

\parindent 是调整段落内缩的程度，这里调整成 0，也就是说各段落都不内缩的意思，也可以调整成其他的值，LATEX 就会依这个值去内缩。当然，不去设定的话，LATEX 就会依他的默认值去内缩。

### 3.4.4 章节结构

本文区当然是我们写文章的主要地方，及一些微调。在 LATEX 的文稿里头，章节标题的形成都是由同样的指令来控制的，这样有一个好处，临时插入章节标题及其内文时，我们不必去理会标题编号及目录的问题，也不必去理会要用什么字型、及字号要多大，LATEX 会自动计算处理，字号也会和内文使用的字号互相配合调整，使用者就专心在内文构思、写作即可。以下由列表来了解整个章节结构：

| 深度标号 | 指令 | 作用及注意事项 |
|---|---|---|
| -1 | \part{} | 这是最大的结构，我们中文通常称为「部」。 |
| 0 | \chapter{} | 章。在 article 类别里头没有章。 |
| 1 | \section{} | 节。 |
| 2 | \subsection{} | 小节。 |
| 3 | \subsubsection{} | 次小节。 |
| 4 | \paragraph{} | 段落。 |
| 5 | \subparagraph{} | 小段落。 |

章节标题的内容就是直接写入指令的大括号里头就可以了，LaTeX 在排版时会自动使用粗体、加入章节编号及纳入目录里头。

至于第一栏的深度标号（secnumdepth），book/report 类别的深度标号是 2，article 的是 3。这是什么意思呢？就是说 book/report 类别的文稿，在 \subsection{} 以后（subsection 本身仍会编号），章节就不再编号了；同样的，在 article 类别的文稿，在 \subsubsection{} 以后就不编号了。但仍然会独立出一单独行来表示这个是标题。不编号了的章节内容，当然也就不纳入目录里头了。这当然是可以更改的，只要更改 LaTeX 的 secnumdepth 这个变量的值就可以了，这个往后会提及如何更改 LaTeX 的默认值。像这篇文章，在 preamble 区就有一个设定：

```
% let the depth of report to subsubsection
\setcounter{secnumdepth}{3}
```

所以，这篇文章虽然使用的是 report 类别，但是章节的深度标号是标在 3，也就是说会编号到 subsubsection 为止，但这仍然是没有编入目录中的。

下一章就让我们开始实际动手吧！但……，怎么到现在都没有完整介绍指令呢？那我怎么会知道有什么指令可以使用？这是因为 LaTeX 的指令很多，直接介绍的话，一方面记不住，二方面也不容易了解他的实际作用，所以，我们将会在下章举例时穿插在里头说明，等这份文件接近尾声时，再来整理个指令速查表，这样以后查指令就很方便了，不必去死记，只要知道有个这样功能的指令就够了。

# 第 4 章

# 实际上排版玩看看

好了，现在正式来玩看看吧！本章主要是简单的实例说明，先进入状况再谈其他。实际应用比高谈阔论有用多了。

先来个「最高指导原则」：学会控制空间，你就学会排版了！刚学排版的朋友，往往会把所学到的东西去想法子布满你所有的空间（就是一个页面），但实际上，你要调整的，其实是各个部份的空间配置，抽象一点说，也就是一整个页面当中，没有文字、图表的部份才是排版真正的重点。你先听听就好，过一段时间的熟悉后，再来回头思考这个「玩弄」空间的原则。:-)

由于本文有 HTML 版本，为了转换上不造成失真，实例的部份，文稿上只写程序代码，结果的部份都是编译好的 PDF 格式档案，置放于网站上，可在线阅览或下载，简单的例子则不另制作独立的 PDF 档，请参阅本文的 PDF 格式档案内容。

## 4.1 简单的实例

这里就把前一章所谈到的一些内容整理成一个文稿，先来试试看，这里先使用 report 类别文稿，因为 article 类别文稿是没有 chapter 的：

```
% example1.tex
\documentclass{report}
\begin{document}
This is my first {\LaTeX} typesetting example.\\
This is my first \LaTeX{} typesetting example.\\
This is my first \LaTeX\ typesetting example.\\
I am Mr. Edward G.J. Lee, G.J. is a abbreviation of my name.\\
I am Mr.\ Edward G.J. Lee, G.J. is a abbreviation of my name.\\
Please see Appendix A. We will be there soon.\\
Please see Appendix A\null. We will be there soon.
\end{document}
```

使用编辑器编辑，然后存档成 exmaple1.tex，这样就可以编译了：

```
latex example1.tex          => 产生 example1.dvi
dvips -Ppdf example1.dvi  =>  产 生 example1.ps
ps2pdf example1.ps          => 产生 example1.pdf 或
dvipdfm[x] example1.dvi     => 由 example1.dvi 直接产生 example1.pdf 或
pdflatex example1.tex       => 由 example1.tex 直接产生 example1.pdf
```

编译好的 PDF 档可在此下载或阅览：

http://edt1023.sayya.org/tex/latex123/example1.tex
http://edt1023.sayya.org/tex/latex123/example1.pdf

### 4.1.1 关于换行

每行最后加了个 \\，这表示强迫换行的意思，否则 LaTeX 会依版面预设的宽度来换行，就不会是一个句子一行了，大家可以把这个 \\ 拿掉，再来编译试看看结果，就会知道怎么一回事了。也可以使用 \newline 这个指令，当然，我们都会聪明的选用较短的指令。而且 \\ 可以控制换行时的间隔，这在 \newline 则不行。例如：

```
Please see Appendix A. We will be there soon.\\[1cm]
Please see Appendix A\null. We will be there soon.
```

这样的话，两行之间的行距就是原来的行距再加上 1cm。甚至，也可以是负数的参数，这样行距就会变成原来的行距减去 1cm，当然，如果设过头了的话，两行可能会重迭在一起。既然，这里使用的是方括号，表示这些参数是可以省略的。

另外，\linebreak[n] 也可以强迫换行，n 代表由 1–4 的建议值，数值愈大表示愈是强烈建议，不设定的话，就是换或不换两种选择，没有中间地带。和前面所说的不同处是，这种换行会把原来那一行句子的长度平均布满版面上行宽的长度。例如：

```
Please see Appendix A. We will  be  there  soon.\linebreak
Please see Appendix A\null. We will be there soon.
```

排版后会表现成：

Please see Appendix A. We will be there soon.
Please see Appendix A. We will be there soon.

### 4.1.2 关于缩排

第一行缩排了！这是因为我们完全没有分章节，所以，LaTeX 就把这些内容当做是引言的

部份，依 LaTeX 的安排，引言开头是会缩排的。要解决这个问题，可有两种方法：

1. 在第一行之前加入 \noindent 来指示 LaTeX 不要去缩排。但是这只作用在下指令的地方，其他该缩排的地方还是会缩排。

2. 在 preamble 区加入 \parindent=0pt，这表示让全文的缩排为 0pt，当然，这就表示全文都不要缩排了。

## 4.2 加入章节标题

在 LaTeX 里头，要加入章节标题实在是太容易了，也不必去管字体的大小及置放的位置，尽管加上去就对了！LaTeX 会替我们安排一切。我们这里仍然以 report 类别来说明，因为 article 类别里头，没有章，只能适用于较简单的短文。

```
% example2.tex
\documentclass{report}
\begin{document}
This is the first experience of \LaTeX.
\chapter{Aesop Fables}
\section{The Ant and the Dove}

An ant went to the bank of a river to quench its thirst, and
being carried away by the rush of the stream, was on the
point of drowning.

A Dove sitting on a tree overhanging the water plucked a
leaf and let it fall into the stream close to her. The Ant
climbed onto it and floated in safety to the bank.

\section{The Dog in the Manger}

A dog lay in a manger, and by his growling and snapping
prevented the oxen from eating the hay which had been
placed for them.

“What a selfish Dog!” said one of them to his companions;
“he cannot eat the hay himself, and yet refuses to allow
those to eat who can.”

\chapter{The Eagle and the Arrow}

An eagle sat on a lofty rock, watching the movements of a
Hare whom he sought to make his prey.
```

```
An archer, who saw the Eagle from a place of concealment,
took an accurate aim and wounded him mortally.
\end{document}
```

编译出来的结果：

http://edt1023.sayya.org/tex/latex123/example2.tex
http://edt1023.sayya.org/tex/latex123/example2.pdf

请注意他什么时候会缩排，什么时候会换页。report 类别，新的一章会换页，如果想节省一点空间，可以换用 article 类别，\chapter{} 改用 \section{}，原来 \section{} 就改用 \subsection{}，这样就不会换页，内容就会连续下去了。大家可以试着把 report 改成 article 及 book 再重新编译一次，试试看结果有何不同。

## 4.3 加入 title page 信息

这是指内页的第一页，我也不知道这个中文专有名词是什么，在 LaTeX 里头，我们就称为 title page。在 LaTeX 的标准格式里，他包括了标题（title）、作者名字（author）、日期（date）及感谢词（thanks）。要注意的是，在 report/book 类别，title page 是自成一单独页的，但在 article 类别里，他是和本文连起来的。我们就以上面的伊索寓言的文章为例，要修改的地方是 preamble 区及本文区的 \maketitle：

```
% example3.tex
\documentclass{report}
\title{Aesop Fables}
\author{Aesop\thanks{Thanks to the reader.}
        \and Nobody\thanks{Thanks to nobody.}}
\date{\today}
\begin{document}
\maketitle
This is the first experience of \LaTeX.
\chapter{Aesop Fables}
\section{The Ant and the Dove}
 ...
```

排版出来的结果如下：

http://edt1023.sayya.org/tex/latex123/example3.tex
http://edt1023.sayya.org/tex/latex123/example3.pdf

我们可以发现，这一页是不编页码的，从下一页开始才是第一页。作者可以有多个，使

用 \and 指令来连接。日期不一定要有，如果没有 \date{\today} 这个指令，那还是有日期，但只能固定在今天。如果内容过长，他会自动折行，但也可以手动加 \\ 来强迫换行，不管如何换行，整个句子是居中排列的。\maketitle 是下在本文区的开头，如果不下这个指令，那编译时不会有什么错误，只是就没有 title page 了。

## 4.4 加入目录（Table of Contents）

加入目录（Table of Contents）对 LaTeX 而言，更是轻而易举的事情，只要在本文开头加个 \tableofcontents 指令就成了！依上面的例子，修改成：

```
% example4.tex
\documentclass{report}
\title{Aesop Fables}
\author{Aesop\thanks{Thanks to the reader.}
          \and Nobody\thanks{Thanks to  nobody}}
\date{\today}
\begin{document}
\maketitle
\tableofcontents
This is the first experience of \LaTeX.
\chapter{Aesop Fables}
\section{The Ant and the  Dove}
 ...
```

排版出来的结果如下：

http://edt1023.sayya.org/tex/latex123/example4.tex
http://edt1023.sayya.org/tex/latex123/example4.pdf

这里千万要注意的是，\tableofcontents 要加在 \maketitle 的后面，否则目录会印在 title page 之前。而且要编译两次。第一次产生 example4.toc，然后第二次编译再跟据这个 toc 文件，真正编入目录。

目录是包括图表目录的（List of Figures, List of Tables），但我们目前还没有谈到图表的排版，因此暂时略过，等谈到时再来看要如何加入图表目录。

## 4.5 加入摘要（abstract）

这不一定会有，如果要加入的话，可使用 abstract 环境，在这个环境中的文章，左右会

缩排。要注意的是，只有 article/report 类别才有 abstract，book 类别不能使用这个环境。

```
% example5.tex
\documentclass{report}
\title{Aesop Fables}
\author{Aesop\thanks{Thanks to the reader.}
        \and Nobody\thanks{Thanks to nobody}}
\date{\today}
\begin{document}
\maketitle
\begin{abstract}
The tale, the Parable, and the Fable are all common and popular
modes of conveying instruction. Each is distinguished by its own
special characteristics.
\end{abstract}
\tableofcontents
\chapter{Aesop Fables}
\section{The Ant and the Dove}
 ...
```

排版出来的结果如下：

http://edt1023.sayya.org/tex/latex123/example5.tex
http://edt1023.sayya.org/tex/latex123/example5.pdf

report 类别的摘要自成一页，不编页码，且不会编入目录中，这和一般的论文格式可能会不一样，使用时请注意。artcile 的类别则仍然是和本文相连的，会出现在文章标题之后。

abstract 和 summary 在较正式的论文是有区分的，通常 abstract 在文前；summary 则在文后。但目前一般性的文章则没有这样区别，通通当成「摘要」。通常，摘要里头是不用批注、无交互参照也不使用公式图表的。

## 4.6 加入批注

在 LaTeX 里头，批注可有两种方式，一种是脚注（footnote），一种是边注（marginal note）。通常 LaTeX 的脚注默认是由阿拉伯数字在编号，置于页底部。在没有部（part）的情形下，report/book 类别，编号每章会从头起算，article 类别则会连续，而且，会使用 footnotesize 的字体印出。边注则不编号，字体是正常大小。

### 4.6.1 脚注（Footnote）

在所要加注的那个字后，使用 \footnote{} 指令即可，解说的文字就写入大括号之内，一般 LaTeX 的指令在此都仍然有有作用，会印在此页的底部，以小一点的字来印出，并加上编号。以下我们就试试看在 Dove 这个字来做脚注。请注意，Dove 这个字和 \footnote{} 之间是没有空白的。

```
% example6.tex
\documentclass{report}
\title{Aesop Fables}
\author{Aesop\thanks{Thanks to the reader.}
          \and Nobody\thanks{Thanks to  nobody}}
\date{\today}
\begin{document}
\maketitle
\tableofcontents
This is the first experience of \LaTeX.
\chapter{Aesop Fables}
\section{The Ant and the  Dove}

An ant went to the bank of a river to quench its thirst, and
being carried away by the rush of the stream, was on the
point of drowning.

A Dove\footnote{Pigeon, an emblem of peace.}
sitting on a tree overhanging the water plucked a
leaf and let it fall into the stream close to her. The Ant
climbed onto it and floated in safety to the bank.
 ...
```

排版出来的结果如下：

http://edt1023.sayya.org/tex/latex123/example6.tex
http://edt1023.sayya.org/tex/latex123/example6.pdf

### 4.6.2 边注（Marginal note）

边注只是把 \footnote{} 换成 \marginpar{} 而已，内容仍然写入大括号内。但和脚注不一样的是，他没有编号（因为就在旁边，无此必要），他的字体也不会小一号，和内文的字体大小是一样的，这在后面讨论到字型的时候会谈到如何改变字体的大小。

```
% example7.tex
\documentclass{report}
```

```
\title{Aesop Fables}
\author{Aesop\thanks{Thanks to the reader.}
        \and Nobody\thanks{Thanks to  nobody}}
\date{\today}
\begin{document}
\maketitle
\tableofcontents
This is the first experience of \LaTeX.
\chapter{Aesop Fables}
\section{The Ant and the  Dove}

An ant went to the bank of a river to quench its thirst, and
being carried away by the rush of the stream, was on the
point of drowning.

A Dove\marginpar{Pigeon,  an  emblem  of  peace.}
sitting on a tree overhanging the water plucked a
leaf and let it fall into the stream close to her. The Ant
climbed onto it and floated in safety to the bank.
 ...
```

排版出来的结果如下：

http://edt1023.sayya.org/tex/latex123/example7.tex
http://edt1023.sayya.org/tex/latex123/example7.pdf

## 4.7 字型的相关调整

TEX/LATEX 的字型系统算是相当复杂的，这里不多谈其中原理，站在用户的角度，我们只要知道怎么使用就行了。在这里，我们说字型（font），指的是字型本身的一个总称，或称为字体，在字的形状的时候，我们就称为字形（font shape）。

LATEX 使用的字型选字机制，以目前新版本的 LATEX 而言，是使用 1993 年发行的 NFSS(New Font Selection Scheme) 第二版为标准。当然，仍然是建立在 TEX 字型机制的基础上的，这已超出这篇文章的范围。

### 4.7.1 LATEX 对字型的属性描述

在 LATEX 里，对于字型的描述，使用了五种属性来说明，这五种属性，也是 LATEX 宏中常要使用到的参数，甚至是错误讯息标示字型来源的时候，会把字型的这些属性给显示出

来。

1. 字型编码（font encoding）
   这里所谓的字型编码，指的是各个个别的字在一个字型里头的排列顺序及安排方式。原始的 TEX 字型编码我们就称为 OT1(Old TEX text encoding)，这是预设的，如果都不指定字型编码，那所使用的就是 OT1 编码。在目前新一代的字型编码里头，字的安排方式及内容和 OT1 不一样，例如 T1[1]，这在往后提到改变字型编码时会再谈到，我们目前就不去调整字型编码，使用默认的 OT1，其他的编码这里就不多谈了。

2. 字族（font family）
   指同一设计类型的字型集合的名称， 例如罗马字族（roman）、打字机字族（typewriter）等等，通常前面会冠上制作商或制作人的名称，例如 Knuth 教授设计的，称为 ‘Computer Modern Roman’，Adobe 公司制作的罗马字族称为 ‘Adobe Times’。我们预设使用的，当然就是 Knuth 教授所设计的 Computer Modern fonts。以下为一些例子：

   | 简称 | 代表意义 |
   |---|---|
   | cmr | Computer Modern Roman |
   | cmss | Computer Modern Sans Serif |
   | cmtt | Computer Modern Typewriter |

3. 字型系列（font series）
   这是指字型的 weight（胖瘦）及 width（长扁）来区分的。例如粗、细字体，一般我们正常用的是 medium，粗体则是 bold。以下是一些例子：

   | 简称 | 代表意义 |
   |---|---|
   | m | medium |
   | b | bold |
   | bx | Bold extended |
   | sb | Semi-bold |
   | c | Condensed |

4. 字形（font shape）
   这个望文生义，就是字的形状。例如意大利斜体（italic）、斜体（slant）、small caps 等等。以下是几个例子：

[1] 正式名称是 Cork’s TEX extended text encoding 又称为 Text Companion encoding。这里的 T1 和 Type 1 字型规格无关，他是字型编码方式，他把字型里头有关一些重音符号字母单独视为一个单独的字，而非如 OT1 是由一般字母和重音符号组合而成。

| 简称 | 代表意义 |
|---|---|
| n | 正常字（normal），指 upright 或 roman |
| it | Italic |
| sl | Slanted |
| sc | Small Caps |

5. 字号（font size）
   预设的字号是 10pt（10 point），十点字。不加单位的话，默认的就是 pt。请注意，非标准 LaTeX 类别的预设字号可能会不一样。

我们对字型要调整改变的，就是这些字型属性的设定值。LaTeX 已设定好方便的指令给我们使用。

### 4.7.2 调整字族、字型系列、字形的指令

| | 字型 | 标准指令 | 宣告式指令（环境） | 旧用法 |
|---|---|---|---|---|
| 字形 | textup | \textup{textup} | {\upshape textup} | |
| | *italic* | \textit{italic} | {\itshape italic} | {\it italic} |
| | *slant* | \textsl{slant} | {\slshape slant} | {\sl slant} |
| | small caps | \textsc{small caps} | {\slshape small caps} | {\sc small caps} |
| 系列 | medium | \textmd{medium} | {\mdseries medium} | |
| | **boldface** | \textbf{boldface} | {\bfseries boldface} | {\bf boldface} |
| 字族 | roman | \textrm{roman} | {\rmfamily roman} | {\rm roman} |
| | sans serif | \textsf{sans serif} | {\sffamily sans serif} | {\sf sans serif} |
| | typewriter | \texttt{typewriter} | {\ttfamily typewriter} | {\tt typewriter} |

先别吓了一跳，这是有迹可循的。其中 upright, medium, roman 都是一样的，这是一般的正常字，就不必麻烦去设定他了，除非是要在特定字型范围里头，重新改变成正常字体。 从前面所说的简称的字符串，再和 text, family, series, shape 去配对来使用，这样只要记得简称就行了，例如：italic 的就是 \textit{}。不然也可以使用「偷吃步」的旧用法，其实这也不是什么偷吃步，他是原始 Plain TeX 所定义的，在旧版的 LaTeX 2.09 也相容他而沿用，并加以扩充。但如果使用旧用法，那有时组合式的表示时可能会无效，例如粗斜体这种粗体和斜体设定混合时，就无法产生粗斜体了，这时还是得乖乖使用正统标准 LaTeX 的表示法。

要注意的是，大括号的位置，宣告式的指令，整个作用范围是连指令一起包住的，他可以 当成环境来使用，例如 \begin{itsahpe}，\end{itshape}，这样在这个环境内的文字就通通会使用 italic 斜体，也可以不加参数使用，例如 \itshape，这样以下的文字通通会使

用 italic 斜体，直至另一个改变字型的指令出现为止。标准指令的作用范围则是当做指令的一个参数，这些参数是出现在指令后的大括号内的。现在就来实际编译个例子试看看：

```
% example8.tex
\documentclass{report}
\title{\bfseries Aesop Fables}
\author{Aesop\thanks{Thanks to the reader.}
        \and Nobody\thanks{Thanks to nobody}}
\date{\today}
\begin{document}
\maketitle
\tableofcontents
\chapter{Aesop Fables}
\section{The \textsl{Ant} and the \textsl{Dove}}

\itshape
An antwent to the bank of a river to quench its thirst, and
being carried away by the rush of the stream, was on the
point of drowning.
\upshape

A \textsl{Dove} sitting on a tree overhanging the water plucked a
leaf and let it fall into the stream close to her. The \textbf{\textsl{Ant}}
climbed onto it and floated in safety to the bank.

\section{The {\it Dog}\/ in the Manger}

A \textbf{\textit{dog}} lay in a manger, and by his growling and snapping
prevented the oxen from eating the hay which had been
placed for them.

“What a selfish Dog!” said one of them to his companions;
“he cannot eat the hay himself, and yet refuses to allow
those to eat who can.”

\chapter{The \textsc{Eagle} and the Arrow}

An \textsc{eagle} sat on a lofty rock, watching the movements of a
Hare whom he sought to make his prey.

An archer, who saw the \textsc{Eagle} from a place of concealment,
took an accurate aim and wounded him mortally.
\end{document}
```

我们把 title page 的标题改成粗体（请注意，宣告式或旧用法，大括号是把指令和文字整个括住的），把 Dove 改成 slant 斜体，把 dog 改成 italic 粗斜体[2]，把 ant 改成 slant 粗

[2]请注意，这两种斜体是不一样的，slant 是一般正常的字，只是把他倾斜个角度而已，但 italic 则是另一

斜体，把 eagle 改成 small caps。由于章节标题原本就会转换成粗体，所以章节标题的部份，粗体就不必重复设了。

但这里发现例子里第二章标题中的 Eagle 并没有改变字体，而且以 latex 编译时会产生以下的错误（这些讯息也会在 example8.log 中找到）：

```
...
LaTeX Font Warning: Font shape `OT1/cmr/bx/sc' undefined
(Font)                  using `OT1/cmr/bx/n' instead on input line 4.
...
LaTeX Font Warning: Some font shapes were not available, defaults substituted.
...
```

现在我们看到了前面所谈的属性简称，这在 LaTeX 就会使用这种属性来表示而发出讯息，这里 OT1/cmr/bx/sc 就表示了 OT1 编码，Computer Modern Roman 字族，Bold extended 系列，而且是 small caps 形状的字型，错误讯息显示，他并没有定义，因此，这个字型将会使用默认的字型来代替，这里就是以 n 正常形状的 bx 系列字型来替代。所以，字型指令并不是都可以随意组合的，有些是根本就没有这种字型，有些则是没有用宏去定义好，这样 LaTeX 就取不到字了，但别担心，顶多就是使用默认的字型罢了！

另一个很奇怪的地方，就是第一章、第二节的标题，为什么是 {\it Dog}\/ in the...}？这个插入的 \/ 是什么东西？这是 TeX 系统调整斜体字（包括 iatlic 及 slanted）和正常字之间的空白的一个指令，称为 italic correction。这样，在斜体字和正常字之间的空白才会正常。那为什么其他的斜体指令没有加这个调整呢？这是因为 LaTeX 宏在设计时就有考虑到这个问题，所以 \textit{} 这类标准指令都会自动调整 italic correction，不必由我们手动调整。

另外，章节标题本就会自动转换成粗体，标题上的 dog 为什么没有变粗体？这在前面有提到过，这种旧用法有时是无法复合使用的，粗体又斜体的指令会用不上来。因此，建议尽量使用 LaTeX 的第一种标准指令来改变字型。使用 {\it ...} 或 {\itshape ...}[3] 这种指令的话，就得时时注意 italic correction 的问题，也得注意是否可以复合使用指令的问题，所以，还是不要偷懒的好。:-)

后下是排版出来的结果：

http://edt1023.sayya.org/tex/latex123/example8.tex
http://edt1023.sayya.org/tex/latex123/example8.pdf

种独特的字型设计。

[3]宣告式指令可以复合使用，但仍然会需要手动做 italic correction。

### 4.7.3 相对字号的调整

接下来谈最后一个字号属性的调整，这在使用上比较单纯，只要知道指令就可以马上拿来使用。但是 TEX/LATEX 系统中，谈到字型，里头一堆地雷，例如前面谈到正常的内文字号是 10pt，现在如果想制作海报，需要 64pt 的字的时候就会发现，设不出来了！正常 LATEX 的定义，字型的大小范围是在 5–24.88pt 之间，超出这个范围的字需要其他的 package 的帮忙。[4]

这里我们先来看看内文 10pt 时各种字号指令、实际例子及其大小（这是相对大小， 会随内文预设字号而自动调整）：[5]

| 指令 | 宵院例子 | 效果 | 宵院的大小（贴敷） |
|---|---|---|---|
| \tiny | {\tiny tiny} | tiny | 5pt |
| \scriptsize | {\scriptsize scriptsize} | scriptsize | 7pt |
| \footnotesize | {\footnotesize footnotesize} | footnotesize | 8pt |
| \small | {\small small} | small | 9pt |
| \normalsize | {\normalsize normalsize} | normalsize | 10pt |
| \large | {\large large} | large | 12pt |
| \Large | {\Large Large} | Large | 14.4pt |
| \LARGE | {\LARGE LARGE} | LARGE | 17.28pt |
| \huge | {\huge huge} | huge | 20.74pt |
| \Huge | {\Huge Huge} | Huge | 24.88pt |

这些字号指令也可以当成环境来使用，例如：

```
\begin{small}
  本文内容
\end{small}
```

这样用也是可以的。

### 4.7.4 绝对字号的调整

通常字型的大小，使用上一节所说的相对字号来调整会比较方便，而且对于整个版面 的配合也会比较恰当，例如行距也会跟着做适当的调整，如果自行用绝对字号的方法 来调整字号的话，常常会造成行距不一致的情形，因此，如非必要，应尽量避免。

[4]这是 LATEX 本身宏定义的问题，因为他主要是针对一般性文件及书籍，TEX 本身的能力，可以让字型放大到 2047pt。

[5]请注意，本文 pdf 格式内文使用 12pt 字号，列表及其中的例子，是由另外 10pt 预设字号所制作的 eps 图档引入，以免失真。

但有时候就是需要做这样的调整，例如本文封面的字号，纵使是 LaTeX 默认的最大字型也觉得稍小了点，这就要另外引入 package 调整了。

这里我们使用 type1cm package 来调整。当然，得使用 Type 1 字型，才可以达到无段放大、缩小的目的[6]，而这个 package 也是配合 Type 1 字型使用的。个别放大的 pk 点阵字，这里就不讨论了，目前绝大部份的 Computer Modern 字型都已有 Type 1 的 free 版本，而且各个 TeX distribution 都会附上，使用上会较方便。以下是 type1cm 的使用方法：

```
...
\usepackage{type1cm}
...
\fontsize{字号}{行距大小}\seclectfont
...
```

还记得如何引用宏套件吗？请参考第 3 章、第 3.4 节，第 3.4.3 小节的说明。

其中的「字号」就是所要指定的大小，通常以 pt 为单位，当然，要使用其他单位也是可以。「行距大小」也是要一并指定，不可省略。最后的 \seclectfont 是让他发生作用的意思，LaTeX 有些关于字型的较低阶指令，要下 \seclectfont 后才会作用，\fontsize{}{} 正是其中之一。

## 4.8 原文照列

什么是原文照列？一般 LaTeX 遇到倒斜线会认为是一个指令的开始，如果连整个指令都要印出的时候呢？这时就要用到原文照列的指令及环境了。

### 4.8.1 原文照列指令

如果只是一小段的文字要原文照列，那使用指令会比较方便，这个指令就是 \verb|文字内容|，其中的 | 这个符号可以使用其他非字母的符号代替，只要前后相同就行了，例如：\verb+文字内容+ 这样也是可以的。

[6] LaTeX 系统中的字型放大，在 10pt 以上，是以 1.2 的倍数为次方来放大的，因此，正文 10pt 的字号的话，不会有 13pt 这种大小的字型，LaTeX 会选用最相近大小的字型来替代。

### 4.8.2 原文照列环境

如果是一整段的内容要原文照列的话，使用环境会比较方便，那便是 verbatim 环境。不管是哪一种原文照列的情形，预设是使用打字机字族的字型来显示的。底下是一个简单的例子，说明原文照列指令及环境的使用：

```
% example9.tex
\documentclass{article}
\begin{document}
The example of \verb|\verb{}| command and \texttt{verbatim} environment.

\section{\textbackslash{}\texttt{verb} command}

When you want to express you home directory, you can \verb|echo $HOME|
varient to display your home directory in your sh script.

\noindent
\verb*|This is 4 space here.|

\section{\texttt{verbatim} environment}

Here is a sh script to determine if on GNU/Linux system.

\begin{verbatim}
#!/bin/sh
Date=‘date ”%y%m%d”‘
if ‘uname‘ = Linux
then
  Mail=/var/spool/mail/edt1023
  Target=/mnt/hd
else
  Mail=/var/mail/edt1023
  Target=/mnt/pub
fi
\end{verbatim}
\end{document}
```

这里会发现一些奇怪现象，例如 \verb* 那个星号是什么意思呢？就是让空白以 LJ 的方式表示出来的意思，verbatim 环境也是可以这样使用。例如 example9 中的：

```
\verb*|This is 4 space here.|
```

也可以写成：

```
\begin{verbatim*}
This is 4 space here.
\end{verbatim*}
```

差别在于，环境的上下行会多空出个空白行出来。

另外，标题为什么不使用 \verb|\verb| 就好了呢？原因是原文照列的指令和环境都不能当做其他指令的参数，标题本身就是一个指令，所以 \verb 不能在里头。

使用 \textbackslash 这么长的叙述，而不用 $\backslash$ 这个简单的方式，原因是这个文稿有使用 LaTeX2HTML 来转成 HTML 格式，使用 LaTeX 的替代表示法会转成一般的符号，但使用后者的方式则会转成图档，所以这里就使用 LaTeX 的替代表示法了。

底下是排版出来的结果：

http://edt1023.sayya.org/tex/latex123/example9.tex
http://edt1023.sayya.org/tex/latex123/example9.pdf

## 4.9 加入中文

这里只说明如何使用 CJK package 的情形，原因是一般 TeX distribution 会附上（有些发行套件并没有附上，这时只好自行安装了）。CJK package 是把中文的部份包在一个环境里头，在这个环境内就可以使用中文，离开这个环境就又回复到原本的英文环境，底下由例子来说明。

```
\documentclass{article}
\usepackage{CJK} % 使用 CJK 宏套件
\begin{document}
% 进入 CJK 环境，并使用 Big-5 码及 hwmm 这个字型
\begin{CJK}{Bg5}{hwmm}
\section{CJK 宏套件}
这是一个测试，关于 CJK package 的测试。
\section{桃花源记节录}
初狭，纔通人；复行数十步，豁然开朗。土地平旷，屋舍俨然。有良田、美池、%
桑、竹之属，阡陌交通，鸡犬相闻。其中往来种作，男女衣着，悉如外人；黄发、%
垂髫， 并怡然自乐。见渔人，乃大惊，问所从来；具答之，便要还家，设酒、杀鸡、% 作
食。村中闻有此人，咸来问讯。自云：「先世避秦时乱，率妻子邑人来此绝境，%
不复出焉；遂与外人闲隔。」问今是何世；乃不知有汉，无论魏、晋。此人一一% 为
具言所闻，皆叹惋。余人各复延至其家，皆出酒食。停数日，辞去。此中人语% 云：
「不足为外人道也。」
\end{CJK}
\end{document}
```

就这么简单。只是编译要改由 bg5latex 而不是原来的 latex 指令，这是为了避开我们 Big-5 码的一些特殊码的关系，还记得为何每行最后要加个百分号 % 吗？这样才不会插入英文的字间空白。编译好的例子如下：

http://edt1023.sayya.org/tex/latex123/example10.tex
http://edt1023.sayya.org/tex/latex123/example10.pdf

详细的 CJK package 的使用中文说明，请参考 CJK package 所附的文件及〈我的 CJK〉一文：

http://edt1023.sayya.org/tex/mycjk/mycjk.html
http://edt1023.sayya.org/tex/mycjk/mycjk.pdf

# 第 5 章

# 空间与位置

前一章曾提到过，学会控制空间就学会排版了！Knuth 教授在他的 *The TEXbook* 一书中也曾形容使用 TEX 排版的情形：一个版面就像一个含有胶水（glue）的页面，然后每一个要排版的内容就是各种不同的 box，在这些 box 还没有固定正确位置时，都是可以移动的（胶水还没有干），一旦排版完成，胶水就干了，于是每个 box 的位置就固定无法再移动了，除非又从头再来。

一个字母、一个单字、一个句子、一个段落、一个符号、一个图形、一个表格都可能构成 一个 TEX 的 box，甚至 box 中还有 box 的情形。这章想讨论的，就是这个 box 如何安置他们到正确的位置，让每个 box 之间的空间都能达到恰到好处，所以，到底是在控制 boxes 的属性、位置，还是调整 glue 的空间，就看各位怎么去看待了（请注意，box 不一定是可见的！在 TEX 里头，glue 是可以调整的。）。

我们前面所讨论到的英文句点后空白的调整、italic correction、\linespread 及 \parindent 这些都是在调整 glue。通常，在 LATEX 系统里头，指定单位常常不会是绝对固定的，会视情形做小限度的自动微调，这是版面空间配置上的需要。

## 5.1 LATEX 中使用的度量单位

要精确描述和调整 LATEX 中的空间及位置，我们必需要有个标准的度量单位。以下都是在 LATEX 常会用到的单位。这里有绝对单位及相对单位之分，除非必要，不然，一般是建议使用相对单位，原因是，他会随着文稿字号改变时跟着做适当的调整。当然，在很讲 求精确、固定大小的显示时，就得使用绝对单位了。

这里如果是阅览 HTML 格式版本，请另参考 PDF 格式版本，以免表示上失真。以下表格中所画出来的长度仅供参考用。

### 5.1.1 绝对单位

| 单位名称 | 意义 | 长度 |
|---|---|---|
| pt | point, 1/72.27 inch | |
| bp | Adobe big point, 1/72 inch | |
| pc | pica, 12pt | |
| mm | millimeter, 1/25.4 inch | |
| cm | centimeter, 10mm | |
| in | inch, 25.4mm | |

这里要注意的是 TEX/LATEX 系统中所谓的点（point），指的是一般的 printer point，也就是 1/72.27 inch，但在 Adobe 的规格中，例如 PostScript 语言中的所谓点，他是 big point，等于 1/72 inch（小数点的部份舍去了），会比一般的 print point 稍微大一点点。

### 5.1.2 相对单位

| 单位名称 | 意义 | 长度 |
|---|---|---|
| em | 约正在使用字型字母 M 的宽度 | ⊢⊣ |
| ex | 约正在使用字型字母 x 的高度 | ⊢ |

在 TEX 里头所谓的 em，其实，精确而言是指在 Knuth 教授设计的 Computer Modern 字型里头的 em-dash 的宽度，由于字母 M 实际上是包在字型上所谓的 em-square 假想方格中，而 em 所指的宽度是指这个 em-square 的宽度，但字母 M 本身并不全占有这个 em-square，因此这样就会造成差异了。所以以字母 M 的宽度来说明的话容易有疑义。LATEX 有个指令 \quad 这就是产生一个正确 em 的宽度的空白，所以在 Knuth 教授的 *The TEXbook* 中，说明 em 就直接说他是一个 'quad' 的宽度。

## 5.2 版面大小

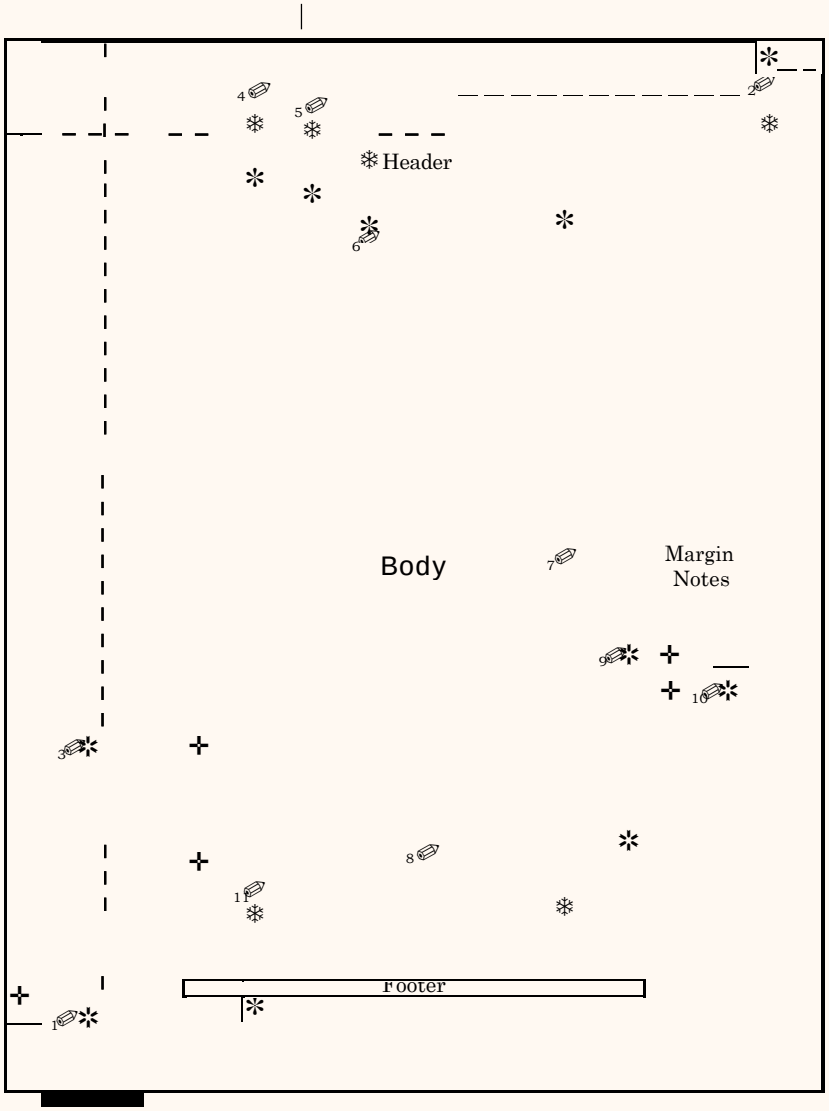
我们对于所能控制的一整张纸的范围都可以称为版面。当然，我们的内文（body）并不是占满整张纸的范围，上下左右都会留有一定的空白。小时候在宣纸上练习写毛笔，老一辈的都会要我们留「天地」，这就是指内文四周的空白，除了视觉上的理由，大概也是人生的哲理吧？:-)

在编辑上，也有人称内文（body）的部份为「版心」或「版口」，四周的空白部份，则称为「版边」。突破版心、版边的设计，就称之为「出血」，例如，以背景图布满整张纸当

做是背景的场合，以这个背景图而言，就无所谓版边了。但这在 LaTeX 通常是不会有这种情况出现，除非特意去指定内文和纸张大小同样范围。

当然，在内文以外的空白，也并非全是空白，他包含了页足（footer），页眉（header）及边注（marginal note）的部份，记载关于页数、批注等信息。

### 5.2.1 版面图解




1 one inch + \hoffset
2 one inch + \voffset
3 \oddsidemargin = 62pt
4 \topmargin = 16pt
5 \headheight = 12pt
6 \headsep = 25pt
7 \textheight = 550pt
8 \textwidth = 345pt
9 \marginparsep = 11pt
10 \marginparwidth = 65pt
\marginparpush = 5pt (not shown)
11 \footskip = 30pt
\hoffset = 0pt
\voffset = 0pt
\paperwidth = 614pt
\paperheight = 794pt

这里所谓的纸张大小，指的是 paperwidth 和 paperheight 所围成的范围，并非实际上手

上拿到的纸张大小，实际在手上的纸张通常会略大于我们这里的所谓纸张，所以，正式列印时，还需做微调或截切才会是真正的这里所谓的纸张大小（版面大小）。

这是 10pt 内文大小，如果不指定纸张的话，LaTeX 预设会使用美式 letterpaper 的大小，如要使用欧、日式的 a4paper 的话，要另行指定。我们可以稍微看一下 LaTeX 预设是如何安排版面空间的。其中 Header（页眉）、Footer（页足）及边注的空间是不含括在内文 Body 里头的，这里是只是单面的图，如果是双面的话，那偶数页和奇数页的边注是要左右对换的，也就是说这个图是奇数页，偶数页的话，边注是在左边。

这里我们来看一下这些值所代表的意义：

| 指令（值） | 意义 |
|---|---|
| \paperwidth | 纸张的宽度 |
| \paperheight | 纸张的高度 |
| \textwidth | 内文（body）的宽度 |
| \textheight | 内文（body）的高度 |
| \headheight | 页眉（header）长度 |
| \headsep | 页眉与内文间的距离 |
| \footskip | 内文底至页足底之距离 |
| \topmargin | 页眉上方的空白 |
| \marginparwidth | 边注的宽度 |
| \marginparsep | 边注与内文的距离 |
| \marginparpush | 两边注间距 |
| \oddsidemargin | 内文左边的空白大小 |
| \hoffset | 微调版面在实际纸张的左右位置 |
| \voffset | 微调版面在实际纸张的上下位置 |

\hoffset 及 \voffset 就是在调整版面在实际纸张上的正确位置，这样印出来的时候才会在实际纸张的中央。

头昏了吗？这很正常，因为 LaTeX 的版面设定对初接触的人来说，是恶名昭彰的困难、麻烦，因此这里不多谈他的设定，刚开始实在没有必要把时间花在这个地方。如果实际想 调整版面，建议使用 geometry package。举个例子，想让各边缘是 2cm 就好，那只要在 preamble 区设定：

```
\usepackage[margin=2cm]{geometry}
```

就可以了，如果以 12pt 大小的字，a4paper 纸张大小的设定的话，以中文而言，大约是每行 40 个中文字，这是内文的宽度。可以视情形自行调整 margin 的值就行了。我们很希望，下一版的 LaTeX 能对这方面做改善，以方便使用者设定。

### 5.2.2 纸张大小

| 纸张 | 大小 | 纸张 | 大小 |
|---|---|---|---|
| a4paper | 21x29.7cm | letterpaper | 8.5x11in |
| a5paper | 14.8x21cm | legalpaper | 8.5x14in |
| b5paper | 17.6x25cm | executivepaper | 7.25x10.5in |

至于如何指定纸张大小，这里先简单说明一下这篇文章的设定，谈到 LaTeX 的文稿类别时会再详细说明。

```
% 本文的设类别设定
\documentclass[12pt,a4paper]{report}
```

所以，这篇文章使用的是 a4paper，内文字型的大小是 12pt。方括号的参数是选项，可以省略，如果省略的话，默认值就是 10pt/letterpaper。

## 5.3 调整横向空间

这里的横向空间，例如 ~ 这单字间正常空白，及 \quad 这个 em 宽度空白，都是在调整横向的空白。但如果是要更大、或更小的空白时该如何调整呢？底下我们就来看看 LaTeX 中有什么控制指令可以运用：

### 5.3.1 调整横向空间的指令

| 指令 | 意义 |
|---|---|
| \hspace{单位} | 向右空出某个度量单位的空白，如果是负数，则是向左 |
| \hfill | 让左右两旁的文字往两边扩张至一个行宽为止 |
| \quad | 空出一个 em 单位的空白 |
| \qquad | 空出二个 em 单位的空白 |
| \thinspace | 空出 1/12 个 em 单位的空白 |
| \enspace | 空出 1/2 个 em 单位的空白 |
| \dotfill | 作用和 \hfill 相同，只是空白变成点 |
| \hrulefill | 作用和 \hfill 相同，只是空白变成一横线 |
| \centering | 此指令以后的文字将会居中排列，左右沿将不切齐 |
| \raggedright | 此指令以后的文字将会居左排列，右沿将不切齐 |
| \raggedleft | 此指令以后的文字将会居右排列，左沿将不切齐 |
| \centerline{} | 将大括号内的文字居中排列 |

一行的行首使用 \hspace{单位} 时将会失效，这时可以加个星号，例如 \hspace*{3em}。

在使用 \centerline{} 的场合，对于短文句很方便，因为他不会影响以下的文字，但他也不会折行，甚至加入换行符号也无效。所以如果文句长度超过一行的行宽，他会超出边界，甚至就首尾的文字就看不到了。

其中 \thinspace 又可以使用简化的 \, 来代替，主要是用在引号中又有引号的情形，通常这种情形，两引号之间要有个间隔，以便区分，例如：

“\,‘Superman’, he said.”

表现出来会是：

“ ‘Superman’, he said.”

这样才能区分出不同引号，否则会变成一个连三个 grave accent 的引号。请注意，由于这个指令是由非字母符号构成，所以，他的作用范围在遇到符号本身后就结束了，后面不必空白或以大括号来限制其作用范围，就好像曾提到过的 \@ 一样的情形。

### 5.3.2 调整横向空间的环境

| | |
|---|---|
| \begin{center}...\end{center} | 让这个环境内的内容置中 |
| \begin{flushleft}...\end{flushleft} | 让这个环境内的内容靠左 |
| \begin{flushright}...\end{flushright} | 让这个环境内的内容靠右 |
| \begin{raggedright}...\end{raggedright} | 让这个环境内的内容靠左，右沿将不切齐 |
| \begin{raggedleft}...\end{raggedleft} | 让这个环境内的内容靠右，左沿将不切齐 |

进入环境，和上一节提到的指令，两者有什么不同呢？最大的不同是，这可以方便的指定一个范围的文句让他作用，而不会影响环境以外的文句。其次，进入环境，纵使和上下行连在一起，没有空出空白行，他也会自动的在上、下行空出个空白行出来，使用指令的话则不会。

咦！这里怎么又有个 raggedright 及 raggedleft？原来他也是可以当环境来使用。由于这两个指令会使以下的内容的左、右沿不切齐，因此使用上要非常小心，除非本来就想让内文的左、右沿不切齐，否则，最好是使用有范围限制的方式。当然，如果这两个指令是 用在某个其他环境范围内，他的作用也将仅限于这个环境内，不会影响这个环境外的文句。

### 5.3.3 引文环境

引文通常就是引用他人的文句，在引文的段落，两旁都会出现内缩的情形，以便和正文相区隔，这也是一种空间的配置，可增加文章的易读性。在 LaTeX 里头有三种引文环境：quote, quotation, verse。这三者看起来很像，但有些微的差异。

| 环境 | 适用时机 | 特性 |
|---|---|---|
| quote | 较短的短引文 | 每个段落第一行不内缩 |
| quotation | 多个段落的长引文 | 每个段落第一行会内缩 |
| verse | 诗歌、词引文 | 每个段落的第一行不内缩，但第二行起会内缩 |

在 verse 的情形，通常会使用 \\ 来换行以便控制每一行的宽度。而且段落间距将不受外在设定的影响，其中 quote 和 verse 环境会预插入适当的段落间距，而 quotation 环境则不会。

底下我们来看看调整横向空间的一个综合实例：

```
% example11.tex
\documentclass{article}
\usepackage{CJK}
\begin{document}
\begin{CJK}{Bg5}{hwmm}
\section{hspace}
\hspace*{2em}这是一个横向空间调整的测试。\\
这是一个\hspace{2em}横向空间调整的测试。\\
这是一个 \hspace{2em} 横向空间调整的测试。
\section{hfill}
这是一个\hfill{}横向空间调整的测试。
\section{quad}
这是一个\quad{}横向空间调整的测试。\\
这是一个 \quad{} 横向空间调整的测试。\\
这是一个\qquad{}横向空间调整的测试。
\section{dotfill}
这是一个\dotfill{}横向空间调整的测试。\\
这是一个 \dotfill{} 横向空间调整的测试。
\section{hrulefill}
这是一个\hrulefill{}横向空间调整的测试。
\section{center}
\begin{center}
这是一个横向空间调整的测试。
\end{center}
\section{flushleft}
\begin{flushleft}
这是一个横向空间调整的测试。
\end{flushleft}
\section{flushright}
```

```
\begin{flushright}
这是一个横向空间调整的测试。
\end{flushright}
\section{quote}
这是节录自伊索寓言的节录故事：
\begin{quote}
An antwent to the bank of a river to quench its thirst, and
being carried away by the rush of the stream, was on the
point of drowning.

A Dove sitting on a tree overhanging the water plucked a
leaf and let it fall into the stream close to her. The Ant
climbed onto it and floated in safety to the bank.
\end{quote}
\section{quotation}
这是节录自伊索寓言的节录故事：
\begin{quotation}
An antwent to the bank of a river to quench its thirst, and
being carried away by the rush of the stream, was on the
point of drowning.

A Dove sitting on a tree overhanging the water plucked a
leaf and let it fall into the stream close to her. The Ant
climbed onto it and floated in safety to the bank.
\end{quotation}
\section{verse}
这是节录自伊索寓言的节录故事，这是节录自伊索寓言的节录故事，%
这是节录自伊索寓言的节录故事，这是节录自伊索寓言的节录故事：
\begin{verse}
An antwent to the bank of a river to quench its thirst, and
being carried away by the rush of the stream, was on the
point of drowning.

A Dove sitting on a tree overhanging the water plucked a
leaf and let it fall into the stream close to her. The Ant
climbed onto it and floated in safety to the bank.
\end{verse}
\section{centering}
\centering
这是一个横向空间调整的测试。\\ % 这里要换行，否则会是 \raggedright 的作用
\raggedright
\section{centerline}
\centerline{这是一个横向空间调整的测试。}
\section{raggedright}
\raggedright
这是一个横向空间调整的测试。
\section{raggedleft}
\raggedleft
```

```
这是一个横向空间调整的测试。
\end{CJK}
\end{document}
```

编译好的结果如下：

http://edt1023.sayya.org/tex/latex123/example11.tex
http://edt1023.sayya.org/tex/latex123/example11.pdf

要注意是指令前后的空白，像 \hspace, \dotfill, \hrulell 这类指令，指令前后空白都会算进去的。\quad, \qquad 这类指令，则后面的空白也是会算入的。另外，由例子中可以看出来，一个 em 的宽度，大约是一个中文字的宽度，所以，我们预设使用 10pt 的字，这个 em 宽度就相当于 10pt 的宽度，所以，我们在第一行插入了 2em 宽度的空白，也就好像是内缩了两个中文字一样。

## 5.4 调整纵向空间

| | |
|---|---|
| \vspace{单位} | 向下空出某个单位的空白（行），负数则是向上 |
| \bigskip | 产生 12pt（11–12pt）的垂直空白（行） |
| \medskip | 产生 6pt（5–7pt）的垂直空白（行） |
| \smallskip | 产生 3pt（2–4pt）的垂直空白（行） |
| \vfill | 和 \hfill 类似，作用是将某段落向上顶，或往下挤 |
| \parskip单位= | 调整全文每个段落间的距离为某个单位 |

其中的 \bigskip, \medskip, \smallskip 并非固定的，他们会视上下文脉络的需要自动做微调，以达到一整页较一致的空间配置。\vspace 如果是出现在一页的第一行或最后一行时，将会失去作用，这时可以加个星号，\vspace*{单位}。

为了维持版面的一致性，使用纵向空间调整的指令时要特别留意，例如章节标题上下的空间、各段落间的空间，进入环境前后所空出的空间，这都有一个固定值，LaTeX 会自动去调整，不必由使用者自行动手，除非是封面这种单独页。所以，使用纵向空间调整指令时，要非常注意整体的一致性，这也是排版上的一个很重要的原则。

这里举这篇文章的内页封面为例来综合说明，横、纵向空间的运用。还记得第 4.3 节的 title page 的指令吗？其实我们也可以自行设计一个独立的内页封面，使用的是 LaTeX 本身的 titlepage 环境。这里的图档引用是我们还没有学习到的，没关系，只要大原则抓住就行了。

```
% example12.tex
```

```
\documentclass[12pt,a4paper]{report}
\usepackage{CJK}        % 引入所需要的 packages
\usepackage{graphicx}
\begin{document}
\begin{CJK}{Bg5}{hwmm}
\begin{titlepage}       % 使用 titlepage 环境
\vspace*{5ex}
  \begin{flushright} % 大标题靠右
    \Huge\textbf{大家来学 \LaTeX}
  \end{flushright}
  \rule{\textwidth}{.256ex}
  \begin{flushleft}     %  版本号码及日期靠左，和大标题之间以一横线隔开
    Version 0.1 draft\\
    \today
  \end{flushleft}       % 图档位于中央偏左
  \vspace{8ex}          % 空出 8ex 的垂直空间
  \hspace{2em}\includegraphics[scale=.75]{cover2.1} % 引入图档，并将这个
  \vspace{8ex}                                      % 图档横向右移 2em
  \begin{flushright} % 作者信息靠右
    By Edward G.J. Lee 李 果 正 \\ Email:
    \texttt{edt1023@info.sayya.org}
  \end{flushright}
\end{titlepage}
\end{CJK}
\end{document}
```

由于配合版面的问题，其中有一些数据有更动，而且也省略了一些我们还没有学习到的 packages，但大结构则和原始文稿一样。所以，和这篇文章的 PDF 格式比较会发现，大标题的字体小了一点，而且没有颜色，也没有超链接。

使用 titlepage 环境后，在 report/book 文稿他会自成一没有页数的单独页，在 article 类别，因为会和内文连接，所以，在选项的部份要多加一个 titlepage 的选项。另外，在 titlepage 里头就不能再使用 \title, \author 指令了，在本文的地方也不必再下 \maketitle 指令。

编译好的例子如下：

http://edt1023.sayya.org/tex/latex123/example12.tex
http://edt1023.sayya.org/tex/latex123/example12.pdf

这里要特别说明的是，引入图档要使用 graphicx package（这是最常用的，也有其他的方法来引用），引入的指令是 \includegraphics，这个我们会在第 9 章会讨论，在这里我们把图档缩小成为原图的 75%，否则整个封面会超出一页。这个图档是由 METAPOST 档案所编译而来的，他是一种 eps 图档（简单的说，是含有边界数据去除不必要周围空白的

ps 档，方便引入或输出至文稿中），编译的方法如下：

```
mpost cover2.mp
```

这样就会产生 cover2.1 这个 eps 图档，这样就可以直接引入了。他的原始码及 eps 图档在：

http://edt1023.sayya.org/tex/latex123/cover2.mp
http://edt1023.sayya.org/tex/latex123/cover2.1

## 5.5 条列环境

条列环境也是属于一种空间的控制，他把一些文字按一定的方式来排列，条列环境中一些起头的符号、文数字或字符串，我们称之为项目卷标（item label），利用这些不一样的排列位置及不一样的项目卷标起头来叙述文句，就可以达到醒目的作用。这是以章节分隔以外，相当常用让内容一目了然的方法，建议多多利用。请千万记得，环境中还可以有环境，而且以下三种的条列方式可以混合交叉使用。

### 5.5.1 项目式条列环境（itemize）

这是以符号来起头醒目的一种条列方式。例如：

```
\begin{itemize}
\item 第一大项，这里是第一大项。
\item 第二大项，这里是第二大项。
 \begin{itemize}
 \item 第一小项，这里是第一小项。
 \item 第二小项，这里是第二小项。
 \end{itemize}
\item 第三大项，这里是第三大项。
\item 第四大项，这里是第四大项。
\end{itemize}
```

排版出来会变成：

- 第一大项，这里是第一大项。
- 第二大项，这里是第二大项。
  - 第一小项，这里是第一小项。
  - 第二小项，这里是第二小项。
- 第三大项，这里是第三大项。
- 第四大项，这里是第四大项。

### 5.5.2 列举式条列环境（enumerate）

这是以数目字或字母或罗马数字来起头醒目的条列方式。同样的例子，改成 enumerate 的话，会排版成：

1. 第一大项，这里是第一大项。
2. 第二大项，这里是第二大项。
   (a) 第一小项，这里是第一小项。
   (b) 第二小项，这里是第二小项。
3. 第三大项，这里是第三大项。
4. 第四大项，这里是第四大项。

### 5.5.3 叙述式条列环境（description）

这是以一个简短文字叙述来起头醒目的条列方式。再把他改成 description 的话，他是以粗体文字来起头醒目的，这些文字要用方括号括住。

```
\begin{description}
\item[第一大项] 这里是第一大项。
\item[第二大项] 这里是第二大项。
  \begin{description}
  \item[第一小项] 这里是第一小项。
  \item[第二小项] 这里是第二小项。
  \end{description}
\item[第三大项] 这里是第三大项。
\item[第四大项] 这里是第四大项。
\end{description}
```

排版出来的结果是：

**第一大项** 这里是第一大项。
**第二大项** 这里是第二大项。
  **第一小项** 这里是第一小项。
  **第二小项** 这里是第二小项。
**第三大项** 这里是第三大项。
**第四大项** 这里是第四大项。

要注意的是，不管哪一种的条列环境，每个项目（item）的文字叙述会自动折行，这相当方便，使用者只要把条列的结构弄妥，专心打每个项目的内容就成了。而且，如果使用方括号括住一些字符、字符串或符号，那带头的标示将会是这些字符、字符串或符号，如果是列举式的条列方式，那么有方括号的将不被编号，会自动跳过，编号顺序则会自动顺延。

## 5.6 线框

线框在排版上占有一定的地位，因为他可以区隔不同的空间，也可以让某些部份突显出来。但是，过多的线框也是会有喧宾夺主的不好副作用，使用上可能要适可而止。这里我 们要谈的是单纯的线框，表格也是线框的一种应用，我们将会在第 9 章，再来谈表格的处理。

### 5.6.1 直线（rule）

直线也是属于方框的一种，就是一个实体长方形，只不过，他的高或宽（线的组细）只有一点点，所以，看起来就一像直线罢了。

```
\rule[上下位置(单位)]{宽}{高}
```

宽及高应不必多做解释，那个上下位置是什么呢？这个上下位置是和基线（baseline）在比较的，如果没有指定，那就是在基线的位置，如果有指定，就依正负值调整和基线的相对位置，正值由基线向上调整，负值则由基线向下调整。这里来看个个实例就了解了：

```
% example13.tex
\documentclass{article}
\parskip=3pt
\parindent=0pt
\begin{document}
This is a line.          % 画高 1pt 宽 3cm 的横线。
\rule{3cm}{1pt}
\rule[1ex]{3cm}{1pt}
\rule[-1ex]{3cm}{1pt}

\rule{1pt}{3cm}          % 画高 3cm 宽 1pt 的直线。

\rule{3cm}{0pt}TEST. % 把 TEST 向右推 3cm。

\rule{2cm}{3cm}          % 画高 3cm 宽 2cm 的实体方框。

\textcolor{blue}{This is color lines.}
\textcolor{red}{\rule{3cm}{1pt}}                % 有颜色的线框。
\textcolor{green}{\rule[1ex]{3cm}{1pt}}
\textcolor{blue}{\rule[-1ex]{3cm}{1pt}}
\end{document}
```

编译好的例子在：

http://edt1023.sayya.org/tex/latex123/example13.tex
http://edt1023.sayya.org/tex/latex123/example13.pdf

由例子中可以看得出来，这个画线指令并不是单单纵、横画线而已，灵活运用的话，可以做出许多不同的效果，例如方条图之类的。

### 5.6.2 文字底线（underline）

有时候我们希望在书写文字的同时，也在其下画线。LaTeX 有现成的指令可以使用，那就是 \underline{文字}，会在文字底线的部份同时画上线条。当然，使用上常常会和现在的横线搞混，因此使用时要特别留意整个页面上是否已经存在了许多横线。

### 5.6.3 方框（box）

这里主要指的文字方框，绘图上的一般框形在第 9 章时再来讨论。这一章的开头就已谈到，整篇文章是由一个个的 box 所构成的，这里的方框则是一个典型的 box，他的地位，当然是和前面所说的 box 一样，TeX/LaTeX 会把他视为一个单一的字母来处理。

我们先来看看最简单的方框的指令：

| 指令 | 意义及作用 |
|---|---|
| \frame{文字} | 将文字内容以可见的方框框住，方框和文字间没有间距 |
| \fbox{文字} | 将文字内容以可见的方框框住，方框和文字间有一定间距 |
| \mbox{文字} | 作用和 \fbox 同，但方框不可见 |

以上的指令都可以形成方框文字。

方框也有可以调整的指令：

| 指令 | 意义及作用 |
|---|---|
| \framebox[宽度][对齐方式]{文字} | 与 \fbox 同，但可指定宽度及对齐方式 |
| \makebox[宽度][对齐方式]{文字} | 与 \mbox 同，但可指定宽度及对齐方式 |

这里的宽度指的是方框的宽度，不指定的话，作用就如同 \fbox 及 \mbox 一样，以包住整个文字范围为宽度，由于我们不容易测知文字内容的宽度有多少，因此宽度的指定可以使用下列相对单位的方式：

| 宽度 | 作用 |
|---|---|
| \width | 这个宽度就是 \fbox 围住时的方框宽度 |
| \height | 这是正常基线至框顶的高度 |
| \depth | 这是正常基线至框底的高度 |
| \totalheight | 这是 \height 和 \depth 之总和 |

例如，我们指定 2\width 他的意思就是以 \fbox 包住此文字内容时，其方框宽度的二倍宽。参数里头的对齐方式，可有下列几种：

| 对齐方式 | 作用 |
|---|---|
| c | center，文字位于方框中央，这是默认值 |
| l | flushleft，文字位于方框左方 |
| r | ushright，文字位于方框右方 |
| s | stretch，文字平均分分布于方框中 |

不指定的话，当然就是位于方框中央位置了。我们也可以指定可见方框的线的粗细，及方框和文字间的间隔距离：

| 指令 | 作用 |
|---|---|
| \fboxrule=单位 | 指定方框线条粗细 |
| \fboxsep=单位 | 指定文字和框缘的间距 |

请注意，这不会影响 \frame{}。如果想特别的置放方框的位置，也可以使用 \raisebox 指令：

```
\raisebox{上下位置(单位)}[深度][高度]{文字内容}
```

这里的上下位置的意义和 \rule 指令里头的一样，只不过，这里的上下位置一定要指定，没有默认值，不指定会编译错误。请千万记得，方框中当然是还可以有方框的。我们现在就来看一个综合的实例：

```
% example14.tex
\documentclass{article}
\parskip=3ex
\parindent=0pt
\begin{document}
\frame{This is frame.}
\mbox{This is mbox.}
\fbox{This is fbox.}

\framebox{This is a framebox with no argumant.}

\framebox[1.5\width]{This is a framebox.}

\framebox[1.5\width][l]{This is a framebox with  \texttt{l}.}
```

```
\framebox[1.5\width][r]{This is a framebox with \texttt{r}.}

\framebox[1.5\width][s]{This is a framebox with \texttt{s}.}

This is baseline.
\raisebox{3ex}[5\height]{This is a raisebox which lift 3ex.}

This is baseline.
\fbox{\raisebox{-3ex}[5\height]{This is a raisebox which lift $-$3ex.}}

\fboxrule=1.5pt
\fboxsep=8pt
\framebox[1.5\width][s]{This is a framebox with \texttt{s}.}
\end{document}
```

这应无需多加解释，编译好的例子在：

http://edt1023.sayya.org/tex/latex123/example14.tex
http://edt1023.sayya.org/tex/latex123/example14.pdf

### 5.6.4 段落方框

另外，也有用于段落文字的方框，可以控制某个段落出现在某特定的空间、位置。我们常常把这种安排称为迷你版面，把版面内容置于一个不可见的方框当中，当然，这个方框和一般的 box，例如字母，仍然处于同样的地位，LaTeX 会把他当成一个字母单位来处理。

```
\parbox[对齐方式][高度][内文位置]{宽度}{文字内容}

\begin{minipage}[对齐方式][高度][内文位置]{宽度}
  段落内容
\end{minipage}
```

这里的第一个选择性参数「对齐方式」：

| | |
|---|---|
| t | top，段落方框的上沿对齐一行的基线 |
| b | bottom，段落方框的下沿对齐一行的基线 |
| c | center，段落方框的中央对齐一行的基线，这是预设 |

\parbox 指令通常用于较短的文字内容，如果是较长的段落，那使用 minipage 环境会比较方便。这些段落方框和上面所谈到的一些方框最大的不同是，段落方框本就是用来排版小段落文句，因此他会和一般正常文章段落一样的处理，例如他会自动断行，碰到空白行也会起新段落，但各段落预设是不缩排的，而且，在安排上会比一般的段落紧凑，一些边缘的间距会缩减成 0pt。

高度不去指定的话，那就是以整个版面经断行处理后，整个文字段落所形成的高度。至于内文位置指的也是 **t**，**b**，**c** 这些，意思是文字段落在方框内的上下位置，当然，这要有指定方框高度时才会有意义，因此，「内文位置」和「高度」是要同时指定的，要非常注意的是，如果指定了高度，但没有指定内文位置，则默认是等于前面使用的「对齐方式」所指定的参数。

这些参数的使用会有些烦复，但这是弹性所带来的「必要之恶」，不必去死记，只有在第一次接触时把他搞清楚就够了，实际要用到时再来查他的详细参数，常用的指令、环境，大概多查几次就自然而然的记起来了。

# 第 6 章

# LaTeX 的标准文稿类别

这章主要是在说明 LaTeX 文稿的类别（document class）[1]，这是 LaTeX 规范文稿整体结构的方法。使用 class 的用意，就是把版面结构处理和实际文稿分开，这样的最大好处就是维持整篇文章排版结构上的一致性，也使文稿内容更清爽简洁，使用者只要专心于文稿内　容的写作即可，如果 class 定义的好，也可以达到一文多变化又不变更文稿内容的目的，只要把引用的 class 换成别的就可以了，其他的可以不必更动。

目前，LaTeX 有五种标准类别用于一般文件，可用于一般的书信、杂志、期刊、报告及论文。但有些期刊、论文会要求一定的结构，这时得依需求另行订定。因此，也有其他的类别存在，标准类别并不是唯一的。甚至，也可以自行撰写自己的文稿类别。当然，我们一般使用是不需要这么讲究，这里只介绍 LaTeX 的标准类别。而且，如果有和他人交换文稿的需求时，我们应该尽可能的使用流通性较广泛的类别。

另外，也有一些是关于 LaTeX 说明文件及 macro 写作时要用到的类别，这些已超出本篇文章的范围。

## 6.1 LaTeX 类别的宣告

LaTeX 的类别，要在文稿的一开头时就宣告（当然，其上有批注是没有关系的），他的一般格式如下：

```
\documentclass[选择性参数]{类别}
```

选择性参数是可以省略的，但类别名称则不能省，一定要指定一个类别。而且只能只有一个类别。

[1]这在旧的版本称为 style，这两个词意义上差不多，这些都是 TeX延伸出来的宏定义，专门用来定义文章的大结构，以便简化使用上及文稿的内容。

## 6.2 类别的选择性参数

选择性参数可以选择多个，各个选项是以逗点分开的。

1. 10pt, 11pt, 12pt
   指定内文一般正常字的大小，预设是 10pt。其他点数没有外来 package 的帮忙不能指定。

2. a4paper, letterpaper, b5paper, executivepaper, legalpaper
   指定纸张大小，预设是 letterpaper。

3. fleqn
   使数学式靠左对齐，预设是居中对齐。

4. leqno
   使数学式编号靠左，预设是靠右。

5. titlepage, notitlepage
   决定 title page 是否独占一页。预设 article 不独占一页，但 report/book 则会独占一页，在这里是可以指定来变更预设行为，例如 article 文稿，指定 titlepage 的话，那 title page 就会独占一页。

6. onecolumn, twocolumn
   文章单栏或两栏式，预设是单栏，也就是不分栏。

7. twoside, oneside
   是否区分奇偶数页。预设 article/report 不区分，book 则会区分。一般的书籍，在装订的部份，他的中央线称为书脊，偶数页会在打开书籍时的左方，而且其中内 容会偏向书脊的中央（此时是向右），反之，奇数页会在右，内容一样会偏左向书 脊，在 oneside 的情形则不做这样的区分，不管奇偶页都会在纸张的中央部位。

8. landscape
   横向打印或纵向打印，预设纵向（portait）。

9. draft
   草稿式编译，这时图档将不会被引入，可加快编译的速度。不过，如果编译是使用 向量字体的话，编译速度应该是还算很快。但使用 draft 的一个好处是，过长的地方会标示出来。

10. openright, openany
    这是在控制，章的开始是否是奇数页（right-hand page）。在 book 类别，预设章会从奇数页开始，report 类别则不会。article 类别没有章，所以，对此一设定会忽略。

## 6.3 类别的种类

这里只列出一般文章使用的类别，其他特殊宏或 LaTeX 正式文件所使用的类别就不列出了，一般使用，这些类别就足够了。

| 类别 | 一般用途 | 特性 |
|---|---|---|
| article | 一般短文 | 无章，连续页方式的安排，无奇偶数页的区分 |
| report | 较长论文 | 章会起新页，预设无奇偶数页的区分 |
| book | 书籍类 | 章会于奇数页起新页，预设有偶数页的区分 |
| letter | 信件 | 英文信件格式 |
| slides | 幻灯片 | 几乎另用外来套件取代 |
| minimal | 测试及写新类别 | 这是最简单的类别，只规定了内文的宽、高，正常字 |

当然，这些用途并不是固定不变的，得看使用者的安排，不想多花时间、精神的话，那就依 LaTeX 预设的格式去使用，至少就不会太离谱。其中 minimal class 是用来测试用的，或者写新的 class 用的，他完全没有版面的安排，例如，没有章节的结构，各种间距也没有定义，默认的字型是正常字，没有其他的变化，几乎所有的变化要自行去定义。

# 第 7 章

# 宏套件

LaTeX 系统已经好久没有更新，有些部份可能会跟不上实际的脚步，而且有些内定的宏定义，经过大家的使用，发觉并不是那么的顺手，尤其是功能的强化方面，因此这章谈谈如何引用他人已经写好的宏，这很重要，尽量避免重复制造轮子，写 TeX/LaTeX macro可说是很专业的工作，要避免破坏了整体的结构，所以先找看看有什么宏套件可以使用。

## 7.1 一般套件的使用

我们曾在第 3.4.3 小节，页 22，提到过简单宏的引用，事实上，有些宏含有许多的参数来做微调，但是每个宏套件的参数都不会一样，因此，使用套件之前要先看一看他所附上的使用手册。几乎大部份的宏套件都有使用手册，如果是系统上就有的宏，那么这些文件通常会放在：

```
$TEXMF/doc   => Unix-like 系 统
$TEXMF\doc   => DOS/Windows 系统
```

这些目录底下，这些文件会有原始 TeX/LaTeX 文稿，也有编译好的 *.dvi 或 PostScript 档可以阅览，为求方便的话，可以将他们转成 pdf 格式来阅览，原因是可以以关键词来搜寻全文，在查指令、环境时会比较方便。在 Unix-like 系统或 Windows 下的 cygwin 环境的话，可以使用 texdoc 这个指令来阅览，例如：

```
texdoc amsguide    => 阅览 amsguide.dvi 这个档的说明
texdoc -s ams      => 查系统上所有含 ams 字样的文件
```

## 7.2 LaTeX 官方文件中的标准宏套件

底下是 LaTeX 官方文件中所附的标准宏套件。虽然是标准宏套件，但一般情形下，使用这些 packages 的机会并不多，都是有特殊需要时才会引入。

### 7.2.1 alltt

这个宏套件提供 alltt 环境，和 verbatim 环境的作用相同，只是 \，{，} 的作用和一般文章中相同会被 LaTeX 解读。这有什么用呢？这样一来 LaTeX 这个特殊标志也可以使用，也可以让环境中的文字具有颜色，或做其他变化，当然，里头的文字默认仍然是使用 打字机字族的。

### 7.2.2 doc

这是用来写 LaTeX 文件的宏套件，这在使用 ltxdoc 这个 class 的同时就会加载 doc package。由于这不是用于一般的文件使用场合，所以，这里就不多谈，有兴趣的话可自行参考他的文件说明。

### 7.2.3 exscale

由于原先的 Computer Modern font 中的数学延伸符号（cmex）只有 10pt 大小的字型（cmex10），内文放大到 large 以上的字型时，例如放大到 Large 时，有些数学符号仍然会维持一定的大小，这时可以使用这个套件，让这些数学符号也跟着放大，例如积分符 号。exscale 只有字型缩放的定义，因此只要把这个套件在 preamble 区引用就可以了，无需任何指令。

不过，这里要说明一下，在 TeX/LaTeX 里字型放大，有时可能会造成表现失真的情形，尤其是数学式子，为了顾及数学式子中各个字母间的空间安排，cmex10 的设计并不适合拿来放大，可以把 cmr5 放大成 10pt 和真正的 cmr10 来比较就会知道表现出来会不一样，因此，如果考虑精确配合的问题，放大数学式子的字型时可能要考虑一下使用场合， 尤其目前采用向量字更是如此。请试试以下的例子，我们把 cmr5、cmr10 及 cmr12 同样放大到 30pt 来看看结果会不会一样：

```
% test-fonts.tex
```

```
\font\largecmr=cmr12 at 30pt
\largecmr
This is cmr12 at 30pt.

\font\largecmr=cmr10 at 30pt
\largecmr
This is cmr10 at 30pt.

\font\largecmr=cmr5 at30pt
\largecmr
This is cmr5 at 30pt.
\bye
```

请注意，这是 TeX 文稿，不是 LaTeX 文稿，所以要使用 tex 或 pdftex 来编译，他的结果如下，大家可很清楚的看得出来，虽然同样是向量字，但放大时的表现并不会一样：

http://edt1023.sayya.org/tex/latex123/test-fonts.tex
http://edt1023.sayya.org/tex/latex123/test-fonts.pdf

根据 Knuth 教授当初设计 METAFONT，他的理念是一个同样的字型在放大的时候，同一个字，他的笔划置放的相对位置应该要随放大的倍数而稍加调整。因此，假如我们只使用 一种向量字体，用在不同的放大倍率的时候，文字符号间的空间配合会产生不一样的结果，尤其是用在数学式子的时候，更加明显。

当然，实用上 METAFONT 虽然也是一种向量字体，但由于太过于复杂，不适合拿来屏幕显示上用，所以才会退而求其次，转成 pk 位图字体来使用。这也就是为什么同样是 cmr 的字型，会有几种不同点数的独立字型的原因，纵使是向量字也是如此。TeX 已经 20 几岁了，但是，我们的字型技术似乎还是没有完全赶上当初 Knuth 教授的理念。

### 7.2.4 fontenc

在第 4.7.1 小节，页 32，曾提到字型编码的问题。要改变字型编码，可以使用这个 fontenc package。以 T1 font encoding 来说：

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

这样就可以了，但由于一些字型，例如欧洲字符，在原来的 Computer modern Type1 字型中安排不一样，所以，有些部份会使用原始的 METAFONT 字型所转换成的 pk 点阵字，这样的话，一般打印机印出来是差异不大，但如果是想制作成 PDF 格式在荧光幕阅

览的话，字型的表现会变得很丑。理想的话，要安装 cm-super Type1 字型，但是一般用户恐怕自行安装字型会有困难。这在 teTeX 2.x 以后的版本，已经有附上 pxfonts 及 txfonts package 及其字型，所以，如果是新近版本的 teTeX 的话，可以由以下的方式来使用：

```
  ...
\usepackage{txfonts}
\usepackage[T1]{fontenc}
  ...
```

其中 txfonts 是仿真 Times 系列的字型，pxfonts 是仿真 Palatino 系列的字型。当然，这里关于字型的问题有点复杂，这不在这篇文章的讨论范围，只能做简单的说明，如果没有 特殊需要，例如，欧洲字符、一些有重音符号的字母，那使用默认的 OT1 编码就行了， 因为这些套件所附的有些字型，只有一种大小的 Type1 字型在缩放，因此使用上恐怕会有失真的情形。

### 7.2.5 graphpap

这是产生方格纸的宏。他提供了一个指令，可以画方格，可以配合 picture 环境来使用，他的语法是：

```
  ...
\usepackage{graphpap}
  ...
\graphpaper[n](x,y)(x1,y1)
  ...
```

其中的 n 如果省略的话，预设是 10，他指的是方格纸的最小刻度单位。(x,y) 及 (x1,y1) 指的是左下角及右上角的坐标值，例如：

```
\documentclass{article}
\usepackage{graphpap}
\begin{document}
\graphpaper(0,0)(360,360)
\end{document}
```

这样会画出以 10 为最小刻度的方格，编译好的例子如下：

http://edt1023.sayya.org/tex/latex123/test-graphpap.tex
http://edt1023.sayya.org/tex/latex123/test-graphpap.pdf

### 7.2.6 ifthen

TeX 本身是一种排版程序语言，当然会有条件判断式来方便写宏，但如果文稿中也充满了条件判断式，将会使文稿复杂化，难以阅读、维护，因此，一般条件判断式大多数使用 在宏定义，而不是写在文稿当中。这个 package 就是在简化条件判断式，以便也可以方便使用在文稿当中。

ifthen package 提供了 \ifthenelse 指令来做条件判断。他后面有三个参数，第一个是条件式，第二个是条件为真的时候要执行的内容，第三个是条件为伪的时候要执行的内容。这里不多谈他的使用，底下只提供一个实例片段：

```
  ...
\usepackage{ifthen}
  ...
\ifthenelse{\isodd{\thepage}}%
  {\setlength{\leftmargin}{10pt}}%
  {\setlength{\leftmargin}{0pt}}
  ...
```

这样奇数页时，leftmargin 会设为 10pt，偶数页时则为 0pt。后面加 % 代表，这三行是一整行，其间没有空白。

### 7.2.7 inputenc

由于 fontenc package 的一些字型编码安排，和一般所谓的 Latin-1 这些编码（input encoding），他们的内容不一定相符，所以，fontenc package 常会和 inputenc package 互相配合使用，以确保在使用欧洲字符、符号时能正确取得到字。例如：

```
  ...
\usrpackage[T1]{fontenc}
\usepackage[latin1]{inputenc}
  ...
\inputencoding{ascii} % 也可以在文稿内文变换
  ...
\inputencoding{latin2}
  ...
\inputencoding{latin1}
  ...
```

当然，我们的文稿如果只是英美语系的文章，那这些都可以不必理会。

### 7.2.8 latexsym

这是 LaTeX 额外提供的符号。在新版的 LaTeX 2ε 并不会自动加载，要自行引入这个独立出来的 package。这主要是提供 lasy* 这些字型里头的符号。如果有使用 amsfonts 或 amssymb package 的话，这些 latexsym 符号应该是可以无需引入（有少数符号是 LaTeX 特有的）。至于各种符号的 package 有哪些内容，可以参考系统上的 symbols* 这些档案，他可能存在的形式是：

```
symbols.dvi
symbols-a4.ps[pdf]
symbols-letter.ps[pdf]
```

或者，也可以从 CTAN 下载最新的版本：

ftp://cam.ctan.org/tex-archive/info/symbols/comprehensive/symbols-a4.pdf

### 7.2.9 makeidx

这是在制作索引时要引入的 package，我们会在第 11.3 节，页 142 再来讨论。

### 7.2.10 newlfont

这是仿真旧版 LaTeX 的字型用法，让他使用新的取字机制的 package。也就是我们在第 4.7.2，页 34 所提到的用法。为免麻烦，我们尽量避免使用旧用法，而使用字型的标准指令。

### 7.2.11 oldlfont

这是仿真旧版 LaTeX 的字型用法的 package。

### 7.2.12 showidx

这个 package 会显示，\index 指令下在什么地方。这也会在第 11.3 节来讨论。

### 7.2.13 syntonly

syntonly package 提供 \syntaxonly 指令，他可以检查语法是否正确，并不会有 *.dvi 檔的输出。但这个 \syntaxonly 指令一定要放在 preamble 区。

### 7.2.14 tracefnt

这是追踪字型使用情形的 package。通常编译时所产生的信息已经很足够，但如果希望有更详细的字型使用信息的话，可以使用这个 package：

```
...
\usepackage[debugshow]{tracefnt}
...
```

请注意，这样会增加编译的时间，而且 *.log 档会很大。

## 7.3 LaTeX 官方文件中的工具组

这些宏套件，LaTeX 官方文件是归类在相关软件（relative software）中，可能会比上一节提到的标准宏套件来得实用些。但也同时可以看得出来 LaTeX 非内建的套件不少，加上其他外来的宏套件，那真的是套件满天飞，我们很希望在可能的情形下 LaTeX team可以考虑将一些必要的套件纳入内建，更加落实版面处理和文稿写作分开的理念。

### 7.3.1 AMS-LaTeX

LaTeX 本身就有排版数学式子的能力，但在比较专业使用时，可能会需要增强他的功能，AMS-LaTeX 是美国数学协会（American Mathematical Society, AMS）所发展的一个增强 LaTeX 数学式子编辑的宏组，是由 AMS-TeX
indexamstex@AMS-TeX 移植过来给 LaTeX 使用的，他主要分成两个部份：amscls 及 amsmath，前者提供符合 AMS 的文件规格的文稿类别，后者可加强原来 LaTeX 的数学模式。我们会在第 10 章，页 125 加以介绍。

### 7.3.2 babel

如果想排版英文以外的其他欧洲国家的语文，例如：德文、法文，那可以利用 babel 宏套件。

### 7.3.3 cyrillic

这是专为排版斯拉夫民族语文，例如：俄文，那可以使用这个套件。

### 7.3.4 graphics

这是处理图形要用到的宏套件。但目前一般都使用功能较完善的 graphicx 宏套件来取代 graphics 了，事实上，引用 graphicx 会自动的引用 graphics，而在指令使用的方便性上，graphicx 较佳，因此我们往后都是以 graphicx 为主来说明的。这两个套件属于 LaTeX的图形工具组，这个工具组包括了和颜色、图形相关的各种宏，我们会在第 9 章，页 93 来讨论。

### 7.3.5 psnfss

这是 Type1 字型的宏套件组，例如：times, charter, mathptmx 等等，他会去使用这些 Type1 字型。但通常这些字型有许多是商业字型，系统上不一定会有，如果没有的话，会去使用 free 的代替字型，或者就不嵌入这些字型了。如果没有这些商业字型，又想要嵌入替代的 Type1 字型的话，可以考虑使用 txfonts 或 pxfonts 宏套件及其所附字型。当然，如果专业使用的话，可能得考虑购买专业的商业字型来使用。

### 7.3.6 array

这是加强原来的 array, tabular 环境的宏套件，可增许多细部微调的功能。这在第 8.4 节，页 83，时会讨论到。

### 7.3.7 calc

这个套件可以让 LaTeX 接受一些简单的代数运算。主要用于微调一些原始预设的长度及计数器（counter）。

### 7.3.8 dcolumn

这是让表格中具有小数点的数字对齐的宏套件。我们会在第 8.9 节，页 88 中详细讨论。

### 7.3.9 delarray

这是加强 array 宏套件的功能，让矩阵或行列式的大分界符号可以使用较简单的指令。这个套件要配合 array 宏套件来使用。通常在 array 宏套件中，这些矩阵或行列式的大分界符号是由 \left 及 \right 来引导才会出来，但使用 delarray 宏则不必如此麻烦。这在第 10 章会讨论到。

### 7.3.10 hhline

这个宏套件会方便在画横线时也可以插入表格的纵线。

### 7.3.11 longtable

longtable 是用在跨页表格。通常在 LaTeX 中的 tabular 表格是当做一个 box 来处理，因此无法再分割，所以无法跨页来表现。这也会在第 8.10，页 90 谈到表格时提及。

### 7.3.12 tabularx

这是 tabular 表格环境的加强版，他可以方便的排版指定宽度的表格。同样的，这会在第 8.3 节，页 79 时提及。

### 7.3.13 afterpage

这个件主要在调整 LaTeX 的浮动环境（floating environment）时，置放浮动对象，例如：图、表的位置。

### 7.3.14 bm

bm 的意思，就是 bold math(symbol)，这会让数学式子以粗体的方式来显示。这个宏套件，提供一个 \bm{} 指令，只要把数学式子置于大括号中就会由粗体来显示。

### 7.3.15 enumerate

这是加强 enumerate 列举式条列环境的宏套件。他可以很方便的指定要使用什么方式来起头，原始的 enumerate 环境，预设第一层是阿拉伯数目字，虽然也可变更，但要重新定义，不是很方便。这里举个例子：

```
% example15.tex
\documentclass{article}
\usepackage{enumerate}
\begin{document}
\begin{enumerate}[Example-1.]
\item This is a item 1.
\item This is a item 2.
  \begin{enumerate}[(1)]
  \item This is a item (1).
  \item This is a item (2).
  \end{enumerate}
\item This is a item 3.
\item This is a item 4.
\end{enumerate}
\end{document}
```

可以指定会顺延显示的有：A, a, I, i, 1，如果这些是属于固定显示的部份，则要以大括号括起来，否则他会顺序计算显示。请试着和第 5.5.2 小节，页 53 的标准 enumerate 环境比较一下。编译后的结果如下：

http://edt1023.sayya.org/tex/latex123/example15.tex
http://edt1023.sayya.org/tex/latex123/example15.pdf

这里请注意一下一些同名的环境、宏套件，例如 array 宏套件及 array 环境，这里的 enumerate 宏套件也是一样。

### 7.3.16 fontsmpl

这是字型 sample 测试 package，他可以是互动的，也可以引用这个 package 后直接使用 \fontsample 这个指令来印出目前使用的字型 sample。

互动的话，要自行输入字族名称。sample 文件在 $TEXMF/tex/latex/tools 目录下，只要下：

```
latex fontsmpl.tex
```

就可以了，他会出现：

```
This is TeX, Version 3.14159 (Web2C 7.4.5)
(./fontsmpl.tex
LaTeX2e <2001/06/01>
Babel <v3.7h> and hyphenation patterns for american, french,
german, ngerman, nohyphenation, loaded.
(/usr/share/texmf/tex/latex/base/article.cls
Document Class: article 2001/04/21 v1.4e Standard LaTeX document class
(/usr/share/texmf/tex/latex/base/size10.clo))     (./fontsmpl.sty)
Please enter a family name (for example `cmr').

\family=
```

只要输入要测试的字型字族，例如 cmr，他就会产生 fontsmpl.dvi 这个档，然后就可以使用 dvips 或 dvipdfm[x] 把他转成 ps/pdf 格式的档案。他只会测试 OT1 及 T1 两种字型编码。

### 7.3.17 ftnright

LaTeX 在两栏式排版（two-column mode）时，他的脚注是置放在各自字段底部。ftnright 会将两栏式排版时，把所有的脚注都置放在右栏底部。这样可以将脚注集中，看起来不会那么凌乱。

### 7.3.18 indentfirst

通常，LaTeX 的章节开头的第一个段落是不缩排的，在第二个段落起才会缩排。如果习惯每个段落都有缩排，可以使用 indentfirst package。这个套件也是引入就可以了，无需任何指令。

### 7.3.19 layout

这是显示目前版面配置的 package。引入这个 package 后，只要在本文区下 \layout 指令，他就会画出目前的版面配置，也会将各种数据显示出来。我们在第 5.2.1 小节，页 44，里头所显示的版面图，就是这样画出来的。

### 7.3.20 multicol

在 LaTeX 宣告文稿类别的同时，我们可以选用 twocolumn 来选择两栏式的排版，再多则不行。在两栏式的排版时，我们可以使用 \onecolumn 及 \twocolumn 指令，在单栏及两栏间变换，但这有一个很严重的缺点，那就是字段变换也会迫使换新页，原来的页面将会显得空旷。

multicol 的目的，不仅突破两栏，可以做多栏式的排版（最多可至十栏的排版），也可以在变换字段编排时在同一页面变换，而不必换新页。他提供了 multicols 环境来做字段的变换。他的使用方法很简单，字段数目及变换完全由环境来控制：

```
...
\usepackage{multicol}
...
\begin{multicols}{栏数}
...
内容，依正常单栏方式书写即可
...
\end{multicols}
```

请注意，引入时 multicol 是没有 ‘s’ 的，而环境中的 multicols 是有 ‘s’ 的。multicol package 处理脚注的方式，和单栏排版相同，就是通通置于本页底部，不分左右字段。

### 7.3.21 rawfonts

这是仿真 LaTeX 2.09 旧版的低阶字型指令，例如 \texrm 代表 10pt 的罗马字族的字。在新版的 LaTeX 2ε 并没有定义这些指令。

### 7.3.22 somedefs

这是写 LaTeX 宏的一些范例定义，可以很容易的更改其中设定来写自己的 package。这不在这篇文章的讨论范围，因此就不多谈了。

### 7.3.23 showkeys

这个 package 会把 \label, \ref, \pageref 等交互参照的指令内容，或文献引用内容，在指令所在处印出来。

### 7.3.24 varioref

这是加强型的交互参照的方式，我们会在第 11 章来讨论。

### 7.3.25 verbatim

这是加强 LaTeX 原来的 verbatim 环境的同名套件。可以在里头使用批注，也可以利用 \verbatiminput{文件名} 指令来引入外来档案，当然，引入后会自动进入 verbatim 环境中。

### 7.3.26 xr

xr 是 eXternal References 的缩写，意思就是交互参照外部的档案。这会在第 11.2 节，页 140 来讨论。

### 7.3.27 xspace

这个 package 会在一个宏结束时聪明的插入适当的空白。我们在第 3.3.2，页 16，时曾谈到 LaTeX 指令的结束的问题，所以，我们得在指令后加 '\ ' 或者 '{}' 来结束一个指令。但如果宏定义时使用了 xspace 宏套件的一个指令 \xspace，那么他就会自动判断指令何时结束，而不必自行插入 '\ ' 或 '{}' 了。

### 7.3.28 theorem

这是 LaTeX 内建的 theorem 环境的加强型宏套件。我们会在第 10，页 125，再来讨论。

## 7.4 宏套件何处寻？

重复发明轮子要尽量避免，所以，如果需要些功能，而 LaTeX 似乎没有，那可以先找看看是不是别人已经有写好类似的功能的宏套件，首先要找的应该是 FAQ 文件：

http://www.tex.ac.uk/faq

在 news://tw.bbs.comp.tex 也有一篇 Chun-Chieh Huang 所维护的中文 FAQ 可以参考。另外 news://comp.text.tex 则是英文讨论组，可以多多利用。如果已经知道套件名或关键词，那可以到：

http://tug.ctan.org/CTANfind.html

去搜寻，搜寻到后可以抓整个目录的压缩文件。通常，安装 package，他里头会说明如何安装，万一没有的话，可以到 bbs/news 上发问，或者下载以下这个 sh script：

http://edt1023.sayya.org/tex/latex123/ltxins.sh

他的使用方法很简单，把他置于执行路径可及之处：

```
ltxins.sh your.dtx(or your.ins)
```

这样就行了，他会产生必要的 *.sty 或 *.tex 及 pdf 格式的说明文件。这里头使用的是 dvipdfm[x]，所以要有安装这两种工具才行。当然，他不会自动安装，你还是要手动把一些档案拷贝到 LaTeX 找得到的地方。这只是一个很简单的工具，因此不保证能成功编译出文件出来。

如果想知道某个套件在系统上是否已安装，安装在什么地方，可使用以下的小工具：

http://edt1023.sayya.org/tex/latex123/ltxpkg.sh

把他拷贝至路径所及之处，使用方法如下：

```
ltxpkg.sh  package-name[.sty]
```

这会列出所查询的套件是否已安装，及安装在什么目录下，及这个 package 是否有预先加载其他的 package(s)。当然，这两个小工具都是 bash script 写的，你的系统要有安装 bash，一般的 Unix-like 系统应该没有问题，Mac OS X 及 Windows/cygwin 环境就不敢保证了。这也是个很简单的小工具，如果是由 \input 指令所加载的其他 .tex 檔，可能就会侦测不出来了，许多特殊细节也是没有去考虑到，因此实际上要以 package 的原始 macro 内容为准。以下为一个使用实例：

```
edt1023:~$ ltxpkg.sh colortbl

The position of this package is  at:
/usr/share/texmf/tex/latex/carlisle/colortbl.sty

The preloaded package(s) of 'colortbl.sty' is(are):
array,color
```

所以，得知 colortbl package，系统上已经安装，而且他会在引入的同时也引入 array 及 color 这两个 packages，除非要变更这两个 package 引入时的选项参数，不然就可以不必引入这两个 packages 了。

# 第 8 章

# 表格的处理

这是属于一般人觉得比较困难，但却是很重要的部份，让我们多花点时间研究。LaTeX 的表格，因为是抽象逻辑的思考方式来制作表格，对一般用户而言，比较不容易转换成直观印象。当然，有些编辑器，例如 GNU Emacs，有方便画 LaTeX 表格的编辑器 script，但这些我们先不去理他，先从 LaTeX 本身表格的结构理解起，这样在使用其他的辅助工具时也会比较得心应手，甚至没有其他工具，只要把握住表格的大结构，制作表格就不会摸不着头绪了。

由于 LaTeX 内建的表格功能算有点阳春，因此这一章会介绍一些外来的宏套件，来弥补 LaTeX 表格功能的不足，这些宏，使用上算相当普遍，几乎所有的 TeX 的各种发行版都会附上，因此不必担心可移植性的问题。

## 8.1 表格的种类

表格的使用，在文章上常常是必备的要件，他有归纳及醒目的作用，当然，表格太多也是会喧宾夺主。通常，我们中文的使用习惯，表格就是大方框内有小方框，文字置于小方框内，甚至某些小方框内还有斜线在分隔。为了排版上的方便及视觉表现上的美观、清楚，在国际上大部份较正式的论文已不使用纵线、斜线，表格通常由横线来做区隔，甚至完全没有线条，使用空间区隔的方式。这种趋势几乎在二十几年前就已开始普遍，只是国内的文件似乎还是很喜欢有纵、斜线在表格之中，好像没有一些框线层层包住就不像表格。如非特殊的表现上的需求，我们应该朝简化表格本身的方向走，将重点置于表格的内容及表格的逻辑结构安排，漂漂亮亮的表格外观加上不当的内容配置，个人觉得这是个失败的表格制作。

另外，等粗的双线条，可能也是得尽量避免，通常粗细不等的外框双线条有装饰的作用， 因此，如果文件是较正式的论文，那就可能要避免，如果是海报、DM 或要让人们填写的

表格之类的，那又是另外一回事，这时封闭性的方框可能会有需要。这些规范只不过是一些惯例，并非一成不变的，得视文件的性质及使用场合来做变化，一个大原则是，如果是以文字叙述为主的文件，那么，表格本身如果比文字内容抢眼太多的话，或许就要考虑简化表格本身了。

我们这里就来比较，有纵线、无纵线、完全没有线框及含双线表格的各种形式的表格，大 家就自由心证，看哪一种表格看起来比较顺眼。由于 HTML 格式在表格的表现上可能会失真，因此这里制作成 PDF 格式供参考：

http://edt1023.sayya.org/tex/latex123/test-tables.tex
http://edt1023.sayya.org/tex/latex123/test-tables.pdf

## 8.2 tabbing 环境

这是 LaTeX 里头最基本的表格形式，除非自行另外定义、绘制，他并没有方便可用的线条指令来区隔，完全使用空间、位置的配置来显示表格内容，这时整个 tabbing 表格在 LaTeX 的地位并不是一个最小单位的 box，LaTeX 不会把整个表格当成一个单位来处理。所以，tabbing 表格是可以跨页的，他可以被分成两半来处理。因此，要和其他文字、图表并排排版时，得另外放进一个 box 中，让他自成一个 box 单位，例如 \parbox 或 minipage 环境里头。

在 tabbing 环境中，第一个列（row）是以 \= 来标示 Tab 宽度来区隔字段（column），这个宽度是由字段里头的字符串宽度所决定的。后续的每个字段是由 \> 这个符号来区隔，每列尾要自行加上 \\ 来换行，最后一行可以不必使用 \\ 换行。tabbing 的基本大结构是：

```
\begin{tabbing}
column1 \= column2 \= column3 \\
item1   \>  item2   \>  item3   \\
itemA   \>  itemB   \> itemC
\end{tabbing}
```

这里特意把他排列整齐（事实上，不排整齐 LaTeX 也会帮忙排好），这样才能看得出来他的表格结构。那如果想调整字段宽度时可以使用 template 的方式，例如：

```
\begin{tabbing}
xxxxxxxxxx\=xxxxxxxxxx\=xxxxxxxxxx \kill
column1 \> column2 \> column3 \\
item1   \> item2   \> item3   \\
itemA   \> itemB   \> itemC
```

```
\end{tabbing}
```

这里以 10 个 x 为字段的宽度，这里的 \kill 表示这一行是不印出来的，只是在表示各个字段的样本宽度，而且他会自动换行。当然，要使用其他的字符串也是可以，例如以表格中最长字符串来取代整个 x 字符串，这样就会让字段宽度刚好都可以容纳其他栏内内容。也可以使用 \hspace{6em} 或其他的长度指令，来指定字段的宽度。

对于字段内文字的控制，tabbing 较不完备，虽然 LaTeX 有提供 \' 让这个符号之前的文字靠左，及 \` 让这个符号之后的文字靠右，但实际运用，可能不是使用者想要的结果，因此 LaTeX 的表格，主要还是以 tabular 环境较为常用。但 tabbing 环境的好处是，他不见得一定要用于表格的排版，例如他也可以表现如条列环境般的另一种表现方式，而且他可以跨页排版。

## 8.3 tabular 环境

这大概是最常使用的表格形式，可以很方便的画线框。这种表格，LaTeX 是把整个表格当成一个单位来处理，就像字母一样，因此他在版面的安排上是和一般的字母一般的处理，所以，这种表格不经特殊处理，无法被分割成两个部份来跨页。

和 tabbing 环境的不同，除了可以有线条之外（tabular 环境，当然也是可以完全没有线条），分隔字段的符号是 &，而且，一定要指定栏内文字的置放位置，栏内文字超出指定的宽度时，会自动折行，还有许多其他更细节的调整。

### 8.3.1 tabular 表格的基本结构

```
\begin{tabular}[t]{lll}
\hline
column1 & column2 & column3 \\
\hline
item1   & item2   & item3 \\
itemA   & itemB   & itemC \\
\hline
\end{tabular}
```

其中 [t] 表示 top，也可以是 b 表示 bottom，或 c 代表 center，这要在前后有文字相并排的时候才会显现作用，因为 LaTeX 会把整个 tabular 表格当成一个字母单位，所以可以和其他文字、图表并排排版。这些参数的意思是和同行文字的对齐方式，top 是表格顶端

和前后文字对齐，bottom 则是表格底部和前后文字对齐，center 则是和表格中央对齐。

换行的方式和 tabbing 环境一样， 其中的 \hline 是画一条横线的意思， 连续两个 \hline\hline 会画双横线， 他本身会自动换行， 因此不必加上换行符号。其中 \begin{tabular}{lll} 的 lll 是在指定各字段内容在小方框内的置放位置，l 表示靠左（left），r 表示靠右（right），c 表示置中（center）。在 {lll} 中加上 bar（|）会画纵线，例如 {|l|l|l|} 这样就会变成传统的大方框、小方框的表格。而两个 bar 就会画双纵线。

tabular 环境内尚可使用另一个 tabular 环境来制作更复杂的表格，这在 tabbing 环境是不被允许的。

### 8.3.2 tabular 环境对字段的调整

1. p{宽度}
   这里的 p 指的是段落（paragraph）。通常用于一个小段落的文字，指定了宽度后里头的文字会自动折行，而且这个段落的顶端会和其他字段的顶端对齐。

2. @{文字、符号或指令}
   这可以作用在本栏的各个列，让他们都出现某个文字、符号或都在某个指令的作用 下。这个指令另外会同时将字段间距缩成 0，置于首尾的话，会有让横线和文字切齐的作用（预设不会切齐，横线两端会多出字段间距的部份）。

3. \multicolumn{字段数}{左右位置}{文字内容}
   跨栏排版，例如一小段文字跨两栏。左右位置可使用 lrc 之一。

4. \cline{a-b}
   画某部份字段的横线，其中的 a-b 指的就是要画线的字段数，例如 \cline{2-3} 就是画第二栏至第三栏的横线。

5. \arrayrulewidth=单位长度
   调整表格线条的粗细，默认值是 0.4pt。使用方法：\arrayrulewidth=1.5pt 即可，但要注意的是要在进入 tabular 环境之前设定好。

6. \tabcolsep=单位长度
   调整两字段的左右间距。请注意，这个值是实际两字段间距值的一半，预设是 6pt。使用方法和 \arrayrulewidth 一样。

7. \doublerulesep=单位长度
   调整画双线时，这两线间的间距，默认值是 2pt。使用方法和 \arrayrulewidth 一样。

8. \arraystretch
   调整表格的上下行距。请注意，这要由 \renewcommand 来重设，因为在 LaTeX 定义出的一个常数值，而这个 \arraystretch 只是这些常数值的倍数，我们要重新改变他才能改变默认倍数。例如：example16.tex 中的使用方法。

在 tabular 环境的参数中，可能是取代原来的参数，例如 p{}。也可能是置放在原参数的前后，如 @{}，这看一下实际例子就可以了解：

```
% example16.tex
\documentclass{article}
\usepackage{textcomp}                    % for \textcelsius
\renewcommand{\arraystretch}{1.2} % 将表格行间距加大为原来的 1.2 倍
\arrayrulewidth=1pt                      % 调整线条粗细为 1pt
\tabcolsep=12pt                          % 调整栏间距为 24pt
\begin{document}
\centering
\section*{SPECIFIC HEATS (20 \textcelsius\ AND 1 ATM)}
\begin{tabular}{@{\sf }lll@{}}           % 第一字段使用 sans serif 字族
\hline
 & \multicolumn{2}{c}{\bf Specific Heats} \\ % 跨二三栏排版，文字置中
\cline{2-3}                                      % 只画二三栏横线
 & $c$ (J/kg$\cdot$K) & $C$ (J/mol$\cdot$K) \\
\hline
Aluminum      & 900  & 24.3  \\
Copper        & 385  & 24.4  \\
Gold          & 130  & 25.6  \\
Steel/Iron    & 450  & 25.0  \\
Lead          & 130  & 26.8  \\
Mercury       & 140  & 28.0  \\
Water         & 4190 & 75.4  \\
Ice ($-$10 \textcelsius) & 2100 & 38 \\
\hline
\end{tabular}
\end{document}
```

textcomp 也是 LaTeX 的标准宏之一，他提供了许多符号，不必进入数学模式也是可以正常使用。但一般编译的话，可能会使用到 pk 点阵字，如果有安装 cm-super Type1 字型的话，可以使用以下的编译方式：

```
latex example16.tex
```

```
dvisp -Pcm-super example16.dvi
ps2pdf example16.ps
```

这样就会完全使用 Type1 字型。如果没有安装 cm-super Type1 字型，则可引用 txfonts 或 pxfonts 宏套件。

@{} 如果完全没有加入任何参数，那么他的作用只是在去掉左右两栏间距而已，大家可以把有关 @{} 的部份拿掉，试着再编译看看，仔细比较看有什么不同。有些专业排版的专家建议把表格前后加个 @{} 去除突出来的横线（实际上就是去除原有左右两边间距的部份）。编译好的例子在：

http://edt1023.sayya.org/tex/latex123/example16.tex
http://edt1023.sayya.org/tex/latex123/example16.pdf

如果 @{} 里头不是指令，而是文字或符号，那这个文字或符号会加在各栏文字内容的前或后。

p{} 指令的使用时机是某一个字段的文字比较多，需限定字段的宽度让他自动折行的情形，例如以下的例子：

```
% example17.tex
\documentclass{article}
\renewcommand{\arraystretch}{1.2} % 将表格行间距加大为原来的 1.2 倍
\begin{document}
\centering
\section*{Yi Syllables Area Character Blocks}
\begin{tabular}{@{}llp{6cm}@{}}
\hline
Start & End & Character Block Name \\
\hline
A000 & A48F & Yi Syllables.

                    Yi also known as Lolo, is a script resembling Chinese
                    in overall shaps that is used in the Yunnan province
                    China. \\
A490 & A4CF & Yi Radicals.

                    Basic units of the Yi syllables. \\
\hline
\end{tabular}
\end{document}
```

这样会把 p{} 指定的字段当成一整个段落来处理，空一个空白行，同样是表示新段落的开始。编译好的例子如下：

http://edt1023.sayya.org/tex/latex123/example17.tex
http://edt1023.sayya.org/tex/latex123/example17.pdf

## 8.4 array 宏套件

这个宏套件可以加强原有 tabular 环境的功能。使用上只要引入 array 宏套件即可，tabular 环境依原来的使用方法，只是多了些相关调整指令。

1. m{宽度}
   这和 p{} 一样的作用，只是置放的位置不一样，此时其他字段的内容会对齐这个段落的中央位置。

2. b{宽度}
   同 p{}，但其他字段的内容会对齐整个段落的底部。

3. >{指令}
   这可以置于 l,r,c,p,m,b 参数之前，是对于某个字段的内容下指令，这个指令会在此一字段内容之前作用。引用了 array package 后，可能会抑制某些 @{指令} 的作用，此时要改用 >{指令}，但这没有去除字段间距的功能，可在前头再加个 @{} 即可。

4. <{指令}
   和 >{指令} 相同，但会在此一字段内容之后才作用。

5. !{指令}
   这是取代 | 的作用，可以方便使用特殊符号来代替原来的纵线。

6. \extrarowheight
   这是在调整字段内容顶端的空间大小，但不会改变底部的空间大小。

## 8.5 tabularx 宏套件

tabularx 宏套件提供一个 tabularx 环境，这是加强型的 tabular 环境，附在 LaTeX 的工具组里头。主要作用是改善 \tabular* 指令，指定表格宽度的功能。

在原始 tabular 环境，加了个星号，可以指定表格的宽度。但由于 \tabular* 这个原始环境，他会去修改栏内空间，而不是实际整个表格方框的宽度，这使得某些文字会超出表格范围，因此，使用上可能 tabularx 会比较方便，他提供了 X 参数来取代原来的 lrc 参数，这个参数实际的作用是 p{} 的功能，因此会实际调整字段方框的宽度，而且里头的文字叙述超过字段宽度时会自动折行。这个套件会自动引入 array package[1]。这里使用这两种环境来排版，大家比较一下他的结果，就知道差异了：

```
% example18.tex
\documentclass{article}
\usepackage{tabularx}
\parindent=0pt
\renewcommand{\arraystretch}{1.2}
\begin{document}
\centering
\section*{\texttt{tabular*} environment}
\begin{tabular*}{8cm}{lll}
\hline
Start & End   & Character Block Name \\
\hline
3400  & 4DB5 & CJK Unified Ideographs Extension A \\
4E00  & 9FFF & CJK Unified Ideographs \\
\hline
\end{tabular*}

\section*{\textsf{tabularx} package}
\begin{tabularx}{8cm}{llX}      % 8cm 减去前两个字段宽度后，剩下的通通给
\hline                          %  第三字段使用，文字超出的部份会自动折行
Start & End   & Character Block Name \\
\hline
3400  & 4DB5 & CJK Unified Ideographs Extension A \\
4E00  & 9FFF & CJK Unified Ideographs \\
\hline
\end{tabularx}
\end{document}
```

tabularx package 并不是都没有缺点的，例如，使用 \verb 指令时会有一些不兼容，另外，在 tabularx 环境内还要有其他的 tabularx 环境时，这个在里头的 tabularx 环境要由大括号括住，不能像 tabular 环境一下的直接巢状使用。编译好的例子在：

http://edt1023.sayya.org/tex/latex123/example18.tex
http://edt1023.sayya.org/tex/latex123/example18.pdf

[1]套件查询，可使用第 7.4 节，页 75，所提到的 ltxpkg.sh 来查询是否有预先加载其他的 packages。

## 8.6 表格线条粗细的控制（booktabs）

由前面几节所述，可以看得出来 LaTeX 表格宏的功能稍嫌阳春了点，对于一些特殊状况可能会无法处理，对于表格外观要求较高的使用者也会感到不足，虽然也可以自行去定义宏，但这样一来不但可能有可移植性的问题，而且也不是每个人都有时间去学习
TeX/LaTeX 宏的写作。我们试图来看看有没有其他的解决方式，这里不得不会提到一些外来的宏套件，但这些套件的使用相当的普遍，几乎可以忽略他的可移植性的问题。

我们前面曾学过 \arraryrulewidth 指令，可以调整线条的粗细，但是这无法各别调整线条，每个在 tabular 表格环境内的线条会调整成一样的粗细。booktabs 宏套件可以很方便的达成这个目的。我们来看看这个提供了什么方便的指令：

| 指令 | 功能 |
|---|---|
| \toprule[线条粗细] | 画表格顶端的横线 |
| \midrule[线条粗细] | 画表格里头的横线 |
| \bottomrule[线条粗细] | 画表格底部的横线 |
| \cmidrule | 指令某个字段画横线，取代原来的 \cline |

使用方法和 tabular 环境差不多，连环境名称都一样，但可在指令后加个方括号来指定线条的粗细，不指定的话，toprule 及 bottomrule 都会比中间的其他线条粗一点。其中 cmidrule 另有更进一步的功能：

\cmidrule[线条粗细](左右是否去边){画线字段}

其中「画线字段」和 \cline 一样，指定字段数即可，例如 2–3。左右去边要表明左（l）或/及右（r），也可由大括号指定要去掉多少（预设 0.5em），如：(lr{0.7em}){2-3}。我们把 example16 拿来改一下，大家试着看看有什么不同，编译好的例子如下：

http://edt1023.sayya.org/tex/latex123/example19.tex
http://edt1023.sayya.org/tex/latex123/example19.pdf

由于屏幕分辨率的关系，如果分不出不同，请由打印机印出来比较，或将档案放大再来观察。这里最粗的是 toprule 及 bottomrule 再来是 midrule，最细的是 cmidrule。而且 booktabs 已经调整过原来 tabular 表格的上下间距，除非想得更大，不然的话，不需另外再去设定 arraystretch 的值了。

## 8.7 彩色表格（colortbl）

彩色表格已经是很普遍，但千万要小心喧宾夺主的情况，也别弄成了大花脸。因此，淡色系可能会比较合适。我们在第 3.4.3.1 小节及 example13 曾提到过 color package 的引用，但并没有详细说明这个套件的用法，而 colortbl 会使用到这些颜色的功能，因此这里稍微说明一下。

### 8.7.1 color 宏套件

这是附在 LaTeX 工具组 graphics package 中的一个宏，使用上非常简单，只要把 color 宏在文稿 preamble 区引上就可以使用颜色了。以下是常要用到的控制指令：

| 指令 | 作用 |
|---|---|
| \color{颜色} | 这会使用文章所有内容都使用这个颜色 |
| \definecolor | 定义颜色 |
| \textcolor{颜色}{文字内容} | 让文字内容使用某特定颜色 |
| \pagecolor{颜色} | 这是在设定背景颜色，本页及其后的页面会使用这个背景颜色 |
| \normalcolor{颜色} | 回复原来的颜色 |
| \colorbox{颜色}{文字内容} | 这是方框背景的颜色 |
| \fcolorbox{框色}{框内背景色}{文字内容} | 这是方框颜色和其内背景颜色不同 |

这里要注意的是，指令里头使用的颜色，必需是有定义过的颜色才能使用。color 宏只定义了一些基本颜色，**r**ed, **g**reen, **b**lue (RGB 模型原色), **c**yan, **m**agenta, **y**ellow, blac**k** (CMYK 模型原色), white，另外一个常用的 gray 灰阶模型（gray-scale），其他的颜色得自行定义。定义颜色的语法如下：

```
\definecolor{颜色名称}{颜色模型}{调色盘值}
```

第一个参数就是自定义的一个颜色名称，颜色模型可使用 rgb、cmyk 或 gray，各颜色深浅值在 0–1 之间，「调色盘值」就是各种原色的值。RGB 颜色的索引值，如果是Unix-like 系统，可找一下 rgb.txt 这个档案，里头就会有各种颜色的索引值，或者，参考 example24.pdf。这里以 bisque 这个颜色为例子，他的 rgb 三原色的深浅比例为 255, 228, 196，各除以 256 得 0.996, 0.891, 0.755，定义方法如下：[2]

```
\definecolor{bisque}{rgb}{.996,.891,.755}
```

[2]每一个原色在计算机上最小可由一个 byte(8-bits) 来储存，共有 256($2^8$) 种变化，所以，三原色可调出
$256^3$ 共 16,777,216 种颜色，但这不表示你的屏幕、打印机有办法显示那么多颜色。

```
\definecolor{mypink}{cmyk}{.1,.8,.4,.1}          % cmyk 模型的例子
```

这样以后就可以使用 bisque 及 mypink 这两种颜色了。gray 则支持灰阶，可以增减他的显现深浅，例如：

```
\definecolor{mygray}{gray}{.6}
或直接定义及使用，不事先定义好颜色名称：
\textcolor[gray]{.3}{文字内容}
\textcolor[rgb]{.2,.5,.7}{文字内容}
```

这样 mygray 会得到浅灰色的效果，他的颜色名称就是 mygray。直接定义及使用虽然也可以，但不建议这么做，因为如果有两个地方要使用同一种颜色时，又得重复定义一次。要 注意的是 gray 不能直接使用，要先定义他的灰色度，其他颜色也不能这样单纯靠一个值来定义他的深浅度。通常我们引用的时候，会加入以下的选项参数：

```
\usepackage[usenames,dvipsnames]{color}
\usepackage{colortbl}
```

这样就可以使用 dvipsnam.def 这个档里头所定义好的颜色， 例如 Salmon, Orchid, BlueViolet· · · · · 等等，请自行查阅这个档案内容，dvipsnames 这个参数也可以不用，只用 usenames 即可。这里引用时请注意顺序，colortbl 要在后面，原因是 colortbl 宏会自动引入 color 及 array 这两个宏，但里头并没有含任何选项参数，所以要抢先去宣告。

### 8.7.2 colortbl 的主要指令

| 指令 | 作用 |
|---|---|
| \columncolor | 让整个字段着色 |
| \rowcolor | 整整个横列着色 |
| \arrayrulecolor{颜色} | 指定线条的颜色 |
| \doublerulesepcolor{颜色} | 指定双并线内间隔的颜色 |

在这里，\columncolor 和 \rowcolor 的参数是一样的，他们的共同语法是：

```
\columncolor[颜色模型]{颜色}[左缘突出长度][右缘突出长度]
```

我们现在就来看个实例，这里头有四个小例子，包括：灰阶横条、部份字段着色、整个表格在着色背景及单一个表格内方框着色：

http://edt1023.sayya.org/tex/latex123/example20.tex
http://edt1023.sayya.org/tex/latex123/example20.pdf

## 8.8 表格的批注（threeparttable）

表格的批注比较麻烦，LaTeX 把 tabular 环境视为一个单位，对里头的文字做脚注的话，将会不翼而飞，有些宏套件有办法在表格内做脚注（例如 longtable package），但却是置于页面底部，和一般内文的脚注混在一起，多数使用者希望的是能把批注就置于表格底 部。解决的方法就是使用 threeparttable package，暂时将表格的某部份分割出来。

如果你的 threeparttable package 的表现和这里的例子有不一样的情形，请更新这个套件，这篇文章使用的是 2003/06/13 v3.0 的版本。

ftp://ctan.unsw.edu.au/tex-archive/macros/latex/contrib/misc.zip

下载后解开压缩文件，把 threeparttable.sty 拷贝至 $TEXMF/latex/misc 目录下，执行一下 texhash 一下即可。他的环境名称和套件名称一样，就是 threeparttable。把
tabular 及批注的指令和内容，通通包在 threeparttable 环境里头即可。

他是使用 \tnote{符号或文字} 先标出要批注的地方，在 tabular 环境结束后，再使用 tablenotes 环境来写批注内容，这两个部份都是整个被 threeparttable 环境包住的。底下这个例子来自这个套件的作者 Donald Arseneau，这里把他和 booktabs package 结合起来用：

http://edt1023.sayya.org/tex/latex123/example21.tex
http://edt1023.sayya.org/tex/latex123/example21.pdf

要注意的是，在 tablenotes 环境下，字体并没有缩小，可参考 example21 里头，使用 footnotesize 的字体大小。

## 8.9 小数点对齐（dcolumn）

这是 LaTeX 的标准宏，用于将表格内的小数点对齐。原来的 tabular 环境的作法是去增加一个字段，那个字段使用 @{.} 来专门排小数点，这样一来两栏的间距会消掉，看起来就像连在一起的数字了，但这样实在是有点 dirty，使用 dcolumn 宏的话，就可以很有规律的去对齐小数点或逗点。

dcolumn 的用法，主要是去取代 tabular 参数中的 lrc 这些参数。他使用的是一个大 D 指令，后接三个参数：

D{文稿输入符号}{排版后输出之符号}{小数字数}

例如以下的表格我们再怎么去排，小数点总是无法对齐，因为 tabular 环境是以整个字符串在处理的：

```
\begin{tabular}{lllll}
\toprule
        & headA & headB & headC & headD \\
\midrule
test1  &  7.879    & 921.661 & 1382.81 & 998.98  \\
test2  &  1.97     & 35.21   & 321.3   & 4791112.11  \\
test3  &  211.97 &  5.2      & 213.629 & 748261594.106 \\
\bottomrule
\end{tabular}
```

我们只要把 tabular 的后面参数改成：

```
  ...
\usepackage{dcolumn}
  ...
\begin{tabular}{lD{.}{.}{3}D{.}{.}{3}D{.}{.}{3}D{.}{.}{3}}
  ...
```

就可以让小数点对齐。这个 3 就是最长的小数字数，我们输入、输出都是英文句点（就是小数点），这样的表示法也可以另外宣告 \newcolumntype 的标准格式，以简化 tabular 参数的输入，即：

```
  ...
\usepackage{dcolumn}
  ...
\newcolumntype{z}[1]{D{.}{.}{#1}}          % 定义一个新的 z 指令
  ...
\begin{tabular}{lz{3}z{3}z{3}z{3}}
  ...
```

那个 #1 就是 z 这个新指令的参数（z 可以是任意的字母或符号），z{3} 其实就是代表 D{.}{.}{3}。中括号里头的 1 代表这个新的 z 后面只接一个参数，在这个例子里就是小数点个数 3。以下是编译好的例子：

http://edt1023.sayya.org/tex/latex123/example22.tex
http://edt1023.sayya.org/tex/latex123/example22.pdf

这里要特别注意的是，在 dcolumn 的效力范围里头，例如以上例子，受 z 指令影响的字段，他会自动进入数学模式，里头要表现数学式的话，前后不必再加 $，否则会跳出数学模式。例如 example22 里头，那些 headA 会变成斜体字，这是因为进入了数学模式，要让他正常的话，就要写成 $headA$ 这样来跳出数学模式。

## 8.10 大型表格（longtable）

这可能有两种情形。一种是很宽的表格，另一种是很长的表格。太宽的表格可考虑旋转一下，让他横放，至于长的表格可以使用 longtable 让他可以跨页连续。如果都不行，那只考虑夹页，图表另外制作，或者试着简化图表一途了。

### 8.10.1 太宽的表格

要把表格横放，方法很多，例如 graphics package 一起 release 的 lscape 宏套件，他会让内文旋转九十度，或者使用 graphics/graphicx package 本身的 \rotatebox 指令，将表格旋转九十度。另外，也可以使用 rotating package 来旋转。这些 package 基本上使用的都是 graphics/graphicx 宏上的旋转指令的功能，所以，不限定只能使用在图表而已。

这里就以 rotating package 为例来说明，他提供了 sidewaystable 及 sidewaysfigure 环境，前者会让表格旋转九十度，后者会让图形旋转九十度。这个套件会自动引用 graphicx 宏，不使用这些套件，使用 graphicx 的 rotatebox{90}{表格} 也是可以达到相同的功能，只不过限制会比较多。这里举一个 rotating 的例子，把表格置于 sidewaystable 环境内就行了：

http://edt1023.sayya.org/tex/latex123/example23.tex
http://edt1023.sayya.org/tex/latex123/example23.pdf

### 8.10.2 太长的表格

表格想跨页，可以使用 tabbing 表格，如果想使用 tabular 表格，又想可以跨页的话，可以使用 longtable package，这是 LaTeX 所附上的工具组。他提供了 longtable 环境来取代原来的 tabluar 环境。如果想要和 booktabs 合用的话，请更新 booktabs package 的版本，目前最新的版本是 2003/03/28 v1.618：

ftp://ctan.unsw.edu.au/tex-archive/macros/latex/contrib/booktabs.zip

我们把前面提到过的 rgb.txt 拿来排成表格参考，例子如下：

http://edt1023.sayya.org/tex/latex123/example24.tex
http://edt1023.sayya.org/tex/latex123/example24.pdf

## 8.11 浮动环境

tabular 表格，LaTeX 都会把他视为一个独立的 box，也就是会把他当成一个字母单位在处理，他不能被分割，常常因为图表稍大些 LaTeX 就会起新页去置放，但这样一来原本的页面就会显得空荡，整个版面看起来很不自然，这种情形下，他们的置放位置就很重要了，使用浮动环境的话，LaTeX 会继续文字的部份，而把图表置放在下一页的顶端。通常，在 LaTeX 的浮动环境下，图表通常会置放在一页的顶端或都是底部，正常是不置放在一页中间的位置，除非强迫指定，有放不下的情形时，就会让他占一整页。因此，LaTeX就得把前后位置经过整体的计算后再来决定图表应该置放在什么地方，这就是所谓的浮动 环境。[3]

### 8.11.1 基本的浮动环境

LaTeX 的浮动环境很简单，就是把表格置于 table 环境当中就可以了。在里头有 \{caption} 指令可以指定表格的标头，而且编译后会自动标上 'Table n:' 字样，后接 caption 的内容，那个 n 会自动编号。

一般国际上较正式的文件，caption 置放的位置惯例是「表上图下」，也就是说表格的标题是置于表格上方，图形则在下方。但 LaTeX 对 caption 的置放位置，是对于不管图表皆置于下方的配置，我们把 caption 置于上方时，caption 和表格的间距将会太小。如果不想手动去调整，可以找 topcapt package 试试看。手动修改的方法如下：

```
...
\begin{table}      % 进入浮动环境
\begin{table}[置放位置选项]
\setlength{\abovecaptionskip}{0pt}
\setlength{\belowcaptionskip}{10pt}
\caption{表格的标题}
\begin{tabular}{表格参数}
   表格内容
\end{tabular}
\end{table}
```

原始的定义，abovecaptionskip 和 belowcaptionskip 的值刚好相反。如果是两栏排版时，要加个星号，\begin{table*}...\end{table*}。

[3]这篇文章并不使用浮动环境，因为这篇文章的原先构想是一种讲义形式的文件，并非一份正式的文件格式。所以这篇文章中，图表并无 caption，他们的位置是当做一般文字内容排版的，为了叙述的连续性，图表都可能置于一页中央。

### 8.11.2 浮动环境选项参数

LaTeX 的浮动环境的配置，有时会不符和我们实际上的期望，这时可加入选项参数。

| 位置选项 | 作用 |
|---|---|
| h(here) | 置于下指令处位置 |
| t(top) | 置于一页的顶端 |
| b(bottom) | 置于本页底部，如空间不够会置于次页 |
| p(page) | 单独占一页，此页没有内文的部份 |
| \suppressfloats[位置] | 抑制浮动对象置放于本页的某处，他会出现在次页 |
| ! | 置于以上选页之前，会更强烈要求达到此选项的作用。但对 p 则无作用 |

如果都没有指定，那预设是 [tbp]。其中 \suppressfloats[位置] 只能指定 t 或 b，不能同时指定好几个，而且 ！的作用会优先，他会忽略 \suppressfloats 的指示。这些指示，LaTeX 经过整体计算后，如果觉得窒碍难行的话，仍然会「抗命行事」的，毕竟他要以大局为重。

# 第 9 章

# 图形的处理

TEX 系统发展的时代，对于图形处理还比较落后，当时，ps/eps/pdf、jpeg/png 这些图档都还不存在，因此这是 TEX 本身的一个盲点，虽然有 METAFONT/METAPOST 这些强大的造字、绘图的程序，但这些工具，一般人恐怕不容易驾驭，因此，可能需要寻求更方便的外来工具。

但 TEX 聪明的地方就是，他本身不能处理的，就预留个位置，让其他辅助的工具来处理，所以，这也是 TEX 20 几岁了，还是能和新的工具配合的原因。图形倒是还好，因为有方便的宏及绘图工具来处理，只要能画的出图来，一切都好说，而绘图的技巧就和 TEX/LATEX 本身的排版技巧不算是直接相关了。

## 9.1 图形的种类

我们使用的图档，基本上分成两大类，一是向量图，不会因缩放而失真，一是位图，会因为缩放而失真，但视使用场合，并不是所有的图档都适合制作成向量图的。不管是哪一种图形格式，都是数字化的结果，在计算机里头储存的都是数字，只不过解释过程不同而已。由于制作高质量的文件通常都使用向量图，因此，我们将会把重点放在向量图，尤其是 eps/pdf 格式。

### 9.1.1 位图形

这种图形应该是占最多数的，使用也最广泛。他是使用自然的方式来储存数字数据，把图形所占的页面想象成是许多很细小的方格子所组成，每一个小格子就代表了一个图素（pixel），这个图素可能代表者各种不同的颜色，只要单位小格子愈多（分辨率愈高），我们人类眼睛就会分辨不出其中的各小格式间的区隔，于是影像就可以平滑的显示出来了。

我们平常常见的图档格式，例如：jpeg, gif, bmp, ico, xpm, png, psd, tiff· · · · · 等等都是属于位图档。由于他是由固定大小的图素实际数字化储存的，所以如果他们放大或缩小，我们的眼睛就会分辨出不同，甚至放得更大些，还可以看得出「格子」出来（这会造成所谓的锯齿状，jaggies），原来的影像就因此失真了。

### 9.1.2 向量图形

向量图档储存的并不是实际各种图素的信息，而只是储存数学运算的基本描述，显像时再马上计算出结果来显示。例如，以一个饼图来说，他的图档可能只有储存圆心所在、圆的半径、颜色索引值等数据，要显示时，马上计算，然后显像（在屏幕或打印机上，最后显像当然仍是要转成位图的），但由于每放大、缩小时都会重新计算过，所以，就不会造成失真了。当然，这会更耗计算机资源，但以目前的计算机软、硬件进步的情形，这些消耗都可以控制在可被忍受的范围。

目前最常见到的向量图档，应该就是 eps/pdf, svg· · · · · 等图档，向量字体也是属于特殊格式的一种向量图。一般比较规律性结构的图，会比较适合使用向量图，自然界的实体影像可能就比较适合使用位图了，但科技不会把脚步停下来，将来的数字化会是怎么样的情形就只能用我们的想象力去填补了。

## 9.2 绘图工具

绘图的工具实在是太多了，这里不可能一一介绍，只能择要的简单说明。我们的重点是排版，因此要知道的是图形怎么安置于版面里头才会使整个版面协调一致，而不是在绘图教学。就请大家自行选个顺手的绘图软件去熟悉，这类工具，大概都是一理通百理通，画笔怎么用简单，要画出象样的图出来会比较困难。

### 9.2.1 原生绘图工具

这是安装 TEX/LATEX 系统，不管是哪一种的发行版都会附上的，但可能会有不易入门的感觉，一旦抓到到了诀窍，这是不假外求的工具。因此，在这篇文章里头，会对这些原生的绘图工具多做一些说明。这里所提到的原生绘图工具，另外一个好处就是可以使用 CJK 环境，意思当然就是说可以在图中插入中文及 LATEX 排版后的结果，这恐怕是许多用户希望的功能，但一般 GUI 式的绘图工具就常常无法完整支持了。

1. LaTeX 的 picture 环境
   在 LaTeX 里头有个标准内建的图形环境，那就是 picture 环境，但他只能绘制一些简单的图形，后来也有人写了 epic 及 eepic package 来增强 picture 环境的功能，这可说是 LaTeX 「原生」的绘图宏，虽然功能不是很强，而且是由指令来指挥绘图，不容易直观的转换过来，但好处是他是和 LaTeX 文稿结合在一起，使用的是 LaTeX 的指令，不是另外引用外来的现成图档，和 LaTeX 的结合当然会比较好。所以，我们在此会加以简单介绍。由于 eepic 引用了 PostScript 的指令，使用 pdflatex 时会无法编译，因此这章只会探讨 picture 环境及 epic package。我们会在第 9.3 节做进一步的说明。

2. PSTricks 宏
   另一个相当有名气的绘图宏组 PSTricks package，功能就相当强了，他仍然是使用了 PostScript 的指令，所以，在 pdflatex/dvipdfm[x] 常会无法编译，要 dvips 才有办法解读。但另有人写了 PDFTricks package，可以转换成 PDF device 认得的指令，所以，在此也会一并简单介绍他们的使用。我们将会在第 9.4 节做进一步的介绍。

3. METAPOST 绘图工具
   在 TeX 系统，则有 METAFONT 及 METAPOST 可供绘图，这可说是 TeX 系统的标准绘图语言，但和 TeX 的语法有很大的不同，是一种 object-oriented 式的 macro 语言，功能相当的强大，甚至可以制作字型。METAPOST 是 METAFONT 的改良版本，主要是让预设的输出是 eps 向量图档[1]，而且可以连续处理多个档案，也可以嵌入 TeX/LaTeX 的语法在里头。

   目前 Knuth 教授编写 TAOCP 使用的绘图工具就是 METAPOST，我们这里就不探讨 METAFONT，在语法上 METAPOST 和 METAFONT 是类同的。由于 METAPOST 所产生的 eps 图档，不管是 latex 或是 pdflatex 都可以顺利引用，所以将会在第 9.5 节独立做进一步的说明。

当然，以上的任一个工具，要详细说明的话，都可以写成一本独立的书，所以，在这里只 能简单的介绍，没有提到的部份，可以参考他们的使用手册。METAFONT 及 METAPOST 则可以另参考 Knuth 教授所写的 *The METAFONTbook*。

[1]严格来说，METAPOST 所产生的图档只是 eps 图档的一个子集，称为 mps，我们这里一并称他为 eps 图档，不做严格区分。

### 9.2.2 外来绘图工具

我们也可以从其他的外来绘图工具来绘制图形，然后再引入图档即可，这样一来就可以使用自己熟悉的绘图工具。绘图的话，当然是以向量图为优先考虑，因为他不会因为缩放而失真。但像一些照片类的图档，就不太适合使用向量图了。

以下列出的都是 free 的绘图工具，一般用途上，功能上不见得会输商业产品，但使用界面上就不一定比商业产品方便。以下只做简单的介绍，至于操作方法，请直接参考其中的使用手册的说明。种类很多，请别眼花撩乱了，就选个一、二种去熟悉他吧！画笔可能很多种，但画图「心法」只有一种。

1. gnuplot
   这是个有点古老，但却非常实用的 $XY$ 及 $XYZ$ 数据数据及数学函数的绘图工具，他有内建的绘图语言来绘图，可以使用交谈式的方式，或写成档案来做批次处理。如果有安装 kile 的话，他有 GUI 的图形界面可以用来方便绘图。他可以输出
   fig 图档，供 xfig 做进一步的修改、编辑，也可以输出 LaTeX 的 picture 环境文稿及 METAPOST 程序代码供引入。几乎主流的操作系统都有他的版本。他的网站及和
   LaTeX 结合的一些 sample，可以参考：
   http://www.gnuplot.info/
   http://www.fnal.gov/docs/products/gnuplot/tutorial/
   http://cips02.physik.uni-bonn.de/~baehren/tex/gnuplot.html

2. GNU plotutils
   这是和 gnuplot 类似的绘图工具及函式库（GNU libplot for C, libplotter for C++），主要用于绘制 2D 科学数据及数学函数向量图。他也支援 xfig 的 fig 图档。而且有现成的函式库，对于写绘图程序的人来说也很方便，像后面会谈到的 pstoedit 就有利用到这个函式库。他的网站在：
   http://www.gnu.org/software/plotutils/

3. xfig
   这也是很古老的 X Window System 下的绘图工具，他的文件格式是公开的，所以 gnuplot 也支持他。他相当于 MS Windows 下的 CorelDraw，预设的输出格式是 eps 图档，但也可以输出 LaTeX picture/epic 文稿。请参考他所附的文件来和 LaTeX/gnuplot 配合使用。他的网站在：
   http://www.xfig.org/
   如果你的平台无法编译 xfig，可以试试 Java 的 jfig：

http://tech-www.informatik.uni-hamburg.de/applets/javafig/

4. tgif

   这是和 xfig 类似的向量绘图工具，也可以输出 eps/pdf 图档供 LaTeX 来引入，gnuplot 也支持 tgif 图档，这个工具和 gif 图档是没有关系的。记得，在好几年前接触 tgif 时，有人把他拿来画卡通影像，效果还不错，当然，他的主要用途不是在画卡通。他的网站在：

   http://bourbon.cs.umd.edu/tgif

5. grace

   这是源自于古老的 ACE/gr 的 Motif 版本 xmgr[2]，由于改变版本为 GNU GPL 发行，所以名称改为 grace，意思是 “GRaphing, Advanced Computation and Exploration of data”，要说是 “Grace Revamps ACE/gr” 也是可以啦！这是类似 gnuplot 的 X Window System 下的绘图工具，但有漂亮的 GUI 操作界面，是 WYSIWYG 的 2D 数据数据绘图工具，他需要 Motif 或 LessTif 函式库，目前尚有少数 xmgr 原来的功能还没有完全移植过来。他的网站在：

   http://plasma-gate.weizmann.ac.il/Grace/

6. GNU Dia

   这很适合拿来画流程图、电路图的一个 X Window System 下的绘图工具，使用 Gtk+ 函式库，类似 Windows 下的 Visio。他可以输出 eps/svg 及一般常见的位图档，也可以输出 METAPOST、LaTeX PSTricks 及 xfig 的图档。

   http://www.lysator.liu.se/~alla/dia/dia.html

7. Ipe

   这是一般性的 X Window System 及 Windows 下的绘图软件，使用 Qt 函式库。他的特点是，主要输出 pdf 图档，并可嵌入 TeX/LaTeX 文字，也就是说图里头的文字可以是 TeX/LaTeX 排版出来的结果，也可以输出 eps 图档及 XML 檔。另外一个好处是，只要是 Ipe 制作出来的 pdf/eps 图档，他可以由 Ipe 重载后予以再次的编修，这对于 eps/pdf 档案的编修非常的方便，这类向量图文件的编修，一般是比较困难的，尤其是 pdf 格式。他的网站在：

   http://ipe.compgeom.org/

[2]原始的 ACE/gr 有两种版本，除了 Motif 的 xmgr 外，还有另一个 XView 的 xvgr 版本，但 XView Widget 和 Motif 之争，最后是 Motif 胜出。

8. skeneil(sketch)
   这在以前，称为 sketch，是一个一般性用途的绘图软件，可以输出多种向量图，包括 svg。他是使用 Python script 语言所写的。他的网站在：

   http://sketch.sourceforge.net/index.html

9. GIMP(GNU Image Manipulation Program)
   这是一般性的绘图软件，有 X Window System 及 Windows 的版本。但他主要处理的是位图，类似 Photoshop 软件。他的网站在：

   http://www.gimp.org/

10. MetaGraf
    这是 METAPOST 的图形界面软件，这样就可以使用 GUI 的方式来使用 METAPOST 的强大绘图功能了。他是 Java 语言所写的，所以要先安装 Java 相关工具组才行。而且，如果是像制作字型这类精确度要求很高的图档的话，可能就不是很合 适了。他的网站在：

    http://w3.mecanica.upm.es/metapost/metagraf.php

11. OpenOffice.org
    这是整合式的文字处理软件，里头也附有绘图工具 OpenOffice.org Draw(oodraw，可单独拿做绘图工具)。他的网站在：

    http://www.openoffice.org/

12. KOffice
    这是另一个整合式的文字处理软件，里头附有许多不同用途的绘图工具，例如：Kivio、Karbon、Krita 及 KChart 等， 他们都可以单独拿来绘图。他的网站在：

    http://www.koffice.org/

### 9.2.3 图形转换工具

由于 dvips 只能接受 eps 图文件及 PostScript 指令，而 pdflatex 则只能接受 pdf/jpeg/png 图档，dvipdfm[x] 除了可接受 pdf/jpeg/png 图档外，大部份情形下，也可以接受 eps 图檔，但硬生生插入的 PostScript 指令，则仍然是无法接受，毕竟他是要生成 pdf 檔的， 除非是由 METAPOST 所制作出来的图档，不然的话，就得使用转换工具将他们转成可被接受的格式，所幸，这方面的工具还算方便。上一节所提到的绘图工具，在某种程度上也是可以转换图档格式。

1. ImageMagick
   我们主要是使用这个软件中的一个工具：convert，他是个功能非常强大的图文件转换程序，但主是用在位图，纵使转成向量图也不是真正的向量图，只是把位图 wrap 进向量图档里头而已，放大时仍然会有锯齿状。他的网站在：

   http://www.imagemagick.org/

2. netpbm
   这是许多图档转换的小工具所组成的图档转换工具组，主要是用在位图的转换。 他的网站在：

   http://netpbm.sourceforge.net/

3. pstoedit
   这是真正各种向量图格式之间的转换工具。向量图的转换，主要是利于编修，例如 eps/pdf 图文件要直接编修的话，一般工具会有困难，如果我们转换成 fig 图档，然后再交给 xfig 去编修；或转成 METAPOST 原始码，使用编辑器进行编修，完成后再转回 eps/pdf，这样就很方便了。他的网站在：

   http://www.pstoedit.net/pstoedit/

4. ps2eps
   通常我们手上的图档不一定是 eps，而是一般的 ps，也就是说除了真正图的部份外，尚有一些空白在图的四周，这代表里头的边界（BoundingBox）没有定好，这样引入图档的时候，除非另做其他处理，不然的话，连原图周围的不必要空白也会引进文稿里去，通常我们只是想要有图的部份，这时可以将这个 ps 档经由 ps2eps 处理过，去除不必要的空白。这个程序的作者是 Roland Bless，使用 perl 所写一个很实用的小工具，在 Windows 系统的话，只要有安装 perl 及 GhostScript 也可以使用，作者也提供了一个 ps2eps.bat 批处理文件供使用。他的网站在：

   http://www.ipv6.tm.uka.de/~bless/ps2eps

   系统上安装 Ghostscript 的话，他也会附上一个 ps2epsi，这个工具也可以利用，但有时会算错 BoundingBox 就是了。

## 9.3 picture 环境

由于这是 LaTeX 内建的绘图环境，最能配合 LaTeX 原来的语法及版面配置，因此我们多花一点时间研究，使用时请另外引入 epic package，这样可使用多一些绘图功能。要注意的

是，LaTeX 的 picture 环境，和坐标息息相关，所以，绘图之前一定脑海里要有个坐标图来定位，而且要有相对长度的想象。

### 9.3.1 进入 picture 环境

进入 picture 环境的方式就像进入其他的环境一样，但他要指定图形对象的大小：

```
  ...
\usepackage{epic}
  ...
\begin{document}
  ...
\begin{picture}(宽,高)(参考原点)    % 进入图形模式
  这里下绘图指令，形成一个或多个图形对象，也可以写入一般文字
  让 latex 去排版。
\end{picture}
  ...
\end{document}
```

指定长、宽等度量时，可以加上单位，如果不加单位，事先也没有指定使用单位，那就是以 pt 为单位，(宽, 高) 是不能省略的，这在坐标图上，就是建立了左下角 (0, 0) 至右上角 (宽, 高) 的参考坐标系。(参考原点) 指的是左下角的原点平移至这个位置，往后就以这个点为原点，这个可以省略，省略的话，原点位置就是 (0, 0)。通常我们都会在进入 picture 环境前先加以指定好单位，例如：

```
  ...
\unitlenght=1mm              % 指定 picture 环境内的度量单位为 mm
\begin{picture}(50, 50) % 要进入 picture 环境前指定
  ...
```

这样在 picture 环境里头就无需使用单位，直接写数字就可以了，而单位就是 mm。

### 9.3.2 picture 环境的绘图指令

在绘制任何线条之前，我们通常会指定开始的位置，否则通通会从（参考）原点开始画起。原则上，picture 环境内，有方向性的图形对象的参考原点，例如直线、箭头直线，他的移动方式，在绘制了图形对象后，如果不再指定起始点，那么，$x$ 轴的位置会平移过去，但 $y$ 轴的位置则维持在原点的位置，这样说有点抽象，只有请大家试着去画看看才能体会了，但最好就是指定好各个图形对象的起始位置，才不容易搞错。

1. \put(启始坐标){图形对象}
   将图形对象置于启始坐标。这个图形对象也就是 picture 环境的绘图指令，也可以是一般的文字叙述，如果是文字，那么会依 LaTeX 的排版方式来显现，在这篇文章里头，有时也会称为「图文对象」。

2. \line(向量坐标){长度}
   以参考原点和向量坐标所构成的斜率画指定长度的直线。不过，LaTeX 的这个画直线的指令，有其限制：
   (a) 两坐标值必需互质。
   (b) 坐标值要在 −6 和 +6 之间的整数。
   (b) 坐标值必需为整数。
   所以，实际上只能画出 25 种斜率的直线，超过这个限制的直线，只能使用较复杂的 \qbezier 指令来画出来。

3. \vector(向量坐标){长度}
   和 \line 指令的作用及使用方法相同，但限制更严格，坐标值要在 −4 和 +4 之间，和 \line 不同的是向量方向的那一端会多了个箭头符号。直线和箭头直线，他们的参考起启点如果没有另行指定，那 $x$ 轴的值是会连续的，也就是说画了一条直线后，再接着画另一条直线，那他的 $x$ 轴起启点是由前一条直线的终点开始，但 $y$ 轴的值则没有这个特性。

4. \circle{半径}
   画圆指令。请注意，如没有使用 \put，则圆心是在原点。如果是使用 \circle*{半径} 则是实心的圆，常常用来画某个粗点。由于圆是以圆心为参考点，并没有方向性，所以，并不像直线一样，$x$ 轴的位置会平移，仍然会以原来的原点为圆心。但如果前面有直线，那么圆心的位置会受前一个直线影响，也就是说圆心的 $x$ 轴位置会是前一条直线的终点的位置，当然，$y$ 轴的位置不会受其影响，正圆及椭圆都是一样。

5. \oval{宽,高}[显示部份]
   画椭圆。「显示部份」指的是要画上半部（t），或是画下半部（b），或是画左下半部（bl），依此类推。不管是否完全画出，圆心仍然是位在完整画出时的圆心位置。

6. \qbezier[曲线总点数]{起点坐标}{控制点坐标}{终点坐标}
   画 quadratic Bézier 曲线。其中的「曲线总点数」，代表整条曲线的总点数，有指定

的话，曲线会变成虚线，不指定的话是实线，至于什么是控制点（control point），他可以控制曲线的弧度，可由数学运算计算出来。有兴趣的话，请参考：

http://www.ursoswald.ch/metapost/tutorial/BezierDoc/BezierDoc.pdf

7. \thicklines
   指定用较粗的线条，无需接任何参数。使用 \thinklines 可还原为默认值。

8. \thinklines
   指定用较细的线条，这是预设的线条组细大小，亦无需接任何参数。

9. \linethickness{粗细单位}
   指定线条的预设粗细。

10. \framebox(宽, 高)[框内位置]{图文对象}
    画实线框。所谓的「框内位置」可有 t, b, l, r, s，表示图文对象置放于方框中的位置。

11. \dashbox{虚线线段长度}(宽, 高)[框内位置]{图文对象}
    画虚线框。

### 9.3.3 简化坐标位置

选定一个坐标定点，我们可以使用 \put(坐标) 的方式来指定，但如果是有规律性重复出现的图形对象，这样一个一个指定，不仅很烦，而且也较耗内存，计算也会比较慢。这时可以使用 \multiput 指令，他的语法如下：

```
\multiput(起启坐标)(坐标递增值){重复次数}{图形对象}
```

这里举一个例子，画一个有格子的坐标系：

```
% example25.tex
\documentclass{article}
\usepackage{epic}
\parindent=0pt
\begin{document}
\unitlength=1mm
\begin{picture}(80, 60)
\multiput(5, 0)(5, 0){15}{\line(0, 1){60}} % 画 15 条直线，每隔 5mm 一条
\multiput(0, 5)(0, 5){11}{\line(1, 0){80}} % 画 11 条横线，每隔 5mm 一条
\thicklines
\put(0, 0){\vector(0, 1){60}} % 画 y 轴
```

```
\put(0，0){\vector(1，0){80}} % 画 x 轴
\put(0，0){\circle*{1}} % 画圆点，实心粗点
\put(-5，-5){$O(0，0)$} % 标上原点的坐标
\put(-5，60){$y$} % 标上 y 轴字样
\put(80，-5){$x$} % 标上 x 轴字样
\end{picture}
\end{document}
```

我们来看看这个 multiput 到底做了些什么事：

```
\multiput(5, 0)(5, 0){15}{\line(0，1){60}}
```

第一个坐标 (5, 0) 是启始坐标，接着的 (5, 0) 是递增值，也就是说 (5, 0), (10, 0), (15, 0). . . (75, 0) 会画后面所接的图形对象，也就是画长度为 60mm 的垂直线 15 次。由于我们在 $x$ 轴及 $y$ 轴是另外画带有箭头的直线，因此，纵横轴的部份可以少画一条直线。$x$ 轴为 0 的 \line 就是在画垂直线，$y$ 轴为 0 的则是在画水平线。试想想看，这些线条如果要由 \put 指令一个一个画上去的话，会有多烦！

画这些除了练习外，主要是给初接触 picture 环境的朋友一个建议，那就是把方格子画上去，有利于绘图时找位置，等真正要画的图画好了，再把方格子拿掉。编译好的例子如下：

http://edt1023.sayya.org/tex/latex123/example25.tex
http://edt1023.sayya.org/tex/latex123/example25.pdf

另外一个简化坐标的方式，就是使用 \shortstack 指令，他的语法如下：

```
\shortstack[位置]{图文对象}
```

这会像迭罗汉一样的把「图文对象」迭在一个字段内，和迭罗汉不同的是，后进的迭在最下面，先进的会被往上堆高，底部的基线是固定的，高度则是往上增高，各图文对象由 换行符号来换行，也就是说可以由换行符号来决定他们之间的间隔。当然，这要自行注意 他的高度，否则会和其上的其他内容重迭。「位置」可为 **l**，**r**，c 之一，是指居中，或靠这个字段的左右边的意思。

\shortstack 的一个特殊的运用，就是在坐标图上标注纵轴的文字，但这通常是用在中文，因为，一般的惯例，纵轴的说明文，英文的话是沿纵轴由下往上写，中文的话是由上往下写。我们把 example25 标上中文，实际要加入的内容为：

```
  ...
\put(-7， 20){\shortstack{这\\[-2pt]里\\[-2pt]是\\[-2pt]纵\\[-2pt]轴}}
\put(30，-6){这里是横轴}
  ...
```

从这里，我们也可以发现，把格子画出来，对于绘图或加入说明文字时的定位非常方便。请注意，这个例子使用了 CJK 环境，要使用 bg5latex 来编译。编译好的例子如下：

http://edt1023.sayya.org/tex/latex123/example26.tex
http://edt1023.sayya.org/tex/latex123/example26.pdf

当然，这样一来，字间距也得手动去调整了，理想的话是应该将中文字旋转才比较能符合原来的字间距，意即，横排时的字间距和行间距，在直排的时候，两者要互换过来，但是这样一来，会造成中文是沿着纵轴往上写的情形，这就不符合惯例了，但这刚好常常用在中英文混合的说明文场合，中英文混合时，是按英文的惯例，沿着纵轴由下往上写，我们 将 example26 修改一下：

```
...
\put(-7，20){\rotatebox{90}{这里是 $y$ $axis$}}
\put(30，-6){这里是 $x$ $axis$}
...
```

请注意，数学式子中的额外空白通常会被忽略。编译好的例子如下：

http://edt1023.sayya.org/tex/latex123/example27.tex
http://edt1023.sayya.org/tex/latex123/example27.pdf

如何恰当的使用，就请大家视需要去调整、运用了。我们甚至可以更进一步的把各别的中文字去分别旋转后再排上去，而且，通常图表的说明文字会比正文小一号，就请大家动手 练习一下啰！这个 \rotatebox 指令，我们还没有学到，会在第 9.6.4 小节，页 122 里说明。

请将以上所谈到的指令，一一去试着画几次，大概就能体会出 picture 环境如何画图了。

### 9.3.4 epic 宏延伸的指令与环境

以下是 epic 所扩充的指令，和 picture 环境配合的话，会使绘图更得心应手。

1. \multiputlist(起启坐标)(坐标递增值)[tbrl]{对象1，对象2·····}
   \multiput 是针对同一个图形对象按规律性来置放，这个指令则是针对，不同的图物形对象按规律性来置放。他是把所有的对象置放在一个 box 中去排列，因此会有 tbrl 的置放位置的选项参数。

2. \matrixput(起启坐标)(递增值1){次数1}(递增值2){次数2}{图形对象}
   这是 \multiput 的两种置放规律版本，可指定两种规律来置放同一图形对象。

3. \grid(宽，高)(横间隔,直间隔)[标注纵横轴坐标起启值]
   画方格的指令。我们在前面有使用 \multiput 来画方格的例子，使用这个 \grid 将更为方便、简洁。选项的部份，就是在方格外围标注他的坐标值，通常使用 (0, 0) 即可。

4. \picsquare{图形对象}
   这只是让圆形、椭圆形，会自动产生一个圆心黑点。

5. \dottedline[点的形式]{点间距}(坐标1)(坐标2)· · · · · ·(坐标n)
   画各种不同形式的虚线。会在各坐标间连成虚线。点的形式可以使用其他的符号代　替。

6. \dashline[延展值]{各点线长}[dash 点间距](坐标1)(坐标2)· · · · · ·(坐标n)
   画各种不同形式的 dash 线。会在坐标间连成 dash 线。延展值从 −100 至无限大，可调整各 dash 的间距。「dash 点间距」只是在调整构成 dash 本身的点的间距，非各 dash 间距。

7. \drawline[延展值](坐标1)(坐标2)· · · · · ·(坐标n)
   这是加强型的画线指令，可以将各坐标间连成一线。延展值与 \dashline 同，负数值会造成类似 dash 线的效果。

8. \putfile{文件名}{绘图指令}
   这是引用外来数据数据来绘图。也就是可以将 $XY$ 坐标值数据数据另存放在外部档案来引用进来，其余的功能和 \put 一样。

更详细的 epic 说明及其实例，可参考 Sunil Podar 所写的文件 *Enhancements to the Picture Environment of LaTeX*：

http://www.ntg.nl/doc/podar/picman.pdf

## 9.4 PSTricks 及 PDFTricks 宏套件

PSTricks 插入了 PostScript 的绘图指令给 dvips 去处理， 所以绘图功能当然比 picture 环境强大， 但有一个缺点是 pdflatex/dvipdfm[x] 编译时会出问题， 得由 latex/dvips/ps2pdf 的方式来制作 pdf 档，或者另外绘制后，利用 graphicx 宏来引入独立的图档。

所以，如果是要制作 pdf 档案的话，那么处理上会比较不方便，而且，如果又是中文的话，那么所制作出来的中文 pdf 檔，以目前的 PDFLaTeX 的话，对于中文将会没有 copy&paste&search 的功能。PDFTricks 虽然可以让 pdflatex 使用 PSTricks，但他是利用了 TeX 系统引用操作系统 shell 的功能，这在非 Unix-like 的操作系统可能会出问题，而且他和 latex/dvipdfm[x] 的兼容性并不是很好。如果制作的是英文文件，那倒是没有关系，反而可以使用 PSTricks 的强大绘图功能。

### 9.4.1 PSTricks 的组成宏

PSTricks 除了主要的 pstricks 宏外，另外还有其他的宏，专门处理各种不同的特殊绘图。例如：pst-node 及 ps-tree 可用来画树形图，pst-grad 可以表现渐层颜色，pst-poly 可以绘制多种的多边形，pst-gr3d 可以画 3D 格子图。通常，一般用途的话，就是引用 pstricks，而他的绘图环境是包在 pspicture 环境里头的，他的默认单位是 1cm，和 picture 环境的 1pt 不同：

```
...
\usepackage{pstricks}
...
\begin{document}
...
\begin{pspicture}(左下角坐标)(右上角坐标)
  这里绘图
\end{pspicture}
...
```

他的其他专用宏并不会自动引用 pstricks，所以要自行引入后，再引用专用的宏，另外，如果要使用颜色的话，由于 pstricks 对颜色的定义和 LaTeX 的宏会有不兼容的情形，因此，我们要引用 David Carlisle 另外写的 pstcol 这个宏来修正他。pstcol 会自动引用 color 及 pstricks 这两个宏，因此，除非要加入选项参数，引用了 pstcol 就不必引用 color 及 pstricks 了。pstcol 也可以引用和 color 一样的参数，例如：

```
...
\usepackage[usenames,dvipsnames]{pstcol}
...
```

由于篇辐的关系，这里不准备详细介绍 PSTricks 的绘图指令，但 TUGIndia(Indian TeX Users Group) 已经完成了相当不错的 PSTricks 入门教材，请大家不要错过：

http://sarovar.org/projects/pstricks/
http://www.tug.org.in/tutorials.html

### 9.4.2 PDFTricks 的使用

这个套件一般系统是没有安装的，请至 CTAN 下载，下载后解开，如果是在 Unix-like 系统，那执行 make 就可以了，否则请将 pdftricks.sty 拷贝至：

```
$TEXMF/usr/share/texmf/tex/latex/pstricks      或
$TEXMF/usr/share/texmf/tex/latex/pdftricks      请自行建立目录
执行 texhash 或 mktexlsr
```

另外，将 pst2pdf 这个可执行文件，拷贝至执行路径可及之处就行了。另请注意他的文件格式，如果是在 Unix-like 系统，则要把他的文件格式改为 Un*x 系统的规格。

PDFTricks 主要是把 PSTricks 宏的引用，由 psinputs 环境包起来，而 pspicture 环境，则另外由 pdfdisplay 环境包起来，这样就可以由 pdflatex 来编译了。例如：

```
\documentclass{article}
\usepackage{pdftricks}
\begin{psinputs}
 \usepackage[usenames,dvipsnames]{pstcol}
 所有 pstricks 的宏引用，都要被 psinputs 环境包起来
\end{psinputs}
  ...
\begin{document}
  ...
\begin{pdfdisplay}
\begin{pspicture}
  这里依正常 pstricks 的指令绘图
\end{pspicture}
\end{pdfdisplay}
  ...
\end{document}
```

这样就可以由 pdflatex 直接编译了。要非常注意的是，执行 pdflatex 时，后面要加上 -shell-escape 参数，这个功能，一般的 TEX 系统是关闭的，要打开来让他可以由 pdflatex 执行当中，停下来等待外部指令的执行完毕后再继续原来未完成的工作。不过，这个方式不保证能在 Windows 系统中正确执行。

这里举一个简单的例子，同时说明如何使用 PSTricks、PDFTricks 及如何引入外部 gnuplot 制作的 picture 环境文稿的例子：

http://edt1023.sayya.org/tex/latex123/test-pstricks.pdf
http://edt1023.sayya.org/tex/latex123/test-pdftricks.pdf
http://edt1023.sayya.org/tex/latex123/test-pstricks.tex
http://edt1023.sayya.org/tex/latex123/test-pdftricks.tex
http://edt1023.sayya.org/tex/latex123/gnuplot-label.dem

最后一个 tar ball 是原始文稿。latex/dvips/ps2pdf 的编译的方式如下：

```
gnuplot gnuplot-label.dem    => 产生 gp-test.tex picture 环境文稿
latex test-pstricks.tex
dvips -o test-pstricks.ps test-pstricks.dvi
ps2pdf test-pstricks.ps
```

pdflatex 的编译方式如下，请不要忘了 -shell-escape 参数：

```
[gnuplot gnuplot-label.dem]     如果前面已制作好，这里就不必执行了
pdflatex -shell-escape test-pdftricks.tex
```

关于这些在 LaTeX 绘图的例子另请参考 Urs Oswald 所举的例子及其说明：

http://www.ursoswald.ch/LaTeXGraphics/overview/overview.html
http://www.ursoswald.ch/LaTeXGraphics/overview/latexgraphics.pdf

从这些例子可以发现，picture 环境及 PSTricks 宏的确是可以拿来绘制精美的图形的。

## 9.5 METAPOST 使用简介

METAPOST 除了可以直接绘图，还可以做一些数学运算后把结果图形化，所以不必如 picture 环境般的一笔一笔的画上去，只要能整理出规律出来，就可以使用数学函数的方式来绘图。在使用上和 picture 环境一样，要有坐标系的观念。

### 9.5.1 如何编译 METAPOST 图档文稿？

通常 METAPOST 文稿我们以 .mp 为他的延伸档名，以便和其他档案辨别，这个文稿使用任何一种编辑器编辑即可。在英文环境，编译 METAPOST 文稿的方法可有两种，中文环境的话，我们会在第 9.5.6 小节讨论：

1. 使用 mpost
   这是最正统的方式。底下以 some.mp 为例，这里假设里头只有一个图档，他的编号是 1：

   ```
   mpost some.mp      % 产生 some.1 及 some.log
   epstopdf some.1    % 产生 some.pdf 图檔
   ```

   some.1 就可以直接引入文稿中了。

2. 使用 mptopdf

   这会产生 pdf 格式的结果，以便给 pdflatex 来引用。其实这只是间接使用了 mpost，主要的编译程序仍然是 mpost，只不过另外会自动去引用外部程序处理转换成 pdf 格式的步骤而已。另外，他对字型的使用有做特别处理，因此如果图文件里头有使用文字的话，可能 mptopdf 会处理的比较好。

   ```
   mptopdf some.mp  % 产 生 some.1 及 some-1.pdf
   ```

   这样也可以产生 some.1，及多出一个 some-1.pdf。但 mptopdf 所产生的 some.1 是真正的 eps 檔，可以直接由 gv/gsview 来阅览，传统 mpost 所产生的 some.1 则是使用 METAPOST 的字型表示法，所以 gv/gview 会无法正常阅览。不过，引进 LaTeX 文稿则没有什么差异。

3. 编译结果的验证方法

   由于 mpost 编译后的 eps 图档，gv/gsview 并不一定能阅览，所以，我们要验证他的编译结果，除了引入文稿中外，还有一个标准的验证方法：

   ```
   tex mproof some.1 ... some.n    % 产生 mproof.dvi
   dvips mproof.dvi                % 产生 mproof.ps
   ```

   可以接受多个图档，看原来的文稿中有几个图档，后面就接几个图档，他会集中所有图文件显示在 mproof.dvi 当中，再利用 dvips/dvipdfm[x] 就可以产生 ps/pdf 檔来阅览，他会加入各个图档的档名及编号。

### 9.5.2 METAPOST 文稿的基本结构

在谈到 METAPOST 文稿的结构前，我们先提一下 METAPOST 使用的度量单位，如果没有标明的话，他预设是使用 big point(bp)，也就是 PostScript 规格中所使用的单位，这和 TeX 本身是使用 printer point(pt) 不一样。这里我们所谈的，主要是用于绘图，至于他的宏功能，这里就不讨论了。他的文稿结构如下：

```
% 和 TeX 一样，批注是使用百分号
beginfig(n);
  这里写 METAPOST 的绘图指令叙述，每个指令叙述以 ; 结尾
endfig;
bye;    % 这一行可以不必 ;。这是通知 mpost 程序跳出，结束绘图，end 亦可。
```

其中的 beginfig(n) 的 n 可以是任何一个数目字，代表经 mpost 编译过后所输出的 eps 图档的延伸档名，如果一次要处理多数图档，只要使用多个 beginfig...endfig 就可以了，但其内的数字则不可相同，以免图档被覆盖。METAPOST 的指令和 TeX/LaTeX 不同

的是前面没有 \ 来起头，而且每个叙述最后一定要加个 ; 来结束，否则编译会出错，例外是 beginfig、endfig 及 bye/end 这些指示可以不必使用 ; 来结束。

### 9.5.3 METAPOST 的九种基本数据型态

这些数据型态可以储存固定的绘图指令组或者是数学表达式，这样的好处是，可以把一组复杂但又常重复的绘图指令集或数学运算，宣告成几个变量来代表，往后要用到时，就使用变量即可，可以让 METAPOST 的程序代码写起来更简洁。

1. numeric
   数值，他的大小要在 −4096 与 4096 之间，中间的计算值也不能超过这个限度的 8 倍。一般宣告的例子如下：

```
numeric a, b, c, d; % 宣告 a,b,c,d 四个变量为数值型态
a := 9;
b := 3*a**2;
c := a++b;
d := c+-+b;
show a, b, c, d;        % 把计算结果显示出来
bye
```

   其中 3*a**2 代表 $(3a)^2$，a++b 代表 $\sqrt{a^2+b^2}$，c+-+b 代表 $\sqrt{a^2-b^2}$。METAPOST 不支持科学计数表示法。这可以存成一个档案，经由 mpost 编译后会显示计算结果，也可以由以下的方式来直接输入计算：

```
edt1023:~$ mpost
This is MetaPost, Version 0.641 (Web2C 7.4.5)
**\relax
* => 变成一个星号时，就可以输入各行叙述了
  ...
```

   这里要注意的是，numeric 型态不一定要事先宣告，可以直接使用，但宣告的用意是在消除前面的指定，也就是说，一经宣告就会消除以前所指定的值。所以，直接使 用的变量，METAPOST 会把他当做是 numeric 型态。

2. pair
   这是指平面坐标值，各 pair 间可做四则运算。例如，我们指定[3]：

[3]METAPOST 的指定（assignment）运算符是 :=，一般的等号才是等于，但第一次指定时两个都可以代表指定运算，其后更改指定的话就一定要使用 :=。

```
beginfig(1);
u := 1mm;              % 指定单位
pair a, b, c, d;       % 宣告四个 pair 形态的变量
a := (0, 0); b := (30u, 0); c := (30u, 30u); d := (0, 30u);
draw a--b--c--d--a;
endfig;
bye
```

这会绘制 30mm 长宽的正方形。也可以使用矩阵的方式来宣告，例如：

```
pair a[];     % 这等于宣告了 a0, a1, a2, a3...
pair a[][];   % 这样宣告也可以，代表 a00, a01...,a12, a13...
```

3. path
   有方向性的 pair 都可以是 path。例如：直线、曲线。

4. color
   颜色，由 rgb 值所组成，例如：(0, 0, 0) 是黑色，(1, 1, 1) 是白色。预先定义好的颜色有:

| 颜色 | rgb 值 |
|---|---|
| black | (0, 0, 0) |
| white | (1, 1, 1) |
| red | (1, 0, 0) |
| green | (0, 1, 0) |
| blue | (0, 0, 1) |

   在 METAPOST 使用颜色比在 LaTeX 里头方便多了，各种颜色的表示，除了原来的 rgb 值的表示法外，也可以在各种预先定义好了的颜色加上其深浅度，例如：

```
withcolor .6red;
withcolor .5white;
color A;               % 宣告 A 为 color 型态的变量
A := (.3,.8,.2);       % 指定 A 的颜色
withcolor A;           % 使用颜色 A
```

5. string
   字符串，在 METAPOST 中，字符串可能两种情形，一种是单纯的字符串，以 " 框起来即可，另外一种是由 btex 及 etex 包住的字符串，这些字符串，METAPOST 并不去处理，而是交给 tex 去处理。

   通常，我们常常需要把计算出来的数值结果标示在图上，这时要把他转换成字符串的型态才能标示上去，我们可以使用 decimal() 函数来转换。因为我们不能把变量值放在标注指令内，否则他将无法进行运算。例如：

```
u := 1cm;
for i=0 upto  10:
  label.top(decimal(i/10), ((i+1/2)*u,1u));
endfor;
```

这时，变量尚未计算出来，因此不能当做 label.top() 的参数，需要在结果计算出来后，马上进行转换成为字符串型态。

6. boolean
   布尔值。

7. picture
   任何可以由 METAPOST 绘出来的图形，皆可宣告成 picture 变量。

8. pen
   画笔。

9. transform
   转换。

### 9.5.4 METAPOST 常用的指令及函数

METAPOST 的指令，基本上可分成两大类，绘图指令（picture command）及卷标指令（label command）。绘图指令用于绘图，例如 draw、fill 等等；卷标指令则用于标注文字，例如 label、dotlabel 等等。绘图指令后面可以再接附加指令（addto command）及修整指令（clip command），例如 withcolor、withpen、clip...to 等等，附加指令及修整指令都是可以省略的；卷标指令接的则是要标注的文字及其位置，这些则不能省略。整体的语法结构算是很紧密的结合在一起，所以这里就不完整列出来了，但实际使用则还算口语化，从实际例子去熟悉用法，可能会比较容易进入状况。

1. 画线（draw--）
   由 draw 指令来开始画图，各坐标值（pair）由两个 hyphen（就是键盘上的减号）--来连接各个 pair，这就会构成直线，最后连接至 cycle 的话，就会形成封闭区域的的各种形状的「框」。有附加指令的情形，例如：

```
pair a, b,  c;
a := (0, 0); b := (1cm, 0); c := (0, 1cm);
draw a--b--c--cycle withpen pencircle scaled 1bp
      withcolor  .8white  dashed withdots;
```

棕色的部份就是附加指令，他的意思是设定线条的粗细为 1bp（默认值是 0.5bp），使用的颜色是灰色，并且是虚线，附加指令可以省略全部或一部份的叙述。如果附 加指令的部份会使用二次以上的话，也可以将他固定下来，这样就不必每 draw 时都要去指定，例如：

```
pair a, b, c;
a := (0, 0); b := (1cm, 0); c := (0, 1cm);
pickup pencircle scaled 1bp;                          % 固定线条粗细
draw a--b--c--cycle withcolor .8white dashed withdots;
```

2. 设定 draw 默认值（drawoptions()）

   如果我想更进一步的设定 draw 的默认值，我们可以使用 drawoptions() 函式，例如前面的例子，可以设成：

```
pair a, b, c;
a := (0, 0); b := (1cm, 0); c := (0, 1cm);
drawoptions(withpen pencircle scaled 1pt dashed withdots withcolor .8white);
draw a--b--c--cycle;
```

   要注意的是，如果有标注文字，这会连图上的标注文字也使用所指定的颜色。

3. 画弧线（draw..）

   和画直线完全相同，只是把两个 hyphen 换成两个句点 .. 来连接各个 pair。

4. 画箭号（drawarrow, drawdblarrrow）

   和 draw 指令的用法一样，只是改成 drawarrow（单箭号）及 drawdblarrow（双箭号）。

5. 画虚线

   在画线指令后附加上 dashed evenly 或 dashed withdots 即可。这是属于附加指令的叙述。我们也可以由 dashpattern() 来自行定义虚线的形式，例如：

```
pair a, b, c;
picture dptn;
a := (0, 0); b := (1cm, 0); c := (0, 1cm);
pickup pencircle scaled 1bp;                          % 固定线条粗细
dptn := dashpattern(on 6bp off 2bp on 2bp off 2bp); % 指 定 dash 的 形 式draw
a--b--c--cycle withcolor .8white dashed dptn;
```

   dashpattern 的部份，on 表示会显示出来的线段，off 表示不显示出来的空白。预设指定的形式是：

```
evenly    := dashpattern(on 3 off 3);
withdots := dashpattern(off 2.5 on 0 off 2.5);
```

这里的 evenly 及 withdots 就是属于 picture 型态的变量。

6. 填色

使用 fill 指令来代替前面的画线指令，后面要加个 withcolor 的指令来指定颜色，否则会填入黑色。要注意的是，fill 必须是封闭区域才有意义，因此，连接 pair 时最后一定要使用 cycle 把区域封闭起来。

要注意的是，他仅仅着色，不画线框，要画线框要另外由 draw 指令来画。或者改用 filldraw 指令，这样可以既画框又填色，只不过，这样一来都是同一种颜色了，如果框线和所填的颜色要不一样，就得经过 draw 及 fill 两道手续。

7. 画圆

draw 后附加个 fullcircle，这画的是正圆。也可以画其他种类的圆：

| 圆的种类 | 指令 |
|---|---|
| 完整正圆 | fullcircle |
| 四分之圆 | quartercircle |
| 半圆 | halfcircle |

```
draw fullcircle scaled 1cm     % 圆心 (0, 0)，直径 1cm 的圆
draw fullcircle scaled 1cm shift (x, y)      % 圆心平移至 (x, y)
draw fullcircle xscaled a yscaled b     % 长短轴各为 a, b 的椭圆
```

前面曾出现的 with... 附加指令都可以使用。其他如半圆、四分之一圆的使用方法相同。

8. 画方框（unitsquare）

前面曾学过由画线连起来可以画正方形，但画方框有更简单的方式，那就是使用 unitsquare 附加指令，例如：

```
draw unitsquare scaled 1cm;    % 左下角是 (0, 0)，宽高 1cm 的正方形
draw unitsquare scaled 1cm rotated 30;       % 将正方形逆时针旋转 30 度
draw unitsquare xscaled 2cm yscaled 1cm;     % 宽 2cm，高 1cm 的矩形
```

而且，也可以使用 shift(x, y) 来平移左下角坐标的位置。

9. 标注文字（label, dotlabel）

我们先把标注文字时的位置弄清楚，再来看怎么标注上去：

```
                top（正上方）              ulft（左上角）  urt（右上角）
lft（左）        •        rt（右）                  •
                bot（正上方）              llft（左下角）   lrt（右下角）
```

由其中的英文字缩写，应不难明白他们的意思。中央的黑点，代表标注文字时所给的坐标位置。所以，我们的文字可以标示在一个定点的八个方位。例如：

```
label.bot("O(0,0)",  (0,0));          % 在原点正下方标示 O(0,0) 字样
dotlabel.bot("O(0,0)", (0,0)); % 和 label 一样，但会有个粗黑点
label.bot(btex  $O(0,0)$  etex,  (0,0));      % O(0,0) 字样经由 \TeX\ 排版
dotlabel.bot(btex  $O(0,0)$  etex, (0,0));
```

由 " 包围的，他的文字是由 METAPOST 自行处理，就是一般的文字表现，但由 btex...etex 包围的，则会交给 TEX 去排版，例如这里进入数学模式，就会以数学斜体来排版。

这里举一个简单的例子，rgb 三原色及灰阶的渐层演色表，顺便使用了我们没有提到的 for loop[4] 及调整字号的方法（这些会特别用底线标示出来）。

```
% test-mpcolor.mp
beginfig(1);
defaultscale := 12pt/fontsize(defaultfont); % 使 用 12 点 字
u := 1cm;
path sqr;
sqr := unitsquare scaled u;
for i=0 upto 10:    % for loop 循环，请注意这里是冒号
  label.top(decimal(i/10),   ((i+1/2)*u,1u));
  fill sqr shifted (i*u, 0) withcolor  i*0.1red;
  fill sqr shifted (i*u, -1.2u) withcolor i*0.1green;
  fill sqr shifted (i*u, -2.4u) withcolor i*0.1blue;
  fill sqr shifted (i*u, -3.6u) withcolor i*0.1white;
endfor;
label.lft("r", (0,.6u));
label.lft("g", (0,-.6u));
label.lft("b", (0,-2u));
label.lft("gray", (0,-3u));
endfig;
bye
```

由 mptopdf 执行的结果如下：

http://edt1023.sayya.org/tex/latex123/test-mpcolor.mp
http://edt1023.sayya.org/tex/latex123/test-mpcolor-1.pdf

### 9.5.5 和 LATEX 的配合

METAPOST 主要是用在 TEX 文稿，但由于他可以引入 TEX 宏，也因此表示可以把 LATEX 的一些指令及宏引进去，这样，编译的时候，该呼叫 tex 的，就会改呼叫 latex。当然，这也表示，我们可以使用 CJK 环境来书写中文了。

[4] METAPOST 有完整的条件判断式及循环，但这已经属于程序设计的范围，因此这篇文章里头并没有詳细说明，只有举一个 for loop 的典型例子，有需要深入时请自行参考文章后面的参考数据。

通常会要呼叫 tex 及 latex 的场合，就是要使用他们的排版功能，主要是用于排文数字，可以加入一般的文字，经过 TEX/LATEX 排版后的漂亮结果，也可以是高质量的数学式子。

1. 加入文数字
   只要是包在 btex. . . etex 里头的文数字都会交给 TEX 去排版。当然，仅仅使用 " 包住也是可以，但就不经过 TEX 处理了。

2. LATEX 的情形
   要使用 LATEX 要经过一些处理，因为 METAPOST 预设是把排版交给 tex 去编译的，这时要把所需要用到的 LATEX 宏及指令包在 verbatimtex. . . etex 当中，例如：

```
verbatimtex
%&latex             % 指示由 latex 编译，而不是预设的 tex
\documentclass{article}
\usepackage{some,packages}
\begin{document}
etex
   ...
verbatimtex         %   以下这段通常不必，但如果有引入其他环境时则不可省略
\end{document}      % 这段放在文稿最后即可
etex
```

这样一来，包在 btex. . . etex 之间的文字就可以使用 LATEX 指令去排版，而 mpost 也会交给 latex 去处理文字的部份。

### 9.5.6 在 METAPOST 中使用中文

METAPOST 文稿中，重点当然是画图，但是，有时也是要加入一些文字，这在英文是很方便，但在我们的 Big-5 中文就很头大了，这里我们使用了一个小工具 b5mp.pl，这个工具主要是引用王佑中[7]先生的 clatex 中的一个函式，把会出问题的 Big-5 中文处理好。然后，我们再加入必要的 LATEX 及 CJK 环境的结构，最后呼叫 mpost 来编译他，这个小工具是由 perl 写的，可以在此下载：

http://edt1023.sayya.org/tex/latex123/b5mp.pl

他的使用方法很简单，把他当成是 mpost 即可，使不使用 .mp 延伸档名都没有关系。如果 fontname 没有指定，那么预设是 aming：

```
b5mp.pl [fontname] your[.mp] => 这样会产生 your.1 (视图档内的编号而定)
perldoc b5mp.pl                => POD 格式的 b5mp.pl 使用说明
```

这个 your.1 就可以被引进 LaTeX 文稿里头去了。由于这个编译出来的 eps 文件，含有中文字型信息，所以 gv 及 gsview 都会无法阅览，epstopdf 也是无法处理，我们可以使用以下的方法来「预视」：

```
tex mproof your.1          => 这会产生 mproof.dvi
dvips mproof.dvi           => 产生 mproof.ps
或
dvipdfm[x] mproof.dvi      => 产生 mproof.pdf
```

这样就可以去预视了。当然，你的 LaTeX 系统要安装好 CJK 套件，否则还是会认不得这些中文字型信息的。mptopdf 除非另做些修改，否则在中文的场合会无法正常使用。我们这里就综合举一个例子：

```
% test-yi.mp
beginfig(1)
u:=3cm;
path p;
p=(0,1u)..(1u,0)...(0,-1u);
fill   p{dir(157)}..(0,0){dir(23)}..{dir(157)}cycle;
draw  p..(-1u,0)..cycle;
fill (0,-.6u)..(0.1u,-.5u)..(0,-.4u)..(-.1u,-.5u)..cycle withcolor white;
fill   (0,.6u)..(.1u,.5u)..(0,.4u)..(-.1u,.5u)..cycle;
label.bot(btex  \Large  仿太极阴阳鱼图 etex,(0,-1.2u));
endfig;

beginfig(2)
a=.7in; b=0.5in;
z0=(0,0);  z1=(a,0);  z2=(0,b);
z0=.5[z1,z3]=.5[z2,z4];
draw z1..z2..z3..z4..cycle;
drawarrow z0..z1;
drawarrow z0..z2;
label.top(btex  \small  横  轴  $x$  etex,  .5[z0,z1]);
label.lft(btex  \small  纵  轴  $y$  etex,  .5[z0,z2]);
label.bot(btex  \Large  许 功 盖 测 试  etex,(0,-.7u));
endfig;
end;
```

这里头有两张图，都有使用到中文，所以我们要使用 b5mp.pl 来代替 mpost 来编译：

```
b5mp.pl akai test-yi   % 使用 akai 字型，并产生 test-yi.1 及 test-yi.2
tex  mproof  test-yi.1  test-yi.2    % 产生 mproof.dvi
dvips  mproof.dvi                    % 产生 mproof.ps
dvipdfmx mproof.dvi                  % 产生 mproof.pdf
```

编译好的例子如下：

http://edt1023.sayya.org/tex/latex123/test-yi.mp
http://edt1023.sayya.org/tex/latex123/mproof.pdf

现在我们再来看看 LaTeX 文稿中要如何引用：

```
% example28.tex
\documentclass{article}
\usepackage{graphicx,CJK,mflogo}
\parindent=0pt
\ifx\pdfoutput\undefined
   \DeclareGraphicsRule{*}{eps}{*}{}
\else
   \DeclareGraphicsRule{*}{mps}{*}{}
\fi
\begin{document}
\begin{CJK}{Bg5}{hwmm}
这是一个 \MP\ 文稿中使用中文的例子，经由 {\tt b5mp.pl} 编译后引入
\LaTeX\ 文稿当中。图中的「太极阴阳鱼」字样是 \MP\ 文稿中就有， 不
是这里键入的。但其实图档里头并没有真正的字型，而是引入文稿， 经
{\tt latex} 编译后，才「合作」真正产生的。

\vspace{10ex}

\begin{figure}[h]
\centering
\includegraphics{test-yi.1}
\caption{太极生两仪}
\end{figure}

\vspace{10ex}

\begin{figure}[h]
\centering
\includegraphics{test-yi.2}
\caption{英文字是数学斜体}
\end{figure}
\end{CJK}
\end{document}
```

编译好的例子如下：

http://edt1023.sayya.org/tex/latex123/example28.tex
http://edt1023.sayya.org/tex/latex123/example28.pdf

### 9.5.7 更多的 METAPOST 的实例

这里只是简单的介绍了 METAPOST，实际要用的话，复杂一点的图形，可能会不太足够，Knuth 教授的 *The METAFONTbook*，这应该最详细的资料了，其中有些地方会和 METAPOST 不一样，但整个结构上是相同的，可和系统上就有的 mpman.ps 这个 METAPOST 作者 John D. Hobby 写的使用手册相互参照就可以清楚不同的地方。

关于 METAPOST 更多的例子，也另请参考 André Heck 所写的 *Learning METAPOST by Doing* 及 Urs Oswald、Vincent Zoonekynd 所举的实例及其解说，examples.zip 檔是 metapost.html 网页的原始码，收录在 CTAN 中：

http://remote.science.uva.nl/~heck/Courses/mptut.pdf
http://www.ursoswald.ch/metapost/tutorial.pdf
http://tex.loria.fr/prod-graph/zoonekynd/metapost/metapost.html
ftp://ctan.unsw.edu.au/tex-archive/info/metapost/examples.zip
http://www.topology.org/tex/conc/mp/mp.zip

最后一个 mp.zip 是 Alan U. Kennington 所著 *differential geometry reconstructed: a unified systematic framework* 一书中使用 METAPOST 绘图的实例原始码。

## 9.6 图形的引入

这里我们使用 graphicx package 来说明，他会自动引入 graphics package，这两个 package，主要是一些指令的参数用法不同，由于 graphicx 的参数用法弹性较大，而且也和 LaTeX 的一些参数的形式较符合，因此，我们就以 graphicx 来说明，引入宏时就引用 graphicx 就可以了。这一节所说的是外来的图档及 METAPOST 图档，而非由 picture 环境或 PSTricks 宏所绘制的图档。

### 9.6.1 引入外来图档的方法

讲绘图工具讲了老半天，到底有了图档，要如何引进文稿里头呢？就是使用 graphicx package 的 \includegraphics 指令：

```
...
\usepackage{graphicx}
...
\begin{document}
```

```
  ...
\begin{figure}                                  % 进入浮动环境
\includegraphics[选项参数]{图文件名称}
  ...
\caption{这里加入图的标题}                      % 图号会自动编号
\label{这里加入引用图文件时的文字标志} % 一定要在 caption 之后
\end{figure}
  ...
```

这样就行了，就这么简单！不使用浮动环境也是可以的。通常目前的 graphicx 宏会自动判断图文件格式，所以延伸档名可以不必写上。但如果引入的图档是 METAPOST 所产生的，那要附上延伸档名，通常是数目字，例如 some-graphic.1，但这在 pdflatex 可能会有误认文件格式的情形发生，所以，要让他知道这是 METAPOST 所产生的 mps 图档。因此，如果文稿中有引用 METAPOST 的图档，最保险的方式是：

```
\documentclass{article}
\usepackage{graphicx}
\ifx\pdfoutput\undefined
  \DeclareGraphicsRule{*}{eps}{*}{}
\else
  \DeclareGraphicsRule{*}{mps}{*}{}
\fi
\begin{document}
  ...
\includegraphics{mpost-image.1}
  ...
\end{document}
```

有一个 ifpdf 宏，做的也是同样的事情。这样的话，使用 pdflatex 编译时会把认不得的图档，当做是 mps；而使用 latex 编译时，会把认不得的图档，当做是 eps。另外一个方法，就是把 *.1 改延伸档名为 *.mps，这样 latex/pdflatex 都会认得他。

也可以在引用 graphicx 时加入文件格式选项参数，指定图档格式，但这里不建议这么做，这样有可能会使用文稿的弹性降低，如果不想改来改去，比较理想的方式是去修改 graphicx 的相关配置文件，但一般使用者恐怕也是搞不清楚这些配置文件要如何编修，因此还是使用条件式的判断来暂时解决这个问题。

如果图档中有 jpeg/png 图档，这在 latex/dvips 是无法处理的，可使用 convert 把他转成 eps 格式。在这种情形下，如果要让 latex/pdflatex 都能正常运作的话，可能要准备两种格式（eps/pdf）的图档了，然后，图档名称不指定延伸文件名，让执行程序各取所需。

有时使用图档时如果会有无法判断 BoundingBox 值的情形，在这种情形下，可以使用 dvipdfm 所附的 ebb 这支小工具来产生必要的 BoundingBox 值。

### 9.6.2 includegraphics 指令的选项参数

我们还没谈到 \includegraphics 选项参数的部份。这个选项参数很多，功能很强大。这些选项参数可以有多个，各选项间以逗点来分隔，他的值的设定是使用等号（请注意，我们这里谈的是 graphicx 宏，而不是 graphics，这在参数使用上不同）。

1. bb
   设定图档的边界（bounding box），含四个值，每个值以空白隔开。例如 bb=98 98 468 430，这个意思就是左下角的坐标是 (98, 98)，而右上角坐标是 (468, 430)，这个参考标准是可被印出纸张的左下角为 (0, 0)。请注意，如果没有指定单位的话，那默认是 bp。而且，这个设定在 pdflatex 会不被接受，此时请改使用 trim 选项参数。

   通常，这是会加上 clip 参数，作用是在修剪引入的图档的四周，但不是很好控制，所以建议图文件由图形处理或转换程序去处理过后再引入会比较好控制。不加 clip 参数，加个星号 \includegraphics* 作用是一样的。

   一般如果是 eps/ps 档，可以使用编辑器去修改他的 BoundingBox 值，无需用到这些不好控制的参数，如何抓到坐标值呢？使用 gv 或 gsview 把图文件加载后，将光标置于图中所要的位置，这些软件就会显示所在处的坐标，然后就可以依自己需要去修改他了。

2. clip
   修剪图的四周指定的边缘。

3. trim
   作用和 bb 一样，也是四个参数，但这里指的是要去除的部份长度值，而非相对于左下角的相对坐标。这个参数可以用在 pdflatex。例如：

   ```
   \includegraphics[trim=7 7 7 7, clip]{some}
   ```

   这会除去 some 这个图档的四周 7bp 的宽度。请注意，图档尽量不要加延伸文件名，让系统自己去判断，这样文稿会比较有弹性。

4. angle
   旋转的角度。旋转指的是逆时针的方向转的，除非使用负数的角度。

5. orgin
   旋转的中心点。

6. width
   这是指图形的宽度，会自动伸缩调整，长度亦会等比例调整。

7. height
   这是指图形的高度，会自动伸缩调整，宽度亦会等比例调整。

8. totalheight
   这是指图形的总高度，即 height 再加上 depth 的值。会自动伸缩调整，宽度亦会等比例调整。

9. scale
   按一定比例缩放，这没有单位，这是缩放倍数。

### 9.6.3 指定图文件的搜寻路径

如果图档很多，一个比较方便的方法就是在目前工作目录下，新开一个子目录来专门置放图档，这样在文件的维护上也会比较好维护。他的语法如下：

```
\graphicspath{{路径一}{路径二}{路径三}...}
\graphicspath{{images/}}      %  纵使只有一个子目录，也不可省略大括号
\graphicspath{{:images:}} % Mac 系统的表示法
```

LaTeX 系统默认找图文件的路径是 TeX 默认会去找的路径及目前的工作目录（通常目前的工作目录会先找）。或者，也可以修改 TEXINPUTS 变量的值（会较有效率，也较省内存），例如，以 unix-like 系统下的 sh shell 为例：

```
TEXINPUTS = "images/:" ; export  TEXINPUTS
```

这样会首先搜寻目前工作目录下的 images 这个子目录，找不到的话，才会去 TeX 默认的搜寻目录去找，这样一来就会很有效率，而且会较省内存。所以，这个方法比较建议使 用。

当然，也可以直接在 \includegraphics{} 的参数里头就直接把路径写进去，但这是最不建议的方法，不管效率或是文稿可移植性都会很差。

### 9.6.4 图文的旋转

我们常常会需要某些图文在特别的情况下旋转一下，\rotatebox 这个指令，其实我们前面举的例子当中就曾使用过，我们现在来看看详细的使用方法：

\rotatebox[选项参数]{角度}{图文对象}

角度和 \includegraphics 的 angle 选项参数一样，但使用方法则简化了，直接写上数值即可，当然预设是逆时针方向旋转。选项参数的部份可以有三个小选项：

1. origin
   设定旋转中心点的位置，可以使用 lrctbB 或其中两个的组合，其中 B 代表的是基线（baseline），其他的依其英文字母就可理解他的意义，如 t 是 top，r 是 right，c 是 center。默认的位置是左下角， 文字的话， 则是左下角的参考点（reference point），旋转就是以此点所构成的的轴心线来转的。

2. x, y
   这是以左下角为原点，直接设坐标，来表示 origin 所能表现的更精确中心点位置。

3. units
   设定旋转的特殊弧度。其中 units=-360，这样会把预设的逆时针旋转，变成顺时针旋转。

旋转不限于简单的图文对象，甚至一整个表格、图形环境都可以拿来转。要注意的是，编 译成 *.dvi 档的话，有可能 dvi viewer 会不支持旋转效果的解读，此时要把他由 dvips 来转成 ps 檔，或直接使用 pdflatex 编译成 pdf 档，再来预视。

### 9.6.5 图文的缩放及延展

1. \scalebox{水平缩放倍数}[垂直缩放倍数]{图文对象}
   垂直缩放倍数可以省略，省略时代表等于水平缩放倍数。

2. \reflectbox{图文物件}
   这其实是 \scalebox{-1}[1]{图文对象} 的意思，会得到镜射的效果。

3. \resizebox{宽度}{高度}{图文物件}
   \resizebox*{宽度}{总高度}{图文物件}
   这是在改变原图文对象的大小。宽高使用 ! 代替的话，会依另一个值做同比例的改变。如果加个星号，如 \resizebox*{}{}{}，他的作用是会把深度（depth，基线以下的称为深度，基线以上的称为高度 height，两者之和称为总高度，totalheight）也考虑进去，否则只考虑高度，但深度不一定会有，这时 height 就等于 totalheight。基线、深度等术语的意义，请复习一下第 3.3.1 小节，页 14。

我们在表格及图形的章节里花了比重不少的篇幅，主要的用意是想打破 LaTeX 所能使用领域的刻板印象，个人还觉得这些内容实在还不够，但要硬塞入这篇入门级的文件，已经不恰当，时间、能力允许的话，希望将来有机会另外专文来做更详细一点的介绍。

# 第 10 章

# 数学排版

好啦！这章是 LaTeX 的拿手把戏了。就让我们就来见识一下 LaTeX 的威力吧！光这一章的内容就可以写一本厚厚的书了，所以，只能点到为止，先小酌一番。这一章的内容，他们排版精确性，以 PDF 格式的内容为准，HTML 格式的内容，仅供参考。

对于较复杂的数学式子，除非是自行定义宏，否则 LaTeX 内建所提供的排版数学式子的能力可能会有不足，这时可以使用美国数学协会所开发的 AMS-LaTeX 宏套件，目前所有的 TeX 发行版应该都会附上，而且也会附上另一套 AMSFonts 宏套件及其字型。这个套件的使用，这里并不做详细的说明，只在必要的时候附带提及，可以另外参考系统上所附的 amsldoc.dvi 文件及 *The LaTeX Companion* 这本书第八章，这个部份网络上可以抓得到，檔名是 ch8.pdf，CTAN 有收录：

http://www.ctan.org/tex-archive/info/companion-rev/ch8.pdf

## 10.1 进入数学模式（math mode）的方法

我们平常写文章的模式无法正确处理数学式子间的空间位置，而且要键入次方、方根、积分……等等符号，会有困难，因此，所有的数学式子都得进入数学模式来处理。在数学模式下，不仅大部份文字、符号会采用斜体字，而且空间会另做安排，额外的空白会被 LaTeX 忽略，在数学模式中要键入一般的正常文字，要退出数学模式，或者由 \mbox{} 或 \textmr{} 包围起来才行。

LaTeX 的数学模式有两种，一种是和内文排列在一起的随文数式（math inline mode），他是和一般正常文字混在一起排版的；另外一种是独立的展式数式（math display mode），他会单独成一行，而且上下会和正常文字有一定的空间来区隔。

### 10.1.1 随文数式（math inline mode）

这是在夹杂在一般文章内的数学式子，是随着整个文章段落一起排版的。

1. $ 数学式子 $
   其实，我们在前面的章节的例子里，就已经常常在使用了，只是没有详细说明。由 两个钱字符号 $ 所包围的内容就会进入随文的数学模式，在一般文字段落内要使用到一些数学式子的话，这是最方便的方法。为什么是使用钱字符号？因为 Knuth 教授认为数学是很「昂贵」的！真正文章中要写钱字符号时，要把他 escape，写成 \$，大概是指，平常不必把钱看得太重的意思吧（这是我猜的）！:-)

2. \begin{math} 数学式子 \end{math}
   如果数学式子很长，那么使用环境的方式亦可。但是这个环境和一般的环境不同的 是，他不会在上下行区隔出来，而是随着其他正常文字一起排版的。要非常注意的 是，在这个环境的上下行不要留空白行，否则会另起段落排版，那就不是我们所要 的随文数式了。

3. \( 数学式子 \)
   这是 \begin{math} 数学式子 \end{math} 省略写法。

我们来试试看，到底进入数学模式和不进入数学模式会有什么不同：

```
f(x,y)=3x+4y                    % 不进入数学模式
f(x, y) = 3x + 4y               % 不进入数学模式，空白是有作用的
$f(x,y)=3x+4y$                  % 进入数学模式
$f(x, y) = 3x+ 4y$              % 数学模式中留不留空白，及留几个空白，作用都一样
sin(2x)=-sin x cos x            % 这样排版出来会惨不忍睹喔！
$\sin (2x) = -\sin x \cos x$
f(x,y) = 3(x+y)y / (2xy-7)      % 这样排版出来也是会惨不忍睹！
$f(x,y) = 3(x+y)y / (2xy-7)$
```

排版出来的结果会是：

| 误 | 正 |
|---|---|
| f(x,y)=3x+4y | $f(x,y) = 3x + 4y$ |
| f(x, y) = 3x + 4y | $f(x,y) = 3x + 4y$ |
| sin(2x)=-sin x cos x | $\sin(2x) = -\sin x \cos x$ |
| f(x,y) = 3(x+y)y / (2xy-7) | $f(x,y) = 3(x+y)y/(2xy-7)$ |

可以看得出来，英文字的部份变成斜体字了，而且加号、逗点、等号前后的空白也不一样。但是函数名则还是使用正常字体，这在后面第 10.1.3 会谈到数学模式中的游戏规则。

### 10.1.2 展式数式（math display mode）

通常独立的数学式子，我们不会使用一般文章一样的做法去换行，而是让他进入展式数式的数学模式，他会独立成一行，有需要的话也可以加入编号，以方便在文章中引用。和随文数式另一个很大的不同是，展示数式会适当的选用较大的数学符号及字体，尤其是较复杂的数学式子的时候。

1. \begin{displaymath} 数学式子 \end{displaymath}
   这会使数学式子独立成一行。

2. \[ 数学式子 \]
   这种方式也可以，也比较常用。这两种的展示数式都不会编号。

3. \begin{equation} 数学式子 \end{equation}
   这种使用方式，亦会独立成一行，而且会附上编号。equation* 则不附编号。

使用展式数式要注意的是和上下文章不要空出空白行出来，里头也不要空出空白行。请不要使用 TEX 里头的 $$ 指令，这在 LATEX 并没有完整去重定义他，这在某些 LATEX 指令的效果上会没有作用。

### 10.1.3 在数学模式中的一些游戏规则

在数学模式中，由于一些空间的安排和一般文章段落不一样，因此在编辑文稿时，会有一些地方需要注意。

1. 关于标点符号
   在数学模式中，我们要注意一下标点符号的问题，一般而言，数学式后面如果有标 点符号，在随文数式，这个标点符号不能纳入数学模式中；反之，在展式数式的场 合，这些标点符号则要纳入数学模式中。例如：

```
Let $f(x)=\sqrt[4]{x+1}$ and $g(x)=\sqrt{9-x^2}$,... % 逗点不纳入数学模式
Let
\[ f(x)=\sqrt[4]{x+
1}
\]
and
\[
```

```
g(x)=\sqrt{9-x^2},          %  逗点纳入数学模式，标点符号也独立成行
\]
...
```

所以，展式数式，如果数学式最后有个句点或逗点的话，请不要怀疑，你搞对了！:-)

2. 数学模式的斜体字

数学模式里头，默认会使用斜体字，但这些斜体字是数学斜体，和一般文章中的斜 体是不一样的，他字母间的距离比较宽，也没有所谓的连体字（ligature），因此，如果需要这些效果，可以指定要使用斜体字，这样就会表现和一般文章一样的斜体 了。例如：

```
\textit{proffer} normal italic.     % 正常文章的斜体
$proffer\ math\ mode\ iatlic.$      % 数学斜体
$\textit{proffer\ math\ mode\ normal\ italic.}$ % 指定为正常斜体
```

表现出来会是：

*proffer normal italic.*
*prof fer math mode iatlic.*
*proffer math mode normal italic.*

当然，这种情形很少发生，正常排版的话，无需特别去指定使用一般文章的斜体。

3. 例外不使用斜体字的情形

一般函数名是不使用斜体字的，例如 log、三角函数名⋯⋯等等，为了避免失误打错，可以直接使用指令的方式，例如 \log、\sin、\tan⋯⋯等等，这样虽然是在数学模式中，也会使用一般的正常字体。TEX/LATEX 系统提供了预先定义好的 32 种函数名供使用：

```
\arccos  \cos   \csc  \exp  \ker     \limsup  \min
\arcsin  \cosh  \deg  \gcd  \lg      \ln       \Pr
\arctan  \cot   \det  \hom  \lim     \log      \sec
\arg     \coth  \dim  \inf  \liminf  \max      \sin
\sinh    \sup   \tan  \tanh
```

这样往后只要是函数名就直接在数学模式中使用这些现成的指令就行了。当然，如 果是这里没有涵盖的函数名，就得自行加以注意了。

另外，单位名、化学元素、数字、简写缩写文字等都不使用斜体字。但例外的例外，物理中的常数名则仍然是要使用斜体字，例如光速 $c$。

4. 不要加入换行指令或插入空白行
   一个数学式子，在 LaTeX 是视为一个单独的单位或段落，而且，在这个特殊的段落里，LaTeX 会抑制 line break 及 page break 的机制，所以，除非是矩阵及矩阵方程式外，不能去强迫换行也不能插入空白行。

## 10.2 数学符号

我们打字，通常是无法打出一些特殊数学符号，纵使字型里头有这种符号，但由于要和其他符号、文字调整他们的相对位置，因此，除了一些常用的运算符号外，数学符号通常是使用指令的方式来键入。我们在第 7.2.8 小节里头曾提到 symbols-a4.pdf 符号表，这个总表非常重要，几乎罗列了目前所有可用的现成符号，并且会标明需要引入什么 package，所以，这里就不把符号表列出来，以节省篇幅。

当然，有些编辑器的宏会设定好方便的按钮方式来插入这些数学符号，但建议开始接触 的时候，多花点时间亲自键入，等熟悉以后再来使用这种方便的设定来增加生产力。理由 是，编译错误或校稿修改时才知道要改什么地方，连 \sum、\infty、\int 是什么符号都不知道的话，那么要微调就变成很困难了，而且什么地方错误也常会搞不清楚，这些后续 的动作所花的时间，可能会比你刚开始学指令所花的时间还多。

## 10.3 各种数学式子的书写方法

我们这里就正式来看看数学式子到底是如何书写，这里不做符号的列表，直接举例子，如对相关符号指令的书写有疑问，请自行查阅 symbols-a4.pdf。

### 10.3.1 分式（fraction）

| 书写方式 | 排版结果 |
| --- | --- |
| $f(x,y)=3(x+y)y/(2xy-7)$ | $f(x,y)=3(x+y)y/(2xy-7)$ |
| $f(x,y)=\frac{3(x+y)y}{(2xy-7)}$ | $f(x,y)=\frac{3(x+y)y}{(2xy-7)}$ |

简单的分式，直接使用 / 就可以了，否则就要使用 \frac{分子}{分母} 这个分式的的指令。我们可以看到，随文数式为了和前后文配合，分式的情形会把字母、符号缩小，如果太复杂的分式，就不适合使用随文数式了，要把他单独列出来：

```
\[
f(x,y)=\frac{3(x+y)y}{(2xy-7)}
\]
\begin{equation}     % 有编号的情形
f(x,y)=\frac{3(x+y)y}{(2xy-7)}
\end{equation}
```

这样，排版出来的结果是：

$$f(x,y)=\frac{3(x+y)y}{(2xy-7)}$$

$$f(x,y)=\frac{3(x+y)y}{(2xy-7)} \tag{10.1}$$

请注意，和上下行的原来文字不要有空白行，展式数式会自动处理。更复杂的分式的例子：

```
\[
\frac{\frac{a}{x-y}+\frac{b}{x+y}}{\frac{x-y}{x+y}+\frac{a-b}{a+b}}
\]
```

排版出来的结果是：

$$\frac{\frac{a}{x-y}+\frac{b}{x+y}}{\frac{x-y}{x+y}+\frac{a-b}{a+b}}$$

如果觉得，字体似乎太小了，可以指定字体：

```
\[
\frac{\frac{\displaystyle  a}{\displaystyle x-y}+
\frac{\displaystyle  b}{\displaystyle x+y}}
{\frac{\displaystyle  x-y}{\displaystyle x+y}+
\frac{\displaystyle  a-b}{\displaystyle a+b}}
\]
```

排版出来的结果是：

$$\frac{\frac{\displaystyle a}{\displaystyle x-y}+\frac{\displaystyle b}{\displaystyle x+y}}{\frac{\displaystyle x-y}{\displaystyle x+y}+\frac{\displaystyle a-b}{\displaystyle a+b}}$$

可指定的字体大小有：

| | |
|---|---|
| \displaystyle | 展示数式的标准字体大小 |
| \textstyle | 随文数式的标准字体大小 |
| \scriptstyle | 第一层上下标字体大小 |
| \scriptscriptstyle | 第二层上下标字体大小 |

### 10.3.2 上下标

| 书写方式 | 排版结果 |
| --- | --- |
| $(a+b)^2=a^2+2ab+b^2$ | $(a+b)^2=a^2+2ab+b^2$ |
| $\cos 2x=\cos^2x-\sin^2x$ | $\cos 2x=\cos^2 x-\sin^2 x$ |
| $(y^m)^n=y^{mn}$ | $(y^m)^n=y^{mn}$ |
| $^a_bY^c_d$ | ${}^a_bY^c_d$ |
| $e^{t \cos\theta}$ | $e^{t\cos\theta}$ |
| $\lim_{n \to \infty}\sum_{i=1}^n{\frac{1}{n}}$ | $\lim_{n\to\infty}\sum_{i=1}^{n}\frac{1}{n}$ |
| $y_1=1/3(x_1+\omega x_2+\omega^2x_3)$ | $y_1=1/3(x_1+\omega x_2+\omega^2 x_3)$ |

需要注意的是，不管是上下标，如果里头有两个以上的字符都要当做上下标时，要使用大括号把他括住，否则会只作用在第一个字符而已。而且，上下标是左右两边都能标注的。这里的 \to 指令，是 \rightarrow 的简写，就是向右的单箭号。相对的向左的单箭号 \leftarrow，他的简写是 \gets。

另外，像 \sum、\lim、\int 这类符号，如果是在展示数式的时候，他的上下目标表现会和随文数式不一样，符号也会比较大，例如：

```
\[
\lim_{n \to \infty}\sum_{i=1}^n{\frac{1}{n}}\sin\frac{k}{n}
\]
```

表现出来会变成：

$$\lim_{n\to\infty}\sum_{i=1}^{n}\frac{1}{n}\sin\frac{k}{n}$$

### 10.3.3 根号

| 书写方式 | 排版结果 |
| --- | --- |
| $\sqrt{x^2+y^2}$ | $\sqrt{x^2+y^2}$ |
| $\sqrt[5]{a+\sqrt{b}}$ | $\sqrt[5]{a+\sqrt{b}}$ |

## 10.4 矩阵（array）

矩阵的排版方式和第 8.3 节所谈的 tabular 表格类似，也是以 & 来区隔字段，以 \\ 来换行，只不过，矩阵的情形是在数学模式里头。

其中的分界符号（delimiter），在 LaTeX 是由 \left 及 \right 指令来引导，这些分界符号会随里头式子的多寡，自动重设大小。至于有什么分界符号可以使用，也请自行查一下 symbols-a4.pdf。我们来看例子：

```
\[
A = \left(     % 视 \left 后面跟的是什么分界符号，就是使用什么
\begin{array}{clr}
  t_{11} & t_{12} & t_{13} \\
  t_{21} & t_{22} & t_{23} \\
  t_{31} & t_{32} & t_{33}
\end{array}\right)
\]
```

排版结果是：

$$A = \left( \begin{array}{clr} t_{11} & t_{12} & t_{13} \\ t_{21} & t_{22} & t_{23} \\ t_{31} & t_{32} & t_{33} \end{array} \right)$$

是不是和排版表格一样呢？指定的位置 lcr 的意思，和排版表格时是一样的。如果右边不需分界号，那么可以使用 \right. 来代替，例如：

```
\[
g(x,y)  =  \left\{\begin{array}{ll}
                f(x,y),  &  \mbox{if $x<y$} \\     % 文字的部份要用 \mbox
                f(y,x),  &  \mbox{if $x>y$} \\     %  包住，让他使用正常字体
                0,       &  \mbox{otherwise.}
               \end{array} \right.
\]
```

排版的结果是：

$$g(x,y) = \left\{ \begin{array}{ll} f(x,y), & \mbox{if } x<y \\ f(y,x), & \mbox{if } x>y \\ 0, & \mbox{otherwise.} \end{array} \right.$$

AMS-LaTeX 另外提供了 matrix 环境，这是没有分界符号的，另外有固定分界符号的环境：

| 環境 | 分界符號 | 環境 | 分界符號 |
|---|---|---|---|
| matrix | 無分界號 | bmatrix | 方括號 [ ] |
| vmatrix | 单垂直线 \| \| | Vmatrix | 双垂直线 \| \| \| \| |
| pmatrix | 小括號 ( ) | Bmatrix | 大括號 { } |

这样就可以不必使用 \left 及 \right 来引导出分界符号，我们把上面的例子再排版一

次：

```
  ...
\usepackage{amsmath}                % 要记得引用 amsmath 宏
  ...
\[
A =
\begin{pmatrix}                     % 使用小括号
  t_{11} & t_{12} & t_{13} \\
  t_{21} & t_{22} & t_{23} \\
  t_{31} & t_{32} & t_{33}
\end{pmatrix}
\]
```

排版出来的结果是：

$$A = \begin{pmatrix} t_{11} & t_{12} & t_{13} \\ t_{21} & t_{22} & t_{23} \\ t_{31} & t_{32} & t_{33} \end{pmatrix}$$

看起来会比较紧凑一点。matrix 环境仍然可以使用 \left 及 \right 来加上分界号。

### 10.4.1 矩阵方程式

LaTeX 有一个内建的 eqnarray 环境，来排版矩阵方程式，但这个环境对数学式子的空间安排会有潜在的瑕疵，因此，许多 LaTeX 专家建议使用 AMS-LaTeX 所提供的 align 环境。因此，这里只讨论 align 环境。而且，使用 align 环境的好处是，每一个数学式子只需要一个 & 即可，排版出来会依这个 & 来对齐，如果有数个数学式在同一行，那各个数学式子也是使用 & 来区隔。所有式子要向左对齐的话，只要把 & 置于行首就可以了。

align 环境中的每行数学式子都会加以编号，要不编号的话，可在每行数学式子换行符号前加个 \notag 指令，这一行便不会编号，要所有的数学式子都不编号的话，就使用 align* 环境。要使用特别指定的符号来编号的话，可使用 \tag{符号} 放在这一行的换行指令前。使用 subequations 环境则会有子编号。我们来看个例子：

```
  ...
\usepackage{amsmath}        % 要记得引用 amsmath package
  ...
\begin{subequations}        % 让编号同数，但以英文小写为子编号
\begin{align}                 % 韩信点兵，同余方程式
x & \equiv 2 \pmod 3 \\     % 三三数之剩二x &
\equiv 3 \pmod 5 \\         % 五五数之剩三x &
\equiv 2 \pmod 7      % 七 七 数 之 剩 二
```

```
\end{align}
\end{subequations}
```

排版出来的结果是：

$$x \equiv 2 \pmod 3 \tag{10.2a}$$
$$x \equiv 3 \pmod 5 \tag{10.2b}$$
$$x \equiv 2 \pmod 7 \tag{10.2c}$$

## 10.5 定理

LaTeX 有一个 \newtheorem 指令来定义，Theorem, Lemma, Definition· · · · · 等等环境，但他的功能算是满阳春的，因此，这里也附带说明美国数学协会的 amsthm package 的使用，他仍然是建立在 LaTeX 的 \newtheorem 而来的，因此可以配合 LaTeX 原有的定义。

### 10.5.1 原始 LaTeX 的定义

定理环境，在 LaTeX 里头是要我们自行去指定环境名称，他的语法是：

```
\newtheorem{环境名称}{定理名称}[章节层次]
```

编号如要加入章节编号，可加入选项参数的部份，例如 chapter 则会按 chapter 的编号来成为定理、定义的编号。我们来看看实际的例子：

```
 ...
\newtheorem{defi}{Definition}        % 在 preamble 区先定义好环境名称
 ...
\begin{defi}
Let $f$ be continuous on the half-open interval $[a, b)$ and suppose
$\lim_{x\rightarrow  b^-}|f(x)|=\infty$. Then,
\[
\int_a^bf(x)dx=\lim_{t\rightarrow  b^-}\int_a^tf(x)dx
\]
provided this limit exists and is finite, in which case we say the
integral converges. Otherwise, we say it diverges.
\end{defi}
```

表现出来的将会是：

**Definition 1.** *Let $f$ be continuous on the half-open interval $[a, b)$ and suppose $\lim_{x\to b-} |f(x)| = \infty$. Then,*

$$\int_a^b f(x)dx = \lim_{t\to b^-} \int_a^t f(x)dx$$

*provided this limit exists and is finite, in which case we say the integral converges. Otherwise, we say it diverges.*

如果是定理的话，接着可能需要排版证明，我们可以直接使用 amsthm package 的 proof 环境，来排版证明的部份。

### 10.5.2 amsthm 宏套件

我们在上一个小节里头看到了 LaTeX 预设的定理、定义环境，但弹性很小，这里我们来使用 amsthm 宏套件，并且试着排版中文：

```
...
\usepackage{CJK}                          % 引入 CJK 环境
\usepackage{amsmath,amsthm,amssymb} % 引入 AMS 数学环境
\theoremstyle{remark}                     % 内文使用正常字体
\newtheorem{cdefi}{\bf 定义}              % 改用粗体，预设 remark style 是斜体
...
\begin{CJK}{Bg5}{hwmm}
\begin{cdefi}
设函数 $f:[a,b]\rightarrow\mathbb{R}$ 为可微分，且 $(f')^2$ 为可积，则称
\[ L(s)=\int^b_a\sqrt{1+(f'(x))^2}
dx
\]
为 $f$ 之图形自点 $(a,f(a))$ 至点 $(b,f(b))$ 之弧 $s$ 的弧长。
\end{cdefi}
\end{CJK}
```

排版出来的结果是：

定授 1. 投函敷 $f$ : $[a, b] \to$ R 民可微分，且 $(f')^2$ 民可積，则朊

$$L(s) = \int_a^b \sqrt{1 + (f'(x))^2}dx$$

$$L(s) = \int_a^b \sqrt{1 + (f'(x))^2}dx$$

民 $f$ 之圆形自贴 $(a, f(a))$ 至贴 $(b, f(b))$ 之弧 $s$ 的弧长。

其中 theoremstyle 有三种 style，plain, difinition，remark，预设使用的是 plain，即定理名称会用粗体字，内文则是 italic 斜体字，但这在中文并不适当，所以改由 remark style 来排版。要注意的是定理环境内的 plain style 虽然使用 italic 字型，但他并没有进

入数学模式，因此里头的数学式子仍然要进入数学模式来排版。另外，实数的那个特殊符 号 R 要引用 amssymb package 才会有，这也是美国数学协会发展的。

amsthm 另提供了 \theoremstyle 给使用自行定义除了 plain、definition 及 remark 三种形式外的 style。另外，LaTeX 本身的工具组也提供了一个 theorem 宏套件，能更有弹性的来细部微调，也有现成的指令，可以去改变所要使用的字体，可以参考系统上就有的 theorem.dvi 这个说明档。

## 10.6 数学模式中的字型及空间调整

我们前面已有谈到调整数学模式字体大小的四个指令。这里我们再来看看其他的调整指令。

### 10.6.1 数学字体的改变

以下的指令，相信大家从他的简写就可以知道意思：

| 指令 | 作用 | 实例 |
|---|---|---|
| \mathrm | 正常字体 | ABCabc |
| \mathtt | 打字机字族 | ABCabc |
| \mathbf | 粗体字 | **ABCabc** |
| \mathsf | sans serif | ABCabc |
| \mathit | italic 斜体 | *ABCabc* |
| \mathcal | 数学花体字 | *ABC* |

其他的套件会有更多的不同字体，请参考系统里头的 symbols-a4.pdf。要注意的是，有些字型并不是都完全有各种字体组合，像花体字并没有小写字母。

### 10.6.2 数学模式中调整间距

正常情况下，数学模式中的空间调整应不必用户去操心，但程序毕竟不会思考，有些特殊场合仍然需要人为的调整。

| 指令 | 作用 | 指令 | 作用 |
|---|---|---|---|
| \quad | 空出一个 em 单位的空白 | \qquad | 空出两个 em 的空白 |
| \, | 加入 1/6 quad 的空白 | \! | 减去 1/6 quad 的空白 |
| \; | 加入 5/18 quad 的空白 | \: | 加入 2/9 quad 的空白 |

其中 \,、\quad 及 \qquad 可以用在一般的文本模式及数学模式，其他的只能使用在数学模式中。我们把上一个弧长定义的例子来试着调整一下：

```
...
\begin{cdefi}
设函数 $f:[a,b]\rightarrow{\mathbb R}$ 为可微分，且 $(f')^2$ 为可积，则称
\[ L(s)=\int^b_a\!\sqrt{1+(f'(x))^2}\,d
x
\]
为 $f$ 之图形自点 $(a,f(a))$ 至点 $(b,f(b))$ 之弧 $s$ 的弧长。
\end{cdefi}
...
```

排版出来的结果是：

定授 1. 投函敷 $f$ : $[a,b] \to \mathbb{R}$ 民可微分，且 $(f')^2$ 民可稹，则朊

$$L(s) = \int_a^b \sqrt{1 + (f'(x))^2}\,dx$$

$$L(s) = \int_a^b \sqrt{1 + (f'(x))^2}\,dx$$

民 $f$ 之圆形自贴 $(a, f(a))$ 至贴 $(b, f(b))$ 之弧 $s$ 的弧长。

但是，要注意的是，如果文章有点长，数学式子也不少，那一定要注意整体的一致性，要 调整的话就全文相同的地方都要去调整，否则就使用默认值就可以了，至少他不会太离谱。

# 第 11 章

# 一篇文章、一本书的完整结构

好了，写文章最后也要整理成册，这也是排版系统要负责的部份。如果只是简单几百、几 千字的小文章，那很容易，只要个文章题目，章节标题，那也就够了。但如果是较正式的论文，那可能还有目录、参考文献、索引·····等等，甚至一本书籍的话，也要有个封面，及送印刷厂时要用到的裁切记号（crop marks）。如果要置放在网页上的，那还得注意网络超链接互动的问题，所以，这些细节算是满琐碎的，但却是必要的。

当然，个人也并不是什么排版、印刷的专家，只能谈谈我所知道的事项，如果需要补充或修正，请有这方面经验的朋友，不吝提供心得及指正。个人出版，这实际上不是梦，尤其网络发达的今日。

## 11.1 目录（Contents）

目录的问题，如果不讲究的话，使用 LaTeX 预设的就行了。就像第 4.4 节所举的例子一样。但如果要做调整的话，除非熟悉 LaTeX 宏的写法、定义，否则就得使用现成的宏套件，例如 minitoc 可让目录更紧凑，titletoc 更可做相当幅度的调整及美化。

在 LaTeX 文稿内，\tablofcontents 可以排版一般的章节目录。\listoffigures 指令可以排版图目录，\listoftables 指令则可排版表目录。但图表的话是指有进入浮动环境，使用 \caption 指令，有编号的图表而言。请注意，这些目录指令的置放位置会影响实际目录出现的顺序，没有特殊需求的话，一般的顺序是文、图、表。

### 11.1.1 更改目录标题名称

默认的情形下，在目录开头都会有个标题来引导，例如：**Contents**、**List of Figueres** 及 **List of Tables** 等，但是这在中文的情形看起来会不相称，我们可以去更改默认值。更

改 LaTeX 默认值得视原来这个值是以什么形式出现，在目录是以指令定义的形式出现，所以我们要使用 \renewcommand 这个指令去重定义他。

原来的这些 **Contents** 标题是怎么「弄」出来的呢？如果手头上没有相当的参考书籍，可以参考他的原始定义，例如这篇文章是使用 report class，那么找一下：

```
/usr/share/texmf/tex/latex/base/report.cls      % Unix-like 系 统
C:\texmf\tex\latex\base\report.cls               % DOS/Windows 系统
```

这个档（依安装的地方不同，可能会有不同的路径），搜寻 Contents 这个关键词，就可以发现，他们原来的定义是：

```
\newcommand\contentsname{Contents}
\newcommand\listfigurename{List of Figures}
\newcommand\listtablename{List of Tables}
```

这样就清楚了，我们要重指定的是 contentsname、listfigurename 及 listtablename。其他的情形请依此类推。现在我们来把他改成中文：

```
\renewcommand\contentsname{目~录~}
\renewcommand\listfigurename{图~目~录}
\renewcommand\listtablename{表~目~录}
```

这里以 CJK 宏为例，由于我们需要中文环境，所以这些更改要放在 CJK 环境中，如果只是更改成其他英文字样，那我们置于 preamble 区就可以了。

### 11.1.2 目录的深度

通常，有编号的章节或有 caption 的图表才会编入目录中，但如果想让目录的结构更细，那么我们就得更改列入目录的深度。目录深度的表现形式是一种计数器（counter），他的名称是 tocdepth。以这篇文章的 report class 为例，他的默认值是（请自行查一下 report.cls）：

```
\setcounter{tocdepth}{2}
```

所以会计算到 subsection，以下的就不列入了（请参考第 3.4.4 小节的章节深度标号）。我们只要在 preamble 区，使用 \setcounter 指令去重新指定，就会改变他的目录深度。

### 11.1.3 额外的目录

这是指没有编入目录， 但想自行加进去的情形， 例如： 章节指令使用了星号就不会编号，图表目录没有使用 \caption 指令， 也不编入目录了， 这时我们可以使用 \addcontentsline 指令来把他们手动加进去。我们来看看文图表的三种不同情况：

```
\addcontentsline{toc}{章节名}{标题}
\addcontentsline{lof}{figure}{标题}
\addcontentsline{lot}{table}{标题}
```

这样就会把这些纳入目录，但是，这还是没有编号的。目录中所显示的页数，就是这些指令（图表）所在的页数。

## 11.2 交互参照（Cross References）

所谓的参照，指的是在文章某处提及某个其他的章节，或某个页数，甚至是某个图表，某个数学式子及某个列举项目，排版系统必需要有这样的功能来自动达成这种效果，而 LaTeX 本身提供了三个简单易用的指令来自动处理，他会自动计算相对的章节、页数。

当然，由于网络的发达，超链接上的交互参照也变得是不可或缺，但 TeX/LaTeX 毕竟是平面排版系统，并没有这样的原始功能，但我们可以经由宏套件来达成这样的目的，hyperef 宏套件就是为此而写的，这样就可以让 LaTeX 排版的结果去转换成 PDF/HTML 格式的时候，也有超链接的功能。

广义的来说，包括目录的参照、文献参照、批注的参照及外部档案的参照（例如，参照某个外部档案的某个章节）都是属于交互参照的一部份，但这些议题我们另外单独讨论，因为他不在 LaTeX 所提供的三个基本参照的指令范围内。

### 11.2.1 一般的交互参照

LaTeX 提供了三组基本参照的指令：

```
\label{名称}      %  置放于要被引用之处，以一个名称来标记他
\ref{名称}        % 引用 \label 所标记处的章节
\pageref{名称}    % 引用 \label 所标记处的页数
```

这里头的名称都是自行取名的，但为了避免重复，个人使用上一般使用上会加入章节或图表的代号，例如：

```
...
\section{\LaTeX\ 的文稿结构}
\label{sec:struct}
...
\begin{figure}
\includesgrapics{fontstruct}
\caption{字型的结构}
\label{fig:struct}
\end{figure}
...
请参考第 \ref{sec:struct} 节，页 \pageref{sec:struct}。请
参考图 \ref{fig:struct}，页 \pageref{fig:struct}。
...
```

这两个 struct 代表不同的参照处，当然，尽量避免这种情形发生，可加入 fontstruct 之类的来区别，但前面冠上章节、图表的简名，有助于看文稿时清楚区别。请注意，其中 sec:、fig: 都不是必要的，只是这样比较容易辨识，而且不容易名称重复。

要非常注意的的几个重点是：

1. 有参照的文稿一定要编译两次才能正常显示。
2. 能编号的章节、图表、列举项目、数学式、定理才能参照，虽然他们不一定要编号。
3. 图表的参照 \label 一定要在 \caption 之后，不能在前。

### 11.2.2 超链接交互参照（hyperlink）

这当然是要像 PDF/HTML 格式的档案才有超链接交互参照的可能，像 *.dvi、*.ps 这类格式的档案，先天上并没有这种设计。而 LaTeX 本身并没有内建这种功能，我们可以使用 hyperref 宏套件来达成这个目的。现在一般的 TeX 发行版应该都会附上这个宏，如果没有话，可以在以下网站下载、安装：

ftp://ftp.tug.org/pub/tex/hyperref/

这个套件会让原本 LaTeX 有交互参照的地方，在制作成 PDF 格式时也会有超链接的功能。他的配置文件是 hyperref.cfg，为了各种文稿使用上的弹性，可以把这个文件在工作目录上建立一个，这样会依这个配置文件来执行，可参考本文文稿的原始码，里头会有一个设 定档供参考。当然也是可以在 preamble 区来设定，但这就只能使用在特殊的文稿上了。

他的使用方法，这里不多做说明，可以参考本文的原始码里头的使用方法，或参考〈由 TeX/LaTeX制作中文 PDF 檔〉一文：

http://www.study-area.org/tips/latex/chpdf.html
http://www.study-area.org/tips/latex/chpdf.pdf

及 hyperref 所附的使用手册。

## 11.3 索引（index）

索引的排版方法上并不算困难，困难的是要选出哪个字词需要索引，及把各个字词加入索引指令。我们引用 LaTeX 的标准宏 makeidx，并在其他加上一个 makeindex 指令，然后在文稿结束前印出索引，下 printindex 就可以了。我们在需要编入索引的名词后加上 \index{名词} 经过编译后就会自动把索引及其相对的页数计算出来。

### 11.3.1 索引的结构及编译

我们来看看文稿里头要加入什么要件：

```
...
\usepackage{makeidx}
\makeindex
...
要索引的名词\index{要索引的名词}
...
\printindex             % 一定要有这个指令才会印出索引
\addcontentsline{toc}{chapter}{索引}    % 把他加入目录
...
```

编译的的程序如下：

```
latex your.tex
makeindex your.idx
latex  your.tex
```

### 11.3.2 索引值的制作

索引值（key）里头 |、@ 及 ! 有特殊的意义，要索引他们时前面要加 " 来 escape 他。我们来看这些符号实际上有什么作用：

| | |
|---|---|
| abc\index{abc} | 这是一般正常的索引 |
| xyz\index{abc!xyz} | 表示 xyz 是 abc 下的一个子索引 |
| abc\index{abc\|textit} | 表示这个索引值的页数使用 italic 斜体排版 |
| abc\index{abc@\textbf{abc} | 表示索引值是 abc，但使用粗体排版 |
| \'abc\index{abc@\'abc} | 表示依 abc 来排序索引，而不是后面的 ábc |

制作索引是需要细心与耐心的，这方面更详细的数据可以参考系统上的 makeindex.dvi 及 manpages.dvi。要注意的是 \index{} 最理想是紧接在要索引的名词后，前后都不留空白，有多个 \index{} 相连时亦同，这会让文件维护增加困难，因此，视每个人的习惯，可以考虑索引在整篇文稿最后才加进去。这份文件也制作了简单的索引，但这只是当个例子供参考，在制作上有点粗糙，因此，实际上可能会漏掉很多，而且，中文的处理仍有待加强。

制作索引的时候， 他的表示法要细心的注意一下， 前后相同索引值的表示法要一样，例如 \index{abc@\textbf{abc} 和 \index{abc@{\bf abc} 这会造成两个不同的索引，虽然印出来的是一样。而且，\verb|abc| 这种方式就行不通，因为 | 这个符号在索引指令内有他的特殊作用，要改用其他的符号代替。如果是和 hyperref 配合的话，abc\index{abc|textit} 也会行不通，因为 hyperref 对超链接的索引是自动加上 |hyperpage，如果已经有 |textit 了的话，就不会加上去了，这样一来超链接的部份会被忽略，解决的方法只能去重定义索引方法，或在编译出来的 *.idx 或 *.ind 上做另外处理。例如：

```
abc\index{abc|textit}
latex 编译后的情形是：
\indexentry{abc|textit}{143}          % *.idx 檔
经 makeindex 编译后的情形是：
\item abc, \textit{143}               % *.ind 文
件这样只编号改变字体，并没有超链接。而我们要的是：
\item abc, \textit{\hyperpage{143}}   % 这样才能又变更编号字体又能超链接
```

这个议题比较深入一点，解决的方式可能需要大家一同来研究、研究，不是不能解决，而是方式在使用上是否方便的问题。

索引的处理，他的信息实际上是产生在编译后的 *.idx 文件里头，然后经由 makeindex 外部程序编译后，转换成 *.ind，然后 latex 再次编译的时候，才把这个 *.ind 引进来，这个 *.ind 其实就是一个 LaTeX 的文稿，他把所有的索引值及页数，包在一个 theindex 环境中来引入排版的。

说明这些的用意就是暗示，我们可以由外部处理程序去动手脚，把索引的部份再「加工」，包括中文数据的处理也是一样，下一节要谈到的参考文献的处理机制也是类似的情

形，而这也就是为什么 latex 要执行好几次的原因，也正因为这样，我们才有动手脚的机会，例如 makeindex 就有 -s 参数，可以接受外部的 style 档，或者，如果工作目录上有 *.mst（makeindex style）这个档，也会优先去参考锺，这样就可以产生不同形式的索引结果。当然，参考文献可以另单独的一个文献外部文件，索引的话，目前则没有办法这样做，是否可以比照参考文献的做法，由外部档案来处理呢？就请大家脑筋急转弯一下了，这样也可以让文稿更容易维护。

### 11.3.3 更改索引标题

默认的索引标题是 **Index**，我们可以在 preamble 区来更改他（中文的话，请放在本文区 CJK 环境内），例如，设要中文名称的话，可更改为中文：

```
\renewcommand\indexname{索~引}
```

## 11.4 参考文献（Bibliography）

参考文献可以经由 LaTeX 内建的 thebibliography 环境来制作。长篇文稿，也可以使用 BibTeX 由外部档案来制作。至于参考文献的格式，就要符合邀稿单位的规格了，这里不多做说明。

### 11.4.1 thebibliography 环境

在进入 thebibliography，编译后他会自成一个独立的章节，如果是 article 类别的文稿，他会自动印出 **Referrences** 的字样为标题，如果是 report 或 book 类别的文稿，他会印出 **Bibliography** 的字样为标题。

在 thebibliography 环境里头，他是由 \bibitem 指令来列出数据的，我们来看一下他的语法：

```
\begin{thebibliography}{99}     %  参考文献印出之编号最宽为两个字母宽
\bibitem[标记一]{键值一} 参考数据一
\bibitem[标记二]{键值二} 参考数据二
 ...
\edn{thebibliography}
```

所谓的「标记」这是选项参数，如果没有的话，则正常引用后会在甲用处使用阿拉伯数字外加方括号来显示；如果有加入的话，引用后会使用所加入的标记来显示。那个「键值」指的就是引用时的关键词，后面所接的「参考数据」就是书籍、论文等信息。其中的 {99} 只表示在最后的参考文献印出来的时候，最开始的编号统一在两个字母宽，如果都没有使用「标记」，那么就是两个数目字宽，如果有使用「标记」，那么要设在最长标记的字母宽，否则印出时会无法对齐。

我们引用的时候是使用 \cite{键值} 这个指令，他会显示参考数据中的编号值，且以方括号括起来。我们来看看一些设定及引用的例子，：

```
\begin{thebibliography}{KDE}      %  参考文献中印出的编号最宽为三个字母宽
\bibitem{KDEt} Knuth, D.E., \textit{The \TeX{}book},
Reading, Massachusetts: Addison-Wesley, 1989.
\bibitem[KDE]{KDEm} Knuth, D.E., \textit{The \MF{}book},
Reading, Massachusetts: Addison-Wesley, 1986.
 ...
\end{thebibliography}
```

参考文献印出的结果请参考本文件后面关于参考数据的部份。至于引用方式及其引用情形如下（颜色的部份是因为使用 hyperref 套件的超链接）：

| 引用方式 | 引用结果 |
|---|---|
| 请参考 \cite{KDEt} | 请参考 [1] |
| 请参考 \cite[1989]{KDEt} | 请参考 [1, 1989] |
| 请参考 \cite{KDEm} | 请参考 [KDE] |
| 请参考 \cite[1986]{KDEm} | 请参考 [KDE, 1986] |
| 请参考 \cite{KDEt,KDEm} | 请参考 [1, KDE] |

如果你现在是在观看 PDF 格式的档案，那在 Xpdf 或 Adobe Acrobat Reader 都可以使用鼠标来点一下，看看真正在后面印出时是什么样子。再次强调，参考文献的规格请依邀稿单位的要求，这里所列出来的不是「标准」。

### 11.4.2 更改标题名称

前面已提到过更改目录及索引的标题，同样的方法，我们也可以更改参考资料的标题，只是要注意引用的文稿类别是什么。

```
\renewcommand\refname{参~考~资~料}      %  article  类别文稿
\renewcommand\bibname{参~考~文~献}      % report/book 类别文稿
```

### 11.4.3 BibTEX 简介

如果常常有写论文的机会，整理出自己的一份参考文献数据库可以节省许多时间，正常情 况下，使用 bibtex 来处理外部文献档案的情形，只有引用到的文献才会印出来，这样也就不必担心印出一堆不相关的文献了。另外一个好处是，这个参考文献数据库可以另外独 立维护，所有的文章都用这一份数据库，这在维护上会很方便，也减少错误的机会。

BibTEX 本身提供一个外部的 bibtex 工具程序，在 latex 编译过文稿后，再利用 bibtex 编译一次文稿，最后再使用 latex 重编译过。而参考文献数据库是按一定的格式写于 *.bib 档案里头，在文稿中则以 \bibliogrphy 指令来引入，编译过程中自然会去参考这个外部考文献数据库。他的使用情形如下（以 your.bib 为例）：

```
...
\begin{document}
\bibliogrphiystyle{plain}        % 指 定 style 檔
...
\bibliography{your.bib}          % *.bib 延伸檔名可以省略
...
\end{document}
```

编 译 过 程 ：
latex example
bibtex example
latex example

#### 11.4.3.1 *.bib 檔的格式

*.bib 档的格式自成一格，和写在原来文稿里头的不同，视数据的性质，要把他标明出来，例如书籍类是 @book 来开头，期刊文章使用 @article 来起头，我们来看一个例子：

```
@book{ KDEt,
   author       = "Knuth, Donald E.",
   year         = "1989",
   title        = "The {\TeX}book",
   publisher    = "Addison-Wesley",
   address      = "Reading, Massachusetts",
   volumn       = " ",
   edition      = " ",
   month        = " ",
   series       = " ",
   note         = " ",
}
@article{ somekey,
```

```
    author      = "Someone",
    year        = "2004",
    title       = "The {\TeX} Journal",
    journal     = "SayYa-Publisher",
    volumn      = " ",
    number      = " ",
    pages       = " ",
    month       = " ",
    note        = " ",
}
```

每行后的逗点是必要的，名字的话 Knuth, Donald E. 或 Donald E. Knuth 这两种方式，bibtex 都能认得，但姓摆在前面的时候其后要加个逗点，如果是两位以上的作者时要以 and 来连接。虽然可以使用 LaTeX 的语法，这时他整个要由大括号括起来，而且，批注符号 % 不被接受。红色的项目是必要的项目，其他项目可以列进去，也可以省略，要加进去的话，则以 " " 空出来，这样以后有这方面的数据时再填进去。

显现的形式是受 *.bst 格式档在控制的，所以，不必要的标点符号不要自行加进去，书名的字体显示也无需加进去。

引用的方式同样是使用 \cite 指令，一般只要要引用到的数据才会印出来，如果要全部 *.bib 里头的数据都印出来的话，可以加个 \nocite{*} 指令。

在使用中文的情形下，bibtex 程序认不得中文，在 CJK 环境下编译会出问题，我们可以先编辑一个 *.cbib 档，然后再使用 bg5conv 来把他转成 *.bib 檔：

```
bg5conv < some.cbib > some.bib
```

这样，在文稿里头引入 some.bib 就可以了。

#### 11.4.3.2 格式檔

BibTeX 的格式檔是 *.bst（bibliography style），我们上面所引用的是 plain 其实就是引用 plain.bst 这个格式档，这是最基本的格式，在编译时期会依这个格式档来印出参考文献的显现形式。其他尚有：

| | |
|---|---|
| plain | 依字母的顺序印出，比较顺序为 author, year, title |
| unsrt | 依引用的先后次序印出 |
| abbrv | 与 plain 相同，但 first name, month, title, journal 以缩写印出 |
| alpha | 引用处显示 [作者年份] 来取代数目字。 |

已经有许多人发表过特定的格式档，但这些对于中文则无法完全合乎我们的使用习惯，例如标点符号及书名号，但我们可以去更改他们的格式，这方面的数据请参考系统上的 btxdoc.dvi 及 btxhak.dvi 这两个说明档。

这也算是目前的一个值得去研究的空间，尤其是中文及 Unciode 编码文件索引、排序及排版的问题，这在英文语系算是比较容易解决，都有现成的格式范例可以运用，但中文就比 较缺乏这方面的范例。在吴聪敏教授的《cwTEX 排版系统》[5] 一书里头，有对这方面做过努力，使用的是外部程序工具 cwbibtex，再和 cw.bst 格式档配合的话，有不错的结果。

## 11.5 附录（Appendix）

排版附录使用的是 \appendix 指令，这个指令以后和正常一般编辑即可，不同的附录以 \chapter 来区隔，但印出来的时候会标上大写字母，而不是原来的 **Chapter** 字样。也是可以有 \section 指令，这资了大写英文字母外，会紧接着附上阿拉伯数字，例如 **A.1**、**B.2** 等等。

### 11.5.1 改变附录的标题

在英文环境，附录是以 **Appendix** 为标题来开始的，在中文环境下我们要把他改成中文：

```
\renewcommand\appendixname{附~录}
```

article 类别的文稿并不印出 **Appendix** 字样，因此也就没有 appendixname 来更改。

## 11.6 大型文稿的维护

通常我们写一篇文章，大概都是一个文稿写到底，但如果超过一百页的文稿的时候，维护起来会比较困难，所以 LATEX 提供了 \input 及 \include 指令来将外部档案引进来，当做是文稿的一部份来编译，这样就可以按章节来把文稿分开处理、维护。

他们的使用方法很简单，一个是含主要正常结构的主文件，其他的则没有 preamble 区，只有本文区，就是按照一般文稿中的章节的部份来书写就可以了。以这一篇文件的例子来

说，他的主档是 latex123.tex，主要的内容如下：

```
\documentclass[12pt,a4paper]{report}
 ...
 这里是 preamble 区的内容
 ...
\begin{document}
\begin{CJK}{Bg5}{hwmm}
 ...
%begin{latexonly}
\input{story.cjk}
\input{preparation.cjk}
\input{syntax.cjk}
\input{start.cjk}
\input{space.cjk}
\input{class.cjk}
\input{package.cjk}
\input{table.cjk}
\input{graphic.cjk}
\input{math.cjk}
\input{abook.cjk}
\input{theend.cjk}
\input{fdl.cjk}
%end{latexonly}

\begin{htmlonly}
\input{story}
\input{preparation}
\input{syntax}
\input{start}
\input{space}
\input{class}
\input{package}
\input{table}
\input{graphic}
\input{math}
\input{abook}
\input{theend}
\input{fdl}
\end{htmlonly}
 ...
\end{CJK}
\end{document}
```

这里分成两段来处理是因为要使用 bg5latex 来编译文稿，但 latex2html 本身已经可以处理未经前置处理的 LaTeX 中文文稿，所以分成两个部份来分别处理。\input{} 里头的案名，如果没有加附档名的话，预设他的延伸档名是 *.tex。CJK package 另外提供了

\CJKinput 及 \CJKinclude 两个指令，也可以用来引入中文文稿。

### 11.6.1 input 和 include 的差异

这两个指令都可以用来引进外部文稿，但有一些细节不一样。\input 的情形，他可以另外指定延伸檔名，\include 则不行，他一定要是 *.tex 的延伸檔名。在引入的时候 \include 会起新页，\input 则不一定，要视文稿类别而定。最重要的差异在于 \include 可以在 preamble 区和 \includeonly{} 指令配合，这样就不必每次都要编译整份文稿，只编译新文稿就可以了，页数、参照还是会正确显示，这在排版大型书籍的时候就很好用了。

## 11.7 裁切记号（crop marks）

裁切记号是用在排版完成，送印后会有个标记符号，让印刷厂可以跟据这些标记符号来做截切、剪裁的工作。LaTeX 本身并没有内建这种功能，这得和 crop package 来配合。

由于个人也并没有这方面的实际经验，因此这里不多做说明，请自行参考 crop 宏套件的说明。

# 第 12 章

# 后记

大概简单介绍了 LaTeX 的使用，这样够用了吗？很可能是不够的，尤其是想变动风格的时候，但一般使用，不讲求花俏，应该可以用了，剩下的只是熟练的问题。当然，许多细节可能并没有说明清楚，但方向知道了，其他的自行查阅就行了。真碰到问题时，可以到 bbs/news 上询问。

有许多没有提及的东西，在这一章交代一下，有些是可以自行查阅的，有些则是尚未成熟，可能还不到真正实用的阶段（也可能是我自己也不会用啦！:）。

1. 微调
   由于微调牵涉到对 LaTeX macro 的一定程度的认识，因此并没有说明得很清楚，理想的话应该先把 TeX/LaTeX 宏的写法先做简单的介绍，这个部份可能另外专文介绍较妥，毕竟这篇文件是定位在入门级教材。

2. 中文的处理
   中文的处理还有很多模糊地带，例如，索引、参考文献及中文直排。目前的其他中文 TeX/LaTeX 也没有介绍，例如 ChiTeX、cwTeX 及 PUTeX 等等。
   TeX/LaTeX 系统的字型机制算是较复杂，安装字型更是一般用户的梦魇，英文写作比较容易解决，通常系统上都会安装好，中文的话就比较麻烦，除了详细去介绍外，我们使用中文字型应该有个大家认同的规格才行，否则我这份文件拿到其他的中文 TeX/LaTeX 系统上的时候，就必须修改一下，至少换个字型名称才能顺利编译。

3. 实例嫌不够完整
   尤其是表格、图形处理及数理排版的部份，并没有交待得很完整。这里头当然牵涉 到许多的背景知识的问题，不单纯在排版本身，这在其他的排版系统一样会碰到同 样的情况。当然，有很大的部份是我个人经验不足的关系啦！:-)

4. 各种文件格式的介绍

   TEX/LATEX 系统中的文件格式多如牛毛，包括一些中间产生的档案都有他特定的目的，但我们并没有多做介绍。原因是，这些档案都牵涉到他的运作机制，这么一来 连运作机制也要说明才行，这样会使篇幅大增，而且也会扰乱了初学者的学习步调，因此，只能留待往后有机会再来介绍。

5. 现有宏库的整理

   LATEX 的宏实在是太多了，现有资料大多是英文的，是有必要整理出一份有系统的中文速查表，以免重复去制造轮子。

6. 重音符号、欧洲字符

   这些都没有真正接触到。这些内容，个人不敢造次，因为并不很熟悉，因此，得要 有懂欧洲语系的朋友来个完整的介绍才行。

7. Unicode 编码文件的处理

   这方面也没介绍，但由于这是一篇入门级的 LATEX 教学文件，这方面的内容应该是由另外的专文来介绍可能会比较恰当。

8. 投影片的制作

   投影片、幻灯片的议题目前很流行，LATEX 也是可以制作精美的投影片，这方面的文件可以参考：

   http://www.miwie.org/presentations/presentations.html

   当然，以上的数据是使用在英文语系，中文的话，我们得另做介绍。

9. 和 XML/SGML/HTML 的配合

   这个完全没有提到，一方面 XML/SGML 的内容没有想象中简单，他们的应用范围实在是太广了，这方面的内容得慢慢来补充。

10. 和数据库系统的配合

    这方面在目前 seaching 挂帅的 Internet 是相当重要的课题，这当然是不适合放在这份文件，得另外在进阶的文件做更进一步的介绍。TEX 系统的生命力、可塑性相当强，因此和其他的文件系统的结合能力也就比较容易达成。

11. GUI 图形界面

    这个部份也没有深入介绍，最重要是卡在中文的问题上，这方面需要有兴趣的朋友 共同来研究，虽然命令行环境的生产效率很高，但为了顾及一般使用者的习惯，方 便入门的 GUI 图形界面的确有其需要。

这份文件，感谢行政院研考会委办，朝阳大学洪朝贵授与辅仁大学毛庆祯教授共同主持的「政府机关数据文件交换之电子文件格式应用研究」计划做部份的补助，很高兴他们都能认同自由文件、自由软件，在目前社会体制下的存在价值。

这份文件的 PDF 格式所嵌入的中文字型，采用的是王汉宗博士所捐赠的三十几套的 TTF 向量字体，王博士以前就曾捐赠过十三套的 TTF 向量字体，再加上这次的三十几套，我们自由软件社群的中文字型就更充实了。更难得的是，这些字型都是采用 GNU GPL 的授权，和这份 GNU FDL 自由文件配合起来，相得益彰，感谢王汉宗博士的慷慨无私及对自由软件、自由文件的认同。

这份文件采用的是 PDF 及 HTML 格式，这当然得需要网络资源才能呈现在各位面前，感谢 kenduest 及交大数学和 ctshieh 及淡江数学提供这方面的资源，没有他们的帮忙，文件就无法呈现在各位眼前了！

也希望大家能对这份文件多多的指正及建议。写作期间，已经接到许多朋友的来信指正，很感谢他们。这份文件的 HTML/PDF 格式及原始文稿，可以在以下网站取得：

http://edt1023.sayya.org/tex/latex123/index.html
http://edt1023.sayya.org/tex/latex123/latex123.pdf
http://edt1023.sayya.org/tex/latex123/latex123-v1.0-src.tar.gz
http://MathNet.math.tku.edu.tw/~edt1023/tex/latex123/index.html
http://MathNet.math.tku.edu.tw/~edt1023/tex/latex123/latex123.pdf
http://MathNet.math.tku.edu.tw/~edt1023/tex/latex123/latex123-v1.0-src.tar.gz

第 13 章

# GNU 自由文件许可证原文

## GNU Free Documentation License

Version 1.2, November 2002



### 0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document “free” in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of “copyleft”, which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

# 1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The “Document”, below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as “you”. You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A “Modified Version” of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A “Secondary Section” is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document’s overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The “Invariant Sections” are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The “Cover Texts” are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A “Transparent” copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not “Transparent” is called “Opaque”.

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque

formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The “Title Page” means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, “Title Page” means the text near the most prominent appearance of the work’s title, preceding the beginning of the body of the text.

A section “Entitled XYZ” means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as “Acknowledgements”, “Dedications”, “Endorsements”, or “History”.) To “Preserve the Title” of such a section when you modify the Document means that it remains a section “Entitled XYZ” according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

## 2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

## 3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document’s license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front

cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

## 4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

**A.** Use in the Title Page (and on the covers, if any) a title distinct. from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.

**B.** List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.

**C.** State on the Title page the name of the publisher of the Modified Version, as the publisher.

**D.** Preserve all the copyright notices of the Document.

**E.** Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.

**F.** Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.

**G.** Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.

**H.** Include an unaltered copy of this License.

**I.** Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.

**J.** Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.

**K.** For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.

**L.** Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.

**M.** Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.

**N.** Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.

**O.** Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled “Endorsements”, provided it contains nothing but endorsements of your Modified Version by various parties–for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

## 5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled “History” in the various original documents, forming one section Entitled “History”; likewise combine any sections Entitled “Acknowledgements”, and any sections Entitled “Dedications”. You must delete all sections Entitled “Endorsements”.

## 6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

## 7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an “aggregate” if the copyright resulting from the compilation is not used to limit the legal rights of the compilation’s users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document’s Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

## 8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled “Acknowledgements”, “Dedications”, or “History”, the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

## 9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License.

However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

## 10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See http://www.gnu.org/copyleft/.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

## ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

> Copyright © YEAR YOUR NAME. Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License".

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the "with. . . Texts." line with this:

> with the Invariant Sections being LIST THEIR TITLES, with the Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.

# 參 考 資 料


[1] Knuth, Donald E., *The TEXbook*, Reading, Massachusetts: Addison-Wesley, 1989. 11.4.1

[KDE] Knuth, Donald E., *The METAFONTbook*, Reading, Massachusetts: Addison-Wesley, 1986. 11.4.1

[2] Clark, Malcolm, *TEX by Topic*, Reading, Massachusetts: Addison-Wesley, 1992.

[3] Abrahams, Paul W., Berry, Karl and Hargreaves, Kathryn A., *TEX for the Impatient*, Reading, Massachusetts: Addison-Wesley, 1990.

[4] Goossens, Michel, Mittelbach, Frank, and Samarin, Alexander, *The LATEX Companion*, Reading, Massachusetts: Addison-Wesley, 1993.

[5] 吳聰敏、吳聰慧，《cwTEX 排版系统》，台北：吳聰敏，2002。 11.4.3.2

[6] 郑郁耀、郭雅思，《LATEX 数理排版：由入门到应用》，台北：全华，1995。

[7] 王佑中，《Linux 中文应用手册》，台北：第三波，1995。 9.5.6

[8] 朱浤源，《撰写博硕士论文实战手册》，台北：正中，1999。

[9] 傅祖慧，《科学论文的写作审查及发表》，台北：中华农学会，1980。

# 使用許可證聲明

# 索 引