

L^AT_EX 的快速入門

蔡炎龍
政治大學應用數學系

July 19, 2007

前言

1

1.1 這份文件的目的是

這份文件是希望提供有心想學 $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ 的人，一份快速入門的文件。我心目中的主要讀者是研究生，所以我們會以最快的速度去討論怎麼樣把一篇論文完成，包括 $\text{BibT}_{\text{E}}\text{X}$ 的論文管理。但另一方面來說，我又希望可以更廣泛的讓 $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ 帶入一般文件處理，而不只是在論文上面，所以我會將中文 $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ 一併帶入。我們選用的中文 $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ 是 $\text{CJK-L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ ，在每一種主要平台上都有。

在學 $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ ，尤其是中文 $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ ，常常會有兩個困難點，一個是安裝上面的問題，一個是安裝後的學習。如果我們一談起安裝，可能就沒完沒了，而且討論完大概也沒機會進入正題。所以我們用另外的機會解說安裝的問題，現在假定是已經安裝完成了。如果是在政大，可以到應用數學系電腦室，我們已經設好我們這篇文章討論應有的 $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ 環境。我們需求的環境是：

- 有 $\text{CJK-L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ 的 $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ 環境。
- 可以在 $\text{PDFL}^{\text{A}}\text{T}_{\text{E}}\text{X}$ 下正確使用的 UTF-8 編碼中文字型。

另外，為了順利的使用 $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ ，你應該要有個順手的純文字編輯器。我個人推薦的編輯器如下：

- Texmaker (Unix-like 系統 / Windows, 易上手的 $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ 專用編輯器)
- Notepad++ (Windows)
- Vim, Emaces (Unix-like 系統, 如果屬於 Unix Geek)
- TeXShop (Mac OS X, 事實上我最偏好這一個)

我們只準備使用 $\text{PDFL}^{\text{A}}\text{T}_{\text{E}}\text{X}$ ，也就是說，所有的 $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ 檔，都直接產生 PDF 文件。中文編碼我們只準備使用 UTF-8，這除了是個潮流，也讓英文和中文基本上用的流程是完全一樣，更重要的是我們選擇的簡報程式 Beamer 套件要用 UTF-8 + $\text{PDFL}^{\text{A}}\text{T}_{\text{E}}\text{X}$ 比較相容。

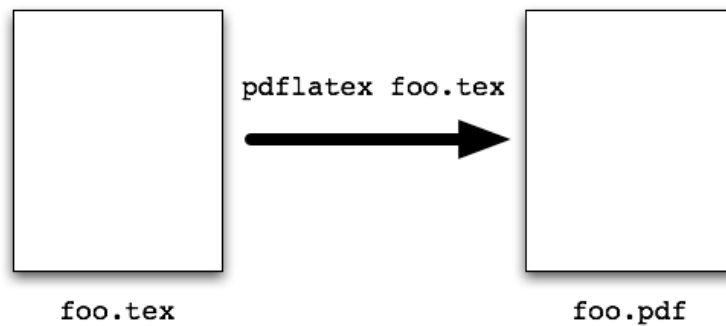
1.2 版本資訊

這份文件是在 2007 年 7 月 16 日完成第一版初稿, 7 月 18 日改用 Xe \LaTeX 進行改版。

L^AT_EX 極速入門

2

很多人說 L^AT_EX 很難，其實 L^AT_EX 實在沒什麼難的。我們只不過是做一個純文字檔，存成 `.tex` 這樣的檔案，然後使用 `pdflatex` 這個指令，馬上就產生一篇高品質的 PDF 文件。



我們這裡很快的來看一下這個 `.tex` 的純文字檔應該長什麼樣子。

2.1 最簡單的 L^AT_EX 文件

最簡單的 L^AT_EX 檔案是長這個樣子。

```
\documentclass{article}
\begin{document}
  內文, 文章的內容
\end{document}
```

可以試打一些內容進去看看，存成 `.tex` 檔，再用 `pdflatex` 編譯。要注意目前還不能用中文。

2.2 完整的 L^AT_EX 格式

一份完整的 L^AT_EX 文件的架構大概如下。

```
\documentclass{article}
設定區, 我們還不會
\begin{document}
\title{文章的標題}
\author{作者}
\maketitle
內文, 文章的內容
\end{document}
```

框起來的部份就是我們需要打字進去的地方。你可以試打一些東西進去, 然後 L^AT_EX 會自動幫你印出標題、作者、有分節的文件。是不是非常容易? L^AT_EX 的一個特性就是, 你可以**專注在文章的內容上, 要美化什麼的可以最後慢慢調**。

2.3 L^AT_EX 的數學符號

很多人聽說 L^AT_EX, 都是聽說它對數學符號處理功力很強。我們來看看要怎麼打入數學符號。L^AT_EX 有兩種數學模式, 分別是:

- 隨文模式 (inline mode)
- 展示模式 (display mode)

我們來看看怎麼樣使用。

2.3.1 隨文模式

所謂隨文模式就是數學式子要插在文中, 使用的方式是把數學式子放入兩個 \$ 的符號中。比方說下面這個例子:

The formula $f(x) = x^3 - 2x + 6$ is important in this case.

The formula $f(x) = x^3 - 2x + 6$ is important in this case.

2.4 展示模式

所謂展示模式的數學式子，是把數學式**獨立、置中**表示。展示模式有很多下指令的方式，我們可以把數學式子用 “`$$. . . $$`”，“`\[. . . \]`”，或 “`\begin{equation} . . . \end{equation}`” 等方式表示，比方說

The formula

$$f(x) = x^3 - 2x + 6$$

is important in this case.

The formula
`\[`
 $f(x) = x^3 - 2x + 6$
`\]`
 is important in this case.

CJK-L^AT_EX 的快 速入門

3

我們使用的環境，限定在用 UTF-8 編碼的 CJK-L^AT_EX 並用 PDFL^AT_EX 編譯。這樣的方式可以讓中文英文 L^AT_EX 使用起來幾乎完全一樣，而且做中文簡報也不會出問題。

我們來看 CJK-L^AT_EX 文件的基本架構。

3.1 CJK-L^AT_EX 基本使用法

我們可以看基本的 CJK-L^AT_EX，和原本 L^AT_EX 幾乎是一樣的。

```
\documentclass{article}
\usepackage{CJKutf8} % 舊版可能要用 CJK
\begin{document}
\begin{CJK}{UTF8}{cwmu} % 把 cwmu 換成要用的字型
內文，文章的內容
\end{CJK}
\end{document}
```

我們可以發現，除了框起來的部份，其他和英文 L^AT_EX 其實是一樣的。使用 UTF-8 編碼和 CJK-L^AT_EX 有很多好處，其中之一是使用起來和英文的 L^AT_EX 幾乎是一樣的 - 連編譯方式也一樣，不需像使用 Big5 編碼或其他一些中文 L^AT_EX 要有轉檔的動作。

3.2 cwT_EX 的五套字型

政大應數系電腦室採用台大吳聰敏教授的 cwT_EX 字型，由李果正先生轉成可供 CJK-L^AT_EX 使用的版本。這裡五套字型名稱如下：

- cwm 明體
- cwk 楷書
- cwhb 黑體
- cwy 圓體
- cwsu 仿宋體

要注意的是在完整安裝 cwTeX 字型的系統下，用 Big5 和 UTF-8 使用的字型名稱是不一樣的，我們是採用 UTF-8 版。事實上，李果正先生的新版 cwTeX for CJK 字型已經改成不管什麼編碼都用一樣名稱，請查明你使用的系統，依相對的名稱改變。

使用 AMS-L^AT_EX

4

4.1 引入 AMS-L^AT_EX

AMS 美國數學學會的 L^AT_EX 套件已然成為一種標準。通常會用到的有三個套件：

- **amssymb**: 提供一些原本 L^AT_EX 沒有的符號，比方說 \mathbb{R} , \mathbb{C} , 等等。
- **amsmath**: 提供一些好用的環境，比方說 `align` 環境等等。
- **amsthm**: 提供比較好的使用定理、定義等的環境。

如果使用一般的 `article class`，建議每次都把三個套件讀進來：

```
\usepackage{amssymb, amsmath, amsthm}
```

4.2 使用 AMS Article Class

使一個使用 AMS-L^AT_EX 的方式是使用 AMS 提供的個文章類型，叫 AMS Article。要使用就是設定使用 `amsart`：

```
\documentclass{amsart}
```

它會自動讀入 `amsmath`, `amsthm` 兩個套件，和部份 `amssymb` 套件（比方說有 \mathbb{R} ）。如果需要全套的 `amssymb`，還是要自行讀入：

```
\usepackage{amssymb}
```

定理環境的使用

5

我們寫數學文章，總會出現定義，定理，證明等等。我們在 $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ 要處理這些東西是很容易的。

5.1 基本定理環境

在設定區設定：

```
\newtheorem{thm}{Theorem}
```

意思是我們要先建一個新的定理環境，叫做 `thm`，顯示時標示為 “Theorem”。比方說：

Theorem 1. *The statements of the theorem.*

```
\begin{thm}
The statements of the theorem.
\end{thm}
```

5.2 定理的編號

你可以發現定理的編號會自動從 1, 2, 3, 等等編下去。但是有時我們要依節次來標，比方說第一節的第一個定理的編號是 1.1，然後 1.2, 1.3, 這樣下去，要怎麼做呢？很容易，加個 `section` 參數就好。比如說在設定時我們設：

```
\newtheorem{thm}{Theorem}[section]
```

那麼在本節（第 5 節）使用定理環境會變成下面這個樣子。

Theorem 5.1. *The statements of the theorem.*

```
\begin{thm}
The statements of the theorem.
\end{thm}
```

我們如果定了兩個定理環境，他們原本是互不相干的，所以會各自編號。比如說如果我們有設了 `thm`, `lem` 兩個定理環境：

```
\newtheorem{thm}{Theorem}
\newtheorem{lem}{Lemma}
```

引用起來會是像這樣子：

Lemma 1. *The statements of the lemma.*

Theorem 1. *The statements of the theorem.*

```
\begin{lem}
The statements of the lemma.
\end{lem}
\begin{thm}
The statements of the theorem.
\end{thm}
```

這樣子的編號方式，我們無法知道 Lemma 7 和 Theorem 3 倒底是哪一個先出現，哪一個後出現。要用統一的編號。比方說引理 1 之後是定理 2 等等，就要用下面的方式宣告定理環境。

```
\newtheorem{thm}{Theorem}
\newtheorem{lem}[thm]{Lemma}
```

請比較和以前有什麼不一樣。

Lemma 1. *The statements of the lemma.*

```
\begin{lem}
The statements of the lemma.
```

Theorem 2. *The statements of the theorem.*

```
\end{lem}
\begin{thm}
The statements of the theorem.
\end{thm}
```

最後, 如果需要完全沒有編號的定理環境, 就要像下面這樣加上星號。

```
\newtheorem*{mainthm}{Main Theorem}
```

Main Theorem. *The statements of the theorem.*

```
\begin{mainthm}
The statements of the theorem.
\end{mainthm}
```

5.3 不同定理風格 (amsthm)

使用 `amsthm`, 可以指定三種不同的定理風格: `plain`, `definition`, 和 `remark`。使用方式是在定義定理環境之前, 先下達:

```
\theoremstyle{定理風格}
```

舉例來說, 假設我們定義下列的定理環境:

```
\theoremstyle{plain}
\newtheorem{thm}{Theorem}

\theoremstyle{definition}
\newtheorem{defn}{Definition}

\theoremstyle{remark}
\newtheorem{rmk}{Remark}
```

會產生如下的效果。

Theorem 1. *The statements of the theorem.*

Definition 1. The statements of the definition.

Remark 1. The statements of the remark.

```
\begin{thm}
The statements of the theorem.
\end{thm}
\begin{defn}
The statements of the definition.
\end{defn}
\begin{rmk}
The statements of the remark.
\end{rmk}
```

5.4 定理的引用

我們會引用到的定理, 就用 `\label{引用代碼}` 來標記。比方說

Theorem 2. *The statements of the theorem.*

```
\begin{thm}\label{T:major}
The statements of the theorem.
\end{thm}
```

要引用的時候, 就是用 `~\ref{引用代碼}`:

Please refer to Theorem 2.

```
Please refer to Theorem~\ref{T:major}.
```

插入圖片

6

6.1 插入圖片的基本方法

這裡建議使用 `graphicx` 套件：

```
\usepackage{graphicx}
```

假設我們要插入 `pic.png` 這個圖檔，使用

```
\includegraphics(width=5cm){pic.png}
```

即可。自然，`width` 是可依你需要設定的。建議使用的圖檔格式為：

```
.png, .pdf, .jpg
```

6.2 圖片置中

使用 `center` 環境即可：

```
\begin{center}  
\includegraphics(width=5cm){pic.png}  
\end{center}
```

6.3 `figure` 的使用方法

上面的圖形基本上是在那插入，就會放在那裡。但正式排版中，常會依版面情況調整位置，且會有提示文字。這時要使用 `figure` 環境。一般要置中，

又有說明的圖會這樣引用：

```
\begin{figure}
\begin{center}
\includegraphics(width=圖形寬度){檔案名稱}
\end{center}
\caption{圖形的文字說明}
\end{figure}
```

\LaTeX 會幫你把圖形放在它認為合適的地方。如果你對放的位置很有意見，可以加入 **h**, **t**, **b**, 或 **p** 等參數改變。比方說使用

```
\begin{figure}(h) ...
```

這些參數代表你希望放置的位置分別是：

- **h**: 放在此處
- **t**: 放在頂端
- **b**: 放在底端
- **p**: 在本頁

事實上你也可以同時用 **[htbp]**，這是告訴 \LaTeX 你希望放在這一頁，但到底怎麼放讓 \LaTeX 自己「看著辦」。

6.4 圖形的引用

圖形的引用其實和定理引用一樣。你只要在想引用的圖提示文字加上 `\label`，比方說：

```
\caption{圖形的提示文字}\label{引用代碼}
```

要引用時則如下範例：

```
參考圖~\ref{引用代碼}...
```

就可以了。

列表

7

我們這裡介紹怎麼樣在 \LaTeX 使用文書處理常用的列表。

7.1 基本列表

要分點列表的基本方式如下：

- 第一項
- 第二項
- 第三項

```
\begin{itemize}
\item 第一項
\item 第二項
\item 第三項
\end{itemize}
```

7.2 數字列表

我們再看要以 1, 2, 3 等標示的列表怎麼做。

1. 第一項
2. 第二項
3. 第三項

```
\begin{enumerate}
\item 第一項
\item 第二項
\item 第三項
\end{enumerate}
```

7.3 定義型列表

第三種定義型列表使用方式如下。

第一項 第一項的說明

第二項 第二項的說明

第三項 第三項的說明

```
\begin{description}
\item〔第一項〕 第一項的說明
\item〔第二項〕 第二項的說明
\item〔第三項〕 第三項的說明
\end{description}
```

陣列和表格

8

8.1 陣列的使用

陣列就是如同矩陣型的排列。我們可以看下一個例子。

```
1 2 3
4 5 6
7 8 9
```

```
\{
\begin{array}{ccc}
1 & 2 & 3 \\
4 & 5 & 6 \\
7 & 8 & 9
\end{array}
\}
```

這裡要說明一下。

```
\begin{array}{ccc}
```

是表示要用陣列，這個陣列有三行，每一行都要對齊中間 (c)。對齊的方式有三種選擇：

- c: 對齊中間
- l: 對齊左邊
- r: 對齊右邊

我們要一一列輸入，要換行時用 “\\” 換行，每一欄用 “&” 隔開。

8.2 表格的使用方式

表格的使用方式非常接近陣列的使用。

item 1	item 2	item 3
1	2	3
4	5	6

```
\begin{tabular}{ccc}
item 1 & item 2 & item 3 \\
1 & 2 & 3 \\
4 & 5 & 6 \\
\end{tabular}
```

如果要加橫線, 加入 `\hline`:

item 1	item 2	item 3
1	2	3
4	5	6

```
\begin{tabular}{ccc} \hline
item 1 & item 2 & item 3 \\ \hline
1 & 2 & 3 \\ \hline
4 & 5 & 6 \\ \hline
\end{tabular}
```

加直線更方便, 在對齊設定那加就可以了:

item 1	item 2	item 3
1	2	3
4	5	6

```
\begin{tabular}{|c|c|c|} \hline
item 1 & item 2 & item 3 \\ \hline
1 & 2 & 3 \\ \hline
4 & 5 & 6 \\ \hline
\end{tabular}
```

8.3 一般的括號和會變大的括號

在 \LaTeX 裡, 要打出小括號到大括號方法如下:

- 小括號: ()

- 中括號: []
- 大括號: \{ \}

問題是如果你想打一個矩陣, 配合 `array` 使用, 會出現一個好笑的結果:

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$$

```
\{
(\begin{array}{ccc}
1 & 2 & 3 \\
4 & 5 & 6 \\
7 & 8 & 9
\end{array})
\}
```

要改正這個缺點, 我們要用「可自調大小的括號」。方式很簡單, 在左邊的括號前加 `\left`, 右邊加 `\right` 就可以。比方說:

$$\left(\begin{array}{ccc} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{array} \right)$$

```
\{
\left(\begin{array}{ccc}
1 & 2 & 3 \\
4 & 5 & 6 \\
7 & 8 & 9
\end{array}\right)
\}
```

8.4 陣列和括號的應用

注意前面「會變大小的括號」是一定要成對出現的。如果我們已經用了 `\left`, 一定要有 `\right`。不過, 我們其實可以只要一邊, 比方說左邊的括號, 而右邊可用 `\right.` 表示不要顯示任何括號。我們來看一個應用:

$$|x| = \begin{cases} x, & \text{if } x \geq 0 \\ -x, & \text{if } x < 0 \end{cases}$$

```
\{  
|x| = \left{  
\begin{array}{rr}  
x, & \mbox{if } $x \geq 0$ \\  
-x, & \mbox{if } $x < 0$ \\  
\end{array} \right.  
\}
```

這裡我們新學一件事，那就是**如果數學式中我們要打入一些純文字，可以用 `\mbox` 指令**。而在 `\mbox` 裡面的是純文字，所以再打數學符號就要加上錢的符號了。

BibTeX 入門

9

9.1 為什麼要用 BibTeX?

使用 BibTeX 並不是在 LaTeX 下引用論文唯一方式。開始的時候，用 BibTeX 可能還會覺得比較麻煩，因為你為了論文，還要建立一個 `.bib` 純文字檔，內容是你要引用的論文資訊。我們為什麼要這麼麻煩呢？把要引用的論文全寫在本文那個 LaTeX 檔不是很好嗎？這其實不是那麼好。

我們常會碰到這個情況：你找了一堆論文，其實你也不知道哪篇對你有用，所以暫時沒有放入你文章後面的參考資料清單中。有一天，你發現某某篇有用，結果一時之間找不到那篇在哪裡！就算你真的把論文弄好，每次還要手動排序！萬一有一天你發現你論文格式用的不合教授/期刊的要求，你還得重新修正！

如果用 BibTeX，你完全不用擔心這件事！你覺得有可能參考的，你就把它建檔進去，**BibTeX 只會列出你真的有引用的文章，而且幫你排序**，你也可以**隨時指定，更換論文排列和引用樣式**。更方便的是，如果你下一篇文章也是同領域，使用 BibTeX 可以再用完全一樣的 `.bib` 檔。你也可以為了新的文章加入新的論文，但是不會影響原來文章的編譯。更棒的事可能是，你可以和大約同領域的教授，同學共同分享 `.bib` 檔，這樣大家都可以省些力氣！

9.2 BibTeX 的檔案內容

BibTeX 的檔案是一個純文字，以 `.bib` 為副檔名的檔案，內容就是紀錄每一篇你有興趣文章的資訊。我們看一個例子會更加明白：

```
@article{tx07,  
  Author = {Tsai, Yen-lung and Xia, Eugene Z.},  
  Journal = {Proc. Amer. Math. Soc.},  
  Volume = {135},  
  Number = {8},  
  Pages = {2365-2367}}
```

```
Title = {Non-abelian local invariant cycles},
Year = {2007}}
```

這裡面的內容我們不一定要這樣自己慢慢照這樣打入，所以我們不詳細說明。

9.3 文章的引用

最要注意的是上面的例子中一開始的 “`tx07`” 是我們要引用這篇文章的引用代碼。你在本文中要引用的地方，請打入

```
~\cite{tx07}
```

就可以了。這裡要討論這個引用代碼的編法。很多人照 L^AT_EX 之父 Leslie Lamport 的範例，使用 作者:代碼 做為引用方式。比方說我們要引用 Wilfried Schmid 在 1973 年的 “Variation of Hodge structure: the singularities of the period mapping” 可能會用

```
schmid:vhs
```

當做引用代碼。這樣的方式，我在實做時發現相當困擾，因為有時很難想到一個好的代碼，有時弄了一個代碼，要引用時忘了自己原來的代碼，還要回頭去查。因此，像不少 L^AT_EX 使用者的建議一樣，我會建議直接用 {作者 + 年份} 做為引用代碼。比方說 Schmid 的文章，我們就用：

```
schmid73
```

這樣引用。你在研究的過程，你很容易記下來某某人在某某年做了什麼，所以這樣引用其實更合理方便。當然，這只是提供參考，你可以找出自己合適的方式。

這裡還有個問題，就是你怎麼讓 L^AT_EX 知道你要使用的 BibTeX 檔在哪裡？假定我們把我們 BibTeX 檔存成 `reference.bib`，那麼使用方式就是在 `\end{document}` 之前加入兩行：


```
\bibliographystyle{plain}    % 使用 plain 格式, 可換其他格式
\bibliography{reference}     % 使用 reference.bib
```

就可以了。

9.4 BibTeX 的編譯

使用 BibTeX 的 L^AT_EX 文件, 編譯過程有時有點讓人困惑。我們這裡假設以 `foo.tex` 為我們的 L^AT_EX 檔 (BibTeX 檔叫什麼無妨, 只要我們在文中引用正確的 `.bib` 檔就可以), 我們要產生正確引用的 PDF 檔要用

```
pdflatex foo.tex
bibtex foo.tex
pdflatex foo.tex
pdflatex foo.tex
```

第一次的 `pdflatex`, 你的 L^AT_EX 系統「看到」你要引用的文章, 可是它根本沒有資訊。所以再叫 BibTeX 把需要文章的資訊都抄下來。第二次/第三次 `pdflatex` 就是依得到的文章資訊, 可能排序, 決定出現的篇號, 再填入引用的地方。

9.5 如何建立 BibTeX 檔?

其實我們不需要自己打入那些 BibTeX 的資料, 有很多更簡單的方式!

9.5.1 使用 MathSciNet

MathSciNet 是美國數學學會 (AMS) 推出的線上論文查詢系統。幾乎所有重要數學期刊的文章都可以查到, 所以你可以確定某些文章是不是登出來, 在哪登出來的。MathSciNet 可以把找到的文章直接顯示成 BibTeX 格式, 所以你只要 `copy` 和 `paste` 就可以轉貼到你的 `.bib` 檔裡面, 完全不用自己打字!

9.5.2 使用 BibTeX 輔助軟體

BibTeX 輔助軟體可以有比較親切的界面, 讓你方便輸入文章資訊。更重要的是, 如果你的電腦裡有那篇文章, 你可以做一個連結。有一天你想看看這篇文章, 點個兩下就可以打開, 所以你再也不用擔心找不到那篇文章! 我個人推薦兩套在不同平台上免費 BibTeX 輔助軟體:

JabRef (Windows, Unix-like 或其他可跑 Java 的系統)

BibDesk (Mac OS X)