

@ 算法时空

# 目录

摘要	I
ABSTRACT	II
第1章 结构	1
1.1 节	1
1.1.1 小节	1
1.2 文件	1
第2章 文字	2
2.1 举例	2
2.2 代码抄录	3
第3章 公式	4
3.1 初学 $\text{\LaTeX}$ 常见的数学符号误用	4
第4章 图片	5
第5章 表格	7
第6章 引用	8
第7章 版式	9
第8章 辅助	10
8.1 MathJax	10
8.2 Microsoft Math Solver	10
8.3 表格自动转换	10
8.4 PDF代换法	10
第9章 进阶	11
9.1 Mac篇	11
致谢	12
参考文献	13

## 摘要

简单明了.

关键词: X; Y; Z

# ABSTRACT

Clear & Simple.

**Keywords:** X; Y; Z

# 第1章 结构

## 1.1 节

### 1.1.1 小节

## 1.2 文件

要将 $\text{\LaTeX}$ 作为数学公式排版以及整合工具来用,也就是其他格式的文件尽量单独制作,最终利用各种`include`命令将其整合到文档之中,这样能够极大地提升工作效率.

## 第2章 文字

### 2.1 举例

书写, 用心.  
段之间以空行分隔.

## 2.2 代码抄录

可以直接列出代码(设定缩进距离为2, 每行编号并从0开始):

```
0  for (int i = 0; i < n; ++i)
1    for (int j = 0; j < n; ++j)
2      cout << i * j << ' ';
```

也可以使用`VerbatimInput`命令直接包含代码文件.

代码变量是`i`, 复杂的是`int i = 0`.

## 第3章 公式

最常见的是行内公式, 例如集合  $X$  中的元素  $a$ , 初学者容易直接使用普通符号而不是数学模式. 另外一个问题是斜体, 最好使用斜体方式.

行间公式一般带编号, 我们可使用 `label` 引用, 例如公式 3.1 给出了渐近记号.

$$\log(n!) = \Theta(n \log n) \quad (3.1)$$

注意写一段就编译一段, 这样容易查错, 特别是公式.

### 3.1 初学 L<sup>A</sup>T<sub>E</sub>X 常见的数学符号误用

我们给出几个典型例子(括号内附有代码).

- 是  $\log x$  (`\log{x}`) 而不是  $\log x$  (`\log x`);
- 是  $\min$  (`\min`) 和  $\max$  (`\max`) 而不是  $\min$  (`min`) 和  $\max$  (`max`);
- 是  $\Pr$  (`\Pr`) 而不是  $Pr$  (`Pr`);
- 是  $\sin x$  (`\sin{x}`) 和  $\cos x$  (`\cos{x}`) 而不是  $\sin x$  (`\sin x`) 和  $\cos x$  (`\cos{x}`);
- 是  $x \times y$  (`x \times y`) 而不是  $x * y$  (`x * y`);
- 是  $\arg$  (`\arg`) 而不是  $arg$  (`arg`);



## 第4章 图片

一般采用浮动的图，以选项 `[!htbp]` 标记，如图4.1所示，注意其中还有两个子图4.1(a)和4.1(b)。



图 4.1 主图

如果有特殊的需要，可以用固定位置的图片，在后面加上 `[h]` 选项即可，例如图4.2。



图 4.2 算法三部曲

当然也不一定能够完全在当前位置，可能当前位置不够会挤到下一页。

不要用屏幕截图，实验结果可以用软件的导出图，<sup>1</sup> 数据可以导出文本利用表格或者抄录形式。

<sup>1</sup>得用PNG格式，其他的展示型图像可以用JPG格式。

有时候我们会使用Excel的图表,可以将其复制到一个空的工作表再存为PDF格式,随后将PDF文件包含到图中即可.

# 第5章 表格

一般采用浮动的表格, 以选项[!htbp]标记, 注意表格不能写“如下所示” 要写“如5.1所示”. 表5.1使用了单元格不同位置的标记(例如c/1/r标记).

	Books	
1	Introduction to Algorithms	3
2	The Algorithm Design Manual	2
3	Algorithms	4

表 5.1 算法三部曲

表5.2下面还有子表5.2(a)和5.2(b), 注意命名不同.

0	1	2	3
4	5	6	7

000	001	010	011
100	101	110	111

(a) 子表(十进制) (b) 子表(二进制)

表 5.2 主表

最好要把每一页填满, 这样排版问题会少很多(例如调节图片的大小).



图 5.1 算法三部曲

## 第6章 引用

最常见的引用是参考文献的引用. 例如参考文献[1]的标记为SX(也即`\bibitem{SX}`), 于是我们可以使用`\cite{SX}`实现引用参考文献[1]的效果, 而这本书便是使用 $\text{\LaTeX}$ 排版.

引用需要排次序, 所以需要两次编译, 第1次识别所有的标记并编号, 第2次将编号填入并显示.

其他引用一般使用`label`给出标记, 再用`ref`命令引用. 例如以`\label{eq:log_factorial}`给出公式标记`eq:log_factorial`, 再用`\ref{eq:log_factorial}`引用得到3.1. 为了区别不同的标记, 可用前缀配合冒号区分: 公式的前缀`eq`, 图的前缀`fig`, 表的前缀`tab`. 列举如下:

- 公式标记为`eq:log_factorial`, 也即公式3.1.
- 图标记为`fig:books`, 也即图4.2.
- 表标记为`tab:trilogy`, 也即表5.1.

## 第7章 版式

我们在`NCL.def`引入的版式设计非常少, 如果需要调整可改动该文件, 例如更改页眉的九章论文, 或者公式或者图表的编号格式.

如果每章首页不需要任何页眉页脚, 可使用我们定义的`echapter`命令. 当然还有其他方法: <https://tex.stackexchange.com/questions/19738/why-doesnt-pagestyleempty-work-on-the-first-page-of-a-chapter>.

另外, `thesis.tex`和`abstract.tex`也有少量排版设定, 可酌情增减.

## 第8章 辅助

如今有很多辅助工具可以帮助我们更好地完成 $\text{\LaTeX}$ 文档.

### 8.1 MathJax

有些“所见即所得”Markdown编辑器能很好地配合MathJax显示 $\text{\LaTeX}$ 公式. 平时可以先用此类软件写一些片段, 若能正确展示, 再复制到 $\text{\TeX}$ 文件中, 这样文本编写效率会更高. 例如Mark Text(<https://github.com/marktext/marktext>)或Typora(<https://typora.io>).

### 8.2 Microsoft Math Solver

Microsoft Math Solver(“微软数学”)这款APP的主要功能是求解数学问题, 但是我们可以用来处理复杂的公式, 在平板上手写识别后可以复制 $\text{\LaTeX}$ 源代码.

### 8.3 表格自动转换

对于数据量较大的表格, 可以寻找转换工具(也有很多在线版本), 会节约很多时间.  
另外, 实验数据最好由编程语言提供的软件包转换.

### 8.4 PDF代换法

如果使用其他软件绘制的图表, 可以转换成PDF文件再以图片形式统一包含, 这样可以节约不少时间. 例如:

```
\begin{table}[h]
  \centering
  \includegraphics[width=.8\textwidth]{table.pdf}
  \caption{形为表格实为图片}
\end{table}
```

需要注意的是, PDF需要裁剪成合适的尺寸(macOS可以使用自带的Preview完成). 对于Word而言, 可以为指定的表格设置合适的纸张大小以及边距, 这样导出的PDF文件可直接使用.  $\text{\TeX}$  Live还提供了pdfcrop工具(命令行方式)来裁白边, 注意MacTeX将其安装在/Library/TeX/texbin目录下.

## 第9章 进阶

### 9.1 Mac篇

我们给出一些设定方法, 在某些特定需求下可供选用.

#### 小技巧

- 使用TeXShop时清理aux文件有时候没法处理子目录, 特别是使用了`\include`命令时. 在终端输入`defaults write TeXShop AggressiveTrashAUX YES`即可.

#### iCloud同步问题

如果要在iCloud上试一下 $\text{\LaTeX}$ 项目同步, 但是每次编译产生的中间文件很多, 会频繁地引起不必要的数据读写, 影响效率. 为了避免这个问题, 在Mac上找到我用的XeLaTeX命令, 也即文件:

```
~/Library/TeXShop/Engines/XeLaTeX.engine
```

复制到当前文件夹改名`CLOUD.engine`, 以后编译使用可在`typeset`里选择这个`CLOUD`执行. 我们想把这些中间文件放到`/Users/x/TMP`下, 这样即可在本地处理. 如果项目有多个文件夹且用`\include`命令包含文件, 暂时文件夹里得仿照文件结构新建若干空文件夹, 否则会报错.

`CLOUD.engine`应改为:

```
#!/bin/tcsh

set path= ($path /Library/TeX/texbin /usr/texbin /usr/local/bin)
xelatex -output-directory=/Users/x/TMP -file-line-error -synctex=1 "$1"
open -a TeXShop /Users/x/TMP/`echo $1 | sed 's/\(.*\)\\..*/\1\\.pdf/'`
```

上述方案来自<https://tex.stackexchange.com/questions/67211/use-texshop-preview-window-when-different-output-dir-is-set>, 不过他使用的是`bash`而不是XeLaTeX原先使用的`tcsh`, 因此代码中会用到

```
open -a TeXShop /Users/x/TMP/$(echo $1 | sed 's/\(.*\)\\..*/\1\\.pdf/')
```

当然也可以使用`latexmk`解决方案.

不过从整体上来说, iCloud方案还是有点麻烦, 不太建议使用. 另外, 似乎预览时更新不太同步, 暂时没去思考解决方案.

感觉每次清理完所有中间文件再同步到GitHub私有仓库会更好一些, 但是数据安全问题需要注意.

## 致谢

感谢Donald E. Knuth和Leslie Lamport, 感谢 $\text{T}_\text{E}\text{X}$ 和 $\text{L}^\text{A}\text{T}_\text{E}\text{X}$ .



## 参考文献

- [1] Steven S. Skiena. 算法设计指南 (第2版) [M]. 谢颢, 译. 北京: 清华大学出版社, 2017.