

# Latex 速查手册

杨浩哲

2019 年 10 月 7 日

SAIUST

## 目录

<b>第一章 安装与配置方案</b>	<b>9</b>
1.1 各平台安装方法最佳实践 . . . . .	9
1.1.1 Windows 平台 . . . . .	9
1.2 中文支持 . . . . .	11
1.2.1 Linux 平台 . . . . .	12
1.2.2 MAC 平台 . . . . .	12
1.3 常用包 . . . . .	12
1.4 L <sup>A</sup> T <sub>E</sub> X 内置命令 . . . . .	14
1.4.1 查看帮助文档 . . . . .	14
1.4.2 其他命令 . . . . .	14
1.5 快速入门 . . . . .	15
<b>第二章 页面处理</b>	<b>16</b>
2.1 页眉和页脚 . . . . .	17
2.2 页版式与页边距 . . . . .	18
2.2.1 页面大小 . . . . .	18
2.3 纸张方向 . . . . .	19
2.4 文字竖排 . . . . .	20
2.5 换行 . . . . .	20
2.6 换页 . . . . .	22
2.7 分栏 . . . . .	23
2.8 水印 . . . . .	26
<b>第三章 单位与距离</b>	<b>27</b>
3.1 单位 . . . . .	27
3.1.1 长度 . . . . .	27
3.1.2 数字 . . . . .	28
3.2 距离 . . . . .	29
3.2.1 距离变量的定义和设置 . . . . .	29
3.2.2 常用距离 . . . . .	30
3.2.3 距离的运算 . . . . .	31
3.3 常用版面距离 . . . . .	31

目录	3
----	---

3.3.1 字间距 . . . . .	31
3.3.2 行间距 . . . . .	31
3.3.3 段间距 . . . . .	32
3.3.4 首行缩进 . . . . .	32
3.3.5 悬挂缩进 . . . . .	32
3.4 行之间插入空隙 . . . . .	33
3.5 设置目录间距 . . . . .	33

#### **第四章 标题和目录** 33

4.1 标题介绍 . . . . .	33
4.2 添加目录 . . . . .	35
4.3 定义目录深度 . . . . .	35
4.4 添加标题引用 . . . . .	36
4.5 重新定义标题序号格式 . . . . .	37
4.5.1 英文环境 . . . . .	38
4.5.2 中文环境 . . . . .	38
4.6 更改目录标题 . . . . .	38
4.7 取消标题序号 . . . . .	39
取消标题序号的演示 . . . . .	39

#### **第五章 文字样式** 40

5.1 一般效果 . . . . .	40
5.1.1 正常输入 . . . . .	40
5.1.2 斜体 . . . . .	40
5.1.3 加粗 . . . . .	41
5.1.4 添加线 . . . . .	41
5.2 号字 . . . . .	42
5.3 字体 . . . . .	43
5.3.1 基本字体类型 . . . . .	44
serif . . . . .	44
sans-serif . . . . .	44
monospace . . . . .	44
5.3.2 字族 . . . . .	44
5.3.3 字体设置 . . . . .	44

<b>目录</b>	<b>4</b>
5.3.4 找到字体 . . . . .	45
5.3.5 局部调整字体 . . . . .	46
5.4 调整位置 . . . . .	47
5.5 上下标 . . . . .	49
5.6 特殊字符 . . . . .	49
5.7 标注拼音 . . . . .	50
5.8 文字阴影 . . . . .	51
5.9 文字高亮 . . . . .	51
 <b>第六章 段落样式</b>	<b>51</b>
6.1 段落间距 . . . . .	51
6.2 对齐 . . . . .	51
6.2.1 左对齐 . . . . .	51
6.2.2 右对齐 . . . . .	52
6.2.3 居中 . . . . .	52
6.2.4 两端对齐 . . . . .	52
6.2.5 分散对齐 . . . . .	53
6.3 环境的嵌套 . . . . .	53
6.4 引文环境 . . . . .	54
6.5 抄录环境 . . . . .	55
6.6 普通边框 . . . . .	55
6.7 其他环境 . . . . .	58
 <b>第七章 列表</b>	<b>58</b>
7.1 基本用法 . . . . .	58
7.2 自定义 Bullet . . . . .	61
7.2.1 自定义标签 . . . . .	61
7.2.2 自定义序号 . . . . .	64
7.3 版面与布局 . . . . .	67
7.3.1 对齐与间隔 . . . . .	67
7.4 其他 itemize 布局 . . . . .	68
 <b>第八章 表格</b>	<b>68</b>
8.1 表格基础用法 . . . . .	68

<b>目录</b>	<b>5</b>
8.2 表格合并 . . . . .	69
8.3 表格列宽调整 . . . . .	72
8.3.1 参数 p . . . . .	73
8.3.2 tabular 的附加命令 . . . . .	74
8.4 横置表格 . . . . .	74
8.5 长表格 . . . . .	76
8.6 表格美化 . . . . .	78
8.7 添加表格说明 . . . . .	79
8.8 自动生成表格工具 . . . . .	79
8.8.1 Tables Generator . . . . .	79
8.8.2 LatexTool . . . . .	80
<b>第九章 图片</b>	<b>83</b>
9.1 基本使用 . . . . .	84
9.2 指明图片路径 . . . . .	88
9.3 图片优先级读取 . . . . .	89
9.4 添加图片说明 . . . . .	89
9.5 Tikz 绘图 . . . . .	89
<b>第十章 公式环境</b>	<b>91</b>
10.1 行内公式 . . . . .	92
10.1.1 数学环境中的普通文本 . . . . .	92
10.2 行间公式 . . . . .	92
10.2.1 单行行间公式 . . . . .	93
10.2.2 多行行间公式 . . . . .	93
10.3 公式编辑工具推荐 . . . . .	97
10.3.1 Mathpix . . . . .	97
10.3.2 在线 L <sup>A</sup> T <sub>E</sub> X 公式编辑器 . . . . .	97
<b>第十一章 Float</b>	<b>98</b>
11.1 Float 基本介绍 . . . . .	98
11.1.1 添加 Float 说明 . . . . .	98
11.1.2 更改 Float 说明格式 . . . . .	100
11.2 Figure . . . . .	100
11.2.1 插入单张图片 . . . . .	101

目录	6
----	---

11.2.2 插入多个图片 . . . . .	101
11.2.3 列出图片目录 . . . . .	102
11.3 Table . . . . .	103
11.3.1 列出表格目录 . . . . .	103

<b>第十二章 注释与引用</b>	<b>103</b>
-------------------	------------

12.1 脚注 . . . . .	103
12.1.1 基本使用 . . . . .	103
12.1.2 更改标记风格 . . . . .	103
12.1.3 重新计数脚注标记 . . . . .	104
12.2 边注 . . . . .	105
12.2.1 基本使用 . . . . .	105
12.2.2 marginnote 宏包 . . . . .	105
12.3 标签 . . . . .	106
12.3.1 其他的引用需求 . . . . .	106
12.4 参考文献 . . . . .	107
12.4.1 直接引入 . . . . .	107
12.4.2 bibtex 引入 . . . . .	109

<b>第十三章 颜色</b>	<b>111</b>
----------------	------------

13.1 引入 . . . . .	111
13.2 使用 . . . . .	112
13.3 定义 . . . . .	112
13.4 其他环境下的颜色设置 . . . . .	114
13.4.1 表格 . . . . .	114
13.5 超链接与引用 . . . . .	114
13.6 分割线 . . . . .	115
13.7 颜色相关工具 . . . . .	115
13.7.1 LatexColor . . . . .	115

<b>第十四章 命令和环境</b>	<b>115</b>
-------------------	------------

14.1 命令 . . . . .	116
14.1.1 命令的基本使用 . . . . .	116
14.1.2 命令的定义 . . . . .	116
14.1.3 命令的重新定义 . . . . .	118

14.2 环境 . . . . .	118
14.2.1 环境的基本使用 . . . . .	118
14.2.2 环境的定义 . . . . .	118
14.2.3 环境中的命令 . . . . .	120
14.2.4 环境的重新定义 . . . . .	121
14.3 高级命令与环境 . . . . .	122
14.4 计数器 . . . . .	123
14.4.1 基本介绍 . . . . .	123
14.4.2 生成计数器 . . . . .	124
14.4.3 计数器风格 . . . . .	125
<b>第十五章 代码环境方案</b>	<b>126</b>
15.1 程序代码 . . . . .	126
15.1.1 抄录环境 . . . . .	126
15.1.2 listing . . . . .	126
15.1.3 minted . . . . .	126
15.1.4 最佳实践: tcolorbox . . . . .	126
15.2 伪代码 . . . . .	126
15.3 树结构 . . . . .	126
<b>第十六章 编写结构</b>	<b>126</b>
<b>第十七章 库使用</b>	<b>126</b>
17.1 minipage . . . . .	126
17.2 tcolorbox . . . . .	127
17.3 adjustbox . . . . .	127
<b>第十八章 高级使用</b>	<b>127</b>
<b>第十九章 模板收录</b>	<b>127</b>

## 插图

1 L <sup>A</sup> T <sub>E</sub> X 扩展 . . . . .	10
2 设置 GistID 共享 VSCode 配置 . . . . .	10
3 VSCode 编译选项示意 . . . . .	11

表格	8
----	---

4 Table-Generator 网站截图	80
5 TikZ and PGF examples 网站截图	90
6 Mathpix 官网示例	97
7 caption 示例	99
8 居中示例	100
9 子图演示	102
10 latexcolor 网站截图	115

## 表格

1 TexLive 中的常用程序	14
2 xwatermark 宏包的页面指定方式	26
3 L <sup>A</sup> T <sub>E</sub> X 中的间距	27
4 L <sup>A</sup> T <sub>E</sub> X 内置的几种表格	28
6 表格说明	79
7 Float 的参数	98
8 参考文献的显示格式	110
9 L <sup>A</sup> T <sub>E</sub> X 内置的序号计数器	123
10 L <sup>A</sup> T <sub>E</sub> X 内置的控制计数器	124

# 第一章 安装与配置方案

## 1.1 各平台安装方法最佳实践

### 1.1.1 Windows 平台

Windows 下推荐采用 TexLive&VSCode&SumatraPDF。

在[TexLive](#)官网下载并安装 TexLive，随后安装[Sumatra PDF](#)和[VSCode](#)

安装好 TexLive 后，在命令控制台中输入

Code

```
1 latex -v  
2 xelatex -v
```

如果能够输出版本信息，证明 Tex 已经安装好了。

随后需要简单更改一下 TexLive 的配置，找到 TexLive 安装根目录下的texmf.cnf文件，打开并添加如下的一行参数，其余配置维持不变：

Code

```
1 openout_any = a
```

如果不添加该参数，在添加参考文献时可能会报错。

随后，在 VSCode 中安装 Tex 扩展，快捷键Ctrl+Shift+X打开扩展面板，查找并选择LateX Workshop并安装：



图 1: LATEX 扩展

随后需要在 vscode 的设置中配置编译命令，从而能够用快捷键迅速编译，配置主要是配置编译的工具和食谱（recipes，即工具的联合使用），官网提供了很详细的说明[文档](#)，以及[这篇博客](#)也提供了比较详细的配置方法。

如果嫌麻烦，目前的 vscode 配置支持和其他用户分享 GistID 来共享配置，可以使用我的配置对应的 GistID（可能仍需要配置 PDF 阅读器的位置）：

999ba45b6ca3350048541fd7f72ef07a

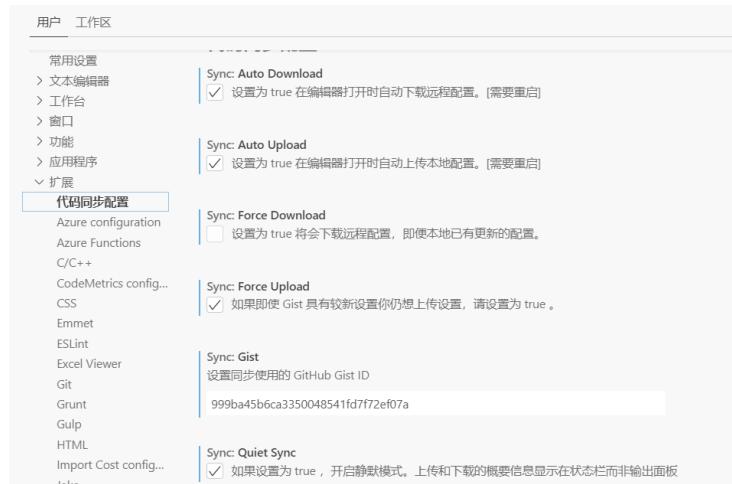


图 2: 设置 GistID 共享 VSCode 配置

安装好后，任意新建一个 tex 后缀的文档，输入以下内容（注意用半角

标点符号，以及去掉前面的注释符号%<sup>1</sup>)：

```
% \documentclass{article}
% \begin{document}
%     hello world
% \end{document}
```

正常编译后如果能够生成一个 pdf 文档，那么就说明安装完成了。

如果使用了之前的配置 ID，那么快捷键是 Shift+F10

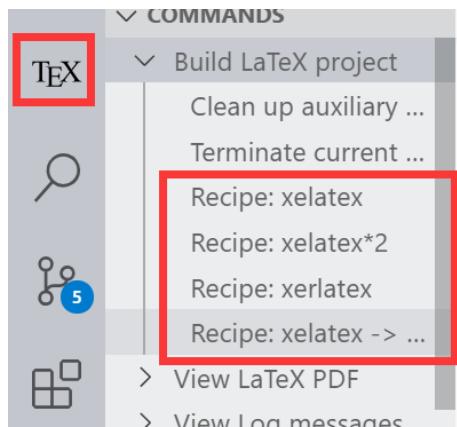


图 3: VSCode 编译选项示意

## 1.2 中文支持

LATEX 有比较多的中文支持方法，目前最好用的是ctex包。以及虽然 ctex 提供了一些文档类，但是还是推荐用添加包的方式来使用，这样会比较容易兼容纯英文方案。

```
\usepackage[UTF8,fontset=windowsnew,heading=true]{ctex} % 使用包的方式添加时推荐使用的参数
```

之后默认情况下不需要其他配置，就可以完美的支持中文了，但是注意，需要使用XeLaTeX编译。

<sup>1</sup>因为未知原因，代码环境无法插入该语句，还未解决

### 1.2.1 Linux 平台

不熟悉，暂略

### 1.2.2 MAC 平台

不熟悉，暂略

## 1.3 常用包

LATEX 由于其开放的环境，有成千上万的包提供用户使用，在一般情况下，比较常用的包和配套的参数如下所示：

```
\usepackage{amsmath}% 提供数学公式支持
\usepackage{graphics}% 用于添加图片
\usepackage{graphicx}% 加强插图命令
\usepackage{subfigure}% 用于添加子图
\usepackage{wrapfig}% 提供图片环绕风格支持
\graphicspath{{./}{./contents/}{./contents/fig/}}% 设置图片可能存在的
→ 路径
\newcommand{\figpath}[1]{contents/fig/#1} % 设置图片路径

\usepackage{fontspec}% 用于配置字体
\usepackage[table]{xcolor}% 用于各种颜色环境
\usepackage{enumitem}% 用于定制 list 和 enum
\usepackage{float}% 用于控制 Float 环境，添加 H 参数（强制放在 Here）
\usepackage[colorlinks,
            linkcolor=black,
            urlcolor=blue,
            anchorcolor=blue,
            citecolor=green]{hyperref}% 用于超链接，另外添加该包目录会
→ 自动添加引用。

\usepackage[most]{tcolorbox}% 用于添加各种边框支持，most 参数表示将全
→ 部命令支持都添加进来
```

```
\usepackage[cache=true,outputdir=./out]{minted}%
↪ 设置 cache=false, 如果输出设置了其他目录那么 outputdir 参数也有手
↪ 动指定, 否则会报错。
\tcbuselibrary{minted}%
加载 tcolorbox 的代码风格

\usepackage[a4paper, left=4cm, right=4cm, top=3cm, bottom=1cm]{geometry}%
↪ 用于控制版式

\usepackage{appendix}%
用于控制附加文件

\usepackage[UTF8,fontset=windowsnew,heading=true]{ctex} %
用于提供中
↪ 文支持
\ctexset{
    section = {
        number = 第\chinese{section}章,
        format = \zihao{3}\bfseries,
    },
    subsection = {
        number = \arabic{section}.\arabic{subsection},
        format = \Large\bfseries
    },
    subsubsection = {
        number =
            ↪ \arabic{section}.\arabic{subsection}.\arabic{subsubsection},
        format = \Large\bfseries,
    },
}
\usepackage{multirow} %
用于表格环境多行多列的支持
\usepackage{pdfpages} %
用于插入 pdf 页
\usepackage{ulem} %
用于好看的下划线、波浪线等修饰
\usepackage{fixltx2e} %
用于文本环境的下标

\usepackage{adjustbox} %
用于调整"盒子"位置
\usepackage{longtable}%
用于提供长表格支持
\usepackage{nameref} %
用于在引用时附加描述而不只是计数值
```

## 1.4 L<sup>A</sup>T<sub>E</sub>X 内置命令

L<sup>A</sup>T<sub>E</sub>X 中除了编译器相关的命令外，还提供了许多别的非常方便的命令，列举如下：

### 1.4.1 查看帮助文档

在熟练 L<sup>A</sup>T<sub>E</sub>X 中的操作后，如果需要查看某个宏包的文档，可以尝试在命令行输入：

Code

```
1 texdoc xxx
```

### 1.4.2 其他命令

命令名	备注
bibtex	参考文献支持。
makeindex,xindy	索引支持。
dvips	将 DVI 转换为 PostScript
xdviX	WindowSystem 下的 DVI 阅读器。
dviconcat,dviselect	从 DVI 文件中复制和粘贴页面。
dvipdfmx	将 DVI 转换为 PDF，是(前面提到过的)pdfTEX 的一套替换方案。
psselect,psnup,...	PostScript 实用程序。
pdfjam, pdfjoin, ...	PDF 实用程序。
context,mtxrun	ConTEXt 和 PDF 处理工具。
htlatex,...tex4ht	(LA)TEX 到 HTML(还有 XML 等其他格式) 的转换器

表 1: TexLive 中的常用程序

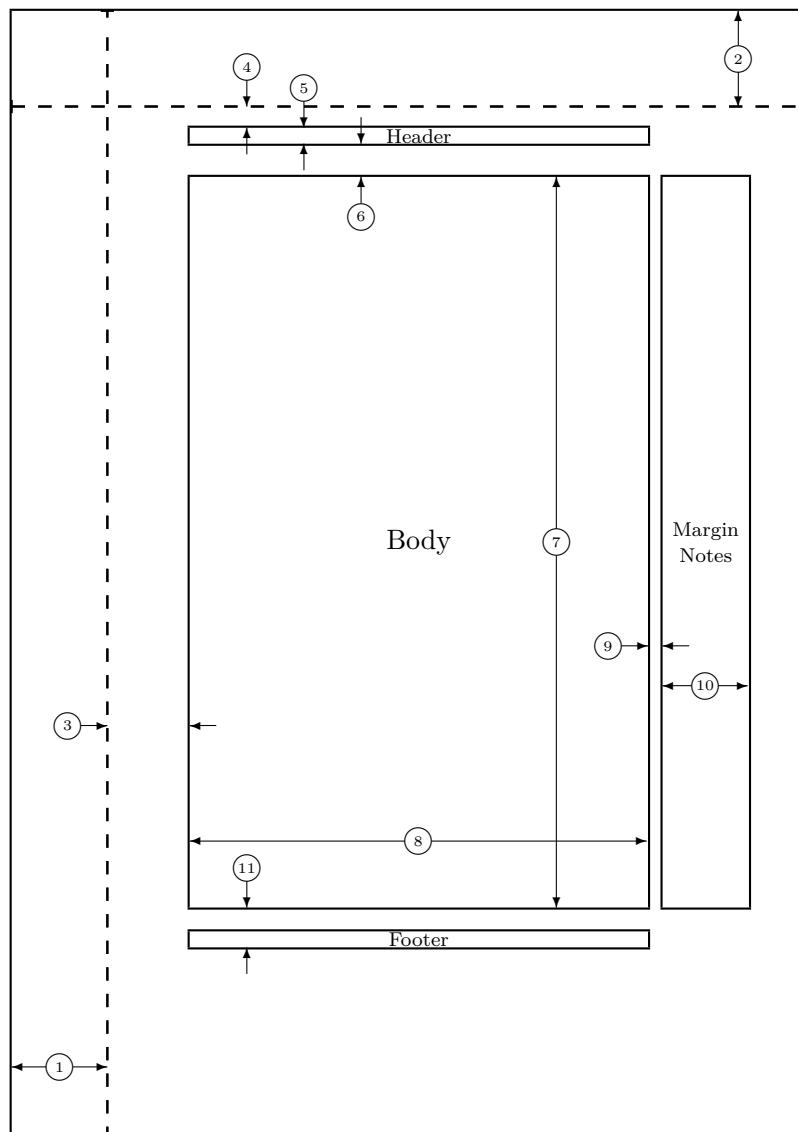
## 1.5 快速入门

在按上面的方式安装好并完成编译测试后，可以按以下顺序查看：

- 第十四章：命令和环境
- 第二章：页面处理
- 第四章：标题和目录
- 第三章：单位与距离
- 第五章：文字样式
- 第六章：段落样式
- 第七章：列表
- 第九章：图片
- 第八章：表格
- 第十章：公式环境
- 第十一章：Float
- 第十二章：注释与引用
- 第十三章：颜色

## 第二章 页面处理

LATEX 中的页面结构如下图所示：



```
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
11 \footskip = 30pt             10 \marginparpush = 5pt (not shown)
                                \voffset = 0pt
                                \paperwidth = 597pt
                                \paperheight = 845pt
```

LATEX 中各种命令的本质就是将文字、图片等“资源”放到一个个盒子中，并将这些盒子按一定的规则排布在页面上。文字默认都展示在 Body 主体中，如果需要设置页眉和页脚以及边注，就需要特殊的命令。

该图使  
用layout宏包，  
\layout命令  
生成

## 2.1 页眉和页脚

设置页眉和页脚需要的宏包有fancyhdr和titlesec两种。经过多次测试后发现，fancyhdr中的命令有时会因为其他宏包的引入而失效，同时考虑到在使用中更改多种页面风格的需求，因此使用titlesec在大多数情况下是一个更好的选择<sup>2</sup>。

本节仅介绍 titlesec<sup>3</sup>的使用方法。

宏包引入：

```
\usepackage{titlesec}
```

通过使用newpagestyle来定义一种页面风格，定义分奇数页和偶数页两种，当版式不分奇偶时，则使用奇数页（奇数页是 odd）的格式，定义方法如下：如：

使用定义好的页面风格需要在正文中声明，在序言区使用会出错。：

如果要在页眉或页脚处添加分界线，使用该宏包实现也非常的简单：

<sup>2</sup>要注意，该宏包和 fancyhdr 只能选择一个引入，否则会因为定义冲突而报错

<sup>3</sup>该宏包已开源，[链接](#)

```
\newpagestyle{mystyle}{%
    \headrule % 添加页眉处分割线
    \footrule % 添加页脚处分割线
    \setheadrule{1mm}%
    \setfootrule{0.4pt}%
    \sethead
    [第\thesection{}章 ~~\sectiontitle] [] [\thepage]%
    {第\thesection{}章 ~~\sectiontitle}{}{\thepage}
    \setfoot
    [] [\LaTeX{}速查手册] []
    {}{\LaTeX{}速查手册}{}%
}
```

## 2.2 页版式与页边距

设置页版式和页边距等需要的包是geometry

### 2.2.1 页面大小

geometry 宏包提供默认的页面大小用于设置，如下：

```
\usepackage[a4paper]{geometry}%
    \usepackage[a4paper=true]{geometry}%
    \usepackage[a4paper=false]{geometry}%
    ↳ 全局版式
    ↳ 数设置全局版式
    ↳ 数设置全局版式
```

注意赋值会被忽略掉，以上三种方式的效果是相同的。

geometry 提供的版式参数一共有 6 类，如下所示：

类型	示例	备注
a[0-6]paper	a0paper, a1paper...	ISOA 标准
b[0-6]paper	b0paper, b1paper...	ISOB 标准
c[0-6]paper	c0paper, c1paper...	ISOC 标准
b[0-6]j	b0j, b1j, b2j...	JIS(Japanese Industrial Standards)B 标准
ansi[a-e]paper	ansiapaper, ansibpaper...	未知
其他	letterpaper, executivepaper, legalpaper.	未知

除了使用 `geometry` 提供的版面大小，还可以通过`papersize`属性自定义纸张大小：

```
\usepackage[papersize={20cm, 15cm}]{geometry}% {宽, 高}
```

## 2.3 纸张方向

在不改变版面大小的情况下，将所有页面的纸张横置只需要在添加`geometry`包的时候添加`landscape`参数：

```
\usepackage[landscape]{geometry}
```

如果需要单张的纸张更改纸张方向，实现方式比较多，但是都不能非常完美的完全更改过来（如页眉页脚在左右两侧等）。经过多个解决方案的尝试，该需求的最佳实践<sup>4</sup>需要使用宏包`etoolbox`，并且在`序言区`声明添加以下代码：

---

<sup>4</sup>参考自[StackExchange](#)

```
% \usepackage{etoolbox}
\makeatletter
\def\ifGm@preamble{\@firstofone}
\appto\restoregeometry{%
\pdfpagewidth=\paperwidth
\pdfpageheight=\paperheight}
\apptocmd\newgeometry{%
\pdfpagewidth=\paperwidth
\pdfpageheight=\paperheight}{}{%
\makeatother}
```

随后，如果如果需要更改纸张方向，则在更改的地方使用以下命令（以默认竖向为例）：

```
\newgeometry{hmargin=3cm,vmargin=5cm,landscape} % 前面两个间距的参数
↪ 可以省略
```

随后的页面将完美的横置过来，直到想要重新恢复竖排时，使用以下命令：

```
\restoregeometry
```

## 2.4 文字竖排

由于 L<sup>A</sup>T<sub>E</sub>X 主要用于学术用途，因此竖排的需求不是很常见。经查阅在[这篇文章](#)中有提到过如何实现文字竖排，留做坑，日后有需要再填。

## 2.5 换行

在 L<sup>A</sup>T<sub>E</sub>X 中，单次换行并不会在排版中真正的换行：

**Code effects**

```
tex 文件中的第一行  
tex 文件中的第二行
```

tex 文件中的第一行 tex 文件中的第二行

如果需要换行,有四种方式,分别是空一行、使用双斜线、使用`\newline`命令和使用`\par`命令,这四种命令在一般情况下等价,如下:

**Code effects**

```
tex 文件中的第一行  
  
tex 文件中的第二行\\  
tex 文件中的第三行\newline  
tex 文件中的第四行\par  
tex 文件中的第五行
```

tex 文件中的第一行  
tex 文件中的第二行  
tex 文件中的第三行  
tex 文件中的第四行  
tex 文件中的第五行

另外,如果要临时变化与下一行的行间距,可以在`\`命令后添加`[offset]`参数:

**Code effects**

tex 文件中的第一行

tex 文件中的第二行`\[1em]`

tex 文件中的第三行`\newline %\newline` 命令没有办法添加参数

tex 文件中的第四行

tex 文件中的第一行

tex 文件中的第二行

tex 文件中的第三行

tex 文件中的第四行

最后，还有一种换行方式`\linebreak`，除了换行，还会将当前行分散对齐，效果如下：

**Code effects**

tex 文件中的第一行`\linebreak`

tex 文件中的第二行`\linebreak`

tex 文件中的第三行

tex 文件中 的 第 一 行  
tex 文件中 的 第 二 行

tex 文件中的第三行

## 2.6 换页

有时会遇到重新另起一页的需求（如目录结束后从新的一页开始），需要使用`\clearpage`或`\newpage`命令

`\tableofcontents`

`\newpage`

## 2.7 分栏

分栏主要的需求是，全部分栏，部份分栏，在全部分栏中临时不分栏。

如果需要全部分栏，只需要在初始文档类的声明中添加twocolumn参数即可，如下：

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

如果要对某一部分内容分栏，则需要使用multicols环境<sup>5</sup>。

### Code effects

```
\begin{multicols}{2}% 列数
```

多栏环境下，可以添加任何\LaTeX{}命令，但是不允许浮动，本手册中提及的长表格的使用页必须在单栏环境下才可以使用。  
→ 及的长表格的使用页必须在单栏环境下才可以使用。

```
\end{multicols}
```

```
\begin{multicols}{3}
```

多栏环境下，可以添加任何\LaTeX{}命令，但是不允许浮动，本手册中提及的长表格的使用页必须在单栏环境下才可以使用。  
→ 及的长表格的使用页必须在单栏环境下才可以使用。

```
\end{multicols}
```

多栏环境下，可以添加任何\LaTeX{}提及的长表格的使用页必须在单栏命令，但是不允许浮动，本手册中提及的长表格的使用页必须在单栏环境下才可以使用。

多栏环境下，可以添加任何\LaTeX{}但是不允许浮动，本的使用页必须在单栏命令，手册中提及的长表格的使用页必须在单栏环境下才可以使用。

可以通过添加参数的方式选择显示在分栏上方的文字：

<sup>5</sup>表格支持多行多列的宏包是 multirow，两者只是名字相近

**Code effects**

```
\begin{multicols}{2}[这里是分栏示例]
```

多栏环境下，可以添加任何`\LaTeX{}`命令，但是不允许浮动，本手册中提及的长表格的使用页必须在单栏环境下才可以使用。

```
\end{multicols}
```

这里是分栏示例

多栏环境下，可以添加任何`\LaTeX{}`提及的长表格的使用页必须在单栏环境下才可以使用。

列中间的分隔由`\columnsep`来控制，可以通过更改它的数值来控制间距：

**Code effects**

```
\setlength{\columnsep}{0.4pt}% 在 multicols 环境前设置
```

```
\begin{multicols}{2}[这里是分栏示例]
```

多栏环境下，可以添加任何`\LaTeX{}`命令，但是不允许浮动，本手册中提及的长表格的使用页必须在单栏环境下才可以使用。

```
\end{multicols}
```

这里是分栏示例

多栏环境下，可以添加任何`\LaTeX{}`提及的长表格的使用页必须在单栏环境下才可以使用。

同样，列中间也可以添加有颜色的分割线（默认黑色），来让分隔更清晰：

**Code effects**

```
\setlength{\columnseprule}{1pt}
\renewcommand{\columnseprulecolor}{\color{red}}
\begin{multicols}{3}[这里是分栏示例]
    多栏环境下，可以添加任何\LaTeX{}命令，但是不允许浮动，本手册中提及的长表格的使用页必须在单栏环境下才可以使用。
\end{multicols}
```

这里是分栏示例

多栏环境下，可以添加任何\LaTeX{}命令，但是不允许浮动，本手册中提及的长表格的使用页必须在单栏环境下才可以使用。

在全多栏环境中，有时会需要插入跨栏的文字、表格或图片，对于表格和文字， $\text{\LaTeX}$  提供了星号环境来实现多栏环境中的跨栏插入：

```
\begin{figure*}
    ...
\end{figure*}
\begin{table*}
    ...
\end{table*}
```

单栏文本在双栏环境中的插入要更困难一些，暂时没有太好的办法，只能使用\onecolumn临时声明，如下：

```
\onecolumn 使用这种方法来插入部份的单栏文本
\twocolumn
```

注意必须在最后重新声明双栏环境，否则要么无法编译通过，要么之后会一直是单栏环境。

## 2.8 水印

水印的实现方式有几种，各有优缺点，这里选择最方便容错性更高且支持中文的的宏包`xwatermark`。

该宏包使用只需要引入，然后定义水印即可，水印分文字水印和颜色水印两种，均可以定制大小、位置、显示页数等参数，同时可以多个水印同时使用，水印之间能够互相嵌套，非常方便。

使用方法：

```
\usepackage[printwatermark, disablegeometry]{xwatermark}

\newwatermark[page=1,color=gray!20,angle=45,scale=4,xpos=0,ypos=0]
    {\SAILIST}
\newwatermark[pages=2-4,color=gray!20,angle=45,scale=3,xpos=0,ypos=0]
    {\LaTeX{}速查手册}
\newwatermark[page={5,6,7},
    color=gray!20,angle=45,scale=3,xpos=0,ypos=0]
    {\LaTeX{}速查手册}
```

上述的代码已经在在本手册中执行，可以通过查看封面和目录页观察其效果。

每种水印必须指定页数，该宏包支持的页数指定方式有 8 种：

page=x	pages=x-y	pagex={x,y,z}	firstpage
lastpage	allpages=true	evenpages=true	oddpages=true

表 2: xwatermark 宏包的页面指定方式

关于该宏包的更多使用方式，如图片水印等，参考该宏包的[文档](#)

## 第三章 单位与距离

### 3.1 单位

$\text{\LaTeX}$  中使用的单位可以分为长度和数字两种，长度单位如厘米、毫米等，数字单位是指数字的显示方式，将数字以原本形式展示（阿拉伯数字），以字母方式（a、b、c...），以罗马数字…等

#### 3.1.1 长度

$\text{\LaTeX}$  中的长度单位比较多，有以下几种：

eng	含义	单位转换
mm	毫米	$1 \text{ mm} = 2.845 \text{ pt}$
pt	点	$1 \text{ pt} = 0.351 \text{ mm}$
bp	大点	$1 \text{ bp} = 0.353 \text{ mm} > 1 \text{ pt}$
dd	迪多	$1 \text{ dd} = 0.376 \text{ mm} = 1.07 \text{ pt}$
pc	排卡	$1 \text{ pc} = 4.218 \text{ mm} = 12 \text{ pt}$
sp	定标点	$65536 \text{ sp} = 1 \text{ pt}$
cm	厘米	$1 \text{ cm} = 10 \text{ mm} = 28.453 \text{ pt}$
cc	西塞罗	$1 \text{ cc} = 4.513 \text{ mm} = 12 \text{ dd} = 12.84 \text{ pt}$
in	英寸	$1 \text{ in} = 25.4 \text{ mm} = 72.27 \text{ pt}$
ex	ex	$1 \text{ ex} = \text{当前字体尺寸中 x 的高度}$
em	em	$1 \text{ em} = \text{当前字体尺寸中 M 的宽度}$

表 3:  $\text{\LaTeX}$  中的间距

对水平距离的设置常用 *em*，而对垂直距离的设置，如行距，常用 *ex*。

### 3.1.2 数字

LATEX 中，经常会遇到数字格式的问题，如itemize环境，或者章节前的序号，它们的格式实际上都是通过 LATEX 里的计数器（从 0 开始）、具体数值和数字风格转换方式来实现的，其中，默认的数字风格的转换有如下几种：

表示	含义	示例	备注
\arabic	阿拉伯数字	0,1,2...	
\roman	小写罗马数字	i,ii,iii...	
\Roman	大写罗马数字	I,II,III...	
\alph	小写字母	a,b,c...	范围为 0-25
\Alph	大写字母	A,B,C...	范围为 0-25

表 4: LATEX 内置的几种表格

注意，它们的使用无法通过简单的在方法中嵌套数字来使用（如直接使用\u{alpha}），而是需要结合计数器<sup>6</sup>来显示，如：

#### Code effects

当前的章节号为\arabic{section}

当前的章节号为 3

对于中文支持，我们经常看到章节号会出现“一、二、三”，这在ctex的文档中提到，这其实是由zhnumber包来完成的，其中有关计数器的内容主要是由\u{chinese}{counter}来完成的，如：

<sup>6</sup>有关更多计数器的内容，查看14.4：计数器

**Code effects**

```
当前的子章节号为\chinese{subsection}
```

当前的子章节号为一

## 3.2 距离

### 3.2.1 距离变量的定义和设置

LATEX 排版中，有很多需要指定距离的需求，如果每次都使用长度单位，一旦更改会非常的麻烦，因此 LATEX 可以定义一个距离变量，方式如下：

```
\newlength\examplelen
```

注意：距离变量是唯一的，因此如果定义重名会报错。

距离变量定义好后，可以随时根据需要设置距离变量的长度，如下：

**Code effects**

```
% 事先已经使用了 \newlength\examplelen  
\setlength{\examplelen}{3em}  
长度演示\hspace{\examplelen}长度演示\\  
长度演示\hspace[3em]长度演示\\
```

```
\setlength{\examplelen}{2em}  
长度演示\hspace{\examplelen}长度演示
```

长度演示      长度演示  
长度演示      长度演示

长度演示      长度演示

### 3.2.2 常用距离

LATEX 中常用的距离有以下几种：

```
\ linewidth - 当前行的宽度  
\ columnwidth - 当前分栏的宽度  
\ textwidth - 整个页面版芯的宽度  
\ paperwidth - 整个页面纸张的宽度
```

在单栏文本中，\columnwidth 和 \textwidth 保持一致；在多栏文本中：

$$\text{textwidth} = n * \text{columnwidth} + (n - 1) * \text{columnsep} \quad (\text{其中 } n \text{ 是分栏数})$$

在 minipage 环境中，除了 \paperwidth 之外，其它三种都会根据 minipage<sup>7</sup> 的宽度发生改变（因为虚拟出了一个小的纸张页面），并且在 minipage 环境结束的时候恢复原样。

在 parbox 中，\textwidth 和 \columnwidth 不会改变，不过 \linewidth 会发生变化。

\linewidth 是相对最灵活的宽度值。在 list 环境里（包括 enumerate 和 itemize 等环境），在 \parbox 里，\linewidth 都会发生变化。总的来说，当

- 需要在列表环境中使用表格、图片等宽度的时候，用 \linewidth
- 需要充满整个页面宽度的时候，用 \textwidth（比如 figure/table 等）
- 需要充满整个分栏的时候，用 \columnwidth（比如 figure/table/tabularx/tabu 等）

<sup>7</sup> minipage 的介绍和用法，参考17.1： minipage

### 3.2.3 距离的运算

所有的长度单位都可以通过在前面添加数字（正负、小数整数均可）来表示其倍数长度，如：

```
-2.5\textwidth
```

如果是单位相同的两个长度，可以直接进行加减操作：

```
\textwidth-4\fboxsep
```

如果是不同长度的单位，可以使用dimexpr宏包进行运算：

```
% \usepackage{dimexpr}  
\dimexpr (\textwidth -8\tabcolsep)
```

由于使用较少，因此关于该宏包的具体使用方法暂略。

## 3.3 常用版面距离

### 3.3.1 字间距

一般没有调整字间距的需求，如果实在需要，可以查阅使用microtype（用于英文）和ctex宏包（用于中文）中的CJKglue命令。

这里有一篇[中文博客](#)，但没有经过验证

### 3.3.2 行间距

```
\linespread{1.3}% 设置行距为 1.3 倍行距
```

### 3.3.3 段间距

```
\setlength{\parskip}{0.5em}
```

### 3.3.4 首行缩进

默认有首行缩进，如果需要默认首行不缩进，则使用indentfirst包。

指定某段首行缩进（默认情况下不需要加，在使用indentfirst宏包后如果要指定缩进可以使用）：

```
% \usepackage{indentfirst}  
\indent 正常文本
```

指定某段首行不缩进，在段首加：

```
\noindent 正常文本
```

设置缩进量

```
\setlength{\parindent}{2em}
```

### 3.3.5 悬挂缩进

如果需要设置悬挂缩进，那么首先要取消首行缩进。

```
\noindent\hangafter=1 % 设置从第一行之后开始悬挂缩进
```

设置悬挂缩进量：

```
\setlength{\hangindent}{2em}
```

## 3.4 行之间插入空隙

插入指定长度大小的空隙，可以使用\vspace：

### Code effects

```
%\vspace{宽度大小} 和\vspace*{宽度大小}  
上一行\vspace{1cm}\\\n  
下一行\\  
正常间距测试
```

上一行

下一行  
正常间距测试

## 3.5 设置目录间距

```
\renewcommand{\baselinestretch}{0.75}\normalsize  
\tableofcontents  
\renewcommand{\baselinestretch}{1.3}\normalsize % 在设置后设置回原来  
→ 的间距大小
```

# 第四章 标题和目录

## 4.1 标题介绍

LATEX 中一共支持 7 个级别的标题，分别如下：

```
\part{part} % 级别为-1  
\chapter{chapter} % 级别为 0  
\section{section} % 级别为 1  
\subsection{subsection} % 级别为 2  
\subsubsection{subsubsection} % 级别为 3  
\paragraph{paragraph} % 级别为 4  
\ subparagraph{subparagraph} % 级别为 5
```

其中part级别和chapter级别仅在 report 和 book 的文档类中才可用。

```
\section{标题 A}  
\subsection{二级标题 A}  
\subsection{二级标题 B}  
\subsubsection{三级标题 A}  
\subsubsection{三级标题 B}  
\paragraph{段落 A}  
\paragraph{段落 B}  
\ subparagraph{子段落}  
\section{标题 B}
```

## 第一章 标题 A

### 1.1 二级标题 A

### 1.2 二级标题 B

#### 1.2.1 三级标题 A

#### 1.2.2 三级标题 B

段落 A

段落 B

子段落

## 第二章 标题 B

## 4.2 添加目录

要想在文章中添加目录，只需要在要添加的位置加上一句 `\tableofcontents`：

```
\tableofcontents
\section{标题 A}
\subsection{二级标题 A}
\subsection{二级标题 B}
\subsubsection{三级标题 A}
\subsubsection{三级标题 B}
\paragraph{段落 A}
\paragraph{段落 B}
\subparagraph{子段落}
\section{标题 B}
```

### 目录

第一章 标题 A	1
1.1 二级标题 A . . . . .	1
1.2 二级标题 B . . . . .	1
1.2.1 三级标题 A . . . . .	1
1.2.2 三级标题 B . . . . .	1
第二章 标题 B	1

LATEX 需要编译两次<sup>8</sup>才可以完整的将目录显示出来，因为 LATEX 必须通过第一遍编译获取所有目录的引用，才可以在下一次编译的时候添加新的目录。

## 4.3 定义目录深度

默认情况下目录只支持到 `subsubsection`，如果需要增加目录深度，需要在序言位置就要使用以下的代码：

<sup>8</sup>实际上所有的具有引用性质的内容都需要编译两次才可以正常的显示出来，另外有一些需要跨页的（如 `longtable`），也需要多次的编译才可以正常的显示

```
\setcounter{tocdepth}{4}% 设置目录级数  
\setcounter{secnumdepth}{4}% 设置在几级目录前标记序号
```

## 目录

第一章 标题 A	2
1.1 二级标题 A . . . . .	2
1.2 二级标题 B . . . . .	2
1.2.1 三级标题 A . . . . .	2
1.2.2 三级标题 B . . . . .	2
1.B.2.1 段落 A . . . . .	2
1.B.2.2 段落 B . . . . .	2
第二章 标题 B	2

两行代码中的数字和4.1：标题介绍中备注的数字相同。

数字的设置不必相同，互相并不矛盾，根据需要更改即可，其中`tocdepth`表明目录支持到几级深度，`secnumdepth`表明标题序号标记到几级深度。另外第一遍编译的时候可能会看到目录中增加了目录级数但是章节号并没有被添加，这是正常现象，完全更新会延迟到下一次编译。

## 4.4 添加标题引用

使用`hyperref`宏包后，生成的目录会自动添加到具体章节的链接和书签（如果没有该宏包则目录没有链接）。但是如果在文章内引用某章节，仍然需要为章节添加引用<sup>9</sup>。

为章节添加引用的方法是使用`\label`唯一标记，该方法适用于任何你想引用某个位置的说明的地方，以本小节，其代码如下：

```
\subsection{添加标题引用}\label{sub:ref-in-title}
```

在使用时，则需要使用`\ref`命令，如：

<sup>9</sup>该方法不局限于为章节添加引用，参考12.3：标签

**Code effects**

在`\ref{sub:ref-in-title}`一节提到了如何对标题添加引用。

在[4.4](#)一节提到了如何对标题添加引用。

有时，不仅需要看到引用的序号，还想要知道相应的标题的名字。要完成这个需求，需要使用nameref包：

**Code effects**

```
% \usepackage{nameref}  
如何对标题添加引用可以查  
→ 看\ref{sub:ref-in-title}:\nameref{sub:ref-in-title}
```

如何对标题添加引用可以查看[4.4:添加标题引用](#)

如果嫌同时使用两个引用命令来完成上面的操作有些麻烦，可以通过重新定义一条引用命令<sup>10</sup>的方法来合并引用（本手册即使用这种方法）：

```
\newcommand{\Ref}[1]{\ref{#1}: \nameref{#1}}
```

**Code effects**

如何对标题添加引用可以查看`\Ref{sub:ref-in-title}`

如何对标题添加引用可以查看[4.4: 添加标题引用](#)

## 4.5 重新定义标题序号格式

有的时候会遇到需要自己定制标题序号的需求，对于中文环境和英文环境，相应的定制方式略有不同：

<sup>10</sup>如何定义一条命令参考[第十四章：命令和环境](#)

### 4.5.1 英文环境

英文环境，暂略（可以使用`titlesec`系列宏包）

### 4.5.2 中文环境

中文环境要更改序号需要使用`ctex`提供的命令，如下：

```
\ctexset{
  section = {
    number = 第\chinese{section}章,
    format = \zihao{3}\bfseries,
  },
  subsection = {
    number = \arabic{section}.\arabic{subsection},
    format = \Large\bfseries
  },
  subsubsection = {
    number =
      \arabic{section}.\arabic{subsection}.\arabic{subsubsection},
    format = \Large\bfseries,
  },
}
```

使用方法如上所示（这也是本手册使用的参数），每个级别的标题都可以分别自定义，其中`number`表示序号的格式，相关的知识可以查看[3.1.2：数字](#)的描述。`format`表示每个标题的格式，相当于对每个标题添加什么样的命令。

## 4.6 更改目录中的标题文字

有的时候，因为标题过长，会导致目录不好看，这个时候，可以通过给标题添加参数的方式，定义显示在目录中的标题的内容。

```
\section[短标题(显示在目录中的标题)]{长标题(显示在章节中的标题)}  
% 其他标题类方法同理
```

可以通过查看本节在目录和章节中的标题来认识其效果，其对应的代码为：

```
\subsection[更改目录标题]{更改目录中的标题文字}
```

## 4.7 取消标题序号

有的时候需要不显示标题的序号，只需要在命令后加 \* 号即可。

```
\section*[No Title Section]
```

但这时标题也不会出现的目录中，如果需要标题仍然在目录中出现需要在此处添加额外的声明，如下：

```
\section*[No Title Section]\addcontentsline{toc}{section}{Title of  
→ the section}  
% 三个参数分别表示添加的位置是\highunderline{正文目录(toc)}, 级别  
→ 是\highunderline{section}, 目录中显示的文字是\highunderline{Title  
→ of the section}
```

其中，\addcontentsline命令还可以用于添加其他类型的目录，如表格（第一个参数为lot），图片（lof）等，但并不是很常用。

## 取消标题序号的演示

使用代码如下，注意添加后会在目录中失去页码的引用：

```
\subsubsection*[取消标题序号的演示]  
\addcontentsline{toc}{subsubsection}{取消标题序号的演示}
```

## 第五章 文字样式

### 5.1 一般效果

#### 5.1.1 正常输入

在中文环境里，中文默认字体为宋体。

##### Code effects

```
普通文字\\n  
normal text
```

普通文字  
normal text

#### 5.1.2 斜体

在中文环境里，中文默认的斜体效果是楷体而不是倾斜。

##### Code effects

```
\\textit{斜体文字}\\n  
\\textit{Italian text}
```

斜体文字  
*Italian text*

斜体实际上有两种，一种是 italic，一种是 slanted，其中 italic 指“倾斜的字体”，而 slanted 则是指“字体的倾斜”，有细微的差别：

**Code effects**

```
\textit{斜体文字}\textit{Italian text}\\\textsl{倾斜文字}\textsl{slanted text}
```

斜体文字 *Italian text*

倾斜文字 *slanted text*

### 5.1.3 加粗

在中文环境里，中文默认的加粗效果是**黑体**而不是**宋体**加粗。

**Code effects**

```
\textbf{加粗文字}\\\textbf{normal text}
```

**加粗文字**

**normal text**

### 5.1.4 添加线

LATEX 提供默认的下划线underline但是有很多的缺点因此不推荐使用，推荐使用ulem宏包中的命令，该宏包还提供了许多其他的修饰操作，列举如下：

**Code effects**

```
% \usepackage{ulem}
\underline{下划线} \\
\uline{双下划线} \\
\wavy{波浪线} \\
\sout{中间删除线} \\
\xout{斜删除线} \\
\dashuline{虚线} \\
\dotuline{加点}
```

下划线

双下划线

波浪线

中间删除线

斜删除线

虚线

加点

## 5.2 字号

一般可以使用以下几种字号大小

**Code effects**

```
\tiny{tiny-极小} \\
\scriptsize{scriptsize-代码大小} \\
\footnotesize{footnotesize-脚注大小} \\
\small{small-小} \\
\normalsize{normalsize-正常} \% 默认字号
\large{large-大} \\
\Large{Large-很大} \\
\LARGE{LARGE-极大} \\
\huge{huge-巨大} \\
\Huge{Huge-很巨大}
```

tiny-极小

scriptsize-代码大小

footnotesize-脚注大小

small-小

normalsize-正常

large-大

Large-很大

LARGE-极大

huge-巨大

Huge-很巨大

在[CTex 手册](#)里包含了这些字号对应的磅数以及中文环境中重新设置这些大小的命令。

### 5.3 字体

LATEX 中，中文和英文环境各自的字体需要分别调整。首先介绍一下基本的字体分类：

### 5.3.1 基本字体类型

**serif** 即**衬线字体**, 指的是具有末端加粗、扩张或尖细末端, 或以实际的衬线结尾的一类字体。

serif 总是在文字末端做文章, 这样做的目的是增强可读性, 也就是说在字号比较小的时候, serif 一族的字体仍然是比较好辨认的。

在 L<sup>A</sup>T<sub>E</sub>X 中, 衬线字体以罗马字族 (Roman) 的形式存在的。

**sans-serif** 即**无衬线字体**。sans 是法语前缀, 意为 “无”, 在表意明确的情况下, 可以称为 sans。

sans 字体比较圆滑, 线条粗细均匀, 适合做艺术字、标题等, 与 “衬线字体” 相比, 如果字号比较小, 看起来就会有些吃力。

在 L<sup>A</sup>T<sub>E</sub>X 中, 无衬线字体以无衬线字族的形式存在。

**monospace** 即**等宽字体**, 有的字体不同的标点、英文字母、汉字宽度是不一样的, 在显示代码的时候就需要使用这种字体, 对齐后会很舒服。

### 5.3.2 字族

一个字族是指一组专门设计的、一起协调使用的字体。一般有正文, 粗体, 斜体, 斜粗体四种组成, 有时根据具体需要没有斜粗体也可以, 有时还会出现超过四种字形的字族。

L<sup>A</sup>T<sub>E</sub>X 默认有三种字族, 分别叫做 mainfont, 表示一般的罗马字族, sansfont, 表示无衬线字族, monofont, 表示等宽字族。

### 5.3.3 字体设置

使用 XeLaTeX 编译可以使用电脑中的系统字体, 而不只是使用 L<sup>A</sup>T<sub>E</sub>X 的内置字体, 所以这是目前使用 XeLaTeX 编译的一大原因。

如果不添加 cte 宏包, 那么一般使用 fontspec 宏包来完成字体 (主要是英文字体) 的设置; 如果使用了 ctex 宏包, 那么中英文字体均可以使用 ctex 宏

包提供的命令来设置，这里仅介绍使用 ctex 宏包的设置方法。按照本文的方法，基本能够满足日常使用，如果需要更高度的定制化，可以查看 xeCJK、fontspec、cjk 宏包，ctex 宏包的实现继承了这些宏包，因此在其文档中没有仔细解释如何使用这些命令。

ctex 宏包使用 setxxxfont 设置英文字体族字体<sup>11</sup>，使用 setCJKxxxfont 设置中文字体，中英文各有四种，分别是 main（正文默认），sans（无衬线），mono（等宽），math（数学环境）以设置罗马字族的英文字体为例：

```
\setmainfont[UprightFont=xxx,
  BoldFont=xxx,
  ItalicFont=xxx,
  BoldItalicFont=xxx,
  SmallCapsFont=xxx,
  SlantedFont=xxx]
]{Latin Modern Roman}
```

在大括号内设置整个字族的字体，但是有时候一个字体可能会缺少一些字形，如缺少斜体等，这个时候可以设置可选参数中的七个字形。

下面介绍字体参数应该如何填入。

### 5.3.4 找到字体

如果使用 XeLaTeX 编译，那么就可以使用系统字体<sup>12</sup>，否则只能使用在 Tex 安装目录下安装的字体。

在命令行输入以下命令，可以在当前目录得到一个文本文件<sup>13</sup>，里面显示了所有可用的字体文件。

**Code**

```
1 fc-list > fontlist.txt
```

<sup>11</sup> 这也是 fontspec 宏包使用的方法

<sup>12</sup> 其使用 fontconfig 库查找和调用字体

<sup>13</sup> 由于汉字编码原因，Windows 下总需要把字体列表输出的文件中防止乱码。

该命令也可以加上各种选项来进行筛选，如只需要列出所有中文字体<sup>14</sup>，使用命令：

#### Code

```
1 fc-list -f "%{family}\n" :lang=zh > zhfont.txt
```

一般，会得到如下的文字

C:/Windows/fonts/msyh.ttc: Microsoft YaHei,微软雅黑：  
style=Regular,Normal,obyčejné...

或者这样的文字

FZYaoTi,方正姚体  
FandolSong  
SimHei,黑体  
Microsoft YaHei,微软雅黑  
STFangsong,华文仿宋

可以直接使用字族名，或者文件名来设置字体，如：

```
\setmainfont{Microsoft YaHei}
\setmainfont[
    ItalicFont=STFangsong,
]{msyh.ttc}
```

### 5.3.5 局部调整字体

其中，mainFont 影响的是\rmfamily和\textrm命令的效果。

sansFont 影响的是\sffamily和\textsf命令的效果。

monoFont 影响的是\ttfamily和\texttt命令的效果。

<sup>14</sup>关键是其中的语言选项 zh。日文使用 ja，韩文使用 ko，英文使用 en

xxfamily 影响的是作用域内的，textxx 影响的是命令内的参数

**Code effects**

```
\sffamily san 文本开始的地方\\\texttt{mono 文本}\textsf{sans 文本}\rmfamily roman 文本}sans 文本结束
```

san 文本开始的地方  
mono 文本 sans 文本 roman 文本 sans 文本结束

## 5.4 调整位置

如果需要将文字上下偏移，可以使用\raisebox

**Code effects**

```
正常文本 \raisebox{1cm}{提升文本} \raisebox{-1cm}{下降文本} 正常文本
```

提升文本

正常文本

正常文本

下降文本

如果需要增加文字之间的间隙，可以使用\hspace

**Code effects**

```
正常文本\hspace{1em}正常文本\\
```

正常文本 正常文本

如果要填满空隙，可以使用\hfill：

**Code effects**

```
A \hfill A \hfill A \hfill A \\  
A \hspace{\stretch{2}} A \hfill A \hfill A \\  
A \hfill A \hfill A\hspace{0.5cm} A \hfill A \hfill A  
→ \hspace{0.5cm} A
```

A	A	A	A
A		A	A
A	A	A	A

注意第三个示例，每个\hfill占用的宽度在计算后都是一样的，但是会减去其他的固定长度的宽度。

另外，在数学公式中，需要使用其他的命令来帮助缩进（以下这些命令在普通环境也可以使用）

**Code effects**

```
$a\qquad{}b$\% 两个 quad 空格，两个字符 "M" 的宽度  
$a\quad{}b$\% quad 空格，一个字符 "M" 的宽度  
$a\; b$\% 大空格 1/3 字符 "M" 的宽度  
$a\;;b$\% 中等空格 ^~12/7 字符 "M" 的宽度  
$a\,,b$\% 小空格 1/6 字符 "M" 的宽度  
$ab$\% 没有空格  
$a\!b$\% 缩进 1/6 字符 "M" 的宽度
```

a	b
a	b
a	b
a	b
a	b
ab	
ab	

## 5.5 上下标

在数学环境中，使用上下标可以通过简单的 $\underline{\phantom{x}}$ ,  $\hat{\phantom{x}}$ 来实现，如下：

### Code effects

```
$\hat{A}^e_1, \underline{a}_3, b^5$  
$\hat{A}^{\text{ext}}_{\text{ext}}$ % 多个字母需要放在上下标中需要用花括号括起来
```

$A_l^e, a_3, b^5 A_{\text{ext}}^{\text{ext}}$

在普通环境中，不能直接使用数学环境中的方法，需要引入`fixltx2e`包来实现文本环境中的上下标

### Code effects

```
% \usepackage{fixltx2e}  
普通文本 \textsubscript{下标文本} \textsuperscript{上标文本}
```

普通文本<sub>下标文本</sub><sup>上标文本</sup>

## 5.6 特殊字符

$\text{\LaTeX}$  里存在一些特殊字符，如果需要在普通环境里使用，需要声明，具体的字符内容如下：

字符	#	%	%	{	}	$\sim$	$\_$	&	\
命令	<code>\#</code>	<code>\%</code>	<code>\%</code>	<code>\{</code>	<code>\}</code>	<code>\sim{}{}</code>	<code>\_{}{}</code>	<code>\&amp;</code>	<code>\textbackslash{}{}</code>

在数学环境中，直接使用小括号 () 和中括号 []，可能会不太美观，可以使用`\left`和`\right`来使其具备自动调整大小的能力：

**Code effects**

```
$\left( \text{foo} \right)$  
$\left[ \text{foo} \right]$  
$\left( \text{foo} \right.\left. \text{right}\right)$ %left 和 right 命令必须成对出现, 如果需要一边  
→ 某有括号, 可以将括号变成点来去掉相应位置的括号。  
$\left. \text{left} \right. \text{foo} \left. \text{right}\right]$ %left 和 right 命令必须成对出现, 如果需要一边  
→ 某有括号, 可以将括号变成点来去掉相应位置的括号。
```

(*foo*) [*foo*] (*foo* *foo*)

另外注意：大括号不可以使用这两个命令，否则会编译报错。

对于单双引号，正确用法是使用键盘左上角的反引号来表示左引号，靠近 L 键的引号表示右引号，其中键入一个表示单引号，连续两个表示双引号（在中文环境中这种差异并不明显，主要是在英文环境中会存在问题）：

**Code effects**

```
正确的‘单引号’与“双引号”\\  
' 英文中的单引号 ' \\  
" 英文中的双引号 " \\  
‘ 中文中的单引号 ’ \\  
“ 中文中的双引号 ”
```

正确的‘单引号’与“双引号”

‘ 英文中的单引号 ’  
” 英文中的双引号 ”  
‘ 中文中的单引号 ’  
“ 中文中的双引号 ”

## 5.7 标注拼音

可以使用xpinyin宏包实现，具体方法略。

## 5.8 文字阴影

可以使用 `shadowtext` 宏包实现，具体方法略。

## 5.9 文字高亮

文字高亮推荐使用 `tcolorbox` 实现，可定制性强。（包括带颜色的下划线，圆角高亮等，都可以使用该宏包来实现）。

# 第六章 段落样式

## 6.1 段落间距

段落间距的相关设置，可以参考[3.3：常用版面距离](#)一节。

## 6.2 对齐

对齐分为左对齐、右对齐、居中、两端对齐和行尾的分散对齐

### 6.2.1 左对齐

#### Code effects

```
\begin{flushleft}  
    左对齐环境  
\end{flushleft}  
\raggedright 左对齐声明
```

左对齐环境

左对齐声明

### 6.2.2 右对齐

#### Code effects

```
\begin{flushright}  
    右对齐环境  
\end{flushright}  
\raggedleft 右对齐声明
```

右对齐环境

右对齐声明

### 6.2.3 居中

#### Code effects

```
\begin{center}  
    居中环境  
\end{center}  
\centering 居中声明
```

居中环境

居中声明

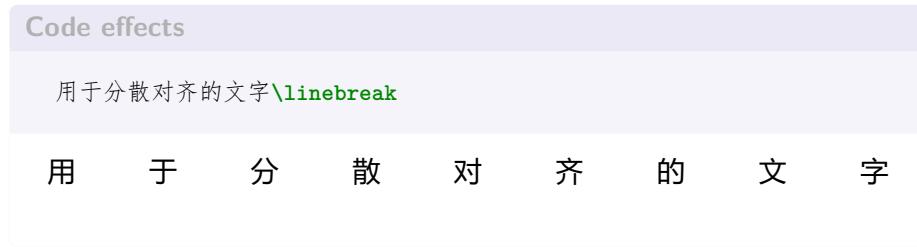
这三种环境和相应的命令基本上能够在任意环境中嵌套并显示出相应的效果。

### 6.2.4 两端对齐

两端对齐需要使用ragged2e宏包，并使用\justifying命令

### 6.2.5 分散对齐

分散对齐之前提到过，在换行的时候使用\linebreak即可：



## 6.3 环境的嵌套

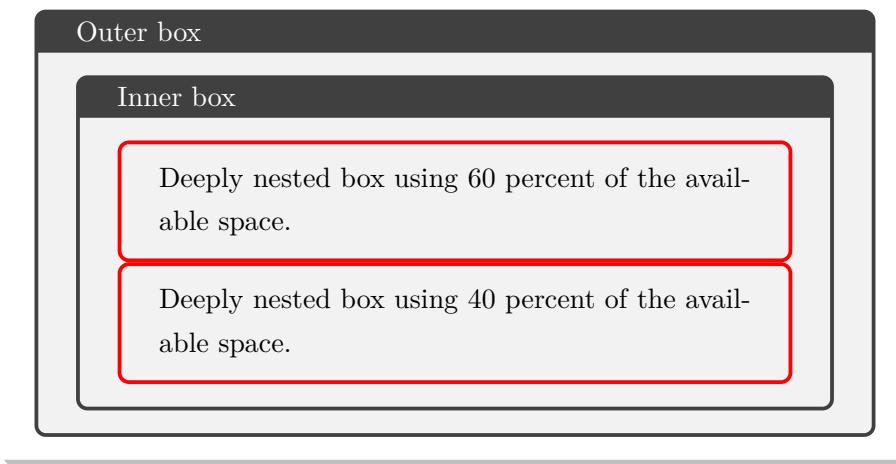
一般情况位于document环境中的文字均为普通文本，但是环境可以嵌套，被嵌套到特殊环境中的文字会从内到外依次根据所嵌套的环境产生相应的效果。如下图所示，使用了多个tcolorbox互相嵌套后的效果：

**Code effects**

```
\begin{tcolorbox}[title=Outer box]
\begin{tcolorbox}[title=Inner box]

\begin{tcolorbox}[colframe=red,beforeafter skip=0pt]
Deeply nested box using 60 percent of the available space.
\end{tcolorbox}

\begin{tcolorbox}[colframe=red,beforeafter skip=0pt]
Deeply nested box using 40 percent of the available space.
\end{tcolorbox}
\end{tcolorbox}
```



该环境需要使用tcolorbox包

## 6.4 引文环境

引文环境为quotation，最多支持六级嵌套，效果如下：

**Code effects**

```
正常环境下的文字  
\begin{quotation}  
引文环境下的文字  
\begin{quotation}  
二级引文环境  
\end{quotation}  
\end{quotation}  
正常环境下的文字
```

正常环境下的文字

引文环境下的文字

二级引文环境

正常环境下的文字

## 6.5 抄录环境

抄录环境为verbatim会将所有的符号原封不动的继承下来：效果如下：

**Code effects**

```
\begin{verbatim}  
这是抄录环境！@#￥%……&* () {}_+";>?  
\end{verbatim}
```

这是抄录环境！@#￥%……&\* () {}\_+";>?

## 6.6 普通边框

如果要限制一行中文字的宽度，可以使用垂直盒子\parbox：

**Code effects**

```
% \parbox[<position>]{<width>}{<text>}, 第一个参数一般省略, 没有太  
→ 多的使用场景。  
\parbox{5em}{示例文本示例文本示例文本示例文本示例文本示例文  
→ 本示例文本示例文本}
```

示例文本示  
例文本示例  
文本示例文  
本示例文本  
示例文本示  
例文本示例  
文本示例文  
本

对文字添加最普通的段落边框, 使用内置的\framebox命令即可:

**Code effects**

```
\framebox{示例文本示例文本示例文本示例文本示例文本示例文本示  
→ 例文本示例文本示例文本示例文本示例文本示例文本示例文本示  
→ 例文本示例文本示例文本示例文本示例文本示例文本示例文本示  
→ 例文本示例文本示例文本示例文本示例文本示例文本示例文本}
```

示例文本示例文本示例文本示例文本示例文本示例文本示例文本示例文本示例文本示例文本

遗憾的是直接使用 `framebox` 不能换行, 如果要换行, 可以再嵌套一  
层parbox:

**Code effects**

```
\framebox{  
  \parbox{\textwidth-2\fboxsep-2\fboxrule}{示例文本示例文本示例文  
  ↳ 本示例文本示例文本示例文本示例文本示例文本示例文本示  
  ↳ 例文本示例文本示例文本示例文本示例文本示例文本示例文  
  ↳ 本示例文本示例文本示例文本示例文本示例文本示例文本示  
  ↳ 例文本示例文本示例文本示例文本示例文本示例文本}  
}  
\framebox{  
  \parbox{\textwidth-4\fboxsep}{示例文本示例文本示例文本示例文  
  ↳ 示例文本示例文本示例文本示例文本示例文本示例文本示例文  
  ↳ 文本示例文本示例文本示例文本示例文本示例文本示例文本示例文  
  ↳ 示例文本示例文本示例文本示例文本示例文本示例文本示例文  
  ↳ 文本示例文本示例文本示例文本}  
}
```

示例文本示例文本示例文本示例文本示例文本示例文本示例文本示  
例文本示例文本示例文本示例文本示例文本示例文本示例文本示  
文本示例文本示例文本示例文本示例文本示例文本示例文本示  
本示例文本示例文本示例文本示例文本示例文本示例文本示  
示例文本示例文本

示例文本示例文本示例文本示例文本示例文本示例文本示例文本  
示例文本示例文本示例文本示例文本示例文本示例文本示例文本  
示例文本示例文本示例文本示例文本示例文本示例文本示例文本  
示例文本示例文本示例文本示例文本示例文本示例文本示例文本  
示例文本示例文本示例文本示例文本示例文本示例文本示例文本  
示例文本示例文本示例文本

边框距离文字的间距用长度变量 \fboxsep 表示：

Code effects

```
\setlength{\fboxsep}{Opt}  
\framebox{没有边框间隔}
```

没有边框间隔

## 6.7 其他环境

还有诗歌环境 verse 等文档类自带的环境，使用方式大同小异。但实现各种段落样式的最佳实践是通过 tcolorbox 来定义一个个盒子的样式。本书中所有的代码样例和边框等均依托于 tcolorbox 实现。

# 第七章 列表

列表环境有三种，分别是 itemize, enumerate, description。

## 7.1 基本用法

生成没有序号的列表

**Code effects**

```
\begin{itemize}
    \item 第一个
    \item 第二个
    \item 第三个
\begin{itemize}
    \item 第三个 +1
    \item 第三个 +2
\end{itemize}
\end{itemize}
```

- 第一个
- 第二个
- 第三个
  - 第三个 +1
  - 第三个 +2

生成有序号的列表

**Code effects**

```
\begin{enumerate}
    \item 第一个
    \item 第二个
    \item 第三个
    \begin{enumerate}
        \item 第三个 +1
        \item 第三个 +2
    \end{enumerate}
\end{enumerate}
```

1. 第一个
2. 第二个
3. 第三个
  - (a) 第三个 +1
  - (b) 第三个 +2

生成有标签描述的列表

**Code effects**

```
\begin{description}
    \item[itemize] 没有序号
    \item[enumerate] 有序号
    \begin{description}
        \item[罗马数字] 第一种
        \item[阿拉伯数字] ...
    \end{description}
    \item[description] 使用文字
\end{description}
```

**itemize** 没有序号

**enumerate** 有序号

**罗马数字** 第一种

**阿拉伯数字** ...

**description** 使用文字

## 7.2 自定义 Bullet

”Bullet” 是指各种 List 前的那个标记，中文译为”子弹”，但觉得还是保留英文比较好一些。一般来说，默认的已经够用，但如果需要重新定义，LATEX 同样支持高度可定制化的方式，另外，为了方便定制，可能需要使用enumitem宏包<sup>15</sup>：

### 7.2.1 自定义标签

如果只是临时更改，那么只需要在\item后添加参数即可，如下：

<sup>15</sup>除了该宏包外，还有easylist，enumerate等宏包可以完成，有需要可以自行了解

**Code effects**

```
\begin{itemize}
    \item[--] 第一行
    \item[This] 第二行 % 注意，虽然可以是文字，但缺少了 description
        中的加粗，这是这两个环境的区别。
    \item 第三行
\end{itemize}
```

– 第一行

This 第二行

• 第三行

如果是统一更改一个 itemize 环境，那么可以在环境开始添加参数，如下：

**Code effects**

```
\begin{itemize}[label=--]
    \item 第一行
    \item 第二行
\end{itemize}
```

– 第一行

– 第二行

如果是需要有一个经常使用的 itemize 环境，那么可以通过 enumitem 宏包直接定义一个，如定义一个todolist（待完成列表）：

**Code effects**

```
% \usepackage{enumitem}
% \usepackage{amssymb} \%square 方法支持
\newlist{todolist}{itemize}{2} %
→ \newlist{<name>}{{<type>}}{<max-depth>}
\setlist[todolist]{label=$\square$}

\begin{todolist}
\item 第一行
\item 第二行
\end{todolist}
```

第一行

第二行

上述的定义方法也可以合并为一个：

**Code effects**

```
\newlist{todolist2}{itemize}{2}
```

如果需要一个 Bullet 在多个环境里共用，那么可以新定义一个 item，如下：

**Code effects**

```
\newcommand{\checkitem}{\item[$\square$]}%
\begin{itemize}
\item 第一行
\checkitem 第二行
\end{itemize}
```

第一行

第二行

我曾经因为需要定义了一个 TOTOList，分享在这里：

**Code effects**

```
\begin{itemize}
    \item[$\square$]
        no checked
    \item[\rlap{\raisebox{0.3ex}{\hspace{0.4ex}\tiny
        \ding{52}}}$\square$]
        failed
    \item[\rlap{\raisebox{0.3ex}{\hspace{0.4ex}\scriptsize
        \ding{56}}}$\square$]
        checked
\end{itemize}
```

no checked

failed

checked

LATEX 中很难找到棱角分明的对号，只能用这种风格的折中一下。另外，关键在于 Bullet 如何用 LATEX 描述，如果需要很多使用，参考上面如何定义一个常用 item。

### 7.2.2 自定义序号

序号的生成与[14.4：计数器](#)和[3.1.2：数字](#)有关，如果需要重新定义序号方式，方法与自定义标签类似，在单个 item 上重新定义需要借助计数器的名称，如：

**Code effects**

```
\begin{enumerate}
    \item 第一行
    \stepcounter{enumi} % 如果指定了序号格式，需要使用之前手动对计数
    \rightarrow 增加 1
    \item[\alph{enumi}: ] 第二行
    \item 第三行
    \begin{enumerate}
        \stepcounter{enumii}
        \item[\roman{enumii}] 二级 +1
        \stepcounter{enumii}
        \item[\roman{enumii}] 二级 +2
    \end{enumerate}
\end{enumerate}
```

1. 第一行

b: 第二行

3. 第三行

i 二级 +1

ii 二级 +2

在 `enumerate` 环境上的定义:

**Code effects**

```
\begin{enumerate}[label=\roman*]
    \item 第一行
    \item 第二行
\end{enumerate}
\begin{enumerate}[label=\alph*--]
    \item 第一行
    \item 第二行
\end{enumerate}
```

i) 第一行

ii) 第二行

a- 第一行

b- 第二行

如果需要重新定义新的序号环境，方法同样与自定义标签类似：

**Code effects**

```
% \usepackage{enumitem}
\newlist{romanlist}{enumerate}{2}
\setlist[romanlist]{label=\roman*}
\begin{romanlist}
    \item 第一行
    \item 第二行
\end{romanlist}
```

i) 第一行

ii) 第二行

## 7.3 版面与布局

### 7.3.1 对齐与间隔

在使用enumitem宏包后，就可以很方便的设置各种间距，有如下的几种：

- 垂直间距
  - topsep
  - partopsep
  - parsep
  - itemsep
- 水平间距
  - leftmargin
  - rightmargin
  - listparindent
  - labelwidth
  - labelsep
  - itemindent

可以通过附加参数的方式来设置，如：

- `\begin{itemize}[topsep=0pt]`
- `\begin{itemize}[labelwidth=10mm]`

如果要应用间隔设置到所有 list，那么可以使用\setlist命令，如

**Code effects**

```
\newlist{romanlist}{enumerate}{2}
\setlist[romanlist]{label=\text{\textit{\roman*}}},leftmargin=1cm}
\begin{romanlist}
    \item 第一行
    \item 第二行
\end{romanlist}
```

i) 第一行

ii) 第二行

## 7.4 其他 itemize 布局

更多布局可以查看[Wiki-List\\_Structures](#)

# 第八章 表格

## 8.1 表格基础用法

单个的表格使用tabular环境，一个最简单的表格如下所示：

**Code effects**

```
\begin{tabular}{cc} %% 有两个字母表示有两列, c 表示居中, 还可以选择
    \rightarrow l 或者 r
    A&B\\ % \\ 表示换行, & 表示分割线
    C&D\\
\end{tabular}
```

A	B
C	D

表格列数需要在参数中表明，行数不限制，使用\backslash换行即可。

如果要添加边框（边框由分割线组成），对列添加分割线需要在参数中相应位置添加”|”，对行分割线则使用”\hline”或”\cline”

#### Code effects

```
\begin{tabular}{|c|c||} % 因为右侧有两道线，所以相应位置显示两条分割  
→ 线  
A&B\\\hline  
C&D\\ \cline{0-1} % 数字代表列数，从零开始，表示从 a-b  
E&F\\ \cline{0-0}  
G&H\\ \cline{1-1} % 边框可以任意重叠，但横向边框显示多条并不好控  
→ 制，因此不推荐使用多条分割线完成特殊需求  
I&J\\ \cline{0-0}\cline{1-1}  
\end{tabular}
```

A	B
C	D
E	F
G	H
I	J

## 8.2 表格合并

有时后，会遇到合并表格的需求，合并表格需要使用multirow这个库，并使用到\multicolumn和\multirow这两个方法，

其中合并多列比较简单，在要合并的位置填入相应的命令，随后减少相应的分隔符即可

## Code effects

```
\begin{tabular}{|c|c|c|c|c|}\hline 1.0&2.0&3.0&4.0&5.0\\\hline 6.0&\multicolumn{3}{c|}{\text{合并三列}}&7.0\\\hline 8.0&9.0&10.0&11.0&12.0\\\hline\end{tabular}
```

1.0	2.0	3.0	4.0	5.0
6.0	合并三列			7.0
8.0	9.0	10.0	11.0	12.0

合并多行同理，不过在相同列的位置需要空出来，并在设置分割线的时候将相应的列的位置空出来

## Code effects

```
\begin{tabular}{|c|c|c|}\hline 1.0&2.0&3.0\\\hline 4.0&\multicolumn{2}{c}{\text{合并三行}}&5.0\\\cline{1-1}\cline{3-3} 6.0&&7.0\\\cline{1-1}\cline{3-3} 8.0&&9.0\\\hline 10.0&11.0&12.0\\\hline\end{tabular}
```

1.0	2.0	3.0
4.0		5.0
6.0	合并三行	7.0
8.0		9.0
10.0	11.0	12.0

如果要同时合并多行多列，则需要对两个命令嵌套使用：

## Code effects

```
\begin{tabular}{|c|c|c|c|} \hline 1.0&2.0&3.0&4.0\\ \hline 5.0&\multicolumn{2}{c|}{\multirow{3}{*}{\text{合并三行两列}}}&6.0\\ \cline{1-1} \cline{4-4} 7.0&\multicolumn{2}{c|}{\text{}}&8.0\\ \cline{1-1} \cline{4-4} 9.0&\multicolumn{2}{c|}{\text{}}&10.0\\ \hline 11.0&12.0&13.0&14.0\\ \hline \end{tabular}
```

1.0	2.0	3.0	4.0
5.0			6.0
7.0	合并三行两列		8.0
9.0			10.0
11.0	12.0	13.0	14.0

### 8.3 表格列宽调整

表格如果不设置宽度，一列的字数如果太多，就有可能超出文档页面大小，变得很丑

1.0	2.0		3.0
1.0	2.0	长文字示例长文字示例长文字示例长文字示例长文字示例长文字示例长文字示例长文字	

因此，需要固定表格大小，从而更好的控制表格，在综合了各个库的命令和用法后，认为使用tabular环境中提供的参数 p 是最优的方式。

### 8.3.1 参数 p

tabular 的参数选项如下：

```
\begin{tabular}[pos]{cols}
...
\end{tabular}
```

其中 [pos] 大多数情况下可以忽略，而 {col} 的位置之前提到过可以填写 `clr` 表示居中、左对齐和右对齐。该位置还有一个可选参数为 `p{wth}`，用于指明相应位置的列占据多大的列宽，如：

#### Code effects

```
\begin{tabular}{|c|c|p{3cm}|c|}
\hline
1.0&2.0&3.0&4.0\\
\hline
1.0&2.0&长文字示例长文字示例长文字示例长文字示例长文字示例长文字
→ 示例长文字示例长文字示例长文字示例长文字示例长文字示例长文字
→ 示例&4.0\\
\hline
\end{tabular}
```

1.0	2.0	3.0	4.0
1.0	2.0	长文字示例长文字 示例长文字示例长 文字示例长文字示 例长文字示例长文 字示例长文字示例 长文字示例长文字 示例长文字示例长 文字示例	4.0

被指明了列宽的列会自动换行，这种方式可以一般情况下的列宽过长的问题。但设置了这种方式后该列是默认左对齐的，如果要设置居中或者右

对齐，需要利用该环境的附加命令。

### 8.3.2 tabular 的附加命令

tabular的参数中提供了以下附加命令，可以用于在相应列的每一个 cell 之前或者之后插入文字或命令，也即我们可以在文字前加入居中命令来实现当列的居中操作，如下：

## Code effects

```
\begin{tabular}{|c|c|}>{\centering}p{7cm}<{-insert}|c|}

\hline
1.0&2.0&3.0&4.0\\

\hline
1.0&2.0&长文字示例长文字示例长文字示例长文字示例长文字示例长文字
→ 示例长文字示例长文字示例长文字示例长文字示例长文字示例长文字
→ 示例&4.0\\

\hline

\end{tabular}
```

1.0	2.0	3.0-insert	4.0
1.0	2.0	长文字示例长文字示例长文字示例长文字 示例长文字示例长文字示例长文字示例长 文字示例长文字示例长文字示例长文字示 例长文字示例-long	4.0

上面的示例为在第三列的每个元素前加入居中命令，并在之后插入字符串”-insert”

## 8.4 橫置表格

横置表格可以使用adjustbox宏包中的adjustbox环境，指定旋转角度即可：

## Code effects

```
% \usepackage{adjustbox}
\begin{adjustbox}{angle=90}
\begin{tabular}{|c|c|}>{\centering p{7cm}<{-insert}|c|}
\hline
1.0&2.0&3.0&4.0\\
\hline
1.0&2.0&长文字示例长文字示例长文字示例长文字示例长文字示例长
    ↪ 文字示例长文字示例长文字示例长文字示例长文字示例长文字示
    ↪ 例长文字示例&4.0\\
\hline
\end{tabular}
\end{adjustbox}
```

1.0	2.0	3.0-insert	4.0
1.0	2.0	长文字示例长文字示例长文字示例长文字示例长文字示 示例长文字示例长文字示例长文字示例长文字示例长文字示 文字示例长文字示例长文字示例长文字示例长文字示例长文字示 例长文字示例示例-long	4.0

## 8.5 长表格

因为表格是一个盒子，因此无法分页，这样会导致长表格会挤在一页看不全，因此需要跨页表格的支持，搜索得到的方法有`longtable`、`supertabular`、`xtab`等，在对比后，我认为使用`longtable`相对来说更加方便，因此仅介绍使用`longtable`<sup>16</sup>。

```
\begin{center}
    % 注意，longtable 中添加居中需要多使用一个\arraybackslash 命令，否
    ↪ 则会报错

    ↪ \begin{longtable}{|l|l|}>{\centering\arraybackslash}p{0.4\columnwidth}|}%
        \hline%
        \multicolumn{3}{r}{Continued on Next Page}\hline%
        \hline%
    \endfoot% 此语句之前均为分页时的结尾行
        \hline%
        \multicolumn{3}{r}{Not Continued on Next Page}\hline%
        \hline%
    \endlastfoot% 此语句之前均为最后一页的结尾行
        \hline
        header 1&header 2&header 3\hline% 标题
        \hline
        Content1&Longer String\hline%
        Content1&Longer String\hline%
        Content1&Longer String\hline%
        Content1&Longer String\hline%
        ...
        Content1&Longer String\hline%
    \end{longtable}%
\end{center}
```

header 1	header 2	header 3
Content1	9	Longer String

Contained on Next Page

<sup>16</sup>注意，`longtable` 在使用中可能需要编译多次才能成型。

Containued on Next Page

Content1	9	Longer String
Not Contained on Next Page		

## 8.6 表格美化

对表格的美化，主要涉及列宽的调整和颜色的设置，列宽的调整先前的章节已经介绍，关于颜色的设置，参考[13.4.1：表格](#)

在表格下如果要设置边框和背景色，需要为 `xcolor` 添加 `table` 参数

```
\usepackage[table]{xcolor}
```

### Code effects

```
\begin{tabular}{lllll}
\toprule
\emph{name} & \emph{foo} && \midrule
Models & A & B & C & D \\
\rowcolor{blue!50} Model X & X1 & X2 & X3 & X4 \\
\rowcolor{green!50} Model Y & Y1 & Y2 & Y3 & Y4 \\
\hline
\end{tabular}
```

name	foo			
Models	A	B	C	D
Model X	X1	X2	X3	X4
Model Y	Y1	Y2	Y3	Y4

如果需要单独设置一列、边框、单个单元格的颜色，可以查阅[colortbl](#)宏包。

## 8.7 添加表格说明

需要使用`table`环境，并使用`\caption`命令，具体可以参考11.1.1：添加Float说明对`float`环境的说明。

Code effects

```
\begin{center}
\begin{table}[H]
\begin{tabular}{|c|c|}
\hline
1.0&2.0\\
\hline
\end{tabular}
\caption{表格说明}
\end{table}
\end{center}
```

1.0	2.0
-----	-----

表 6: 表格说明

## 8.8 自动生成表格工具

自动生成表格的工具网上有很多开源实现，推荐以下几种：

### 8.8.1 Tables Generator

网站链接：[Tables Generator](#)

该网站提供了一个通用的表格转换工具，在其提供的表格环境中填写好后，可以转换为多种格式下的表格代码，其中包括了`LATEX`。

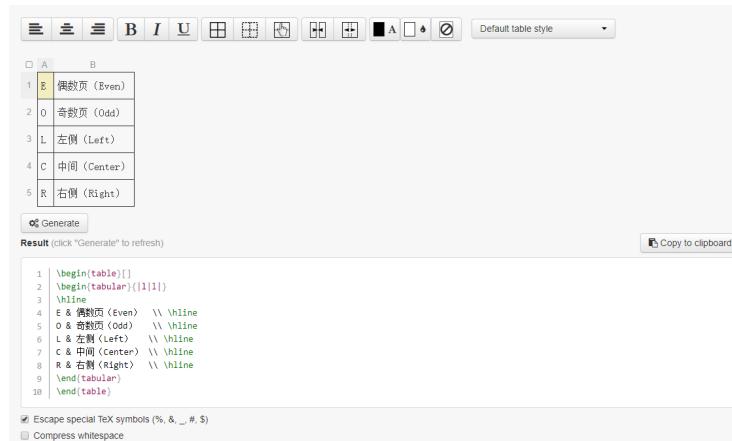


图 4: Table-Generator 网站截图

### 8.8.2 LatexTool

网站链接: [LatexTool](#)

是我写的一个 Python 工具, 因为要考虑列宽, 合并表格等, 在网上找的众多工具并不能符合我的需求, 于是用 Python 写了一个, 目前基本能够实现我的需求, 使用方法也比较简单。

#### Code

```
1 % 直接使用 pip 安装: pip install x2t
2 % 可以直接使用控制台命令: x2t test.xlsx test.xls
3 from LatexTool.ast import tabel
4
5 if __name__ == '__main__':
6     # tab = tabel.Tabel("../test.xlsx",center=False,fill=False)
7     tab = tabel.Tabel("../test.xlsx")
8     print(tab.to_tex())
```

完美支持合并多行多列:

## Code effects

```
\begin{tabular}{|c|c|c|c|}\hline&合并三列&&0.0\\\hline C&D&H&1.0\\\hline&合并两行&\multicolumn{2}{c} {\multirow{3}{*}{\multicolumn{3}{c}{\cline{2-2}\&K\&\multicolumn{2}{c}{\{}{\}}\\ \cline{1-2}\&M\&N\&\multicolumn{2}{c}{\{}{\}}\\\hline\end{tabular}
```

合并三列			0.0
C	D	H	1.0
合并两行	F	合并三行两列	
	K		
M	N		

也可以将大小填充到整个页面，并设置居中：

## Code effects

```
\begin{center}
% \newlength\tablewidth % if haven't define the length
\setlength\tablewidth{\dimexpr (\textwidth -8\tabcolsep)}
\begin{table}[H]
\begin{tabular}{|p{0.29\tablewidth}<{\centering}|p{0.07\tablewidth}<{\centering}|p{0.43\tablewidth}<{\centering}|p{0.21\tablewidth}<{\centering}|}
\hline
\multicolumn{3}{|c|}{\合并三列}&0.0\\
\hline
C&D&H&1.0\\
\hline
\multirow{2}{*}{\合并两行} & \multicolumn{2}{c|}{\合并三行} \\
\hline
\cline{2-2}
&K&\multicolumn{2}{c|}{\{}\\
\cline{1-2}
&N&\multicolumn{2}{c|}{\{}\\
\hline
\end{tabular}
\end{table}
\end{center}
```

合并三列			0.0
C	D	H	1.0
合并两行	F	合并三行两列	
	K		
M	N		

## 第九章 图片

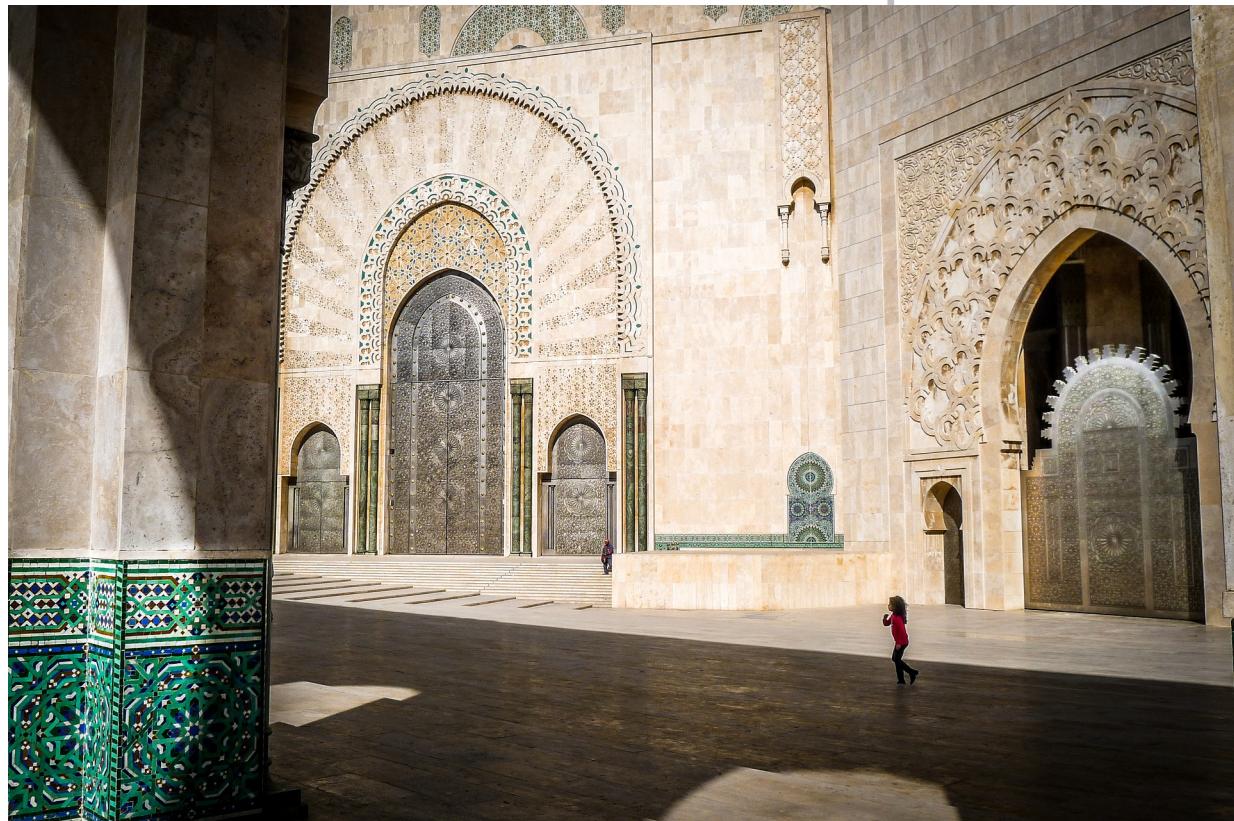
插入图片使用的命令是`\includegraphics[参数]{图片路径}`，在添加包支持后，可以支持 eps、png、jpg、pdf 等多种格式，推荐使用 png 和 eps 两种格式的图片

本手册用于说明的图片均来源于免费素材网站[pixabay](#)，之后不再说明。

## 9.1 基本使用

Code effects

```
\includegraphics{contents/fig/mosque.jpg}
```



如上图所示，有时会遇到图片过大的情况，因此需要加入参数调整图片的大小，因为`\columnwidth`是去掉了各种页边距后并考虑了分栏时的当前页面宽度，因此一般以该宽度作为基准调整图片大小：

**Code effects**

```
\includegraphics[width=0.5\columnwidth]{contents/fig/mosque.jpg} %  
↪ 具体大小根据图片宽高比调整小数即可
```



如果是长条形图片，则可以使用\textheight调整图片高度的形式来适应页面。

**Code effects**

```
\includegraphics[height=0.4\textheight]{contents/fig/ocean.jpg}
```



注意，一般不同时调整图片长和宽，因为会导致图片失真。同时该命令

还有scale (缩放) 参数 (直接指定数字, 不需要单位), 但也一般不使用, 因为在图片大小不统一的情况下, 不便控制缩放比例。

如果需要旋转图片, 则使用angle参数, 直接指定数字表示度数

#### Code effects

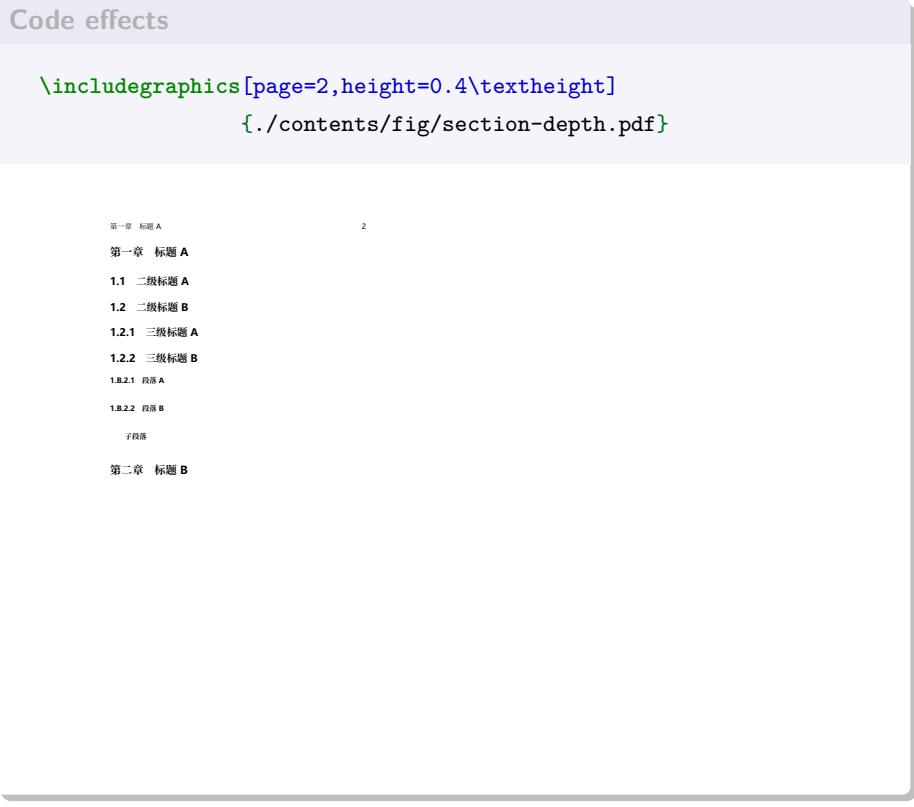
```
\includegraphics[width=0.3\columnwidth,angle=45]
{contents/fig/mosque.jpg}
\includegraphics[width=0.3\columnwidth,angle=-45]
{contents/fig/mosque.jpg}
```



如果是添加 PDF 文档, 可能会存在多页的问题, 需要使用page参数:

**Code effects**

```
\includegraphics[page=2,height=0.4\textheight]  
{./contents/fig/section-depth.pdf}
```



第一部分 标题 A  
第一章 标题 A  
1.1 二级标题 A  
1.2 二级标题 B  
1.2.1 三级标题 A  
1.2.2 三级标题 B  
1.2.2.1 四级 A  
1.2.2.2 四级 B  
子目录  
第二章 标题 B

如果要裁剪，则需要使用clip和trim参数，clip 参数声明该图片需要裁剪（如果不声明则 trim 参数无效），trim 则指明裁剪的边界：

**Code effects**

```
% 参数用法: trim = <left> <lower> <right> <upper>
\includegraphics[clip,trim =2cm {0.4\paperheight} 2cm
→ {0.4\paperheight},width=0.3\columnwidth]{contents/fig/mosque.jpg} \\
\hspace{0.3\columnwidth}\\
\includegraphics[width=0.3\columnwidth]{contents/fig/mosque.jpg}
```



注意，如果是直接指明长度，则不需要加 {...} 将长度包裹起来，如果是带有运算性质的长度，那么则需要用 {...} 将长度包裹起来，否则无法识别。

## 9.2 指明图片路径

\includegraphics命令支持相对路径（ $\text{\LaTeX}$ 编译时路径）和绝对路径两种路径，但即使是相对路径，有时也会因为更改而导致目录失效，因此可以以下命令指明图片目录，从而在插入图片时候只需要添加图片名字即可。

```
\graphicspath{{./}{./contents/}{./contents/fig/}}% 设置图片可能存在的
→ 路径
```

设置后  $\text{\LaTeX}$  编译时会自动寻找目录下的图片。

### 9.3 图片优先级读取

如果不加扩展名，编译时会依次寻找同名且为图片格式后缀的文件，在序言区声明以下内容可以更改其优先级。

```
\DeclareGraphicsExtensions{.eps,.png,.jpg}%
← 用
```

### 9.4 添加图片说明

参考11.2：Figure一节对Figure环境的说明。

### 9.5 Tikz 绘图

LATEX 内内置了绘图方法，可以直接绘制出矢量图，[TikZ and PGF examples](#)网站上提供了大量精美（或复杂）的 Tikz 绘图示例：

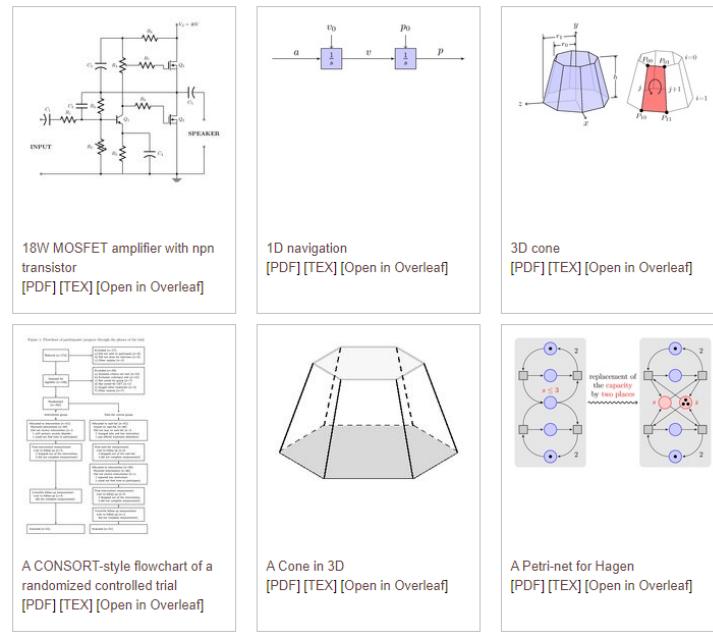
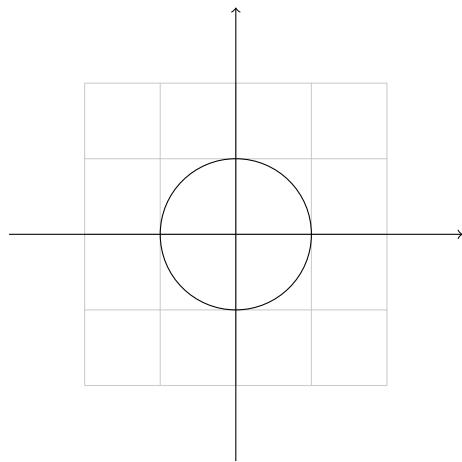


图 5: TikZ 和 PGF 例子 网站截图

使用 Tikz 的一个示例如下：

**Code effects**

```
\begin{tikzpicture}
    \draw[step=1,color=gray!40] (-2,-2) grid (2,2);
    \draw[->] (-3,0) -- (3,0);
    \draw[->] (0,-3) -- (0,3);
    \draw (0,0) circle (1);
\end{tikzpicture}
```



理论上 Tikz 可以绘制出你想要的任何图片，但是实际上由于该宏包过于复杂，除了技术狂魔外我不推荐直接使用内置的 Tikz 宏包来绘制矢量图... 而是通过其他的手段（如 Adobe Illustrator, Matplotlib）导出矢量图后以图片的形式添加，因此这里仅仅简单的介绍一下 Tikz，具体的使用略。

## 第十章 公式环境

行内公式、行间公式、下标显示、label 添加、公式编辑工具

公式环境也叫做数学环境，是用来处理各种各样的数学符号的，并且支持多种展示属性，一般情况下，使用`amsmath`包即可

## 10.1 行内公式

行内公式可以使用`math`环境, `\(..\)`或者`$...$`三种方法:

### Code effects

```
% \usepackage{amsmath}
正常文本嵌套行内公式 A: \begin{math}f_i(x)=a+b\end{math}。 \\
正常文本嵌套行内公式 B: \left(f^2(x)=a*b\right)。 \\
正常文本嵌套行内公式 C: g(l;\theta)=a-b。
```

正常文本嵌套行内公式 A:  $f_i(x) = a + b$ 。

正常文本嵌套行内公式 B:  $f^2(x) = a * b$ 。

正常文本嵌套行内公式 C:  $g(l; \theta) = a - b$ 。

### 10.1.1 数学环境中的普通文本

如果要在公式内添加解释文本，英文可以直接支持，但是会转换成公式默认字体（itali 斜体），并且不支持中文输入，因此如果要在公式环境内添加解释，需要使用`amsmath`包中的`\text`命令：

### Code effects

```
$g(l;\theta)=a-b,\text{正常文本 normal text}, 正常文本 normal
\text{text}$
```

$g(l; \theta) = a - b$ , 正常文本 normal text, normaltext

## 10.2 行间公式

首先一定要注意，**行间公式不能有空行！** 所以每一条公式必须紧靠着。如果要换行，需要使用`\\\`声明（仅在**多行公式**环境中有效）。

### 10.2.1 单行行间公式

单行行间公式不支持换行符（会忽略换行符）。

如果不需要对行间公式进行编号，可以直接使用`\[...]`或`$$...$$`

#### Code effects

公式前文本

```
\[
    x = a+b; \\
    y = c+d;
\]
```

公式后文本

公式前文本

$$x = a + b; y = c + d;$$

公式后文本

如果需要对一整个行间公式进行编号，可以使用equation环境：

#### Code effects

```
\begin{equation}
    x = a+b; \\
    y = c+d;
\end{equation}
```

$$x = a + b; y = c + d; \quad (1)$$

### 10.2.2 多行行间公式

如果要分段定义编号，可以使用eqnarray环境，会对每一部分单独编号，  
如果不需要编号，在当前行添加\nonumber：

**Code effects**

```
\begin{eqnarray}
x = a+b; \\
y = c+d; \nonumber \\
z = e+f;
\end{eqnarray}
```

$$x = a + b; \tag{2}$$

$$y = c + d;$$

$$z = e + f; \tag{3}$$

如果要对多个公式单独编号，但有对其中几个公式完成组编号，可以在公式中嵌套split环境（注意在环境结束后添加换行符）：

## Code effects

```
\begin{eqnarray}
\begin{split}
a = 1; \\
b = 2; \\
c = 3; \\
\end{split} \\
x = a+b; \\
y = c+d; \nonumber \\
z = e+f;
\end{eqnarray}
```

$$a = 1; \quad (4)$$

$$b = 2; \quad (4)$$

$$c = 3;$$

$$x = a + b; \quad (5)$$

$$y = c + d;$$

$$z = e + f; \quad (6)$$

如果要重新开始编号或从指定数字开始编号, 使用 \setcounter{equation}{1} 方法。

## Code effects

```
\begin{eqnarray}
\setcounter{equation}{2}
\begin{split}
a = 1; \\
b = 2; \\
c = 3; \\
\end{split} \\
\setcounter{equation}{7}
x = a+b; \\
y = c+d; \\
z = e+f;
\end{eqnarray}
```

$$a = 1;$$

$$b = 2; \tag{2}$$

$$c = 3;$$

$$x = a + b; \tag{7}$$

$$y = c + d; \tag{8}$$

$$z = e + f; \tag{9}$$

如果需要对齐，则使用align环境，并在要对齐的地方使用 & 声明：

## Code effects

```
\begin{align}
&f(x) = a+b+c; \\
g(x) = a; \\
\end{align}
```

$$f(x) = a + b + c; \tag{10}$$

$$g(x) = a; \tag{11}$$

同时，在`eqnarray`环境中可以直接使用`align`的方式进行对齐

## 10.3 公式编辑工具推荐

### 10.3.1 Mathpix

网站链接：[Mathpix](#)。

如果有图片类型的公式，可以用该软件截图生成 LATEX 公式，多平台支持，且准确率很高，即使多行公式也能有很高的准确率，同时免费用户每天有 50 次截图的限额，非常划算。

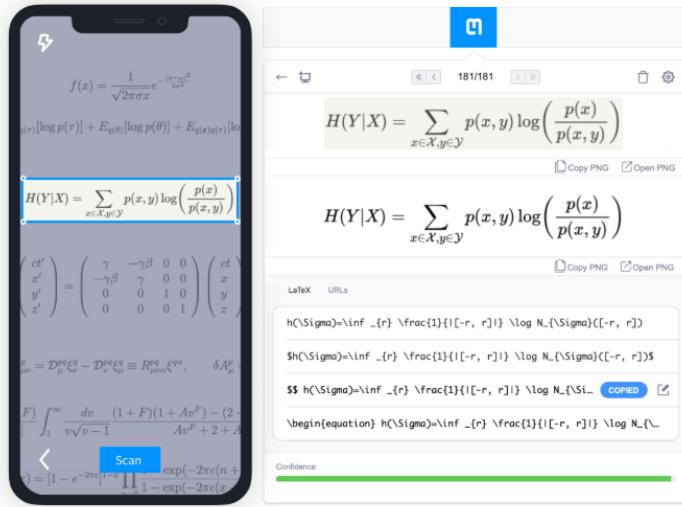


图 6: Mathpix 官网示例

### 10.3.2 在线 LATEX 公式编辑器

网站链接：[eqneditor](#)

网站丑了点，但是能够对公式进行所见即所得的编辑，比较友好，目前暂时没有找到比它更好的替代方案，不过如果配置时使用的是 Tex 专门的 IDE，其公式编辑应该也比较方便（不过可能不支持所见即所得）

## 第十一章 Float

### 11.1 Float 基本介绍

Float是一类环境，指的是放置位置不固定的内容，如表格或图片，有时如果强行插入在当前段落，可能会造成空白，因此可以适当的在后续的几页中放置，但如果靠自己调整，一定是十分麻烦，因此便 Float 环境，它可以自动的寻找合适的位置插入。一般来说，Float 环境支持五种参数，如下：

h	将 Float 尽可能的放在当前位置 (here)，注意是尽可能，如果放不下，可能还是会放在别的位置
t	放在一页的顶部 (Top)
b	放在一页的底部 (Bottom)
p	放在一个特定的放 Float 的位置
!	重写 L <sup>A</sup> T <sub>E</sub> X 内置的对于“好”的 Float 位置的偏好。
H	将 Float 环境准确的放到该位置，相当于只使用 Float 提供的其他特性，不使用 Float 的“浮动”特性，使用该参数需要使用 <u>float</u> 包。

表 7: Float 的参数

在后续的使用中，为了方便演示，一律使用H参数，其他参数的效果请读者自行试验。

#### 11.1.1 添加 Float 说明

通过在 Float 环境中使用\caption{描述}命令，可以为该 Float 添加描述，如图：

```
\begin{figure}[H]
    \includegraphics[width=0.5\columnwidth]
        {contents/fig/mosque.jpg}
    \caption{caption 示例}
\end{figure}
```



图 7: caption 示例

如果要居中，则在 Float 环境中添加 \centering 即可（也可以在 Float 外或者 Float 内嵌套 center 环境。）

```
\begin{figure}[H]
    \centering
    \includegraphics[width=0.5\columnwidth]{contents/fig/mosque.jpg}
    \caption{居中示例}
\end{figure}
```



图 8: 居中示例

### 11.1.2 更改 Float 说明格式

默认中文下, LATEX 会将表格编号为表 n., 将图片编号为图 n., 有时候我们可能有重新定制显示的需求, 这实际上需要重新设置 caption, 具体如何设置请参考该宏包。

## 11.2 Figure

插入图片的基本使用可以参考[第九章:图片](#)一节, 这里主要描述使用Figure环境实现的各种图片的排版操作。

### 11.2.1 插入单张图片

在上一节已经有所演示，只需要用figure环境包裹\includegraphics命令即可。

### 11.2.2 插入多个图片

插入多个图片可以直接在figure环境中插入多个\includegraphics命令，但有时候需要对多个图片分别描述，因此需要使用subfigure环境，如下：

**Code effects**

```
\begin{figure}[H]
    \subfigure[图片 A]{ \label{subfig:show-a} % 标签永远都是可选项,
        \rightarrow 需要引用的时候才需要添加
        \includegraphics[width=0.4\columnwidth]{mosque.jpg}
        \hspace{0.05\columnwidth} % 声明水平间距, 可以不添加, 或者根
        \rightarrow 据实际大小灵活更改
    }
    \subfigure[图片 B]{%
        \includegraphics[width=0.4\columnwidth]{mosque.jpg}
    }
    \caption{子图演示}
    \label{fig:sub-figure-show}
\end{figure}
% 如果需要居中, 添加 center 环境或者使用\centering 命令即可
```



(a) 图片 A



(b) 图片 B

图 9: 子图演示

### 11.2.3 列出图片目录

有时会有单独将所有的图片列出目录的需求, 此时使用 \listoffigures 命令。

一般是在文章目录后列出图片目录和表格目录。

## 11.3 Table

### 11.3.1 列出表格目录

有时会有单独将所有的表格列出目录的需求，此时使用`\listoftables`命令。

一般是在文章目录后列出图片目录和表格目录。

## 第十二章 注释与引用

### 12.1 脚注

#### 12.1.1 基本使用

脚注使用`\footnote{内容}`命令：

Code effects

被脚注的文本`\footnote{脚注内容}`

被脚注的文本<sup>a</sup>

<sup>a</sup>脚注内容

#### 12.1.2 更改标记风格

需要重新定义`\thefootnote`来实现更改标记风格：

被脚注的文本`\footnote{脚注内容} \\ \renewcommand{\thefootnote}{\roman{footnote}}`  
被脚注的文本`\footnote{脚注内容}`

被脚注的文本<sup>17</sup>

被脚注的文本<sup>xviii</sup>

关于各种风格的数字显示方式，查看

### 12.1.3 重新计数脚注标记

如果只是要指定某个脚注的下标，可以直接通过添加参数的方式：

Code effects

被脚注的文本 \footnote[12]{脚注内容}

被脚注的文本<sup>l</sup>

'脚注内容

脚注标记的计数也是用 LATEX 中的计数器来计数的，名字是`footnote`，因此要重新计数，只需要重置计数器<sup>xix</sup>即可：

被脚注的文本 \footnote{脚注内容} \\ \setcounter{footnote}{5}

被脚注的文本 \footnote{脚注内容}

被脚注的文本<sup>xx</sup>

被脚注的文本<sup>vi</sup>

<sup>17</sup>脚注内容

<sup>xviii</sup>脚注内容

<sup>xix</sup>关于计数器，可以参考14.4：计数器

<sup>xx</sup>脚注内容

<sup>vi</sup>脚注内容

## 12.2 边注

### 12.2.1 基本使用

边注是指在书两侧的注释，其在某种程度上比脚注更方便一些，因为边注直接显示在页面的一边，读者不需要将视线移动太多就能看到相应的注释。就像这样。

最简单的边注，使用`\marginpar{left}{right}`命令。如果是单侧布局（即不分奇偶页），那么边注仅显示在右侧，如果是双侧布局，则会显示在相应的外侧。如果是双栏布局，那么就会显示在靠近的一侧。

### 12.2.2 marginnote 宏包

内置的这两种命令对于一些环境的支持不是很好，比如显示在公式的一侧，或者是显示在脚注的一侧，如果原生命令无法满足需求，那么就可以使用扩展宏包`marginnote`。

首先使用 marginnote 宏包后，`geometry`就增加了可以控制边注大小和间隔的参数，如下：

```
\usepackage [top=Bcm, bottom=Hcm, outer=Ccm, inner=Acm,
heightrounded,
marginparwidth=Ecm, marginparsep=Dcm]{geometry}
```

在设置好边距后，如果要添加边注，使用`\marginnote`命令：

#### Code effects

这里是一段难懂的命令`\marginnote{这是一段解释的文字。}`。

这里是一段难懂的命令。

这是一段解释的文字。

上面使用的命令如果更换为 marginpar，在演示环境中会报错，这也是为什么推荐使用 marginnote 的原因。

## 12.3 标签

LATEX 支持在任意位置添加标签，从而可以很方便的建立链接，如：

### Code effects

如何实现一段标签可以查看该示例：`\label{ex: 标签示例}`

如何实现一段标签可以查看该示例：

要引用只需要使用内置命令`\ref`即可：

### Code effects

如何使用引用可以参考`\ref{ex: 标签示例}`节的介绍。

如何使用引用可以参考12.3节的介绍。

### Code effects

我们建议初学者浏览`\pageref{sub: 快速入门}`页来查看如何入门`\LaTeX{}`

我们建议初学者浏览15页来查看如何入门 LATEX

注意带有引用的位置均需要编译两次来完成，第一次编译只会形成问号，这是正常的。

### 12.3.1 其他的引用需求

`ref` 命令只会显示相应位置的节号，而 `label` 命令实际上只是建立一种链接关系，因此，如果要更改引用部分的文字，我们可以使用`hyperref`宏包中的`\hyperref`命令：

从实现中可以看到标签的定义支持中文和一些标点，另外最好遵循某种规范，方便寻找

**Code effects**

```
% \usepackage{hyperref}  
这里有一个\hyperref[ex: 标签示例]{标签示例}
```

这里有一个[标签示例](#)

注意如果链接到一个 URL，那么使用的是`\href`命令，而且两者的参数填写方法也不一样：

**Code effects**

```
关于 hyperref 宏包的具体参数，可以参考其文档，或者查看  
\href{https://en.wikibooks.org/wiki/LaTeX/Hyperlinks}  
{Wiki-Hyperlinks}
```

关于 hyperref 宏包的具体参数，可以参考其文档，或者查看 [Wiki-Hyperlinks](#)

另外，nameref宏包提供了显示章节号的命令`\nameref`：

**Code effects**

```
% \usepackage{nameref}  
在\nameref[ex: 标签示例]有一个示例
```

在[标签](#)有一个示例

## 12.4 参考文献

参考文献在 L<sup>A</sup>T<sub>E</sub>X 中有多种管理方式，并且均可以被很方便的引用。

### 12.4.1 直接引入

如果参考文献的数量比较少，不需要管理，那么可以直接使用 L<sup>A</sup>T<sub>E</sub>X 的内置环境`thebibliography`，该环境会直接在相应位置生成参考文献列表：

**Code effects**

```
\begin{thebibliography}{9}
\bibitem{lamp99}
  Leslie Lamport,
  \textit{\LaTeX: a document preparation system},
  Addison Wesley, Massachusetts,
  2nd edition,
  1994.

\bibitem{SMOalgo}
  Platt, John C,
  \textit{Sequential Minimal Optimization: A Fast Algorithm
    for Training Support Vector Machines}

\end{thebibliography}

\begin{thebibliography}{1}
\bibitem{tmc}
  Petersen, Pedersen, \textit{『The Matrix Cookbook』}
\end{thebibliography}
```

## 参考文献

- [1] Leslie Lamport, *\LaTeX: a document preparation system*, Addison Wesley, Massachusetts, 2nd edition, 1994.
- [2] Platt, John C, *Sequential Minimal Optimization: A Fast Algorithm for Training Support Vector Machines*

## 参考文献

- [1] Petersen, Pedersen, *『The Matrix Cookbook』*

上述的代码中，环境后的参数为最大标签数（必填项），但是要注意的是，其中的参数不是数字大小，而是数字的位数，如果是一位数字，那么最大标签数是 9；如果是两位，那么最大标签数是 99（即使你所填的数字是 59）。

每一个参考文献，都用一个bibitem来标识，bibitem之后到下一个bibitem之前的内容均属于该 bibitem。

如果要引用其中的某一个参考文献，使用cite命令<sup>vii</sup>：

#### Code effects

在这时候，有人发现了求解 SVM 的更快、更有效率的算法 \cite{SMOalgo}

在这时候，有人发现了求解 SVM 的更快、更有效率的算法 [2]

使用这种方法，跟目录等交叉引用项一样，需要用相应的编译器编译两次。

### 12.4.2 bibtex 引入

直接引入虽然方便，但是当参考文献很多的时候，会很不方便，因为一般情况下，没有引用到的文献是不需要添加在参考文献中的，而且不同的组织，其参考文献如何排序都是有标准的，这些如果我们如果使用直接引入的方法，就必须全都考虑，就会很麻烦。由此，我们可以使用bibtex来帮助我们管理参考文献和辅助引用。

bibtex基于.bib文件，该文件可以被当做是数据库，可以把我们使用过的所有的参考文献都按照 bibtex 的格式放进去。其中，每一篇参考文献的格式均如下所示：

```
@article{AbedonHymanThomas2003,
    author = "Abedon, S. T. and Hyman, P. and Thomas, C.",
    year = "2003",
    title = "Experimental examination of bacteriophage latent-period
             evolution as a response to bacterial availability",
    journal = "Applied and Environmental Microbiology",
    pages = "7499--7506",
    ...
}
```

<sup>vii</sup>标签能够具有颜色和链接属性是因为 hyperref 库

其中第一个参数是该引用的标识，而针对不同的引用（论文、期刊、书籍、网页... 等），只需要按需填写不同的参数即可，如引用网页的示例格式如下：

```
@misc{website:fermentas-lambda,
    author = "Fermentas Inc.",
    title = "Phage Lambda: description \& restriction map",
    month = "November",
    year = "2008",
    url =
        \url{http://www.fermentas.com/techinfo/nucleicacids/maplambda.htm"
}
```

上述的内容可以全部存放到一个.bib 文件中，也可以分门别类存放到多个.bib 文件中，均可，只需要最后在使用的时候，在需要显式参考文献列表的位置添加如下的两行代码即可：

```
\bibliographystyle{plain}
\bibliography{bf1, bf2, ..., bf3}
```

其中bibliographystyle命令是表明参考文献展示的格式，包括排序方式，展示规范等，共有 8 种，分别为：

plain	按字母的顺序排列，比较次序为作者、年度和标题.
unsrt	样式同 plain，只是按照引用的先后排序.
alpha	用作者名首字母 + 年份后两位作标号，以字母顺序排序.
abbrv	类似 plain，将月份全拼改为缩写，更显紧凑.
ieeetr	国际电气电子工程师协会期刊样式.
acm	美国计算机学会期刊样式.
siam	美国工业和应用数学学会期刊样式.
apalike	美国心理学学会期刊样式.

表 8: 参考文献的显示格式

第二行 `bibliography` 命令表示在这里显示参考文献列表，参数填写参考文献都在哪些文件里，可多不可少。在具体编译完成后，相应位置只会显示文章中引用过的参考文献，.bib 文件中其他的参考文献则不会显示。

这种方式在文献的管理上确实比直接引用要好很多，但是也有一个非常麻烦的地方，那就是要用这种方式编译文件，需要编译四次，以使用 xelatex，编译 main.tex 为例，四次编译依次为：

#### Code

```
1 xelatex main.tex
2 bibtex main.aux
3 xelatex main.tex
4 xelatex main.tex
```

注意第二次的参数是 .aux 文件而不是 .tex 文件，是在第一次编译后生成的，bibtex 需要使用 .aux 文件来获取 .tex 文件中引用了哪些文献，从而生成相应的 .bb1 文件。.bb1 文件的内容其实就是直接由 bibtex 从 .bib 文件中排好格式并且挑选出来相应文献后的 `thebibliography` 环境，随后两次编译则是编译交叉引用的正常步骤。

回头来看 bibtex 和 .bib，它们实质上是依托于 tex 系统而形成的一套文献管理软件，其通过借助 .tex 文件编译后形成的引用信息来生成 LATEX 支持的参考文献环境。

## 第十三章 颜色

### 13.1 引入

如果想要更好的定义、使用和设置颜色，推荐使用

```
\usepackage{xcolor}
```

xcolor 包中定义了一些基本的颜色，有：

black,blue, brown,cyan,darkgray,gray,green,lightgray,lime,magenta,olive  
orange,pink,purple,red,teal,violet, ,yellow

另外 xcolor 中还定义了 dvips 中的 68 种标准色，如果要使用，需要在引入包时添加参数：

```
\usepackage[dvipsnames]{xcolor}
```

## 13.2 使用

正如之前所使，几乎在任何一个可以更换颜色的地方，都可以更换相应位置的颜色。比如，**改变文字的颜色**：

### Code effects

```
{\color{red}这里放文字}\\
\textcolor{red}{这里放文字}% 两者在设置文字颜色时等价
```

这里放文字  
这里放文字

在大多数情况下，一些库或者自己生成的边框等，都可以通过设置其颜色属性或在外侧包裹 color 命令来实现变更颜色的方法。

## 13.3 定义

如果 xcolor 自带的颜色不足以使用，xcolor 还提供了灵活的定义新颜色的方法，示例如下：

```
\definecolor{light-gray}{gray}{0.95} % 定义灰度
\definecolor{orange}{rgb}{1,0.5,0} % 定义普通的 rgb 颜色（色值归一化为
↪ [0-1])
\definecolor{orange}{RGB}{255,127,0} % 定义普通的 rgb 颜色（色值范围
↪ 为 0-255）
\definecolor{orange}{HTML}{FF7F00} % 使用 HTML 设计时常用的方式，颜色
↪ 由三个 16 进制数值组成
\definecolor{orange}{cmyk}{0,0.5,1,0} % 使用印刷的色域，输入 cmyk 值
```

定义的颜色名字和 xcolor 自带的颜色名字具有相同的效果。

另外，在一些情况下，颜色还可以通过这样的一种语法来定义：

```
% 默认底色是白色
\colorlet{wblue}{blue!20} % 表示 20% 的蓝色 +80% 的白色
\colorlet{bblue}{blue!20!black} % 表示 20% 的蓝色 +80% 的黑色
\colorlet{bbg}{blue!20!black!30!green} % 表示 20% 的蓝色 +30% 的黑
↪ 色 +50% 的绿色
```

在很多无法包裹 color 命令的情况下（如环境参数设置），大多是采用这种混合颜色（color mixes）的方式来设置颜色，如使用 tcolorbox 定义一个盒子：

#### Code effects

```
\begin{tcolorbox}[colframe=blue!50!, colback=blue!20!]
    盒子内部
\end{tcolorbox}
```

盒子内部

另外，如果要创建颜色的别名，也可以使用 \colorlet：

```
\colorlet{mycolor}{someothercolor}
```

一般来说，新定义的颜色可以放到任何位置，但是为了规范，一般都倾

向于将颜色统一定义在一个位置。

## 13.4 其他环境下的颜色设置

### 13.4.1 表格

关于表格的颜色设置，参考[8.6：表格美化](#)的说明

## 13.5 超链接与引用

超链接与引用的颜色，可以很方便的通过[hyperref](#)这个库来设置，只需要在导入库的时候定义参数即可，如：

```
\usepackage[colorlinks,  
linkcolor=black,  
urlcolor=blue,  
anchorcolor=blue,  
citecolor=green]{hyperref}
```

#### Code effects

```
\href{https://en.wikibooks.org/wiki/TeX/Hyperlinks}{Wiki-Hyperlinks}
```

Wiki-Hyperlinks

关于设置的颜色对应哪个区域，以及更多的参数，可以参考[Wiki-Hyperlinks](#)

## 13.6 分割线



## 13.7 颜色相关工具

### 13.7.1 LatexColor

网站链接：[LatexColor](#)

点击看好的颜色直接复制定义颜色的语句，比较方便。

Swatch	Color name♦	Hex Triplet ♦ R G B	LATEX
	Air Force blue	#5D8AA8	\definecolor{airforceblue}{rgb}{0.36, 0.54, 0.66}
	Alice blue	#F0F8FF	\definecolor{aliceblue}{rgb}{0.94, 0.97, 1.0}
	Alizarin	#E32636	\definecolor{alizarin}{rgb}{0.82, 0.1, 0.26}
	Almond	#EFDECD	\definecolor{almond}{rgb}{0.94, 0.87, 0.8}
	Amaranth	#E52B50	\definecolor{amaranth}{rgb}{0.9, 0.17, 0.31}
	Amber	#FFBF00	\definecolor{amber}{rgb}{1.0, 0.75, 0.0}

图 10: latexcolor 网站截图

## 第十四章 命令和环境

LATEX 中除了正常的文字外，就只有命令和环境，一切均有命令和环境组成。甚至，环境也只是对命令的一种封装。完全可以将环境理解为互相联系的一组命令。

## 14.1 命令

### 14.1.1 命令的基本使用

LATEX 中大部分结构均为命令，调用命令的一般格式如下：

```
\command[]{}{}...{}
```

其中 command 是命令名，使用反斜杠表示<sup>viii</sup>，[]中的内容是可选项，一般省略，会被默认值替代，{}中的内容是必选项，如果加大括号则为括号中的内容，否则则为命令后直到域边界<sup>ix</sup>的内容。

#### Code effects

```
\textbf 粗体字
```

粗体字

### 14.1.2 命令的定义

如果要定义一个命令，使用\newcommand命令<sup>x</sup>，如下：

#### Code effects

```
\newcommand{\lmd}{$\lambda$}\lmd\lmd{}\\lmd{Try it}
```

λ

λ

λ

Try it

其中 \lmd 定义了一条名字为 lmd 的命令，第二个大括号中表示命令所代表的内容，在这里是一个符号加换行符。这种方式常用于代替某变量或符

<sup>viii</sup>所以如果要在 LATEX 中表示反斜杠需要使用 \textbackslash

<sup>ix</sup>该命令属于内置命令

号，方便统一修改。注意不能够使用该命令定义一个已经存在的命令。<sup>xi</sup>

如果要定义带有参数的变量，则需要添加参数选项：

#### Code effects

```
\newcommand{\myy}[1]{$y_{\#1}$}
\myy{1}, \myy{2}, \myy{3}, \myy{n}
```

$y_1, y_2, y_3, y_n$

其中<sub>[1]</sub>表示该命令可以有一个参数，<sub>#1</sub>表示参数放置的位置。

LATEX 命令最多支持九个参数，只需要更改参数数量，并在命令内容里添加相应的参数位置即可<sup>xii</sup>

LATEX 还支持默认值，但只允许设置第一个参数有默认值，如下：

#### Code effects

```
\newcommand{\eqs}[3][2]{${(\#3+\#2)^{\#1}, \#1$}\}
\eqs{a}{b}
\eqs[5]{x}{y}
\eqs{a}{b}{c}
```

$(b + a)^2, 2$

$(y + x)^5, 5$

$(b + a)^2, 2$

c

其中<sub>[3]</sub>表示该方法有三个参数，<sub>[2]</sub>表示该方法第一个参数的默认值为2。注意上面的示例还演示了一个参数可以在不同的位置使用多次。

对第三行命令调用的解释：该命令只选取了前两个作为参数，第三个由于大括号实际上是作为域的定界符，显示时会被忽略，因此实际上只代表了 c 这一个字符，按顺序显示在该命令后面。

<sup>xi</sup>但可以重新定义该命令，参考14.1.3：命令的重新定义

<sup>xii</sup>注意，参数数量和具体不同参数的个数必须相同

### 14.1.3 命令的重新定义

LATEX中最灵活的一点就是命令可以被重新定义，这使得我们可以更灵活的控制文档，但也因为这个原因，很多包之间对命令的复写会导致其相互冲突，如enumerate宏包和enumitem宏包。

命令的重新定义和命令的定义方式几乎一样，只有两点不同：

- 使用\renewcommand命令
- 定义的命令必须是已经定义过的。

## 14.2 环境

### 14.2.1 环境的基本使用

环境的使用需要一对命令：\begin和\end，如居中环境的使用：

#### Code effects

```
\begin{center}  
居中环境中的文字会居中。  
\end{center}
```

居中环境中的文字会居中。

使用环境和使用命令非常相似，其一般格式如下：

```
\begin{envi}[]{}...  
...  
\end{envi}
```

### 14.2.2 环境的定义

环境的定义使用\newenvironment命令，基本格式如下：

```
\newenvironment{name}[num][default]{before}{after}
```

一个示例如下，从这个示例里可以看到环境之间可以互相嵌套：

#### Code effects

```
\newenvironment{titleenv}
{
    \begin{center}居中标题\end{center}
}
{
    \\\rule{\textwidth}{1mm}
}

\begin{titleenv}
    正文内容
\end{titleenv}
```

居中标题

正文内容

---

环境的参数使用方法和命令的参数使用方法完全一致，这里仅提供一个示例：

Code effects

```
\newenvironment{king}[2][Title Name]
{
    \begin{center} \textbf{\#1: #2}\\
}
{
    \end{center}
}
\begin{king}{参数二位置}
    这是多参数省略一参的一个示例
\end{king}
\begin{king}[参数一位置]{参数二位置}
    这是多参数的一个示例
\end{king}
```

Title Name: **参数二位置**  
这是多参数省略一参的一个示例

**参数一位置: 参数二位置**  
这是多参数的一个示例

注意上面的示例中，参数只能放到before区域，不能放到after区域；另外可以通过在开始时添加一个环境头，结束时添加一个环境尾来实现一定的复杂逻辑。

另外，LATEX 中这种环境的定义是最为基本的定义，这种环境基本是可以随意嵌套的，但还有一些环境实际上更为复杂，也不是通过这种定义手段定义的，因此随意嵌套可能会出问题<sup>xiii</sup>

### 14.2.3 环境中的命令

在环境中可以定义命令，这样定义命令仅可以在环境中使用，如itemize环境中的item命令。

定义方法没有其他的区别，仅在使用参数的时候，需要使用连续的两个

<sup>xiii</sup>具体的解释可以查看[这篇回答](#)

# 来表示命令的参数<sup>xiv</sup>，一个示例如下：

Code effects

```
\newenvironment{topics}[1]
{
    \newcommand{\topic}[2]{ \item{##1, ##2} }
    Topics:#1
    \begin{itemize}
}
{\end{itemize}}
\begin{topics}{标题}
    \topic{参数一}{参数二}
    \topic{参数一}{参数二}
\end{topics}
% \eitem{a}{b} 环境外使用会报错误: Undefined control sequence.
```

Topics: 标题

- 参数一, 参数二
- 参数一, 参数二

#### 14.2.4 环境的重新定义

环境的重新定义参考命令的重新定义和环境的定义方法，基本完全相同：

<sup>xiv</sup>否则会报：! Illegal parameter number in definition of \topics.

**Code effects**

```
\renewenvironment{titleenv}[1][居中标题]{  
    \begin{center}#1\end{center}  
}{\rule{\textwidth}{1mm}}  
  
\begin{titleenv}[我的标题]  
正文环境  
\end{titleenv}
```

我的标题

正文环境

### 14.3 高级命令与环境

部份宏包使用了更底层的机制，使得它们可以完成和一般形式不同的命令和环境定制，如对参数名的支持：

**Code effects**

```
\begin{tcolorbox}[colframe=blue!50!,colback=blue!20!]  
盒子内部  
\end{tcolorbox}
```

盒子内部

如果要实现这种效果，可以去学习底层的 TeX 命令，也可以使用宏包 xkeyval 来完成<sup>xv</sup>，精力有限不多介绍，留坑以后填。

<sup>xv</sup> 虽然该宏包的使用也和一般语法不同

## 14.4 计数器

### 14.4.1 基本介绍

LATEX 内置的计数器有 23 个，其中 17 个为序号计数器，6 个为控制计数器（只是用途不同，本质都是计数器的一种）。

计数器名	用途	计数器名	用途
part	部序号计数器	equation	公式序号计数器
chapter	章序号计数器	page	页码计数器
section	节序号计数器	footnote	脚注序号计数器
subsection	小节序号计数器	mpfootnote	小页环境中的脚注序号计数器
subsubsection	小小节序号计数器	enumi	排序列表第 1 层序号计数器
paragraph	段序号计数器	enumii	排序列表第 2 层序号计数器
subparagraph	小段序号计数器	enumiii	排序列表第 3 层序号计数器
figure	插图序号计数器	enumiv	排序列表第 4 层序号计数器
table	表格序号计数器		

表 9: LATEX 内置的序号计数器

bottomnumber	控制每页底部可以放置浮动体的最大数量，默认值为 1
dbltopnumber	双栏排版时，控制每页顶部可放置跨栏浮动体的最大数量，默认值为 2.
secnumdepth	控制层次标题的排序深度，book 和 report 默认为 2，article 默认为 3
topnumber	控制每页顶部可放置浮动体的最大数量，默认为 2
totalnumber	控制每页中可放置浮动体的最大数量，默认值为 4
tocdepth	控制章节目录的目录深度，文类 book 和 report 默认值为 2，而 article 默认值为 3.。通常 secnumdepth tocdepth

表 10: L<sup>A</sup>T<sub>E</sub>X 内置的控制计数器

#### 14.4.2 生成计数器

新建一个计数器：

```
\newcounter{NameOfTheNewCounter}
```

如果在新建时，需要你的计数器在其他计数器增加的时候被清零，可以通过额外添加计数器参数的方式来实现：

```
\newcounter{NameOfTheNewCounter}[NameOfTheOtherCounter]
```

对一个已存在的计数器，如果需要它在某个其他计数器增加的时候被清零，可以通过以下的命令：

```
\counterwithin*[NameOfTheCounter]{NameOfTheOtherCounter}
```

为某个计数器加 1：

```
\stepcounter{equationschapter}
```

指定该计数器的值：

```
\setcounter{NameOfTheNewCounter}{number}
```

为某个计数器增加任意数值（该数值可以是负数），可以使用以下命令：

```
\addtocounter{NameOfTheNewCounter}{number}
```

如果要获取某个计数器的值（上述的命令有的参数需要传入 number 而不是某个 counter），那么使用以下的命令来返回一个 number：

```
\value{NameOfTheNewCounter}
```

### 14.4.3 计数器风格

请参考[3.1.2：数字](#)

## 第十五章 代码环境方案

### 15.1 程序代码

#### 15.1.1 抄录环境

#### 15.1.2 listing

#### 15.1.3 minted

#### 15.1.4 最佳实践: tcolorbox

### 15.2 伪代码

### 15.3 树结构

## 第十六章 编写结构

TODO: 引入文件/引用问题/编译参数/目录/目录问题/代码风格/abspath

TODO: 更改目录结构后, 对各种库、bibtex 的影响和解决

## 第十七章 库使用

### 17.1 minipage

minipage 宏包用于在一页中创建一个小小的页面（实质是一个盒子），也具有很强的可定制性。

## 17.2 tcolorbox

本手册使用的全部带颜色的边框均为 `tcolorbox` 的实现，其基本的使用非常简单，在熟悉 L<sup>A</sup>T<sub>E</sub>X 的基本操作后，可以查看 `tcolorbox` 的宏包说明书来完成，其所有的边框，间隔，圆角，文字，分割线都可以自定义，同时还对代码的显示进行了封装，可以更自由的定制代码的显示风格。

## 17.3 adjustbox

是一个用于任意调整“盒子”位置的宏包，盒子是 L<sup>A</sup>T<sub>E</sub>X 中的基本单位，所以该宏包在一些场合的定制性非常的强。

# 第十八章 高级使用

TODO: 有生之年系列：文档类撰写、Latex 语言深入、库撰写、  
Tex 编译结构 TODO:MakeFile 编写

# 第十九章 模板收录

国赛模板：[Github 链接](#)

TODO: 简历模板、PPT 模板、笔记模板、书模板...