科技论文排版技术
辅助教材

庞冬青

2022 年 5 月 10 日

前言

本书是 LaTeX 入门教材，内容涵盖了天津大学研究生公共课“科技论文排版技术”的讲课内容，并包含了一部分由于时间关系课堂上略去的但是很实用的例子。本书的内容编排强调实用性，而不是系统性。

本书的 tex 分发版本为 texlive，这个版本是 linux 下的默认 tex 版本，同时可以在 windows 下使用。texlive 的下载地址是：https://tug.org/texlive/acquire-iso.html。推荐选择最近的镜像下载，iso 比较大，但是下载速度一般情况下是很快的。

安装速度依赖于电脑配置和系统，Windows 下比较老的电脑要安装很久。texlive 大约每年 4 月下旬更新，推荐使用可移植安装方式。全选安装后占用空间约 7G，系统分区不够大的情况下建议放在其他分区。早期 Windows 系统下安装曾经出现过命令行不可用的问题，需要用手动添加环境变量的方式解决，最近两年没有同学报告过这个问题。

苹果系统下的安装参见手册 2.1。

本书的例子都在 linux 系统下经过实际编译测试，测试版本为注明的 texlive 分发版本。大部分基础代码一般不会因为分发版本的不同产生编译问题，一些功能包因为版本更新缘故，会导致少量的旧代码失效，一些 bug 也可能会在新版本中被修复。极少部分代码涉及操作系统，比如系统字体或超链接打开视频文件这样的代码，给出的代码例子仅仅在笔者电脑中可以正确运行，这点需要额外注意。

书后的索引主要是包和运行命令，以及 tikz 的库。

虽然本书的代码可以 copy 后使用，强烈建议自己输入代码方式学习，尤其是数学公式。不推荐初学者依赖代码补全功能。

本书计划每年在四月末上完课后分发，每年八月初开始基于当年的 texlive 版本修订工作。本书计划中还有第八章内容，将给出一些常用设置代码，但是这部分工作预计会晚两年开始。

写书是一个永无止境的事情，在书写过程中总是不断找到各种小错误，尤其是描述部分的代码和文字，所以必须设置一个 deadline 来终结工作。

因此本书计划每年五一前分发出去，随后发现的各种错误将在下一般版本中被修订。

本书将以开源方式分发，可以在下面的链接中免费下载。

http://ull.tju.edu.cn/education/course/latex.html

目录

前言 i

1 编译方式 1

1.1 使用 latex 编译 . . . 2
1.2 使用 pdflatex 编译 . . . 5
1.3 加载 xeCJK 包且使用 xelatex 编译 . . . 7
1.4 参考文献 . . . 8
1.4.1 natbib 包和 bibtex 编译方式 . . . 8
1.4.2 biblatex 包和 biber 或 bibtex 编译方式 . . . 10
1.5 索引 index 的编译 . . . 11
1.6 符号表 nomencl 的编译 . . . 14
1.7 专业术语表 glossaries 的编译 . . . 16
1.8 画图包 pstricks 的编译 . . . 18
1.9 文件格式转换 . . . 19

2 排版 21

2.1 基础设置 . . . 21
2.1.1 字号 . . . 22
2.1.2 行间距 . . . 23
2.1.3 字体 . . . 24
2.1.4 缩进 . . . 25
2.1.5 横向空白和纵向空白 . . . 25
2.1.6 对齐 . . . 25
2.1.7 分页 . . . 27
2.1.8 边注 . . . 27
2.1.9 图文混排 . . . 28
2.1.10 常用类的选项 . . . 29

2.1.11 划线 30
2.1.12 条目 31
2.2 页面设置包 geometry 34
2.3 颜色包 xcolor 37
2.4 盒子 box 38
2.5 插图 41
2.5.1 基本用法 41
2.5.2 设置 caption 形式 44
2.5.3 子图 45
2.6 表格 47
2.6.1 基本功能 47
2.6.2 三线表 49
2.6.3 表格颜色 50
2.6.4 单元格内换行 makecell 包 51
2.6.5 同行合并列 52
2.6.6 同列合并行 multicol 包 53
2.6.7 表格脚注 54
2.6.8 跨页表格 longtable 55
2.6.9 参考书和一些有用的包 55
2.7 超链接包 hyperref 56
2.8 文章 article 类 59
2.9 参考文献 61
2.9.1 bib 格式的文献条目 61
2.9.2 natlib 包 +bibtex 编译 64
2.9.3 natbib 包情况下分章参考文献 66
2.9.4 biblatex 包 +biber 编译 68
2.9.5 biblatex 包 +bibtex 编译 69
2.9.6 biblatex 的常见功能介绍 70
2.10 版面设计 titlesec 与天津大学研究生论文的格式设置 72
2.10.1 设置页眉页脚 75
2.10.2 章节名称的样式 79
2.10.3 公式图表 81
2.10.4 目录样式 82
2.10.5 题目页设计 83
2.10.6 其他常用书籍排版 83

2.11 页眉页脚包 fancyhdr 85
2.12 脚注 footnote 86
2.13 信 letter 类 87
2.14 背景水印 87
2.14.1 background 87
2.14.2 eso-pic 88
2.15 文章内分栏 89
2.15.1 多栏 multicol 包 89
2.15.2 交互两栏 paracol 91
2.15.3 更复杂的多栏版面 flowfram 92

3 数学公式 95
3.1 行间公式 95
3.2 独立公式环境 95
3.3 amsmath 包的基本用法 96
3.3.1 自定义函数 96
3.3.2 amsmath 包的公式环境 97
3.4 分式和 displaystyle 101
3.5 开根号的细节 102
3.6 括号 103
3.7 积分 105
3.8 矢量 106
3.9 矩阵 108
3.10 狄拉克符号 109
3.11 复杂上下标 109
3.12 一些比较常见的特殊形式 111
3.13 结构型公式 112
3.14 标注重点 113
3.15 数学符号 114
3.16 mathtools 116
3.17 表格里的公式 117
3.18 定理 118
3.19 网页公式 118

4 画图包 tikz **121**

4.1 直线 . . . 122
4.1.1 坐标 . . . 122
4.1.2 线参数 . . . 124
4.1.3 箭头和 arrows.meta 库 . . . 127
4.2 矩形、圆、椭圆和弧 . . . 129
4.2.1 矩形 . . . 129
4.2.2 圆和圆弧 . . . 131
4.2.3 椭圆和椭圆弧 . . . 132
4.3 曲线 . . . 134
4.3.1 任意曲线 controls 方式 . . . 134
4.3.2 任意曲线 bend 方式 . . . 135
4.3.3 抛物线 . . . 136
4.3.4 正弦余弦曲线 . . . 136
4.3.5 plot 语句 . . . 136
4.4 一些功能 . . . 137
4.4.1 标识角度 angles 库 . . . 137
4.4.2 剪切 clip . . . 138
4.4.3 网格 grid . . . 139
4.5 循环 foreach . . . 139
4.6 渐变色和混色 . . . 140
4.6.1 渐变 shade . . . 140
4.6.2 透明 transparent . . . 142
4.6.3 光学混色 blend . . . 143
4.6.4 奇偶原则 . . . 143
4.7 分层 . . . 144
4.8 数学库 math . . . 146
4.9 node . . . 146
4.9.1 基本功能 . . . 147
4.9.2 命名 node 并使用 node 名坐标 . . . 149
4.9.3 edge 操作 . . . 150
4.9.4 node 的形状和 shapes.geometric . . . 151
4.9.5 node 的其他库 . . . 158
4.10 定义样式 . . . 159
4.10.1 修改默认样式 style . . . 159

4.10.2 自定义样式 . . . 160
4.10.3 自定义带参数的样式 . . . 160
4.10.4 修饰库 . . . 162
4.11 三维坐标 . . . 163
4.11.1 默认三维坐标 . . . 163
4.11.2 3d 库 . . . 165
4.11.3 tikz-3dplot 包 . . . 166
4.11.4 perspective . . . 168
4.12 坐标计算 calc 库 . . . 168
4.13 几何做图 . . . 171
4.13.1 through . . . 171
4.13.2 相交 intersections . . . 171
4.13.3 相切 . . . 172
4.13.4 calc 库和几何做图 . . . 173
4.14 matrix 库 . . . 173
4.15 思维导图 . . . 174
4.16 电路 . . . 175
4.17 pgfplots . . . 179
4.17.1 二维函数画图 . . . 179
4.17.2 二维数据画图 . . . 180
4.17.3 三维函数画图 . . . 181
4.17.4 三维 map 图 . . . 181
4.17.5 从数据画三维图 . . . 182
4.18 基于 tikz 和 pgf 的科学包或 tikz 库 . . . 182
4.18.1 高亮数学公式 . . . 182
4.18.2 物理类 . . . 183
4.18.3 量子 . . . 184
4.18.4 光路图包 tikz-optics . . . 184
4.18.5 工程 . . . 184
4.18.6 chart . . . 185
4.18.7 其他 . . . 185
4.19 基于 tikz 和 pgf 的其他包 . . . 185
4.19.1 装饰纹包 pgfornament . . . 185
4.19.2 tikz-among-us . . . 185
4.19.3 tikz-truchet . . . 186

4.19.4 动物包 . . . 186
4.19.5 符号 . . . 186
4.20 几个光路图的例子 . . . 186

5 一些有用的包 191
5.1 standalone 类 . . . 191
5.2 计算机程序语法高亮：listings . . . 192
5.3 tcolorbox . . . 195
5.3.1 数学公式 . . . 195
5.3.2 theorems . . . 195
5.3.3 tcblisting . . . 196
5.4 动图：animate . . . 198
5.4.1 利用多张图片实现动图 . . . 198
5.4.2 tikz 直接画动态图 . . . 199
5.4.3 tikz 多张画图方式 . . . 201
5.4.4 时序控制 . . . 202
5.5 试卷：exam 类 . . . 205
5.6 其他一些不务正业的包 . . . 208

6 化学分子式 chemfig 211
6.1 基本设置 . . . 211
6.2 键的基本语法 . . . 212
6.3 分子结构 . . . 215
6.4 结构复用 . . . 221
6.5 对称 . . . 222
6.6 Lewis 形式 . . . 223
6.7 高分子 . . . 223
6.8 一些细节 . . . 224
6.9 与 tikz 混用 . . . 225
6.10 chemmove . . . 226

7 演示稿 beamer 229
7.1 beamer 模板 . . . 229
7.2 frame . . . 230
7.3 beamer 自定义的分栏环境 . . . 232
7.4 目录、暂停、显示顺序和跳转 . . . 233

7.5 图片 236
7.6 简单的设置语句 238
7.7 beamer 内置的 box 环境 242
7.8 beamer 模板设计 243
7.9 两个重新定义 frame 的例子 244

索引 **247**

第一章　编译方式

本章主要介绍常用的编译步骤。最开始的三个小节是上课前必须掌握的，后面的编译技巧则随着学习各种包逐渐掌握。编译方式这章的内容，有些是逐渐学习的，有些是课堂没有时间详细讲，需要课下练习的。这个辅助教材的内容不是完全按照讲课的顺序进行的。

本课程的例子，除了 pstricks 和 listings 的例子外，都是采用 pdflatex 编译的。汉字都是采用 CJKutf8 包。本书中的极少部分例子需要 xelatex 编译，更换编译方式时会特别指出。本书则是 xelatex 编译的，汉字采用 xeCJK 包。

本书编译命令行是 linux 下的结果，Windows 下命令可能需要加.exe 后缀名。Texworks 是一个集成环境，可以用鼠标方式选择编译命令。最新版 Windows 的 linux 子系统 WSL 可能更方便使用命令行，请自行测试。

调用文档

调用文档的命令为 texdoc，下面的命令可以调用 texlive 分发版的文档，获得安装说明。Windows 下命令行调用文档成功可以说明命令行语句是可用的，安装成功后可以先测试一下 texdoc 命令。

```
texdoc texlive
```

lshort-zh 是 texlive 自带的 LaTeX 介绍 lshort 的中文翻译版，没有-zh 获得的是英文版。这里特别感谢 ctex 小组持续地更新该文档的中文版。调用中文手册

```
texdoc lshort-zh
```

基本上所有导言中用\usepackage{包名}语句加载的包，都可以用 texdoc 调用相关文档。几乎所有文档都是 pdf 格式的，除了一些特别古老的包，如 CJK，是其他格式的，文本格式或网页格式。这些格式的文档也都可以通过 texdoc 调用相应的阅读器（一般是浏览器）自动显示。

texdoc 命令会自动调用 pdf 阅读器，linux 下是 xpdf，可以在 texdoc.cnf 文件里修改 texdoc 命令调用的 pdf 阅读器。Windows 下应该是调用系统默认 pdf 阅读器。

1.1 使用 latex 编译

tex 文件是文本文件，原则上可以用任何文本编辑器编写。下面是一个最简单的 tex 文件的例子，为了简单起见，本书中我们将所有 tex 文件的例子都命名为 a.tex。

```
\documentclass[12pt, a4paper]{article} %默认是 10pt
%这里是导言部分，加载包，进行一些设置
\begin{document} %开始文档

Hello!%不会断行，自动添加一个空格
New day!

\end{document}
```

latex 是最初始的编译方式，texlive 中 texworks 里选择 latex 编译方式即可编译 a.tex 文件。在命令行下可以使用下面的语句进行 latex 方式的编译。

```
latex a.tex %扩展名可以省略
```

latex 方式编译获得的是 dvi 格式的文件，用 xdvi 命令可以查看 dvi 文件。texlive 自带的 dvi 阅读器为 xdvi，texlive 的 texworks 里可以自动显示编译结果。命令行下的语句为

```
xdvi a.dvi
```

现今最常用的通用文件格式是 pdf 格式，可以使用下面的命令将 dvi 文件转换为 pdf 文件。

```
xdvipdfmx a.dvi
```

latex 编译方式时插入图片的格式只能是 eps。

latex 编译方式看起来有些过时了，但是我们还是需要学习一下，这样可以更好地编译（理解编译）pstricks 包画出的图形。一些功能包，例如 chemfig，虽然支持 latex 编译，但是必须加载包的特殊选项才能在编译后正确地将 dvi 文件转换为 pdf 文件，且只有 pdf 文件才能正确显示结果。坚持这种编译方式必须要阅读包的文档，常见的大型功能包的手册（文档）都

会给出不同编译方式的要点说明。

附属文件

除了自动生成的 dvi 格式文件外，这个最简单的例子还自动生成了两个附属文件：a.aux 和 a.log。a.log 里保存了编译过程的信息，log 文件可以方便我们查看错误信息，帮助我们修正 tex 文件中的错误。需要注意的是：a.log 里的信息和编译过程中调试窗口显示的信息虽然有大量重合，但是内容并不是完全一致的。

在编译过程中生成的附属文件不要删除，否则会增加编译时间。因此一般建议一个 tex/pdf 一个目录，方便文件管理。如果图比较多，可以再建一个或多个子目录放插入的图片。如果有目录、索引、参考文献和术语表，编译生成的附属文件还会增加，分别有不同的扩展名。

关于附属文件还有一个值得说明的地方：如果坚信自己的 tex 文件代码正确，但是编译总是出现错误，可以将自动生成的附属文件全部删除，然后重新编译。在某些极端错误情况下，会产生这样的特例。错误信息被附属文件记录下来，没有随着再次编译自动修正，导致编译过程产生虚假错误。

一般情况下，第一次编译正确，随后几次编译错误，说明代码是错误的。第一次编译错误，随后几次编译正确，说明代码是正确的。一般编译两次足以判断代码是否正确，但是特殊情况下也有可能需要编译三、四次才会得到正确的显示结果。

tex 文件的基本要点

一般的 tex 分发版本都自带具有语法高亮功能的编辑器，比如 texlive 的编辑器。还有一些分发版本添加了一些额外的编辑功能，例如基础代码补全、自动环境加载等。甚至于一些分发版本还内置了一些公式形式的图形界面。不推荐初学者依赖这些高级功能，因为养成良好的编程习惯比快速获得正确结果重要。

从这个最简单的例子可以看出 tex 文件的几个基本特点：

1. 命令在 utf8 编码环境下用反斜杠引导。
   如果在网上看到用斜杠引导命令的 tex 文件例子，有可能该文档使用的是其他编码。现今 utf8 编码已经成为各个操作系统中的常用编码，一般不存在操作系统不支持 utf8 编码的特殊情况。
2. 第一行是\documentclass{类名}，除非加载特别的包外，一个 tex 文件只有一个\documentclass。tex 的基础语法文档是 classes，该文档

介绍了默认的常用类和基础语法功能。

```
texdoc classes
```

不建议初学者一开始就阅读该文档，做完基本练习后再阅读该文档更合适。

3. \documentclass语句和 document 环境之间是导言部分。一般在导言中加载包并设置包的参数。**但是涉及到中文时一般需要将设置参数放到中文环境后，所以也在 document 环境之内。**
4. tex 文件中\begin{环境名}和\end{环境名}构成环境。必须**有且只有一个** document 环境。

```
\begin{document}
...
\end{document}
```

begin 和 end 语句必须配对，配对原则是：先 begin 的后 end。

```
\begin{document} %先 begin
\begin{equation}
f(x)
\end{equation}
\end{document} %后 end
```

养成良好的编程习惯，先将配对写好再填写内容，这样不容易代码出错。**配对错误发生时编译过程给出的错误代码行号信息往往是最后一个无法配对的地方，而不是配对错误发生的地方。**大型文档时，往往需要在提示的错误代码行号位置往前查找错误，而不是提示的错误代码行号处。**最容易出现编译错误的地方是环境名拼写错误。**尤其是对初学者而言，这个错误非常常见，仔细阅读错误信息是可以看出来的。

5. 回车不分行，分行需要空白行，多个空白行等价于一个空白行（请自行练习）。利用这个特点，每行一句话，结合 vim 编辑器，可以非常快速调整语句顺序。

input 和 include

如果写大型文件，比如本书，可以采用 input或 include方式，将内容放在几个文件中，而不是一个大的 tex 文件。input 和 include 的用法是一样的。

```
\documentclass[12pt, a4paper]{article}
```

```
\begin{document}
\input{b.tex}
%或者
\include{b} %tex 文件可以不用加扩展名
\end{document}
```

扩展名随意，比如 b.txt 亦可，但是 txt 文件时必须加扩展名，tex 文件时只需要文件名。

```
\input{b.txt}
```

b.tex 里的内容为

```
Hello!
New day!
```

input 和 include 的区别在于，input 方式加载内容被视为纯插入，编译后不会产生文件 b 的附属文件。include 方式加载内容，编译后会产生文件 b 的相应附属文件。如果编译过程需要 b 的附属文件，就必须采用 include 方式加载内容。

1.2 使用 pdflatex 编译

pdf 格式是最通用的文件格式，使用 pdflatex 编译方式可以一步到位获得 pdf 文件。目前绝大部分包都支持 pdflatex 编译方式。下面的例子包含了中文环境 CJK。

```
\documentclass[12pt,a4paper]{article} %默认是 10pt

\usepackage{CJKutf8} %最基础的中文包，与 pdflatex 配合使用

\begin{document}
\begin{CJK}{UTF8}{gkai} %gkai 是楷体，宋体是 gbsn

下面9个字符，必须以下面的方式才能正确显示。其中百分号是单行注释符。

\%
\# %变量
\$ %行间公式符号，必须配对
\{ \} %模块符号，必须配对
\^{} %公式上标
```

```
\_{} %公式下标
\~{} %间距调整符

四个反斜杠的打法，显示效果略有不同，两个用了行间公式方法。

\verb|\|a
\textbackslash a
$\backslash$a
$\setminus$a

\end{CJK}
\end{document}
```

这个例子除了注释外，还有几点需要额外说明。

1. 使用 CJKutf8 包，不要使用 CJK 包，这样生成的 pdf 文件里的中文可以 copy。
2. 百分号是单行注释符，多行注释用下面的模块

```
\iffalse
需要注释掉的语句
\fi
```

3. 例子中的符号，绝不能出现在 tex 文件的文本中，否则会**报错且有时难以查找错误**。

CJKutf8 有几个固有缺陷，下面的情况需要改用 xeCJK 包，并且使用 xelatex 方式编译。

1. 偏僻汉字。
   CJKutf8 包的汉字集有限，如果汉字无法显示，则说明超出了汉字集的范围，需要 xeCJK 包。
2. 代码中有汉字。
   使用 CJKutf8 包时，汉字只能出现在文本中，不能出现在代码中。常见的代码中出现汉字的情况为
   - 代码中包含中文路径。
   - listings 包（计算机语言高亮包）显示的代码中有中文。
   - 汉字画图。（画图包文本框里的中文除外）。
3. 多种字体。
   CJKutf8 只有两种字体，楷体 gkai 和宋体 gbsn。

虽然有这样那样的不足，CJKutf8 包也有几个优点，除了写本书外（代码中有中文），笔者基本都采用 pdflatex 和 CJKutf8 包。

1. 支持度最广。可能有些工具包不支持 xelatex 编译方式或者不支持 xeCJK。这是两种可能，不能将 xelatex 和 xeCJK 等价看待。xeCJK 必须采用 xelatex 编译，但是 xelatex 可以支持全英文或者其他语言。
2. 与基础命令最融洽。不需要特别设置，latex 基本语法对汉字有效，比如加粗。
3. 编译速度快。xelatex 第一次的编译速度比 pdflatex 略微慢一点，不过后续编译速度差别也不是很大。

后两点其实不太重要，主要是第一点，有些情况 xeCJK 包会产生微小偏差，比如字符无法显示，这种错误来自包匹配问题，只能通过切换编译方式检查。特殊情况 xelatex 编译结果也会出现微小偏差，而 pdflatex 是最稳定的。随着 xeCJK 的发展，相信该包是写中文文档的未来趋势。

1.3 加载 xeCJK 包且使用 xelatex 编译

xeCJK 包的中文环境代码如下

```
\documentclass[12pt,a4paper]{article}
\usepackage{xeCJK} %加载包
\setCJKmainfont{SimSun} %默认中文字体
\setmainfont{Times New Roman} %默认英文字体，不设置则为 tex 默认

\setCJKfamilyfont{heiti}{黑体} %必须是自己电脑中有的字体

\newcommand{\heiti}{\CJKfamily{heiti}} %heiti 是自己起的名

\begin{document}
中文

{\CJKfamily{heiti}中文} %使用字体

{\heiti 中文} %更简洁的使用方式

中文
abcdefg flammaa
\end{document}
```

也可以用下面的语句将\bf和\textbf设置成黑体，\it和\textit设置成楷体。

```
\setCJKmainfont{SimSun}[BoldFont=SimHei, ItalicFont=KaiTi]
```

xelatex 如果不设置，默认中文字体是没有\bf和\textbf的。如果需要宋体加粗，可以采用伪加粗方式。天津大学研究生论文摘要二字要求宋体加粗，而不是黑体，所以上面将粗体设置为黑体就不合适了。

```
\setCJKmainfont[AutoFakeBold]{SimSun}
```

xelatex 包修改了默认的中文图表 caption 设置，汉化了 chaptername，做了不少汉化工作，后面讲到的汉化语句大部分都是基于 pdflatex 编译情况的。

更普遍的大型汉化包是 ctex，集成了很多功能。

```
\usepackage{ctex}
```

1.4 参考文献

参考文献有两种方式，下面分别介绍。

1.4.1 natbib 包和 bibtex 编译方式

```
\documentclass[12pt,a4paper]{article}

\usepackage{CJKutf8} %解决汉字 copy 问题
\usepackage[T1]{fontenc} %解决 ffl 等 copy 问题
\usepackage{xcolor}

\usepackage[super,square,comma,sort&compress]{natbib}

\usepackage[unicode,colorlinks,linkcolor=blue]{hyperref}%超链接

\begin{document}
\begin{CJK}{UTF8}{gkai}

第一个参考文献\cite{fluid}第二个参考文献\cite{ol}。

\clearpage
第一个参考文献\cite{fluid}
```

```
博士论文\cite{wang}

\bibliographystyle{unsrt} %参考文献格式，扩展名为 bst
\bibliography{a}

\end{CJK}
\end{document}
```

参考文献内容写在 bib 文件中，这里采用 a.tex 同名文件 a.bib，bib 文件可以不与 tex 文件同名。

```
@book{fluid,
author={朗道},
title={流体力学},
publisher={高等教育出版社},
address={北京},
edition={第三版},
year={1980},
}

@article{ol,
author={M. Bertero and C. De Mol and G. A. Viano}, %作者用 and 分隔
title={Restoration of optical objects using regularization},
journal={Optics Letters},
volume={13},
issue={2},
pages={51--53}, %起始页码，也可以是单页码
year={1978},
}

@phdthesis{wang,
author={王某某},
title={博士论文},
school={天津大学},
address={天津},
year={1980},
}
```

参考文献的编译分四步，命令行下 bibtex 编译时不加扩展名，图形界面选

择 bibtex 编译即可。

```
pdflatex a.tex
bibtex a %此处是 tex 文件名，不是 bib 文件名
pdflatex a.tex
pdflatex a.tex
```

也可以在\documentclass之前加\write18{}语句一步编译到位，

```
\immediate\write18{bibtex \jobname}
```

此时需要 pdflatex 编译三次，因为这句相当于 pdflatex 编译时，自动地用 bibtex 语句编译了一次，后面两次 pdflatex 编译不能省略。这个方法无法在 biber 编译时使用，因为默认配置只允许调用有限几个编译命令，其中最常用的是 bibtex 和 makeindex 两个命令。

投稿时，如果需要网络编译的话，需要的是编译生成的 a.bbl 文件，而不是初始的 a.bib 文件。

大型杂志社都给出了自己的参考文献格式 bst 文件，制作 bst 文件的命令为

```
latex makebst
```

天津大学硕士和博士论文的 bst 文件可以选择 zharticle.bst，**书和硕士、博士论文都需要给出地址，不能没有地址项，地址项也不能空白。**这个文件不在 texlive 的 bst 文件目录下，可以找到该文件，将其复制到 tex 文件所在目录，也可以将其复制到 bst 所在目录，然后运行 texhash 命令。

```
texhash
```

texhash 命令可以更新整个 texlive 目录的索引，自行安装了新的包或格式文件，texhash 一下即可使用。

1.4.2 biblatex 包和 biber 或 bibtex 编译方式

biblatex 是新的参考文献包，，可以指定用 biber 编译，也可以指定用 bibtex 编译。

```
\documentclass[12pt,a4paper]{article}

\usepackage{CJKutf8} %解决汉字 copy 问题
\usepackage[T1]{fontenc} %解决 ffl 等 copy 问题
\usepackage{xcolor}
```

```
%使用 biblatex 代替 natbib style=numeric,sorting=none,
\usepackage[backend=biber]{biblatex} %也可以选择 bibtex 编译

\addbibresource{a.bib} %添加 bib 文件

\usepackage[unicode,colorlinks,linkcolor=blue]{hyperref}%超链接

\begin{document}
\begin{CJK}{UTF8}{gkai}

第一个参考文献\cite{fluid}第二个参考文献\cite{ol}。

\clearpage
第一个参考文献\cite{fluid}

\printbibliography[title=参考文献] %汉化很容易

\end{CJK}
\end{document}
```

参考文献的编译分四步，命令行下 biber 或 bibtex 编译时不加扩展名，图形界面选择 biber 编译即可。如果 backend=bibtex，第二步则是 bibtex 编译。

```
pdflatex a.tex
biber a %这里是 tex 文件名
pdflatex a.tex
pdflatex a.tex
```

biblatex 能够支持更复杂的参考文献格式，但是 natbib 目前投稿更常用。

1.5 索引 index 的编译

使用 makeidx 包用来自动获得索引，例子如下

```
\documentclass[12pt]{article}

\usepackage{setspace}

\usepackage{makeidx}
```

```
\makeindex %必须有，自动生成 idx 文件

\usepackage[unicode,colorlinks,linkcolor=blue]{hyperref}%超链接

\begin{document}
\begin{spacing}{1.25}

laser\index{laser} is useful. %设置索引

\clearpage %分页
second place\index{laser} is useful. %再次设置同名索引
\printindex %生成的索引有两个页码

\end{spacing}
\end{document}
```

编译时分三步编译，

1. pdflatex 编译 a.tex 文件

   ```
   pdflatex a.tex
   ```

   编译后自动生成 a.idx 文件，此时生成的 pdf 文件没有索引内容
2. makeindex 编译 a.idx 文件

   ```
   makeindex a.idx
   ```

   编译后生成 a.ilg 和 a.ind 文件
3. pdflatex 再次编译 a.tex 文件，最好编译两次

   ```
   pdflatex a.tex
   ```

   获得有 index 的 pdf 文件

在\documentclass前加下面的语句可以一次到位编译。

```
\immediate\write18{makeindex \jobname.idx}
```

\write18只适合有限命令，最常见的就是 bibtex 和 makeindex。因为 makeindex 也编译符号表，所以这个语句还是非常有用的。只不过这个语句没有在 windows 下测试成功过。

很多时候需要将索引再分类，比如第四章将所有 tikz 的子库和相关功能包放在一起显示，这时只需要写为\index{tikz!放在索引tikz下的其他索引}

即可。

可以通过文件设定 index 的格式，此时需要加载 imakeidx 包，

```
\usepackage{imakeidx}
```

默认是两栏，字母，可以在\makeindex语句处修改，比如改为三栏，

```
\makeindex[columns=3, title=Alphabetical Index]
```

还可以在\makeindex语句里加载格式文件 myindex.ist，该文件放在 tex 文件所在目录即可。

```
\makeindex[options=-s myindex.ist]
```

下面是一个格式文件的例子，并不好看

```
headings_flag 1 %插入分组 title，下面设定 title 格式
heading_prefix "\n\\centering\\large\\sffamily\\bfseries\\noindent\\
textbf{"heading_suffix "}\\par\\nopagebreak\n"

item_0 "\n \\item \\small "

delim_0 " \\hfill " %页码右对齐
delim_1 " \\hfill "
delim_2 " \\hfill "
```

这个文件修改的时候要注意的是，各部分是用\分隔的，\large是命令，所以一般情况下不能随便删除第一个\。上面的例子比较难看，下面这个好点

```
headings_flag 1

heading_prefix "\\par\\large \\textbf{"
heading_suffix "}\ \\*\~\\\\\*"

item_0 "\n \\item \\small "

delim_0 " \\hfill "
delim_1 " \\hfill "
delim_2 " \\hfill "
```

1.6 符号表 nomencl 的编译

使用 nomencl 包生成符号表

```
\documentclass[12pt]{article}

\usepackage{nomencl}
\makenomenclature

\usepackage[unicode,colorlinks,linkcolor=blue]{hyperref}

\begin{document}
\[
I=I_0\exp(-\alpha l)
\]
\nomenclature{$\alpha$}{absorption}
\nomenclature{$l$}{sample length}

\printnomenclature
\end{document}
```

分三步编译

1. pdflatex 编译 tex 文件

   ```
   pdflatex a.tex
   ```

   编译后获得 a.nlo 文件
2. 使用 makeindex编译 a.nlo 文件，手册里的语法是

   ```
   makeindex a.nlo -s nomencl.ist -o a.nls
   ```

   获得 a.ilg 和 a.nls 文件。pdflatex 编译 tex 文件，最好编译两次。

   ```
   pdflatex a.tex
   ```

   编译后获得有符号表的 pdf 文件。

如果希望一次到位，可以在\documentclass前加下面的语句

```
\immediate\write18{makeindex \jobname.nlo -s nomencl.ist -o \
jobname.nls}
```

然后只用 pdflatex 编译即可。

有时前面的符号项比较长，导致描述部分不能左对齐，用下面的语句调整符号和描述之间的间距，形成对齐的两栏样式。

```
\printnomenclature[4cm]
```

上面的例子得到的术语表是混排的，还可以分组，一般情况将字母和希腊字母分开，比如说在希腊字母前加标识 [G]，希腊字母就会与一般的英文字母分开排序。

```
\nomenclature[G]{$\alpha$}{absorption}
```

排序时按分组标志的字母顺序排序，例子中希腊字母加 G 阅读方便而已，如果有多个分组，比如分组名为 F，调换希腊字母组的顺序，可以将 G 改为顺序在 F 前的英文字母。

需要手动排布顺序，可以使用编号。

```
\nomenclature[01]{$\alpha$}{absorption}
\nomenclature[02]{$l$}{sample length}
```

两位数时需要补 0，否则排序不正确。这类似于电脑目录文件名显示时的排序规则。

分组手动排序，[分组, 编号]

```
\nomenclature[P,1]{$\alpha$}{absorption}
\nomenclature[S,1]{$l$}{sample length}
\nomenclature[S,2]{$x$}{position}
```

给分组加小标题略微麻烦一点，需要 ifthen 包，并自己定义标题

```
\usepackage{ifthen}
\renewcommand{\nomgroup}[1]{%
  \item[\bfseries
  \ifthenelse{\equal{#1}{P}}{Physics constants}{%
  \ifthenelse{\equal{#1}{S}}{Other symbols}{}}%
  ]}
```

还有一种办法语句类似，使用 etoolbox 包

```
\usepackage{etoolbox}
\renewcommand\nomgroup[1]{%
  \item[\bfseries
  \ifstrequal{#1}{P}{Physics constants}{%1
  \ifstrequal{#1}{N}{Number sets}{%2
```

```
  \ifstrequal{#1}{O}{Other symbols}{}}}%配对 3 个右花括号
]}
```

汉化方式为下面的语句

```
\renewcommand{\nomname}{术语表}
```

一般情况下符号表的间距都太大，可以用下面的命令调整间距，写到导言里即可

```
\setlength\nomitemsep{-5pt}
```

更复杂的修改样式的方法可以参看手册。

1.7 专业术语表 glossaries 的编译

使用 glossaries 包可以产生术语表。超链接包必须放在 glossaries 包之前，否则术语表无法超链接。

```
\documentclass[12pt]{article}

\usepackage[unicode,colorlinks,linkcolor=blue]{hyperref}%超链接

\usepackage{glossaries}
\makeglossaries
\loadglsentries{k.txt} %加载 entry 文件

\begin{document}

laser is useful.

\clearpage
\gls{YAG} and \gls{Ti} are both lasers.
\printglossary

\end{document}
```

术语文件 k.txt 里的内容为

```
\newglossaryentry{YAG}{
name={YAG laser},
sort={laser},
```

```
description={an example}}

\newglossaryentry{Ti}{
name={Ti:sapphire laser},
sort={laser},
description={an example}}
```

术语表项加了 sort，会按 sort 排布术语表，相同 sort 的放在一起。sort 不是必须的，但是 description 是必须的。

分三步编译

1. pdflatex 编译 tex 文件

```
pdflatex a.tex
```

编译后获得 ist 和 glo 文件

2. makeglossaries编译 glo 文件

```
makeglossaries g.glo
```

编译后获得 glg 和 gls 文件

3. pdflatex 编译 tex 文件，最好编译两次。

```
pdflatex a.tex
```

编译后获得有术语表 glossary 的 pdf 文件。

术语表如果不打印，可以作为词语或语句替换使用。比如某文件反复出现的某个词，翻译时拿不准怎么翻译的某个词，或者开介绍信写通知时，只在术语表内修改内容，加载不同文件，不修改主体 tex 文件的格式，就可以修改内容，等等。此时术语表项可以没有 sort 项，但是不能没有 description 项，描述置空即可。

```
\newglossaryentry{com}{
name={张三},
description={}}

\newglossaryentry{date}{
name={2022年2月16日},
description={}}
```

1.8 画图包 pstricks 的编译

下面的代码是 pstricks体系下 pst-optic 包的一个透镜的例子，

```
\documentclass{standalone}
\standaloneconfig{border=5pt}

\usepackage{xcolor}
\usepackage{pst-optic} %光学包，透镜成像等

\begin{document}

\begin{pspicture}[showgrid=true](-5,-2.2)(7,4)
\rput(1.5,1.5){%
\lens[lensType=DVG,lensGlass=true,lensWidth=0.5,rayColor=red,
focus=-2,AB=2,spotAi=270,spotBi=90]}
\end{pspicture}
\end{document}
```

pstricks 画图包有多种编译方式，最初始的编译方式是分三步编译

1. latex 编译 a.tex 文件，得到 dvi 文件

   ```
   latex a.tex
   ```

2. 将 dvi 文件转换为 ps 文件

   ```
   dvips a.dvi
   ```

3. 将 ps 文件转换为 pdf 文件

   ```
   ps2pdf a.ps
   ```

最简单的是用 xelatex 直接编译，可以一步到位生成 pdf 文件。

```
xelatex a.tex
```

利用 pst2pdf 可以获得 pdf 文件，还可以获得其他格式的文件。

```
pst2pdf a.tex -c -p
```

编译后自动生成的 images 文件夹中包含了 pdf 图片和 png 图片。

这门课选讲的画图包是 tikz，但是不能否认，pstricks 有着更丰富的专业包，而且 pstricks 在三维立体图形方面优于 tikz。如果真做对比的话，

tikz 语法更接近初始的 tex 语法规则，对空格不敏感，而 pstricks 必须严格书写代码，如果没有空格的地方有空格，就会报错或无法正确显示。化学分子式包 chemfig 是基于 tikz 的，所以更推荐使用 tikz 包。

1.9 文件格式转换

texlive 自带了一些文件格式转换命令。下面这些常用转换语句不会降低清晰度，如果原图是矢量高清的，结果也是矢量高清的。

dvi 转 pdf

```
xdvipdfmx name.dvi
```

ps 转换为 eps，-B 必须有，这样 ps 图片的空白会被截去，最后只有图片部分，即获得一个有尺寸的 tight bounding box。-l 扩展 box 使得边缘有 1pt 的留白。

```
ps2eps -B -l name.ps
```

-R 是旋转，加号是顺治针转 90 度，减号是逆时针转 90 度，上标符号^是转 180 度。

```
ps2eps -B -l -R + name.ps
```

如果有同名 eps 文件存在，该命令默认是不覆盖的，可以选择删除同名 eps 文件，或者使用强制选项-f。

```
ps2eps -B -l -f name.ps
```

eps 转换为 pdf

```
epstopdf name.eps
```

ps 也可以直接转换为 pdf，之所以选择 ps 到 eps 到 pdf，是因为很多情况下 ps 直接转换为 pdf 没有 tight bounding box，导致 pdf 图片有非常大的空白边缘。这就是 ps2eps 命令的-B 选项的作用。但是有的 ps 文件本身就有尺寸设置，这种情况就可以直接转换到 pdf 了。

```
ps2pdf a.ps
```

adobe 的 reader 有快照工具，可以选择局部内容并打印到文件，即打印为 ps 文件。这种方式获得的 ps 文件经过上面的转换步骤，最终获得的 pdf 文件清晰度不会降低，而且可以获得边缘空白最少的图片。因为图片尺

寸是自动去除空白后计算出来的 tight bounding box 的尺寸，而不是靠鼠标选取的。高版本的 adobe reader 可以截取局部直接到 pdf 和 png，据说也不会降低清晰度。其他 pdf 阅读器，好像只有福昕的阅读器可以快照打印到 ps 文件。

pdf 文件到 png 文件的格式转换，linux 下是 convert 命令。convert 命令可以很方便地转换各种图片格式。网页图片格式 webp 比较特殊，转换命令是 cwebp 和 dwebp。

第二章　排版

排版这章涉及的内容比较多且杂乱。索引、术语表和符号表的内容放在编译讲述。texlive 自带了一份简明手册，中文版的为 lshort-zh-cn.pdf，感谢 ctex 小组持续不断地翻译更新该手册。查阅手册的命令为 texdoc。

```
texdoc lshort-zh
```

LaTeX 排版充分体现了 LaTeX 的特点：做简单的事情很复杂，做复杂的事情很简单。

2.1　基础设置

这一小节主要内容是不需要加入额外的包时的排版功能，但是脚注会单独列出。最基本的知识可以参考手册 classes。

```
texdoc classes
```

建议先做一些练习，然后再看 classes 的手册。

默认的几个常用类为 article、report、book、letter。除了 book 的默认选项是 twoside（区分奇偶页）外，其余默认选项都是 oneside，不区分奇偶页。letter 类比较特殊，后面会单独讲用法。

article 类没有 chapter（章），book 和 report 都包含 part、chapter、section、subsection、subsubsection。用法如下

```
\part{飞秒激光器原理} %都可以有 part
\chapter{激光器原理} %article 没有 chapter
\section{激光器简介}
\subsection{激光器的历史}
\subsubsection{激光器的诞生}
```

不建议修改层数（深度），下面的语句可以修改小节的层数，

```
\setcounter{secnumdepth}{2} %默认是 3
```

1 是\section有效，2 是\subsection有效，3 是\subsubsection有效，默认是 3。

2.1.1 字号

类的选项里可以设置三种字号：10pt、11pt 和 12pt，默认是 10pt。这个字号是指一套九种字号，从小到大分别是：\tiny、\scriptsize、\footnotesize、\small、\normalsize、\large、\Large、\LARGE、\huge和\Huge。12pt 意思是\normalsize字号为 12pt，相当于小四字号。

内置字号的默认大小如下表所示，不建议修改默认字号大小，建议采用自定义字号。

tiny	5pt	6pt	6pt
scriptsize	7pt	8pt	8pt
footnotesize	8pt	9pt	10pt
small	9pt	10pt	10.95pt
normalsize	10pt	10.95pt	12pt
large	12pt	12pt	14.4pt
Large	14.4pt	14.4pt	17.28pt
LARGE	17.28pt	17.28pt	20.74pt
huge	20.74pt	20.74pt	24.88pt
Huge	24.88pt	24.88pt	24.88pt

内置字号可以用花括号括起来，仅仅对花括号里的内容生效，还是很方便的。`{\tiny tiny}`得到的结果是 tiny，其余字体不变。

anyfontsize 包

只有添加 anyfontsize 包后，设置任意字号的语句\fontsize才真正起作用。

```
\usepackage{anyfontsize}
```

任意字号的设置语句为

```
\fontsize{size}{baselineskip}\selectfont
```

字号的设置语句是字号 size 和绝对行距 baselineskip 一起设置的，后面的\selectfont必须存在才能生效。

```
\fontsize{18pt}{24pt}\selectfont
```

\fontsize语句并不改变九种内置字号的大小，内置字号也是同时设置字号和绝对行距的，因此可以使用\normalsize修改回默认字号。

可以定义新命令，比如定义五号字和小五号字：

```
\newcommand{\fontfive}{\fontsize{10.5pt}{12pt}\selectfont}
\newcommand{\fontsmallfive}{\fontsize{9pt}{11pt}\selectfont}
```

使用如同\small，可以放在花括号局域环境内使用。

```
{\fontsmallfive 小五}小五
```

小五小五

小四是 12pt，四号是 large，但是小三和三号都没有对应的内置字号。

2.1.2 行间距

设置绝对行距

```
\baselineskip=34pt %必须放在 document 环境之内
```

因为默认字号也是同时设置 baselineskip 的，所以使用默认字号\normalsize语句可以很方便地把上面语句的设置修改回来。

设置段间距使用设置长度命令\setlength，段间距是\parskip

```
\setlength\parskip{20pt}
```

这里需要注意的是，中间空白一行是分段，强制分行符不是分段，只是分行，因此不受该语句的影响。

Latex 的长度单位很丰富，最常用的是 cm，mm，pt，em。

最方便的是使用 setspace 包

```
\usepackage{setspace}
```

使用方式是

```
\begin{spacing}{1.25} %类似 word 的设置，1.25 倍行距，随字体大小变化
内容
\end{spacing}
```

2.1.3 字体

内置六种字体，花括号构成局部有效，rm 是默认字体。

```
{\rm text} {\it text} {\tt text} {\sf
text} {\sl text} {\sc text}
```

text *text* `text` text *text* TEXT

少量文字也可以写成更规范的形式，

```
\textrm{text} \textit{text} \texttt{text}
\textsf{text} \textsl{text} \textsc{text}
```

text *text* `text` text *text* TEXT

\textbf{}这样的形式支持强制换行符\\，不支持空白行分段。大量文字应该采用{\bf ...}形式。

文字加粗的要点如下

1. \bf语句可以作用在直体字体上，但是两个语句的顺序很重要，\bf先作用在文字上。

   ```
   {\bf\rm text} {\rm\bf text}
   ```

   text **text**

2. \bf语句作用在其他字体上，用简单形式一定会失效。

   ```
   {\bf\it text} {\it\bf text}
   ```

   text **text**

3. 其他字体加粗，起码是两种形式混用，顺序也很重要。前两个是对的，后两个不起作用。

   ```
   {\it\textbf{text}}
   {\bf\textit{text}}
   {\textbf{\it text}}
   {\textit{\bf text}}
   ```

   text ***text*** *text* **text**

4. 全部使用字体规范形式，这时不存在顺序问题。

   ```
   \textbf{\textit{text}}
   \textit{\textbf{text}}
   ```

   text ***text***

5. 加粗语句对 sc 字体无效。

用的最多的还是粗体\textbf{}和意大利斜体\textit{}两种情况。

更多字体可以参考下面的手册

```
texdoc psnfss
```

2.1.4 缩进

默认英文小节后第一段首行不缩进，因此专门有一个包修正这一点。

```
\usepackage{indentfirst}
```

设置首行缩进量，可以写到导言部分，\setlength是设置长度的常用语句

```
\setlength\parindent{2em} %两个标准字号的汉字
```

如果使用 CJKutf8 包，还可以使用下面的中文缩进语句

```
\CJKindent %必须写到 CJK 环境之内，xelatex 不能使用
```

悬挂缩进

```
\hangafter=1 %第一行后悬挂缩进，默认为 1，可以不写
\hangindent=2em %设置缩进量
\noindent    %首行无缩进，必须直接放在文字前
```

首行无缩进要紧跟文字，先设置悬挂缩进，后设置首行无缩进。这三句对当前段有效。

2.1.5 横向空白和纵向空白

横向空白语句为\hspace{1cm}，纵向空白语句为\vspace{1cm}。使用\hskip1cm和\vskip1cm也可以，这种方式后面文字要有空格。

强制换行符为两个反斜杠，\\。可以在强制换行符后面加长度参数定义纵向空白，\\[3mm]。

\hfill和\vfill表示横向填充空白和纵向填充空白。

```
\hfill 前面自动填充空白
```

前面自动填充空白

纵向填充空白请自行练习。

2.1.6 对齐

默认为两边对齐。

1. 少量文字的一行内居中

   ```
   \centerline{少量文字}
   ```

2. 大段文字的居中使用 center 环境，这种方式上下有附加间距。

```
\begin{center}
内容
\end{center}
```

3. 居中设置对下面都生效，或者对花括号内局部有效，没有附加间距，也常用于图表环境。

```
{\centering 大量文字}
```

4. 左对齐和右对齐

```
{\flushleft 文字，支持换行} %左对齐
{\flushright 文字，支持换行} %右对齐
```

也可以采用环境形式

```
\begin{flushleft} %或 flushright
文字
\end{flushleft}
```

5. 左对齐和右对齐

```
{\raggedright 文字，支持换行} %右参差不齐是左对齐
{\raggedleft 文字，支持换行} %左参差不齐是右对齐
```

6. 局部右对齐，对齐点是不缩进的文字起始位置

```
\llap{abc右对齐}
\llap{右对齐}
```

7. 局部左对齐，对齐点是不缩进的文字起始位置

```
\rlap{左对齐abc}
\rlap{左对齐}
```

8. 局部居中对齐，对齐点是不缩进的文字起始位置

```
\clap{abc居中}
\clap{居中}
```

三个局部对齐语句在格式设置代码中非常有用。

9. 默认页面是上下分散对齐的，如果存在大量图表和长公式时，一页能放的内容不多，分散太开并不好看，可以设置为顶部对齐，下面留一些空白。

```
\raggedbottom %顶部对齐
\flushbottom %上下分散对齐
```

10. 垂直居中需要加载包 midpage

```
\usepackage{midpage}
```

```
\begin{midpage}
垂直居中内容
\end{midpage}
```

2.1.7 分页

强制分页可以用两种方式

```
\clearpage %推荐
\newpage
```

无论哪种方式，如果后面没有内容，都不会产生新的一页。

2.1.8 边注

最好先设置边注的宽度，不能超过右边距

```
\setlength\marginparwidth{4em} %使用 em 对中文更好一些
```

还可以修改边注与文字间的间距，这个适合宽边注的书的排版。一般情况可以不修改。

```
\setlength\marginparsep{3mm} %默认 10pt
```

文字旁边可以加入边注，只需要\marginpar{边注是自动换行的}。边注是自动跟随文字的，可以插图和公式，equation 公式环境（带编号的公式）也可以。插入图与文字等效，跟随文字效果比较好。\marginpar{\includegraphics[width=1cm]{fig/tju.pdf}}

边注是自动换行的

插入公式会略微往下一点。\marginpar{\[f(x)\]}这里例子显示的效果不是因为上面的图导致放不下，不插入图片，公式的位置也靠下。插入图、表和公式这些情况都只适合宽边注的排版，有些书将大部分插图都放在边注里面了，效果也很好。

$f(x)$

边注的时候采用行间公式文字跟随效果最好，因为行间公式在 LaTeX

$f(x)$ 里被视为文字。`\marginpar{\centering$f(x)$}`。

对于书且使用默认的 twoside 选项，边注会按奇偶页自动变换位置。奇数页在右边，偶数页在左边。article 类，边注默认在右边，可以使用下面的语句改到左边

```
\reversemarginpar
```

这本书采用 geometry 包进行页面设计，选择使用装订线，边注从左边距开始，装订线宽度部分保留空白。 从左边距开始

修改回到右边的语句为

```
\normalmarginpar
```

有一个专门的包，marginnote，

```
\usepackage{marginnote}
```

2.1.9 图文混排

不规则分栏或者图文混排可以使用 minipage 环境。

```
\begin{minipage}[b]{3cm}
\begin{figure}[H]
\centering
\includegraphics[width=3cm]{fig/tju.pdf}
\caption{天大校徽}
\end{figure}
\end{minipage}
\begin{minipage}[b]{3cm}
\begin{table}[H]
\centering
\caption{简单表格}
\begin{tabular}{l l l}\toprule
a &b &c\\
\midrule
1 &2 &3\\
4 &5 &6\\
\bottomrule
\end{tabular}
\end{table}
\end{minipage}
```

```
\begin{minipage}[b]{6cm}
使用figure和table环境，用minipage环境做出不等分的分栏效果时，也需要
float 包。
\end{minipage}
```

图 2-1: 天大校徽

表 2-1 简单表格

a	b	c
1	2	3
4	5	6

使用 figure 和 table 环境，用 minipage 环境做出不等分的分栏效果时，也需要 float 包。

minipage 环境是等价为文字的。

后面讲的 parbox 也可以做出图文混排效果，但是不能使用 figure 和 table 环境。推荐使用 minipage 环境。

当然，如上例所示，minipage 环境也可以并排放置几个 figure 环境（带 caption）的图片，实现多图任意排版功能。

2.1.10 常用类的选项

基础类的默认选项如下所示：

article	letterpaper,10pt,oneside,onecolumn,final
report	letterpaper,10pt,oneside,onecolumn,final,openany
book	letterpaper,10pt,twoside,onecolumn,final,openright

最常用的类选项除了字号外，就是纸张大小。

纸张大小	高度	宽度
a4paper	297mm	210mm
a5paper	210mm	148mm
b5paper	250mm	176mm
letterpaper	11in	8.5in
legalpaper	14in	8.5in
executivepaper	10.5in	7.25in

默认是 letterpaper，比较小，一般会修改为 a4paper。

```
\documentclass[a4paper,12pt]{article}
```

文章默认是单栏 onecolumn，但是很多文章都是双栏格式，

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

book 类会生成独立的标题页，如果希望文章有独立的标题页，可以设置为

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

默认为纵向，横向选项为 landscape

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

默认公式居中、编号右对齐，可以用 fleqn 设置公式左对齐，fleqn 选项时公式编号还是右对齐的。做数学手册时或者自己做常用数学公式笔记时比较推荐公式左对齐，方便查阅。

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

可以用 leqno 选项设置公式编号左对齐，公式仍然居中，这样的书也曾经见到过。

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

默认 book 类是 twoside，区分奇偶页，采用下面的语句修改为 oneside，

```
\documentclass[oneside]{book}
```

默认 book 类是 openright，从右边页开始，也可以修正为 openany

```
\documentclass[openany]{book}
```

2.1.11 划线

一条 textwidth 宽度的横线

```
单线
\hrule
双线
\vskip1mm\hrule\vskip1pt\hrule\vskip1mm
前后调整纵向间距后就不太紧密了
```

单线
双线
前后调整纵向间距后就不太紧密了

\hrule与文字间距太近，需要\vskip语句调整间距。

\rule[高度]{长度}{宽度}，可以画横线和竖线，方括号里第一个参数高度是调整横线高度或竖线起始位置高度的，省略时是 0pt。

```
文字\rule[3pt]{2cm}{1pt} %横线
\rule[1pt]{1pt}{2cm} %竖线
```

文字

文字的下划线语句为

```
\underline{abch} \underline{gfd}
\underline{中文}
```

abch gfd 中文

如上，默认的下划线高度参差不齐，并不美观。最简单的包为 umoline，可以添加下划线、中间线和上划线命令。

```
\usepackage{umoline}
```

更为复杂一点的包为 ulem，包含单下划线、双下划线、波浪线、虚下划线、点下划线等多种线型。

```
\usepackage{ulem}
```

2.1.12 条目

LaTeX 自带三个条目环境，分别是 itemize，enumerate 和 description。itemize 默认是圆点开始第一层，可以嵌套四层。

设置前导符的方式如下，i 是第一层，ii 是第二层，iii 是第三层，iv 是第四层。

```
\renewcommand{\labelitemi}{\tiny$\blacksquare$}
\renewcommand{\labelitemii}{$\bullet$}
\renewcommand{\labelitemiii}{\small$\bigstar$}
```

默认间距很宽，可以改变 itemsep，写到环境里面对当前环境有效。

```
\setlength\itemsep{-5pt}
```

因为本书已经设置了第一层的间距，所以下面的代码就只修改了第二、三层的间距。

```
\begin{itemize}
\item 第一条
\item 第二条
\begin{itemize}
\setlength\itemsep{-5pt}
\item 第a条
\item 第b条
\begin{itemize}
\setlength\itemsep{-5pt}
\item 第i条
\item 第ii条
\end{itemize}
\end{itemize}
\end{itemize}
```

- 第一条
- 第二条
  - 第 a 条
  - 第 b 条
    - 第 i 条
    - 第 ii 条

itemize 环境可以使用任意的前导，采用加方括号形式即可。

```
\begin{itemize}
\item[aa] 第一条
\item[bbb] 第二条
\item 第三条
\end{itemize}
```

aa 第一条
bbb 第二条
- 第三条

使用 enumitem 包可以方便设置 item 项，后面的设置采用了该包的语句。

```
\usepackage{enumitem}
```

本书修改了第一层的间距。

```
\setenumerate[1]{itemsep=0pt,partopsep=0pt,parsep=\parskip,topsep=5
pt}
\setitemize[1]{itemsep=0pt,partopsep=0pt,parsep=\parskip,topsep=5pt}
\setdescription{itemsep=0pt,partopsep=0pt,parsep=\parskip,topsep=5pt
}
```

enumitem 包有大量的设置语句帮助设置 itemize、enumerate 和 description 环境的格式，比基本设置语句方便得多。

enumerate 环境是自动编号的条目。

```
\begin{enumerate}
\item 第一条
\item 第二条
\item 第三条
\end{enumerate}
```

1. 第一条
2. 第二条
3. 第三条

换成其他方式编号语句如下

```
\begin{enumerate}[label=\alph*.]
\item 第一条
\item 第二条
\item 第三条
\end{enumerate}
```

a. 第一条
b. 第二条
c. 第三条

```
\begin{enumerate}[label=\roman*.]
\item 第一条
\item 第二条
\item 第三条
\end{enumerate}
```

i. 第一条
ii. 第二条
iii. 第三条

description 则用于定义

```
\begin{description}
\item[italy] 意大利 %英文的定义是粗体，中文需要额外设置
\item[自然数] 1,2,3,...
\item[有理数] 整数和分数
\item[实数] 实数的定义为...
\item[特别长的名词情况] 与itemize环境对齐方式不同
\end{description}
```

italy 意大利
自然数 1,2,3,...
有理数 整数和分数
实数 实数的定义为...
特别长的名词情况 与 itemize 环境对齐方式不同

设置中文黑体样式

```
\begin{description}[font=\bfseries]
\item[italy] 意大利
\item[自然数] 1,2,3,...
```

```
\item[有理数] 整数和分数
\item[特别长的名词情况] 与itemize环境对齐方式不同
\end{description}
```

italy 意大利
自然数 1,2,3,...
有理数 整数和分数
特别长的名词情况 与 itemize 环境对齐方式不同

除了上面的三个环境外，verse 和 quote、quotation 三个环境也可以有 item 项，此时没有前导符号。一般情况下这三个环境使用强制换行符而不是 item 项。

2.2 页面设置包 geometry

默认纸张大小是 letterpaper，默认页面布局如下：默认的文字宽度 textwidth，10pt 是 345pt，11pt 是 360pt，12pt 是 390pt。默认的文字高度 textheight，11pt 是 38 baselineskip，12pt 是 36 baselineskip。默认的左右留白比较宽，用基本语句修改页面布局比较麻烦，一般采用 geometry 包。

页面布局

页面设置包为 geometry，在加载包时就可以设置一些常用参数

```
\usepackage[a4paper,bindingoffset=1cm,top=2cm,bottom=2cm,left=2cm,
right=2cm]{geometry}
```

a4paper 是纸张大小，geometry 支持更多的纸张大小。**纸张大小只能导言中设定一次。** bindingoffset 设置装订预留宽度，然后设置上下左右边距。如果设定了装订预留宽度，实际的左右留白是左右边距加上装订预留宽度（书是分奇偶页的，oneside 就是左边距相加）。

包的选项参数相当于默认的页面设置。我们可以在文章中重新设置页面参数，比如上下左右边距。

```
\newgeometry{left=5cm,right=5cm,top=5cm,bottom=5cm}
\savegeometry{L2} %L2 是自定义的名字，用于重复使用页面设置
```

\newgeometry{}语句后的页面会改变为新的布局。重新回到导言的布局只需要下面的命令

```
\restoregeometry
```

再次使用 L2 布局只需要再次加载即可

```
\loadgeometry{L2}
```

横排

横排可以在两个地方设置，

```
\documentclass[landscape,12pt]{article}
```

或者

```
\usepackage[landscape,paper=a4paper,...]{geometry}
```

如果仅仅需要某一页或几页局部横排，不能通过\newgeometry{}修改为横排，所以需要 lscape 包。

```
\usepackage{lscape}
```

使用 landscape 环境

```
\begin{landscape}
需要横排的内容
\end{landscape}
新的一行 %环境外的文字会自动分页到下一页竖排
```

此时文字图表内容横排，页眉页脚的位置不变，即布局不变。

文章内改变纸张大小

一些包括图纸的书很需要这个功能，必须使用 pdfpages 包，geometry 包不能在文章中局部改变 papersize。

```
\usepackage{pdfpages}
```

用下面的语句改变 papersize，papersize 不能用\newgeometry语句在文档中间修改。

```
\pdfpagewidth=40cm
\pdfpageheight=30cm
```

此外，还必须修改 layoutsize（文字、装订线宽度、上下左右边距（包括页眉页脚部分）和文字宽度的总和），否则页码和文字都不能自动扩展或收缩

到新纸张大小。默认就是 layout 大小和纸张大小相同。比较复杂的页面布局，layout 尺寸可以小于纸张尺寸，这种情况比较少见。

```
\newgeometry{layoutwidth=40cm,layoutheight=30cm,left=2cm,right=2cm,
top=2cm,bottom=2cm}
```

这样可以获得一个页面布局正确的结果，页码和文字都在适当的位置。

再次回到原来的纸张大小时，除了将 papersize 重置外，还必须修改 layout。恢复到默认设置、加载已经保存的设置或者重新设置都可以重置 layout。

```
\pdfpagewidth=21cm\pdfpageheight=29.7cm %纸张大小改回到 A4
\restoregeometry %回到默认设置
%如果需要也可以设置新布局
\newgeometry{layoutwidth=21cm,layoutheight=29.7cm,left=2cm,right=2cm
,top=2cm,bottom=2cm}
```

这样书中某页是大图纸情况，可以加载任意尺寸的大图和说明文字，页码可以自动编码且在布局规定位置显示，比如说底部居中。

从这个例子可以看出 papersize 和 layoutsize 的关系。如果不修改 papersize，只修改 layoutsize，比如扩大 layoutsize，内容将按 layoutsize 排布，但是显示按 papersize 显示，内容会被切掉。

如果只修改 papersize，比如扩大 papersize，layoutsize 保持不变，则文字内容会按 layoutsize 排布，只占据 pdf 页面的一部分空间，包括页码都是按原来的 layoutsize 居中，实际显示效果页码的位置会很偏。

有一种特殊的情况可以不修改 layoutsize，只修改 papersize，就是不存在页眉和页脚时。可以用`parbox{大的宽度}{长文字}`来做到文字横向排布的扩展。大的图纸也可以放在大的 papersize 里面，但是不能添加 caption，因为 caption 跟页码（页眉页脚）一样，是按原来的 layoutsize 默认居中的，也是按原来的 layoutsize 换行的。当然，如果使用`\caption{parbox{大的宽度}{长文字}}`也可以手动逐个地解决问题。

总之，对于放置大图纸、大表格，papersize 是一定需要修改的。

最后还需要特别指明的是，局部修改 papersize 可以做到单页横排效果，但是与上面介绍的局部横排的显示效果不同，此时页眉页脚布局是横排的。

2.3 颜色包 xcolor

默认只有七种颜色，black、white、red、blue、cyan、green、yellow。加载 xcolor 包默认颜色扩展到 19 种。

```
\usepackage{xcolor}
```

xcolor 包允许用冒号调颜色深浅

```
\pagecolor{yellow!5} %设置背景色为浅黄，数值范围 0 到 100
```

也允许用多重冒号混色，颜色设置语句可以放在花括号局部生效

```
{\color{blue!70!black}blue}^^I
```

blue

wave 参数用可见光波长定义颜色，数值范围是 363–814

```
{\color[wave]{633}633nm laser}
```

633nm laser

默认模式是 rgb，但是内置了很多颜色名，需要加载包时设置选项，比如说

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

svgnames 内置了许多颜色名，比如 DarkGreen 这样的。手册后面有各个参数情况下的调色板，对照看选一个自己喜欢的组合即可。

自定义颜色的语句为

```
\definecolor{bar}{rgb}{0.5,0.3,0.3}
{\color{bar} define}
```

define

大段文字情况下，用上面的加花括号的方式局部改变文字颜色很方便。小段文字，使用\textcolor{颜色名}{文字}语句更方便。

```
\textcolor{red}{textcolor is
red\\强制换行符}默认黑色
```

textcolor is red
强制换行符默认黑色

\textcolor命令支持强制换行符，但是不支持空白行分段。大段文字更改颜色，用上面说的花括号构成局域环境方式。

使用参数用下面的例子

```
\textcolor[wave]{633}{textcolor is red}
```

textcolor is red

2.4 盒子 box

这一小节的内容需要 xcolor 包。

mbox 和 fbox

最简单的 box 是 mbox 和 fbox，mbox 没有框，fbox 有文本框

```
\mbox{text}和\fbox{text}
```

text和text

colorbox

fcolorbox 是可以定义框颜色和背景色的 box

```
\fcolorbox{框颜色}{背景色}{文字}
```

```
\fcolorbox{red}{gray!20}{text}
```

text

不加前面的 f 就是 colorbox，没有边框只有背景色

```
\colorbox{yellow}{text}
```

text

rotatebox

可以将 box 旋转任意角度，如将 text 旋转 30 度

```
\rotatebox{30}{text}
```

text

比较下面的结果

```
\colorbox{yellow}{\rotatebox{30}{text}}  %先旋转后涂色
\rotatebox{30}{\colorbox{yellow}{text}}  %先涂色后旋转
```

text text

box 里的换行

box 不能自动换行，也不支持强制换行，需要换行必须使用 vbox 或 parbox，

```
first \vbox{abc\\abcdefghi}
```

abc
first abcdefghi

改为下对齐

```
first \vtop{abc\\abcdefghi}
```

first abc
abcdefghi

parbox 的语句为\parbox{宽度}{内容}

```
\parbox{\textwidth}{large auto large auto large auto large auto large auto large auto large auto large auto large auto large auto}
```

对上面的 box 涂色，先 parbox 后涂色

```
\colorbox{yellow}{\parbox{\textwidth}{large auto large auto large auto large auto large auto large auto large auto large auto large auto large auto}}
```

large auto large auto large auto large auto large auto large auto large auto large auto large auto large auto

parbox 一共有四个参数，因为高度可以自动按内容扩展，所以高度可以省略。**parbox 的宽度必须设置**。另外两个参数是控制对齐的，可选择 b 或 t 或 c。

```
\parbox[外部对齐位置][高度][内部文字位置]{宽度}{内容}
```

```
first \parbox[t][1cm][b]{6cm}{内外对齐方式可调} outside
```

first 内外对齐方式可调 outside

填表时可以用 parbox 换行。

makebox 和 framebox

mbox 和 fbox 的复杂版本是 makebox 和 framebox，常用于版面设计

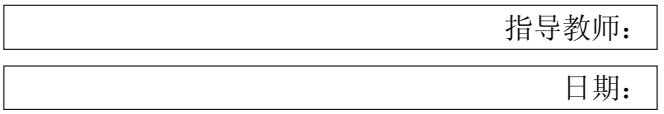
```
\makebox[0.7\textwidth][r]{指导教师：}\\[2mm]
\makebox[0.7\textwidth][r]{日期：}
```

指导教师：

日期：

使用 framebox 可以使得前面的 makebox 看得更清楚

```
\framebox[0.7\textwidth][r]{指导教师：}\\[2mm]
\framebox[0.7\textwidth][r]{日期：}
```



升降 box 的位置，可以用于图文混排的微调

```
内容\raisebox{2mm}{上升文字}内容\raisebox{-2mm}{上升文字}内容
```

内容上升文字内容上升文字内容

framed

无论是 vbox、parbox、makebox 等，都无法内容分页，box 是一个整体，如果放不下的话会整个移到下一页，都放不下就显示不全。需要 box 自动分页可以使用 framed 包。framed 包非常适合多页大型表格的填写。

```
\usepackage{framed}
```

放在 framed 环境内的内容很长也会自动分页，并且每页都自动添加边框

```
\setlength\FrameSep{2mm} %设置内容距离边框的间距，必须放环境外
\begin{framed}
长内容
\end{framed}
```

framed 环境的边框线宽为\FrameRule。加一条横线时，因为横线默认位置是文字起始位置，默认长度是文字宽度，因此需要横线水平位置左移\FrameSep加上\parindent，线长度等于\textwidth加上 2 倍的\FrameSep。横线的高度位置一般需要略微调整一下比较好看。

```
\begin{framed}
长内容

\vskip-5pt\hskip-\FrameSep\hskip-\parindent
\rule{\textwidth+2\FrameSep}{\fboxrule}

长内容
\end{framed}
```

长内容

长内容

但是如果重新设置了\FrameSep，此时增加的\FrameSep长度会以修改\textwidth的长度为代价，整体框宽度不改变。因此并不需要修改上面的画横向的语句。

```
\setlength\FrameSep{2mm}
\begin{framed}
长内容

\vskip-5pt\hskip-\FrameSep\hskip-\parindent
\rule{\textwidth+2\FrameSep}{\fboxrule}

长内容
\end{framed}
```

长内容

长内容

有很多分页大型表格是这样的样式，用 framed 包来做非常合适。分页表格包 longtable 不能单元格自动分页。

2.5 插图

2.5.1 基本用法

不添加任何包就可以插图，但是一般都需要添加 graphicx来实现图片大小等基本功能。

```
\usepackage{graphicx}
```

单纯的\includegraphics[参数]{图片名}是文本，figure 构成独立的图片环境。一般情况都应该指定图片大小，使用宽度或高度都可以，另一个按比例变化。但是除非特殊情况，不要同时指定宽或高，否则会扭曲图片。

```
\begin{figure}[ht!] %ht! 表示当前位置或 top
\centering %一般都加这句使得图片居中
\includegraphics[width=4cm]{fig/tju.pdf}
\caption{天大校徽}
\end{figure}
```

\begin{figure}[位置参数]推荐使用[ht]!，表示图片放在当前位置或下一页 top 位置，[h]! 太严格了，做不到会自动转成[ht]!。b 表示底部，p 表示单独成页。这两个参数偶尔也很有用。

图 2-2: 天大校徽

图片自动编号，如果仅仅给出 caption，不加编号，可以加星，即\caption*{}。

缩放图片还可以使用 scale 参数

```
\includegraphics[scale=0.3]{fig/tju.pdf}
```

旋转角度可以使用 angle 参数

```
\includegraphics[scale=0.3, angle=30]{fig/tju.pdf}
```

对图片加框使用 fbox，因为\includegraphics[参数]{图片名}是文本，可以放到文本框内

```
\fbox{\includegraphics[width=3cm]{fig/tju.pdf}}
```

加一个正框，里面图片旋转，需要 framebox，framebox 同时设置宽高的时候用圆括号。

```
\framebox(3cm,3cm){\includegraphics[width=3cm,
angle=30]{fig/tju.pdf}}
```

关于图片名有几点需要特别说明。

1. latex 编译方式只能使用 eps 格式的图片。
2. pdflatex 和 xelatex 编译方式可以使用 jpg、png 和 pdf 格式的图片，可以不加扩展名（linux 下），但是不能使用 eps 和 ps 格式的图片。
3. 图片名有点时，如果出现编译错误（与版本相关），使用花括号将. 扩展名前的部分括起来。可以不加扩展名。

```
\includegraphics[width=4cm]{fig/{t.j.u}.pdf}
```

4. 图片名有空格时，使用双引号将. 扩展名前的部分括起来。

```
\includegraphics[height=4cm]{"t j u".pdf} %此处设定图片高度
```

其中第 3 条，估计用双引号也可以，texlive 2021 版没有这个编译错误了，所以也无法调试，最早也没有这个错误，可能是某版本的短暂 bug。推荐不省略扩展名，可以准确分辨出同名不同格式的图片并正确加载。

2.5.2 设置 caption 形式

设置 figurename，加在导言或文件开头，对所有图片都生效，这里加在里面，只对当前图片生效。

```
\begin{figure}[ht!]
\renewcommand{\figurename}{Fig.} %加在 figure 环境内只对当前图片有效
\captionsetup{labelsep=space} %使用空格取代冒号
\centering
\includegraphics[width=4cm]{fig/tju.pdf}
\caption{天大校徽}
\end{figure}
```

Fig. 2-3 天大校徽

更复杂的 caption 结构设计，可以参看手册

```
texdoc caption
```

默认 caption 间距就挺好，修改用下面的语句，负号更容易看出区别。

```
\abovecaptionskip=-5pt %直接设置
\belowcaptionskip=-5pt
```

或者

```
\addtolength\abovecaptionskip{-5pt} %基础上加减
\addtolength\belowcaptionskip{-5pt}
```

双 caption 可以使用 bicaption 包。

```
\usepackage{bicaption}
```

设置中英文 caption，对于已经设置好的中文 caption 格式，只使用第二个语句设置 second 就行了。**这两句不能放在 figure 环境里使用**，必须放在外面。

```
\captionsetup[figure][bi-first]{name=图} %放在外面
\captionsetup[figure][bi-second]{name=Figure}
\begin{figure}[ht!]
\centering
\includegraphics[width=4cm]{fig/tju.pdf}
\bicaption{天大校徽}{The school badge of Tianjin University}
\end{figure}
```

图 2-4: 天大校徽
Figure 2-4: The school badge of Tianjin University

标签放在 bicaption 的任何一个 caption 内容里（花括号内）都可以正确引用图的序号。

2.5.3 子图

子图有两个包，subfigure 和 subfig，subfig 据说更新一点，但是手册是同一年的。

subfigure

加载子图包 subfigure。

```
\usepackage{subfigure}
```

使用框架是

```
\begin{figure}[ht!]
\centering
\setcounter{subfigure}{0} %多次使用 subfigure 时重新编号
\subfigure[a is first]{\includegraphics[width=3cm]{fig/tju.pdf}}\
hspace{5mm}
\subfigure[标签加载后面\label{f2}]{\includegraphics[width=3cm]{fig/
tju.pdf}}\\
\subfigure[]{\includegraphics[width=3cm]{fig/tju.pdf}}\hspace{5mm}
\subfigure[]{\includegraphics[width=3cm]{fig/tju.pdf}}
\caption{天大校徽。\label{f-total}}
\end{figure}
```

引用图使用图\ref{f2}。对于图来说，**\label{标签名}要加在 caption 内容里，即花括号内**，subfigure 的 caption 内容的位置是方括号内，所以标签要加在方括号里面，这样可以直接引用子图序号。

图的 caption 和子图的 caption 可以同时加标签，名字不同不会互相影响的。比如图\ref{f-total}。

subfig

加载 subfig 包

```
\usepackage{subfig}
```

子图结构如下

```
\begin{figure}
\centering
\subfloat[one]{\label{sub-1}
\includegraphics[width=5cm]{fig/tju.pdf}}
\subfloat[two]{\label{sub-2}
\includegraphics[width=5cm]{fig/tju.pdf}}
\caption{tju}
\end{figure}
```

subfig 不需要重新设置子图编号，每次会重新开始，这点比 subfigure 方便一些。

(a) one (b) two

图 2-5: tju

subfigure 包和 subfig 包测试的时候不能同时加载，否则会报错。

默认子图是 abcd，修改是用下面的语句，比如修改为小写罗马数字，

```
\renewcommand{\thesubfigure}{\roman{subfigure}}
```

2.6 表格

表格的基础环境是 tabular，独立环境是 table，tabular 的手册是 array。

```
texdoc array
```

2.6.1 基本功能

表格基础包是 array，但是最基本的表格功能不需要加载 array 包。除非做一个极简表格，否则 array 包还是要加载的。

```
\usepackage{array} %加载后才可以使用一些功能
```

表格的基本环境是 tabular，tabular 环境是文本。表格的独立环境是 table，table 里可以有 caption，一般表格的 caption 在表格上面。

tabular 的参数，l 表示左对齐，r 表示右对齐，c 表示居中对齐，p{长度} 表示设置这一列的宽度（必须有单位），内容超出自动换行。m{长度} 类似，但是 m 的结果是上下居中对齐，p 是顶端对齐，b 也可以，表示底部对齐。一般情况下推荐用 m。**有多少列就需要写多少个参数**。

表格内容，一行内的单元格由 & 分开，每行用强制换行符\\分开。

```
\begin{table}[ht!] %当前位置
\caption{这是一个简单表格，caption默认居中。}
\centering
\begin{tabular}{l r c p{2.5cm} m{2.5cm}}\hline %一条横线
name &value &description &功能 &说明\\
 \hline
苹果 &1元 &水果 &对健康生活很有好处 &对健康生活很有好处\\
\hline %一条横线
\end{tabular}
\end{table}
```

表 2-2 这是一个简单表格，caption 默认居中。

name	value	description	功能	说明
苹果	1 元	水果	对健康生活很有好处	对健康生活很有好处

列方向如果需要线，可以在 tabular 的参数表相应位置加竖线。横线还需要\hline语句。

```
\begin{tabular}{|l|l|l|}\hline
a&b&c\\ \hline
e&f&g\\ \hline
h&i&j\\
\hline
\end{tabular}
```

a	b	c
e	f	g
h	i	j

同时改变横线和竖线的线宽可以用下面的语句，如果写到 table 环境内只对当前 table 有效。

```
\setlength{\arrayrulewidth}{5pt}
\begin{tabular}{|l|}
\hline A \\ \hline
\end{tabular}
```

A

可以通过下面的设置拉长列宽和行高，如果写到 table 环境内只对当前 table 有效。

```
\setlength{\extrarowheight}{10pt} %行高
\setlength{\tabcolsep}{15pt} %列宽
\begin{tabular}{|l|l|l|}\hline
a&b&cccc\\ \hline
e&f&g\\ \hline
h&i&j\\
\hline
\end{tabular}
```

a	b	cccc
e	f	g
h	i	j

>{前语句}对齐方式<{后语句}设置某列前置语句和后置语句，下面的例子前面加第，后面加日，表格内只需要输入数字，重复的文字不需要再输入。波浪号控制间距自动调整到美观。

```
\begin{tabular}{>{第~}c<{~日} l}\toprule
1 &看书\\
2 &计算\\
3 &总结\\
\bottomrule
\end{tabular}
```

第 1 日	看书
第 2 日	计算
第 3 日	总结

2.6.2 三线表

科技论文中常用的是三线表，使用三线表包 booktabs

```
\usepackage{booktabs} %三线表
```

三线表的三条线分别是顶部粗线\toprule、中间细线\midrule和底部粗线\bottomrule。直接在后面加方括号设置宽度，可以修改线宽。默认设置就很漂亮了，下面的例子仅仅为了表示可以这么做。

```
\centering
\begin{tabular}{l r c m{2.5cm}}\toprule[2pt]
name &value &description &功能\\ \midrule
苹果 &1元 &水果 &对健康生活很有好处\\
荔枝 &15元 &水果 &上火\\
```

```
\bottomrule
\end{tabular}
```

name	value	description	功能
苹果	1 元	水果	对健康生活很有好处
荔枝	15 元	水果	上火

booktabs 包还定义了特殊的线，线长为默认表格宽度。

```
\specialrule{线宽}{上间距}{下间距} %tabular 环境中使用
```

2.6.3 表格颜色

使用 colortbl包设置表格颜色。

```
\usepackage{colortbl}
```

或者在 xcolor 包加 table 选项

```
\usepackage[svgnames,table]{xcolor} %推荐这种，更多功能
```

然后，可以设置线的颜色

```
\arrayrulecolor{blue} %放在 tabular 环境内只对当前表格有效
```

可以设置行的背景色，而且是双色相间的

```
%rowcolors[是否加横线] 起始行奇数行颜色偶数行颜色
\rowcolors[\hline]{1}{cyan!30}{yellow!40} %第一行有颜色
\begin{tabular}{l c r l}
goods &price &quantity &total\\
apple &0 &5 &5\\
banana&1 &3 &3\\
pear &1 &3 &3\\
orange&2 &2 &4\\
\rowcolor{red} %仅下面一行换色
apple &0 &5 &5\\
banana&1 &3 &3\\
\end{tabular}
```

goods	price	quantity	total
apple	0	5	5
banana	1	3	3
pear	1	3	3
orange	2	2	4
apple	0	5	5
banana	1	3	3

使用可以\columncolor{颜色}设置列颜色，这个可以放到列的前置语句里，对整列生效。还可以使用\cellcolor{颜色}设置某个格子的背景色，放在单元格内容前。

```
\begin{tabular}{>{\columncolor{cyan!50}}l
>{\columncolor{green!50}}c r l}
goods &price &quantity &total\\
apple &0 &5 &5\\
banana&1 &3 &3\\
pear &\cellcolor{yellow}1 &3 &3\\
orange&2 &2 &4\\
\rowcolor{red} %仅下面一行换色
apple &0 &5 &5\\
banana&1 &3 &3\\
\end{tabular}
```

goods	price	quantity	total
apple	0	5	5
banana	1	3	3
pear	1	3	3
orange	2	2	4
apple	0	5	5
banana	1	3	3

2.6.4 单元格内换行 makecell 包

表格的单元格内不支持换行。使用 makecell 包可以很容易对单元格断行，并且设置对齐方式

```
\usepackage{makecell}
```

在单元格内使用语句

```
\makecell[对齐方式]{第一行内容\\第二行内容}
```

默认的与表格其他单元格竖直方向对齐方式是上下居中方式。如果双重指定对齐方式，比如外部底部对齐，内部左对齐，如下面第二行的例子，第二个左对齐需要花括号括起来。也可以加p{宽度}这样的对齐方式，但是必须用双重花括号括起来。

```
\begin{tabular}{l l l}\toprule
name &description &price\\ \midrule
apple &A is good &\makecell[l]{1 in summer good\\2 in winter}\\
\hline
apple &A is good &\makecell[b{l}]{1 in summer good\\2 in
winter}\\ \hline
apple &A is good &\makecell[b{{p{2cm}}}]{1 in summer good\\2 in
winter}\\
\bottomrule
\end{tabular}
```

name	description	price
apple	A is good	1 in summer good 2 in winter
apple	A is good	1 in summer good 2 in winter
apple	A is good	1 in summer good 2 in winter

2.6.5 同行合并列

在同一行中合并 n 列的语句为\multicolumn{n}{对齐方式}{合并后单元格内容}。例子中 Item 这里相当于合并了两列，与下面居中对齐。\cmidrule{第n列-第m列}是三线表包中从第 n 列到第 m 列划一条细横线的语句。不使用三线表包时，\cline{第n列-第m列}用法相同。

```
\begin{tabular}{l l r}\toprule
\multicolumn{2}{c}{Item} &\\ \cmidrule{1-2} %1–2 列划线
name &description &price\\ \midrule
```

```
A &A is good &1\\
\bottomrule
\end{tabular}
```

	Item	
name	description	price
A	A is good	1

2.6.6 同列合并行 multicol 包

同列合并行需要 multirow 包。

```
\usepackage{multirow}
```

下面的例子，相当于合并了 3 行，默认对齐方式是竖直居中

```
\begin{tabular}{l l l}\toprule
序号 &原子 &分类\\ \midrule
1 &Li &\multirow{3}{5cm}{第一主族}\\
2 &Na &\\
3 &K &\\ \midrule
4 &Be &\multirow[b]{3}{5cm}{第二主族}\\ %改为底部对齐
5 &Mg &\\
6 &Ca &\\
\bottomrule
\end{tabular}
```

序号	原子	分类
1	Li	
2	Na	第一主族
3	K	
4	Be	
5	Mg	
6	Ca	第二主族

2.6.7 表格脚注

表格的脚注希望显示在表格下面，而不是页面底部，需要加载包threeparttable。

```
\usepackage{booktabs}
\usepackage{threeparttable} %表格内脚注
```

代码如下

```
\begin{table}[h]
\begin{center}
\begin{threeparttable}
\caption{表头脚注\tnote{\dag}}
\begin{tabular}{l l l}\toprule
name &description &price\\ \midrule
apple\tnote{1} &A is good &1\tnote{a}\\
\bottomrule
\end{tabular}

\begin{tablenotes}
\footnotesize
\item [\dag] 表头说明
\item[1] 水果
\item[a] 单位：元
\end{tablenotes}
\end{threeparttable}
\end{center}
\end{table}
```

表 2-3 表头脚注[†]

name	description	price
apple[1]	A is good	1[a]

[†] 表头说明
[1] 水果
[a] 单位：元

这种方式可以满足大部分表格脚注的要求，但是没有超链接。因为脚注跟随表格，所以超链接用处也不是很大。

2.6.8 跨页表格 longtable

使用 longtable 包可以很方便的使得表格跨页。

```
\usepackage{longtable}
```

longtable 环境为

```
\begin{longtable}{l}
\caption{环境内可以加caption}\\
\endhead %每页都有 caption，否则只有第一页有
表格内容
\end{longtable}
```

caption 放在前面，每页都会重复 caption。如果希望后面的 caption 不同，比如添加 (continue)，则需要写成下面的样式

```
\caption{caption的内容}\\
\endfirsthead %第一页 caption 是上面花括号里的
\caption{caption的内容(continue)}\\
%这里还可以放每页重复的表格行
\endhead %其他页的 caption 是上面花括号里的
```

longtable 环境里面不能添加\centering，有些表格设置语句也不能添加。比如要使得列间距扩大，要写为下面的形式

```
\begin{longtable}{@{\extracolsep{10pt}} m{1cm}| m{14cm} }
```

longtable 可以使得表格跨页，但是并不能使得单元格内容跨页。

可以定义\NewLine并使用它手动分页。

```
\newcommand\NewLine{\setlength\parfillskip{0pt}\tabularnewline}
```

threeparttablex 包是将 threeparttable 与 longtable 功能结合的拓展包。

2.6.9 参考书和一些有用的包

表格有本集大成的书，Typesetting tables with LaTeX，作者是 Herbert Voss，2011 年。基本常用用法都罗列出来了。

一些可能有用的包如下：

- datatool 包，从数据文件导入数据方式做表
- tabularx 包，设置表格宽度
- widetable 包，与 tabularx 包接近，根据表格宽度自动计算列间距。

- diagbox 包，画斜线表头的。slashbox 已经不再 texlive 里了。

2.7 超链接包 hyperref

做复杂的事情很简单，这点在排版这一章中体现在超链接包 hyperref 上，该包尽量放在其他包后面，但是 glossaries 包例外。加载包的语句为

```
\usepackage[unicode,colorlinks,linkcolor=blue]{hyperref}
```

中文环境推荐 CJKutf8 包，而不是 CJK 包，这样自动生成的 pdf 文件书签中文显示不会出现乱码。

包的参数最常用的是 unicode：设置字符集；colorlink：链接带颜色，参考文献、索引等有一套默认的标准颜色设置；linkcolor=blue：设置 label 的引用链接为蓝色，因为默认是红色，所以如果觉得红色太醒目，设置为蓝色比较好看。

默认自动包含目录标题超链接（也可以修改为页码超链接）、index、参考文献、label 的引用、网址、文件等超链接，并且自动生成 pdf 文件书签。比较常用的包选项设置为：

pagebackref=flase/true	参考文献带页码超链接，默认无
hyperindex=true/false	默认索引超链接
hyperfootnotes=true/false	默认脚注超链接
linktoc=section/page/all/none	目录默认小节标签超链接
linktocpage=false/true	目录仅页码超链接，默认是标题
colorlinks=false/true	超链接显示不同颜色
linkcolor=red	设置文字超链接颜色，默认红色
citecolor=green	参考颜色超链接颜色，默认绿色
filecolor=cyan	网址和文件超链接颜色，默认青色
runcolor=filecolor	运行超链接（视频等），默认同文件
urlcolor=magenta	网址超链接颜色为品红
hidelinks	隐藏颜色和边框
bookmarks	pdf 自动书签
bookmarksnumbered	pdf 书签带章节号

除了默认外，图表编号、公式编号和需要引用的地方都需要通过语句\label{标记名}设置标记，然后通过\ref{标记名}引用，也可以通过\pageref{标记名}引用页码，公式一般用\eqref{标记名}引用，在数学公式一

章中还会介绍。标记名不能数字开头，不带点，推荐使用连字符。pdflatex 编译时不能是中文，xelatex 编译是标记名可以使用中文。

下面介绍常用的例子。

章节的标记和引用

不在任何图表公式环境内的标记视为对章节的标记，所以标记章节时放在标题花括号内外都可以。其他标记都必须放在环境内，否则会误标记到章节。

```
\section{第一节\label{sec-1}}
\subsection{第二节}\label{sec-2}

第\ref{sec-1}节在\pageref{sec-1}页
第\ref{sec-2}节在\pageref{sec-2}页
```

图表的标记和引用

图表的 label 要添加到图表环境里或者 caption 里面，引用时使用\ref{标记名}获得图表的编号，使用\autoref{标记名}获得带 caption name 的结果。

```
\begin{figure}
\caption{aaa\label{fig-1}}
\end{figure}

\begin{table}\label{table-1}
...
\end{table}

图\ref{fig-1}在\pageref{fig-1}页，
\autoref{table-1}在\pageref{table-1}页。
```

如果不修改图表的 figurename 和 tablename，这样引用的结果没有问题。如果修改了，就需要重新定义自动引用名。

```
\renewcommand{\figureautorefname}{图}
\captionsetup{labelsep=doublespace}
\begin{figure}[ht!]
\renewcommand{\figurename}{图}
\centering
```

```
\includegraphics[width=4cm]{fig/tju.pdf}
\caption{天大校徽\label{tju-logo}}
\end{figure}
使用\lstinline!\autoref{tju-logo}!得到\autoref{tju-logo}，使用\
lstinline!图\ref{tju-logo}!得到图\ref{tju-logo}。
```

图 2-6 天大校徽

使用\autoref{tju-logo}得到图 2-6，使用图\ref{tju-logo}得到图2-6。

网页、文件和运行程序

网页有两种形式，一种是直接显示网址，点击网址文字可以跳转到默认浏览器打开链接。

```
\url{www.baidu.com}
```

还有一种是设置点击标识，可以是文字或图片，点击标志可以跳转到默认浏览器打开链接。

```
\href{www.baidu.com}{百度}
\href{www.tju.edu.cn}{\includegraphics[width=1cm]{fig/tju.pdf}}
```

也可以跳转到文件，比如 pdf 文件，注意最好给出全路径文件名，这样不容易出错。pdflatex 编译时全路径文件名不能是中文，否则使用 xelatex 编译。

```
\href{全路径文件名}{链接标识名}
```

还可以使用 run 来播放视频，下面的例子全部是仅仅 linux 下测试有效。run 语句表示运行 mplayer 命令播放文件，windows 系统应该需要给出播放器的命令行命令。**run 语句是否可行与 pdf 阅读器相关**。即使 linux 下，不同 pdf 阅读器的代码测试结果也是不同的，xpdf 阅读器结果全部正

确，qpdfview 则只能用 mplayer 打开视频，acroread 则全部不行。所以 run 方式是否可行深度依赖于系统和 pdf 阅读器。

```
\href{run:mplayer 全路径文件名}{视频}
```

也可以使用 run 来运行程序

```
\href{run:全路径可运行程序名}{程序}
```

其他命令也可以用 run 打开文件，或者给出全路径文件名，或者是当前目录下的文件名。

```
\href{run:xloadimage 当前目录下的文件名}{打开}\\
\href{run:xloadimage 全路径文件名}{open}
```

2.8 文章 article 类

文章类默认可以添加题目、作者、时间，用\thanks{脚注内容}加载题目里的脚注。

```
\title{You Have Only One Life}
\author{作者1 \and 作者2\thanks{email}}
\date{} %取消日期
\maketitle %显示题目
```

默认是有日期的，也即不写\date语句，会采取默认结果

```
\date{\today} %每次编译时改变
```

指定日期要填写日期

```
\date{2021年10月1日}
```

默认的题目等字号都有些太大，仅仅是自己写文章用，可以手动直接修改

```
\title{\large\bf You Have Only One Life\vskip-0.5em}
\author{\small 作者1 \and \small 作者2\thanks{email}}
\date{\vskip-0.5em\small 2021年10月1日}
```

从例子可以看出，这是一个很繁琐的工作。注意：调整纵向间距的语句一定要放在题目里和日期里，不要放在作者里，因为 maketitle 的原始代码里作者是以表格的方式构成的。这也是为什么作者的字体大小的改变需要逐项都修正的原因，作者很多时就非常麻烦了。杂志社做模板时一般都是重新定

义 maketitle 命令。自己做模板直接按照 maketitle 命令的定义方式做效果更方便一些。具体可以参照 classes 文档。

```
texdoc classes
```

文章类的默认有摘要环境

```
\begin{abstract}
Abstract text ...
\end{abstract}
```

默认自动添加居中单列一行的 Abstract，汉化方式为

```
\renewcommand{\abstractname}{摘要}
```

文章默认是单栏，但是我们看到的很多文献是双栏的，在加载类的时候使用双栏选项。

```
\documentclass[12pt,twocolumn]{article}
```

这样文章的题目、作者和日期，也即 maketitle 的内容，是单栏，其余都是双栏。如果希望摘要仍然是双栏，可以有两种做法，简单的就是加载 abstract 包。

```
\usepackage{abstract} %摘要包
```

使用时很简单

```
\twocolumn[
\maketitle
\begin{onecolabstract}
Abstract text ...
\end{onecolabstract}
]
\saythanks %双栏格式显示标题脚注
```

这么做 thanks 脚注有时显示不对，可以使用下面的语句

```
\twocolumn[
\begin{@twocolumnfalse}
\maketitle
\begin{abstract}
Abstract text ...
\end{abstract}
```

```
\end{@twocolumnfalse}
]
\saythanks
```

这样显示无误，脚注也是双栏的。

默认双栏按内容填充，并不好看，添加 balance 包，可以使得内容不满时，双栏底部是对齐的。

```
\usepackage{balance} %双栏内容不满的页面底部对齐
```

索引默认是双栏的，但是单栏的书添加 balance 后并不会自动 balance，需要使用下面的命令。

```
\balance
\printindex
```

2.9 参考文献

参考文献目前有两套体系，一套是 natlib 包，用 bibtex 命令编译，参考文献的格式使用 bst 文件。这是目前杂志社常用的方式，texlive 里内置了许多杂志社的 bst 格式文件。另一套是 biblatex 包，用 biber 命令编译，这是比较新的参考文献格式方法。参考文献本身的条目，都可以采用 bib 格式。biblatex 和 biber 这种方法还可以采用其他格式。我个人习惯于 bib 格式，因为它相当简洁，大部分时候足够用。下面主要以 bib 格式来讲述参考文献的引用。

2.9.1 bib 格式的文献条目

bib 文件格式是最常见的参考文献条目格式之一，很多杂志社都提供文献的 bib 格式方便大家引用。下面是文章类的例子。

```
@article{huminglie-zgjg-2021, %文献的自定义引用名
title={光纤激光器泵浦的飞秒光学参量振荡器研究进展}, %文章题目
author={胡明列 and 王珏 and 范锦涛}, %作者用 and 分开
journal={中国激光}, %杂志名
volume={48}, %卷号
issue={19}, %期号
pages={1901001}, %页码，有页码范围的使用 55–59
year={2021}, %年，最后的逗号可以省略
}
```

@article 是文章，参考文献的类别，后面的都是域。bib 常见类参见下表。

表 2-4 bib 文件的常见类

类	@ 类名
公开出版的图书	@book
硕士论文	@mastersthesis
博士论文	@phdthesis
会议论文集	@proceedings
会议论文集中的一篇	@inproceedings
会议论文	@conference（等价于 @inproceedings）
无出版商或作者的图书	@booklet
书籍的一部分章节	@inbook
书籍中带独立标题的章节	@incollection
技术文档	@manual
其他	@misc
教育，商业机构的技术报告	@techreport
未出版的论文，图书	@unpublished

每一个参考文献类别的域分为必要域和可选域，总结如下。

- @article

必要域 author, title, journal, year

可选域 volume, number, pages, month, note

- @book

必要域 author/editor, title, publisher, year

可选域 volume/number, series, address, edition, month, note

- @mastersthesis

必要域 author, title, school, year

可选域 type, address, month, note

- @phdthesis

必要域 author, title, year, school

可选域 address, month, keywords, note

- @proceedings

必要域 title, year

可选域 editor, volume/number, series, address, month, organization, publisher, note

■ @conference

必要域 author, title, booktitle, year

可选域 editor, volume/number, series, pages, address, month, organization, publisher, note

■ @inproceedings

必要域 author, title, booktitle, year

可选域 editor, volume/number, series, pages, address, month, organization, publisher, note

■ @booklet

必要域 title

可选域 author, howpublished, address, month, year, note

■ @inbook

必要域 author/editor, title, chapter and/or pages, publisher, year

可选域 volume/number, series, type, address, edition, month, note

■ @incollection

必要域 author, title, booktitle, publisher, year

可选域 editor, volume/number, series, type, chapter, pages, address, edition, month, note

■ @manual

必要域 title

可选域 author, organization, address, edition, month, year, note

■ @misc

必要域 none

可选域 author, title, howpublished, month, year, note

■ @techreport

必要域 author, title, institution, year

可选域 type, number, address, month, note

- @unpublished

必要域 author, title, note

可选域 month, year

zharticle.bst

天津大学的本科生和研究生论文的 bst 文件，可以采用 texlive 版本自带的 zharticle.bst 文件，这是 Xu Yuan 从 seuthesis.bst 修订的，除了博士论文和硕士论文的标识要求不同以外，其余皆可满足天津大学的要求。可以自行删除 [D]:[Master's Thesis] 和 [D]:[PhD Thesis] 后面的:[Master's Thesis] 和:[PhD Thesis]，天津大学只要求标识为 [D]。zharticle.bst 文件并不在 bst 默认搜索目录下，而是在 texlive/2021/texmf-dist/doc/latex/seuthesis/zharticle/下，可以复制该文件到自己的 tex 文件目录下使用。如果希望添加到系统里，可以复制该文件到系统默认搜索目录下，并且执行命令 texhash，重建索引即可。但是仅仅使用一两次的 bst 不推荐这样的方法。

```
texhash
```

缺乏必要域会报错，但是如果缺乏可选域也会获得错误结果，这其实取决于 bst 文件的规定。按 zharticle.bst 的规定，书、硕士和博士论文，都必须有地址，否则会输出不正确的结果。如果天津大学论文的参考文献格式要求没有地址项，则还需要自行修改 zharticle.bst 文件。这个规定是考虑到很多国际出版社是存在不同地区公司的，而大学会有不同城市的校区，因此显示地址是合理的要求。

doi 域

上面列出的 bib 域里是没有 doi 的，但是在.bib 文件添加 doi 域并不会报错，只是不会显示，比如使用 unsrt 格式时，bib 里的 doi 域不会显示出来。但是目前很多杂志社的 bst 支持 doi，比如使用 elsarticle-num 时，会显示出来 doi 号且自动为超链接格式，可以自动打开网页，即从 doi 网站进入到该文献所在的网页。因此推荐在 bib 文件中都添加 doi 项。下面介绍的 biblatex 包是支持 doi 项的。

2.9.2 natlib 包 +bibtex 编译

使用 natlib 包的语句为

```
\usepackage[super,square,comma,sort&compress]{natbib}
```

super 表示文献为上标格式，square 表示引用编号自动添加方括号，compress 表示自动压缩，即 [5-8]，而不是 [5,6,7,8]。

在文章里使用 cite 语句，还有最常见的使用 citenum 语句引用参考文献的序号。

```
飞秒光纤激光器还可以作为飞秒光学参量振荡器的泵浦源\cite{huminglie-zgjg-2021}和双光子荧光显微镜的光源。参见参考文献\citenum{huminglie-zgjg-2021}
```

文章后面的文献列表将按照加载的格式文件排列。

```
\bibliographystyle{zharticle} %使用 zharticle.bst 作为格式文件
```

向杂志社投稿文章时一般会使用杂志社的类，而不是 article，这时有两种情况。杂志社的类包含参考文献格式，就不用\bibliographystyle语句了。如果杂志社的类不包含参考文献格式，则需要该语句。有的大杂志社如 ieee 或 acs，有不止一种参考文献格式，加载与投稿杂志匹配的格式。大部分的 bst 文件在 texmf-dist/bibtex/bst 目录下。下面是几个例子。

```
\bibliographystyle{IEEEtranSN} %ieee
\bibliographystyle{IEEEtran} %ieee
\bibliographystyle{elsarticle-num} %elsevier
```

然后指定自己写的 bib 文件，

```
\bibliography{a} %bib 文件名是 a.bib
\bibliography{bib文件名} %可以是其他文件名
```

编译方式：分四步编译，pdflatex 命令编译 tex 文件 3 次，bibtex 命令编译 bib 文件 1 次。

```
pdflatex a.tex
bibtex a %编译的是主 tex 的文件名
pdflatex a.tex
pdflatex a.tex
```

可以写多个 bib 文件，任意起名，比如 a1.bib 和 a2.bib，加载时用逗号分割

```
\bibliography{a1,a2}
```

编译时还是上面的步骤，bibtex 编译的是 a，不带扩展名

可以通过下面的语句自己编写 bst 文件，执行完该语句后，通过选择可以自动建立 bst 文件，但是总有不能满足要求的地方需要修改 bst 文件。

```
latex makebst
```

bst 格式文件虽然比较复杂，修改也比较麻烦，但是 natbib+bibtex 方式仍然是最简洁的、冗余文件最少的编译方法。

2.9.3 natbib 包情况下分章参考文献

chapterbib 包

分章参考文献需要加载 chapterbib 包，chapterbib 包与 biblatex 包不匹配，必须使用 natbib 包。

```
\usepackage[super,square,comma,sort&compress]{natbib}
\usepackage[sectionbib]{chapterbib} %参考文献为不编号小节
```

文件结构必须是每章单独一个 tex 文件，比如原始文件为 a.tex，第一章文件为 c1.tex，第二章为 c2.tex。在 a.tex 里以 include 形式加载 c1.tex 和 c2.tex。

```
\include{c1.tex} %不写扩展名也可以，推荐写全名
\include{c2.tex}
```

\input{文件名}和\include{文件名}最大的不同是，后者编译时对所有加载的文件都单独建立辅助文件，这样就可以对 c1.tex 和 c2.tex 进行 bibtex 编译。c1.tex 和 c2.tex 除了章节内容和引用外，最后都包含各自的同名 bib 文件。

```
章节内容
\bibliographystyle{elsarticle-num}
\bibliography{c1} %与 c1.tex 对应的 bib 文件
```

```
章节内容
\bibliographystyle{elsarticle-num}
\bibliography{c2} %与 c2.tex 对应的 bib 文件
```

编译时按下面的顺序编译

```
pdflatex a.tex %编译主文件
bibtex c1 %编译第一章参考文献
bibtex c2 %编译第二章参考文献
```

```
pdflatex a.tex %第二次编译主文件
pdflatex a.tex %第三次编译主文件
```

最后生成的 pdf 文件，参考文献直接放在章的最后，不单独分页，参考文献本身没有小节编号，目录不显示分章的参考文献。这也是分章参考文献最常见的格式。默认 bibname 是 Bibliography，如果希望修改名字，可以加到 a.tex，也可以加到任何一章中，即 c1.tex 和 c2.tex 都可以。bibname 是对全局有效的，只需要添加一次就可以修改名称，比如修改成 References 或者汉化。

```
\renewcommand{\bibname}{References}
\renewcommand{\bibname}{参考文献}
```

如果去掉包的 sectionbib 选项，

```
\usepackage{chapterbib}
```

参考文献就会以不编号的章的格式显示，默认 book 类的设置使得分章的参考文献从奇数页开始，按章的格式显示。这样会添加许多空白页，所以推荐加载 secitionbib 选项。

还可以选择双重参考文献，使用下面的语句

```
\usepackage[duplicate,sectionbib]{chapterbib}
```

然后在主文件 a.tex 末尾添加

```
\backmatter
\bibliographystyle{elsarticle-num}
\bibliography{s1,s2}
```

文章最后会重复一遍分章参考文献，但是这种情况并不常见。

bibunits 包

分章参考文献也可以加载 bibunits 包。

```
\usepackage[super,square,comma,sort&compress]{natbib}
\usepackage[sectionbib]{bibunits} %
```

bibunits 可以使用一个 tex 文件 a.tex 和一个 bib 文件 ref.bib，tex 和 bib 文件不需要强制同名，不用\include语句方式。不需要加载 natbib 包也可以获得正确结果，但是不加载 natbib 包参考文献没有超链接，所以推荐加

载。

下面的例子是按书格式的结构，

```
\chapter{laser}
\begin{bibunit}[plain]
aaa\cite{fluid}
\putbib[ref] %ref 是 bib 文件，一个 bib 文件就行
\end{bibunit}

\chapter{math}
\begin{bibunit}[plain] %可以选择不同的参考文献样式
aaa\cite{fs-1}
\putbib[ref]
\end{bibunit}
```

编译的时候使用下面的语句

```
pdflatex a.tex %编译主 tex 文件一次
bibtex bu1 %bu1 文件是自动生成的
bibtex bu2 %bu2 文件是自动生成的，有几个环境就有几个文件
pdflatex a.tex %第二次编译主 tex 文件
pdflatex a.tex %第三次编译主 tex 文件
```

写书的时候，还是更推荐 chapterbib 包使用\include语句方式。

2.9.4 biblatex 包 +biber 编译

使用 biblatex 包，且使用 biber编译

```
\usepackage[backend=biber]{biblatex} %指定 biber 编译方式
\addbibresource{a.bib} %指定 bib 文件
```

biblatex 的格式文件后缀为.bbx，使用方法为添加选项 style=bbx 文件名

```
\usepackage[backend=biber,style=numeric]{biblatex}
\usepackage[backend=biber,style=authoryear]{biblatex}
\usepackage[backend=biber,style=authortitle]{biblatex}
```

文章中 cite 的引用方式相同。biblatex 可以在 numeric 方式下用\citeauthor{label}引用作者。但是 biblatex 没有定义\citenum命令，可以自己定义，但是没有超链接。

```
\DeclareCiteCommand{\citenum}{}{\printfield{labelnumber}}{}{}
```

biblatex 方式比较容易汉化

```
\printbibliography[title=参考文献] %输出参考文献列表
```

编译方式：分四步编译，pdflatex 命令编译 tex 文件 3 次，biber 命令编译 bib 文件 1 次。

```
pdflatex a.tex
```

编译后获得 a.aux, a.log, a.out, a.run.xml 和 a.pdf 文件，还获得 a.bcf 文件，查看 a.run.xml 文件可以看到该过程所需的文件。此时生成的 a.pdf 文件不能正确显示引用。

```
biber a
```

编译后获得 a.bbl 和 a.blg 文件

```
pdflatex a.tex
pdflatex a.tex
```

编译后获得正确引用的 a.pdf 文件。因为有的时候比较复杂的文件需要编译两次才能正确显示，所以这里建议一般情况都编译两次。

可以使用 sorting 选项，并输出两种不同的排序方式的参考文献列表

```
\usepackage[sorting=nyt]{biblatex} %默认 nyt 方式排序
...
\printbibliography
\newrefcontext[sorting=ydnt] %重新设置排序方式为 ydnt
\printbibliography %再次输出参考文献
```

可以在 bib 文件里添加 doi 选项，且选择是否打印 doi

```
\usepackage[sorting=nyt,doi=true]{biblatex} %false 则不显示
```

2.9.5 biblatex 包 +bibtex 编译

使用 biblatex 包，但是使用 bibtex 编译

```
\usepackage[backend=bibtex]{biblatex}
\addbibresource{a.bib} %指定 bib 文件
```

文章中 cite 的引用方式相同 bibtex 编译。

biblatex 方式比较容易汉化

```
\printbibliography[title=参考文献] %输出参考文献列表
```

编译方式：分四步编译，pdflatex 命令编译 tex 文件 3 次，bibtex 命令编译 bib 文件 1 次。

```
pdflatex a.tex
```

编译后获得 a.aux, a.log, a.out, a.run.xml 和 a.pdf 文件，还获得 a-blx.bib 文件（没有 a.bcf 文件），查看 a.run.xml 文件可以看到该过程所需的文件。此时生成的 a.pdf 文件不能正确显示引用。

```
bibtex a %第二步用 bibtex 编译
```

编译后获得 a.bbl 和 a.blg 文件

```
pdflatex a.tex
pdflatex a.tex
```

最后 pdflatex 编译两次，获得正确引用结果。

只要是使用 biblatex 包，格式文件都是 bbx，bibtex 方式编译也是如此。

虽然最后的 pdf 文件都是一样的效果，但是推荐使用 natlib 包 +bibtex 编译或者 biblatex 包 +biber 编译方式，不建议使用 biblatex 包 +bibtex 编译方式。

2.9.6 biblatex 的常见功能介绍

biblatex 不需要额外的包就能够支持很多复杂功能。

分章目录

使用 refsection 选项，每一次\chapter语句都会有参考文献。也可以每个 part、小节或小小节等开始参考文献。

```
\usepackage[backend=biber,refsection=chapter]{biblatex}
\addbibresource{c.bib}
\begin{document}
\mainmater
\chapter{}
...
%按无编号 section 格式开始参考文献
\printbibliography[title=参考文献,heading=subbibliography]
```

```
\chapter{}
...
%按无编号章格式开始参考文献
\printbibliography[title=参考文献]
\backmatter
\end{document}
```

也可以使用 refsection 环境方式，此时 biblatex 的选项中不能再添加 refsection 选项。

```
\usepackage[backend=biber]{biblatex}
\addbibresource{c.bib}
\begin{document}
\mainmater
\begin{refsection} %可以在章开始前
\chapter{}
...
\printbibliography[title=参考文献,heading=subbibliography]
\end{refsection}

\chapter{}
\begin{refsection} %也可以在章开始后
...
\printbibliography[title=参考文献]
\end{refsection}
\backmatter
\end{document}
```

也可以在末尾集中分章节打印，或者章后打印一次，末尾集中分章节再打印一次，相当于 copy 了一次。

```
\usepackage[backend=biber]{biblatex}
\addbibresource{c.bib}
\begin{document}
\mainmater
\begin{refsection} %可以在章开始前
\chapter{}
...
\printbibliography[title=参考文献]
\end{refsection}
```

```
\chapter{}
\begin{refsection} %也可以在章开始后
...
\printbibliography[title=参考文献]
\end{refsection}
\backmatter
\printbibheading[title=参考文献] %默认 heading 是 Bibliography
\printbibliography[section=1,heading=subbibliography]
\printbibliography[title=第二章,section=2,heading=subbibliography]
\end{document}
```

refsection 环境可以使用每章独立的 bib 文件。导言部分添加多个 bib 文件

```
\addbibresource{s1.bib}
\addbibresource{s2.bib}
```

也可以使用下面的语句表示这是部分 bib 文件

```
\addsectionbib{s1.bib}
\addsectionbib{s2.bib}
```

refsection 环境添加参数

```
\begin{refsection}[s1.bib]
...
\end{refsection}
```

注意 biber 命令编译的时候**不要编译 bib 文件名 s1 和 s2，编译主文件名**，如果是 a.tex，编译过程如下

```
pdflatex a.tex
biber a
pdflatex a.tex
pdflatex a.tex
```

2.10 版面设计 titlesec 与天津大学研究生论文的格式设置

一共三个包，titlesec、titleps和 titletoc，三个包是一个手册，都可以用下面的语句调出

```
texdoc titlesec
```

用 titlesec 等包排版时，至少需要编译两次才可以正确显示。编译一次正确，再编译错误，表示有代码错误。第一次编译有错误，但是再次多次编译正确，说明代码正确。

加载包时，因为 titlesec 需要加参数，一般写成下面的分开形式

```
\documentclass{book} %默认区分奇偶页
\usepackage[clearempty]{titlesec} %偶数页无内容自动空白页
\usepackage{titleps,titletoc}
```

书的默认选项是区分奇偶页的，每一章从奇数页开始。章的起始页默认没有页眉页脚，即 empty 样式。当偶数页刚好没有内容时，比较美观的做法是该偶数页为空白页，没有页眉和页脚。clearempty 选项使得不需要语句\cleardoublepage，偶数空白内容页就可以采用 empty 页面。本教程采用的就是默认的 empty 页面。

一般国内的本科生和研究生论文格式都要求章的起始页有页眉页脚，要求偶数空白页仍然保持页眉页脚，这样从对称的角度比较美观，因此不需要这个选项。

下面主要讲 book 类默认区分奇偶页的排版，book 类不区分奇偶页的排版，很多设置就没有必要了，使用 oneside 参数可以不再区分奇偶页。

```
\documentclass[oneside]{book} %不区分奇偶页
```

需要先了解一下默认的版面。默认样式有四个

empty	无页眉，无页脚
plain	无页眉，页脚居中显示页码
headings	无页脚，页眉包含章节标题和页码
myheadings	无页脚，页眉包含页码

书包含三个部分，frontmatter、mainmatter 和 backmatter。

可以没有 frontmatter，但是不能省略 backmatter，否则格式排版会有显示不正确的地方。如果没有 backmatter，在内容最后使用 clearpage 语句也可以解决这个问题。article 类如果设置页眉页脚时也经常会有编译问题，内容最后也需要添加 clearpage 语句。article 类默认是没有页眉页脚的，所以单纯的 article 类是不需要最后 clearpage 的。article 类添加目录时，内容最后也要 clearpage 一下。

一般 backmatter 后面是索引、参考文献等，这些其实本质上都是独立的章，所以不会出现问题。backmatter 部分还可以添加额外的章，但是

不要直接写内容，或者小节及其内容。如果这样做，就会连着 backmatter 之前的章节编号，相当于 backmatter 不起作用了。此时即使前面有 backmatter，最后也要添加 clearpage 才能不编译错误。只是就算不再编译错误，这样的结构没有任何意义。

这三部分的默认格式如下：

- `\frontmatter`
  第一页无页眉，页码在底部居中位置，即 plain 样式。第二页如果有内容，页眉两边分别是页码和章名。页眉无横线。页脚为空。即 headings 样式。
- `\mainmatter`
  第一页无页眉，页码在底部居中位置，即 plain 样式。第二页如果有内容，页眉两边分别是页码、章号和章名（偶数页）。第三页开始奇数页的页眉，两边分别是小节号和小节名、页码。页眉有横线。页脚为空。即 headings 样式。
- `\backmatter`
  同 frontmatter，无章号。

frontmatter 部分，可包含前言、参数表等内容。mainmatter 包含主要内容和附录。backmatter 一般包含参考文献、索引、术语表、后序，研究生论文的参考文献、成果和致谢等内容一般放在 backmatter。

frontmatter 和 backmatter 可以有 chapter 但是不排序号、不分小节，因为 frontmatter 时章号`\thechapter`为 0。frontmatter 里的章和目录、章节、backmatter 里的章是自动加入目录的。如果添加了 hyperref 包，pdf 文件会自动生成书签，书签包含除了目录外的所有章节。如果希望书签里也包含目录，则需要额外添加语句，这又分为两种情况。

1. 一种是目录里也包含目录和目录页码，此时可以在`\tableofcontents`后面添加语句将目录添加到目录里。

   ```
   \addcontentsline{toc}{chapter}{Contents}
   ```

   这句写到`\tableofcontents`之前不影响在目录里添加目录，但是 pdf 书签点击时会显示不正确，所以要写到后面。
2. 目录里不添加目录，单独生成 pdf 标签，添加下面的语句，这两个语句在 hyperref 包里。

   ```
   \pdfbookmark[0]{Contents}{contentsname} %[0] 可以省略
   ```

或者

```
\currentpdfbookmark{Contents}{contentsname} %当前级别
```

效果不如第一种情况，点击书签后会显示目录第一页下面的位置，而不是目录标题的位置。但是对多行目录，的确可以找到目录的第一页。

经常遇到的附录，默认章号按 ABCD 排序，要写在\backmatter之前的 mainmatter 里。附录可以有小节，backmatter 里的内容没有小节。

```
\appendix %这条语句后，默认按 ABCD 排章节号
\chapter{数学}
\chapter{量子}
\backmatter %参考文献、索引、术语表等
```

下面我们按天津大学研究生论文要求讨论怎么修改排版。因为 CJKutf8 包对汉字只有伪黑体（加粗），所以采用 xeCJK 包，并采用 xelatex 编译。然后就可以使用其他多种字体，比如微软黑体。

2.10.1　设置页眉页脚

主体部分的样式

论文主体的章的第一页（首页）无页眉，页脚在底部居中，这个是默认的 plain 样式，所以要**重新定义** plain 样式才可以修改，使用\renewpagestyle语句。重新定义语句必须写在 mainmatter 之后，表示修改的是 mainmatter 部分的 plain 样式。

天津大学研究生论文要求页眉和页脚字号不同，所以还需要添加字号的设置。

```
\mainmatter %下面的修改必须写在 mainmatter 之后
\renewpagestyle{plain}[\fontsize{10.5pt}{12pt}\selectfont]{
\sethead[][天津大学硕士学位论文][] %方括号偶数页，分别为左中右
{}{第~\thechapter~章~~\chaptertitle}{} %花括号奇数页
\headrule
\setfoot[][\fontsize{9pt}{11pt}\selectfont\thepage][]{}{\fontsize{9pt
}{11pt}\selectfont\thepage}{}
}
```

第二页开始，可以采用定义新样式的方法，\newpagestyle，样式可以自己起名，下面例子起的名是 main。

```
\newpagestyle{main}[\fontsize{10.5pt}{12pt}\selectfont]{
\sethead[][天津大学硕士学位论文][] %方括号偶数页
{}{第~\thechapter~章~~\chaptertitle}{} %花括号奇数页
\headrule
\setfoot[][\fontsize{9pt}{11pt}\selectfont\thepage][]{}{\fontsize{9pt
}{11pt}\selectfont\thepage}{}
}
```

新样式的定义可以放在 frontmatter 之前，也可以放在 mainmatter 之后。上面只是定义，使用定义好的样式的语句为\pagestyle{样式名}，写到\mainmatter之后

```
\mainmatter
\pagestyle{main} %使用 main style，先定义后使用即可
```

前言部分的样式

接着修改前言部分的样式，最前面的创新性声明部分要求无页眉和页脚。这种样式已经内置，称为 empty。所以只需要在\frontmatter后使用即可。这里需要使用\thispagestyle，该语句仅仅对当前页有效。必须每一页都声明采用 empty，因为\frontmatter之后默认章的首页是 plain，后面是有页眉的。frontmatter 里的单页（不在章里的页）是默认为页码在上的，即章名空缺时的默认格式。所以题目页和独创性声明页需要两次\thispagestyle语句。

```
\frontmatter
\thispagestyle{empty}
自行设计题目页
\thispagestyle{empty}
独创性说明页
```

摘要页开始，以及目录，要求无页眉，只有页脚居中，大写罗马字母作为页码。没有横线。这种情况，摘要也是章，只是\frontmatter部分章不编号。按照天津大学研究生论文的要求，此时默认章的首页是 plain 样式，这是对的，因此只需要修改\pagestyle也为 plain 样式，除章首页外其他页有效。

```
\pagestyle{plain} %内置 plain 即可满足要求。
```

自己写书时希望前言部分有特殊页眉页脚，在这里采用自定义的格式即可。

从摘要开始，页码编号，且为罗马大写字母。但是默认从\frontmatter开始已经页码编号。使用\maketitle产生的题目页不计入编号，自行设计的题目页仍然会编号。因此，摘要开始前要重置页码编号为 0。

```
\pagenumbering{Roman} %页码为大写罗马字母样式
\setcounter{page}{0} %页码为 0，下面开始编号
\chapter{摘~~要}
```

empty 样式只是不显示页码，页码本身还是存在且逐页编号的，所以要重新设置编号。重新设置编号的语句都是\setcounter{编号内容}{起始编号减一}。

单双页重置

书的默认选项是 twoside，章起始页奇数页码，偶数页无内容也存在。论文主体部分要求 twoside，但是题目和独创性说明需要 oneside。从摘要开始 twoside，最后参考文献、发表文章和致谢等又可能需要 oneside。需要增加一个新包 ifthen，在文章内重置单双页。

```
\usepackage{ifthen}
```

在需要的地方设置 twoside 为真或假，假就是 oneside

```
\setboolean{@twoside}{false} %oneside
\setboolean{@twoside}{true}  %twoside
```

目前天津大学研究生论文不要求后面重置为 oneside。

一般情况

天津大学的要求比较简省，这本书在前沿和目录这里仍然定义了页眉，因为书的前言和目录都比较长，因此有页眉比较好。按中文书习惯，把页码放到了页脚。empty 样式用于题目和版权页。

```
\newpagestyle{mypre}{
\sethead[\chaptertitle][][]{}{}{\chaptertitle}
\setfoot[][\thepage][]{}{\thepage}{}
}
```

上面设置自定义格式的语句可以放在 frontmatter 之前，也可以放在开始前言内容之前。

```
\pagestyle{mypre} %章首页仍然是 plain 样式
```

附录

附录在 mainmatter 里，自动 ABCD 编号，如果对附录有格外要求，可以同上所述重新设置一遍页眉页脚。

backmatter 的样式

从\backmatter开始，章没有编号，页眉也需要没有章编号的，因为在主体部分的页眉汉化了 chaptertitlename，所以需要再次定义页眉样式，包括重新定义首页页眉 plain（需要写到\backmatter之后）。

```
\renewpagestyle{plain}[\fontsize{10.5pt}{12pt}\selectfont]{
\sethead[][天津大学硕士学位论文][]
{}{\chaptertitlename}{} %花括号奇数页
\headrule
\setfoot[][\fontsize{9pt}{11pt}\selectfont\thepage][]{}{\fontsize{9pt
}{11pt}\selectfont\thepage}{}
}
```

新定义 back 样式：

```
\newpagestyle{back}[\fontsize{10.5pt}{12pt}\selectfont]{
\sethead[][天津大学硕士学位论文][] %方括号偶数页
{}{\chaptertitlename}{} %花括号奇数页
\headrule
\setfoot[][\fontsize{9pt}{11pt}\selectfont\thepage][]{}{\fontsize{9pt
}{11pt}\selectfont\thepage}{}
}
```

```
\backmatter
\pagestyle{back} %首页样式为重新定义的 plain
```

如果后面要求 oneside，所以还需要添加

```
\setboolean{@twoside}{false}
```

设置 oneside 为真（twoside 为假），后续章节页码连续编号，发表文章和致谢如果都是一页，页码是连续的。目前研究生论文不要求重置为 oneside，如果成果页是单页，致谢页码是奇数。

2.10.2　章节名称的样式

章节名称的样式主要是两个命令，\titleformat和\titlespacing，前者定义样式，后者定义间距。下面以章为例说明语句用法。

```
\titleformat{章节段落等}[内置样式]{前导设置语句，一般定义字体大小、字体样式、是否居中等}{章名前的内容，比如第几章}{章名前的内容与章名之间的间距}{章名前的前导设置语句，可以改变字体大小}[后续语句，可省略{多行语句必须加花括号}]
```

间距为

```
\titlespacing*{章节段落等} {左间距}{上间距}{下间距}{右间距，可省略}
```

手册 24 页（倒数第二页）给出了默认结果，这个结果间距比较大，章节名格式不符合中文书籍的一般情况。

```
\titleformat{\chapter}[hang]{\normalfont\huge\bfseries}{\
chaptertitlename\ \thechapter}{20pt}{\Huge}
\titleformat{\section}{\normalfont\Large\bfseries}{\thesection}{1em
}{}
\titleformat{\subsection}{\normalfont\large\bfseries}{\thesubsection
}{1em}{}
\titleformat{\subsubsection}{\normalfont\normalsize\bfseries}{\
thesubsubsection}{1em}{}

\titlespacing*{\chapter} {0pt}{50pt}{40pt}
\titlespacing*{\section} {0pt}{3.5ex plus 1ex minus .2ex}{2.3ex plus
 .2ex}
\titlespacing*{\subsection} {0pt}{3.25ex plus 1ex minus .2ex}{1.5ex
plus .2ex}
\titlespacing*{\subsubsection}{0pt}{3.25ex plus 1ex minus .2ex}{1.5
ex plus .2ex}
```

样式设置

因为涉及大量的自定义字体，因此需要加载包 anyfontsize

```
\usepackage{anyfontsize}
```

设置字体，基本中文字体为新宋体，基本英文字体为 Times New Roman，设置黑体命令，这里是微软的黑体 simhei。

```
\usepackage{xeCJK}
\setCJKmainfont[AutoFakeBold]{SimSun} %可使用 bf 加粗
\setmainfont{Times New Roman}
\setCJKfamilyfont{heiti}{SimHei}
\newcommand{\hei}{\CJKfamily{heiti}}
```

反复使用的字号，可以定义一下，新要求的页码与页眉字号是不相同的。

```
\newcommand{\fontfive}{\fontsize{10.5pt}{12pt}\selectfont}
\newcommand{\fontsmallfive}{\fontsize{9pt}{11pt}\selectfont}
```

天津大学研究生论文主题部分，按要求设置为：

```
% 15pt 相当于小三
\titleformat{\chapter}[hang]{\fontsize{15pt}{18pt}\selectfont\bf\hei\
filcenter}{第~\thechapter~章}{1em}{}
% 14pt，即 large，相当于四号
\titleformat{\section}{\large\bf\hei}{\thesection}{1em}{}
\titleformat{\subsection}{\large\bf\hei}{\thesubsection}{1em}{}
% 12pt，即 normalsize，相当于小四号
\titleformat{\subsubsection}{\bf\hei}{\thesubsubsection}{1em}{}
```

这里将章号和章名之间的间距设置为 1em 了，论文模板显示的效果小于 1em，感觉不好看。

下面的语句设置间距

```
\titlespacing*{\chapter}{0pt}{10pt}{30pt}
\titlespacing*{\section}{0pt}{18pt}{18pt}
\titlespacing*{\subsection}{0pt}{12pt}{12pt}
\titlespacing*{\subsubsection}{0pt}{9pt}{9pt}
```

摘要的样式

摘要要求二号字，18pt，而且要求宋体粗体，所以要加一个参数，使得字体没有定义粗体也可以粗体显示，即\bf对宋体有效。注意不要加在包选项参数里，要加在主字体那里，对某个字体单独设置。Times New Roman 是有 bold 的，不需要加参数就可以粗体。

```
\setCJKmainfont[AutoFakeBold]{SimSun}
```

摘要和目录的 format，不编号

```
\titleformat{\chapter}[hang]{\fontsize{18pt}{20pt}\selectfont\bf\
filcenter\vspace{-1cm}}{}{1em}{}{}
```

2.10.3　公式图表

图表的 caption 都需要汉化一下，修改一下默认样式，这里需要 caption 包。

```
\usepackage{caption}
```

设置语句为：

```
\captionsetup[figure]{name=图,labelsep=space,font=small, labelfont=
small}
\renewcommand{\thefigure}{\arabic{chapter}-\arabic{figure}}

\captionsetup[table]{name=表, labelsep=space,font=small, labelfont=
small}
\renewcommand{\thetable}{\arabic{chapter}-\arabic{table}}

\renewcommand{\theequation}{\arabic{chapter}-\arabic{equation}}
```

如果觉得空一个英文空格太小，可以在导言（不能在 document 环境之后）添加语句

```
\DeclareCaptionLabelSeparator{doublespace}{\ \ }
```

然后将参数修改为 labelsep=doublespace。

同理，设置字体大小为五号字

```
\DeclareCaptionFont{small-5}{\fontsize{10.5pt}{12pt}\selectfont}
\captionsetup[figure]{name=图,labelsep=doublespace,font=fontfive,
labelfont=small-5}
\captionsetup[table]{name=表, labelsep=doublespace,font=fontfive,
labelfont=small-5}
```

上面的例子中的字体设置都可以用定义新字号完成，但是涉及代码完整性，书中大部分例子都是展开的形式。

2.10.4 目录样式

目录样式的语句是\titlecontents

```
\titlecontents{章节}[左间距]{前置代码}{有编号样式}{无编号样式}{页码样式}[后置代码]
```

下面的例子按天津大学研究生论文的要求，在目录中显示第 1 章，按要求节和小节各错后 2 个汉字间距。

```
\titlecontents{chapter}[0em]{}{第~\thecontentslabel~章\hspace{1em}}{}{\titlerule*[0.5pc]{.}\contentspage}[]

\titlecontents{section}[4em]{}{\contentslabel{2em}}{}{\titlerule*[0.5pc]{.}\contentspage}

\titlecontents{subsection}[7em]{}{\contentslabel{3em}}{}{\titlerule*[0.5pc]{.}\contentspage}
```

这里需要说明一下。LaTeX 的设计理念跟 word 不大一样。这个例子中章格式的第一个参数采用 0em 表示第几章前无缩进，后面的留白 1em 表示与后面章名之间的间隔，xelatex 对中英文混编的效果导致加起来，章名前空了 4em。

小节设置 4em，是小节名的缩进，不是小节号的缩进。小节号的缩进是语句\contentslabel{间距}里设定的。也即前一个参数一定要大于后一个参数。之所以章名缩进可以设置为 0em，是因为没有使用\contentslabel{间距}语句。这样做是因为要汉化，而且可以彻底汉化，即目录也可以显示第一章。但是如果采用\contentslabel{间距}方式，则章的设置语句，0em 的设置就不对了。

这种设计理念的好处是可以非常整齐划一的显示目录。当小节号从 1.1 变为 1.23 时，压缩的是小节号与小节名之间的间距，保持了整体的形式美，不会形成犬压交错的目录。

在目录里汉化章号时不能使用thechapter作为章的编号，因此目录这里thechapter是 0，还没开始编号。这里的章编号是\thecontentslabel。天津大学论文没要求章号汉化，数字汉化推荐新包 zhnumber代替 CJKnumb 包。使用 pdflatex 编译时需要加参数设置编码为 UTF8。

```
\usepackage[encoding=UTF8]{zhnumber}
```

目录格式里的汉化的语句是

```
\zhnumber{\thecontentslabel}
```

章的地方汉化语句为

```
\zhnumber{\thechapter}
```

不带章号的目录

参考文献和致谢放在 backmatter 后，默认不编号，但是可以在目录里显示。

如果在正文中有章或小节不准备编号，带 * 即可。

```
\section*{不编号的小节}
```

但是加星后就不会在目录中自动显示，需要手动添加。比如一些教材每章后的总结和习题等内容是不编号的，手动添加到目录即可显示。这里的尺寸是按前面的研究生论文的格式要求做出来的，小节名起始位置与原来的小节号起始位置左对齐。

```
\section*{习题}
\titlecontents{section}[2em]{}{}{}{\titlerule*[0.5pc]{.}\
contentspage}
\addcontentsline{toc}{section}{习题}
```

2.10.5　题目页设计

题目页设计推荐使用前面介绍过的各种 box 完成，可以获得比较精确的位置设定。题目、姓名、学院、指导教师等都推荐采用无框架线的表格形式，冒号单独为一列比较美观。

独创声明页的签名部分推荐使用 makebox 语句，前面有例子。

2.10.6　其他常用书籍排版

分章目录

有些多人写的合集每章都会有目录，可以使用下面几个语句：

```
\startcontents %开始目录
\printcontents{前缀}{开始层}[目录深度]{代码}
\stopcontents %停止目录，某些小节可以不显示在目录
```

```
\resumecontents %重新添加，不常用
```

一般开始层为 1，从小节开始，目录深度 1 层显示 section 目录，2 层显示 subsection。显示 subsection 时需要定义 subsection 样式。

部分内容目录时，不需要\tableofcontents。下面是一个例子，目录做在章的样式内，目录在章标题和章内容之间。注意最后方括号里的多行语句需要花括号括起来。在章的样式外定义目录格式即可，定义全局有效，写到前面后面都行。

```
\titleformat{\chapter}[hang]{\huge\bf}{\chaptertitlename~~\
thechapter}{20pt}{}[{\normalsize\rm\vspace{2pt}\titlerule\vspace{8
pt}
\startcontents
\printcontents{}{1}[2]{}\vspace{6pt}\titlerule}]
```

可以同时使用\tableofcontents和章内目录，目录样式全书可以是统一的，比如默认设置就比较适合。下面的例子是为了测试不一样的章内目录效果设置的。

```
\titlecontents{chapter}[2.5em]{\vspace{1em}}{\contentslabel{2.5em
}}{}{\titlerule*[0.5pc]{.}\contentspage[(\thecontentspage)]}[]
\titlecontents{section}[2.5em]{}{\contentslabel{2.5em}}{}{\titlerule
*[0.5pc]{.}\contentspage[(\thecontentspage)]}[]
\titlecontents{subsection}[5.7em]{}{\contentslabel{3em}}{}{\
titlerule*[0.5pc]{.}\contentspage[(\thecontentspage)]}[]
```

上面这个例子中，使用\contentspage[(\thecontentspage)]给页码加圆括号，不能直接加圆括号。\titlerule*[0.5pc]{.}定义小节标题到页码之间的圆点样式。\stopcontents语句加在某个小节前，这一章的后面小节不再目录中显示，不会影响到其他章的小节目录。

如果希望前面的总目录和后面的章内目录样式不一样，那么需要设置\printcontents的参数为 l，

```
\printcontents{l}{1}[2]{}
```

然后定义 lsection 和 lsubsection 的样式

```
\titlecontents*{lsection}...
\titlecontents*{lsubsection}...
```

章内目录仍然是默认标题超链接的，这点很方便。

也可以放在章样式外，但是这样会显示在章标题之前，twoside 时会额外增加两页，并且需要特别定义页眉页脚样式。一般带章目录的书籍并不采用这样的样式。

紧凑目录

有些书的小节目录不断行，只需要加星，\titlecontents*，此时目录样式一般会设置成下面的样式，

```
\titlecontents{chapter}[2.5em]{\vspace{1em}}{\contentslabel{2.5em
}}{}{\titlerule*[0.5pc]{.}\contentspage[(\thecontentspage)]}[]
\titlecontents*{section}[2.5em]{}{\thecontentslabel\hspace{0.5em
}}{}{\ \contentspage[(\thecontentspage),]}[\hspace{3em}]
```

上面的例子设置小节号前无缩进，小节号与小节标题间距 0.5em，小节标题与小节页码空一格，小节页码加圆括号并跟随逗号，最后跟随 3em 的空白，这个空白包含了小节号占据的空间，所以逗号与小节号之间的空白比 3em 小。

如果希望再加一层，包含小小节，不能完全照搬上面的例子，需要略做修改，小小节号所占空间用小小节号前横向空白定义，前面缩进仍然是 0em，而后面的空白不需要再定义横向空白 3em，否则会重复定义。

```
\titlecontents*{subsection}[0em]{}{\hspace{3em}\thecontentslabel\
hspace{0.5em}}{}{\ \contentspage[(\thecontentspage),]}
```

例子是用逗号分隔，也可以用章节号 § 前引，或者点号 • 前引等等，这样最后不会有一个逗号显得比较突兀。上面例子的逗号可能改成点比较好看。

2.11 页眉页脚包 fancyhdr

fancyhdr是一个比较简单的页眉页脚包，据说国外杂志社模板用的就是这个包。相对 titlesec 来说，它定义的奇数页和偶数页的页眉页脚语句，并且可以换行。titlesec 的页眉页脚不能换行，多行需要 parbox 这样的语句。

```
\usepackage{fancyhdr}
```

加载包后，下面是一个简写命令的例子

```
\lhead{左页眉}
\chead{中页眉\\可以换行}
```

```
\rhead{右页眉}
\lfoot{左页脚}
\cfoot{中页脚}
\rfoot{右页脚}
\pagestyle{fancy}
```

奇偶页不同仍然可以用方括号和花括号表示

```
\lhead[左页眉]{左页眉}
```

fancyhdr 包的内容很丰富，fancyhdr 的效果 titlesec 也都可以做出来，只是代码不够简洁罢了。

2.12 脚注 footnote

脚注可以说是相当复杂的一个东西，因为有太多特例需要额外语句才能完成。脚注默认是章内自动连续编号的，加载 hyperref 包后脚注自动超链接。最简单的语句为

```
内容\footnote{脚注内容}。
```

在小节标题使用脚注需要添加语句\protect，

```
\section{小节内加脚注\protect\footnote{小节内脚注}}
```

希望脚注按页编号，而不是章内连续编号，可以使用下面两个包，一个是 footnpag，另一个是 footmisc，后者功能更多一些。

```
\usepackage{footnpag}
\usepackage[perpage]{footmisc}
```

footnpag 包的功能比较单一，footmisc 包功能略多一些。

这样的脚注无法重复使用。那么希望重复使用脚注，就必须手动添加编号。

```
第一个脚注\footnote{编号重启为1}

第二个脚注采用指定编号形式\footnotemark[2]
\footnotetext[2]{脚注内容}

%设置编号为 2，下一个自动编号脚注编号为 3
\setcounter{footnote}{2}
第四个脚注\footnote{编号重启为3}
```

```
重复使用第二个脚注\footnotemark[2]
```

在表格里加脚注的例子，放到表格那一小节了，参见第54页。

2.13 信 letter 类

现在虽然都 email 了，但是一般 cover letter 还是以信的格式来书写，所以简单介绍一下书信的格式。

```
\documentclass{letter} %书信专门的类

%结束问候语等移动左侧，默认居中
\longindentation=0pt

%写在 document 之前的部分是自己的，经常用不用改
\signature{自己的名字}
\address{自己的地址\\Tianjin University} %日期是自动的

\begin{document}
\begin{letter}{对方地址}
\opening{Dear Mr. Smith,} %开头敬称
内容主体
\closing{Best regards,} %结束时的问候语
\end{letter}
\end{document}
```

2.14 背景水印

2.14.1 background

使用 background 包可以很方便添加水印

```
\usepackage{background}
```

默认的选项是缩放 scale=10，位置页面中间，透明度 opacity=0.5，angle=60 度。这些可以在包选项里修改，也可以在语句内修改。如果不同页添加不同水印，语句内修改比较合适。

```
\backgroundsetup{contents={\includegraphics[width=1cm]{tju.pdf}},
angle=0}
```

设置语句写到\begin{document}之前，是设置所有页的水印，否则是设置当前页及以后页的水印。

background 包虽然可以非常方便的设置水印，将水印移到任意位置，对奇偶页设置不同的水印等等，但是如果做官方信纸这样的需要多重图像的，没有测试成功，此时可以使用 eso-pic 包。

2.14.2 eso-pic

eso-pic 包设置水印的语句相对麻烦，但是它可以设置多个图，下面的例子是做天津大学信笺用的。

```
\documentclass{article}

\usepackage[left=2.5cm,right=2.5cm,top=4.7cm,bottom=2.5cm]{geometry
}
\usepackage{fancyhdr}

\usepackage{graphicx}
\usepackage{eso-pic}

%put 从左下角算坐标，默认 cm，可以 put 多张图
\newcommand\BackgroundPic{
\put(2.3, 24.5){
\parbox[t]{\textwidth}{
\includegraphics[keepaspectratio,width=7cm]{tju-5.pdf}%
}}
\put(14.5, -0){
\parbox[t]{\textwidth}{
\includegraphics[keepaspectratio,width=7cm]{tju-7.pdf}
}
}
}

%不加 * 号，每页都添加 logo，* 号对当前页加 logo
\AddToShipoutPicture{\setlength{\unitlength}{1cm}\BackgroundPic}


\begin{document}
%写到里面方便改写为中文的页眉页脚
\lfoot{92 Weijin Road, Nankai District\\Tianjin 300072, China}
```

```
\cfoot{Tel: +86 22 27404204}
\pagestyle{fancy}

内容
\clearpage
内容
\end{document}
```

2.15 文章内分栏

2.15.1 多栏 multicol 包

使用 multicol 包。

```
\usepackage{multicol}
```

分栏环境是 multicols，注意末尾有 s。

```
\setlength\columnsep{15mm}^^I%必须放在 multicols 环境外
\begin{multicols}{3}
multicol 宏包能够在一页之中切换单栏和多栏，也能处理跨页的分栏，
且各栏的高度分布平衡。只有multicols环境中的内容才是多栏的。
\end{multicols}
后面还是单栏，所以multicol包很方便。下面看看加入表格和图片的效果。
```

multicol 宏包能够在一页之中切换单栏和多栏，也能处理跨页的分栏，且各栏的高度分布平衡。只有 multicols 环境中的内容才是多栏的。

后面还是单栏，所以 multicol 包很方便。下面看看加入表格和图片的效果。

```
\begin{multicols}{3}
\includegraphics[width=3cm]{fig/tju.pdf}

\begin{tabular}{l l l}\toprule
a &b &c\\
\midrule
1 &2 &3\\
4 &5 &6\\
\bottomrule
\end{tabular}
```

```
如果不使用figure和table环境，图表是比较容易做出效果的。
\end{multicols}
```

a	b	c
1	2	3
4	5	6

如果不使用 figure 和 table 环境，图表是比较容易做出效果的。

如果使用 figure 和 table 环境，需要添加 float 包

```
\usepackage{float}
```

图表这样的浮动体就可以在 multicols 环境内使用，可以添加 caption。

```
\begin{multicols}{3}
\begin{figure}[H]
\includegraphics[width=3cm]{fig/tju.pdf}
\caption{天大校徽}
\end{figure}
\begin{table}[H]
\caption{简单表格}
\begin{tabular}{l l l}\toprule
a &b &c\\
\midrule
1 &2 &3\\
4 &5 &6\\
\bottomrule
\end{tabular}
\end{table}
注意图表位置用[H]
\end{multicols}
```

图 2-7 天大校徽

表 2-5 简单表格　　注意图表位置用 [H]

a	b	c
1	2	3
4	5	6

多栏环境内跨多栏（单栏）放置一张宽图，可以使用figure*的环境，表也是一样加星即可。

```
\begin{figure*}
\includegraphics[]{}
\caption{}
\end{figure*}
```

2.15.2 交互两栏 paracol

paracol 包可以很方便在两栏之间互换输入，

```
\usepackage{paracol}
```

设置两栏比例

```
\columnratio{0.7}
```

设置两栏背景色，可以分别设置，下面的语句设置的是右栏的背景色

```
\backgroundcolor{c[1]}{yellow!30}
```

两栏，交互使用

```
\begin{paracol}{2}
\columncolor{blue} %左边文字颜色
一段文字放在左边
\switchcolumn
\columncolor{red} %右边文字颜色
另一段文字
\switchcolumn
回到左边栏
\switchcolumn
回到右边
```

```
\end{paracol}
```

一段文字放在左边
回到左边栏

另一段文字
回到右边

2.15.3 更复杂的多栏版面 flowfram

flowfram 包用于非常规的杂志排版，例如 science 那样的三栏文章、三栏宽度（文字宽度）或两栏宽度图片的样式。

```
\usepacage{flowfram}
```

其手册则是 ffuserguide，flowfram.pdf 里是这个包的源代码，不是用户手册。

```
texdoc ffuserguide
```

flow frame，文字自动填充的，类似多栏那样一个 flow frame 填满后自动流到下一个 flowframe。

```
\newflowframe[页码表]{frame的宽度}{frame的高度}{起点x位置}{起点y位置}[label]
```

static frame，固定的 frame，指定位置指定内容。

```
\newstaticframe[页码表]{frame的宽度}{frame的高度}{起点x位置}{起点y位置}[label]
```

使用 static frame 时，

```
\begin{staticcontents*}{static frame label}
一般是图表
\end{staticcontents*}
```

每一个\newstaticframe按标签对应一个staticcontents*环境。设置 static frame 的页相当于特殊位置指定排版了 box，所以都需要特殊指定相应的 flow frame。所以很多情况下图片会放在上角或下角，而不是中间，因为这样 flow frame 的计算量会小一点。最后就得到图文任意混排的效果，但是版面越复杂，宽高和位置的计算量越多。

dynamic frame 跟 static frame 在使用上差不太多。

```
\newdynamicframe[页码表]{frame的宽度}{frame的高度}{起点x位置}{起点y位置}[label]
```

对应的内容是 dynamiccontents* 环境里的内容。

```
\begin{dynamiccontents*}{label}
hcontentsi \end{dynamiccontents*}
```

还可以设置 id，但是 label 更方便一些。

第三章　数学公式

3.1　行间公式

使用下面的代码可以获得下面的结果。

```
行间公式环境为一对美元符号，必须配对，$f(x)=\exp(x)$，行间公式一般采用压缩格式，比如$\int_a^b f(x)dx$，公式里的空格不会起作用，比如$\sum_i^N f_i g_i$。公式插入后行间距不会立刻自动调整，直到不能放下后才会扩展行间距。如果希望采用舒展模式，可以使用$\displaystyle \int_a^b f(x)dx$。这样会使得行间距变得不可接受。如果希望舒展的公式形式，同时不撑开行间距，则利用行间公式是文本的特点，使用整体缩放方式获得希望的效果，比如\resizebox{0.7\width}{0.7\height}{$\displaystyle\int_a^b f(x)dx$}。该命令可以设定宽度和高度比例，也可以使用感叹号获得按比例缩放，比如\resizebox{0.6\width}{!}{$\displaystyle\int_a^b f(x)dx$}。这样可以保持行间距不变、公式形式优美的结果。当然遇到$1\times n$矩阵这样的行间公式，行间距总会拉长的，\LaTeX 仅仅能做到尽量优美。
```

行间公式环境为一对美元符号，必须配对，$f(x)=\exp(x)$，行间公式一般采用压缩格式，比如 $\int_a^b f(x)dx$，公式里的空格不会起作用，比如 $\sum_i^N f_i g_i$。公式插入后行间距不会立刻自动调整，直到不能放下后才会扩展行间距。如果希望采用舒展模式，可以使用 $\displaystyle \int_a^b f(x)dx$。这样会使得行间距变得不可接受。如果希望舒展的公式形式，同时不撑开行间距，则利用行间公式是文本的特点，使用整体缩放方式获得希望的效果，比如$\resizebox{0.7\width}{0.7\height}{$\int_a^b f(x)dx$}$。该命令可以设定宽度和高度比例，也可以使用感叹号获得按比例缩放，比如$\resizebox{0.6\width}{!}{$\int_a^b f(x)dx$}$。这样可以保持行间距不变、公式形式优美的结果。当然遇到 $1\times n$ 矩阵这样的行间公式，行间距总会拉长的，LaTeX 仅仅能做到尽量优美。

3.2　独立公式环境

不用加载任何包就可以使用独立公式环境 equation。equation 环境是带编号的公式环境，默认格式为公式居中，公式编号右对齐。

```
\documentclass{article}
\begin{document}
\begin{equation}
f(x)=\sin x
\end{equation}
\end{document}
```

$$f(x) = \sin x \tag{3-1}$$

如果希望公式编号左对齐，可以在 documentclass 的选项里添加 leqno，结果是公式居中，公式编号左对齐（无缩进）。

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

如果希望公式左对齐（可设置缩进量），公式编号右对齐，下面的语句设置公式缩进量为无缩进

```
\documentclass[fleqn]{article} %公式左对齐，编号右对齐
\setlength{\mathindent}{0pt} %公式无缩进
```

不编号的公式环境为

```
\[
f(x)=\cos x
\]
```

$$f(x) = \cos x$$

原来一直喜欢这种简写方式，现在变得更推荐下面的加星方式，即采用equation*环境，取消和添加编号更方便一些。

3.3 amsmath 包的基本用法

默认情况只能满足公式的基本功能，一般都会加载 amsmath 包，ams 是美国数学学会的缩写。

```
\usepackage{amsmath} %美国数学学会数学包
```

3.3.1 自定义函数

amsmath 包预先定义了一些常用数学函数，如三角函数、反三角函数、指数函数、对数函数等。如果某函数不存在，可以使用下面的语句，大部分情况它们的显示效果是相同的。

```
\[
\text{sgn}(x){\rm sgn}(x)\mathrm{sgn}(x)
\]
```

$$\text{sgn}(x)\mathrm{sgn}(x)\mathrm{sgn}(x)$$

结果虽然是等价的，但是这三种方式还是有细微差别，推荐使用最后的\mathrm{}方式。因为\text{}方式，如果修改了文本的字体，就无法保证在公式中得到的结果是直体。{\rm }方式在 tex 中是有效的，它的问题在于部分杂志社排版时可能是将 tex 公式转其他公式软件，{\rm }方式不兼容（\text{}一般也兼容）。因此\mathrm{}对于函数名来说是最准确和保险的。

也可以采用自定义函数方式，笔者不推荐这种方式，不利于截取部分内容做交流。不过自己写书或做长笔记时，如果有大量的未预先定义函数需要复用，还是可以采用这样的方式的，数学系推荐这样的方式。

```
\documentclass{article}
\usepackage{amsmath}
\DeclareMathOperator{\sgn}{sgn} %自定义 sgn 函数，必须在 document 之前
\begin{document}
\[
\sgn(x) %使用自定义函数
\]
\end{document}
```

公式中的\text{文字}会直接显示文字，包括汉语

```
\[
\text{激光光强}\propto|\mathcal{E}|^2
\]
```

$$\text{激光光强}\propto|\mathcal{E}|^2$$

3.3.2 amsmath 包的公式环境

加载 amsmath 包后，就可以使用以下公式环境。

equation* 环境，星号表示无编号。

```
\begin{equation*}
f(x)=\ln x
\end{equation*}
```

$$f(x)=\ln x$$

这个是与上面的简写形式等价的，但是如果采用这样的形式，比较容易恢复编号。

aligned 环境用于一个公式编号的多行公式，& 用于设置对齐位置，&&

用于多个对齐位置的设定，会自动添加一点空白。hspace 用来设置横向空白。

```
\begin{equation}
g(x)=
\begin{aligned}
&2&&+\sin x& &x<0\\
&\exp(-x)&&+\cos(x)& &x\geq 0
\end{aligned}
\end{equation}
```

$$g(x)=\begin{aligned}&2&&+\sin x& &x<0\\&\exp(-x)&&+\cos(x)& &x\geq 0\end{aligned} \tag{3-2}$$

aligned 环境可以设置公式编号的位置，默认是居中，用[b]改为底部。

```
\begin{equation}
\begin{aligned}[b]
f&=g(x)\\
&=\sin x
\end{aligned}
\end{equation}
```

$$\begin{aligned}f&=g(x)\\&=\sin x\end{aligned} \tag{3-3}$$

split 环境使用方法接近 aligned 环境，可以自行练习。

下面的情况使用单 & 号，每列公式都添加了一点空白分割开，每个单元公式等号处对齐。

```
\[
\begin{aligned}
x&=y & a&=b &i&=j\\
x_1&=y_1 & a_1&=b_1 &i_1&=j_1\\
x_1+1&=y_1+1 & a_1+1&=b_1+1 &i_1+1&=j_1+1
\end{aligned}
\]
```

$$\begin{aligned}x&=y & a&=b &i&=j\\x_1&=y_1 & a_1&=b_1 &i_1&=j_1\\x_1+1&=y_1+1 & a_1+1&=b_1+1 &i_1+1&=j_1+1\end{aligned}$$

关于 aligned 环境，比较这两种写法，第一种是两个 aligned 环境，可

以加 box，用一个 & 符号设置为 x 前对齐。

```
\[
f(x)=
\boxed{\begin{aligned}
&ax\\
&bx^2
\end{aligned}}\hspace{5mm}
\begin{aligned}
0<&x<1\\
-1<&x<0
\end{aligned}
\]
```

$$f(x)=\boxed{\begin{aligned}ax\\bx^2\end{aligned}}\qquad\begin{aligned}0<x<1\\-1<x<0\end{aligned}$$

第二种是一个 aligned 环境，x 前对齐需要两个 &，第一个表示这是另起一列，第二个是对齐位置

```
\[
f(x)=
\begin{aligned}
&ax\hspace{5mm} & 0<&x<1\\
&bx^2 & -1<&x<0
\end{aligned}
\]
```

$$f(x)=\begin{aligned}ax\qquad&0<x<1\\bx^2\qquad&-1<x<0\end{aligned}$$

align 环境用于多行公式多个编号情况

```
\begin{align}
f(x)&=a_0+a_1x\label{eq-a}\\
g(x)&=\exp(x)\label{eq-b}
\end{align}
```

$$f(x)=a_0+a_1x \tag{3-4}$$
$$g(x)=\exp(x) \tag{3-5}$$

这里我们顺便学习一下公式引用。align 环境里，label 要加在换行符之前，引用公式时有两种方式，用 ref 引用没有括号，`\ref{eq-a}`，得到3-4，用 eqref 引用自动添加括号，`\eqref{eq-b}`，得到(3-5)。

align* 环境用于多行公式无编号情况

```
\begin{align*}
f(x)&=a_0+a_1x\\
g(x)&=\exp(x)
\end{align*}
```

$$f(x)=a_0+a_1x$$
$$g(x)=\exp(x)$$

还有一种情况，使用 align 环境，多行公式每个都有编号且对齐，但是局部有特定行不加编号。比如最常见的情况，某行是省略符，可以在这行的强制换行符\\前添加语句\notag，表示这行不再编号。使用\nonumber也可以，但是手册是\notag。

```
\begin{align}
a&=f(x)\\
b&=g(x)\\
&\vdots\notag\\
h&=z(x)
\end{align}
```

$$a = f(x) \tag{3-6}$$
$$b = g(x) \tag{3-7}$$
$$\vdots$$
$$h = z(x) \tag{3-8}$$

subequations 环境用于子公式编号情况

```
\begin{subequations}\label{sub-a}
\begin{equation}
f(x)=ax^2
\end{equation}
\begin{equation}\label{sub-b}
g(x)=\exp(x)
\end{equation}
\end{subequations}
```

$$f(x) = ax^2 \tag{3-9a}$$
$$g(x) = \exp(x) \tag{3-9b}$$

子公式分别设置 label，引用时自动加编号，\eqref{sub-a}，获得(3-9)。

subequations 环境里使用 align 环境更好，能够对齐公式。

```
\begin{subequations}
\begin{align}
f(x)&=\sin x+\cos x\label{sub-aa}\\
g(x)&=\exp(\mathrm{i}x)-\exp(-\mathrm{i}x)\label{sub-bb}
\end{align}
\end{subequations}
```

$$f(x) = \sin x + \cos x \tag{3-10a}$$
$$g(x) = \exp(\mathrm{i}x) - \exp(-\mathrm{i}x) \tag{3-10b}$$

写书的时候，某个重要公式多次出现，可以通过\tag{\ref{公式label}}重复使用第一次的编号。即手动设置公式编号为\ref{公式label}获得的编

号。只要是前面设置完 label 的公式编号都可以在后面多次重复使用。

```
\begin{equation}
f(x)=a_0+a_1x\tag{\ref{eq-a}}
\end{equation}
```

$$f(x)=a_0+a_1x \qquad (3\text{-}4)$$

下面的例子都将以简洁的无编号公式环境给出，方便快速练习。

3.4 分式和 displaystyle

\frac{分子}{分母}构成分式，

```
\[
f(x)=\frac{1}{1+x^2}
\]
```

$$f(x)=\frac{1}{1+x^2}$$

多重分式只需要合理嵌套即可

```
\[
f(x)=\frac{1}{\frac{1}{1+x^2}}
\]
```

$$f(x)=\frac{1}{\frac{1}{1+x^2}}$$

这样的公式下面的公式太小，可以使用 displaystyle

```
\[
f(x)=\frac{1}{\displaystyle\frac{1}{1+x^2}}
\]
```

$$f(x)=\frac{1}{\displaystyle\frac{1}{1+x^2}}$$

因为这种情况非常多，因此内置了简写方式\dfrac，等价于\displaystyle \frac

```
\[
f(x)=\frac{1}{\dfrac{1}{1+x^2}}
\]
```

$$f(x)=\frac{1}{\dfrac{1}{1+x^2}}$$

反之，如果在独立的公式环境中采用行间公式的压缩形式，可以使用\tfrac

```
\[
f(x)=\tfrac{1}{2}(\sin x+\cos x)
\]
```

$$f(x)=\tfrac{1}{2}(\sin x+\cos x)$$

式中的求和、积分，如果在分子分母上，默认为行间公式形式，如果觉得舒展开好看，也需要加\displaystyle

```
\[
\frac{\displaystyle\sum_{i=0}^{i=N} x^i}{x}
\]
```

$$\frac{\displaystyle\sum_{i=0}^{i=N} x^i}{x}$$

3.5 开根号的细节

LaTeX 的公式排版很注意细节，下面的例子，第二个写法可以使得三个开根号的横线完全在同一水平线上。下面的结果需要放大后才能看出效果。

```
\[
\sqrt{x}\sqrt{y}\sqrt{z}
\]
\[
\sqrt{x}\sqrt{\smash[b]{y}}\sqrt{z}
\]
```

$$\sqrt{x}\sqrt{y}\sqrt{z}$$

$$\sqrt{x}\sqrt{y}\sqrt{z}$$

顺便练习一下开根号，前面方括号里定义次数

```
\[
\sqrt[n]{x}
\]
```

$$\sqrt[n]{x}$$

如果只定义次数，位置可能不好看

```
\[
\sqrt[n]{\frac{a}{x}}
\]
```

$$\sqrt[n]{\frac{a}{x}}$$

好看的样子是调整左右和上下位置

```
\[
\sqrt[\leftroot{-2}\uproot{12}n]{\frac{a}{x}}
\]
```

$$\sqrt[\leftroot{-2}\uproot{12}n]{\frac{a}{x}}$$

3.6 括号

默认了五种括号大小，除了普通的括号外，从小往大依次是 big, Big, bigg, Bigg，不需要配对

```
\[
\Bigg(\bigg(\Big(\big((\frac{1}{x}\Bigg)
\]
```

$$\Bigg(\bigg(\Big(\big((\frac{1}{x}\Bigg)$$

big Big bigg Bigg 还可以灵活地应用到其他符号，如除号或绝对值符号等

```
\[
\Bigg|\frac{x}{y}\Bigg/\frac{a}{b}\Bigg|
\]
```

$$\Bigg|\frac{x}{y}\Bigg/\frac{a}{b}\Bigg|$$

使用\left(\right)可以自动改变括号大小，注意 **left 和 right 必须配对使用**

```
\[
\left(\frac{1}{x}\right)
\]
```

$$\left(\frac{1}{x}\right)$$

括号可以多重配对

```
\[
\left(1+\left(\frac{1}{x}\right)+f(x)\right)
\]
```

$$\left(1+\left(\frac{1}{x}\right)+f(x)\right)$$

方括号、花括号、绝对值和范数等，都支持\left和\right配对方式，

```
\[
\left[\frac{1}{x}\right]
\left\{\frac{1}{x}\right\}
\left|\frac{1}{x}\right|
\left\|\frac{1}{x}\right\|
\]
```

$$\left[\frac{1}{x}\right]\left\{\frac{1}{x}\right\}\left|\frac{1}{x}\right|\left\|\frac{1}{x}\right\|$$

left(和 right) 中间不能换行，要换行需要使用\right.和\left.辅助配对。

```
\[
\begin{aligned}
xxx \left(\int_t yyy\right.\\
\left. zzz ... \right)
\end{aligned}
\]
```

$$xxx \left(\int_t yyy \right. \\ \left. zzz... \right)$$

这种方式得到的左右括号大小并不一致，可以采用\vphantom{里面内容不显示}方式配对。

```
\[
\begin{aligned}
xxx \left(\int_t yyy\right.\\
\left.\vphantom{\int_t} zzz ... \right)
\end{aligned}
\]
```

$$xxx \left(\int_t yyy \right. \\ \left. zzz... \right)$$

从上面的例子也可以看出\left.对齐的地方是\left(的位置。

自动配对括号有时也并不完美，可以换种形式表达相同的公式，比如指数形式。这里就不举例了。

cases 和单边括号

最常用的单边括号是左花括号，内置了 cases 环境，cases 环境左边是已经设置好对齐的，不需要额外对齐符

```
\[
f(x)=
\begin{cases}
ax&x>0\\
bx^2 &x<0
\end{cases}
\]
```

$$f(x)= \begin{cases} ax & x>0 \\ bx^2 & x<0 \end{cases}$$

右单边括号

```
\[
\left.
\begin{aligned}
&ax\\
&bx^2
```

```
\end{aligned}
\right\}\text{单边大括号}
\]
```

$$\left.\begin{aligned} ax \\ bx^2 \end{aligned}\right\} \text{单边大括号}$$

mathtools 包定义的 rcases 环境，右单边大括号，参见第116页。

3.7 积分

积分用\int_{下标}^{上标}表示，上下标如果只有一个数字或字母，花括号可以省去。养成良好编程习惯，最好不省去花括号

```
\[
\int_{a}^{b} f(x)dx=120
\]
```

$$\int_{a}^{b} f(x)dx=120$$

默认的积分有四重，分别是 int, iint, iiint, iiiint

```
\[
\int f(x)dx  \iint_{S} f(x,y)dxdy  \iiint_{V} f(x,y,z)dxdydz
\iiiint_{\Omega} f(x,y,z,t)dxdydzdt
\]
```

$$\int f(x)dx \iint_{S} f(x,y)dxdy \iiint_{V} f(x,y,z)dxdydz \iiiint_{\Omega} f(x,y,z,t)dxdydzdt$$

多重积分的写法为

```
\[
\idotsint_\Omega f(x_1,\dots,x_k)
\]
```

$$\idotsint_\Omega f(x_1,\dots,x_k)$$

环积分

环积分需要加载 esint 包

```
\usepackage{esint} %环积分包
```

积分命令前加 o 即可

```
\[
\oint f(x,y)dl
\]
```

$$\oint f(x,y)dl$$

二重环积分

```
\[
\oiint f(x,y)dS
\]
```

$$\oiint f(x,y)dS$$

bigints 包，超大积分号。公式太大了或者换行时你想让积分号包含所有行，可以使用这个包。

```
\usepackage{bigints}
```

默认的积分上下脚标写到侧面，如果希望写到下面，使用下面的语句

```
\[
\int\limits _{A}^{B} f(x)dx
\]
```

$$\int\limits_{A}^{B} f(x)dx$$

如果一篇文章中希望积分脚标都使用上下形式，可以加载 amsmath 包的选项

```
\usepackage[intlimits]{amsmath}
```

注意如果加载了 esint 包，必须也添加相同选项，否则会使得 amsmath 的 [intlimits] 选项失效

```
\usepackage[intlimits]{amsmath}
\usepackage[intlimits]{esint}
```

3.8 矢量

我习惯用黑体表示矢量或矩阵或张量，加载 bm 包，bm 包是标准 LaTeX 包，不是 ams 体系。bm 包的黑体保持斜体，有些书的公式黑体是正体的，使用的是\mathbf{}。

```
\usepackage{bm} %黑体且斜体
```

叉乘，这里可以比较一下\bm{}和\mathbf{}效果

```
\[
\nabla\times\bm{A}\hspace{1cm}\nabla\times\mathbf{A}
\]
```

$$\nabla\times\boldsymbol{A}\qquad\nabla\times\mathbf{A}$$

点乘

```
\[
\nabla\cdot\bm{A}
\]
```

$$\nabla\cdot\boldsymbol{A}$$

梯度

```
\[
\bm{E}=\nabla\phi(\bm{r})
\]
```

$$\boldsymbol{E}=\nabla\phi(\boldsymbol{r})$$

梯度算符，这里学习偏微分符号

```
\[
\nabla=\bm{i}\frac{\partial}{\partial
x}+\bm{j}\frac{\partial}{\partial y}+\bm{k}\frac{\partial}{\partial z}
\]
```

$$\nabla=\boldsymbol{i}\frac{\partial}{\partial x}+\boldsymbol{j}\frac{\partial}{\partial y}+\boldsymbol{k}\frac{\partial}{\partial z}$$

时间微分，内置定义到四重，分别是 dot, ddot, dddot, ddddot

```
\[
\dot{\bm{E}}\hspace{3mm}\ddddot{x}
\]
```

$$\dot{\boldsymbol{E}}\quad\ddddot{x}$$

有些书习惯用上箭头表示矢量

```
\[
\vec{E}\vec{\nabla}
\]
```

$$\vec{E}\vec{\nabla}$$

3.9 矩阵

矩阵环境是圆括号 pmatrix, 方括号 bmatrix, 花括号 Bmatrix, 绝对值 vmatrix, 模 Vmatrix，行用换行符\\分隔，列用表格单元格符号 & 分隔

```
\[
\begin{pmatrix}
a &b\\
c &d\\
\end{pmatrix}
\]
```

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix}$$

跨越数列的点号，\dots是位置偏下的三个横点，\cdots是位置居中的三个横点，\vdots是三个竖点，\ddots是对角线方向的三个点。

```
\[
\begin{pmatrix}
a &b &c &\cdots &z\\
b &c &d &\cdots &a\\
\vdots &\vdots &\vdots &\ddots &\cdots\\
z &a &b &\cdots &y\\
\end{pmatrix}
\]
```

$$\begin{pmatrix} a & b & c & \cdots & z \\ b & c & d & \cdots & a \\ \vdots & \vdots & \vdots & \ddots & \cdots \\ z & a & b & \cdots & y \end{pmatrix}$$

二项式为\binom, \dbinom和\tbinom

```
\[
2^k-\binom{k}{1}2^{k-1}+\binom{k}{2}2^{k-2}
\]
```

$$2^k - \binom{k}{1}2^{k-1} + \binom{k}{2}2^{k-2}$$

不用加括号的情况用 array 环境，本质就是公式里的表格，必须指定列数和列的对齐方式

```
\[
\begin{array}{cc}
a&b\\c&d
\end{array}
\]
```

$$\begin{array}{cc} a & b \\ c & d \end{array}$$

nicematrix 包提供了更多的矩阵形式，且手册一直在更新。

```
\usepackage{nicematrix}
```

3.10 狄拉克符号

左矢\langle和右矢\rangle，\hat是算子符号

```
\[
\langle i|\hat{H}|j\rangle=H_{ij}
\]
```

$$\langle i|\hat{H}|j\rangle = H_{ij}$$

正交性

```
\[
\langle i|j\rangle=\delta_{ij}
\]
```

$$\langle i|j\rangle = \delta_{ij}$$

完备性

```
\[
\sum_i |i\rangle\langle i|=\hat{I}
\]
```

$$\sum_i |i\rangle\langle i| = \hat{I}$$

左右矢可以用\left和\right配对

```
\[
\left\langle i\left|\frac{\partial}{\partial
x}\right|j\right\rangle
\]
```

$$\left\langle i\left|\frac{\partial}{\partial x}\right|j\right\rangle$$

3.11 复杂上下标

公式里的上下标不支持换行，换行可使用\substack

```
\[
\sum_{\substack{0<i<m\\0<j<n}} x_{ij}
\]
```

$$\sum_{\substack{0<i<m\\0<j<n}} x_{ij}$$

如果还需要设置对齐方式，可使用 subarray 环境，比如设置左对齐

```
\[
\sum_{\begin{subarray}{l}i\\0<j<n\end{subarray}} x_{ij}
\]
```

$$\sum_{\substack{i\\0<j<n}} x_{ij}$$

设置为居中对齐

```
\[
\sum_{\begin{subarray}{c}i\\0<j<n\end{subarray}} x_{ij}
\]
```

$$\sum_{\substack{i\\0<j<n}} x_{ij}$$

求和号加撇不能直接加，需要是细小间隔或花括号

```
\[
\sum\thinspace'x_i
\sum{'}x_i
\]
```

$$\sum{}'x_i \sum{}'x_i$$

更美观的方式为

```
\[
\sum\nolimits'x_i
\]
```

$$\sum\nolimits' x_i$$

或者

```
\[
\sideset{}{'}\sum_i x_i
\]
```

$$\sideset{}{'}\sum_i x_i$$

\sideset可以在符号前上下和后上下都加入脚标

```
\[
\sideset{_a^b}{_c^d}\prod_i^N x_i
\]
```

$$\sideset{_a^b}{_c^d}\prod_i^N x_i$$

加撇的时候可以省去一个上标号

```
\[
\sideset{_a^b}{_c'}\sum_i^N x_i
\]
```

$$\sum_{i}^{N}{}_{a}^{b}{}_{c}^{\prime} x_i$$

看上去好像这么复杂的脚标不常见，下面是一个复变函数里的积分的例子

```
\[
\sideset{}{_C}\int\limits_a^b f(z)dz
\]
```

$$\int_{a}^{b}{}_{C} f(z)dz$$

3.12 一些比较常见的特殊形式

等号上有文字，使用\overset

```
\[
f(x)\overset{\text{def}}{=}x^2-1
\]
```

$$f(x) \overset{\text{def}}{=} x^2-1$$

等号上有文字，使用\underset

```
\[
f(x)\underset{\text{def}}{=}x^2-1
\]
```

$$f(x) \underset{\text{def}}{=} x^2-1$$

这两个命令对解释公式定义非常有用，比如

```
\[
\text{等压：}\hspace{5mm}\overset{[\text{焓的变化}]}{\Delta
H}=\overset{[\text{内能变化}]}{\Delta
U}+\overset{[\text{对外界做功}]}{P\cdot\Delta
V}=\overset{[\text{从外界获得热量}]}{\Delta Q}
\]
```

$$\text{等压：}\quad \overset{[\text{焓的变化}]}{\Delta H} = \overset{[\text{内能变化}]}{\Delta U} + \overset{[\text{对外界做功}]}{P\cdot\Delta V} = \overset{[\text{从外界获得热量}]}{\Delta Q}$$

公式上面横向大花括号，使用\overbrace

```
\[
\overbrace{a+b+\cdots+z}^n
\]
```

$$\overbrace{a+b+\cdots+z}^{n}$$

公式下面横向大花括号，使用\underbrace

```
\[
\underbrace{a+b+\cdots+z}_n
\]
```

$$\underbrace{a+b+\cdots+z}_{n}$$

箭头上下有文字

```
\[
A\xleftarrow[belowbelow]{above}B
\]
```

$$A \xleftarrow[belowbelow]{above} B$$

```
\[
A\xrightarrow[below]{above}B
\]
```

$$A \xrightarrow[below]{above} B$$

右箭头太常见了，所以有一个简写，

```
\[
\lim_{x\to 0}
\]
```

$$\lim_{x\to 0}$$

3.13 结构型公式

需要加载 amscd 包

```
\usepackage{amscd}
```

amscd 定义好了向左、向右、向上、向下的箭头结构。上下箭头不换行，否则会在一条线上。联用就得到更复杂的公式结构。

```
左上角 @>向右箭头上面的文字>向右箭头下面的文字> 右上角\\
@A上箭头左边文字A上箭头右边文字A @V下箭头左边文字V下箭头右边文字V\\
左下角 @<向左箭头上面的文字<向左箭头下面的文字< 右下角
```

```
\[
\begin{CD}
X @>above>j> T\\
@A{b}A{aa}A @V{b}V{cc}V\\
B @<ii<below< D
\end{CD}
\]
```

$$\begin{array}{ccc} X & \xrightarrow[j]{above} & T \\ {\scriptstyle b}\uparrow{\scriptstyle aa} & & {\scriptstyle b}\downarrow{\scriptstyle cc} \\ B & \xleftarrow[below]{ii} & D \end{array}$$

3.14 标注重点

使用{\color{颜色}换色的部分公式}

```
\[
{\color{red}\sin x}{\color{blue}\cos x}
\]
```

$$\color{red}\sin x\color{blue}\cos x$$

boxed 只能用于 equation，aligned，不能用于其他多行公式

```
\begin{equation}
\boxed{f(x)=\sin x}\cos x
\end{equation}
```

$$\boxed{f(x)=\sin x}\cos x \tag{3-11}$$

加载 empheq 包，可以给公式加框

```
\usepackage{empheq}
```

```
\begin{empheq}[box=\fbox]{equation}
f(x)=\sin x
\end{empheq}
\begin{empheq}[box=\fbox]{align}
f(x)&=\sin x\\
g(x)&=\cos x
\end{empheq}
```

$$\boxed{f(x)=\sin x} \tag{3-12}$$

$$\boxed{\begin{aligned} f(x)&=\sin x \quad (3\text{-}13)\\ g(x)&=\cos x \quad (3\text{-}14) \end{aligned}}$$

或加底色

```
\begin{empheq}[innerbox=\colorbox{yellow!60}]{equation}
f(x)=\sin x
\end{empheq}
```

$$f(x) = \sin x \tag{3-15}$$

3.15 数学符号

数学符号

一些数学符号在 amssymb 包

```
\usepackage{amssymb}
```

笔者平时尽量用 binary 符号，少打字，主要是基本够用。特殊情况下还是要加载其他包的。ams 有自己的数学符号。

常用字体 amsfonts 包

amsfonts 是最常见的数学符号包

```
\usepackage{amsfonts}
```

加载后可以使用\mathcal{}得到下面的结果

```
\[
\mathcal{ABCDEFGHIJKLMNOPQRSTUVWXYZ}
\]
```

$$\mathcal{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$$

使用\mathfrak{}得到

```
\[
\mathfrak{ABCDEFGHIJKLMNOPQRSTUVWXYZ}
\]
```

$$\mathfrak{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$$

\mathfrac这个字体有小写

```
\[
\mathfrak{abcdefghijklmnopqrstuvwxyz}
\]
```

$$\mathfrak{abcdefghijklmnopqrstuvwxyz}$$

使用\mathbb{}得到

```
\[
\mathbb{ABCDEFGHIJKLMNOPQRSTUVWXYZ}
\]
```

$$\mathbb{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$$

mathrsfs

加载 mathrsfs 包可以得到旧花写，很多老书或老文献里会出现

```
\usepackage{mathrsfs}
```

使用\mathscr{}得到旧花写，现在的书一般采用\mathcal{}代替这个字体，但是一些经典的老书或文章会采用这个字体。

```
\[
\mathscr{ABCDEFGHIJKLMNOPQRSTUVWXYZ}
\]
```

$$\mathscr{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$$

oplotsymbl

一些新的不常用的特殊符号需要加载 oplotsymbl 包

```
\usepackage{oplotsymbl}
```

```
\[
\trianglepadot\ \trianglepacross
\]
```

$$\trianglepadot\ \trianglepacross$$

LaTeX 的数学符号非常强大，常用的手册是下面这两个

```
texdoc maths-symbols %常用符号
texdoc symbols %符号大全
```

符号大全笔者也没研究过，非常丰富。符号这块笔者是用多少学多少，用多了也就不用查笔记了。比如 oplotsymbl 包是因为有学生问才去找的，符号大全里也有介绍。

单位

国际单位制包

```
\usepackage{siunitx}
```

微米的 μ 是斜体，因此需要加载 upgreek 包换成直体

```
\usepackage{upgreek}
```

单位包的确很好用，大部分常见单位都定义了简写形式。

投稿时不要轻易使用 siunitx 包，有些杂志社的编辑器可能不支持。采用杂志社给出模板的默认包比较合适，加载杂志社的类后尝试一下命令，如果编译出错就是没有加载这个包。如果杂志社给出的模板有某些包，直接用就行了，一般不会出错。投稿时尽量不要自己加包。

3.16 mathtools

加载时建议分开写

```
\usepackage{amsmath}
\usepackage{mathtools}
```

mathtools 包定义了一些常用的格式细节调整。

rcases 环境，

```
\[
\begin{rcases}
f+ax\\
bx
\end{rcases}
\]
```

$$\begin{rcases} f+ax \\ bx \end{rcases}$$

矩阵对齐

```
\[
\begin{pmatrix*}[r]
a &-b\\
-c &d\\
\end{pmatrix*}
\]
```

$$\begin{pmatrix*}[r] a & -b \\ -c & d \end{pmatrix*}$$

手册有许多细节美感的公式例子，比如\vdots与等号居中对齐

```
\begin{align*}
a&=b\\
&\vdotswithin{=}\\
&=c\\
\shortvdotswithin{=}
&=d
\end{align*}
```

$$\begin{aligned} a &= b \\ &\vdots \\ &= c \\ &\vdots \\ &= d \end{aligned}$$

数学系的可以好好研究一下这个包，大部分常见的结果 amsmath 就足够了。

3.17 表格里的公式

表格里的行间公式就不多说了，如果需要表格里的公式为独立环境，要把整个独立环境放在花括号里。而且不能使用 l 这样的对齐方式，需要加载 array 包，并使用`m{宽度}`这种方式。

```
\documentclass[11pt]{article}

\usepackage{amsmath}
\usepackage{array}

\begin{document}
\begin{table}[ht!]%
\centering
\begin{tabular}{ m{7cm} }
\hline
{\begin{equation}f(x)=\sin x\end{equation}}\\
\hline
{
\begin{align}
&f(x)=\sin x\\
&g(x)=\cos y
\end{align}
}\\
\hline
\end{tabular}
\end{table}
\end{document}
```

3.18 定理

新定理命令是标准 LaTeX 语句，不需要额外的包，可以很方便的自定义新的定理环境，方括号里是可选项，默认编号不带小节。

```
\newtheorem{新定理环境名}{新环境显示名称}[section]
```

```
\newtheorem{homework}{作业}[section]
\begin{homework}[曲线积分]
作业内容
\end{homework}
```

作业 3.18.1 (曲线积分)
作业内容

可以规定某些环境一起编号，下面的例子，joke 环境会单独编号，不会与 homework 同时编号，但是 amuse 环境与 joke 环境混合在一起编号。

```
\newtheorem{新定理环境名}[一起编号的环境名]{新环境显示名称}[section]
```

```
\newtheorem{joke}{笑话}[section]
\newtheorem{amuse}[joke]{开心的事} %不需要再规定 section
\begin{joke}
笑话
\end{joke}
\begin{amuse}
开心一刻
\end{amuse}
```

笑话 3.18.1 笑话

开心的事 3.18.2 开心一刻

使用 ntheorem 包可以方便设定 theorem 的格式。

```
\usepackage{ntheorem}
```

3.19 网页公式

mathjax 支持网页上直接写 LaTeX 公式，下面是一个网络引擎的例子，公式带编号且自动编号，支持 amscd。如果自己做网站，可以将 mathjax 库下载到本地，链接方式改为本地。

```
<!DOCTYPE HTML>
```

```
<html>
<body>

<script type="text/javascript" id="MathJax-script" async src="https
://cdn.bootcss.com/mathjax/2.7.1/latest.js?config=TeX-AMS-
MML_HTMLorMML">
</script>

<script type="text/x-mathjax-config">
  MathJax.Hub.Config({ TeX: { equationNumbers: { autoNumber: "AMS" }
   } });
</script>

<script type="text/x-mathjax-config">
 MathJax.Hub.Config({extensions: ["tex2jax.js"],jax: ["input/TeX", "
 output/HTML-CSS"],tex2jax: {inlineMath: [ ['$','$'], ["\\(","\\)"]
 ],displayMath: [ ['$$','$$'], ["\\[","\\]"] ],processEscapes: true
 },"HTML-CSS": { availableFonts: ["TeX"] }, });
</script>

\begin{equation}
f(x)
\end{equation}

in text, \(\sin x\), \(\int f_n(x)dx\)

\[ \cos x \]

\begin{align}
f(x)\\
g(x)
\end{align}

$\int_a f(x)dx$

\[\require{AMScd}
\begin{CD}
A @>j>> T\\
@AA{aa}A @VV{cc}V\\
B @= D
```

```
\end{CD}
\]

</body>
</html>
```

写本书时这个国外的网络引擎是可用的，有时打开会很慢。网络引擎也有国内地址的，但是不同版本号支持的强度不同，有的版本支持 amscd，有的不支持，需要实际测试一下。

建网站的时候把 mathjax 下载到本地，链接到本地目录即可。

第四章　画图包 tikz

LaTeX 体系有两大矢量画图包体系，pstricks 和 tikz。之所以称为体系，是因为基于这两个包衍生出了许多有用的包或库，能够实现许多设计好的功能。还有一些矢量画图包相对比较小众，比如 asymptote。而网上 LaTeX 画图的例子也多是基于这两个包。

tikz 和 pstrics 比较来说，tikz 继承了 LaTeX 忽略空格的特点，语句的鲁棒性更强。语句末尾按 C 语言风格加分号，语法上更规范易懂。但是 tikz 没有定义面这个概念，所以目前在三维上不如 pstricks。比起 pstricks，tikz 画图有一个很大的好处：跟 LaTeX 一样，大部分时候对空格不敏感，这极大地减少了代码纠错的压力。

包的丰富程度上，pstricks 因为发展较早，所以更占优，尤其是 pstricks 有一个化学实验包 pst-labo，做得非常好。但是基于 tikz 的平面化学分子式包 chemfig 功能强大，给 tikz 体系加分不少。两个体系笔者都没有见到化学分子的三维立体球棍模型。

这章主要学习 tikz 画图体系，tikz 可以直接 pdflatex 编译生成 pdf，大部分情况下对 xelatex 的支持也很好。三维公式、数据画图包 pgfplots 讲课的时候分开了，参考书仍然放在 tikz 这一章，因为它们是一个体系的。化学分子包 chemfig 单独为一章。

pgf 是 tikz 的底层，所以 tikz和 pgf是一个文档，下面的命令调出的是同一个文件。

```
texdoc tikz
texdoc pgf
```

加载 tikz 包的语句为

```
\usepackage{tikz}
```

加载 tikz 库的语句为

```
\usetikzlibrary{tikz库}
```

tikz 画图环境为 tikzpicture，这个环境视同文字，很方便图文混排。

```
直线视为行间文字
\begin{tikzpicture}
\draw (0,0)--(1,0);
\end{tikzpicture}
```

直线视为行间文字

tikz 画图语句用分号结尾，如果缺失分号会编译报错。

直接添加到 figure 环境内可以构成独立的图片。

```
\begin{figure}[ht!]
\begin{tikzpicture}
\draw (0,0)--(1,0);
\end{tikzpicture}
\caption{tikz画出的图。}
\end{figure}
```

除了极少数简单图形和动画外，一般情况推荐使用 standalone 类。使用这个类，每一个 tikzpicture 构成 pdf 的一页，大小跟随图片大小变化。笔者一般一个 tex 文件一幅图，生成的 pdf 文件本身是带 tight bounding box，可以作为图片直接使用。文件结构参见第191页。

4.1 直线

4.1.1 坐标

tikz 画图基于坐标点，默认单位是厘米。坐标点可以是 (x,y) 形式，也可以是 (角度: 径矢长度) 形式，哪个方便用哪个。默认长度是 cm，矢量高清图缩放不影响清晰度，所以一般不建议修改默认长度。生成的图形是相对结果，不是绝对结果。所以下面的画图语句结果是一样的，都得到 1cm 的水平线。

```
\begin{tikzpicture}
\draw (0,0)--(1,0);
\end{tikzpicture}
```

```
\begin{tikzpicture}
\draw (10,0)--(11,0);
\end{tikzpicture}
```

```
\begin{tikzpicture}
\draw (0,0)--(0:1);
\end{tikzpicture}
```

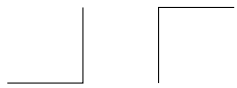
单位可以随时修改，如果在 draw 参数里面修改，对该语句所有坐标适用。

```
\begin{tikzpicture}
\draw (0,0)--(1,0)--(1,5mm)--(2,0);
```

```
\draw [x=5mm,y=5mm] (0,-1)--(0,-2)--(1,-1);
\draw [dashed] (0,-1)--(2,-1);
\end{tikzpicture}
```

划线有两个简省语句，表示先横线后竖线的-|和表示先竖线后横向的|-。

```
\begin{tikzpicture}
\draw (0,0)-|(1,1);
\draw (2,0)|-(3,1);
\end{tikzpicture}
```



可以用内置函数（不需要加载任何库）计算坐标，此时坐标值需要花括号括起来。

```
\begin{tikzpicture}
\draw (0,0)--({2*sqrt(2)*cos(30)},{2*sqrt(2)*sin(30)});
\end{tikzpicture}
```

可以用 turn 修饰坐标，表示转多少角度，走多长距离。turn 修饰的坐标如果是公式计算出来的，需要额外添加花括号。

```
\begin{tikzpicture}
\draw [thick] (0,0)--(1,0)--([turn]30:2)--([turn]{{asin(-0.5)}}:1);
\end{tikzpicture}
```

额外的花括号也可以这么加

```
([turn]{{asin(-0.5)}:1})
```

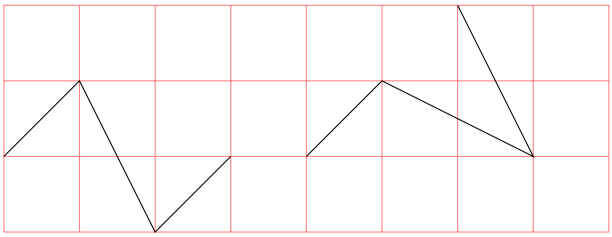
只有一个花括号会报错。

turn 语句没有加号好用，单加号表示当前坐标与起点坐标相加，双加号表示当前坐标与前面的坐标相加。

```
\begin{tikzpicture}
\draw (0,0)--+(1,0)--+(1,1)--+(0,1)--cycle;
\draw [color=brown] (2,0)--++(1,0)--++(0,1)--++(-1,0)--cycle;
\end{tikzpicture}
```

单加号和双加号虽然可以混用，但是不推荐混用。如下面的例子，单加号和双加号会互相影响，增加计算坐标时的困难度。

```
\begin{tikzpicture}
\draw [step=1,red!50] (0,-1) grid (8,2);
\draw (0,0)--+(1,1)--++(2,-1)--+(1,1);
\draw (4,0)--++(1,1)--+(2,-1)--++(1,1);
\end{tikzpicture}
```



turn 语句其实没有双加号语句方便，上面 turn 语句例子的效果用双加号是更简单。

```
\begin{tikzpicture}
\draw [thick] (0,0)--(1,0)--++(30:2)--++(0:1);
\end{tikzpicture}
```

坐标还有一种方式是上下左右，双加号和单加号意思如前面，默认单位是厘米。

```
\begin{tikzpicture}
\draw (0,0)--++(up:1)--++(left:1)--++(down:1)--++(right:1);
\end{tikzpicture}
```

坐标点可以先用\coordinate语句定义，然后调用名字使用，这样可以起名后多次复用。

```
\begin{tikzpicture}
\coordinate (A) at (2,0);
\coordinate (O) at (0,0);
\coordinate (B) at (30:2);
\draw (A)--(O)--(B);
\end{tikzpicture}
```

4.1.2 线参数

线的参数有颜色、线宽、线型、圆角大小等。

内置线宽为 ultra thin, very thin, thin, semithick, thick, very thick, ultra thick。thin 的效果等价于\hrule。自定义线宽的方式是 line width=2pt。

内置线型为 dashed, dotted, dash dot 和 dash dot dot，可以用 loosely 和 densely 修饰这两种线型。

默认是黑色的直的细线，无圆角。

划线讲究一笔画下来，中间不要停顿。最后加 cycle 表示回到起点的闭合线。

```
\begin{tikzpicture}
\draw [color=blue, line width=2pt, rounded corners=5pt]
(0,0)--(1,0)--(1,1)--(0,1)--cycle;
\end{tikzpicture}
```

颜色、线宽、线型等参数是一条线的基本参数，都不可以在一个 draw 语句内改变，但是圆角参数是可以的，圆角参数修饰的是--，所以放在其前面。这个功能在画腔体的时候挺有用的。

```
\begin{tikzpicture}
\draw [color=blue, line width=2pt, rounded corners=5pt]
(0,0)--(3,0)[rounded corners=0pt]--(3,1)[rounded
corners=15pt]--(0,1)[rounded corners=0pt]--cycle;
\end{tikzpicture}
```

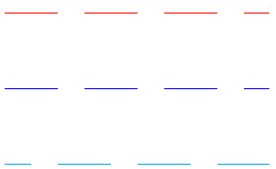
dash 线型

虚线样式可以自定义，on 表示划线，off 表示留白。

```
\begin{tikzpicture}
\draw [dash pattern=on 2pt off 4pt on 4pt off 4pt] (0,0)--(6,0);
\end{tikzpicture}
```

可以使用 phase 来移动线型，默认 phase 是 0pt，但是语句不一样，不设置 phase 时需要加 pattern。

```
\begin{tikzpicture}
\draw [red] [dash pattern=on 20pt off
10pt] (0,0) -- ++(3.5,0);
\draw [blue] [dash=on 20pt off 10pt phase
0pt] (0,-1) -- ++(3.5,0);
\draw [cyan] [dash=on 20pt off 10pt phase
10pt] (0,-2) -- ++(3.5,0);
\end{tikzpicture}
```



画双色虚线可以使用 postaction，需要准确计算两色交替位置。

```
\begin{tikzpicture}
\draw[postaction={draw,red,dash pattern= on 3pt off
5pt,dash phase=4pt,thick}]
    [blue,dash pattern= on 3pt off 5pt,thick] (0,0)
rectangle (1,1);
\end{tikzpicture}
```

计算量的确比较大，但是如果计算准确，其实可以画多色线。下面这个例子是笔者非常想要的结果。

```
\begin{tikzpicture}
\draw[postaction={draw,red,dash pattern= on 1cm off
3cm,dash phase=0cm,thick}] [postaction={draw,green,dash
pattern= on 1cm off 3cm,dash phase=1cm,thick}]
[postaction={draw,yellow,dash pattern= on 1cm off
3cm,dash phase=2cm,thick}] [blue,dash pattern= on 1cm
off 3cm,dash phase=3cm,thick] (0,0) rectangle ++(1,1);
\end{tikzpicture}
```

double 线型

double 线性可以画出双线的效果，可以定义两种不同的颜色，默认是中间白两边黑，即 draw=black, double=white。可以定义双线的间距 double distance，整个双线的宽度等于两倍的 line width 加上 double distance。double distance between line centers 这个参数，意思是上面和下面 4pt 红线的中线的间距为 5pt，蓝线就只剩下 1pt 了。

```
\begin{tikzpicture}
\draw [double] (0,0)--(2,0);
```

```
\draw [draw=red, double=blue, line width=4pt, double
distance=2pt] (3,0)--++(2,0);
\draw [draw=red, double=blue, line width=4pt, double distance
between line centers=5pt] (5.2,0)--++(2,0);
\end{tikzpicture}
```

4.1.3 箭头和 arrows.meta 库

划线语句的一个非常有用的参数是箭头参数。基本箭头不需要添加额外的库。

```
\begin{tikzpicture}
\draw [->] (0,0)--(1,0);
\draw [<-] (2,0)--++(1,0);
\draw [<->] (4,0)--++(1,0);
\end{tikzpicture}
```

默认箭头太难看了，推荐使用 latex 或 Latex 箭头。

```
\begin{tikzpicture}
\draw [-latex] (0,0)--(1,0);
\draw [latex-] [red] (0,-0.5)--++(1,0);
\draw [Latex-Latex] [blue] (0,-1)--++(1,0);
\end{tikzpicture}
```

这样比较麻烦，加载 arrows.meta 库，加载 tikz 库的命令是\usetikzlibrary{库名}。

```
\usetikzlibrary{arrows.meta}
```

使用>=箭头形状来重新定义默认箭头，例如设置默认箭头大于号和小于号等于 Latex 形式，Latex 可以添加参数，小写的 latex 不能。双箭头用两个大于号或小于号，多个箭头用多个大于小于号。sep 调整多个箭头的间距。加点可以多个箭头是断续的。

```
\begin{tikzpicture}[>={Latex[width=4pt,length=10pt]}]
\draw [{<<[sep=2mm]}->>] (0,0)--(3,0);
\draw [{<<[sep=2mm].<[sep=2mm]}->.>>] (4,0)--++(4,0);
\end{tikzpicture}
```

或者使用 scale width 和 scale length

```
\begin{tikzpicture}[>={Latex[scale
width=1.5,scale length=2]}]
\draw [<->] (0,0)--(2,0);
\end{tikzpicture}
```

使用 reversed 可以反转箭头方向

```
\begin{tikzpicture}[>={Latex[reversed]}]
\draw [<->] (0,0)--(2,0);
\end{tikzpicture}
```


可以只画一半

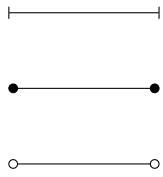
```
\begin{tikzpicture}
\draw [-{Latex[harpoon]}] (0,0)--(2,0);
\draw [-{Latex[harpoon,swap]}] (3,0)--++(2,0);
\draw [-{Latex[right]}] (6,0)--++(2,0);
\draw [-{Latex[left]}] (9,0)--++(2,0);
\end{tikzpicture}
```


可以设置线与箭头不同色，还可以设置不同的箭头填充颜色和箭头线宽（默认跟随线的宽度）

```
\begin{tikzpicture}
\draw [-{Latex[color=black]}] [red, line width=1pt] (0,0)--(2,0);
\draw [-{Latex[color=blue,fill=yellow, line width=0.5pt]}] [red,
line width=2pt] (3,0)--++(2,0);
\end{tikzpicture}
```

arrows.meta 库包含了许多箭头类型，比如直线和弯勾，用 scope 环境可以很方便更换箭头类型。参见 tikz 手册 16 小节和 16.5 小节。

```
\begin{tikzpicture}[>=Bar]
\draw [<->] (0,0)--(2,0);
\draw [Circle-Circle] (0,-1)--++(2,0);
\begin{scope}[>={Circle[open]}]
\draw [<->] (0,-2)--++(2,0);
\end{scope}
\end{tikzpicture}
```



可以多种箭头联用，且对每个箭头都定义不同的参数，每种箭头支持的参数可以查阅手册，上面的例子都是普遍的参数。

```
\begin{tikzpicture}
\draw [-{Latex[red, sep=1mm] . Stealth[blue, sep=1mm].
Stealth[yellow,sep=1mm] . Circle[red]}](0,0)--(2,0);
\end{tikzpicture}
```


还有一种特殊的用法，下面的语句得到一个向上的箭头，单边也行。

```
\begin{tikzpicture}[>=latex]
\draw [<->](0,0);
\end{tikzpicture}
```

4.2 矩形、圆、椭圆和弧

tikz 内置了画矩形、圆、椭圆和弧的命令。

4.2.1 矩形

画矩形的语句有两种，第一种是给出一对对角的坐标值。

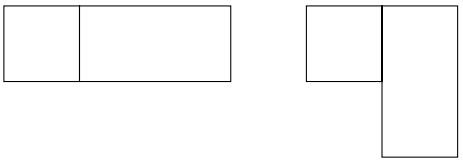
```
\begin{tikzpicture}
\draw [green] (x1,y1) rectangle (x2,y2);
\end{tikzpicture}
```

另一种是利用加号，给出一个角的坐标值和用双加号获得的另一个角的坐标值，即长宽方向的延展长度。

```
\begin{tikzpicture}
\draw [green] (x1,y1) rectangle ++(x2,y2);
\end{tikzpicture}
```

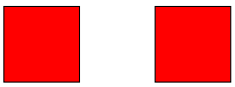
如果只画一个矩形，单加号和双加号没有区别。但是如果一个语句连着画两个矩形，单加号和双加号的区别就很大。

```
\begin{tikzpicture}
\draw (0,0) rectangle ++(1,1) rectangle ++(2,-1);
\draw (4,0) rectangle +(1,1) rectangle +(2,-1);
\end{tikzpicture}
```



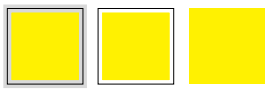
凡是闭合曲线都可以填充颜色，默认边框颜色为黑色，填充为白色，所以只给出 fill 的颜色还是会有边框颜色。

```
\begin{tikzpicture}
\draw [fill=red] (0,0) rectangle ++(1,1);
\draw [draw=black,fill=red] (2,0) rectangle ++(1,1);
\end{tikzpicture}
```



如果不希望有边框颜色，可以采用\fill语句，或将边框设置为白色，但是这两种方式效果不同。\fill语句填充范围是带边框中线围成的区域，而[draw=white] 方式填充范围是不带边框的内部区域。这个例子也可以看出边框宽度是从中线两边扩展的。

```
\begin{tikzpicture}
\draw [draw=gray!30,fill=yellow,line width=3pt] (0,0) rectangle
++(1,1);
\draw [draw=white,fill=yellow,line width=3pt] (1.2,0) rectangle
++(1,1);
\fill [fill=yellow,line width=3pt] (2.4,0) rectangle ++(1,1);
\draw [very thin] (0,0) rectangle ++(1,1);
\draw [very thin] (1.2,0) rectangle ++(1,1);
\end{tikzpicture}
```



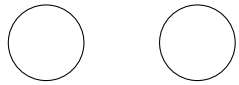
任何闭合曲线都可以填充，即使不闭合的曲线也可以填充，不闭合曲线填充时边框会少一个边。

```
\begin{tikzpicture}
\draw [fill=yellow,line width=2pt] (0,0)-| ++(1,1)--cycle;
\draw [fill=yellow,line width=2pt] (2,0)-| ++(1,1);
\end{tikzpicture}
```

4.2.2 圆和圆弧

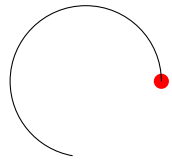
画圆的语句需要给出圆心坐标和半径。有一种简写方式，[radius=0.5] 可以用 (0.5) 代替。

```
\begin{tikzpicture}
\draw (0,0) circle [radius=0.5];
\draw (2,0) circle (0.5);
\end{tikzpicture}
```



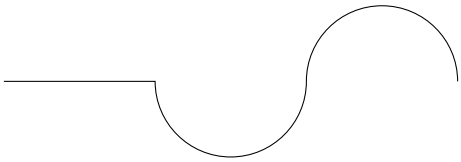
圆弧需要给出起点坐标、半径和起始角度。角度取值范围是 $[-360, 360]$ 区间。

```
\begin{tikzpicture}
\fill [red] (0,0) circle (0.1);
\draw (0,0) arc [start angle=0, end angle=260,
radius=1];
\end{tikzpicture}
```



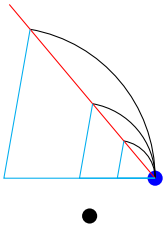
这好像与我们习惯的思维方式不同，圆规画弧是圆心坐标、半径和起始角度。从起点坐标开始画图其实是数控运动的一般情况，这样可以做到直线和弧线的连接。两个弧线语句可以连续使用，第一个弧线画完后，坐标值就移到终点，第二个弧线以第一个弧线终点开始继续画弧。

```
\begin{tikzpicture}
\draw (0,0)--(2,0) arc [start angle=180, end angle=360, radius=1]
arc [start angle=180, end angle=0, radius=1];
\end{tikzpicture}
```



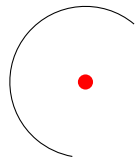
上面只是一个简单的例子，很容易确定圆心，那么从起点坐标画弧的圆心到底在什么位置呢？可以练习一下下面这个例子。起点坐标和起始角度都相同情况下，半径不同，圆心也不同。始点坐标和圆心都在一条直线上。

```
\begin{tikzpicture}
\fill (0,0) circle (0.1);
\fill [blue] (30:1) circle (0.1);
\draw (30:1) arc [start angle=0, end angle=80, radius=0.5];
\draw (30:1) arc [start angle=0, end angle=80, radius=1];
\draw (30:1) arc [start angle=0, end angle=80, radius=2];
\draw [red] (30:1)--++(130:3);
\draw [cyan] (30:1)--++(180:0.5)--++(80:0.5);
\draw [cyan] (30:1)--++(180:1)--++(80:1);
\draw [cyan] (30:1)--++(180:2)--++(80:2);
\end{tikzpicture}
```



很多时候我们希望像圆规那样画图，可以采用 shift 参数。shift 表示平移，坐标需要用花括号括起来。

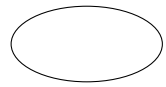
```
\begin{tikzpicture}
\fill [red] (1,0) circle (0.1);
\draw [shift={(50:1)}] (1,0) arc [start
angle=50, end angle=260, radius=1];
\end{tikzpicture}
```



4.2.3 椭圆和椭圆弧

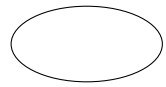
画椭圆的语句需要给出圆心坐标、x 半径和 y 半径。

```
\begin{tikzpicture}
\draw (0,0) ellipse [x radius=1, y radius=0.5];
\end{tikzpicture}
```



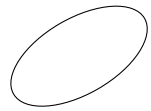
有一种简写方式，[x radius=1, y radius=0.5] 可以用 (1 and 0.5) 代替。

```
\begin{tikzpicture}
\draw (0,0) ellipse (1 and 0.5);
\end{tikzpicture}
```



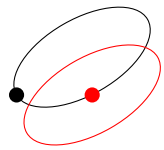
基本语句画的都是正向椭圆，通过旋转来改变椭圆方向。rotate 表示围绕原点 (0,0) 旋转，后面的参数是角度。

```
\begin{tikzpicture}
\draw [rotate=30] (0,0) ellipse (1 and 0.5);
\end{tikzpicture}
```



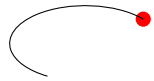
当希望围绕椭圆圆心旋转时，需要 rotate around={角度:(旋转基准点坐标)}

```
\begin{tikzpicture}
\fill (0,0) circle (0.1);
\draw [rotate=30] (1,0) ellipse (1 and 0.5);
\fill [red] (1,0) circle (0.1);
\draw [red,rotate around={30:(1,0)}] (1,0)
ellipse (1 and 0.5);
\end{tikzpicture}
```



椭圆弧也是 arc，需要给出起点坐标、起始角度、x 半径和 y 半径。

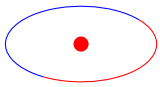
```
\begin{tikzpicture}
\fill [red] (0,0) circle (0.1);
\draw (0,0) arc [start angle=40, end angle=240, x
radius=1, y radius=0.5];
\end{tikzpicture}
```



可以通过平移来获得围绕圆心画椭圆。

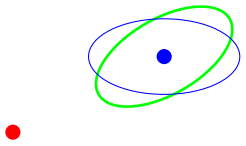
```
\begin{tikzpicture}
\fill [red] (2,0) circle (0.1);
\draw [blue] [shift={({1+cos(40)},{0.5*sin(40)})}] (1,0) arc
[start angle=40, end angle=240, x radius=1, y radius=0.5];
```

```
\draw [red] [shift={({1+cos(-120)},{0.5*sin(-120)})}] (1,0) arc
[start angle=-120, end angle=40, x radius=1, y radius=0.5];
\end{tikzpicture}
```



scope 环境也可以用来做平移和旋转，注意顺序，写到后面的平移先操作，然后围绕椭圆心（此时已经平移到了 (2,1) 点）旋转 30 度。使用 xshift 和 yshift 是必须加单位。使用 shift=(x,y) 时，不加单位默认是厘米，加单位则为指定单位。

```
\begin{tikzpicture}
\fill [red] (0,0) circle (0.1);
\begin{scope}[color=green, line width=1pt, rotate
around={30:(2,1)}, xshift=2cm,yshift=1cm]
\draw (0,0) ellipse (1 and 0.5);
\end{scope}
\fill [blue] (2,1) circle (0.1);
\draw [blue] (2,1) ellipse (1 and 0.5);
\end{tikzpicture}
```



所有的弧线都可以加箭头。

```
\begin{tikzpicture}
\draw [-latex] (0,0) arc [start angle=30,
end angle=120, radius=1];
\end{tikzpicture}
```

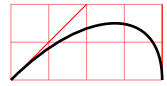
4.3 曲线

4.3.1 任意曲线 controls 方式

语句 controls 的设计理念很不错，但是实际并不好用。controls 的用法如下所示，controls 后面的第一个坐标与前面的坐标（即该段曲线的起点坐标）构成该段曲线起点处的切线，第二个坐标与后面的坐标（即该段曲线的

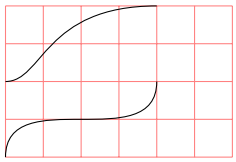
终点坐标）构成该段曲线终点处的切线。所以使用 controls 语句需要计算构成切线的坐标值。

```
\begin{tikzpicture}
\draw [step=0.5,red!50] (0,0) grid (2,1);

\draw [red] (0,0)--(1,1);
\draw [red] (2,1)--(2,0);
\draw [line width=1pt] (0,0) .. controls (1,1)
and (2,1) .. (2,0);
\end{tikzpicture}
```



或者用 left、right、up、down 方式。

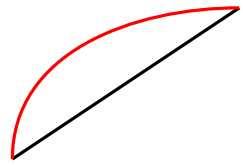
```
\begin{tikzpicture}
\draw [step=0.5,red!50] (0,-1) grid (3,1);

\draw (0,0) .. controls +(right:0.5) and +(left:1.5) .. (2,1);
\draw (0,-1) .. controls +(up:1) and +(down:1) .. (2,0);
\end{tikzpicture}
```



4.3.2 任意曲线 bend 方式

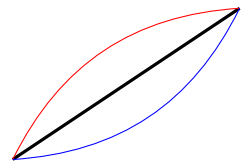
bend 方式定义的是出角和入角，即切向方向的角度。

```
\begin{tikzpicture}[very thick]
\draw (0,0) to (3,2);
\draw [red] (0,0) to [out=90,in=180] (3,2);
\end{tikzpicture}
```



也可以 bend 左右，但是没有上下。

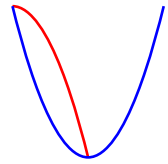
```
\begin{tikzpicture}
\draw [very thick] (0,0) to (3,2);
\draw [red] (0,0) to [bend left] (3,2);
\draw [blue] (0,0) to [bend right] (3,2);
\end{tikzpicture}
```



4.3.3 抛物线

比较常用的曲线形状是抛物线，默认是起点弯曲的，可以修改为终点弯曲。

```
\begin{tikzpicture}
\draw [red,line width=1pt] (-1,2) parabola
(0,0);
\draw [blue,line width=1pt] (-1,2) parabola
[bend at end] (0,0) parabola (1,2);
\end{tikzpicture}
```



4.3.4 正弦余弦曲线

正弦余弦曲线不是很好用，偶尔需要单独画一小段时使用。

```
\begin{tikzpicture}
\draw (0,0)--(1.57,1);
\draw [color=red,thick] (0,0) sin (1.57,1);
\end{tikzpicture}
```

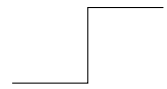
完整的一条曲线很麻烦，下面有更容易的方法。

```
\begin{tikzpicture}
\draw [x=1.57cm,y=1cm] (0,0) sin (1,1) cos (2,0) sin (3,-1) cos
(4,0);
\end{tikzpicture}
```

4.3.5 plot 语句

plot 用来函数画图。最简单的是将一系列点连成线。

```
\begin{tikzpicture}
\draw plot coordinates {(0,0) (1,0) (1,1) (2,1)};
\end{tikzpicture}
```



函数画图，使用函数时或表达式太长时都需要花括号括起来。三角函数时，r 表示弧度。默认变量是\x，不需要定义即可使用。红色线是定义\t做变量。

```
\begin{tikzpicture}
\draw [line width=1pt,color=blue] [domain=3:5,smooth,
samples=100] plot (\x,{(\x-4)*(\x-4)});
\draw [line width=1pt,color=red] [domain=-2.5:2.5,smooth,
samples=100,variable=\t] plot (\t,{exp(-\t*\t)*cos(16*\t r)});
\end{tikzpicture}
```

tikz 内置了一些常用函数，还内置了随机数。rand 表示 $[-1,+1]$ 区间的伪随机数，rnd 为 $[0,1]$ 区间。

```
\begin{tikzpicture}
\draw plot [only marks, mark=ball, mark size=1.5, domain=-1:1,
samples=20] (\x,rand*1.2);
\end{tikzpicture}
```

4.4 一些功能

4.4.1 标识角度 angles 库

因为我们经常会遇到画两条直线之间夹角的情况，所以有一个专门的库 angles做这个事情。

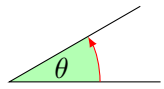
```
\usetikzlibrary{angles}
```

如果给角度加文字注释，还需要 quotes 库。

```
\usetikzlibrary{quotes}
```

文字注释需要双引号括起来。

```
\begin{tikzpicture}
\coordinate (A) at (2,0);
\coordinate (B) at (0,0);
\coordinate (C) at (30:2);
\draw (A)--(B)--(C) pic
[draw=red,-latex,fill=green!30,angle
radius=12mm,"$\theta$"] {angle=A--B--C};
\end{tikzpicture}
```



还可以在角度标签后面用花括号加参数，比如换个颜色，换个字体大小。

```
\draw (A)--(B)--(C) pic [draw=red,-latex,fill=green!30,angle radius
=12mm, "$\theta$"{blue,font=\tiny}] {angle=A--B--C};
```

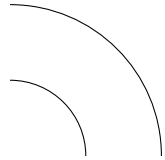
或者用 shift 换个位置，当不采用默认单位厘米时，需要指定单位。shift 参数也可以分开写为 {xshift=7mm, yshift=2mm}

```
\draw (A)--(B)--(C) pic [draw=red,-latex,fill=green!30,angle radius
=8mm,"$\theta$"{shift={(6mm,1.5mm)}}] {angle=A--B--C};
```

4.4.2 剪切 clip

clip 语句可以设置一个闭合曲线框，超出该区域的部分会被剪切掉。下面的例子，最后得到的图形大小是剪切框的大小，而不是圆的大小。

```
\begin{tikzpicture}
\clip (0,0) rectangle (2,2);
\draw (0,0) circle (1);
\draw (0,0) circle (2);
\end{tikzpicture}
```

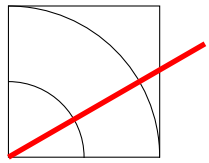


如果加参数 [draw] 会显示这个剪切框。

```
\clip [draw] (0,0) rectangle (2,2);
```

将剪切放在 scope 里，只在 scope 里做剪切，不影响下面的结果，所以红色线不会被剪切。

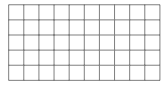
```
\begin{tikzpicture}
\begin{scope}
\clip [draw] (0,0) rectangle (2,2);
\draw (0,0) circle (1);
\draw (0,0) circle (2);
\end{scope}
\draw [red,line width=2pt] (0,0)--(30:3);
\end{tikzpicture}
```



4.4.3 网格 grid

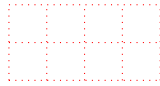
打网格的语句是 grid，help lines 是默认的辅助线 style 名，默认设置是 very thin 的灰色线。

```
\begin{tikzpicture}
\draw [step=0.2, help lines] (0,0) grid (2,1);
\end{tikzpicture}
```



可以直接设置颜色和线型等参数。

```
\begin{tikzpicture}
\draw [step=0.5, red, dotted] (0,0) grid (2,1);
\end{tikzpicture}
```



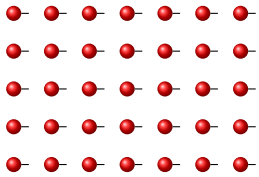
4.5 循环 foreach

foreach 语句用来重复画图，注意 foreach 语句后面没有分号。下面的单行语句可以省略花括号，多行语句必须加花括号。如果是等间距的一系列值，可以省略中间结果，写三个值就行，其余用三个点代替。如果是不等间距的则需要全部列举。

```
\begin{tikzpicture}
\foreach \x in {0,1,...,6}
{
\fill [fill=red] (\x,0) circle (0.3);
\fill [fill=blue] (\x,0) circle (0.1);
}
\end{tikzpicture}
```

可以是多个变量。

```
\begin{tikzpicture}
\foreach \x in {0,0.5,...,3}
\foreach \y in {0,0.5,...,2}
{
\shade [ball color=red] (\x,\y) circle
(0.1);
\draw (\x+0.1,\y)--(\x+0.2,\y);
}
\end{tikzpicture}
```



也可以是变量配对。

```
\begin{tikzpicture}
\foreach \a/\c in {20/red,40/blue,60/brown}
\draw [\c] (0,0)--(\a:1);
\end{tikzpicture}
```

foreach 语句在画晶体结构这样重复性高的情况非常有用。

4.6 渐变色和混色

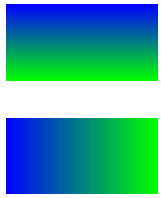
4.6.1 渐变 shade

\shade用来做渐变色效果。默认渐变色为黑白，默认渐变方向是上下渐变。

```
\begin{tikzpicture}
\shade (0,0) circle (1);
\end{tikzpicture}
```

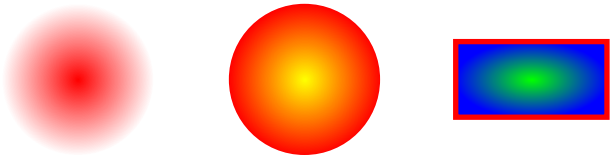
可以设置为上下渐变色或左右渐变色。

```
\begin{tikzpicture}
\shade [top color=blue,bottom color=green]
(0,0) rectangle (2,1);
\shade [left color=blue,right color=green]
(0,-1.5) rectangle ++(2,1);
\end{tikzpicture}
```



或者按径向渐变，如果没有设置 outer color，默认是白色。

```
\begin{tikzpicture}
\shade [shading=radial, inner color=red] (0,0) circle (1);
\shade [shading=radial, inner color=yellow, outer color=red]
(3,0) circle (1);
\shade [inner color=green, outer color=blue, draw=red, line
width=2pt] (5,-0.5) rectangle ++(2,1);
\end{tikzpicture}
```



还可以设置角度使得渐变色方向倾斜

```
\begin{tikzpicture}
\shade [shading=axis, left color=yellow, right
color=blue, shading angle=45] (0,-2) rectangle
++(2,1);
\end{tikzpicture}
```

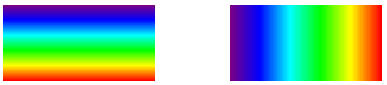
tikz 的 ball color 非常漂亮，有三维立体感。

```
\begin{tikzpicture}
\shade [ball color=red] (0,0) ellipse (1 and 0.5);
\shade [ball color=green] (2,0) circle (0.5);
\definecolor{tjublue}{RGB}{0,70,140}
\shade [ball color=tjublue] (4,0) circle (0.5);
\end{tikzpicture}
```

彩虹色需要自己设置彩虹效果。

```
%设置彩虹色，rainbow 是自己起的名字，里面的数值可变
\pgfdeclareverticalshading{rainbow}{100bp}{color(0bp)=(red);color(25bp)=(red);
color(35bp)=(yellow);color(45bp)=(green);color(55bp)=(cyan);color(65bp)=(blue);

%使用彩虹色
\begin{tikzpicture}
```

```
\shade[shading=rainbow] (0,0) rectangle (2,1);

\shade[shading=rainbow,shading angle=90] (3,0) rectangle +(2,1);
\end{tikzpicture}
```



以上都不需要额外的库。shadings 库定义了颜色轮，颜色轮的代码很复杂。

```
\usetikzlibrary{shadings}
```

定义了三种颜色轮，color wheel, color wheel white center, color wheel black center，颜色环是通过奇偶混色原则做成的。

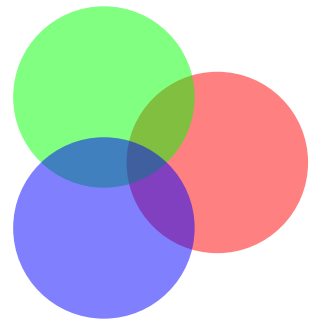
```
\begin{tikzpicture}
\shade [shading=color wheel] (0,0) circle (1);
\shade [shading=color wheel white center] (3,0) circle (1);
\shade [shading=color wheel,even odd rule] (6,0) circle (1) (6,0)
circle (0.5);
\end{tikzpicture}
```

颜色轮的代码计算量很大，如果想做多周期色彩环，可以用 pgfplots 包的函数画图方式做。

4.6.2 透明 transparent

通过 opacity 参数设置填充色的透明度。

```
\begin{tikzpicture}[opacity=0.5]
\fill [red] (0:1cm) circle (1.2);
\fill [green] (120:1cm) circle (1.2);
\fill [blue] (-120:1cm) circle (1.2);
\end{tikzpicture}
```



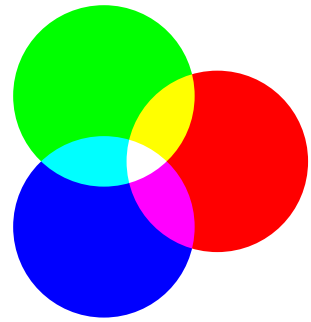
透明的效果是浅色，但是浅色不等于透明，画光路图时可以用透明来显示透明、分束器的效果。

```
\begin{tikzpicture}
\draw [red,line width=2pt] (0,0)--(2.5,0);
\fill [cyan!25] (1,0) ellipse (0.08 and 0.4);
\fill [cyan,opacity=0.3] (2,0) ellipse (0.08 and
0.4);
\end{tikzpicture}
```


4.6.3 光学混色 blend

光学三原色为红绿蓝，三色混合是白色，定义 blend group=screen 可以获得光学三原色的混色效果。

```
\begin{tikzpicture}[blend group=screen]
\fill [red] (0:1cm) circle (1.2);
\fill [green] (120:1cm) circle (1.2);
\fill [blue] (-120:1cm) circle (1.2);
\end{tikzpicture}
```



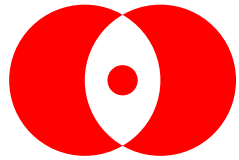
blend group 参数还有多个参数，其他效果可以自行练习或参考手册 23.3 节。

4.6.4 奇偶原则

tikz 的画图原则是覆盖，先画的在下面，后画的在上面。如果两个图案写在一句内，可以定义奇偶原则。奇偶原则：同向的（比如都是逆时针或顺

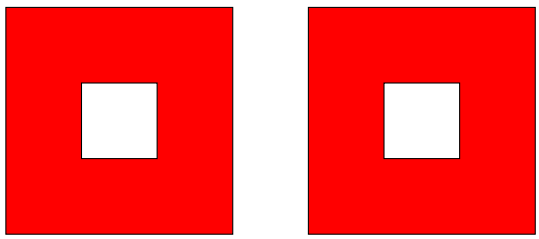
时针）图案，偶数为无填充，奇数为有填充。

```
\begin{tikzpicture}
\fill [fill=red,even odd rule] (0,0) circle (1)
(1,0) circle (1) (0.5,0) circle (0.2);
\end{tikzpicture}
```



顺时针图形和逆时针图形默认是 nonzero rule，就是一个是顺指针一个是逆时针，重叠部分是无填充。圆和矩形是同向的，所以需要加奇偶原则才能起作用。

```
\begin{tikzpicture}
\fill [draw=black,fill=red] (0,0)--(0,1)--(1,1)--(1,0)--cycle
(-1,-1)--++(3,0)--++(0,3)--++(-3,0)--cycle;
\fill [draw=black,fill=red]
(4,0)--++(1,0)--++(0,1)--++(-1,0)--cycle
(3,-1)--++(0,3)--++(3,0)--++(0,-3)--cycle;
\end{tikzpicture}
```



4.7 分层

tikz 画图默认是先画的在下面，后画的在上面。tikz 可以很方便定义层和层顺序，打破原来的覆盖原则。

```
\pgfdeclarelayer{层名称A}
\pgfdeclarelayer{层名称B}
\pgfsetlayers{层顺序，从底部到上面的顺序}
\begin{pgfonlayer}{层名称A}
环境内画图语句的结果在A层
\end{pgfonlayer}
```

下面的例子定义了五层，规定层顺序为 ABCDE，A 在最下面，E 在最上面，虽然先画的是 E 的蓝色线，但是最后呈现的是 E 的蓝色线。

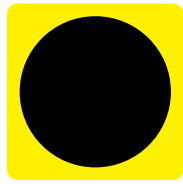
```
\begin{tikzpicture}
\pgfdeclarelayer{A}
\pgfdeclarelayer{B}
\pgfdeclarelayer{C}
\pgfdeclarelayer{D}
\pgfdeclarelayer{E}
\pgfsetlayers{A,B,C,D,E}

\begin{pgfonlayer}{E}
\draw [blue,line width=2pt] (0,0)--(2,0);
\end{pgfonlayer}

\begin{pgfonlayer}{D}
\draw [cyan,line width=2pt] (0,0)--(2,0);
\end{pgfonlayer}

\begin{pgfonlayer}{C}
\draw [green,line width=2pt] (0,0)--(2,0);

\end{pgfonlayer}

\begin{pgfonlayer}{B}
\draw [red,line width=2pt] (0,0)--(2,0);

\end{pgfonlayer}

\begin{pgfonlayer}{A}
\draw [black,line width=2pt] (0,0)--(2,0);

\end{pgfonlayer}
\end{tikzpicture}
```

可以加载 backgrounds 库。

```
\usetikzlibrary{backgrounds}
```

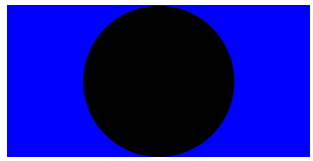
然后使用自动扩展的背景框和颜色，framed 和 show background rectangle，写哪一个都可以。

```
\begin{tikzpicture}[background
rectangle/.style={rounded
corners,fill=yellow},framed,show background
rectangle]
\fill (0,0) circle (1);
\end{tikzpicture}
```



以及直接使用定义好的 background layer

```
\begin{tikzpicture}
\begin{scope}[on background layer]
\fill[blue] (-2,-1) rectangle ++(4,2);
\end{scope}
\fill (0,0) circle (1);
\end{tikzpicture}
```



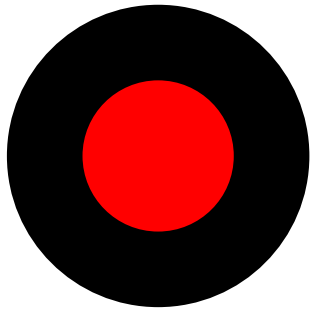
4.8 数学库 math

加载数学库 math。

```
\usetikzlibrary{math}
```

数学库 math 可以定义变量，给变量赋值，这样很多时候多次重复的长度、半径等画图参数可以通过修改变量赋值一次性修改。

```
\begin{tikzpicture}
\tikzmath{\x=1;}
\fill (0,0) circle (\x+1);
\fill [red] (0,0) circle (\x);
\end{tikzpicture}
```



数学库的功能还有很多，可以定义函数，有循环和判断语句，所以可以像其他语言那样编程画图，一些有规则的复杂图案可以用少量语句完成。

4.9 node

tikz 的 node 不仅仅是文本框，node 非常强大好用。

4.9.1 基本功能

最基本的用法是在某点放置文本框，坐标值默认情况是文本框中心点位置。默认是无边框白色填充。可以设置文本框的边框颜色、填充色、字体等。

```
\begin{tikzpicture}
\node at (0,0) {$x_1$};
\node at (2,0) [draw] {italic};
\node at (4,0) [fill=yellow!20] {italic};
\node at (6,0) [node font={\bf,\Large,\color{blue}}] {italic};
\node at (9,0) {$\displaystyle\int_0^a f(x)dx$};
\end{tikzpicture}
```

x_1 italic italic **italic** $\displaystyle\int_0^a f(x)dx$

node 文字换行需要设置 align 参数，align=left, right, center, flush center，flush left, flush right 然后才可以使用换行符。还可以定义文本框的文字宽度 text width，然后自动换行。有 text height 和 text depth，但是不推荐使用。扩展纵向距离可以使用 minimum height。

```
\begin{tikzpicture}
\node [fill=red!30,align=left] at (0,0){set text\\here};
\node [fill=blue!20, text width=4cm, align=center] at (4,0){The
width is 4cm, and the height is 1cm};
\node [fill=blue!20, text width=4cm, minimum height=2cm,
align=center] at (9,0){The width is 4cm, and the height is 1cm};
\end{tikzpicture}
```

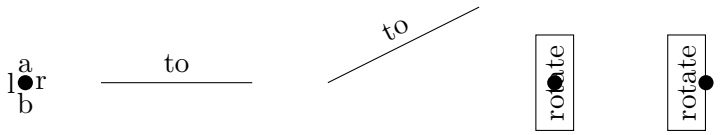
set text here

The width is 4cm, and the height is 1cm

The width is 4cm, and the height is 1cm

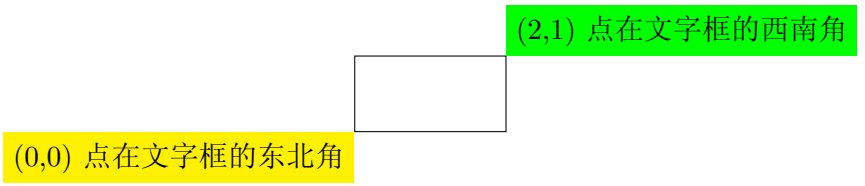
node 有位置参数，可以将文本框方式在上下左右等位置。也可以是 above left, above right, below left, below right，一共八个默认位置。还可以放置的划线的上下位置，设置文字自动随线倾斜。可以用 rotate 参数旋转文字，旋转是绕 node 的坐标点旋转。

```
\begin{tikzpicture}
\fill (0,0) circle (0.1);
\node at (0,0) [right] {r};
\node at (0,0) [left] {l};
\node at (0,0) [above] {a};
\node at (0,0) [below] {b};
\draw (1,0)--node [above] {to} ++(2,0);
\draw (4,0)--node [above,sloped] {to} ++(2,1);
\fill (7,0) circle (0.1);
\node at (7,0) [draw,rotate=90] {rotate};
\fill (9,0) circle (0.1);
\node at (9,0) [draw,rotate=90,above] {rotate};
\end{tikzpicture}
```



除了上下左右外，node 还定义了 anchor，anchor 跟上下左右是反的。东南西北方位表示坐标点在文本框的哪个角。

```
\begin{tikzpicture}
\draw (0,0) node [anchor=north east ,fill=yellow]
{(0,0)点在文字框的东北角} rectangle (2,1) node [anchor=south west,
fill=green] {(2,1)点在文字框的西南角};
\end{tikzpicture}
```



LaTeX 中图片是被视为文字的，所以可以加到 node 里。下面这个效果是因为天津大学校徽的 pdf 图片是透明的，如果不是透明的就无法填充黄色了。但是 clip 语句可以很方便的切边，从方图片制作圆头像。

```
\begin{tikzpicture}
\node at (0,0) {\includegraphics[width=2cm]{fig/tju.pdf}};
\fill [red] (0,0) circle (0.1);
```

```
\clip (3.2,0) circle (1);
\fill [yellow!60] (2,-1.2) rectangle ++(2.4,2.4);
\node at (3.2,0) {\includegraphics[width=2cm]{fig/tju.pdf}};
\end{tikzpicture}
```


有时希望将 node 放置在线的上下位置，但是不想在中间位置，还可以使用 pos。使用 pos 甚至可以在线的外面，可以是-0.2 或 1.2 位置。

```
\begin{tikzpicture}
\draw (0,0)-- node [above,pos=0.8] {0.8位置} (5,0);
\draw (6,0)-- node [above,pos=1.2] {1.2位置} ++(4,0);
\end{tikzpicture}
```

0.8 位置 1.2 位置

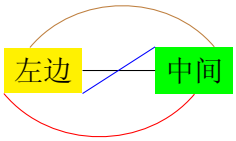
4.9.2 命名 node 并使用 node 名坐标

我们可以命名（标记）node，

```
\node (name) at (x,y) {text};
```

并且将命名视为坐标点使用。坐标点有三种情况，分别是 (name), (name.anchor) 和 (name.angle)，角度最好用，取值范围是 $[-360:360]$。

```
\begin{tikzpicture}
\node (L) [fill=yellow] at (-2,0){左边};
\node (M) [fill=green] at (0,0){中间};
\draw (L)--(M);
\draw [blue] (L.south east)--(M.north west);
\draw [red] (L.south west) to [bend right=50] (M.south);
\draw [brown] (L.120) to [bend left=50] (M.50);
\end{tikzpicture}
```



可以添加 positioning 库。

```
\usetikzlibrary{positioning}
```

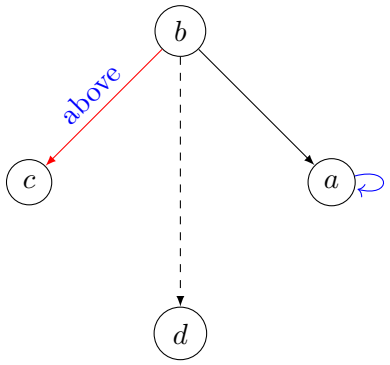
然后定义 node 放在另一个 node 的上下左右多少距离处。这个例子相当于将数据图加载后，在图上任意位置添加文字、箭头、数学公式等。对于发表文章时图内嵌入漂亮的公式或者其他特殊标记非常有用。

```
\begin{tikzpicture}
\node (tju) at (0,0) {\includegraphics[width=3cm]{fig/tju.pdf}};
\node [below=2mm of tju] {天津大学校徽};
\node [left=-2mm of tju] {\rotatebox{90}{天津大学校徽}};
\node at (tju.0) [below,rotate=90] {天津大学校徽};
\node at (tju.90) [shift={(0.8,0.3)}] {天津大学校徽};
\end{tikzpicture}
```

4.9.3 edge 操作

edge 操作一般结合 node 用，所以这里放在 node 坐标这一节了。但是 edge 本质上是一种 path 操作，这里的 node 只起到以 node 名为坐标的用处。edge 非常适合画复杂的关系图。

```
\begin{tikzpicture}
\node foreach \name/\angle in {a/0,b/90,c/180,d/270}
(\name) at (\angle:2) [circle,draw] {$\name$};

\draw [-latex] (b) edge (a)
edge [red] node [above,sloped,blue] {above}  (c)
edge [dashed] (d)
(a) edge [blue,loop right] (a);
\end{tikzpicture}
```



edge 还可以结合 quotes 库用，也很方便，注意第二个字符串后面还有一个单引号表示该字符串放在下面。单引号和字符串的双引号之间可以空格也可以不空格。字符串样式如前面 quotes 库的例子。如果需要对放在下面的字符串修改样式，需要写成下面例子中的样式。表示放在下面的单引号、表示颜色的 blue 和表示放置位置的 near end 都是字符串"end" 的参数，要一起放在花括号内。位置可以用 pos=1 这样的样式，也很方便。

```
\begin{tikzpicture}
\draw (0,0) edge ["above"{red},"below" ','start" near start,
"end"{',blue, near end}] (6,0);
\end{tikzpicture}
```

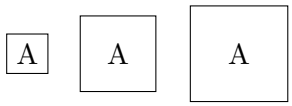
start above

below end

4.9.4 node 的形状和 shapes.geometric

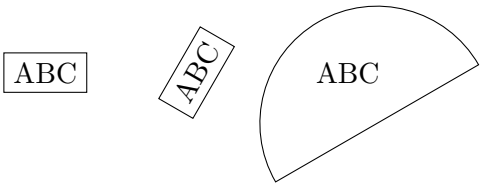
node 的通用参数为 text width, text height（不建议使用），minimum size，minimum width，minimum height，inner sep，inner xsep，inner ysep 等。

```
\begin{tikzpicture}
\node at (0,0) [draw] {A};
\node at (1.2,0) [minimum size=1cm,draw]
{A};
\node at (2.8,0) [inner sep=0.5cm,draw]
{A};
\end{tikzpicture}
```



node 可以旋转，一些形状可以只旋转边框，不旋转文字。

```
\begin{tikzpicture}
\node at (0,0) [draw] {ABC};
\node at (2,0) [draw,rotate=60] {ABC};
\node at (4,0) [draw,semicircle,shape border uses incircle,shape
border rotate=30] {ABC};
\end{tikzpicture}
```



node 默认边框形状是上面例子中的方框，内置了一些常用形状。不需要加载任何库就可以使用圆形。

```
\begin{tikzpicture}
\node at (0,0) [circle,fill=yellow] {B};
\end{tikzpicture}
```

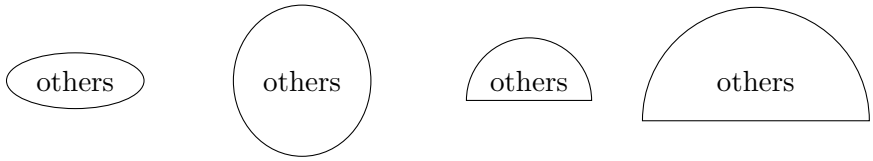
其他边框形状需要加载 shapes.geometric 库。

```
\usetikzlibrary{shapes.geometric}
```

然后可以使用椭圆 ellipse、半圆形 semicircle、菱形 diamond、梯形 trapezium、正多边形 regular polygon、星形 star、等边三角形 isosceles triangle、风筝 kite、飞镖 dart、扇形 circular sector、圆柱 cylinder。

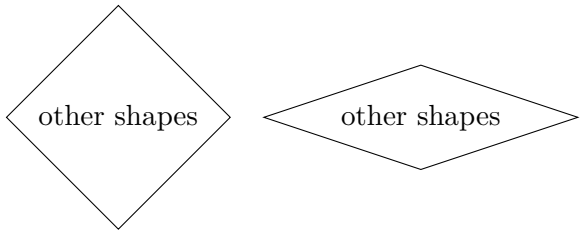
椭圆和半圆比较简单，没有特殊选项，可以用基础选项 minimum width 和 minimum height 改变纵横比。只设置一个有时候会等比例缩放。

```
\begin{tikzpicture}
\node at (0,0) [ellipse,draw] {others};
\node at (3,0) [ellipse,minimum height=2cm,draw] {others};
\node at (6,0) [semicircle,draw] {others};
\node at (9,0) [semicircle,minimum height=1.5cm,draw] {others};
\end{tikzpicture}
```



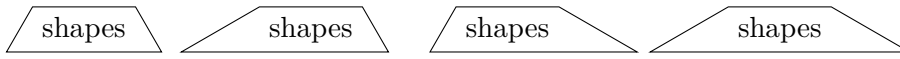
菱形 diamond 有 aspect 参数，用来调整纵横比。

```
\begin{tikzpicture}
\node at (0,0) [diamond,draw] {other shapes};
\node at (4,0) [diamond,aspect=3,draw] {other shapes};
\end{tikzpicture}
```



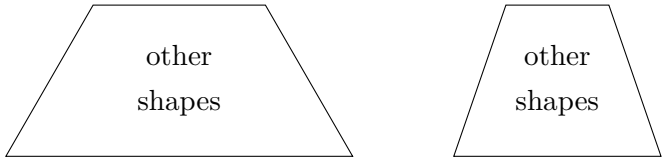
梯形参数比较多，可以设置左右角度和角度。

```
\begin{tikzpicture}
\node at (0,0) [trapezium,draw] {shapes};
\node at (3,0) [trapezium,trapezium left angle=30,draw] {shapes};
\node at (5.6,0) [trapezium,trapezium right angle=30,draw]
{shapes};
\node at (9.2,0) [trapezium, trapezium angle=30,draw] {shapes};
\end{tikzpicture}
```



梯形有 trapezium stretches=false/true 选项，false 时梯形不改变基础比例形状，相似地缩放。true 时则可以拉伸。

```
\begin{tikzpicture}
\node at (0,0) [trapezium,minimum height=2cm,draw,align=center]
{other\\shapes};
\node at (5,0) [trapezium,trapezium stretches=true,minimum
height=2cm,draw,align=center] {other\\shapes};
\end{tikzpicture}
```

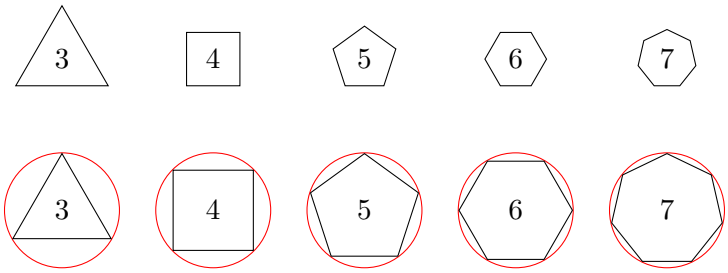


梯形还有 trapezium stretches body=false/true 选项，横向拉伸时保持角度不变。

```
\begin{tikzpicture}
\node at (0,0) [trapezium,trapezium stretches=true,minimum
width=2cm,draw,align=center] {A};
\node at (5,0) [trapezium,trapezium stretches body=true,minimum
width=2cm,draw,align=center] {B};
\end{tikzpicture}
```

正多边形 regular polygon 有一个参数，regular polygon sides= 整数。没有文字时，如果设置 minimum width 相等，外接圆半径是相等的。有文字时，如果文字所占空间比较大，会自动扩展边框，否则仍然是相等的。

```
\begin{tikzpicture}
\foreach \n in {3,...,7}
{
\node at (\n*2,0) [regular polygon, regular polygon
sides=\n,draw] {\n};
\draw [red] (\n*2,-2) circle (0.76);
\node at (\n*2,-2) [regular polygon, regular polygon
sides=\n,draw,minimum width=1.5cm] {\n};
}
\end{tikzpicture}
```

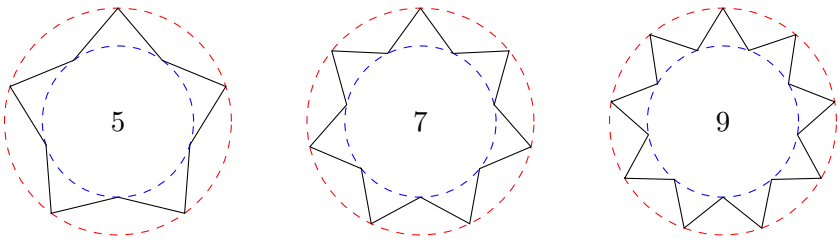


星形 star 的参数有 star points, star point height, star point ratio。

```
\begin{tikzpicture}
\foreach \n in {5,7,9}
\node at (\n,0) [star, star points=\n,draw] {\n};
\end{tikzpicture}
```

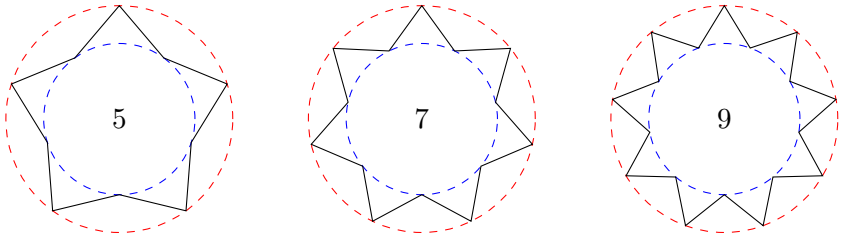
设置最小尺寸和 star point height，星形的外圆直径等于最小尺寸，内圆直径等于外圆半径减去 star point height。

```
\begin{tikzpicture}
\foreach \n in {5,7,9}
{
\draw [red, dashed] (\n*2,0) circle (1.5);
\draw [blue, dashed] (\n*2,0) circle (1);
\node at (\n*2,0) [star, star points=\n, star point height=0.5cm,
minimum size=3cm, draw] {\n};
}
\end{tikzpicture}
```



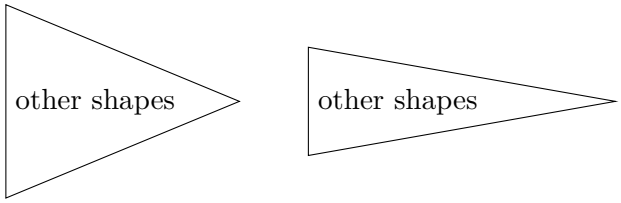
还可以设置 star point ratio，该比例即外圆半径比内圆半径，上面的例子该比例是 1.5。

```
\begin{tikzpicture}
\foreach \n in {5,7,9}
{
\draw [red, dashed] (\n*2,0) circle (1.5);
\draw [blue, dashed] (\n*2,0) circle (1);
\node at (\n*2,0) [star, star points=\n, star point ratio=1.5,
minimum size=3cm, draw] {\n};
}
\end{tikzpicture}
```



等边三角形的参数有顶角 isosceles triangle apex angle= 度数, 是否拉伸 isosceles triangle stretches=false/true,

```
\begin{tikzpicture}
\node at (0,0) [isosceles triangle,draw] {other shapes};
\node at (4,0) [isosceles triangle,isosceles triangle apex
angle=20,draw] {other shapes};
\end{tikzpicture}
```



风筝形 kite 参数有上顶角 kite upper vertex angle 和下顶角 kite lower vertex angle，顶角 kite vertex angles，即上下顶角相等情况，这时类似 diamond。

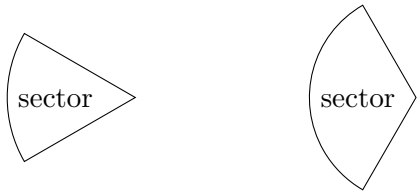
```
\begin{tikzpicture}
\node at (0,0) [kite,draw] {kite};
\node at (5,0) [kite,kite upper vertex angle=120, kite lower
vertex angle=30, draw] {kite};
\end{tikzpicture}
```

飞镖形 dart 有 dart tip angle 和 dart tail angle，

```
\begin{tikzpicture}
\node at (0,0) [dart,draw] {dart};
\node at (5,0) [dart,dart tip angle=30, dart tail angle=150,
draw] {dart};
\end{tikzpicture}
```

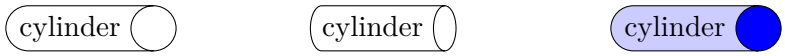
扇形 circular sector 参数有 circular sector angle

```
\begin{tikzpicture}
\node at (0,0) [circular sector,draw] {sector};
\node at (4,0) [circular sector, circular sector angle=120,draw]
{sector};
\end{tikzpicture}
```



圆柱形 cylinder 参数有纵横比 aspect，填充 cylinder uses custom fill=false/true, cylinder end fill 和 cylinder body fill。

```
\begin{tikzpicture}
\node at (0,0) [cylinder,draw] {cylinder};
\node at (4,0) [cylinder,aspect=0.5,draw] {cylinder};
\node at (8,0) [cylinder,cylinder uses custom fill=true, cylinder
end fill=blue, cylinder body fill=blue!20,draw] {cylinder};
\end{tikzpicture}
```



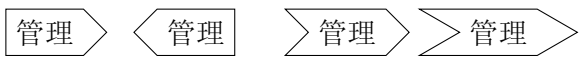
4.9.5 node 的其他库

node 还内置了许多常用符号，需要加载 shapes.symbols 库

```
\usetikzlibrary{shapes.symbols}
```

这里只介绍比较常用的 signal。

```
\begin{tikzpicture}
\node at (0,0) [signal,draw] {管理};
\node at (2,0) [signal,signal to=west, draw] {管理};
\node at (4,0) [signal,signal to=east,signal from=west, draw]
{管理};
\node at (6,0) [signal,signal to=east,signal from=west, signal
pointer angle=60, draw] {管理};
\end{tikzpicture}
```



node 的箭头库 shapes.arrows也比较实用。

```
\usetikzlibrary{shapes.arrows}
```

单箭头 single angle 和双箭头 double angle 用途比较广，参数差不多。

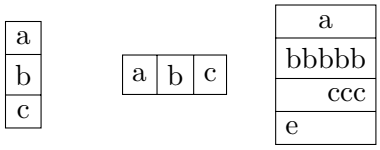
```
\begin{tikzpicture}
\node at (0,0) [single arrow, draw] {淬火};
\node at (2,0) [single arrow, minimum width=1.5cm, minimum
height=2cm,draw] {淬火};
\node at (4,0) [single arrow, single arrow head extend=0.5cm,
draw] {淬火};
\node at (6,0) [single arrow, single arrow head indent=0.1cm,
draw] {淬火};
\node at (8,0) [single arrow, single arrow tip angle=60, draw]
{淬火};
\end{tikzpicture}
```

文字分割库。

```
\usetikzlibrary{shapes.multipart}
```

这里介绍一下 rectangle split，这个有些时候非常有用。默认是纵向分割的。rectangle split parts=n 表示分为 n 个格子，后面的内容必须小于等于 n，多出来的不会显示。使用 rectangle split part align={对齐列表} 可以对每个格子设置不同的对齐方式。这个库表现计算机原理时挺有用。

```
\begin{tikzpicture}
\node at (0,0) [rectangle split, rectangle split parts=3, draw]
{a\nodepart{two}b\nodepart{three}c};
\node at (2,0) [rectangle split, rectangle split parts=3,
rectangle split horizontal, draw]
{a\nodepart{two}b\nodepart{three}c};
\node at (4,0) [rectangle split, rectangle split parts=4,
rectangle split part align={center,left,right,left}, draw]
{a\nodepart{two}bbbbb\nodepart{three}ccc\nodepart{four}e};
\end{tikzpicture}
```



callout 库对理工科的用处不大，需要时参看手册。

```
\usetikzlibrary{shapes.callouts}
```

misc 库做 ppt 时可能有些用处。

```
\usetikzlibrary{shapes.misc}
```

4.10 定义样式

4.10.1 修改默认样式 style

tikz 的设置语句为\tikzset，设置样式的语句如下

```
\tikzset{样式名/.style={样式}}
```

默认样式名为 every picture

```
\tikzset{every picture/.style={line width=5pt}}
```

默认 node 样式名为 every node，例如将 node 都设置为可换行居中对齐，字号为 small。

```
\tikzset{
every node/.style={
align=center,
font={\small},
}}
```

4.10.2 自定义样式

例如设置图形的默认颜色和线宽，在 tikzpicture 环境内设置 style，当前环境有效，否则对下面的都生效。定义好的样式可以复用。

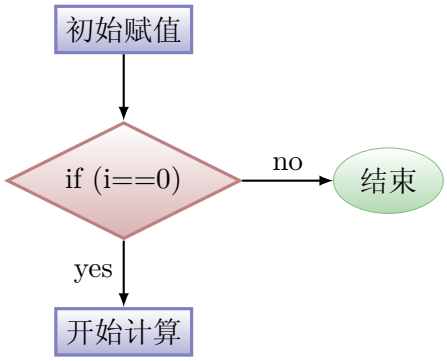
```
\definecolor{tjublue}{RGB}{0,70,140}
\tikzset{tju/.style={color=tjublue,line
width=2pt}}

\begin{tikzpicture}[tju]
\fill (0,0) circle (0.5);
\end{tikzpicture}
\begin{tikzpicture}[tju]
\fill (0,0) circle (0.5);
\end{tikzpicture}
```

4.10.3 自定义带参数的样式

下面的例子自定义带参数的 node 样式，使用时直接调用自定名 fang 即可。虽然自定义的 node 形状是方框，但是可以覆盖为其他形状。node 的各种参数仍然可以使用和覆盖。

```
\tikzset{
fang/.style={
rectangle,
minimum size=6mm,
very thick,
draw=#1!50!black!50,
top color=white,bottom color=#1!50!black!30,
```

```
font=\it,
}}

\begin{tikzpicture}[thick]
\node (a) at (0,0) [fang=blue] {初始赋值};
\node (b) at (0,-2) [fang=red,diamond,aspect=2,font=\rm] {if
(i==0)};
\node (c) at (3.5,-2) [fang=green,ellipse,very thin] {结束};
\node (d) at (0,-4) [fang=blue] {开始计算};
\draw [-latex] (a)--(b);
\draw [-latex] (b)-- node [above] {no} (c);
\draw [-latex] (b)-- node [left] {yes} (d);
\end{tikzpicture}
```



上面的例子还可以带多个参数，需要指定几个参数，调用的时候多个花括号并列方式指定参数。不过上面手册的这个配色很漂亮，一般不需要两个颜色参数来配色。

```
\tikzset{
fang/.style 2 args={
rectangle,
minimum size=6mm,
very thick,
draw=#1!50!black!50,
top color=white,bottom color=#2!50!black!30,
font=\it,
}}
\begin{tikzpicture}
\node (a) at (0,0) [fang={red}{blue}] {初始赋值};
\end{tikzpicture}
```

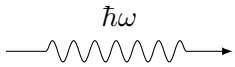
4.10.4 修饰库

修饰库很丰富。

```
\usetikzlibrary{decorations.shapes,decorations.pathmorphing,
decorations.markings}
```

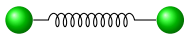
下面举几个理工科常用的例子。第一个例子是光子（声子），snake 只需要 decorations.pathmorphing 库。

```
\begin{tikzpicture}
\draw [decorate,decoration={snake,pre length=15pt, post
length=15pt, segment length=8pt, amplitude=4pt}] [-latex] (0,0)--
node [above=2mm] {$\hbar\omega$} (3,0);
\end{tikzpicture}
```



第二个例子是弹簧

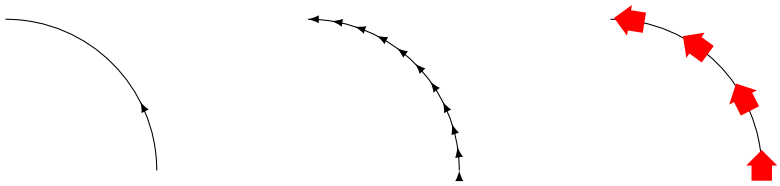
```
\begin{tikzpicture}
\node (a) [circle,ball color=green] at (0,0) {};
\node (b) [circle,ball color=green] at (2,0) {};
\draw [decorate,decoration={coil, pre length=2mm, post
length=2mm, segment length=1mm, aspect=0.5}] (a)--(b);
\end{tikzpicture}
```



第三个例子是沿线画箭头，需要 decorations.markings

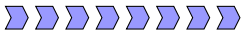
```
\begin{tikzpicture}
\draw (0,0) arc [start angle=0, end angle=90, radius=2];
\draw [decorate,decoration={markings,mark=at position 0.3 with
{\arrow{latex}}}] (0,0) arc [start angle=0, end angle=90,
radius=2];
\draw (4,0) arc [start angle=0, end angle=90, radius=2];
\draw [decorate,decoration={markings,mark=between positions 0.0
and 1.0 step 3.14mm with {\arrow{latex}}}] (4,0) arc [start
angle=0, end angle=90, radius=2];
\draw (8,0) arc [start angle=0, end angle=90, radius=2];
\draw [decorate,decoration={markings,mark=between positions 0.0
```

```
and 1.0 step 9.42mm with {\node [single arrow,fill=red, single
arrow head extend=2pt, transform shape]{};}}] (8,0) arc [start
angle=0, end angle=90, radius=2];
\end{tikzpicture}
```



加载 decorations.shapes 库可以添加其他符号，注意 signal 的参数在 decoration={} 之外。

```
\begin{tikzpicture}
\draw [decorate,draw,fill=blue!40,decoration={shape backgrounds,
shape size=3mm, shape=signal, shape sep=4mm}, signal from=west,
signal to=east] (0,0)--(3,0);
\end{tikzpicture}
```



另外，文字库 decorations.text 和分形库 decorations.fractals 可以自行查阅手册练习。

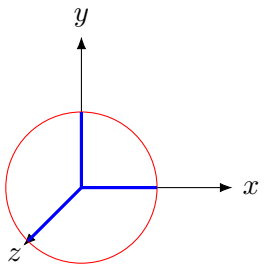
4.11 三维坐标

4.11.1 默认三维坐标

tikz 自带三维坐标，其默认效果如下图的坐标系所示，也可以自设置坐标方向，需要设置三维单位矢量在二维平面的单位矢量，下图蓝色线为默认三维坐标的自定义单位矢量结果。

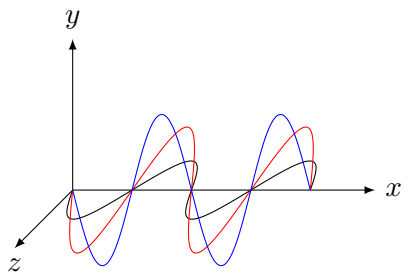
```
\begin{tikzpicture}[>=Latex]
\draw [->] (0,0,0)--(2,0,0) node [right] {$x$};
\draw [->] (0,0,0)--(0,2,0) node [above] {$y$};
\draw [->] (0,0,0)--(0,0,2) node [pos=1.15] {$z$};
\draw [red] (0,0) circle (1);
\begin{scope}[x={(1cm,0cm)},y={(0cm,1cm)},z={(-135:1cm)}]
\draw [very thick,blue] (0,0,0)--(1,0,0);
\draw [very thick,blue] (0,0,0)--(0,1,0);
\draw [very thick,blue] (0,0,0)--(0,0,1);
```

```
\end{scope}
\end{tikzpicture}
```



内置的三维坐标可以绕轴旋转。下面的例子中黑色线在 xz 平面，红色线是绕 x 轴旋转 30 度的结果，蓝色线是绕 x 轴旋转 90 度。

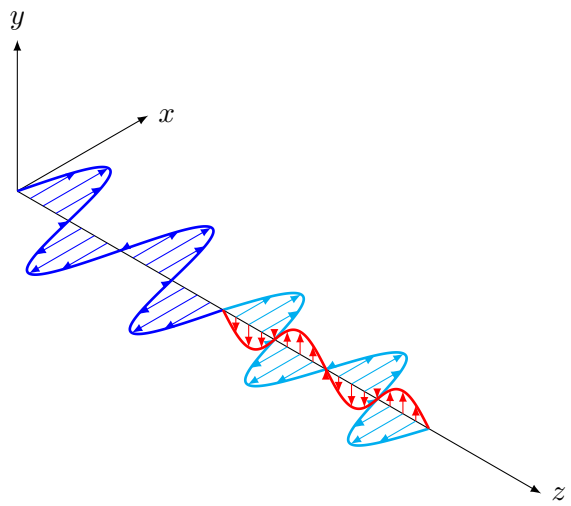
```
\begin{tikzpicture}
\draw [-latex] (0,0,0)--(4,0,0) node [right] {$x$};
\draw [-latex] (0,0,0)--(0,2,0) node [above] {$y$};
\draw [-latex] (0,0,0)--(0,0,2) node [below] {$z$};
\draw [domain=0:pi,smooth,samples=100,variable=\x] plot
(\x,0,{sin(4*\x r)});
\draw [red] [domain=0:pi,smooth,samples=100,variable=\x,rotate
around x=30] plot (\x,0,{sin(4*\x r)});
\draw [blue] [domain=0:pi,smooth,samples=100,variable=\x,rotate
around x=90] plot (\x,0,{sin(4*\x r)});
\end{tikzpicture}
```



可以自定义 x, y, z 轴的单位矢量方向，下面是个电磁波的例子。

```
\begin{tikzpicture}[x={(30:1cm)},y={(0cm,1cm)},
z={(-30:1cm)},>=latex]
\draw [->] (0,0,0)--(2,0,0) node [right] {$x$};
\draw [->] (0,0,0)--(0,2,0) node [above] {$y$};
\draw [->] (0,0,0)--(0,0,8) node [right] {$z$};
```

```
%x 偏振
\draw [line width=1pt, color=blue]
[domain=0:pi,smooth,samples=100,variable=\z] plot({sin(4*\z
r)},0,\z);
\foreach \z in {0.19625,0.3925,...,3}
\draw [->,blue] (0,0,\z)--({sin(4*\z r)},0,\z);
%光参量过程
\draw [line width=1pt, color=cyan]
[domain=pi:2*pi,smooth,samples=100,variable=\z]
plot({0.8*sin(4*\z r)},0,\z);
\foreach \z in {3.33625,3.5325,...,6.2}
\draw [->,cyan] (0,0,\z)--({0.8*sin(4*\z r)},0,\z);

\draw [line width=1pt, color=red]
[domain=pi:2*pi,smooth,samples=100,variable=\z]
plot(0,{0.3*sin((4*\z+pi) r)},\z);
\foreach \z in {3.33625,3.5325,...,6.2}
\draw [->,red] (0,0,\z)--(0,{0.3*sin((4*\z+pi) r)},\z);
\end{tikzpicture}
```



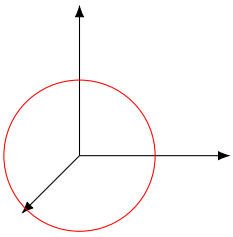
4.11.2 3d 库

3d 库是基于默认三维坐标（伪投影）的结果，3d 库提供了一些三维画图样式。

```
\usetikzlibrary{3d}
```

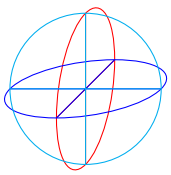
下面的例子与默认三维坐标的第一个例子是相同的。

```
\begin{tikzpicture}[->,>=Latex]
\draw (0,0,0) -- (xyz spherical cs:radius=2);
\draw (0,0,0) -- (xyz spherical cs:radius=2,latitude=90);
\draw (0,0,0) -- (xyz spherical cs:radius=2,longitude=90);
\draw [red] (0,0) circle (1);
\end{tikzpicture}
```



3d 库定义了 canvas 和 plane，然后在该 plane 画图得到的是投影效果。比如下图，画圆的语句会得到椭圆。

```
\begin{tikzpicture}
\begin{scope}[canvas is zy plane at x=0,red]
\draw (0,0) circle (1cm);
\draw (-1,0) -- (1,0) (0,-1) -- (0,1);
\end{scope}
\begin{scope}[canvas is zx plane at y=0,blue]
\draw (0,0) circle (1cm);
\draw (-1,0) -- (1,0) (0,-1) -- (0,1);
\end{scope}
\begin{scope}[canvas is xy plane at z=0,cyan]
\draw (0,0) circle (1cm);
\draw (-1,0) -- (1,0) (0,-1) -- (0,1);
\end{scope}
\end{tikzpicture}
```



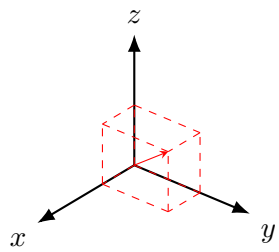
4.11.3 tikz-3dplot 包

tikz-3dplot是一个包，不是 tikz 的库，需要使用加载包的语句。

```
\usepackage{tikz-3dplot}
```

这个包按坐标系旋转结果得到三维坐标系，一开始需要设置旋转参数。设置旋转参数的语句要放在 tikzpicture 环境外。然后在 tikzpicture 参数里使用旋转后的坐标系为主坐标系。这个包设置了三维点的投影点，不需要自己再算各个方向的投影坐标了。画晶体结构的时候会很方便。

```
\tdplotsetmaincoords{60}{130}
\begin{tikzpicture}[scale=2,tdplot_main_coords,>=Latex]
\coordinate (O) at (0,0,0);
\tdplotsetcoord{P}{.8}{55}{60}
\draw[thick,->] (0,0,0) -- (1,0,0) node[anchor=north east]{$x$};
\draw[thick,->] (0,0,0) -- (0,1,0) node[anchor=north west]{$y$};
\draw[thick,->] (0,0,0) -- (0,0,1) node[anchor=south]{$z$};
\draw[-stealth,color=red] (O) -- (P);
\draw[dashed, color=red] (O) -- (Px);
\draw[dashed, color=red] (O) -- (Py);
\draw[dashed, color=red] (O) -- (Pz);
\draw[dashed, color=red] (Px) -- (Pxy);
\draw[dashed, color=red] (Py) -- (Pxy);
\draw[dashed, color=red] (Px) -- (Pxz);
\draw[dashed, color=red] (Pz) -- (Pxz);
\draw[dashed, color=red] (Py) -- (Pyz);
\draw[dashed, color=red] (Pz) -- (Pyz);
\draw[dashed, color=red] (Pxy) -- (P);
\draw[dashed, color=red] (Pxz) -- (P);
\draw[dashed, color=red] (Pyz) -- (P);
\end{tikzpicture}
```



这个包的手册到 2012，后面就没有再更新了，但是有些功能还是不错的。

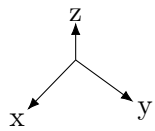
4.11.4 perspective

perspective 是 tikz 的库，加载时语句为

```
\usetikzlibrary{perspective}
```

这个库也是按旋转角度设置三维坐标系，但是角度参数顺序与 tikz-3dplot 包相反。

```
\begin{tikzpicture}[>=Latex,3d view={130}{60}]
\draw[->] (0,0,0) -- (1,0,0) node[pos=1.2]{x};
\draw[->] (0,0,0) -- (0,1,0) node[pos=1.2]{y};
\draw[->] (0,0,0) -- (0,0,1) node[pos=1.2]{z};
\end{tikzpicture}
```



画出来的效果是方向一致，但是 z 轴尺度不同，比 tikz-3dplot 要小。因为这个库是透视效果库，手册给出了透视图的例子，具体可看手册。

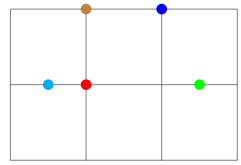
4.12 坐标计算 calc 库

坐标计算，不是坐标值计算，坐标值计算是数学函数计算。加载 calc 库的语句为

```
\usetikzlibrary{calc}
```

对坐标进行计算时，需要加行间公式符号 $。基本操作是对整个坐标的 (x, y) 进行加减乘法。

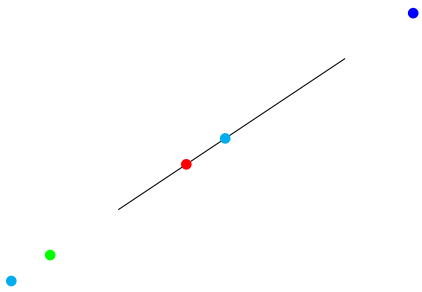
```
\begin{tikzpicture}
\draw [help lines] (0,0) grid (3,2);
\fill [red] (1,1) circle (2pt);
\fill [blue] ($2*(1,1)$) circle (2pt);

\coordinate (a) at (0.5,1);
\fill [cyan] (a) circle (2pt);
\fill [brown] ($2*(a)$) circle (2pt);
\fill [green] ($(a)+(2,0)$) circle (2pt);
\end{tikzpicture}
```



使用 $(a 点坐标)! 数值!(b 点坐标)$ 可以实现沿 ab 连线移动的坐标点。该坐标点距离 a 点长度由数值给定。数值可以是比例，表示坐标点距离 a

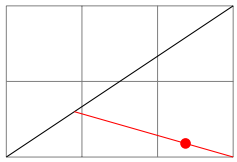
点的长度为 ab 长度乘以比例，也可以是距离 a 点坐标的长度。比例和长度都可以是负值，表示逆向延展。

```
\begin{tikzpicture}
\draw (0,0)--(3,2);
\fill [red] ($(0,0)!0.3!(3,2)$) circle (2pt);
\fill [blue] ($(0,0)!1.3!(3,2)$) circle (2pt);
\fill [green] ($(0,0)!-0.3!(3,2)$) circle (2pt);
\fill [cyan] ($(0,0)!1.7cm!(3,2)$) circle (2pt);
\fill [cyan] ($(0,0)!-1.7cm!(3,2)$) circle (2pt);
\end{tikzpicture}
```



上面的语句可以连用，按从左到有顺序进行，红色点在黑色直线 0.3 长度处的点连线 (3,0) 坐标长度距离黑色线与红色线交点为 0.7 处。

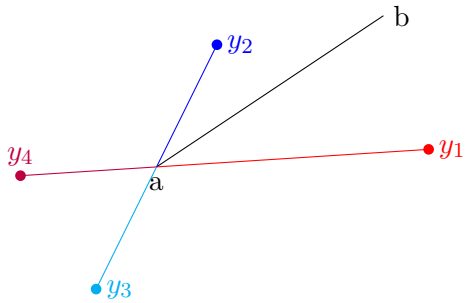
```
\begin{tikzpicture}
\draw [help lines] (0,0) grid (3,2);
\draw (0,0) -- (3,2);
\draw[red] ($(0,0)!.3!(3,2)$) -- (3,0);
\fill[red] ($(0,0)!.3!(3,2)!.7!(3,0)$)
circle (2pt);
\end{tikzpicture}
```



使用 $(a 点坐标)! 数值! 角度:(b 点坐标)$ 语句，ab 绕 a 点转动后，再按数值缩放与 a 点的距离，获得最后的坐标。角度为负是逆时针，角度为正是顺治针。从例子中可以很容易看出，y2 点和 y3 点在一条直线上。

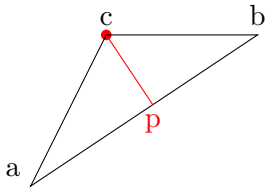
```
\begin{tikzpicture}
\coordinate (a) at (0,0);
\coordinate (b) at (3,2);
\draw (a) node [below] {a}--(b) node [right] {b};
\coordinate (y1) at ($(a)!1.0!-30:(b)$);
\fill [red] (y1) circle (2pt);
```

```
\draw [red] (a)--(y1) node [right] {$y_1$};
\coordinate (y2) at ($(a)!0.5!30:(b)$);
\fill [blue] (y2) circle (2pt);
\draw [blue] (a)--(y2) node [right] {$y_2$};
\coordinate (y3) at ($(a)!-0.5!30:(b)$);
\fill [cyan] (y3) circle (2pt);
\draw [cyan] (a)--(y3) node [right] {$y_3$};
\coordinate (y4) at ($(a)!-0.5!-30:(b)$);
\fill [purple] (y4) circle (2pt);
\draw [purple] (a)--(y4) node [above] {$y_4$};
\end{tikzpicture}
```



语句 $(a 点坐标)!(c 点坐标)!(b 点坐标)$，得到从 c 点做 ab 连线的垂直投影。

```
\begin{tikzpicture}
\coordinate (a) at (0,0);
\coordinate (b) at (3,2);
\coordinate (c) at (1,2);
\fill [red] (c) circle (2pt);
\draw (a) node [above left ] {a}--(b) node
[above] {b}--(c) node [above] {c}--cycle;
\draw [red] (c)--($(a)!(c)!(b)$) node
[below] {p};
\end{tikzpicture}
```



语句 $(a 点坐标)!(c 点坐标)! 角度:(b 点坐标)$，得到从 c 点做 ab 连线的投影 p 点，cp 与 ab 的角度为给定角度。但是带角度时例子测试很不成功，不知道长度按什么规则计算的。

4.13 几何做图

结合 calc 库，tikz 的几何做图功能很不错。

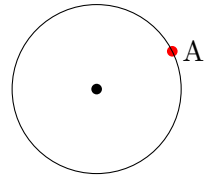
4.13.1 through

加载 through 库。

```
\usetikzlibrary{through}
```

through 表示通过某点，最简答的用法必须在 node 的选项里使用，且只能画圆。

```
\begin{tikzpicture}
\coordinate [label=right:{A}] (a) at (2,1.5);
\fill [red] (a) circle (2pt);
\fill (1,1) circle (2pt);
\node at (1,1) [draw,circle through={(a)}] {};
\end{tikzpicture}
```



4.13.2 相交 intersections

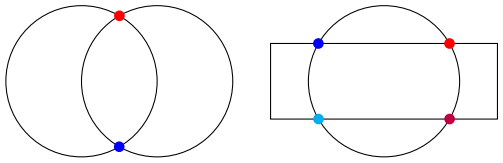
加载 intersections 库。

```
\usetikzlibrary{intersections}
```

相交库可以得到任意曲线的相交点。比如两个圆有两个相交点，而一个矩形可能与圆有四个交点。

```
\begin{tikzpicture}
\path [draw, name path=a] (0,0) circle (1);
\path [draw, name path=b] (1,0) circle (1);
\path [name intersections={of=a and b}];
\coordinate (c1) at (intersection-1);
\coordinate (c2) at (intersection-2);
\fill [red] (c1) circle (2pt);
\fill [blue] (c2) circle (2pt);

\begin{scope}[xshift=4cm]
\path [draw, name path=a] (0,0) circle (1);
\path [draw, name path=b] (-1.5,-0.5) rectangle ++(3,1);
\path [name intersections={of=a and b}];
```

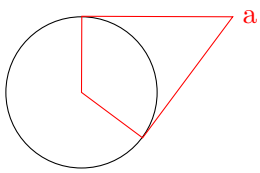
```
\coordinate (c1) at (intersection-1);
\coordinate (c2) at (intersection-2);
\coordinate (c3) at (intersection-3);
\coordinate (c4) at (intersection-4);
\fill [red] (c1) circle (2pt);
\fill [blue] (c2) circle (2pt);
\fill [cyan] (c3) circle (2pt);
\fill [purple] (c4) circle (2pt);
\end{scope}
\end{tikzpicture}
```



4.13.3 相切

tikz 的相切不是库，而是做成了坐标系。tikz 的坐标系有我们前面常用的直角坐标系、极坐标系、三维坐标系、node 坐标系，还有不怎么常用的重心坐标系和相切坐标系。相切坐标系必须加载 calc 库后方能使用。手册上的例子是画一个 node 的圆。相切点也可以是多个，用 solution=n 表示第 n 个相切点。

```
\begin{tikzpicture}
\coordinate (a) at (3,2);
\node [circle,draw] (c) at (1,1) [minimum size=2cm] {};
\draw [red] (a) node [right] {a}--(tangent
cs:node=c,point={(a)},solution=1)--(c.center)--(tangent
cs:node=c,point={(a)},solution=2)--cycle;
\end{tikzpicture}
```



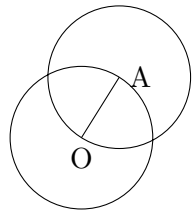
但是实际上我们可以将 node 画圆语句改为画圆语句，结果一样。这是因为 tikz 允许我们标记图形，下面的语句将 (c) 从标记 node 修改为标记圆，结果是一样的。这样就很方便我们做几何图形了。

```
\draw (c) (1,1) circle (1);
```

4.13.4 calc 库和几何做图

通过两点画双圆，veclen$(x,y)=\sqrt{x^2+y^2}$。

```
\begin{tikzpicture}
\coordinate [label=below:{O}] (o) at (0,0);
\coordinate [label=right:{A}] (a) at (0.5,0.8);
\draw (o)--(a);
\draw let
\p1=($(a)-(o)$),
\n1={veclen(\x1,\y1)}
in
(o) circle (\n1)
(a) circle (\n1);
\end{tikzpicture}
```



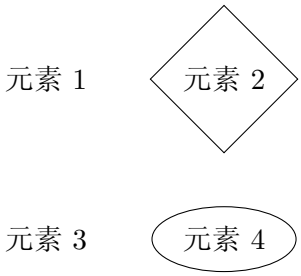
4.14 matrix 库

加载矩阵库 matrix

```
\usetikzlibrary{matrix}
```

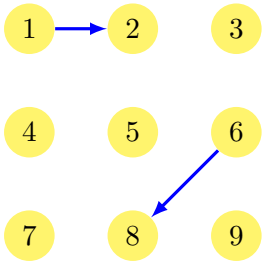
矩阵库是 node 作为矩阵元素

```
\begin{tikzpicture}
\matrix[column sep=7mm, row sep=7mm]
{
\node (a11) {元素1}; &\node [diamond,draw] (a12) {元素2};\\
\node (a21) {元素3}; &\node [ellipse,draw] (a22) {元素4};\\
};  %注意这个分号
\end{tikzpicture}
```



可以定义矩阵名，然后按矩阵元素调用。

```
\begin{tikzpicture}
\matrix (a) [matrix of nodes,
nodes={circle,fill=yellow!70,minimum size=2pt}, column
sep=7mm,row sep=7mm]
{
1 &2 &3\\
4 &5 &6\\
7 &8 &9\\
};  %注意这个分号
\draw [-latex,very thick,blue] (a-1-1)--(a-1-2);
\draw [-latex,very thick,blue] (a-2-3)--(a-3-2);
\end{tikzpicture}
```



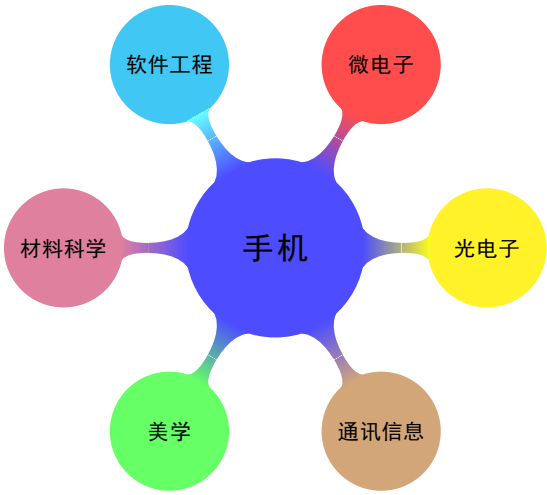
4.15 思维导图

加载思维导图库 mindmap，

```
\usetikzlibrary{mindmap}
```

并不喜欢这样的思维导图，觉得太占地方了，且形式比较单一。

```
\begin{tikzpicture}[small mindmap,concept color=blue!70,every
node/.style={node font={\normalsize,\bf}}]
\node [concept] {\large 手机}
child [grow=0, concept color=yellow!90] {node[concept]{光电子}}
child [grow=60, concept color=red!70] {node[concept]{微电子}}
child [grow=120, concept color=cyan!60] {node[concept]{软件工程}}
child [grow=180, concept color=purple!50] {node[concept]{材料科学}}
child [grow=240, concept color=green!60] {node[concept]{美学}}
child [grow=300, concept color=brown!70] {node[concept]{通讯信息}};
\end{tikzpicture}
```



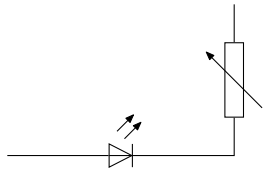
4.16 电路

tikz 的电路需要加载的包比较多，电路必须指定标准，加载的时候以指定标准的方式加载。下面加载时指定 IEC 标准。。

```
\usetikzlibrary{circuits.ee,circuits.ee.IEC,circuits.logic.IEC}
```

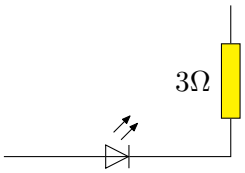
仅仅加载包是不行的，还必须在 tikzpicture 环境的参数里指定标准。电子元件用方括号表示，默认位置在前后两个坐标的中间位置处。

```
\begin{tikzpicture}[circuit ee IEC]
\draw (0,0) to [diode={light emitting}]
(3,0) to [resistor={adjustable}] (3,2);
\end{tikzpicture}
```



电阻可以添加阻值，也可以涂色

```
\begin{tikzpicture}[circuit ee IEC]
\draw (0,0) to [diode={light emitting}] (3,0) to
[resistor={info={$3\Omega$},fill=yellow}] (3,2);
\end{tikzpicture}
```



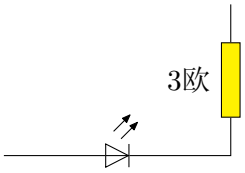
如果使用了 xeCJK 包，需要写成上面的形式，但是如果不使用 xeCJK 包，无论是 pdflatex（CJKutf8）还是 xelatex（全英文）编译情况，都可以使用下面的默认设置。

```
\draw (0,0) to [diode={light emitting}] (3,0) to [resistor={ohm=3 ,
fill=yellow}] (3,2);
```

原因可能是 xeCJK 无法匹配 amsmath 的`$\mathrm{\Omega}$`，没有显示。如果写成 info 样式才可以正确显示。

可以使用 circuit declare unit= 语句重置默认单位 ohm，

```
\begin{tikzpicture}[circuit ee IEC,circuit declare
unit={ohm}{\text{欧}}]
\draw (0,0) to [diode={light emitting}] (3,0) to [resistor={ohm=3
,fill=yellow}] (3,2);
\end{tikzpicture}
```



但是不能重置为`$\Omega$`，因为设置语句本身是`$\mathrm{<value>\Omega}$`，xeCJK 无法显示`$\mathrm{\Omega}$`。修改为中文单位要设置为 欧，取代结果是`$\mathrm{<value>\text{欧}}$`。看上去很麻烦，但是可以通过这种方法将所有单位都改为中文。

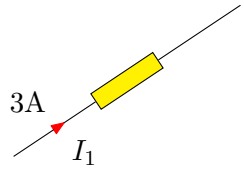
如果使用 info 方式，也可以写为中文的欧姆，但是不如上面的结果紧凑，因为 xeCJK 包中英文混排时会自动扩展英文到两边中文的间距。更细致的调节比较罗嗦，不如上面的设置语句简单方便。

```
\begin{tikzpicture}[circuit ee IEC]
```

```
\draw (0,0) to [diode={light emitting}] (3,0) to [resistor={info={3
欧},fill=yellow}] (3,2);
\end{tikzpicture}
```

使用 info 方式标记器件参数是很方便灵活的。下面是用 info 做电流标记的例子，`info'`表示标记在下方，`current direction'`则是电流换个方向。电流箭头的位置使用 pos= 方式最方便，可以自动倾斜表示电流方向的三角的角度跟随电线方向。pos=0.2 是电阻前后两个坐标连线的 0.2 处，不止起点到电阻的 0.2 处。

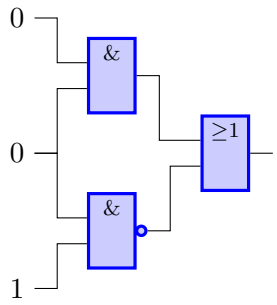
```
\begin{tikzpicture}[circuit ee IEC]
\draw (0,0) to [current direction={red,pos=0.2, info={3A},
info'={$I_1$}}, resistor={fill=yellow}] (3,2);
\end{tikzpicture}
```



指定逻辑电路标准，然后可以使用内置的逻辑电路符号。

```
\begin{tikzpicture}[circuit logic IEC,every circuit
symbol/.style={logic gate IEC symbol
color=black,fill=blue!20,draw=blue,very thick}]
\matrix[column sep=7mm]
{
\node (i0) {0}; & &\\
&\node [and gate] (a1) {}; &\\
\node (i1) {0}; & &\node [or gate] (o) {};\\
&\node [nand gate] (a2) {}; &\\
\node (i2) {1}; & &\\
};
\draw (i0.east)-- ++(right:3mm) |- (a1.input 1);
\draw (i1.east)-- ++(right:3mm) |- (a1.input 2);
\draw (i1.east)-- ++(right:3mm) |- (a2.input 1);
\draw (i2.east)-- ++(right:3mm) |- (a2.input 2);
\draw (a1.output)-- ++(right:3mm) |- (o.input 1);
\draw (a2.output)-- ++(right:3mm) |- (o.input 2);
\draw (o.output)-- ++(right:3mm);
```

```
\end{tikzpicture}
```



更多逻辑电路符号可以加载包

```
\usetikzlibrary{shapes.gates.logic}
\usetikzlibrary{shapes.gates.logic.US} %美标
\usetikzlibrary{shapes.gates.logic.IEC} %IEC 标准
```

除此之外，推荐一个更新活跃的包 circuitikz

```
\usepackage{circuitikz}
```

电子领域，还有一个继电器符号包 tikz-relay

```
\usepackage{tikz-relay}
```

展示时序的包 tikz-timing 很漂亮，非常推荐。

```
\usepackage{tikz-timing}
```

交换构架 switching architectures 包 sa-tikz手册更新到 2014 年，如果有需要也可以练习一下。

chart 库，加载时是 tikz 的 circuits.plc.sfc 库，手册是 tikz-sfc。

```
\usetikzlibrary{circuits.plc.sfc}
\texdoc tikz-sfc
```

tikz-ladder 是画 ladder diagram 的库。

```
\usetikzlibrary{circuits.plc.ladder}
```

pinoutikz 是标识芯片管脚的包

```
\usepackage{pinoutikz}
```

4.17 pgfplots

虽然 tikz 手册里有专门的 data visualization 部分，但是目前 pgfplots 包成长得相当不错，小一点的数据还是比较推荐使用的。如果数据文件太大，更推荐基于 c++ 的 gnuplot 软件。

加载 pgfplots 包即可，不用再加载 tikz 包。

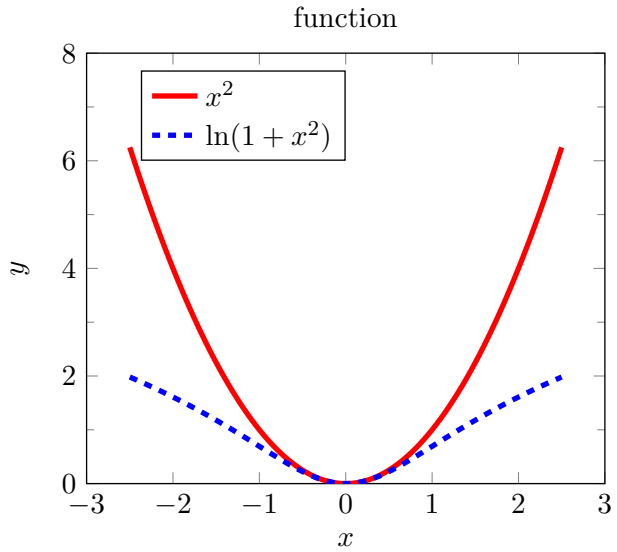
```
\usepackage{pgfplots}
```

4.17.1 二维函数画图

设置有点繁琐，简单的函数画图还是很漂亮的，而且坐标轴标识和图例的公式都可以直接得到 LaTeX 公式的美丽输出。图例位置默认值是东北角，四个角可以简单的 legend pos= 来设置，图中的例子是随意设置图例位置的语句。因为是 tikzpicture 环境内，所以图上在添加指示箭头、公式或者其他图形都非常方便。

```
\pgfplotsset{
every axis legend/.append style={
at={(0.5,0.96)},
anchor=north east,
legend cell align=left,
},
}
\begin{tikzpicture}
\begin{axis}[
title=function,
xmin=-3,xmax=3,
xlabel={$x$},
ylabel={$y$},
ylabel style={yshift=-10pt},
ymin=0, ymax=8,
minor y tick num=1,
legend entries={$x^2$,$\ln(1+x^2)$},
line width=0.5pt,
]
\addplot [
red,
domain=-2.5:2.5,
```

```
samples=51,
smooth,
line width=2pt,
]
{x*x};

\addplot [
blue,
dashed,
domain=-2.5:2.5,
samples=51,
smooth,
line width=2pt,
]
{ln(1+x*x)};
\end{axis}
\end{tikzpicture}
```



4.17.2 二维数据画图

数据画图设置与上面的例子相同，需要修改的仅仅是画图语句。数据文件的分隔符需要特别指明，下面的例子是逗号分隔情况。

```
\addplot [画图参数] table [col sep=comma] {data.txt};
```

多列数据时，有两种方式选取列。一个是 x= 列名，y= 列名，数据文件的列名分隔符必须与数据文件的分隔符一致。还有一个是 x index= 列序号，y index= 列序号，列序号从 0 开始，遵循了 C 语言的风格，gnuplot 的列序号是从 1 开始的。这个例子需要单独的数据文件才能实现，就不给代码了，请自行练习。

4.17.3 三维函数画图

从函数画三维图很漂亮，

```
\begin{tikzpicture}
\begin{axis}
\addplot3[
surf,
domain=-2:2,
domain y=-2:2,
samples=50
]
{exp(-x^2-y^2)};
\end{axis}
\end{tikzpicture}
```

感觉 pgfplots 配色比 gnuplot 好看，但是画图比较慢。

4.17.4 三维 map 图

这个例子仍然是从函数画图。

```
\begin{tikzpicture}
\begin{axis}[view={0}{90},colorbar]
\addplot3[
surf,shader=interp,  %必须写在同一行
domain=-2:2,
domain y=-2:2,
samples=50
]
{exp(-x^2-y^2)*x};
\end{axis}
\end{tikzpicture}
```

4.17.5 从数据画三维图

从数据文件画三维图语句是类似的，但是需要额外注意的是，pgfplots 的数据文件要求与 gnuplot 一样，需要断成块，中间需要一行空行。数据文件要写成，

```
x1, y1, f(x1,y1)
x1, y2, f(x1,y2)
x1, ..., ...
x1, yn, f(x1,yn)
空行
x2, y1, f(x2,y1)
x2, y2, f(x2,y2)
x2, ..., ...
x2, yn, f(x2,yn)
空行
...
```

x 相同的是一组数据块，还是 y 相同是一组数据块，结果是一样的，但是空行隔断是必须有的。

这方面 origin 做得最好，数据不断行也能画，但是 origin 画图也慢，画图最快的是 gnuplot。python 的画图库，自己的数据比较容易，从数据文件画图则比较麻烦。用 python 处理数据，包括给没有断行的数据断行，然后用 gnuplot 画图，个人认为是最优的画图方式。如果需要美化图例或标识，可以 gnuplot 画图到 eps 文件，然后使用 node 完成加载和美化。也可以使用设置参数，gnuplot 直接输出为 tex 文件，然后修改 tex 文件美化。

pgfplots 的画图功能相对来说越来越完善，数据量一般的情况下，pgfplots 画图相对来说比较美观。设置虽然麻烦些，但是做出来的各种效果都还不错。手册比较长，写得还算详细，感兴趣的可以自行练习。

4.18 基于 tikz 和 pgf 的科学包或 tikz 库

4.18.1 高亮数学公式

hf-tikz 包用来高亮数学公式，标记重点步骤。

```
\usepackage{hf-tikz}
```

4.18.2 物理类

推荐使用 pgf-spectra，这是一个元素光谱包，可以画可见光光谱。

```
\usepackage{pgf-spectra}
```

默认是 NIST 数据，如果希望加载 LSE 数据，可以添加包选项

```
\usepackage[LSE]{pgf-spectra}
```

内置了一些原子谱线数据，texlive 2021 必须将 spectra.data.NIST.tex 或者 spectra.data.LSE.tex 文件 copy 到 tex 文件目录下才可以正确编译，因为 pgf-spectra.sty 文件里的 input 是当前目录。修改 pgf-spectra.sty 中的 input 语句里的 spectra.data.NIST.tex 或者 spectra.data.LSE.tex 的目录为安装目录（全路径名），这样可以直接使用。

pgf-spectra 包支持自定义谱线，也可以按 spectra.data.NIST.tex 或者 spectra.data.LSE.tex 的谱线数据格式自行添加自己常用的光谱数据。

画费曼图的包 tikz-feynman，该包必须 lualatex 编译，latex, pdflatex 或 xelatex 编译都无法正确显示。加载语句如下

```
\usepackage{tikz-feynman}
```

经常画费曼图使用该包比较方便，但是偶尔画一下，直接用 tikz 画也可以，省得再去学语句了。

还有一个费米图包是 tikz-feynhand，是 tikz-feynman 的低配版，tikz-feynhand 可以直接 pdflatex 编译。

```
\usepackage{tikz-feynhand}
```

tikzorbital 是画分子轨道的包，不算很漂亮，但是能用。

```
\usepackage{tikzorbital}
```

粒子加速器包 tikz-palattice，内置了一些加速器的常用组件。

```
\usepackage{tikz-palattice}
```

这个包还算漂亮，配色也不错，不是这个领域的功能不好评价。

tikz-planets是一个开始于 2019 年的新包，画星球用的。

```
\usepackage{tikz-planets}
```

4.18.3 量子

quantikz是一个库，画量子电路用的。

```
\usetikzlibrary{quantikz}
```

4.18.4 光路图包 tikz-optics

画光路图的包 tikz-optics，手册更新到 2017，法语手册。这个包感觉会就此放弃，而且功能上也不是很强。

```
\usepackage{tikz-optics}
```

pstricks 画图体系有一个光路图包，手册一直在更新，

```
\usepackage{pst-optexp}
```

pstricks 画图体系还有一个几何光路图包，手册更新到 2016 年，效果还不错。使用时必须注意，pstricks 的语句是空格敏感的，没有空格的地方要严格按手册不能添加空格。

```
\usepackage{pst-optic}
```

直接用 tikz 就可以画出漂亮的光路图，但是凸透镜笔者一般会偷懒为椭圆。pst-optexp 手册封面的光栅彩图用 tikz 也可以做出。

4.18.5 工程

标尺寸的包 tikz-dimline，因为机械画图软件很多很强大，所以这个包也就不是很有用了。

```
\usepackage{tikz-dimline}
\usepackage{wasysym} %diameter 需要
```

轨道示意图是一个 2018 年开始的新包，

```
\usepackage{tikz-trackschematic}
```

tikz-network 是一个画网络理论的新包

```
\usepackage{tikz-network}
```

4.18.6 chart

Texlive 里包含的卡诺图包很多，tikz 自己有一个卡诺图的库

```
\usetikzlibrary{karnaugh}
```

tikz-cd 有点类似数学的 cd 库，加载 tikz-cd 有两种方式，包和库都行

```
\usepackage{tikz-cd}
\usetikzlibrary{cd}
```

4.18.7 其他

语言学包 tikz-dependency 非常漂亮

```
\usepackage{tikz-dependency}
```

tikzmark 是 tikz 的库，

```
\usetikzlibrary{tikzmark}
```

tikzrput 包用来放置图或文字

```
\usepackage{tikzrput}
```

4.19 基于 tikz 和 pgf 的其他包

4.19.1 装饰纹包 pgfornament

pgfornament 包设计请柬或者彩笺时比较有用。

```
\usepackage{pgfornament}
```

还有一个汉化的国风花纹包。

```
\usepackage{pgfornament-han}
```

4.19.2 tikz-among-us

AmongUs 是一个游戏，这个新包做得挺漂亮的，虽然对笔者没啥用。

```
\usepackage{tikz-among-us}
\usepackage{tikz-among-us-fancyhdr}
\usepackage{tikz-among-us-watermark-eso-pic}
```

4.19.3 tikz-truchet

这是一个画瓷砖图案的包。

```
\usepackage{tikz-truchet}
```

4.19.4 动物包

人物包比较推荐，

```
\usepackage{tikzpeople}
```

鸭子包

```
\usepackage{tikzducks}
```

土拨鼠

```
\usepackage{tikzmarmots}
```

动物的合集包

```
\usepackage{tikzlings}
```

4.19.5 符号

tikzsymbols 包可以画各种符号和图标，

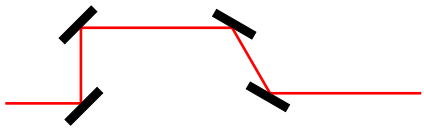
```
\usepackage{marvosym}
\usepackage{tikzsymbols}
```

4.20 几个光路图的例子

光路图经常会用到反射镜，使用 node 画反射镜最方便，因为 node 默认的旋转点是 node 语句放置文本框的坐标点，而设置位置左右后 node 语句的坐标点就在 node 边框上。等同于绕该点可以 360 度旋转。需要注意的是，要将 inner sep 设置为 0pt，这样才能用 minimum width 将反射镜的宽度设置为任意宽度，否则会附加 inner sep 的宽度。而且一定要设定 draw 和 fill 都为 black，并且 draw 的线宽等于光路的线宽，更方便的是设置为极小，这样放大看图片后光路才是真正的反射效果。不能只 fill，不 draw，不设置线宽，否则放大效果就不好了，当然不放大实际上不大看得出来。反

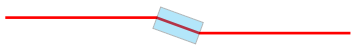
射镜的大小可以根据整体图片的尺寸进行调整，一般情况下同一张图尺寸都是一致的。特殊情况下可以特别设定尺寸，用 shift 语句移动反射点在镜子上的位置。可以将镜子 node 的设置自定义后重复使用。比较好计算的坐标可以分开写，不好计算的写到划线语句里面很方便。

```
\begin{tikzpicture}[>=Latex]
\tikzset{mymirror/.style={draw=black,line
width=0.1pt,fill=black,inner sep=0pt,minimum width=3pt,minimum
height=6mm}}
\draw [red,line width=1pt] (0,0)--++(1,0)--++(0,1)--++(2,0) node
[right,mymirror,rotate=60] {} --++(-60:1) node
[left,mymirror,rotate=60] {} --++(2,0);
\node at (1,0) [right,mymirror,rotate=-45] {};
\node at (1,1) [left,mymirror,rotate=-45] {};
\end{tikzpicture}
```



同理，画光经过任意厚度的介质发生折射，也用 node 画，设置 opacity 的数值显示介质透明。这种情况下计算量其实很小，光线偏折用双加号极坐标，设置 node 宽度为光线偏折走的长度，高度为介质宽度，旋转点为入射点即可。放大很多倍看是有点问题的，考虑到光线宽度就不对了，但是基本上视觉感观是可以接受的。

```
\begin{tikzpicture}
\draw [red,line width=1pt] (0,0)--++(2,0)--++(-20:6mm)--++(2,0);
\node at (2,0) [right,draw=black, very
thin,fill=cyan,opacity=0.3,inner sep=0pt,minimum
width=6mm,minimum height=3mm,rotate=-20] {};
\end{tikzpicture}
```



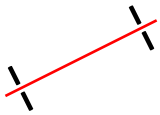
光阑可以用 double 线型画，垂直与平行的时候很方便。中间颜色为白色，所以要放在表示激光的红线下面。

```
\tikzset{myap-1/.style={double,line width=2mm,double
distance=2pt}}
\begin{tikzpicture}
```

```
\draw [myap-1] (1,0)--++(2pt,0);
\draw [red,line width=1pt] (0,0)--++(2,0);
\end{tikzpicture}
```

double 的方式画光阑有一个问题，遇到斜着走的光路就只能用坐标计算，即 calc 库，来解决。但是很多时候光路是通过双加号一笔画出来的，即使 tikz 的 calc 库很好用，这样的情况也不能很好匹配。比较推荐的方法是用 edge，使用 pos 定位很方便移动，这样画出来的光阑与直线画出来的光阑放大多倍后略有差别。下面的例子，预先定义了一个命令和两个 style。这里需要 quotes 库和 positioning 库。因为用 below 设置具体位置，所以可以不加表示在下面的撇。

```
\newcommand\setap{\resizebox{12pt}{2mm}{$|$}}
\tikzset{myap-above/.style={black,sloped,above=-0.9mm}}
\tikzset{myap-below/.style={black,sloped,below=-0.9mm}}

\begin{tikzpicture}
\draw [red,line width=1pt] (0,0) edge
["\setap"{myap-above,pos=0.1}, "\setap"{myap-below,pos=0.1},
"\setap"{myap-above,pos=0.9}, "\setap"{myap-below,pos=0.9}]
++(2,1);
\end{tikzpicture}
```



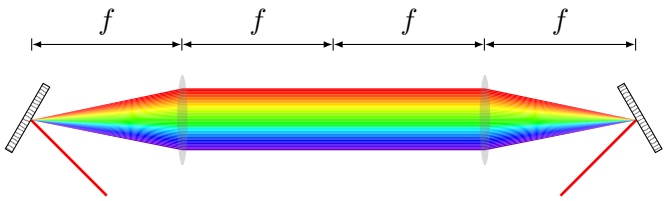
下面的代码还将使用 patterns 库，用来填充 node 显示为光栅。

```
\usetikzlibrary{patterns,patterns.meta}
```

画光栅色散可以使用\foreach语句，颜色必须设置一个中间变量先计算出来再使用。

```
\begin{tikzpicture}[>={Latex[scale width=0.7,scale length=0.7]}]

\draw [|<->|] (-4,1)-- node [above] {$f$} ++(2,0);
\draw [<->|] (-2,1)-- node [above] {$f$} ++(2,0);
```

```
\draw [<->|] (0,1)-- node [above] {$f$} ++(2,0);
\draw [<->|] (2,1)-- node [above] {$f$} ++(2,0);

\draw [red,line width=1pt] (-3,-1)--(-4,0);
\draw [red,line width=1pt] (3,-1)--(4,0);

\foreach \x in {-0.4,-0.38,...,0.4}
{
\tikzmath{\c=\x*310+530;}  %先计算出颜色的波长值
\definecolor{wavecolor}{rgb:wave}{\c} %加 rgb: 方便调用
\fill [wavecolor] (-4,0)-- ++(2,\x)-- ++(4,0)-- ++(2,-\x)--
++(-2,{\x+0.02})-- ++(-4,0)--cycle;
}

\fill [gray,opacity=0.3] (-2,0) ellipse (0.07 and 0.6);
\fill [gray,opacity=0.3] (2,0) ellipse (0.07 and 0.6);

\node at (-4,0)
[left,draw,pattern={Lines[angle=-30,distance=0.5mm,line
width=0.3pt]},pattern color=gray,inner sep=0pt,minimum
width=1mm,minimum height=1cm,rotate=-30] {};
\node at (4,0)
[right,draw,pattern={Lines[angle=30,distance=0.5mm,line
width=0.3pt]},pattern color=gray,inner sep=0pt,minimum
width=1mm,minimum height=1cm,rotate=30] {};

\end{tikzpicture}
```



我们画光路经常会遇到光路是镜像对称的情况，添加 environ 包方便我们定义新环境。

```
\usepackage{environ}
```

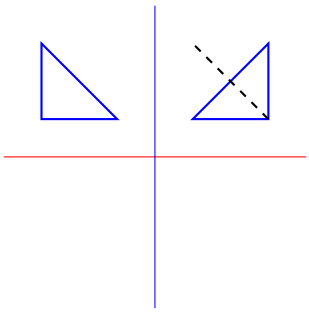
然后定义环境 reverse，无论是镜像还是反演，本质上是坐标变换。下

面的环境定义了 scope 内坐标的改变，不影响外面的语句。

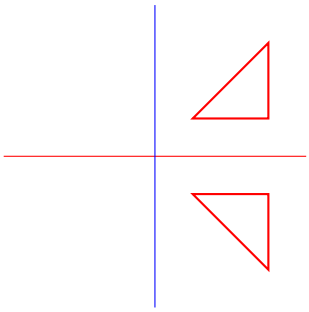
```
\NewEnviron{reverse}[2]{
\BODY
\begin{scope}[xscale= #1,yscale=#2]
\BODY
\end{scope}}
```

然后使用该环境。

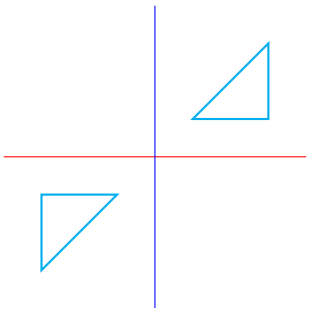
```
\begin{tikzpicture}
\draw [red] (-2,0)--++(4,0);
\draw [blue] (0,-2)--++(0,4);
\begin{reverse}{-1}{1} %关于 y 轴对称
\draw [blue,thick]
(0.5,0.5)-|++(1,1)--cycle;
\end{reverse}
\draw [thick,dashed]
(1.5,0.5)--++(135:{sqrt(2)});
\end{tikzpicture}
```



```
\begin{tikzpicture}
\draw [red] (-2,0)--++(4,0);
\draw [blue] (0,-2)--++(0,4);
\begin{reverse}{1}{-1} %关于 x 轴对称
\draw [red,thick]
(0.5,0.5)-|++(1,1)--cycle;
\end{reverse}
\end{tikzpicture}
```



```
\begin{tikzpicture}
\draw [red] (-2,0)--++(4,0);
\draw [blue] (0,-2)--++(0,4);
\begin{reverse}{-1}{-1} %关于原点反演
\draw [cyan,thick]
(0.5,0.5)-|++(1,1)--cycle;
\end{reverse}
\end{tikzpicture}
```



第五章　一些有用的包

本章介绍一个经常用到的类 standalone 和几个非常有用的包。

5.1　standalone 类

standalone 是一个很好的类，可以将指定的图片环境编译为 pdf 文件中独立的一页，且页面大小随图片大小自动调整。平时使用时推荐每张图片一个 tex 文件，生成一页 pdf，方便将图片插入到文件中。下面是基于 tikz 的文件框架，方括号里的参数还可以是 pstricks，以及 animate。animate 情况只能是一个 animateinline 环境，生成一页动图，不能像 tikz 和 pstricks 那样为多页。

```
\documentclass[tikz]{standalone}
\standaloneconfig{border=5pt} %设置图片四周留白

\usepackage{CJKutf8}
\usepackage{tikz}

\begin{document}
\begin{CJK}{UTF8}{gkai}

\begin{tikzpicture}
\fill [draw=black,fill=yellow] (0,0) circle (1);
\end{tikzpicture}

\end{CJK}
\end{document}
```

standalone 的另一个用处是用 minipage 环境来构建宽度固定长度自动的笔记，非常适合做计算机语言程序的笔记。下面的例子宽度其实也是可以随意调整的。类的参数必须写成`multi=minipage`，否则无法分页。

```
\documentclass[multi=minipage]{standalone}
\standaloneconfig{border={5mm 5mm 5mm 5mm}} %设置四周留白

\usepackage{CJKutf8}

\begin{document}
\begin{CJK}{UTF8}{gkai}

\begin{minipage}{12cm} %设置宽度
内容
\end{minipage}

\end{CJK}
\end{document}
```

standalone 并不是支持所有环境，它支持那些可以明确尺寸的环境。比如说 tabular 是可以的（也需要加 multi=），一张表格的尺寸是明确的，但是可分页的 framed 环境就不行了。美丽的 tcolorbox 环境不行。

standalone 还支持每个公式一页，但是必须通过varwidth=宽度明确每页的宽度。

```
\documentclass[multi=equation,varwidth=6cm]{standalone}
```

5.2 计算机程序语法高亮：listings

本书的程序语法高亮就是用 listings 完成的。下面给出的是本书的设置。

```
\usepackage{listings} %加载 listings 包
%基本设置
\lstset{
language=[LaTeX]TeX, %设置语言，支持 90 多种常用语言
tabsize=2, %设置 tab 键宽度
basicstyle=\small, %设置字号，代码长的话用 small 比较好
frame=tlrb, %设置边框，tlrb 表示四周都有边框
backgroundcolor=\color{gray!10}, %设置背景色
commentstyle=\rm\color{purple}, %设置注释颜色
keywordstyle=\bf\color{blue}, %设置关键词颜色
morekeywords={maketitle,chapter,subsection,subsubsection}, 自定义关
键词
```

```
%mathescape=true,
columns=fullflexible, %强烈推荐这个设置，否则 pdf 有时产生多余空格
texcl=true, %这里设置为真
extendedchars=false, %建议为假
breaklines=true, %自动换行，xelatex 对项支持最好
breakindent=0pt, %换行缩进，此处设置为无缩进
%showspaces=true, 显示不显示空格，显示空格的效果也不大好
aboveskip=6pt, %距离文本的上间距
belowskip=6pt, %距离文本的下间距
}
```

如果需要高亮的语言不在手册语言列表中，可以自己写配置文件，或者加载接近的语言，然后自定义 keywords。

```
\begin{lstlisting}[language=C++]
#include <stdio.h>
int main()
{
  printf("hello, world\n");
}
\end{lstlisting}
```

用方括号 [language=] 方式定义语言，只对该环境有效，因此可以在一个文档内使用多种语言的语法高亮。

可以同时更换语言和背景

```
\begin{lstlisting}[language=HTML, backgroundcolor=\color{blue!20}]
<!DOCTYPE HTML>
<html>
<body>

</body>
</html>
\end{lstlisting}
```

当然也可以使用\lstset重新设置参数。

行间语言高亮为

```
\lstinline!语言!
```

listings 使用方式的代码展示不能简单地使用 listings 环境，因为不能

嵌套环境，否则会产生嵌套错误（配对错误）。上面的例子是 tcolorbox 包里的 tcblisting 环境做出来的，这个环境下面还会讲。不想增加额外的包，也可以使用 verbatim 环境，这个环境将里面的内容当作文本如实展现。

在 beamer 中使用 listings 时，必须加载 frame 的 fragile 选项，**否则会剧烈报错！**

```
\begin{frame}[fragile]{}
使用lstlisting环境
\end{frame}
```

加载 showexpl 包，可以展现代码和结果

```
\usepackage{showexpl}
\lstset{explpreset={numbers=none}} %取消行号
\setboolean{@SX@varwidth}{true} %图片自动宽度，需要 ifthen 包
```

设置结果（图片）box 是 colorbox，取消框，并设置图片边缘留白 5pt。

```
\renewcommand\ResultBox{\colorbox{lightgray}}
\setlength\ResultBoxSep{5pt}
```

然后可以直接使用 LTXexample 环境

```
\begin{LTXexample}
\begin{tikzpicture}
\draw (0,0)--(1,0);
\end{tikzpicture}
\end{LTXexample}
```

得到左边是图片，右边是代码

```
\begin{tikzpicture}
\draw (0,0)--(1,0);
\end{tikzpicture}
```

需要修改什么，就阅读手册后面的源码，然后试着修改设置项。tikz 的手册是 lualatex 编译的，自定义了代码展示环境。tcolorbox 也自定义了 tcolorbox 样式的代码展示环境，下面会讲。一般的需求 showexpl 包就可以满足了，但是这个包对上下文竖直间距的处理很不好。

5.3 tcolorbox

tcolorbox 的手册写得很好，自学也可以的。讲课的时候也只讲了几个简单例子。这个包非常漂亮，几乎年年更新。这里主要讲一下几个预先定义好的环境。

5.3.1 数学公式

\tcboxmath{}可以放在数学公式环境内。左边是默认效果，右边是修改的结果，大部分设置都需要写到里面。

```
\tcbset{}
\begin{equation}
\tcbset{fonttitle=\scriptsize}
\tcboxmath{f(x)}=
\tcboxmath[boxsep=2pt,top=0pt,bottom=0pt,left=0pt,right=0pt,sharp
corners,colback=cyan!20]{\sin x}
\end{equation}
```

$$f(x) = \sin x \tag{5-1}$$

\tcbhighmath{}用法接近，默认配色不同，可以通过 highlight math style 统一设置形式。可以与 empheq 包混用。

```
\tcbset{highlight math style={tcolorbox参数}}
\begin{empheq}[box=\tcbhighmath]{equation}
f=\sin x
\end{empheq}
\begin{equation}
\tcboxmath{f}=\tcbhighmath{\sin x}
\end{equation}
```

$$f = \sin x \tag{5-2}$$

$$f = \sin x \tag{5-3}$$

5.3.2 theorems

加载定理库。

```
\tcbuselibrary{theorems}
```

使用\newtcbtheorem命令定义 tcolorbox 形式的新的定理环境。

```
\newtcbtheorem[定理参数]{新定理环境名}{显示名}{tcolorbox的参数}{label
 separator}
```

下面是一个例子

```
\newtcbtheorem[number
within=chapter]{solution}{习题答案}{colback=green!5,
colframe=green!35!black, fonttitle=\bfseries}{th}
\begin{solution}{此题主要练习...}{s1} %s1 是该定理的标签
一个例子\ref{th:s1} %引用是添加前缀
\end{solution}
```

习题答案 5.1: 此题主要练习...

一个例子5.1

或者使用下面的方式也是等价的。

```
\newtcbtheorem[number
within=chapter]{mysolution}{习题答案}{colback=green!5,
colframe=green!35!black, fonttitle=\bfseries}{th}
\begin{mysolution}[label=s2]{此题主要练习...}{s3}
一个例子\ref{s2}两种方式是等价的\ref{th:s3}
\end{mysolution}
```

习题答案 5.1: 此题主要练习...

一个例子5.1两种方式是等价的5.1

5.3.3 tcblisting

加载 listings 库。

```
\usepackage[most]{tcolorbox}
\tcbuselibrary{listings,skins}
```

定义了 tcblisting 环境，listing only 只显示代码；listing side text 一边显示代码，一边显示结果；listing side text ，上边显示代码，下边显示结

果。tcblisting 环境 xelatex 编译的时候会出现 tab 键显示不正确，pdflatex 编译的时候没有问题。所以使用这个环境且 xelatex 编译时，代码里不要包含 tab 键，尽量使用空格。

```
\begin{tcblisting}{listing only}
listing only只显示代码
\end{tcblisting}
```

\newtcblisting可以定义新的 tcblisting 环境。参数中，side by side，并排方式，不能分页。上下方式加 breakable 参数，可以自动分页。这本书定义了几个环境展示代码和代码结果。基本保持了与 lstlisting 环境显示差不多的效果。

```
\newtcblisting{pcode}{bicolor,colframe=white,boxsep=2pt,top=0pt,
bottom=0pt,left=0pt,right=0pt,sharp corners,sidebyside gap=5mm,
colback=white,colbacklower=white,righthand width=0.3\textwidth,
listing side text,sidebyside align=center seam}

\newtcblisting{wcode}[1][]{bicolor,colframe=white,boxsep=2pt,top=0pt
,bottom=0pt,left=0pt,right=0pt,sharp corners,sidebyside gap=5mm,
colback=white,colbacklower=white,listing side text,sidebyside align=
center seam,righthand width=#1}

\newtcblisting{vcode}{skin=enhanced,breakable,bicolor,colframe=white
,boxsep=2pt,top=0pt,bottom=0pt,left=0pt,right=0pt,sharp corners,
sidebyside gap=5mm,colback=white,colbacklower=white,middle=2pt,
listing and text}

\newtcblisting{cpcode}{bicolor,colframe=white,boxsep=2pt,top=0pt,
bottom=0pt,left=0pt,right=0pt,sharp corners,sidebyside gap=5mm,
colback=white,colbacklower=yellow!70!gray!50,righthand width=0.4\
textwidth,listing side text,sidebyside align=center seam}

\newtcblisting{cwcode}[1][]{bicolor,colframe=white,boxsep=2pt,top=0
pt,bottom=0pt,left=0pt,right=0pt,sharp corners,sidebyside gap=5mm,
colback=white,colbacklower=yellow!70!gray!50,listing side text,
sidebyside align=center seam,righthand width=#1}
```

5.4 动图：animate

重点需要讲的是动图，也最容易出错。个人认为，animate 包最好结合 standalone 使用，然后用超链接方式将动图的 pdf 加载到文件或 ppt 里。因为 animate 的编译非常耗时。还有一个办法就是做好动图后注释掉，最终版本再加载。这样可以节省修改文件其余部分的编译时间。当然，下面的例子都是可以放在文章、书或 beamer 类里的。

5.4.1 利用多张图片实现动图

这种方式跟 gif 动态类似，实质上都是多张图片依次播放。此时必须加载图片包 graphicx。

```
\documentclass[animate]{standalone} %animate 可以不加
\standaloneconfig{border=5pt}

\usepackage{graphicx} %必须有图片包
\usepackage{animate}

\begin{document}

\animategraphics[width=12cm,autoplay,loop]{12}{../fig/fiber_}{0}{23}

\animategraphics[width=12cm,controls,loop]{2}{../fig2/f}{1}{5}

\end{document}
```

\animategraphics的说明如下：

- 第一个是方括号：设定动图参数。
  自动播放动图使用参数 autoplay，带播放按钮不自动播放动图使用参数 controls。参数 loop 是循环播放的意思，否则只播放一次。一般情况总是 autoplay 和 loop 同时使用。controls 随意。如果图片过大，需要设定宽度缩小一下。不推荐同时设定宽度和高度，设定一个，另一个会自动按比例缩放，这样不会扭曲图片。但是如果图片的尺寸不一致，宽度设置可能会带来问题，不设置效果也不好。当然，一般多张图片产生的动图都是相同的尺寸。
- 第二个是花括号，给出每秒多少帧。
- 第三个是花括号，给出图片名称。

图片名称必须包括路径，例子里图片并没有放在当前目录下，而是放在父目录下两个不同子目录里。真正的图片名称的规则是：相同的文件名和编号。此处给出的图片名是相同的文件名那部分，不带编号。第一个例子是推荐的下划线命名规则，比较清晰。第二个例子图片的名字就是简单的 f1、f2、f3。**动图所需的图片名必须连续编号**，但是可以不从 0 开始。起始编号可以任意。pdflatex 编译方式允许的图片格式都可以使用，pdf、jpg 和 png 都行。不用给出具体的图片格式。**图片格式可以混用**。

- 第四个是花括号，给出起始编号。
- 第五个是花括号，给出终止编号。

需要起码两次编译。支持 pdflatex 和 xelatex 编译方式，直接得到 pdf 文件。

必须使用带 javascript 功能的 pdf 阅读器才会出现动图效果，比如 adobe 的 acrobat reader，这个是免费的软件，最新版一般默认支持 javascript，不用再去设置。大部分小巧快速的 pdf 阅读器都无法展现动图效果，包括 texlive 自带的 pdf 阅读器也不行。必须切换到 adobe reader 查看动图效果。

adobe reader 是不支持自动更新的，因此修改并再次编译 tex 文件时，必须先关闭 adebe reader，编译两次后再用 adobe reader 打开查看。

除了 adobe reader 外，手册里还给出了 KDE Okular, PDF-XChange, Foxit Reader 三种 pdf 阅读器，笔者没有测试过。

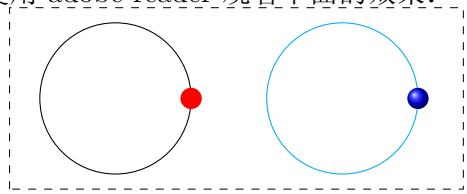
5.4.2 tikz 直接画动态图

这小节讲的是一张图片中有动态模块这样的情况。比如说下面的例子。

```
\documentclass[animate]{standalone}
\standaloneconfig{border=0pt}

\usepackage{xcolor}
\usepackage{graphicx}
\usepackage{animate}

\usepackage{tikz,pgf}
\usetikzlibrary{shapes.arrows,calc}

\begin{document}
```

```
%必须先画个框，能把所有图框起来，这样图形才能稳定住，才能动起来
\begin{animateinline}[autoplay,loop]{12}
\multiframe{24}{iAngle=0+15}{\begin{tikzpicture}
\draw[line width=0.1pt, dashed] (-1.4,-1.2) rectangle (4.6,1.2);
\draw (0,0) circle (1);
\draw [color=cyan] (3,0) circle (1);
\coordinate (a1) at (\iAngle:1);
\fill[red] (a1) circle (4pt);
\coordinate (a2) at ({3+cos(-\iAngle)},{sin(-\iAngle)});
\shade [ball color=blue] (a2) circle (4pt);
\end{tikzpicture}}
\end{animateinline}

\end{document}
```

请使用 adobe reader 观看下面的效果：



另一个例子是加载图片并使图片旋转：

```
\documentclass[animate]{standalone}
\standaloneconfig{border=0pt}

\usepackage{xcolor}

\usepackage{graphicx}
\usepackage{animate}

\usepackage{tikz,pgf}
\usetikzlibrary{shapes.arrows,calc}

\begin{document}
%框的颜色可以设为 white，框就不会显示出来
%图片是圆的，框的尺寸不能直接按直径，需要略大于 2cm 方块旋转的外轮廓尺寸，可能是 graphics 还有一个外框的缘故
\begin{animateinline}[autoplay,loop]{6}
\multiframe{24}{iAngle=0+15}{\begin{tikzpicture}
\draw[color=white,dashed] (-2,-2) rectangle (2,2);
```

```
\node [rotate=\iAngle] at (0,0) {\fbox{\includegraphics[width=2cm]{
fig/tju.pdf}}};
\end{tikzpicture}}
\end{animateinline}

\end{document}
```

这里加了一个图片外框方便估计旋转时可能扫过了面积大小，外框需要比这个面积更大一些。可以加\clip语句保证旋转的面积只是圆本身。

5.4.3 tikz 多张画图方式

这小节的例子是用 tikz 画多种很不相同的图片，每个图片是一个newframe。

```
\documentclass[animate]{standalone}

\usepackage{xcolor}
\usepackage{graphicx}

\usepackage{tikz,pgf}
\usetikzlibrary{calc}
\usepackage{animate}

\begin{document}

\begin{animateinline}[autoplay,loop,begin={\begin{tikzpicture}\
useasboundingbox(-0.2,-0.5) rectangle(3.3,0.8);},end={\end{
tikzpicture}}]{1}

%后接多行 tikz 语句构成一幅图，每张图都不一样
\foreach \x in {0,1,2}
```

```
\fill (\x,0.5) circle (2pt); %三个点

\newframe \node at (0,0) [fill=red!20] {A}; %红色方块，写到一行也可以

\newframe
\node at (1,0) [fill=yellow!20] {B}; %黄色方块

\newframe
\node at (2,0) [fill=blue!20] {C}; %多行语句不需要加花括号，两个方块
\node at (3,0) [fill=green!20] {C};
\end{animateinline}

\end{document}
```

第一个 newframe 前的内容最先展示，第二个展示内容才是第一个 newframe 的内容。相当于省去了一个 frame。

• • •

5.4.4 时序控制

这一小节讲述通过时序文件控制显示顺序，组合加载 newframe 里的内容。首先是主 tex 文件如下：

```
\documentclass[animate]{standalone}

\usepackage{xcolor}
\usepackage{graphicx}



\usepackage{tikz,pgf}
\usepackage{pgfplots}
\usetikzlibrary{calc}
\usepackage{animate}



\begin{document}

%timeline= 时序文件名
\begin{animateinline}[autoplay,loop,begin={\begin{tikzpicture}\
```

```
useasboundingbox(-0.4,-0.3) rectangle(2.3,0.3);},end={\end{
tikzpicture}},timeline=a.timeline.txt]{2}
\newframe \node at (0,0) [fill=red] {A};
\newframe \node at (0.5,0) [fill=yellow] {B};
\newframe \node at (1,0) [fill=green] {C};
\newframe \node at (1.5,0) [fill=blue] {D};
\newframe \node at (2,0) [fill=cyan] {T};
\end{animateinline}
\end{document}
```

windows 系统下，时序文件名建议扩展名设置为 txt，linux 下可以不加任何扩展名。时序文件名本身随意。这里因为 tex 是 a.tex，所以时序文件名设置为了 a.timeline.txt。文件如下：

```
::0
*::1,2,3,4
:1:c,1
::2
::c,3
::c,4
::c,2x4,3,4
::1,4
::c,5
::1
```

上面的时序文件只有代码，timeline 的格式为

```
[*]:速率:frame页ID x 动画页数目:代码
```

最前面的星号表示停顿，如果加星，该行显示后会停顿等待鼠标响应后再开始动画。如果是动画的最后一页加星停顿，且设置了循环 loop，循环会覆盖停顿，自动循环。

如果设置速率，会覆盖默认速率，对后面所有的动画页有效，直到再次开始循环，开始循环时回到默认速率。

c 表示清除。

2x4 表示对第 2 个 frame，也即 B，显示 4 页。2x0 则表示一直存在直到清除。仅仅只有 2 表示 2x1，当前页存在。

上面的例子没有代码，所以连后面的冒号一起省略。

最后的结果是：

1. 第一行，显示一个白色背景，可以没有。
2. 第二行，显示 1234，即 ABCD，停顿等待鼠标响应。
3. 第三行，c 表示清除掉前面所有的内容，显示 1，即 A。从这里开始速率为 1 秒 1 页。
4. 第四行，显示 2，即 B。默认只显示一页，所以不用加清除也只会显示 B。
5. 第五行，清除掉所有的内容，显示 3，即 C。
6. 第六行，清除掉所有的内容，显示 4，即 D。
7. 第七行，清除掉所有的内容，显示 234，即 BCD，设置 B 显示 4 页。
8. 第八行，显示 14，加上前面的 B 设置为在 4 页显示，所以即使没有写显示 B，得到的是 ABD。
9. 第九行，只显示 T，前面的 c 覆盖了 2x4 的 4 页显示设置，所以 B 不再显示。
10. 第十行，只显示 A，第九行的 c 对后面都有效，虽然 B 设置显示 4 页，实际只会显示 2 页
11. 循环

timeline 文件用分号分层，如果改写 timeline 文件为如下形式

```
::0
*::1,2,3,4
:1:c,1
::2
::c,3
::c,4
::c,2x4,3,4
::1,4x0;3x0
::c,5
::1
```

C 就会在后面一直存在，因为::c,5 只清除第一层，第二层的 3x0 会一直存在。如果写为

```
::c,5;c
```

则 C 就不存在了，因为清除了两层，只会留下 T。

5.5 试卷：exam 类

试卷 exam 类很方便教师排版试卷，排版时带着答案排版，编译是通过包的 answers 选项控制答案是否显示。

```
\documentclass[oneside,twocolumn,12pt]{exam} %空白试卷
\documentclass[answers,oneside,twocolumn,12pt]{exam} %显示答案
```

天津大学的本科生试卷一般是 8 开双栏页面

```
\usepackage[paperwidth=37cm,paperheight=26cm,top=3.5cm,bottom=1.8cm
,left=1.7cm,right=1.7cm,bindingoffset=0.5cm]{geometry}
```

有一个黑框，用 background 包和 tikz 包做比较简单

```
\backgroundsetup{scale=1,contents={\begin{tikzpicture}\draw [line
width=1pt,black] (0,0)rectangle ++(33.5,21.4);
\end{tikzpicture}},opacity=1,angle=0,hshift=5pt,vshift=-21pt}
```

exam 类包含了 fancyhdr，所以直接用下面的语句就可以设置页眉

```
\chead{\large\shortstack{天津大学试卷专用纸\\[6mm]学院\underline{\
hspace{3.5cm}}专业\underline{\hspace{5cm}}\hspace{1cm}\underline{\
hspace{1cm}}班\hspace{1cm}年级\underline{\hspace{2cm}}学号\
underline{\hspace{5.5cm}}姓名\underline{\hspace{3cm}}\hspace{1cm}共
\ \protect\pageref{lastpages}~页~\hspace{5mm}第\ \thepage~页~}}

\pagestyle{head}
```

这里在最后一页添加了一个 label，可以自动显示共几页。

```
\label{lastpages} %最后一页添加一个 label
\clearpage %这个命令必须存在
```

其他的设置基本是目测调出来的，考试科目部分设置为 1.7 倍行距。

```
\begin{spacing}{1.7}
\centering
\large 2019\ $\sim$\ 2020 学年第 2 学期期末考试试卷\\
《CCC课程》（A或B卷 共 \pageref{lastpages} 页）\\
\scalebox{0.9}{\large（考试时间：2020 年 7 月 2 日）}
\end{spacing}
```

分数表部分和试卷以下部分设置为 1.5 倍行距，填空题用\fillin[答案

][横线长度]命令。

```
\begin{spacing}{1.5}
\vspace*{-1mm}
\setlength{\tabcolsep}{11pt}
\setlength{\extrarowheight}{2pt}
\begin{tabular}{|c|c|c|c|c|c|c|c|c|c|}\hline
题号 &一 &二 &三 &四 &五 &六 &七 &八 &成绩 &核分人签字\\ \hline
得分 & & & & & &&& & &\\ \hline
\end{tabular}

\vspace{3mm}

一、填空（每空1分，共20分）
\begin{questions}
\question %每小题自动编号
过程需要满足\fillin[能量]守恒和\fillin[动量]守恒。
\end{questions}
\end{spacing}
```

默认填空横向长度用下面的语句设置，根据排版需要可以随时调整。

```
\setlength\fillinlinelength{5cm}
```

答案不加粗

```
\CorrectChoiceEmphasis{\rm}
```

横向下沉一点距离

```
\setlength\answerclearance{0.5ex}
```

简答题和证明计算题部分一般采用 1.25 倍行距或者单倍行距，这样可以压缩题目本身所占空间。

设置简答题和证明计算题答案不加框、不缩进

```
\unframedsolutions
\renewcommand{\solutiontitle}{\noindent}
```

简答题和证明计算题事先计算好宽度和高度，采用 parbox 方式，这样答案是否显示都不影响版面。

```
\parbox[][4cm][t]{0.47\textwidth}{
\question
```

```
为什么？

\begin{solution}
答案
\end{solution}
}
```

如果有图，可以采用两种方式，图在上面，或者图和答案并排。我自己的试卷一般采用上下方式，不用修改上面的代码。并排可以采用 minipage 环境，也可以采用 parbox 嵌套方式，下面是 minipage 方式的代码。此时仍然需要将整个题目（包括答案）用 parbox 封装起来。

```
\parbox[][10cm][t]{0.47\textwidth}{
\question
\begin{minipage}{6cm}
\includegraphics[width=5cm]{../../exam-fig/}
\end{minipage}
\begin{minipage}{9cm}
\begin{solution}

\end{solution}
\end{minipage}
}
```

如果有选择题，用下面的语句即可。

```
\question
下面哪种描述是正确的？\fillin[ABD]
\begin{choices}
\choice 陈述1
\choice 陈述2
\choice 陈述3
\choice 陈述4
\end{choices}
```

选择题建议使用 1.25 倍行距压缩内容所占空间，但是期中小考的时候我经常填空和简答混出，所以小考的试卷我设置为 2 倍行距。填空题因为要留给学生写答案，1.5–2 倍行距比较合适。

如果选项字数比较少的话，也可以使用 oneparchoices 环境把选项放在同一行。

```
\question
下面哪种食品有利于健康\fillin[AB] %下面空行或强制换行

\begin{oneparchoices}
\choice 苹果
\choice 牛肉
\choice 薯条
\choice 炸鸡
\end{oneparchoices}
```

让学生勾选也可以，我个人建议采用填空方式方便阅卷，所以上面的选择题都没有在选项条目那里设置正确答案。

```
\question
下面哪种食品有利于健康？

\begin{checkboxes}
\Correctchoice 苹果 %这条是正确选项，对上面的选择题例子也适用
\Correctchoice 牛肉 %这条是正确选项
\choice 薯条
\choice 炸鸡
\end{checkboxes}
```

或者选项在同一行

```
\question
下面哪种食品有利于健康？

\begin{oneparcheckboxes}
\Correctchoice 苹果
\Correctchoice 牛肉
\choice 薯条
\choice 炸鸡
\end{oneparcheckboxes}
```

exam 类还包含了计分表，因为不使用就没有研究。设置语句还有很多，有需求的时候可以查阅手册。

5.6 其他一些不务正业的包

还有一些非常优秀的非理工科专业的包。

musixtex 五线谱包
mtx 五线谱和歌词，更简单一些
xpiano 钢琴键盘
horoscop 占星术
skak 国际象棋包
psgo 围棋包，基于 pstricks
sudoku 数独
sudokubundle 数独
qrcode 生成二维码

第六章　化学分子式 chemfig

chemfig 包是基于 tikz 的，单纯使用 chemfig 时，加载 chemfig 包并不需要额外加载 tikz 包。如果导言中还加载了 tikz 包，加载 chemfig 包时也不会产生错误。

6.1　基本设置

加载包后可以修改一下基本设置，这是纯属个人感观。chemfig 的设置方式随着版本有较大变化，可能有些老的设置语句会逐渐退出。基本设置写在导言区比较好，但是可以临时修改基本设置。

```
\usepackage{chemfig}

%设置键长
\setchemfig{atom sep=2.5em}
%设置键与元素的间隔，default 是 2pt
\setchemfig{bond offset=2pt}
%设置字符大小
\small
%设置键之间的宽度，指双键、三键间宽度
\setchemfig{double bond sep=0.25em}
%设置键的线宽，还可以设置颜色，默认黑色
\setchemfig{bond style={line width=0.8pt}}
%设置楔形角度，后面的例子用到
\setchemfig{cram width=5pt, cram dash width=1pt, cram dash sep=2pt}
```

chemfig 可以直接在 article、book 和 beamer 中使用，如果需要在 standalone 类中使用，可选择类参数为 tikz 或 chemfig，这两个参数得到的结果是不同的。

```
\documentclass[tikz]{standalone} %每一个 chemfig 一页
```

上课做练习用的是 tikz 选项，每个分子式一页。**tikz 选项对高分子不适用，必须使用 chemfig 选项**

```
\documentclass[chemfig]{standalone} %所有的 chemfig 都在一页
```

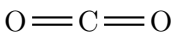
如果希望一张图中画多个分子式，但是图片单独一页，可以采用这种方式。高分子式，无论一个还是多个，可以采用 chemfig 选项获得，不能选择 tikz 选项，tikz 选项输出不正确。更复杂一点的，不仅仅是分子式本身的，比如表示反应或动力学过程，也需要 chemfig 选项。

6.2 键的基本语法

化学键

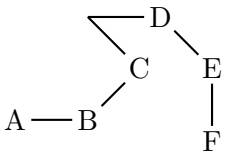
\chemfig{}如行间公式，默认是文本。连字符是单键，等号是双键

```
%等号是双键
\chemfig{O=C=O}
```



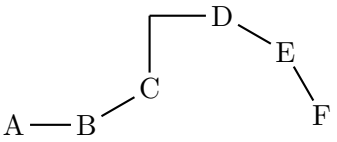
```
%波浪号是三键
\chemfig{H-C~C-H}
```

```
%设定了 8 个方位，0-7，0 是默认可以不写
\chemfig{A-B-[1]C-[3]-D-[7]E-[6]F}
```



增量可以修改，如果设置为 30 度，就是 360/30=12，0～11 个方位，一般不推荐用此种方式。

```
\setchemfig{angle increment=30}
\chemfig{A-B-[1]C-[3]-D-[11]E-[10]F}
```



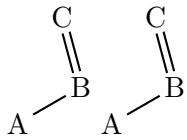
默认值是 45 度。

```
%这里需要改回默认
\setchemfig{angle increment=45}
```

键的第一个参数：键角

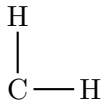
键的第一个参数是角度，如果不使用默认方位，直接写角度就可以，角度范围是 $[-360, 360]$。度数前单冒号是绝对角度，双冒号是相对角度。下面的例子结果是一样的。

```
\chemfig{A-[:30]B=[:105]C}
\chemfig{A-[:30]B=[::75]C}
```



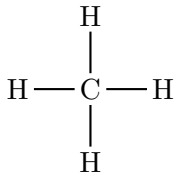
上面的例子都是单链结构，分支用圆括号，表示并在一个原子上的

```
%圆括号里是并在前面原子上的键
\chemfig{C(-[2]H)-H}
```



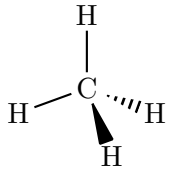
一个原子上可以并多个支链，甲烷分子一个 C 后并了三个括号，表示除了 0 位上的 H，2、4、6 位各有一个单键连着 H 原子。

```
\chemfig{C(-[2]H)(-[4]H)(-[6]H)-H}
```



立体分子甲烷，C 上并了三个单键，加上原来的一个，共四个键。上面的楔形参数定义的是楔形的形状，< 纸面前，<: 纸面后

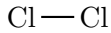
```
\chemfig{C(-[:90]H)(-[:200]H)(<[:290]H)<:[:340]H}
```



第二个键参数：键长

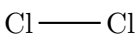
如果是两个字母的原子，原子间距固定使得键长太短，如下

```
\chemfig{Cl-Cl}
```



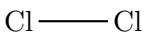
设置键长为固定值，旧版本的`\chemfig*{}`不再适用。

```
% 花括号表示局域有效
{
\setchemfig{fixed length=true}
\chemfig{Cl-Cl}
}
```



键长可以任意设定，设定方式是默认长度的倍数。如果需要一个长键，设置键的第二个参数，设定方式为默认长度的倍数。键的第一个参数是角度，所以设置键长时如果不需要设定角度，前面的逗号不能省略。

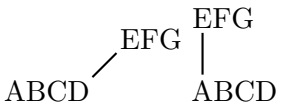
```
\chemfig{Cl-[,1.5]Cl}
```



键的第三和第四个参数：连接的前后分子序数

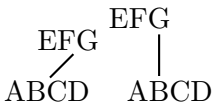
键的第 3 个和第 4 个参数是连接的分子序数，此时不设定键长时，逗号不能省略。下面的例子，ABCD 的第二个分子是 B，EFG 第三个分子是 G。不设定连接的分子，默认连接分子会随键的方向变化。

```
\chemfig{ABCD-[1]EFG}
\chemfig{ABCD-[2]EFG}
```



设定 B 连接到 G 时为

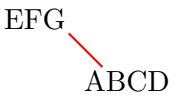
```
\chemfig{ABCD-[1,,2,3]EFG}
\chemfig{ABCD-[2,,2,3]EFG}
```



键的第五个参数：其他 tikz 参数

可以是键的颜色，线宽，线型。

```
\chemfig{ABCD-[3,1.2,2,3,red]EFG}
```



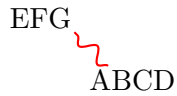
复杂分子讨论动力学时这个功能很有用，可以标记重点讨论的键为不同颜色。

还可以利用 tikz 的修饰库做蛇形键，此处要加入 tikz 的子库

```
\usetikzlibrary{decorations.pathmorphing}
```

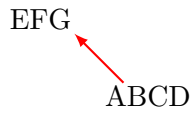
然后将键定义为修饰库里的蛇形

```
\chemfig{ABCD-[3,1.2,2,3,red,
decorate,decoration=snake]EFG}
```



还可以画箭头

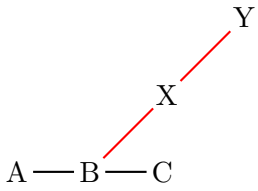
```
\chemfig{ABCD-[3,1.5,2,3,red,-latex]EFG}
```



键的进阶

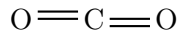
键的参数如果写到键的前面，表示对整个分支都有效

```
\chemfig{A-B([1,1.5,,,red]-X-Y)-C}
```



可以用上下标将键上下平移

```
\chemfig{O=^C=_O}
```

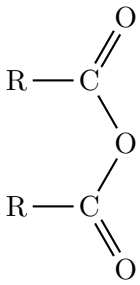


6.3 分子结构

旋转

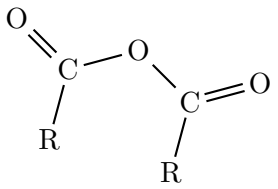
下面是酸酐的例子

```
\chemfig{R-C(=[::60]O)-[::-60]O-[::-60]C(=[::60]O)-[::-60]R}
```



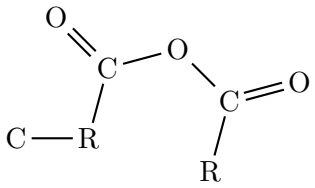
分子最前面加角度，整体旋转后面的分子

```
\chemfig{[:75]R-C(=[::60]O)-[::-60]O-[::-60]C(=[::60]O)-[::-60]R}
```



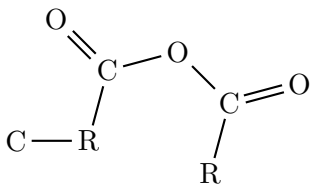
将整体旋转的分子连接到其他分子上，需要一个不存在的分子，第一个例子是将分子 C 设置为白色，不显示

```
\chemfig{C-{\color{white}C}([:75]R-C(=[::60]O)-[::-60]O-[::-60]C
(=[::60]O)-[::-60]R)}
```



也可以使用幻影原子

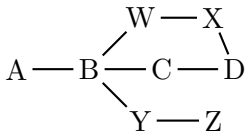
```
\chemfig{C-\phantom{C}([:75]R-C(=[::60]O)-[::-60]O-[::-60]
C(=[::60]O)-[::-60]R)}
```



问号

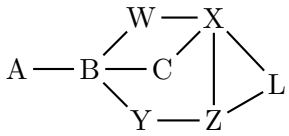
问号表示自动连接两个分子，可以自动成环，X 和 D 自动生成键的连接，键长自动计算出来。

```
\chemfig{A-B(-[1]W-X?)(-[7]Y-Z)-C-D?}
```



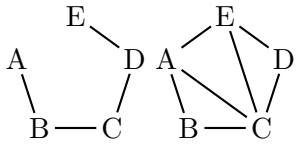
可以是多个分子自动连接

```
\chemfig{A-B(-[1]W-X?)(-[7]Y-Z?-[:30]L?)-C?}
```



问号可以设置编号连接分子，即多个问号。下面的例子，A 设置了问号 a，C 连接到 A，C 和 E 同时设置了问号 a,b，E 连接到 A 和 C。可以比较一下没有问号时候的分子。

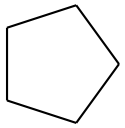
```
\chemfig{A-[:-72]B-C-[:72]D-[:144]E}
\chemfig{A?[a]-[:-72]B-C?[a]?[b]-[:72]D-[:144]E?[a]?[b]}
```



环

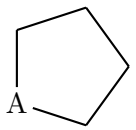
乘号表示环

```
\chemfig{*5(-----)}
```



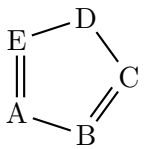
从 A 开始的环，环是等键长的

```
\chemfig{A*5(-----)}
```



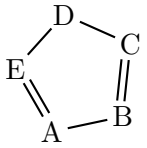
环可以是不同的键构成的

```
\chemfig{A*5(-B=C-D-E=)}
```



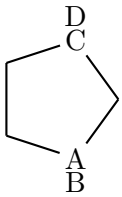
A 前加角度可旋转该环

```
\chemfig{[:30]A*5(-B=C-D-E=)}
```



有时候需要上下加分子，加\vskip2mm是因为这个时候会产生按 A 和 C 构成 box 的问题，导致上下间距不足。

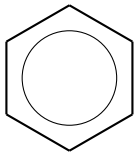
```
\vskip2mm\chemfig{*5(-\chembelow{A}{B}--\chemabove{C}{D}--)}
```



苯环

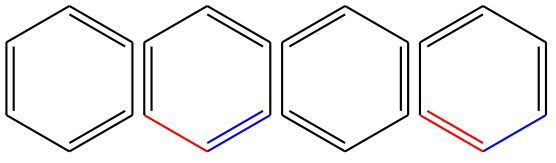
苯环有两种方式表示，一种是中间为圆环

```
\chemfig{**6(------)}
```



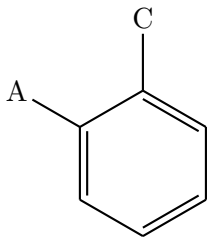
另一种是单双键形式，键按逆时针顺序，第一个键是左下负 30 度方向

```
\chemfig{*6(-=-=-=)}
\chemfig{*6(-[,,,,red]=[,,,,blue]-=-=)}
\chemfig{*6(=-=-=-)}
\chemfig{*6(=[,,,,red]-[,,,,blue]=-=-)}
```



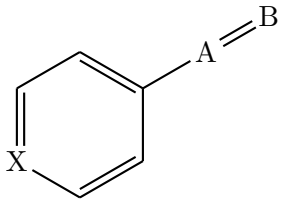
苯环连接到 A 分子，A 的负 30 度方向单键连接到苯环，苯环的第一个键为双键

```
\chemfig{A-[:-30]*6(=-=-=(-C)-)}
```



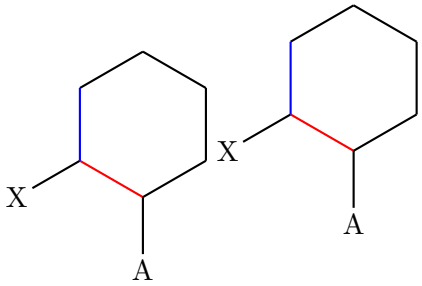
如果分子在苯环内，分子直接写在乘号前

```
\chemfig{X*6(-=-(-A=B)=-=)}
```



在环上连原子时，红色起始键后端连 A，用圆括号并在红色键后。如果想在左下红色键前端连 X，等价于蓝色键后端连 X，似乎应在蓝色键后端并圆括号，但是实际上要在红色键前并圆括号。

```
\chemfig{*6((-X)-[,,,,red](-[6]A)-----[,,,,blue])}
\chemfig{X-[:30]*6(-[,,,,red](-[6]A)-----[,,,,blue])}
```



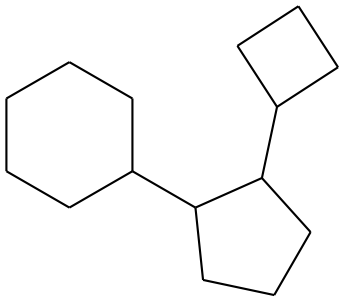
上面的两种写法效果一样，但是图片位置其实是不同的。第一个起笔在红蓝色键交点，第二个起笔在 X。

很多时候，特别是复杂分子时，需要先画出环，定下环结构的方位，第二个例子需要计算 X 连向环的键的键角才能把环放得跟第一个例子一样。虽然第二个例子看上去结构上更简单明了，和手册例子接近，但是其实需要额外的计算量。有些时候是不方便计算键角的，所以第一个例子才是常用的情况。

并环

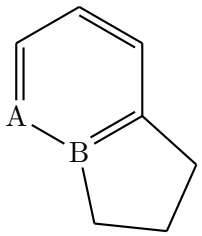
连接是写分支，并环是共享环中的某个键。环通过键连接在一起，环内的键个数是一致的。

```
\chemfig{*6(--(-*5(----(-*4(----))-))----)}
```



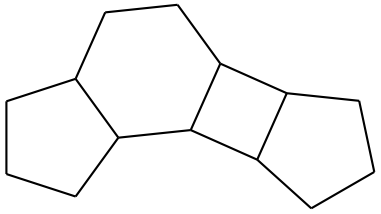
并环，5 不用写共享那个双键，即不共享时 B 后面那个双键，五环只写 4 个键

```
\chemfig{A*6(-B*5(----)=-=-=)}
```



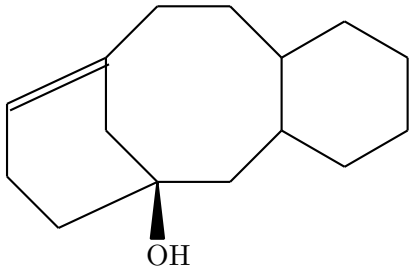
多次并环的一个例子

```
\chemfig{*5(--*6(-*4(-*5(----)--)----)---)}
```



上面的并环都是正多边形的，如果不是正多边形结构，并环就需要设置问号、短键和幻影原子来协助完成。

```
\chemfig{*8(-(-[:-90,0.1]\phantom{C}?[b])(<[:-90]OH)--*6(-----)
----(=^[:205,1.5]-[:-90]-[:-45]-[:25,1.5]?[b])-)}
```



6.4 结构复用

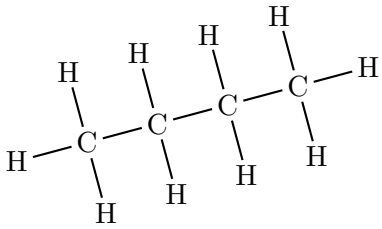
定义 submol 可以复用结构

```
\definesubmol{xy}{CH_2} %定义复用单元
\chemfig{H_3C-!{xy}-!{xy}-!{xy}-CH_3}
```

$H_3C — CH_2 — CH_2 — CH_2 — CH_3$

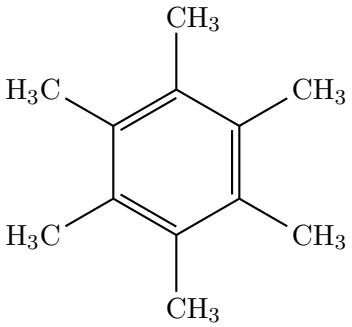
定义替代结构，跟上面本质相同，写法不同，增加可读性

```
\definesubmol\xx{C(-[::90]H)(-[::-90]H)}
\chemfig{[:15]H-!\xx-!\xx-!\xx-!\xx-H}
```



可以方括号定义键从右到分子时的代码，花括号定义键从左到分子时的代码，使用时自动选择

```
\definesubmol\zz[H_3C]{CH_3}
\chemfig{*6((-!\zz)=(-!\zz)-(-!\zz)=(-!\zz)-(-!\zz)=(-!\zz)-)}
```

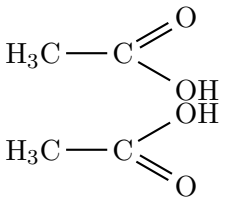


\definesubmol语句还可以用于将大分子拆成较小的分子来画，方便构建结构、查找错误。

6.5 对称

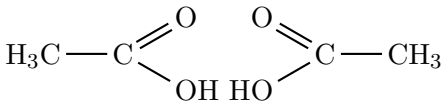
\vflipnext得到上下对称，中间空行是为了更方便地看两个分子是上下对称的。

```
\chemfig{H_3C-C(=[:30]O)-[:-30]OH}

\vflipnext
\chemfig{H_3C-C(=[:30]O)-[:-30]OH}
```



\hflipnext得到左右对称，中间不空行，方便看左右对称。

```
\chemfig{H_3C-C(=[:30]O)-[:-30]OH}
\hflipnext
\chemfig{H_3C-C(=[:30]O)-[:-30]OH}
```



6.6 Lewis 形式

新版本废弃了 lewis，改用宏 charge，lewis 形式在 1.6a 版本中已经不可用。

```
\Charge{45=\:,135=\.,225=\",-45=\|}{X}
```

语法规则为：\Charge{[参数]电荷}{原子}，注意写入参数的方括号在花括号里面。

- \.代表不成键的单个电子，单点
- \:代表不成键的孤对电子，双点
- \"代表，空心细长矩形
- \|代表，单线

参数写到前面对所有的电荷都适用，如下单电子和双电子的点都变大且为蓝色

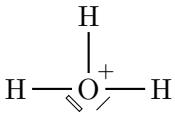
```
\Charge{[.radius=1.5pt, .style={draw=none,
fill=blue}]45=\:, 135=\., 225=\"}{X}
```

参数写到后面则是对当前电荷有效。如下，点可以设置为不同颜色。

```
\Charge{45=\:[{.radius=1pt,.style={draw=none,fill=red}}],
135=\.,225=\"[{"width=3pt,"style={fill=green}}]}{X}
```

下面是水合氢离子的例子。

```
\chemfig{H-\charge{45:1.5pt=$\scriptstyle+$,-45=\|,-135=\"}{O}
(-[2]H)-H}
```



\Charge{}和\charge{}有区别，大写会覆盖使得键长变短。推荐小写。

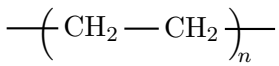
6.7 高分子

高分子是通过定义宏方式得到方括号或者圆括号，@ 表示宏，后面花括号里第一个参数是名字，起名是任意的。花括号里第二个参数是位置，按前

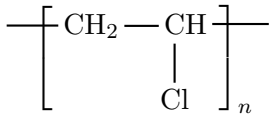
面那个键的长度比例定义括号的位置。第二个语句是定义括号的高度、深度和脚标。高度是距离前面键（水平位置）的高度。

必须编译两次才能正确显示高分子。在 standalone 类中输出高分子时，必须选择类参数为 chemfig，需要手动扩大图片留白才能不切边。

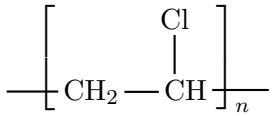
```
\chemfig{\vphantom{CH_2}-[@{left,0.75}]CH_2-CH_2-[@{right,0.25}]}
\polymerdelim[height=5pt, indice=\!\!n]{left}{right}
\vspace{3mm}
```



```
\chemfig{\vphantom{CH_2}-[@{left,0.7}]CH_2-CH(-[6,,1]Cl)-[@{right,
0.3}]}
\polymerdelim[delimiters={[]},height=5pt, depth=30pt,
indice=n]{left}{right}
```



```
\chemfig{\vphantom{CH_2}-[@{aa,0.75}]CH_2-CH(-[2,,1]Cl)-[@{bb,
0.25}]}
\polymerdelim[delimiters={[]},height=30pt, depth=5pt,
indice=n]{aa}{bb}
```



书上还有括号位置不对称的更复杂的例子，需要的时候可以自学。

6.8 一些细节

opus 的位置

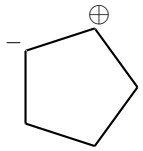
```
\chemfig{A^\oplus-[2,,1]B}
```

实际上希望键连在 A 上，因此代码调整为

```
\chemfig{A\rlap{$^\oplus$}-[2]B}
```

环上时需要设置一个不显示的短键

```
\chemfig{*5(---(-[,0.2,,,white]\oplus)-(-[,0.2,,,white]
\scriptstyle{-})-)}
```



6.9 与 tikz 混用

chemfig{}语句是文本，所以可以在 tikz 图片环境中用 node 语句将分子式加载进来，然后做图。

chemfig 里定义了大量的语句做复杂的分子式反应图，如果单独只学习了 chemfig，可以认真学习手册里的内容。但是如果 tikz 用得很熟练，也可以采用 node 方式做图。手册里的一些例子明显是 tikz 混用方式更容易和简洁，且不用记忆大量语句。**有些例子注意要严格按照手册代码书写，不能缺失空格。有些例子不能修改默认字体大小，否则会匹配不上。**

这两种方案各有优点。chemfig 代码的好处是：严格对齐，可以做到极致优美。tikz 的 node 方案的好处是：灵活性和扩展性强。

另外，希望产生单独的一张分子式图文混排的结果，也可以设置 standalone 类的选项为 minipage，这种方式对高分子也适用。

这里给出一个 tikzpicture 里加载高分子的例子，高分子是比较麻烦且容易出错的。

```
\documentclass[tikz]{standalone} %这里是 tikz，因为是 tikzpicture 环境
\standaloneconfig{border=30pt} %有时需要手动设置留白

\usepackage{graphicx}
\usepackage{chemfig}

\begin{document}

\begin{tikzpicture}
```

```
%node 里一定要同时加载高分子的那两句，不能只加载一句，顺序不能颠倒
\node at (0,0) {
\chemfig{\vphantom{CH_2}-[@{aa,0.75}]CH_2-CH(-[2,,1]Cl)-[@{bb
,0.25}]}
\polymerdelim[delimiters={[]},height=5pt, indice=n]{aa}{bb}
};

\end{tikzpicture}

\end{document}
```

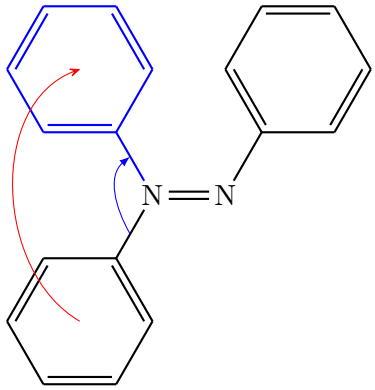
6.10 chemmove

chemfig 包与 tikz 包是深度融合的，依赖的就是\chemmove，其语法为

```
\chemmove[选项]{tikz语句，可以是多条语句}
```

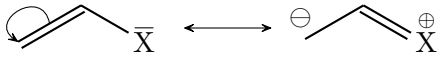
chemfig 为每个环结构（包括苯环）自动定义的中心点的 node 名，可以直接使用，下面的 azobenzene 有两个苯环，因此有两个中心点。如高分子的例子，chemfig 还允许为每个键设置宏，宏名也可以作为坐标。宏写在所有键参数前，不加逗号，默认位置是键的中间位置，即 0.5 的位置。位置取值范围是 $[0,1]$ 区间。利用这些默认的定义，可以很方便表达动力学过程。

```
\chemfig{N(-[@{b1,0.5}:-120]*6(-=-=-=))([,,,,blue]-[@{b2,0.5}:120]
*6(-=-=-=))=N-[:60]*6(-=-=-=)}

\chemmove{
\draw [-latex] [blue] (b1) to [out=120,in=220] (b2);
\draw [red] (cyclecenter1) to [out=150,in=200](cyclecenter2);
}
```



化学反应可以用\schemestart和\schemestop确定起始，然后再画表示动力学的箭头。chemfig 还定义了一些箭头形式用来表示化学反应。下面的例子，双键前先定义宏，位置即在双键起点，双键参数里定义宏，默认位置在双键中间位置。双键本身位移，所以第二个宏的起点看好在双键的一个键上。用 controls 方式画箭头，可以得到更好的表现。

```
\schemestart
\chemfig{@{a1}=_[@{db}::30]-[::-60]\charge{90=\|}{X}}
\arrow{<->}
\chemfig{\chemabove{\vphantom{X}}{\ominus}-[::30]=_[::-60]
\chemabove{X}{\scriptstyle\oplus}}
\schemestop
\chemmove{\draw(db) .. controls +(100:5mm) and +(145:5mm).. (a1);}
```



chemfig 还为分子默认设置了许多类似 node 坐标的位置点，可以使得化学反应式完美对齐，请参考 chemfig 手册 51 页–65 页的例子，这些例子还包括了箭头命令\arrow的各种参数。

手册上还有一些例子个人觉得直接用 tikz 画更方便。

第七章　演示稿 beamer

beamer 类使用起来不难，自定义模板比较麻烦一点。

7.1　beamer 模板

beamer 将一页 ppt 分为了五个部分，这五个部分一般不会同时存在。上面可以有 headline，下面可以有 footline，左边可以有左边栏 sidebar left，右边可以有右边栏 sidebar right，中间的展示内容为 frame 环境。

frame 又包含两部分内容，frametitle 和 frame 环境里的内容。

使用 beamer 类，beamer 类的最常用选项为字体。

```
\documentclass[12pt]{beamer} %8,9,10,11,12,14,17,20
```

默认字体为 10pt。这里的 10pt、11pt 和 12pt 不是 article 的字体大小，已经是相对应扩大的结果。12pt 是一套很大的字体，很适合大教室讲课。10pt 和 11pt 可以放置更多内容，比较适合学术类讲座。

beamer 类的另一个最常用选项是比例，默认是 43，4:3，过去的投影仪一般是这个比例。新的投影仪是 16:9，使用 aspectratio 参数可以修改比例为 16:9，但是建议设置为 16:10。天津大学教室里的投影仪有时调整的不好，16:9 可能会导致内容在白屏外，16:10 保险一点。其余比例可以查看手册。还有就是学生的平板有可能是 4:3 的，因此默认的 43 比例目前仍然是最合适的比例。笔者经常用 beamer 做笔记，一般情况下选用 43 比例，内容放置上比较充足一些。如果笔记内容有很长的公式，16:10 比较好。从使用的角度，纵向比例太小，单页内容就会受限。

```
\documentclass[12pt,aspectratio=1610]{beamer} %默认 43
```

beamer 的默认模板是 default，不设置任何模板就是 default 模板，没有 headline、footline 和左右的 sidebar，只有 frame 环境。即使默认模板什么都没有，在右下角会自动生成一套灰色的导航条按钮，有翻页和跳转等功能。这个导航条不会影响内容放置，也即内容可以覆盖在这个灰色导航条上

面。而且这个灰色导航条上面即使覆盖了内容，也不影响点击跳转功能。

beamer 将模板、模板配色和字体设置分开了，可以分别在导言中添加模板和模板配色，配置字体。

最常用的两个模板是 Warsaw 和 Berkeley。

```
\usetheme{Warsaw} %headline 和 footline
```

Warsaw 模板设置了 headline 和 footline。headline 和 footline 都是两个等分的 box，headline 里显示当前页所在的小节和小小节，footline 里显示作者和演示稿名称。

Berkeley 模板设置了左边栏，左边栏内显示题目、作者、当前页所在的小节和小小节。

```
\usetheme{Berkeley} %左边栏
```

个人认为 Berkeley 模板比较适合 16:10，Berkeley 的 frametitle 占比太大了，从美学角度可能这样的比例更适合，但是不利于更多的内容展示。Warsaw 模板适合 4:3，headline 和 footline 都是极窄的，占用空间不是很大。笔者最喜欢 default，感觉没有必要让导航或其他如 title 这样的挤占展示空间。而且只要 tex 文件有\section，编译生成的 pdf 就有目录书签，不需要额外添加目录导航也可以用 pdf 的书签跳转。

模板内置了一套设置，如果觉得可以接受，就直接使用模板，这样代码量很小。

上面这两个模板的默认配色是深蓝色系，推荐保留默认配色。

设置模板配色的语句为：

```
\usecolortheme{seahorse}
```

seahorse 是浅蓝色系，笔者认为用在 Berkely 模板不好看。beamer 默认了一系列颜色主题，个人觉得好看的不多。beamer 手册有各种模板和配色的展示。

beamer 的字体配置模板为 fonttheme，默认字体不大好看，serif 是一个比较好的选择。

```
\usefonttheme{serif}
```

7.2 frame

每一个 frame 环境是一页 ppt，beamer 的基本框架为：

```
\documentclass{beamer}
\begin{document}
\begin{frame}{填写frametitle} %frametitle 可以省略
内容
\end{frame}
\end{document}
```

也可以采用 frametitle 命令来设置标题

```
\begin{frame}
\frametitle{当前页的标题} %等价于上一个例子
...
\end{frame}
```

可以设置子标题，但是必须 frametitle 存在时才会显示子标题。

```
\begin{frame}
\frametitle{当前页的标题}
\framesubtitle{当前页的子标题} %如果没有 frametitle 不生效
...
\end{frame}
```

frame 的参数

frame 有几个比较常用的参数

1. 内容在顶部，默认是上下居中的

   ```
   \begin{frame}[t]{}
   ...
   \end{frame}
   ```

2. allowframebreak：内容太长自动分页

   ```
   \begin{frame}[allowframebreak]{}
   ...
   \end{frame}
   ```

   这个功能并不完美，经常会遇到分页时希望一页塞得很满，一页相对比较空的情况。所以一般笔者都是手动调整内容方式分页。

3. shrink 自动缩小内容匹配尺寸，手册称 shrink 参数 **evil**，不推荐，比较难看。类似的还有 squeeze 参数，竖直压缩间距，如果内容太长仍

然会超出范围，也不是很好用。基本上还是需要手动调整内容的。

```
\begin{frame}[shrink]{}
内容
\end{frame}
```

4. **fragile 参数，lstlisting 环境必须使用该参数，否则会产生严重编译错误**

```
\begin{frame}[fragile]{}
\begin{lstlisting} %必须添加 fragile 参数
code
\end{lstlisting}
\end{frame}
```

在 frame 环境内，可以使用百分号注释掉单行语句，但是不能使用\iffalse...\fi注释掉多行语句。可以使用它注释掉一个或多个 frame 环境。

7.3 beamer 自定义的分栏环境

beamer 类有一个自定义的分栏环境非常好用，个人觉得，要是能独立出来构成一个包就好了。

```
\begin{frame}{}
\begin{columns}[t] %多栏上下对齐方式，默认居中
\begin{column}{8cm} %分栏，花括号里定义宽度
内容
\end{column}
\begin{column}{0.33\textwidth} %宽度的另外一种定义方式
\end{column}
\end{columns}
\end{frame}
```

如果需要更多分栏，就多嵌套几个 column 环境即可。**所有分栏宽度加起来不能超过文字宽度**。上面例子里的系数 0.33 是按默认页面比例、default 模板情况下设定的，和第一分栏的 8cm 宽度加起来，在文字左右边距各 4mm 情况下，不超过文字宽度。

7.4 目录、暂停、显示顺序和跳转

目录

beamer 里可以加载目录，一般用单独一页 frame 加载目录。

```
\begin{frame}{目录}
\tableofcontents
\end{frame}
```

很多时候我们希望逐条展示目录，可以在后面加暂停选项 pausesections。

```
\tableofcontents[pausesections]
```

每一小节前，还有可能以突出当前目录的形式展示目录，可以添加选项 currentsection

```
\tableofcontents[currentsection]
```

如果目录条目很多，会自动压缩纵向间距，是目录尽量在一页中显示。但是如果目录只有几条，目录的默认间隔很大，并不美观。网上修改beamer@sectionintoc的方法并不起作用，beamerbasetoc.sty 文件里修改也不成功。其实很早就发现，最简单的办法是加一个\parbox，将\tableofcontents语句放进去，间距直接变成最小。然后再使用 setspace 包的 spacing 环境调整间距，设置行距。这个方法语句都是最常用的语句，不增加额外的记忆。

```
\parbox{\textwidth}{
\begin{spacing}{1.5}
\tableofcontents
\end{spacing}
}
```

默认目录不带小节号，可以用下面的语句设置

```
\setbeamertemplate{section in toc}[sections numbered]
\setbeamertemplate{subsection in toc}[subsections numbered]
```

暂停

如果希望在某处暂停，最直接的方式是加\pause语句：

```
\begin{itemize}
```

```
\item 首先\pause
\item 其次\pause
\item 最后\pause
\end{itemize}
```

经常会遇到列表时希望逐条显示，每个 item 后加语句太麻烦，可以采用下面的方式：

```
\begin{itemize}[<+->]
\item 首先
\item 其次
\item 最后
\end{itemize}
```

如果一页中有多个 itemize 环境，可以将选项加到 frame 那里，每个 itemize 环境都会有逐条显示功能。

```
\begin{frame}[<+->]{直接加到frame选项}
\begin{itemize}
\item A
\item B
\end{itemize}
\begin{itemize}
\item A1
\item B1
\end{itemize}
\end{frame}
```

显示规则

加减号只能按顺序逐条显示，可以在\item后面加显示规则，注意显示规则加在尖括号里。

```
\begin{frame}{显示规则}
\begin{itemize}
\item<1-> 首先 %n-表示从 n 往后都显示
\item<2-> 其次
\item<3-> 最后
\end{itemize}  %这个例子的结果与上面的按顺序显示相同

\begin{itemize}
```

```
\item<1-> 第一
\item<2-> 第二
\item<-3> 第三 %-n 表示从 1 到 n 后显示
\item<2,4> 第四 %2 和 4 次显示，其余不显示
\end{itemize}
\end{frame}
```

重点和要点

beamer 默认定义了重点\alert为红色，要点\structure为蓝色。

```
\alert{重点}
\structure{要点}
```

在逐条显示中也可以加入 alert，突出当前条目

```
\begin{frame}{alert}
\begin{itemize}
\item<1-|alert@1> 首先
\item<2-|alert@2> 其次
\item<3-|alert@3> 最后
\end{itemize}
\end{frame}
```

这种情况突出的条目是 alert 的默认红色，希望改换不同的颜色，可以用下面的语句定义两个颜色，突出时的和不突出时的，也可以加粗。加粗的例子对 pdflatex 有效，xelatex 时依赖于设置语句，有时需要改为设置中文黑体。还可以设置 colorbox 等方式来突出条目。

```
\begin{itemize}
\item<1->\alt<1>{\color{red}首先}{\color{gray!30}首先}
\item<2->\alt<2>{\bf\color{purple}其次}{\color{red!30}其次}
\item<3->\alt<3>{\colorbox{blue!30}{最后}}{\color{gray!20}最后}
\end{itemize}
```

跳转

beamer 里可以设置调整按钮，下面的例子是跳转到当前页第二次显示。

```
\begin{itemize}[<+->][label=here]{逐条显示}
\item 首先
```

```
\item 其次
\item 最后
\end{itemize}
\hyperlink{jumptosecond}{\beamergotobutton{Jump to second item}}
\hypertarget<2>{jumptosecond}{}
```

必须另外的方括号里写 label，不能使用同一个方括号，这个 label 是下面的例子要用的，这个例子本身并不需要。

还可以往前调整到某 frame 那里，这个 frame 需要设置好 [label= 名字]，**注意上面的例子中必须是两个方括号分别写入参数**，不能合并。

```
\begin{itemize}[<+->, label=here]{逐条显示} %错误的写法
\begin{itemize}[<+->][label=here]{逐条显示} %正确的写法
```

在其他的 frame 里设置按钮，可以跳转到“逐条显示”这页，如果是上面的逐条显示的某页，还可以直接设置到某页某次显示，下面的例子就是跳转到第二次显示。

```
\hyperlink{here<2>}{\beamerbutton{Jump to other frame}}
```

7.5 图片

beamer 里加载图片直接用`\includegraphics[]{}`语句，这一小节主要讲图片的一些特殊情况。

局部放大

对图片进行局部放大

```
\begin{frame}
\framezoom<1><2>[border](0cm,0cm)(2cm,1.5cm)
\framezoom<1><3>[border](1cm,3cm)(2cm,1.5cm)
\framezoom<1><4>[border](3cm,2cm)(3cm,2cm)
\includegraphics[height=5cm]{fig/tju.pdf}
\end{frame}
```

这里定义了三个 framezoom，对图片天津大学校徽进行局部放大，border 参数表示显示放大区域的边框，后面的坐标表示局域放大区域的对角线坐标。最后的效果是，展示整个校规，然后每翻页一次，依次放大图片的不同区域。这个功能做学术报告时非常有用。

上面这个例子不能直接加 frametitle，否则展示局域放大图片时 title 的存在会影响展示效果，用下面的语句，使得第二次展示，也就是第一次局域放大时，title 被取消掉。

```
\begin{frame}<1>[label=tjuzoom]
\frametitle<1>{title}
\framezoom<1><2>[border](0cm,0cm)(1cm,1.5cm)
\framezoom<1><3>[border](1cm,3cm)(2cm,1.5cm)
\framezoom<1><4>[border](3cm,2cm)(3cm,2cm)
\pgfimage[height=5cm]{tju.pdf} %另一种加载图片的方式
\end{frame}
\againframe<2->[plain]{tjuzoom}
```

多图覆盖

在相同位置展示多张图片，后面的图片覆盖前面的图片，用\llap{\includegraphics<>[]{}}，尖括号内设定显示规则，方括号内设置图片宽高和角度等，花括号内设置图片名字。

```
\centering
\includegraphics<1->[width=6cm]{tju.pdf} %第一句不加 llap
\llap{\includegraphics<2>[width=9cm]{2.jpg}}
\llap{\includegraphics<3->[width=3cm]{d-2.jpg}}
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
\end{frame}
```

\llap语句有一个缺点，如果图片宽高一致，效果还行，如果不一致，后面的图片将以右下角对齐第一张图片的方式覆盖显示。类似的语句都是相同的缺点。

可以使用 xmpmulti 包加载多图，

```
\usepackage{xmpmulti}
```

假设图片名为 d-0、d-1 和 d-2，此时语句比较简单：

```
\begin{frame}{使用xmpmulti包}
\center
\multiinclude[graphics={width=7cm},format=jpg]{d}
\end{frame}
```

这样图片将以相同宽度展示，竖直方向的位置是底边对齐。multiinclude 还

支持选择起始编号，对一组排序图片进行多图展示。

使用下面的语句，覆盖时前一张图片不显示：

```
\multiinclude[<+>][graphics={width=7cm},format=jpg]{d}
```

使用 xmpmulti 包对于宽高比不同的图片，尤其是比例差别极大的情况，仍然无法满足展示需求，最好使用 tikz 的 node 来完成多图展示，必须加载 tikz 包。

```
\begin{frame}{}
\center
\begin{tikzpicture}
\node at (0,0) {\includegraphics<1->[width=7cm]{tju.pdf}};
\node at (0,0) {\includegraphics<2>[width=6cm]{2.jpg}};
\node at (0,0) {\includegraphics<3->[width=6cm]{d-2.jpg}};
\end{tikzpicture}
\end{frame}
```

这样图片的中心在一个位置上，展示效果是最好的。同理，尖括号控制显示规则。这种方式也可以精确控制多图的显示位置，达到最好的显示效果。

beamer 内可以使用动画包，上面讲的并不是动画，而是逐次展示，这两者之间是不同的。

7.6 简单的设置语句

文字对齐方式

两边对齐的设置推荐使用 ragged2e 包，中英文混排效果更好

```
\usepackage{ragged2e}
\justifying\let\raggedright\justifying
```

设置字体

字体模板的加载前面已经说过了，beamer 里设置字体的语句是\setbeamerfont{beamer-font name}{字体}，beamer-font name 是 beamer 已经定义好的字体名。

default 模板最好将 frametitle 设置为加粗的，CJKutf8 加载中文，\bfseries和\bf对汉语是生效的。

```
\setbeamerfont{frametitle}{series=\bfseries}
```

还可以同时设置大小

```
\setbeamerfont{frametitle}{size=\large,series=\bfseries}
```

如果是 xelatex 编译，则需要定义汉语字体，比如设置汉语字体为黑体，或者设置汉语的 bf 为黑体，依赖于具体的字体设置。

```
\setbeamerfont{frametitle}{series=\bfseries\heiti}
```

设置 itemize 和 enumerate 的字体，两个参数分别是字体大小和 baseline 的大小。

```
\setbeamerfont{itemize/enumerate body}{size*={14pt}{20pt}}
```

尺寸

beamer 中定义尺寸的语句是\setbeamersize{} 文字宽度，default 下可以设置为 4mm

```
\setbeamersize{text margin left=4mm, text margin right=4mm}
```

设置左右边栏宽度，如果自定义模板，必须设置左右边栏宽度，然后设置的模板才有效。

```
\setbeamersize{sidebar width left=1.5cm} %right 是右边栏
```

设置 beamer 颜色

beamer 颜色是一对颜色，前景色和背景色。

比如定义演示稿 title 的颜色，得到的是 title 变成了 colorbox 形式。

```
\setbeamercolor{title}{bg=cyan!20,fg=black}
```

设置每页演示的题目，这里要用 titlelike，这样对 framesubtitle 也有效

```
\setbeamercolor{titlelike}{fg=white,bg=blue!70!black}
```

单独设置某项的颜色并不好看，最好是设计模板时使用这些命令。或者在采用内置的模板和配色时，局部修正某项为自己喜欢的颜色，这样更美观一些。

设置 beamer 模板

设置 beamer 模板的语句为\setbeamertemplate{模板名}{内容}，这个语句是模板设计的基础，这一小节先学习插入页码和一些简单设计。

在 footline 插入页码，\insert...是 beamer 内置的导航条，可以插入页码、作者、单位、题目、小节名、小小节名等等。

```
\setbeamertemplate{footline}{\tiny\insertpagenumber}
```

在 default 模板，设置在 footline 插入页码，会使得自动生成的灰色导航条上移。目前笔者更喜欢在左侧边栏插入页码。

在左边栏插入页码，\vfill自动竖直填充，后面的强制换行符后加了 2pt，使得页码位置略靠上一点，否则太靠近底部了。因为只插入页码，所以不用设置左边栏宽度，仅仅靠默认的留白就足以得到需要的结果。

```
\setbeamertemplate{sidebar left}{\vfill\tiny\insertpagenumber\\[2pt]}
```

设置前引符

设置目录的前引符，默认是没有的。

```
\setbeamertemplate{section in toc}[ball] %有编号
\setbeamertemplate{subsection in toc}[square] %无编号
```

设置 itemize 和 enumerate 的前引符形式，一个三层，item、subitem 和 subsubitem，items 是三者全部都设置为一个样式。

```
\setbeamertemplate{itemize item}[square]
\setbeamertemplate{itemize subitem}[square]
\setbeamertemplate{enumerate items}[ball]
```

背景色

简单地改变背景色可以用设置颜色的语句

```
\setbeamercolor{background canvas}{bg=blue!20}
```

更复杂一点的，则需要使用设置模板的语句，比如设置渐进色背景。

```
\setbeamertemplate{background canvas}[vertical shading][top=white,
bottom=blue!40]
```

在背景上打格子，这句与上面的渐进色一起使用效果还可以

```
\setbeamertemplate{background}[grid][step=1cm]
```

插入 logo

插入 logo 的语句很简单，只需要设定 logo 即可，因为 logo 的位置在模板里都设定好了

```
\logo{\includegraphics[height=0.5cm]{fig/tju.pdf}}
```

对于 defalt 模板，logo 的默认位置在右边栏的右下角位置，竖直方向在灰色导航按钮之上。这个位置不是很好看。如果修改这个位置，必须先清空右边栏的 logo，否则会出现两个 logo，清空右边栏的 logo 的语句为：

```
\setbeamertemplate{sidebar right} %右边栏无内容
{
\vfill
% 保留默认的灰色导航按钮
\llap{\usebeamertemplate***{navigation symbols}\hskip0.1cm}
\vskip2pt
}
```

\llap{}必须有，表示内容右端对齐，否则不会显示。这句使得不设置右边栏宽度，仍然可以显示灰色导航按钮。

然后重新设置 logo 的位置，比如加在左边栏，放在左下角

```
\setbeamertemplate{sidebar left}
\vfill
\rlap{\insertlogo} %左端对齐
}
```

default 模板，无论 logo 在左边栏还是右边栏，如果 logo 尺寸比较大且左右文字边距比较小，都会占用和重叠文字空间。还有一种可能是放在边栏，logo 尺寸很小，占用的是 text margin 的空白，这样不会重叠文字空间。有 sidebar 的模板，logo 设置在 sidebar 是很不错的方案。headline 比较宽的模板，logo 放在 headline 也好看。

个人觉得 logo 最佳位置是在 frametitle 右边，这个设置的代码修改量稍微有些大。将 logo 插入右边栏最上面，上和右各留一点边距，然后修改相应模板里的 outertheme 文件中的 frametitle 格式，比如修改

beamerouterthemedefault.sty。将 frametitle 对应的 box 宽度调整得小一点，默认是**\textwidth**，根据 logo 的大小调整一下 box 的宽度。**不是调整\textwidth的大小，是调整模板设置中的 beamercolorbox 宽度**。不要直接用`\setbeamertemplate{frametitle}`修改 frametitle 样式，虽然手册给了一个例子，但是这么做非常不好，frametitle 的设置还包含条件语句，有 framesubtitle 和没有时怎么办，手册上那个简单的例子会导致 framesubtitle 无法显示。比如说默认 43 比例时，box 宽度修改为`0.9\textwidth`，logo 图片 0.7cm 或者 0.8cm，上和右留 0.1cm 边距。

如果放在 frametitle 左边，则最好将插入 logo 的命令写到插入 frametitle 的命令前，在 logo 与 frametitle 之间水平留白一点间距，frametitle 对应的 box 宽度仍然是**\textwidth**。不要使用左边栏插入方式。此时，logo 的尺寸就只能设置的比较小了，否则会扩展 frametitle 的占比。所以笔者觉得 logo 放在 frametitle 右边大小最适宜。上面的讨论和参数的修改都是基于 default 模板。

可以 copy 模板文件到当前 tex 目录下，这样可以测试修改默认模板，不会影响原来的安装文件。beamer 应该是默认当前目录下文件是首选，这样可以很方便学习修改模板文件和测试修改结果。

7.7 beamer 内置的 box 环境

beamer 内置了两个 box 环境，其中 beamercolorbox 是设置模板的主要工具。

简版的 beamercolorbox 称为 beamerboxesrounded，

```
\begin{frame}
\setbeamercolor{uppercol}{fg=black,bg=green!30}
\setbeamercolor{lowercol}{fg=black,bg=gray!30}
\begin{beamerboxesrounded}[upper=uppercol,lower=lowercol,shadow=true
,width=\textwidth]{公式}
\[
f(x)
\]
\end{beamerboxesrounded}
\end{frame}
```

beamerboxesrounded 的效果，可以被 tcolorbox 代替。

beamercolorbox 环境的参数更多一些，下面是一个例子

```
\setbeamercolor{bgcyan}{fg=black,bg=cyan!20}
\begin{beamercolorbox}[wd=\textwidth,left,sep=3pt,shadow=true,
rounded=true]{bgcyan}
\bf \large beamercolorbox is a colorful box with many parameters.
\end{beamercolorbox}
```

7.8 beamer 模板设计

beamer 模板设计的基本语法框架为

```
\setbeamertemplate{名字}
{
\begin{beamercolorbox} %一般都会有一个 box
...
\end{beamercolorbox}

\begin{beamercolorbox} %可以重叠多个 box，计算好尺寸即可
...
\end{beamercolorbox}
}
```

单从效果来说，上面的语句就可以把想做的效果做出来。

更复杂的代码结构，推荐使用下面的方式。

```
\defbeamertemplate*
\usebeamertemplate***
```

模板文件在texmf-dist/tex/latex/beamer目录下，主模板文件内容很少，主要设置分在 innertheme 和 outertheme 文件中，以及字体和颜色文件中。innertheme 主要设置 frame 环境里的内容的格式，outertheme 主要设置 frametitle、headline、footline 和 sidebar 的格式。

虽然理论上，beamer 模板设计主要依赖于\setbeamertemplate语句和 beamercolorbox 环境，但是 beamer 模板设计还是相对繁琐的。LaTeX 总是做简单的事情很复杂。大多数情况还是推荐在内置模板基础上修改，如果想完全从头设计模板，建议从读 default 的四个主要模板文件开始。

7.9 两个重新定义 frame 的例子

第一个例子是 default 模板，用 tcolorbox 做的效果。整个 frame 就是一个 tcolorbox。直接写了 fragile 参数，所以 lstlisting 环境可以直接使用。

```
\documentclass{beamer}

\setbeamersize{text margin left=4mm,text margin right=4mm}

\usepackage{CJKutf8}
\usepackage{listings}
\usepackage{tcolorbox}

%重新定义 frame 环境
\newenvironment{myframe}{\begin{frame}[fragile,environment=myframe
]\startbox}{\stopbox\end{frame}}
\newcommand\startbox[1][]{\vspace{2mm}\begin{tcolorbox}[colback=
cyan!5,colframe=blue!60!black,width=12cm,height=8.9cm,#1]}
\newcommand\stopbox{\end{tcolorbox}}

\begin{document}
\begin{myframe}[title=frame]
here
\[
f(x)
\]
\end{myframe}

\begin{myframe}[title=frame,colback=red!5,colframe=red!60!black]
here
\[
f(x)
\]
\end{myframe}

\end{document}
```

第二个例子 frametitle 在 tcolorbox 外部，tcolorbox 没有 title

```
\documentclass{beamer}
```

```
\setbeamersize{text margin left=4mm,text margin right=4mm}

\usepackage{CJKutf8}
\usepackage{listings}
\usepackage{tcolorbox}

\newenvironment{myframe}[1][]{\begin{frame}[fragile,environment=
myframe]\frametitle{\hskip5pt #1}\startbox}{\stopbox\end{frame}}
\newcommand\startbox[1][]{\begin{tcolorbox}[colback=green!5,
colframe=black!20,width=12cm,height=8cm,#1]}
\newcommand\stopbox{\end{tcolorbox}}

\begin{document}
\begin{myframe}[media][colback=green!5,colframe=blue!50!black]
here
\[
f(x)
\]
\end{myframe}

\begin{myframe}[media][colback=red!5,colframe=red!50!black]
here
\[
f(x)
\]
\end{myframe}
\end{document}
```

这两个例子都是固定 43 比例的，功能比较少，更复杂的结果需要带入更多的参数。tcolorbox 很漂亮，以重新定义环境的方式将 tcolorbox 融合到 beamer 里还是不错的。

Index

abstract, 60
amscd, 112
amsfonts, 114
amsmath, 96
amssymb, 114
anyfontsize, 22
array, 47

background, 87
balance, 61
beamer, 229
biber, 10, 68
biblatex, 10, 68
bibtex, 8, 64
bibunits, 67
bicaption, 45
bigints, 106
bm, 106
booktabs, 49

chapterbib, 66
chemfig, 211
CJKutf8, 5
colortbl, 50
ctex, 8

empheq, 113
environ, 189
epstopdf, 19
esint, 105
eso-pic, 88
exam, 205

fancyhdr, 85
ffuserguide, 92
float, 90
flowfram, 92
footmisc, 86
footnpag, 86
framed, 40

geometry, 34
glossaries, 16
graphicx, 41

hangindent, 25
hyperref, 56

ifthen, 15, 77
imakeidx, 13
include, 4
indentfirst, 25
input, 4

letter, 87
longtable, 55
lscape, 35

makecell, 51
makeglossaries, 17
makeidx, 11

makeindex, 12, 14
marginnote, 28
mathrsfs, 115
midpage, 27
minipage, 28
multicol, 89
multirow, 53

natbib, 8
natlib, 64
nicematrix, 109
nomencl, 14
ntheorem, 118

oplotsymbl, 115

paracol, 91
pdfpages, 35
ps2eps, 19
psnfss2e, 24
pstricks, 18

setspace, 23
showexpl, 194
subfig, 46
subfigure, 45

texdoc, 1
threeparttable, 54
threeparttablex, 55
tikz, 121
    3d, 165
    angles, 137
    arrows.meta, 127
    backgrounds, 145
    calc, 168
    circuitikz, 178
circuits.ee, 175
circuits.ee.IEC, 175
circuits.logic.IEC, 175
decorations.markings, 162
decorations.pathmorphing, 162
decorations.shapes, 162
hf-tikz, 182
intersections, 171
karnaugh, 185
math, 146
matrix, 173
mindmap, 174
perspective, 168
pgf, 121
pgf-spectra, 183
pgfornament, 185
pgfornament-han, 185
pgfplots, 179
pinoutikz, 178
positioning, 150
quantikz, 184
quotes, 137
sa-tikz, 178
shadings, 142
shapes.arrows, 158
shapes.geometric, 152
shapes.multipart, 159
shapes.symbols, 158
through, 171
tikz-3dplot, 166
tikz-among-us, 185
tikz-cd, 185
tikz-dependency, 185
tikz-dimline, 184
tikz-feynhand, 183

tikz-feynman, 183
tikz-network, 184
tikz-optics, 184
tikz-palattice, 183
tikz-planets, 183
tikz-relay, 178
tikz-timing, 178
tikz-trackschematic, 184
tikz-truchet, 186
tikzducks, 186
tikzlings, 186
tikzmark, 185
tikzmarmots, 186
tikzorbital, 183
tikzpeople, 186
tikzrput, 185
tikzsymbols, 186
titleps, 72
titlesec, 72
titletoc, 72

ulem, 31
umoline, 31

write18, 10

xcolor, 37
xdvipdfmx, 19
xeCJK, 7
xmpmulti, 237

zhnumber, 82