

L^AT_EX Tutorials

Simple and professional

Song Gao, Bingkun Huang

Nanjing University

October 28, 2018



- 1 与 \LaTeX 相遇
- 2 文档
- 3 文章结构
- 4 文本环境举隅
- 5 中文支持
- 6 强大的数学公式
- 7 图表与浮动体
- 8 来做幻灯片
- 9 自动化工具
- 10 其他



- 1 与 \LaTeX 相遇
- 2 文档
- 3 文章结构
- 4 文本环境举隅
- 5 中文支持
- 6 强大的数学公式
- 7 图表与浮动体
- 8 来做幻灯片
- 9 自动化工具
- 10 其他



1 与 \LaTeX 相遇

- 上手的准备



预防针

可是同样的编译器，我在latex beamer的模板中写为什么就是对的啊

19:44

你现在用的是什么模板

whhf的啊

beamer那个是阿姨github下的

你可以试试把阿姨用的一些包添加上去，因为latex有的包是专门用来修复一些细节问题的

我转jpg试了一下，还是报错

我试试

```
\includegraphics[
width=0.8\textwidth,
natwidth=1516,natheight=792]
{22.2.5.png}
```

搞定了

stackoverflow上的回答还是有点靠谱的

微笑。



码了

既然不能识别，就手动逼他识别



怎样学习 L^AT_EX

本次讲座入门后，只需做到两点：

- STFW, Search the Friendly Web, 上网搜索
- RTFM, Read the Fruitful Manual, 看手册



介绍

L^AT_EX 是一种基于 T_EX 的排版系统，即使用户没有排版和程序设计知识也可以生成优美的高质量文档，其在复杂表格和数学公式上的表现尤其突出



介绍

L^AT_EX 是一种基于 T_EX 的排版系统，即使用户没有排版和程序设计知识也可以生成优美的高质量的文档，其在复杂表格和数学公式上的表现尤其突出

精确 复杂 细节 品位

光说没用，举些例子——pre 文件夹中两份平时作业



介绍

L^AT_EX 是一种基于 T_EX 的排版系统，即使用户没有排版和程序设计知识也可以生成优美的高质量的文档，其在复杂表格和数学公式上的表现尤其突出

精确 复杂 细节 品位

光说没用，举些例子——pre 文件夹中两份平时作业
LaTeX 入门难，越用越方便；word 入门简单，越用越难



上手试一试

编译你的第一个.tex 文件

```
\documentclass{article}

\begin{document}
This is my first document\\
Hello, \LaTeX!
\end{document}
```



- 1 与 \LaTeX 相遇
- 2 文档
- 3 文章结构
- 4 文本环境举隅
- 5 中文支持
- 6 强大的数学公式
- 7 图表与浮动体
- 8 来做幻灯片
- 9 自动化工具
- 10 其他



2 文档

- 框架
- 两个必要元素
- 环境
- 字体



- 你书写的是包含 LaTeX 代码和实际内容的纯文本文件 (.tex 文件)。
- LaTeX 代码控制实际内容的格式和样式。



```
% xiaoming.tex
% 注释说明
\documentclass[UTF8]{ctexart} % 文档类型
\title{食谱}
\author{小明}
\date{\today} % 以上为“导言区”
\begin{document} % 文档开始
% document 内的内容可与pdf文件一一对应！
\maketitle
现作出如下声明
\begin{itemize}
    \item 不吃胡萝卜
    \item 豆花只吃甜的
\end{itemize}
\end{document} % 以上为“正文区”
```



2 文档

- 框架
- 两个必要元素
- 环境
- 字体



documentclass

- documentclass, 文档类, 是第一个必要元素:
写的是什么, 怎样排版

```
\documentclass[·]{·}
```

% 方括号内为可选项, 可设置纸张大小, 字体大小, 纸张方向, 编码方式, 草稿定稿, 单双面等属性



documentclass

- documentclass, 文档类, 是第一个必要元素:
写的是什么, 怎样排版

```
\documentclass[·]{·}
```

% 方括号内为可选项, 可设置纸张大小, 字体大小, 纸张方向, 编码方式, 草稿定稿, 单双面等属性

- 常用标准文档:
article, book, report, letter, beamer (前三者为基本文档类; ctex 宏集了解一下)
- cls 文件:
自定义的文档类型 (CUMCMThesis-master)



document

document 环境是一份 tex 文档的第二个必要元素，其内为 pdf 文件的具体内容



document

document 环境是一份 tex 文档的第二个必要元素，其内为 pdf 文件的具体内容

```
% xiaoming.tex
% 注释说明
\documentclass[UTF8]{ctexart} % 文档类型
\title{食谱}
\author{小明}
\date{\today} % 以上为“导言区”
\begin{document} % 文档开始
% document 内的内容可与pdf文件一一对应！

\maketitle
现作出如下声明
\begin{itemize}
  \item 不吃胡萝卜
  \item 豆花只吃甜的
\end{itemize}
\end{document} % 以上为“正文区”
```



2 文档

- 框架
- 两个必要元素
- **环境**
- 字体



什么是环境

- 你也许注意到了`\begin` 和`\end`



什么是环境

- 你也许注意到了`\begin` 和`\end`
- `\begin` 和`\end` 语句不是命令，而是对环境进行了定义。
- `begin` 和 `end` 间的环境，代表这块区间应用的**排版规则**
- 可以多个环境层层嵌套（不可交叉!）



% 正确用法:

```
\begin{document}  
  \begin{environment1}  
    \begin{environment2}  
      \end{environment2}  
    \end{environment1}  
\end{document}
```

% 错误用法:

```
\begin{document}  
  \begin{environment1}  
    \begin{environment2}  
  \end{environment1}  
    \end{environment2}  
\end{document}
```



- 模块化与环境的思想
打开“environment_cmp.tex”
- 为什么 section2 每一行都居中了？
- 你能不能通过添加环境的方式，让 section2 第一行靠左？（提示：flushleft）
- 这是否引发了你对优先级的思考？



2 文档

- 框架
- 两个必要元素
- 环境
- 字体



问题

- 字体太丑，或者有字体要求（如 Times New Roman, 俄语, 德语）
- 字体颜色
- 键盘无法输入的符号（如 π , Å, °C）
- 貌似可以用键盘输入的符号（如%, {, ~）
- 其他（空格:

\$	\$
----	----

,

\$	\$
----	----

,

\$	\$
----	----

,

\$	\$
----	----

,

\$	\$
----	----

\$	\$
----	----

,

\$	\$
----	----

)



解决方法

- 了解相关的宏包，以及相关命令设置，如 `fontspec`, `babel` 等等
- 要用去网上查
- 看手册（已准备好，在 LM 文件夹里）
- 以上两条也针对其他琐碎的细节，如字号、行距、盒子等等



- 1 与 \LaTeX 相遇
- 2 文档
- 3 文章结构**
- 4 文本环境举隅
- 5 中文支持
- 6 强大的数学公式
- 7 图表与浮动体
- 8 来做幻灯片
- 9 自动化工具
- 10 其他



- 3 文章结构
 - 标题
 - 章节
 - 多文件编译



- 打开“title.tex”
- 通过添加和取消注释，看看前三种标题页的效果
- 主要是 title,author,date（试试能不能在 cetxart 文档中使用 subtitle）
- 自定义标题页



3 文章结构

- 标题
- 章节
- 多文件编译



- 打开“tutorial to the freshmen of DII.pdf”
- 一篇完整的 article 或者 report, 其文本结构是怎样的?



- 这篇 report 主要是中文内容, 这种分模块 (section), 生成目录 (tableofcontents) 的形式, 是一种十分经典的文章结构
- 这篇 report 主要用了以下几个命令 (尝试寻找其他的章节命令, 比如 \clearpage vs \newpage)

```
\tableofcontents  
\newpage  
\section{•}  
\subsection{•}  
\subsubsection{•}
```

- 章节标题的格式是预定好的, 一般需要用通过其他宏包完成定制 (\CTEXset{ })
- RTFM, 查看 CTEX 宏集手册, 尝试定义出 “第一节” 的章节标题编号
- 联想一下 maketitle, document 正文区里的代码是与 pdf 文件——对应的, 所以 \tableofcontents 应该放在 .tex 文档的哪里?
- 通常主流的 latex 编辑器都能展示出你的各个 section 标题, 尝试通过它更好地调整你的文章结构, 或是快速定位代码



3 文章结构

- 标题
- 章节
- 多文件编译



问题

- 编译“tutorial to the freshmen of DII” 需要多久?



问题

- 编译“tutorial to the freshmen of DII” 需要多久?
- 如果你要编译一整本书呢?



问题

- 编译“tutorial to the freshmen of DII” 需要多久？
- 如果你要编译一整本书呢？
- 与他人合作写一篇 report 或 beamer 等等，是否觉得各方整合起来会遇到很多问题？



问题

- 编译“tutorial to the freshmen of DII” 需要多久？
- 如果你要编译一整本书呢？
- 与他人合作写一篇 report 或 beamer 等等，是否觉得各方整合起来会遇到很多问题？

按文档的逻辑层次，把整个文档分成多个.tex 源文件，便于检索管理以及多人协同编写



以讲座所用 beamer 为例

```
\input{part/preamble}  
\begin{document}  
\frame{\titlepage}  
\frame{\tableofcontents[hideallsubsections]}  
\input{part/try}  
\input{part/docu}  
\input{part/stru}  
\input{part/textenvi}  
\input{part/chin}  
\input{part/formula}  
\input{part/tab&fig}  
\input{part/slides}  
\input{part/auto}  
\input{part/others}  
\end{document}
```



划分文档的主要方式是`\input{}` 和 `\include{}`, 将不同的章节内容写在不同的.tex 文件中, 用上述两种方式引入主文件中。

`input` 相当于 `ctrl+c` 和 `ctrl+v`, 直接把代码复制过来, 稳定、实用

`include` 常和 `\includeonly` 一起使用, 可以选择性地编译子文件, 常用于书籍之类的特大文档

两种方式的编译效果有细节上的不同, 自己试试



截图来自 stackexchange.com, 题为 input vs include

`\input{filename}` imports the commands from `filename.tex` into the target file; it's equivalent to typing all the commands from `filename.tex` right into the current file where the `\input` line is.

`\include{filename}` essentially does a `\clearpage` before and after `\input{filename}`, together with some magic to switch to another `.aux` file, and omits the inclusion at all if you have an `\includeonly` without the filename in the argument. This is primarily useful when you have a big project on a slow computer; changing one of the include targets won't force you to regenerate the outputs of all the rest.

`\include{filename}` gets you the speed bonus, but it also can't be nested, can't appear in the preamble, and forces page breaks around the included text.



我们用的比较多的是 `input`，当你觉得某一部分代码太长时，可以考虑写在别的`.tex` 文件中，精简主文件，平时主要用在下面几种情况

- 导言区繁多的设置（通常会写在名为 `preamble` 的文件中，请看问求的作业模板）
- 异常庞大的表格
- 设置复杂的图片（比如需要使用很多子图 `subfig` 时）

此外可以去了解一下`\endinput` 的使用



- 1 与 \LaTeX 相遇
- 2 文档
- 3 文章结构
- 4 文本环境举隅**
- 5 中文支持
- 6 强大的数学公式
- 7 图表与浮动体
- 8 来做幻灯片
- 9 自动化工具
- 10 其他



4 文本环境举例

- 常用文本环境



(其实会下面的就够了.....)

- 文本环境: `quote`, `quotation`, `abstarct`(写诗的同学可以去了解 `verse`)
- 列表环境: `itemize`, `enumerate`
- 定理类环境: `thm`, `lemma`, `definition`, `proof`
- 抄录和代码环境: `verb`, `lstlisting`
- 脚注: `footnote`



练习和小建议

- 打开练习 list, theorem, listing_code, page
- 想要用好文本环境，更建议看手册而不是上网搜索



- 1 与 \LaTeX 相遇
- 2 文档
- 3 文章结构
- 4 文本环境举隅
- 5 中文支持**
- 6 强大的数学公式
- 7 图表与浮动体
- 8 来做幻灯片
- 9 自动化工具
- 10 其他



5 中文支持

- 了解 C_T_EX



How to get Chinese document?

C_T_EX

C_T_EX 是最新的也是最便利的 L^AT_EX 中文支持的开源项目，可以参考其[使用手册](#)。

- 全面汉化
- 更智能的排版
- 强大的自定义设置

一般使用默认设置即可，有特殊需求可翻阅手册作适当调整。



表: CTEX 宏集组成

类别	文件	说明
文档类	ctexart.cls	标准文档类 article 的汉化版本, 一般适用于短篇幅的文章
	ctexrep.cls	标准文档类 report 的汉化版本, 一般适用于中篇幅的报告
	ctexbook.cls	标准文档类 book 的汉化版本, 一般适用于长篇幅的书籍
	ctexbeamer.cls	文档类 beamer 的汉化版本, 适用于幻灯片演示
宏包	ctex.sty	提供全部功能, 但默认不开启章节标题设置功能, 需要使用 heading 选项来开启
	ctexsize.sty	定义和调整中文字号, 在 ctex 宏包或 CTEX 中文文档类之外单独调用
	ctexheading.sty	提供章节标题设置功能, 在 ctex 宏包或 CTEX 中文文档类之外单独调用



- 文档类

```
\documentclass{ctexart}  
\documentclass{ctexrep}  
\documentclass{ctexbook}  
\documentclass{ctexbeamer}
```

- 宏包

```
\usepackage{ctex}  
\usepackage{ctexsize}  
\usepackage{ctexheading}
```

注意：ctexcap 是默认打开 heading 选项的 ctex 宏包，手册中明确说明其为过时宏包！



两个最常用选项 `heading` 与 `scheme`。
打开 `ctex.tex`, 体会不同选项的区别。

- `ctexart` 类文档与 `ctex` 宏包的默认选项都是 `heading=true,scheme=chinese`
- `heading=false,scheme=plain` 相当于只是支持中文, 所有的版式都还是英文的
- 如果觉得标题居中太蠢了想要靠左, 可以用 `heading=false(scheme=chinese` 是默认选项不用改)



- 1 与 \LaTeX 相遇
- 2 文档
- 3 文章结构
- 4 文本环境举隅
- 5 中文支持
- 6 强大的数学公式**
- 7 图表与浮动体
- 8 来做幻灯片
- 9 自动化工具
- 10 其他



写在前面

除了一些最简单的数学公式，大部分都需要宏包支持。建议平时需要输入公式时就直接使用以下四个宏包

```
\usepackage{amsmath,amsfonts,amssymb,mathtools}
```

这能够支持大多数的数学公式，而不必去纠结什么时候应该用哪一个、哪一个多余的。

在讨论更多的数学环境前，我们统一使用 $\$...\$$ 输入行内 (inline) 公式， $\[...\]$ 输入行间 (displayed) 公式。

注意：凡是要用到数学字符的地方必须进入数学环境，有些字符虽然不在数学环境下也能输入，但是区别很大！

$$1+1=2 \quad 1 + 1 = 2 \quad a \sin x \quad a \sin x$$



6 强大的数学公式

- 数学符号
- 数学结构
- 数学环境



字母表与数学字体

在公式环境内默认使用 `\mathnormal` 字体，与正文环境下默认字体对比如下

- Capital
 - ABCDEFGHIJKLMNOPQRSTUVWXYZ
 - *ABCDEFGHIJKLMNOPQRSTUVWXYZ*
- Lower case
 - abcdefghijklmnopqrstuvwxyz
 - *abcdefghijklmnopqrstuvwxyz*

其明显特征是斜体且间距较正文大一些。

而 `\mathit`, `\mathrm`, `\mathbf`, `\mathsf`, `\mathtt` 这些数学字体通常直接使用相对应的正文字体 `\text**`。



选用正确的字体

- 变量使用默认字体, 如 x, y, z
- e, i 等常量应使用罗马体 `\mathrm`
- 集合 \mathbb{N}, \mathbb{R} 等可使用 `amssymb` 提供的黑板粗体 `\mathbb`
- 向量 v 应加粗, 实现方法有很多, 这里推荐 `bm` 宏包的 `\bm` 命令



希腊字母

● 小写希腊字母

- 一般 `\+小写拼写`，如 `\alpha` 即输出 α
- 注意 μ, ν 分别为 `\mu`, `\nu`
- 一部分希腊字母有变体，这些变体在数学公式中常常用到，只需在前面加上 `var`

$$\epsilon \rightarrow \varepsilon \quad \phi \rightarrow \varphi \quad \theta \rightarrow \vartheta \quad \sigma \rightarrow \varsigma \quad \pi \rightarrow \varpi$$

● 大写希腊字母

- 一部分大写希腊字母与拉丁字母形状相同，直接使用即可，如 A, B
- 将小写希腊字母输入时首字母大写即可得到其大写形式，如 `\Gamma` 得到 Γ
- 大写希腊字母一般用正体，前面加 `var` 或使用 `\mathnormal` 可得到倾斜形式

● 小写希腊字母直立体形式？

- 数字字体宏包 `upgreek`，在前面加上 `up`
- `\pi` \rightarrow `\uppi` $\pi \rightarrow \uppi$



常用符号

表: 数学普通符号 (部分)

\hbar	<code>\hbar</code>	ℓ	<code>\ell</code>	∂	<code>\partial</code>
∞	<code>\infty</code>	$'$	<code>\prime</code>	\emptyset	<code>\emptyset</code>
∇	<code>\nabla</code>	\forall	<code>\forall</code>	\exists	<code>\exists</code>
\neg	<code>\neg</code>	\emptyset	<code>\varnothing</code>	\angle	<code>\angle</code>
\clubsuit	<code>\clubsuit</code>	\diamond	<code>\diamond</code>	\heartsuit	<code>\heartsuit</code>
\spadesuit	<code>\spadesuit</code>	\triangle	<code>\triangle</code>	\square	<code>\square</code>
\flat	<code>\flat</code>	\natural	<code>\natural</code>	\sharp	<code>\sharp</code>



常用符号

表: 可同时用在文本和数学模式中的符号

#	<code>\#</code>	&	<code>\&</code>	%	<code>\%</code>	\$	<code>\\$</code>
_	<code>_</code>	{	<code>\{</code>	}	<code>\}</code>		
¶	<code>\P</code>	§	<code>\S</code>	†	<code>\dag</code>	‡	<code>\ddag</code>
©	<code>\copyright</code>	£	<code>\pounds</code>	...	<code>\ldots</code>		
✓	<code>\checkmark</code>	®	<code>\circledR</code>	✠	<code>\maltese</code>	¥	<code>\yen</code>

表: 希伯来字母

ℵ	<code>\aleph</code>	beth	<code>\beth</code>	daleth	<code>\daleth</code>	gimel	<code>\gimel</code>
---	---------------------	------	--------------------	--------	----------------------	-------	---------------------



数学算子

Math operator

- 巨算符 (large operator)
 - 所谓巨算符是大小可变的运算符
 - 其在行内与行间大小是不同的
 - 一般可以添加有上下标 (数学结构中会讲)
 - 行内行间上下标的形式也有所不同

$$\int_a^b \int_a^b$$

$$\iint_m^n \iint_m^n$$

$$\oint_{|z|=1} \oint_{|z|=1}$$

$$\sum_{n=0}^{\infty} \sum_{n=0}^{\infty}$$

$$\prod_{1 \leq i < j \leq n} \prod_{1 \leq i < j \leq n}$$

$$\bigcup_{i=1}^n \bigcup_{i=1}^n$$

 $\backslash limits$
 $\backslash nolimits$

行内公式尽量避免上下限放在算符上下位置



数学算子

Math operator

• 文字名称的算子

- 使用直立罗马体排印，一般 $\boxed{\backslash + \text{名称}}$ 即可
- 分为两类，带上下限的与不带上下限的

- \log \sin \cosh \arctan \exp
- \lim \max \min \sup \inf

- 带上下限的算子使用起来与巨算符相似

$$\lim_{x \rightarrow 0} \quad \lim_{x \rightarrow 0}$$

- 如果需要定义新的算子，可以在导言区使用 $\backslash \text{DeclareMathOperator}(*)$ ，带 $*$ 表示带上下限

- $\backslash \text{DeclareMathOperator}\{\backslash \operatorname{arccot}\}\{\operatorname{arccot}\}$
 arccot
- $\backslash \text{DeclareMathOperator}\{\backslash \operatorname{sign}\}\{\operatorname{sgn}\}$
 sgn
- $\backslash \text{DeclareMathOperator}\ast\{\backslash \operatorname{limsup}\}\{\operatorname{lim}\backslash, \operatorname{sup}\}$
 limsup



二元运算符与关系符

- $\boxed{+}$, $\boxed{-}$, $\boxed{\times}$, $\boxed{\div}$ 等二元运算符与 $\boxed{=}$, $\boxed{>}$, $\boxed{<}$, $\boxed{\neq}$, $\boxed{\leq}$, $\boxed{\geq}$, $\boxed{\ll}$, $\boxed{\gg}$ 等关系符与前后变量之间会保留一定的空隙, 且行内公式可以在它们的后面断行。

- 二元关系符数量庞大, 除了一些最常用的, 一般有需要的时候随用随查即可。
- 使用 $\boxed{\mathbin}$ 和 $\boxed{\mathrel}$ 可以将其参数当作二元运算符和二元关系符来对待, 可用于定义新运算或一些其他用途。
- 箭头符号与逻辑符号使用时同样可以查阅, 不再赘述。



定界符

$$\left\{ \frac{\partial y_j}{\partial x_i} \middle| i = 1, 2, \dots, n, j = 1, 2, \dots, m \right\}$$

上例中的 $\{ | \}$ 均为定界符，其大小可以根据式子的大小而改变。

- 括号定界符

$() [] \{ \} \langle \rangle \llbracket \rrbracket$

- 非括号定界符

$/ \backslash | \parallel$

- 输入

- $() [] \{ \} \langle \rangle \lfloor \rfloor \lceil \rceil$
- $/ \backslash | \parallel$



如何确定定界符大小

• 自动调整

- 使用 `\left(... \right)`，左右必须成对出现
- 如果只需要一个定界符，可以用 `.` 表示空的定界符，如 `\left.`
- 需要三个定界符的话可使用 `\middle` 放在中间，如前例

• 手动调整

- 提供了 `\big`，`\Big`，`\bigg`，`\Bigg` 四种大小供手动调节
- 可以在后面加上 `l m r` 配套使用
- 自动调整效果不理想时，可以试试手动调整
- 在明确“左右”（如 `\left \right` 或 `\Bigl \Bigr`）时，可直接使用 `<>` 表示尖括号而不致与 `<>` 混淆



标点符号

- 数学环境内应使用英文标点，否则会报错
- 常用的标点符号有，；！？：
- 应注意的是`:`是作为二元关系符存在的（如 $f(x) := x, \{x : x > 0\}$ ），而作为标点的冒号是 `\colon`，其两侧间距不同，表示比例时则最好用 `\mathbin{:}`
- 省略号有很多种，在矩阵中常用到，平时最常用的是 `\cdots` 和 `\ldots` ...（或直接使用 `\dots` 让 latex 自己选择）
- 标点后是无法换行的，像 $f(x, y, z)$ 谁也不希望在中间断行，而 $a_1, a_2, a_3, \cdots, a_n$ 希望能断行时应将标点置于数学环境之外并加上空格
- 数学环境中空格不起作用，但可以分隔各部分让公式结构更清晰



6 强大的数学公式

- 数学符号
- 数学结构
- 数学环境



上下标

- 上标用 \wedge 表示, 下标用 \sub 表示
- 超过一个字符时就应使用 $\{ \}$
- ' 是一种特殊的上标, 相当于 $\wedge\backslash\text{prime}$
- 使用 $\backslash\text{overset}$ 和 $\backslash\text{underset}$ 可以在任意符号上下方添加标记



分式

- 使用 `\frac{num}{den}` 输入分式，先输入分子 (numerator)，后输入分母 (denominator)
- 行内默认使用正文格式 (text style) 分式，行间默认使用显示格式 (display style) 分式，可以使用 `\tfrac` 与 `\dfrac` 指定使用某一种
- 行内使用分式时， a/b 效果往往比 $\frac{a}{b}$ 更好
- 连分式用 `\cfrac`，这里不细讲
- 顺带一提，`\textstyle` 和 `\displaystyle` 可以指定整条公式以哪种方式显示



根式

- `\sqrt[root]{arg}`
- `[]` 内为可选项, 用 `\uproot` 与 `\leftroot` 可以微调 root 位置, 参数为整数
- 被开方数不是整数或结构复杂时, 通常改用等价的指数形式
- 使用 `\vphantom` 占位来获得统一的高度

`\sqrt{\frac{1}{2}} < \sqrt{\vphantom{\frac{1}{2}}2}`

$$\sqrt{\frac{1}{2}} < \sqrt{2}$$

- `\mathstrut` 用来平衡不同高度和深度的字母

`\sqrt{b} \sqrt{y}`

`\sqrt{\mathstrut b} \sqrt{\mathstrut y}`

$$\sqrt{b}\sqrt{y} \rightarrow \sqrt{b}\sqrt{y}$$



矩阵

- 输入矩阵结构需要矩阵环境，它们的区别在于外面的括号不同
 - `matrix` 环境 无括号
 - `pmatrix` 环境 圆括号
 - `bmatrix` 环境 方括号
 - `Bmatrix` 环境 大括号
 - `vmatrix` 环境 单竖线
 - `Vmatrix` 环境 双数线
- 以一个最简单的二阶矩阵为例

```
\[ \begin{pmatrix}
    a_{11} & a_{12} \\
    a_{21} & a_{22}
\end{pmatrix}
```

$$\begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix}$$



6 强大的数学公式

- 数学符号
- 数学结构
- 数学环境



概述

我们先前使用的 $\boxed{\dots}$ 实质是

```
\begin{equation*}  
  ...  
\end{equation*}
```

的缩写，其中 “*” 表示不给该公式标序。

出于各方面的需求，我们并不满足于这样一个简单居中、无法换行的数学环境，而是会有诸多其他方面的要求。这时我们需要更多的数学环境来提供支持。



多行公式

每行均为一个公式

- 简单的多个公式罗列: `gather(*)`
 - 每行公式都是居中显示的
 - 使用 `\\` 换行
 - 某行不需要编号, 在 `\\` 前加 `\notag`
- 公式按关系符对齐: `align(*)`
 - 使用 `\\` 换行, 关系符前加 `&` 表示对齐
 - `&` 一般放在二元关系符 (`=`, `<`, `>` 等) 前面
 - 有时可巧用 `\phantom` 变通
- 多行公式每行都会产生一个编号, 不希望产生的话可以使用 `\notag`



练习：尝试输出以下公式。

$$\left(\frac{\partial A}{\partial x}\right)_y = \lim_{\Delta x \rightarrow 0} \frac{f(x + \Delta x, y) - f(x, y)}{\Delta x} \quad (1)$$

$$\left(\frac{\partial A}{\partial y}\right)_x = \lim_{\Delta y \rightarrow 0} \frac{f(x, y + \Delta y) - f(x, y)}{\Delta y} \quad (2)$$

$$\begin{aligned} & (a + b)(a^2 - ab + b^2) \\ &= a^3 - a^2b + ab^2 + a^2b - ab^2 + b^3 \\ &= a^3 + b^3 \end{aligned} \quad (3)$$



多行公式

- `align(*)` 的另一个用处：排版多列公式

- 每行的基本格式

`a &= b & c &= d & e &= f \\`

- 最后一行不需要 `\\`
 - 按照行来标序
 - `&` 必然为奇数个，出现偶数个会报错（为什么?）

- 关于 `flalign(*)`

- 与 `align(*)` 是类似的
 - 会把多列公式水平方向分散对齐

- 关于 `alignat(*)`

- 与 `align(*)` 也是类似的
 - 不会主动产生间距，需要手动输入间距大小
 - 需要一个参数表示每行要对齐的公式个数



对比 align(*) 与 flalign(*)

- 使用 align

$$x + y = 2 \quad x = 1 \quad (1)$$

$$x - y = 0 \quad y = 1 \quad (2)$$

- 使用 flalign

$$x + y = 2 \quad x = 1 \quad (1)$$

$$x - y = 0 \quad y = 1 \quad (2)$$

巧用 alignat (使用 align 必须借助 `\phantom`)

```
\begin{alignat*}{5}
```

```
&1 & &+2 & &+3 & &+4 & &=10 \\
```

```
&1 & & & &+3 & & & &=4 \\
```

```
& & &+2 & & & &+4 & &=6
```

```
\end{alignat*}
```

$$1 + 2 + 3 + 4 = 10$$

$$1 \quad + 3 \quad = 4$$

$$+ 2 \quad + 4 = 6$$



多行公式

- 如何在多行公式之间插入说明性文字？
 - 每次都退出数学环境输入文字过于麻烦
 - 退出数学环境会破坏上下公式间对齐关系
- `\text` 行内插入文字
- `\intertext` 换行顶格插入文字（前一行的 `\\` 可省略）
- `\shortintertext` 提供更为紧凑的行间距



例:

绝热过程中, $Q = 0$, 不做非膨胀功时, 有

$$dU = \delta W \quad \text{或} \quad dU + p dV = 0 \quad (1)$$

已知

$$dU = \left(\frac{\partial U}{\partial T} \right)_V dT + \left(\frac{\partial U}{\partial V} \right)_T dV \quad (2)$$

对理想气体,

$$dU = C_V dT, \quad \Delta U = \int_{T_1}^{T_2} C_V dT \quad (3)$$

若 C_V 为常数,

$$\Delta U = C_V(T_2 - T_1) = W \quad (4)$$



多行公式

- 多个联系紧密的公式如何共用一个编号？

使用 subequations 环境

- 示例：

```
\begin{subequations}
\begin{align}
a \cdot b &= b \cdot a \\
(a \cdot b) \cdot c &= a \cdot (b \cdot c) \\
a \cdot (b + c) &= a \cdot b + a \cdot c
\end{align}
\end{subequations}
```

$$a \cdot b = b \cdot a \quad (1a)$$

$$(a \cdot b) \cdot c = a \cdot (b \cdot c) \quad (1b)$$

$$a \cdot (b + c) = a \cdot b + a \cdot c \quad (1c)$$



练习：尝试输出如下的麦克斯韦方程组。

• 积分形式

$$\oiint_S \mathbf{D} \cdot d\mathbf{S} = q, \quad (1a)$$

$$\oiint_S \mathbf{B} \cdot d\mathbf{S} = 0, \quad (1b)$$

$$\oint_L \mathbf{E} \cdot d\boldsymbol{\ell} = - \iint \frac{\partial \mathbf{B}}{\partial t} \cdot d\mathbf{S}, \quad (1c)$$

$$\oint_L \mathbf{H} \cdot d\boldsymbol{\ell} = I_0 + \iint \frac{\partial \mathbf{D}}{\partial t} \cdot d\mathbf{S}. \quad (1d)$$

• 微分形式

$$\nabla \cdot \mathbf{D} = \rho_f, \quad (2a)$$

$$\nabla \cdot \mathbf{B} = 0, \quad (2b)$$

$$\nabla \times \mathbf{E} = -\frac{\partial \mathbf{B}}{\partial t}, \quad (2c)$$

$$\nabla \times \mathbf{H} = \mathbf{J}_f + \frac{\partial \mathbf{D}}{\partial t}. \quad (2d)$$



拆分公司

整体为一个公式

行内公式中，对于长公式如

$1 + 2 + 3 + 4 + 5 + 6 + 7 + 8 + 9 + 10 + 11 + 12 + 13 + 14 + 15 + 16 + 17 + 18 + 19 + 20 + 21 + 22 + 23 + 24 + 25 + 26 + 27 + 28 + 29 + 30$ 我们先前提过是可以在二元运算符后换行的，但是对于行间公式则不然，见下例。

$1 + 2 + 3 + 4 + 5 + 6 + 7 + 8 + 9 + 10 + 11 + 12 + 13 + 14 + 15 + 16 + 17 + 18 + 19 + 20 + 21 -$

这时候可以使用 `multline(*)` 环境。（注意不是 `multiline!`）



```
\begin{multline}
  1+2+3+4+5+6+7 \\\
  +8+9+10+11+12+13 \\\
  +14+15+16+17+18+19+20+21 \\\
  +22+23+24+25 \\\
  +26+27+28+29+30
\end{multline}
```

$$\begin{aligned}
 &1 + 2 + 3 + 4 + 5 + 6 + 7 \\
 &\quad + 8 + 9 + 10 + 11 + 12 + 13 \\
 &\quad + 14 + 15 + 16 + 17 + 18 + 19 + 20 + 21 \\
 &\quad + 22 + 23 + 24 + 25 \\
 &\quad + 26 + 27 + 28 + 29 + 30 \quad (1)
 \end{aligned}$$



拆分公式

- 可以看出 `multline` 环境中首行是左对齐的，尾行是右对齐的，中间行则是居中
- 公式的左右边距是通过长度变量 `\multlinegap` 和 `\multlinetaggap` 设置的
- `\shoveleft{}` 和 `\shoveright{}` 可设置中间行像首尾两行那样左/右对齐

$$\begin{aligned}
 &1 + 2 + 3 + 4 + 5 + 6 + 7 \\
 &\qquad\qquad\qquad + 8 + 9 + 10 + 11 + 12 + 13 \\
 &\qquad\qquad + 14 + 15 + 16 + 17 + 18 + 19 + 20 + 21 \\
 &+ 22 + 23 + 24 + 25 \\
 &\qquad\qquad\qquad + 26 + 27 + 28 + 29 + 30
 \end{aligned}$$



```
\setlength{\multlinegap}{4em}  
\setlength{\multlinetaggap}{4em}  
\begin{multline*}  
  1+2+3+4+5+6+7 \\  
  \shoveright{+8+9+10+11+12+13} \\  
  +14+15+16+17+18+19+20+21 \\  
  \shoveleft{+22+23+24+25} \\  
  +26+27+28+29+30  
\end{multline*}
```



拆分公式

- 如果想要在等号处对齐, 可以考虑 `split` 环境
- `split` 环境应当内嵌于 `equation` 中, 使用方法上与 `align` 等类似

```
\begin{equation}
\begin{split}
(AB)(B^{-1}A^{-1})
&= A(BB^{-1})A^{-1} \\
&= AIA^{-1} \\
&= AA^{-1} \\
&= I
\end{split}
\end{equation}
```

$$\begin{aligned}
 (AB)(B^{-1}A^{-1}) &= A(BB^{-1})A^{-1} \\
 &= AIA^{-1} = AA^{-1} \quad (1) \\
 &= I
 \end{aligned}$$



组合成块

如何输入分段函数，如

$$\operatorname{sgn} x = \begin{cases} 1, & x > 0, \\ 0, & x = 0, \\ -1, & x < 0. \end{cases}$$

使用 `cases` 环境 (同样需要内嵌在 `equation` 等环境中)。

```
\[ \operatorname{sign} x = \begin{cases} 1, & x > 0, \\ 0, & x = 0, \\ -1, & x < 0. \end{cases}
```



组合成块

- 使用 `dcases` 环境得到显示格式大小的公式
- 此外使用上述部分环境的 `+ed` 模式能得到更多的将公式组合成块的应用
 - 如: `gathered`, `aligned`, `alignedat`, `multilined`
 - `lgathered` 与 `rgathered` 可以设置环境内的若干行公式左对齐或右对齐, 配合定界符使用
- 以上环境的使用方法不变, 它们都将多行的公式组合成块放在了一个公式环境内
- 这里只举几个例子, 请自行融会贯通

请打开文件 `equation.tex` 并编译。



- 1 与 \LaTeX 相遇
- 2 文档
- 3 文章结构
- 4 文本环境举隅
- 5 中文支持
- 6 强大的数学公式
- 7 图表与浮动体**
- 8 来做幻灯片
- 9 自动化工具
- 10 其他



7 图表与浮动体

- 构建表格
- 插入图片
- 绘图语言
- 浮动体与标题控制



tabular 与 array

- tabular 主要用于文本环境（数学模式下也可使用，但是还是按照文本模式排版）
- array 用在数学环境中，主要用于排版复杂矩阵等包含数学符号的公式
- 二者用法上是相似的，这里只讲 tabular



基本格式

```
\begin{tabular}[<垂直对齐>]{<列格式说明>}  
%   \hline  
    <表项> & <表项> & ... & <表项> \\  
%   \hline  
    ...  
\end{tabular}
```

- 垂直对齐选项可选填
 - t 按表格顶部对齐
 - b 按表格底部对齐
 - < > 垂直居中
- 列格式说明符最基本的有
 - l 本列左对齐
 - c 本列居中
 - r 本列右对齐
 - | 列之间的竖线



- 每行单元格之间以 `&` 相隔, `\\` 用来换行
- 单元格内进行的设置作用域仅局限于本单元格
- `\hline` 表示两行间添加一条横线

例:

居中	靠左	靠右
11111111	2222	4444444
abcdef	ghijklm	nopqrstuvwxyz

这是一张表格:

这是一张表格:

```
\begin{tabular}[b]{|c|lr|}
\hline
\bfseries 居中 & \ttfamily 靠左 & \rmfamily\itshape
    靠右 \\
\hline
11111111 & 2222 & 4444444 \\
abcdef & ghijklm & nopqrstuvwxyz \\
\hline
\end{tabular}
```



• 更多列格式说明符

- `p{<宽>}` 本列具有固定宽度且可以自动换行，用于处理过多字符的单元格
- `@{<内容>}` 将`\texttt{<内容>}` 作为两列之间的间隔，同时会取消列间的空隙
- `*{<计数>}{<列格式说明>}` 将<列格式说明> 重复<计数> 次

姓名	收入	支出	结余	备注
qag	500.01	297.74	202.27	勤俭持家
orz	50000.00	135345.53	-85345.53	这太太太太有钱了



```

\begin{tabular}{|c|*{3}{r@{.}l|}p{4em}|}
\hline
姓名 & \multicolumn{2}{c|}{收入} & \multicolumn{2}{c|}{支出} & \multicolumn{2}{c|}{结余} & \multicolumn{1}{c|}{备注} \\
\hline
qaq & 500 & 01 & 297 & 74 & 202 & 27 & 勤俭持家 \\
\hline
orz & 50000 & 00 & 135345 & 53 & -85345 & 53 & 这太太太太太有钱了 \\
\hline
\end{tabular}

```



单元格的合并与拆分

- 使用 `\multicolumn{cols}{pos}{text}` 来合并列
 - `cols` 选择需要合并的列数 (大于等于 1)
 - `pos` 选择合并得的新单元格对齐方式
 - `text` 输入单元格内容
- 可以只合并 1 列, 目的是暂时改变该列的对齐方式
- 合并行可以使用 `multirow` 宏包, 与 `\cline{i=j}` 配合使用
- 用 `\vline` 制造竖线, 达成拆分目的, 但是间距不易掌握
- 可以在表格内嵌套表格来拆分, 注意被嵌套的表格左右两边都应该是 `@{}`
- `\makecell` 了解一下
- 骚操作有很多, 但是我不会
- 有需要自己查



思考:

```
\begin{tabular}{|p{8em}|p{8em}|p{8em}|}
```

```
\hline
```

默认靠左 & \centering 这里居中 & \multicolumn{1}{c}{
这里也居中} \\%最右边为什么不能用\centering?

```
\hline
```

\raggedleft 右对齐 & \multicolumn{1}{r|}{也右对齐惹}
& 啦啦啦 \\%尝试只排版一排，你发现了什么?

```
\hline
```

```
\end{tabular}
```

默认靠左	这里居中	这里也居中
右对齐	也右对齐惹	啦啦啦

默认靠左	这里居中	这里也居中
------	------	-------

右对齐	也右对齐惹	啦啦啦
-----	-------	-----



定宽表格与长表格

- 定宽表格可以固定表格总宽度（如等于页宽）
- 可以使用 `tabular*` 环境，举个例子

```
\begin{tabular*}{\textwidth}{|c@{\extracolsep{\fill}}}{ccccc|}
```

- 也可以用 `tabularx` 宏包提供的 `tabularx` 环境
- 长表格往往需要分页
- 可使用 `longtable` 宏包提供的 `longtable` 环境



三线表

- 在科技文档中大量使用三线表，三条表格线粗细不同，可以使用 `booktabs` 宏包。
- 大致结构如下

```
%\usepackage{booktabs}
\begin{tabular}{ccccc}
  \toprule
% 表头
  \midrule
% 内容
  \bottomrule
\end{tabular}
```



7 图表与浮动体

- 构建表格
- 插入图片
- 绘图语言
- 浮动体与标题控制



插入图片

- 插入外部图片需要使用宏包 `graphicx`
- 使用的命令与格式

```
%\usepackage{graphicx}  
\includegraphics[<选项>]{<文件名>}
```

- 文件名一般可以省略拓展名，默认与 `tex` 文档处于同一路径下

```
\includegraphics{picname}
```



拓展名

- 最初 \LaTeX 支持的图片格式有限，往往需要将图片转换格式后才能插入
- \XeTeX 支持的图片格式有 EPS, PDF, PNG, JPEG, BMP 等
- 除非遇到同一路径下两张图片名称相同拓展名不同，否则可以省略拓展名



文件位置

- 对于不是同一路径下的图片，可以使用其相对路径或绝对路径来插入图片
- 不论操作系统是什么，两层路径之间统一使用/来分隔
- 相对路径的可移植性比较高，也对于图片较多的情况也有利于文件的整理
 - 如图片 `selfie.jpg` 位于当前路径下一个名为 `pic` 的文件夹内，可使用 `\includegraphics{pic/selfie}`
 - 如图片 `logo.pdf` 位于当前路径的上一层目录下，可使用 `\includegraphics{../logo}`
- 绝对路径不推荐使用，建议把图片复制到文档同一路径或相近路径下
 - `\includegraphics{C:/Users/username/Pictures/xx}`
 - `\includegraphics{home/username/Pictures/xx}`



可选选项

- [] 内可以选填的内容有很多，最常用的是设置其图片大小
- 默认的图片尺寸是其自然尺寸，对于矢量图是制作的尺寸，对于位图则是点阵数处以图形打印度 (DPI)
- 最常用的设置大小的尺寸是 `width`, `height` 和 `scale`
- 举例

```
\includegraphics[width=\textwidth]{pic1}  
\includegraphics[height=4cm]{pic2}  
\includegraphics[scale=.3]{pic3}
```

- 此外还有一些其他选项，查阅手册或阅读刘海洋的《LaTeX 入门》都可以学习
- 对图片还能做一些基本的变换



7 图表与浮动体

- 构建表格
- 插入图片
- 绘图语言
- 浮动体与标题控制



强大的内置绘图功能

- 高端操作，超出本次讲座范围
- ~~其实我也不会~~
- 大佬学会了教教我
- [这里](#)有一个使用 `circuitikz` 绘制电路图的教程，更多的资料也可以从书籍或网站上找到



7 图表与浮动体

- 构建表格
- 插入图片
- 绘图语言
- 浮动体与标题控制



公式和浮动体是 L^AT_EX 完爆 Word 的两个方面!!



浮动体环境

直接插入图片或表格的情况下是图（表）文混在一起的，但大多数情况下我们需要把图（表）插在段落之间或单独成页，并配以标题，同时对图片的确切位置没有精确的要求，只要插在某段话附近即可。这时就需要用到浮动体环境。

- 对于图片与表格分别有两类对应的浮动体环境：figure 和 table
- figure 为例，基本语法格式为（table 类似）

```
\begin{figure}[<允许位置>]  
  <任意内容>  
\end{figure}
```

- <允许位置> 的可选项有 h t b p
• h here t top b bottom p page



- 允许位置的可选项可以只有一个，也可以是多个的组合，如 `[htp]` 则表示不允许 `b`
- 不填则默认是 `[htbp]`
- 顺序无关紧要，因为总是按照 `[htbp]` 的顺序来尝试
- 对于单独的 `h`，很多时候并不能得到令人满意的结果，因此 latex 会放宽为 `[ht]`
- 若一定要用 `h`，可以采用 `[!h]`



浮动体的本质

- Q: 浮动体仅仅是用来插图表的吗?
- A: 不是的, 浮动体里可以放入任意内容 (一段文字, 一段长代码, 一组大型公式等)。
- Q: `figure` 和 `table` 有什么区别, 是不是前者只能用来插图片后者只能用来插表格?
- A: 不是的, 前面已经说过插的东西是任意的。只是习惯上人们这么干, 这与默认生成的标题有关。
- Q: 浮动体的本质是什么?
- A: 是一个可以活动的 “盒子”。
- Q: 人类的本质是什么?
- A: 人类的本质是什么?



caption 与 label

- 使用 `\caption{<title>}` 可以为浮动体加上标题，标题可换行
- 通常对于图片标题在下方，而对于表格标题在上方
- 默认情况下 `figure` 浮动体产生的标题将注明 `Figure X`, `table` 浮动体产生的标题将注明 `Table X`, `X` 是数字，由计数器控制
- 使用 `ctex` 的中文文档则相应地为图 `X` 和表 `X`
- 这就是为什么通常 `figure` 用来插入图片而 `table` 用来插入表格，事实上这是可以更改的
- 在 `caption` 后加入 `\label{<lbl>}`，可用于交叉引用



多栏下浮动体宽度

- 使用命令`\twocolumn` 即可设置文档为双栏
- 多栏下 `figure` 和 `table` 浮动体只会占用一栏 (宽度为`\columnwidth`)
- 可以使用 `figure*` 和 `table*` 得到跨栏排版的浮动体
- 跨栏排版只允许 `tp` 两个位置参数, 通常会排版到下一页的顶部位置



并排排版

- 在一个浮动体内可以插入多个图片或表格，它们既可以共用一个 caption 也可以各自用不同的 caption
- 也可以将图片与文字并排排版，但通常将文字放在 `\parbox` 或者 `minipage` 环境中，整体作为一个子段盒子处理
 - `\parbox[position]{width}{text}`
 - `\begin{minipage}[position]{width}`
text
`\end{minipage}`



子图表环境

- subcaption 与 subfigure, subtable——一个宏包的两种用法
- `\usepackage{caption,subcaption}`

表: 图表的子标题

图	表
---	---

(a) 文字表格

$\sqrt{2}$	1.414...
$\sqrt{3}$	1.732...

(b) 数字表格



方法一：使用\subcaption（配合 minipage 环境）

```
\begin{table}
  \caption{图表的子标题}
  \begin{minipage}[b]{.5\textwidth}
    \centering
    \begin{tabular}{|c|c|}
      \hline 图 & 表 \\\hline
    \end{tabular}
    \subcaption{文字表格}
  \end{minipage}
\end{minipage}
\begin{minipage}[b]{.5\textwidth}
  \centering
  $\begin{array}{|c|c|}
    \hline \sqrt{2} & 1.414\dots \\\hline
    \sqrt{3} & 1.732\dots \\\hline
  \end{array}$
  \subcaption{数字表格}
\end{minipage}
\end{table}
```



方法二：使用 subtable/subfigure 环境

```
\begin{table}
  \caption{图表的子标题}
  \begin{subtable}[b]{.5\textwidth}
    \centering
    \begin{tabular}{|c|c|}
      \hline 图 & 表 \\\hline
    \end{tabular}
    \caption{文字表格}
  \end{subtable}
  \begin{subtable}[b]{.5\textwidth}
    \centering
    $\begin{array}{|c|c|}
      \hline \sqrt{2} & 1.414\ldots \\\hline
      \sqrt{3} & 1.732\ldots \\\hline
    \end{array}$
    \caption{数字表格}
  \end{subtable}
\end{table}
```



- 1 与 \LaTeX 相遇
- 2 文档
- 3 文章结构
- 4 文本环境举隅
- 5 中文支持
- 6 强大的数学公式
- 7 图表与浮动体
- 8 来做幻灯片**
- 9 自动化工具
- 10 其他



诚如本讲义所展示的，使用 \LaTeX 的 beamer 文档类型可以制作幻灯片。其在语法上与普通的文档别无二致，唯一区别在于组成的基本单位变成了**帧** (frame)。



- 8 来做幻灯片
 - 帧
 - 选择主题
 - 动态展示



一帧即为一页

- 每一页的内容都是以 `\begin{frame}` 开始, 以 `\end{frame}` 结尾
- 如果内容简短, 也可以采用 `\frame{<text>}` 的形式
 - `\frame{\titlepage}` %标题页
 - `\frame{\tableofcontents}` %目录页



标题页

- 与 article 类文档等用法相同，只是相比而言更为详细
- 以本讲义的标题页为例

```
\title[\LaTeX 教程]{\LaTeX ~Tutorials}  
\subtitle{Simple and professional}  
\author[高嵩, 黄秉焜]{Song Gao, Bingkun Huang}  
\institute[南京大学]{Nanjing University}  
\date[2018.10.28]{\today}  
\logo{\includegraphics[width=15pt]{NJU.jpg}}
```

- 方括号内为短标题，显示位置视主题模板而定，大括号内为长标题，显示于标题页



文档层次与目录

- 一般最高为 \section, 其次为 \subsection, 而 \subsubsection 及以下很少使用
- 使用\tableofcontents 生成目录
- 有可选项: hideallsubsection 和currentsection 等
- 与\AtBeginSection[]{} 配合使用

```
\AtBeginSection[]
{
  %\begin{frame}
    \tableofcontents[currentsection,
      hideallsubsections]
  %\end{frame}
}
```



帧标题

- 每一帧都有一个 `frametitle` 和一个 `framesubtitle`
- 标准使用格式是这样的

```
%begin{frame}  
\frametitle{标题}  
\framesubtitle{副标题}  
%内容  
%\end{frame}
```

- 这太麻烦了，可以简写成

```
%\begin{frame}{标题}{副标题}  
%内容  
%\end{frame}
```

- 标题和副标题都是可选项



最常用的环境——itemize

- 这是第一层
 - 这是第二层
 - 这是第三层
 - 没有第四层
 - 超出三层会报错

```
\begin{itemize}
  \item 这是第一层
  \begin{itemize}
    \item 这是第二层
    \begin{itemize}
      \item 这是第三层
      \item 没有第四层
    \end{itemize}
  \end{itemize}
  \item 超出三层会报错
\end{itemize}
```



8 来做幻灯片

- 帧
- 选择主题
- 动态展示



主题

- `\usetheme{ }` 选择主题
- `\usecolortheme{ }` 选择颜色主题
- [这个网页](#)列出了所有 beamer 自带的主题与颜色主题
- `\usefonttheme{ }` 还可以选择字体主题
- 按照国际惯例在 slides 中应使用无衬线字体，但是这种情况下的公式非常丑陋
- `\usefonttheme{professionalfonts}` 提供了解决方案，对于数学环境它将改用有衬线字体，正如前面的例子所示



8 来做幻灯片

- 帧
- 选择主题
- 动态展示



项目依条出现

- 这里给出一条最简单的动画显示的命令 `\pause`



项目依条出现

- 这里给出一条最简单的动画显示的命令 `\pause`
- 得到的效果如此页所示



项目依条出现

- 这里给出一条最简单的动画显示的命令 `\pause`
- 得到的效果如此页所示

```
\begin{itemize}
  \item 这里给出一条最简单的动画显示的命令 \verb|\pause|
  \item 得到的效果如此页所示 \pause
\end{itemize}
```



- 1 与 \LaTeX 相遇
- 2 文档
- 3 文章结构
- 4 文本环境举隅
- 5 中文支持
- 6 强大的数学公式
- 7 图表与浮动体
- 8 来做幻灯片
- 9 自动化工具**
- 10 其他



9 自动化工具

- 页面格式
- 目录
- 交叉引用
- 索引
- 文献引用



页码格式修改的需要

- 扉页，或目录，不应该有页码，或者应该单独计页码数
- 页码位置有要求（页面底部正中间，右下角，右上角）



page 计数器

- 页码有个专门的 page 计数器，每隔一页，计数器加一，然后按照你设置的格式要求，选择是否打印页码，打印在何处，页码字体等等



page 计数器

- 页码有个专门的 page 计数器，每隔一页，计数器加一，然后按照你设置的格式要求，选择是否打印页码，打印在何处，页码字体等等
- 使用 `\pagenumbering` 命令对计数器内的值进行调整
 - gobble: 没有页码
 - arabic: 阿拉伯数字
 - roman: 罗马数字
- 打开 `prac` 里面的 `page.tex` 试试



其他设置

- 页面尺寸
- 分栏
-

一般模板会帮你设置好，有需要再上网查，或者看手册



9 自动化工具

- 页面格式
- 目录
- 交叉引用
- 索引
- 文献引用



生成目录

- tableofcontents
- 需要编译两次
- listoffigures 和 listoftables 命令
- 练习 catalog



目录内容、格式

- 读入目录文件
- 目录、编号的深度和格式
- tocbibind, tocloft, titletoc, minitoc 等宏包的使用 (手添目录, 修改间距, 修改字体, 一二三级子目录标题等等)

需要的时候网上查, 或者看手册



9

自动化工具

- 页面格式
- 目录
- 交叉引用
- 索引
- 文献引用



交叉引用的需求

现在是 8102 年，我正在使用 $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ 写我的黎曼猜想证明……

好，这一步可以从引理 8 推出来，让我康康它在哪一页。

(刷刷刷，鼠标往前滚……)

卧槽，我怎么找不到我的引理 8 了。

(哦，它在第 17 页，写上……)

前面好像不太缜密，我再补补。

太好了，这一步也可以从引理 8 推出来，让我康康它在哪一页。

卧槽，怎么 17 页没有引理 8， $(\odot o \odot)$?

哦，跑到 19 页了……不对，我前面那里也要改 $((\S_ \S))$



简单使用

只需要两步：

```
% 定义标签  
\label{•}
```

```
% 引用标签，ref编号引用，pageref页码引用  
\ref{•}  
\pageref{•}
```

打开练习 CrossReferences



标签的命名规范

- 在做练习的过程中，请注意文本中对图片狮子和公式的命名方式：
`fig:lion`, `eq:1`
- 命名规范非常重要，方便记忆，方便查找，方便分类。

- | | |
|--|--------------------------------------|
| • <code>part</code> : 部分 (part) | • <code>fig</code> : 图 (figure) |
| • <code>chap</code> : 章 (chapter) | • <code>tab</code> : 表 (table) |
| • <code>sec</code> : 节 (section) | • <code>eq</code> : 公式 (equation) |
| • <code>subsec</code> : 小节 (subsection) | • <code>fn</code> : 脚注 (footnote) |
| • <code>subsubsec</code> : 小小节 (subsubsection) | • <code>item</code> : 项目 (item) |
| • <code>para</code> : 段 (paragraph) | • <code>thm</code> : 定理 (theorem) |
| • <code>subpara</code> : 小段 (subparagraph) | • <code>algo</code> : 算法 (algorithm) |



pdf 文件功能上的拓展

pdf 文档支持很多特殊功能，如文档标签，超链接，电子表单等等， \LaTeX 也可以输出这样的功能，比如超链接，而这只需要添加一个宏包。

```
% 在导言区（你最好不要忘了导言区是哪里）
```

```
% 插入下面的宏包
```

```
\usepackage{hyperref}
```

```
% 插入后，交叉引用会自动变为超链接。同样，至少编译两次
```

- 为刚才做的 CrossReferences 添加 hyperref 宏包，增加超链接
- 进入练习 Hyperref 中，猜测理解 hypersetup 中的内容。在 pdf 文件中，依次点击目录，公式标签，网页超链接，文件超链接，以及语句的链接。语句的链接可以注意一下，看看它是用了怎样的命令定位的。

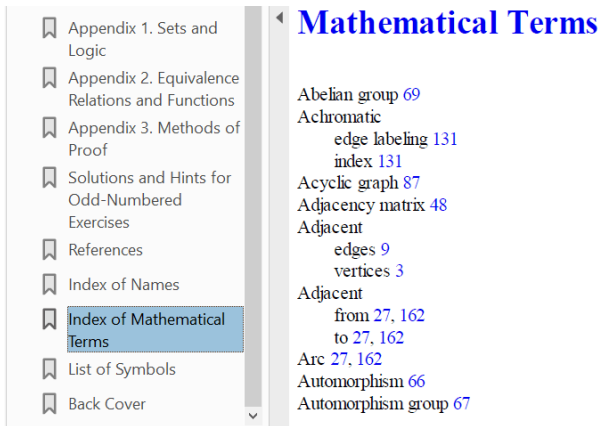


9 自动化工具

- 页面格式
- 目录
- 交叉引用
- **索引**
- 文献引用



其实我觉得很多同学不知道什么叫索引。(快来打脸)



Mathematical Terms

- Abelian group 69
- Achromatic
 - edge labeling 131
 - index 131
- Acyclic graph 87
- Adjacency matrix 48
- Adjacent
 - edges 9
 - vertices 3
- Adjacent
 - from 27, 162
 - to 27, 162
- Arc 27, 162
- Automorphism 66
- Automorphism group 67



步骤

对 tex 文件，你需要做：

导言区使用 `\makeindex`

导言区调用 `makeidx` 宏包

需要索引的地方，使用 `\index{•}` 标记

在放索引的地方（通常是末尾），使用 `\printindex` 命令

编译时，你需要：

```
xelatex file_name  
makeindex file_name  
xelatex file_name
```

注意：

- 配置好你的 `makeindex`
- 不要忘了编译的次数和顺序

进入练习 WordIndex，看看能不能编译出两个 tex 文件的索引



9 自动化工具

- 页面格式
- 目录
- 交叉引用
- 索引
- 文献引用



这块不细讲了，主要是编译的顺序

```
xelatex filename  
bibtex filename  
xelatex filename  
xelatex filename
```

- 在练习 bib 中按照上面的编译顺序，看看能否生成样例中的文件
- 在 tex 文件中用的指令，可去网上具体了解。
- 引用可以在 tex 文件中自己写，但用的比较多的是写在 bib 文件中 (有专门的管理软件)，最后导入，生成引用



- 1 与 \LaTeX 相遇
- 2 文档
- 3 文章结构
- 4 文本环境举隅
- 5 中文支持
- 6 强大的数学公式
- 7 图表与浮动体
- 8 来做幻灯片
- 9 自动化工具
- 10 其他



小建议

- 要学会自学！自学！自学！
- 学会看错误日志（红色的那个），**静下心来读英文**（其实不少人看到红色的字符串就不想认真读，然后自己肉眼挑 BUG，或者问别人），自己解决不了的时候，一个很好的办法，是复制你的错误信息，以及你刚做的事的关键词（比如某个指令名），贴到搜索框搜索（当然是英文）



小建议

- 要学会自学！自学！自学！
- 学会看错误日志（红色的那个），**耐下心来读英文**（其实不少人看到红色的字符串就不想认真读，然后自己肉眼挑 BUG，或者问别人），自己解决不了的时候，一个很好的办法，是复制你的错误信息，以及你刚做的事的关键词（比如某个指令名），贴到搜索框搜索（当然是英文）
- 对于上面这点，举个小例子：`"!Undefined control sequence."`，这是一个伴随你一生的报错，绝大部分都可归为以下三种原因的一种：
 - 拼错了！
 - 没导入宏包！
 - 用了不存在的指令！



小建议

- 另一位主讲人想要分享他昨晚（今天凌晨）的惨痛教训
- 遇到报错信息
Beamer - ! File ended while scanning use of \next
- 尝试注释法肉眼找 bug 半小时无果
- 网上一查，错误原因是这样的

The fix was 2-part

- `\end{frame}` can NOT be indented
- `\end{frame}` cannot have any comments directly after it

Changing this line fixed the error.

- 最后发现是在某个`\end{frame}` 后加了一个空格



小建议

- 基础设施很重要！抽个时间，去专门学习你使用的编辑器是很有必要的。
- 关于上面这点，做些解释：
 - 使用 Tab 键把你正在写的指令瞬间补全，也许就能省下百分之十的时间，积累起来就能让你多活一天！所以找个时间把常用的指令录进编辑器的代码补全里面是不是很有必要？（当然，花时间去网上找拓展包也是一样的）
 - 再比如，自定义一个宏编译，设置为 `xelatex + bibtex + xelatex + xelatex`，这样你按一个键就能完成引用的生成，是不是觉得很舒服？
 - 夜晚看屏幕伤眼，心里也难受，所以换一个护眼又好看的主题皮肤是不是很好？



小建议

- 基础设施很重要！抽个时间，去专门学习你使用的编辑器是很有必要的。
- 关于上面这点，做些解释：
 - 使用 Tab 键把你正在写的指令瞬间补全，也许就能省下百分之十的时间，积累起来就能让你多活一天！所以找个时间把常用的指令录进编辑器的代码补全里面是不是很有必要？（当然，花时间去网上找拓展包也是一样的）
 - 再比如，自定义一个宏编译，设置为 `xelatex + bibtex + xelatex + xelatex`，这样你按一个键就能完成引用的生成，是不是觉得很舒服？
 - 夜晚看屏幕伤眼，心里也难受，所以换一个护眼又好看的主题皮肤是不是很好？
- 熟能生巧，可以渐渐用 latex 代替 word 去做平时的作业，可能一开始会有点慢，但当你渐渐熟悉自己的几个模板之后，速度就能反超 word，还更好看





欢迎大家报名下一届学术软件教学系列讲座主讲人!



资源分享

- <https://www.sharelatex.com/>
- <https://tex.stackexchange.com/>
- <http://tug.ctan.org/tex-archive/> 找文档, 找手册!
镜像站: <http://mirrors.ustc.edu.cn/CTAN/>
- <http://tug.org/>

