latex 使用指南

摘要： 对 latex 中高频使用的语法进行记录

关键字:

目录

一、 文档结构组织

1.1 input vs. include

文档结构组织主要是在应对需要编译的 latex 文档过长时需要面临的问题, 这里结合文档较长时存在的问题来分析`\input`和`\include`命令:

- 文档较长时，tex 代码较长，文本编辑器的功能会大打折扣。这时很自然地想要将文档拆分成多个 tex 文件，在编译时同时引入，这时就可以使用`\input{`文件名`}`命令，扩展名 (.tex) 可省略，**支持嵌套**
- 文档较长,当进行细微调整时编译时间较长,这时可以考虑结合使用`\includeonly{`文件列表`}`和 `\include{`文件名`}`命令[1]，这么做的好处是能提高速度——对于只存在于文件列表内的文件，latex 只会加载其对应的章节标题等信息，并不会将文档内容引入，这能保证对单个文档进行修改时仍能保持和整体文档一致的效果，大大提高了编译速度，不过需要注意的是`\include{}`命令**不可以嵌套使用**

1.2 插入 pdf

latex 插入 pdf 时的两种常用方法及其优缺点表述如下:

- *graphicx* 是对 *graphics* 宏包的扩展或者增强，两个宏包的参数格式有细微差别，后者更多是对前者的补充。因为可以将 pdf 使用`\includegraphics`引入，所以可以将其放置在浮动体或者直接插入到文档中，其相当于将 pdf 当做源文件的一部分插入在文档中，对其进行修改或者加以说明都会更方便一些，就个人体验而言更实用一些。
- *pdfpages* 提供了很多选项来插入 PDF 文件, 其在文档中写到避免产生类似 *graphicx* 插入 pdf‘Overfull `\hbox`’and ‘Overfull `\vbox`’warnings 的提示。自己在使用过程中的体验是该宏包适合用来 pdf 作为一个**独立的页面**插入到源文档中的情况，提供了丰富的命令对插入 pdf 的大小和原页面的大小进行调整，但是如果想对插入的 pdf 进行局部的调整或者加一个标题之类的操作，相对来说明比较麻烦且排版效果也比较差。

如想将导入的 pdf 双栏排列，建议使用 *graphicx* 宏包，双栏排列方法见小节:双栏排列

二、 浮动体环境

一般来说，为了排版的美观，都会将表格图片等内容放入浮动体环境，这块主要是针对插入的图片或者表格过大的情况作说明。

[1] includeonly 是导言区命令，如单独使用 include，相当于 input 命令，不过会在插入 tex 代码前后都有换页符

2.1 插入图片

图片过大的问题相对容易解决，只需要直接在修改`width`的可选参数即可, 引用图片1

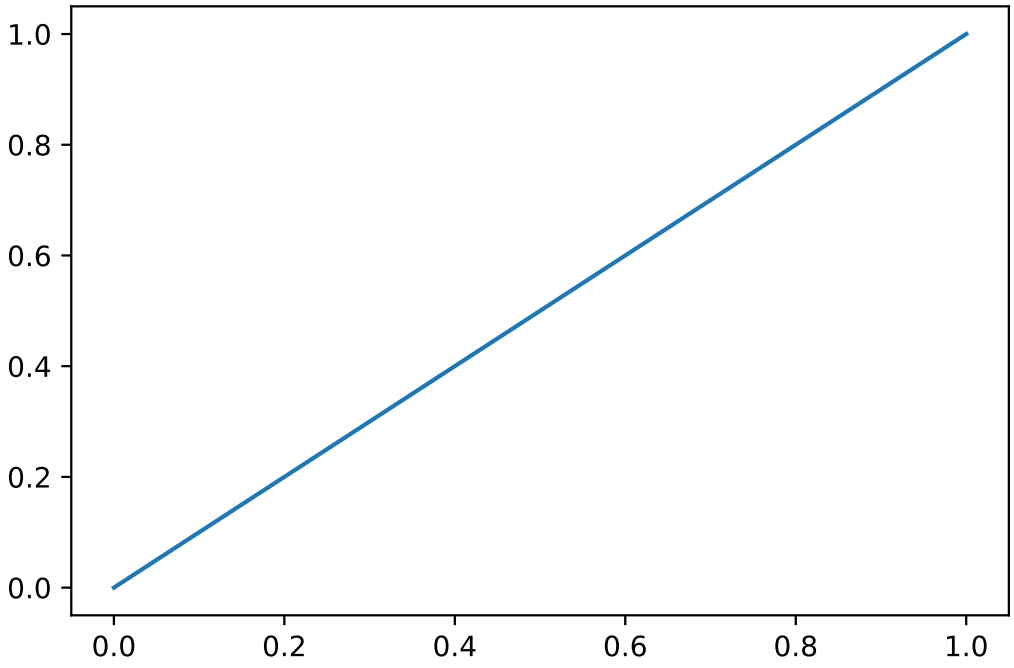
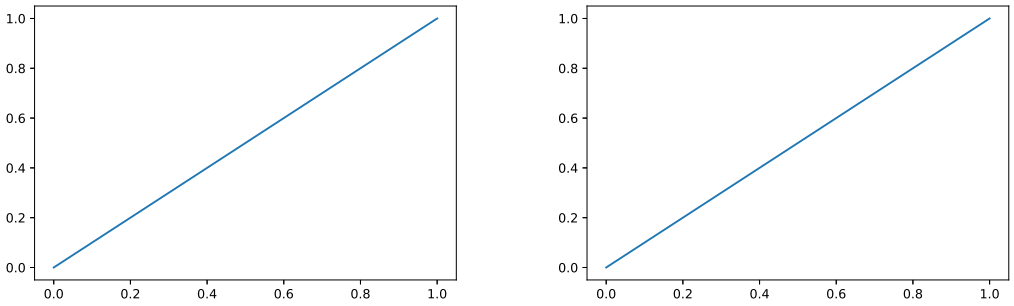


图 **1**　全国煤炭企业前 **50** 强产煤总量

2.2 双栏排列

在引入 pdf 或者图片时 (insert graphics)，为节省页面空间，很多时候将两张图片进行双栏并排显示是一种不错的选择，这也就使用到了浮动体的双栏, 这里给出一种比较简单的浮动体双栏排列的方法：

图 **2**　两个图并排放置



(a) 题注 **1**　　**(b)** 题注 **2**

2.3 插入表格

一般来插入的表格都可以通过 python 的`DataFrame.to_latex()`函数来导出到指定位置，然后通过`input`命令导入使用，经常会用到的导出模板为:

```
df.to_latex('./排版/tex代码/cr8.tex',caption='（$CR_8$）计算结果',label='table',header=False)
```

只需微调函数的参数即可，注意如下:

- 列名如果有数学公式的话导出会加转义符，这样在 latex 编译后不能正常显示。可以设置`escape=False`解决这个问题，不过这样设置数值中的转义符号也会被取消掉，这点在对于数值列中含有”%” 的数据中需要注意
- 不想显示 index 列或者列名列可以通过调节`index`和`header`参数进行修改

插入表格的情况，相对复杂，可能出现的问题包括引入的表格过宽和过长的问题。一个简单的例子如表1:

表 **1**　**A simple Table**

A	B
1	a
2	b
3	c
4	d

2.3.1 宽表格

针对插入表格**过宽**的情况

1. 一般选择使用`graphicx`宏包提供的`\resizebox{h-length%l}{v-length%l}{text}`命令来解决。这里给一个示例，语法如表2所示（代码在`tex`代码文件夹的`widetable`里边。

表 **2**　近十年行业集中率（CR_8）计算结果

年份	2011	2012	2013	2014	2015	2016	2017	2018	2019	2020	2021
行业集中率 (CR_8)	0.288718	0.301209	0.316948	0.331046	0.360429	0.359104	0.346434	0.40707	0.404583	0.477353	0.49315

2. 除了缩小表格内容以外，对于宽度溢出不太严重的表格，也可以选择修改水平排版开始位置，这也能使得本来溢出的表格可以被放下，而且也不会导致字体看起来太小（示例如表3所示）

表 **3**　近十年行业集中率（CR_8）计算结果

年份	2011	2012	2013	2014	2015	2016	2017	2018
行业集中率 (CR_8)	0.288718	0.301209	0.316948	0.331046	0.360429	0.359104	0.346434	0.40707

2.3.2 长表格

长表格主要使用的是`longtable`宏包的 longtable 环境，与 tabular 类似，不过多了一些处理跨页操作的命令。因为`to_latex`函数可以直接导出这种样式的表格，所以在实际中使用这种表格的频率也比较高, 想要导出这种环境的表格时，只需在开始提到的模板中加入`longtable=True`的参数即可获得，使用比较方便[2]。

表 **4**　`longtable` 环境中的命令汇总

环境的水平对齐可选项	
留空	表格居中[3]
`[c]`	表格居中
`[l]`	表格左对齐
`[r]`	表格右对齐

接下一页表格……

[2]longtable 的表格宽度是根据表格内容调整的，当表格属于长而窄的类型时美观度比较有限，这时可以考虑利用 python 的 concat 函数对长表格进行拆分和列拼接

[3]实际上，留空的对齐方式是由一组命令控制的，参见宏包文档。

（续表）

结束表格一行的命令	
\\	普通的结束一行表格
\\[⟨ 距离 ⟩]	结束一行，并增加额外间距
*	结束一行，禁止在此分页
\kill	当前行不输出，只参与宽度计算
\endhead	此命令以上部分是每页的表头
\endfirsthead	此命令以上部分是表格第一页的表头
\endfoot	此命令以上部分是每页的表尾
\endlastfoot	此命令以上部分是表格最后一页的表尾
标题命令	
\caption{⟨ 标题 ⟩}	生成带编号的表格标题
\caption*{⟨ 标题 ⟩}	生成不带编号的表格标题
分页控制	
\newpage	强制分页
\pagebreak[⟨ 程度 ⟩]	允许分页的程度（0–4）
\nopagebreak[⟨ 程度 ⟩]	禁止分页的程度（0–4）
脚注控制	
\footnote	使用脚注[4]，注意不能用在表格头尾
\footnotemark	单独产生脚注编号，不能用在表格头尾
\footnotetext	单独产生脚注文字
长度参数	
\LTleft	对齐方式留空时，表格左边的间距，默认为 \fill
\LTright	对齐方式留空时，表格右边的间距，默认为 \fill
\LTpre	表格上方间距，默认为 \bigskipamount
\LTpost	表格下方间距，默认为 \bigskipamount
\LTcapwidth	表格标题的宽度，默认为 4 in

2.3.3 定宽表格

刘海洋老师的书里有写到，可以去翻翻。用的是tabularx环境。考虑到这种环境不能使用 Python 的to_latex直接导出，更推荐的一种做法是在 python 中将表格进行调整之后再导出。

[4]普通表格中不能用。

三、 插入题注

这是一个题注[5]。

四、 导入参考文献

参考文献的导入主要是使用 zotero 先导出 bib 文件。引文格式可能需要根据不同的使用场景进行调整。

4.1 biblatex 宏包

biblatex宏包提供了比较全面的中文参考文献导入支持，需要注意的是在编译 bib 文件时默认使用biber后端，同时在生成和导入参考文献时与其它宏包有所区别:

```
\documentclass{ctexart}
% 使用符合 GB/T 7714-2015 规范的参考文献样式
\usepackage[style=gb7714-201,backend=bibtex]{biblatex}
% 注意加 .bib 扩展名
\addbibresource{egbibdata.bib}
\begin{document}
见文献\cite{caimin2006}。
\printbibliography%插入参考文献
\end{document}
```

4.2 中文文献的自定义导入

对于中文文献，可以使用gbt7714宏包来生成满足我国标准的参考文献格式:

```
\documentclass{ctexart}
\usepackage{gbt7714}
\bibliographystyle{gbt7714-numerical}
\begin{document}
\cite{...}
...
\bibliography{bibfile}
\end{document}
```

[5]使用比较简单，这里不再赘述

这种宏包导入有一个比较麻烦的是作者在参考文献中出现的位置问题，经测试该库不会自动按照 bib 文件中作者的顺序排列，在多位作者同时完成一篇文章时排序混乱，官方文档提供了一种解决方案，但是比较麻烦，推荐使用biblatex宏包。gbt7714宏包的好处是它对中文文献的引用格式和参考文献的样式有比较好的支持。

按照国标的规定，参考文献的标注体系分为“顺序编码制”和“著者-出版年制”。用户应在导言区调用宏包 gbt7714，并且使用 \bibliographystyle 命令选择参考文献表的样式，比如：

```
\bibliographystyle{gbt7714-numerical} %顺序编码制
或者
\bibliographystyle{gbt7714-author-year} %著者-出版年制
```

此外还可以使用 2005 版的格式 gbt7714-2005-numerical 和 gbt7714-2005-author-year。另外几点注意：

- 可以通过\citestyle{ citation style }命令来改变引用位置的显示格式，可选参数为 super, numbers, author-year, 在顺序编码制下默认引用格式为上角标（sub）.
- 同一位置引用多篇文献时,如果想要在引用位置自动对文献进行排序则需要将sort&compress传递给宏包的可选参数

  ```
  \usepackage[sort&compress]{gbt7714}
  ```

注意该宏包不要和 cite 宏包一同引入，会起冲突。

LaTeX 中更为常用的一种导入参考文献的格式，其中unsrt样式表示参考文献条目按照文章中出现顺序排列。

```
\bibliographystyle{unsrt}
\bibliography{ref}
```

该方法下在引用位置处显示是类似”[4]” 的形式。

附录 A　插入代码

```
data<-read.csv('./data.csv')
names(data)
# model<-lm(log(单价)~.-所属区-朝向-小区,data)
shapiro.test(log(data$单价))
shapiro.test(data$单价)
model <- lm(log(单价) ~ ., data)
summary(model)
sel_model <- step(model)
summary(sel_model)
# 多重共线性
library(car)
library(lmtest)
resettest(sel_model, power = 2)
library(lmtest)
bgtest(sel_model)
bptest(model)
library("ivreg")
model <- ivreg(log(单价) ~ 容积率 + 总楼栋数 + 房龄 + 客厅数 + 卫生间数 + 总楼层+绿化率 +
    小区参考均价 | 容积率 + 总楼栋数 + 房龄 + 客厅数 + 卫生间数 + 总楼层+绿化率 + 交易价值 +
    居住品质, data = data)
summary(model)
```

1.1　input 方法

更为简洁的一种导入方法

```
print("hello Python")
```