

CGD-QP-001

内部

# 公司文档管理办法 (试行)



苏州珂晶达电子有限公司



标    题	公司文档管理办法: (试行)		
编    号	CGD-QP-001	密    级	内部
类    别	质量流程	状    态	试行
关联文档			
责 任 人	赵军	客户单位	
联系方式	zhaoj@cn.cogenda.com	联系方式	
签    署		会    签	
日    期		日    期	

摘要

第一部分为公司文档的管理和排版的要求。第二部分为技术内容和对要求的解释，包括排版软件和排版模板的使用介绍、表格、图片以及代码的排版方式。

修订记录

版本	日期	负责人	备注
0.1	2013.12.25	赵军	a draft
0.2	2014.01.04	沈忱	translate .tex to .cls
0.3	2014.01.10	赵军	refine



# 目录

第一章 要求	1
1.1 适用范围	1
1.2 排版要求	1
1.3 文件存档	1
1.3.1 应有文件	1
1.3.2 存档路径	1
1.3.3 文档编号规则	2
1.3.4 目录名和文件名	2
1.3.5 入档	2
第二章 排版软件及模板的使用方法	3
2.1 软件的安装和使用	3
2.1.1 在服务器上使用	3
2.1.2 在自己电脑上安装	4
2.1.3 Lyx	4
2.2 模板的安装和使用	4
第三章 模板的格式要求	6
3.1 表格	6
3.2 图片	7
3.3 代码	7
3.4 其他格式要求	11
3.4.1 几个字体宏	11
3.4.2 Item 列表	12
第四章 对某些对象的排版要求	13
4.1 用户自己增加宏包	13
4.2 交叉引用和参考文献	13
4.2.1 交叉引用	13

4.2.2	参考文献及引用 . . . . .	13
4.3	数学环境 . . . . .	14
4.4	算法 . . . . .	15

# 代码

2.1	在服务器上设置使用 <b>TexLive</b> 的环境	3
2.2	安装字体	4
2.3	安装 <b>TexLive</b>	4
2.4	安装模板	5
2.5	模板示例	5
3.1	<b>ctable</b> 语法	6
3.2	<b>ctable</b> 示例	6
3.3	三线表的一般 <b>L<sup>A</sup>T<sub>E</sub>X</b> 语法示例	6
3.4	插入图片语法示例	7
3.5	<b>Pascal</b> 代码示例	7
3.6	插入代码的语法	7
3.7	插入代码示例	8
3.8	引入代码文件示例	8
3.9	<b>C</b> 代码示例	8
3.10	<b>Python</b> 代码示例	9
3.11	<b>bash</b> 代码示例	9
3.12	保留段落间距的环境变量	12
4.1	参考文献格式示例	13





# 第一章 要求

为规范公司文档的管理，特制定本办法。目前为试行阶段，欢迎意见和建议。

## 1.1 适用范围

本管理办法适用于所有权归公司的、密级为“内部”及“公开”的正式文档。包括但不限于：

- 公司对外的正式报告，比如技术方案、技术总结及操作手册等。
- 公司对外的书面交流文档。
- 公司对内的质量流程、开发文档以及有存档价值的过程文档等。

不属于此办法管理范围的文档：

- 密级为“秘密”及以上的文档。
- 无存档价值的交流文档。

## 1.2 排版要求

公司要求以后的正式文档尽可能用  $\text{\LaTeX}$  排版，并使用公司统一的模板。具体要求和实现方法见本文档后面的部分。

## 1.3 文件存档

### 1.3.1 应有文件

每一份文档对应的不是一个文件，而是一个目录。这个目录中应有两个文件：

1. 最终的 pdf 文件。
2. 一个 zip 或 gzip 包。包中有生成 pdf 文件需要的所有 tex 文件、图片文件及其他文件如插入代码、原始数据等。

注意事项：

1. 如果这个文档是用 lyx 软件编辑的，则应在 zip/gzip 压缩包中包括原始的 lyx 文件。
2. 如果文档中的插图是用 LibreOffice 等软件制作的，应在 zip/gzip 压缩包中包括 odt/odp 等文件。
3. 如果生成 pdf 文件需要特殊的操作步骤，或有其他需要注意的事项，应当在 README 文件中加以说明。

### 1.3.2 存档路径

本办法管理的文档存在公司服务器上，全公司可读。总路径为：/home/public/document。以本文档为例，其完整的目录为：/home/public/document/procedure/CGD-QP-001/V01/。

其三层路径名：

- **procedure** 表示本文档的类型为质量流程，
- **CGD-QP-001** 为本文档的编号，
- **V01** 表示其版本。

这个路径里面有两个文件，一个 **pdf** 文件，一个原始文件的 **gzip** 包。

### 1.3.3 文档编号规则

文档编号为 **CGD-XX-xxx**，其中 **CGD** 表示 **Cogenda**，**XX** 为字母，表示文档类型。**xxx** 为数字，为本文档的序号。编号 **XX** 对应的类型及其存放路径为：

表 1.1: 文档类型对应的路径

XX	文档类型	存放路径
TP	技术方案	proposal
TR	技术总结	report
MN	操作手册	manual
QP	质量流程	procedure
MS	其他文档	miscellaneous

### 1.3.4 目录名和文件名

每个文档对应的目录名是文档的完整编号，不可以用其他内容。文档内的文件名可以自由命名，但只能由英文字母、数字、下划线（**\_**）或英文句点（**.**）组成，不可以出现其他字符。

### 1.3.5 入档

文件入档流程：

1. 如果还没有文档编号，请先跟公司文档管理人员申请编号，填入文档中。
2. 在 **helium** 下测试一下是否能顺利编译生成 **pdf** 文档。
3. 将所有文件拷给公司文档管理人员。

## 第二章 排版软件及模板的使用方法

公司在试用了多个排版软件之后，决定选用  $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ ，并使用统一的模板。 $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$  的工作方式为：用户编辑的文档为纯文本文件，配上用户需要的图片文件（或者原始数据文件），用  $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$  软件处理成为 ps 或 pdf 文件。如图 2.1 所示。用户对版式的要求全部写在文本文件中，也就是用户需要了解一些  $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$  的排版语法。

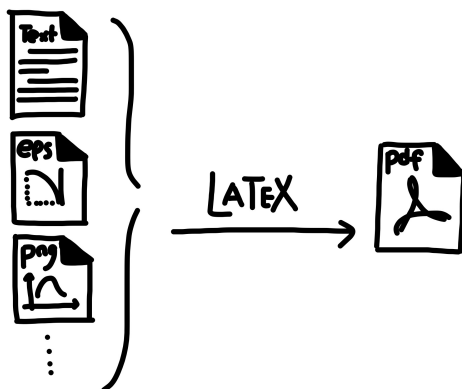


图 2.1:  $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$  工作方式

公司选用  $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$  并采用统一的模板，好处为：

- 统一公司文档的风格。
- 方便不同的人在不同平台下编辑和修改同一个文档。
- 对于大型文档， $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$  崩溃的概率小于其他常见排版软件；文本格式的文件损坏的概率也小。
- $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$  的排版功能非常强大，排出的版面漂亮。
- $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$  免费。

坏处是用户编辑  $\text{T}_{\text{E}}\text{X}$  文本文档时不够直观。对此公司的建议顺序是：

1. 使用纯文本编辑器编辑  $\text{T}_{\text{E}}\text{X}$  文档。（用户可能需要一个  $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$  手册 [1]。）
2. 用  $\text{lyx}$  编辑。
3. 其他可输出  $\text{T}_{\text{E}}\text{X}$  文档的软件。要求生成的  $\text{T}_{\text{E}}\text{X}$  文档适于人工阅读。

### 2.1 软件的安装和使用

目前我们公司使用的  $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$  发行包为  $\text{TexLive}$ ，你可以直接使用公司  $\text{helium}$  服务器上的，也可以在自己的电脑上安装一份来用。以下分别介绍。

#### 2.1.1 在服务器上使用

只需要设置使用  $\text{TexLive}$  软件所需的环境。

```
1 # Set the path environment of TexLive
```

```

2 source /usr/local/texlive/setenv.sh
3 # Or to make it permanent, you can simply do:
4 #   cat /usr/local/texlive/setenv.sh >> $HOME/.bash_profile
5 # test it
6 which xelatex

```

代码 2.1: 在服务器上设置使用 TexLive 的环境

### 2.1.2 在自己电脑上安装

目前我们使用的 TexLive 是个完整的庞大的包, 其 iso 镜像文件 2.4G, 安装后需要 3.5G 的硬盘空间。安装前请在你自己的硬盘上准备充分的空间。镜像文件为: `/home/public/software/tex/texlive2013/texlive2013-*.iso`。对于 Linux 系统, 可能需要先几个字体文件。字体文件为 `/home/public/software/tex/texlive2013/win_fonts.tar.gz`。

假设你准备把字体安装在 `/usr/share/fonts/TTF/`。安装方法为:

```

1 # copy the font files
2 tar -xvzf win_fonts.tar.gz && sudo cp -r win_fonts /usr/share/fonts/TTF/
3 # activate the fonts
4 sudo fc-cache -fv

```

代码 2.2: 安装字体

假设你准备把 TexLive 安装在 `/usr/local/tex`。安装方法为:

```

1 # mount the iso file
2 sudo mount -o loop texlive2013-20130530.iso /mnt/dvd && cd /mnt/dvd
3 # install it
4 export TEXLIVE_INSTALL_PREFIX=/usr/local/tex
5 perl install-tl
6 # then follow the instructions to set the path environment
7 export PATH=/usr/local/texlive/2013/bin/x86_64-linux:$PATH

```

代码 2.3: 安装 TexLive

对于 Windows 用户, 以上镜像文件也支持 Windows。请阅读镜像文件里的说明文件。

### 2.1.3 Lyx

Lyx 是一个直观的编辑  $\text{T}_{\text{E}}\text{X}$  文件的编辑器。不需要 lyx 的人可以跳过这一节。服务器上已经安装了 lyx, 你可以直接使用, 也可以在自己电脑上安装。

在安装模板之前, 需要先启动 lyx 一次以生成 `$HOME/.lyx`。安装模板之后, 启动 lyx 并点击 **Tools** ► **Reconfigure**, 然后重新启动 lyx。

## 2.2 模板的安装和使用

公司的技术报告模板是放在用户自己目录下的, 如此即使用户是在服务器上运行 TexLive, 也需要安装模板。模板的路径: `/home/public/document/template/cgdrep.tar.gz`。安装方法:

```
1 # Unpack the template
2 tar xf /home/public/document/template/cgdrep.tar.gz
3 cd cgdrep
4 # Install the template files
5 ./install.sh
```

代码 2.4: 安装模板

没有报错的话，可以使用  $\text{\LaTeX}$  模板里面的范例了：

```
1 # test the LaTeX example
2 cd example/latex
3 make
4 # There should be several new files and one of them is main.pdf
5 evince main.pdf &
```

代码 2.5: 模板示例

范例中的 `make` 命令编译了四遍，这可能需要些时间。用户如果想快速看到结果的话，可以只编译一遍，命令为 `xelatex main`。这样得到的 `pdf` 结果并不完整：可能没有包含参考文献，交叉引用可能不正确。要得到最后的正式 `pdf` 文件，还是需要 `make` 一下的。

## 第三章 模板的格式要求

我们暂定了表格、图片、代码和其他一些对象的格式要求，并写了一些宏放在模板里。目前这些格式要求并不是最终版，欢迎提出意见。

一般，每一个表格（图片、代码）都当有一对 `Caption` 和 `Label`。`Caption` 是这个表格（图片、代码）的名字，其字号比正文小半号。`Label` 是引用它时用的标识，`Label` 本身的内容并不出现在排版后的 pdf 页面上。以下分别讲如何实现表格、图片和代码格式的实现方法。

### 3.1 表格

用户没有特别需要的话，我们希望表格都采用三线表，首行为其 `header`，用黑体。字号比正文小半号，位置居中。如表 3.1:

表 3.1: 三线表

版本	日期	负责人	备注
1.0	2013.12.24	沈忱、纪冬梅	初稿
1.1	2013.12.25	赵军	二稿

实现这样的表及其浮动位置，可以用公司自己制作的宏 `ctable`，其语法为：

```
1 \begin{ctable}{Label}{Caption}{Alignments}
2  第一行 (header)  \ \ \hline
3  第二行 \ \
4  .....
5 \end{ctable}
```

代码 3.1: ctable 语法

其中 `Alignments` 是列对齐方式，四列左对齐为 `{llll}`。则表 3.1 的完整语法为：

```
1 \begin{ctable}{triline}{三线表}{llll}
2  版本 & 日期      & 负责人      & 备注 \ \ \hline
3  1.0  & 2013.12.24 & 沈忱、纪冬梅 & 初稿 \ \
4  1.1  & 2013.12.25 & 赵军        & 二稿 \ \
5 \end{ctable}
```

代码 3.2: ctable 示例

如果你不想局限于 `ctable` 的功能，也可以用一般的  $\text{\LaTeX}$  表格语法（模板包含了 `tabu` 宏包），表 3.1 的一般  $\text{\LaTeX}$  语法为：

```
1 \begin{table}[htbp]\caption{\label{triline} 三线表}
2 \centering\small\begin{tabu}{llll}\thickhline\rowfont{\bfseries}
3  版本 & 日期      & 负责人      & 备注 \ \ \hline
4  1.0  & 2013.12.24 & 沈忱、纪冬梅 & 初稿 \ \
```

```

5 1.1 & 2013.12.25 & 赵军 & 二稿 \\
6 \thickhline\end{tabu}\end{table}

```

代码 3.3: 三线表的一般 L<sup>A</sup>T<sub>E</sub>X 语法示例

## 3.2 图片

插入图片的语法简单，这里没有设置自己的宏，而是用的是普通语法：

```

1 \begin{figure}[htbp]\centering
2 \includegraphics[width=0.5\textwidth]{file.jpg}
3 \caption{\label{Label}Caption}
4 \end{figure}

```

代码 3.4: 插入图片语法示例

对于图片的大小控制，当用户没有特别需求的时候，建议采用 `textwidth` 的倍数方式。

## 3.3 代码

这里先给出一段文档中插入 Pascal 代码的样子：

```

1 for i:=maxint to 0 do
2 begin
3 { do nothing }
4 end;
5 Write ( 'Case insensitive ' );
6 Write ( 'Pascal keywords .' );

```

代码 3.5: Pascal 代码示例

插入代码的 L<sup>A</sup>T<sub>E</sub>X 语法为：

```

1 \begin{lstlisting}[language=Language,caption={Caption},label=Label]
2 .....
3 ( 代码内容 )
4 .....
5 \end{lstlisting}

```

代码 3.6: 插入代码的语法

其中 `Language` 为这一段代码的语言。支持的语言包括 ABAP、ACSL、Ada、Algol、Ant、Assembler、Awk、bash、Basic、C、C++、Caml、Clean、Cobol、Comal、csh、Delphi、Eiffel、Elan、erlang、Euphoria、Fortran、GCL、Gnuplot、Haskell、HTML、IDL、inform、Java、JVMIS、ksh、Lisp、Logo、make、Mathematica、Matlab、Mercury、MetaPost、Miranda、Mizar、ML、Modelica、Modula-2、MuPAD、NASTRAN、Oberon-2、OCL、Octave、Oz、Pascal、Perl、PHP、PL/I、Plasm、POV、Prolog、Promela、Python、R、Reduce、Rexx、RSL、Ruby、S、SAS、Scilab、sh、SHELXL、Simula、SQL、tcl、TeX、VBScript、Verilog、VHDL、VRML、XML、XSLT。更多信息请搜索关键词 `lstlisting`。代码 3.5 的引用方法为：

```

1 \begin{lstlisting}[language=Pascal,caption={Pascal 示例代码},label=PascalExample]
2 for i:=maxint to 0 do
3 begin
4 { do nothing }
5 end;
6 Write ( 'Case insensitive ' );
7 Write ( 'Pascal keywords.' );
8 \end{lstlisting }

```

代码 3.7: 插入代码示例

目前暂定代码的文字大小为 `\footnotesize`。如果用户需要调整,可以在引用前修改字体,比如大半号为 `\small`,其语法为:

```
1 \lstset{basicstyle=\small\ttfamily}
```

这个命令将作用于后面所有的代码。如果这并不是用户想要的,则需要修改回来:

```
1 \lstset{basicstyle=\footnotesize\ttfamily}
```

同理,用户也可以一次设定语言:

```
1 \lstset{language=Language}
```

则后面每次插入代码时候就不用加入 `language=Language` 这一句了。更详细的用法请搜索关键词 `lstlisting`。

如果代码比较长,在某个原始代码文件里,可以用如下语法,将整个文件的代码引入:

```
1 \lstinputlisting[language=Language,label=Label,caption={Caption}]{Filename}
```

代码 3.8: 引入代码文件示例

下面给出 C、Python 和 Bash 代码示例,欢迎提出排版意见。

```

1 #include <stdio.h>
2 #define N 10
3 /* Block
4  * comment */
5
6 int main()
7 {
8     int i;
9
10    // Line comment.
11    puts("Hello world!");
12
13    for (i = 0; i < N; i++)

```



```

14     {
15         puts("LaTeX is also great for programmers!");
16     }
17
18     return 0;
19 }

```

代码 3.9: C 代码示例

```

1 class BankAccount(object):
2     def __init__(self, initial_balance=0):
3         self.balance = initial_balance
4     def deposit(self, amount):
5         self.balance += amount
6     def withdraw(self, amount):
7         self.balance -= amount
8     def overdrawn(self):
9         return self.balance < 0
10 my_account = BankAccount(15)
11 my_account.withdraw(5)
12 print my_account.balance
13 import unittest
14 def median(pool):
15     copy = sorted(pool)
16     size = len(copy)
17     if size % 2 == 1:
18         return copy[(size - 1) / 2]
19     else:
20         return (copy[size/2 - 1] + copy[size/2]) / 2
21 class TestMedian(unittest.TestCase):
22     def testMedian(self):
23         self.failUnlessEqual(median([2, 9, 9, 7, 9, 2, 4, 5, 8]), 7)
24 if __name__ == '__main__':
25     unittest.main()

```

代码 3.10: Python 代码示例

```

1 #!/bin/sh
2 # renna: rename multiple files according to several rules
3 # written by felix hudson Jan - 2000
4
5 #first check for the various 'modes' that this program has
6 #if the first ($1) condition matches then we execute that portion of the
7 #program and then exit
8
9 # check for the prefix condition
10 if [ $1 = p ]; then
11
12 #we now get rid of the mode ($1) variable and prefix ($2)

```

```
13 prefix=$2 ; shift ; shift
14
15 # a quick check to see if any files were given
16 # if none then its better not to do anything than rename some non-existent
17 # files!!
18
19 if [ $1 = ]; then
20     echo "no files given"
21     exit 0
22 fi
23
24 # this for loop iterates through all of the files that we gave the program
25 # it does one rename per file given
26 for file in $*
27 do
28     mv ${file} $prefix$file
29 done
30
31 #we now exit the program
32 exit 0
33 fi
34
35 # check for a suffix rename
36 # the rest of this part is virtually identical to the previous section
37 # please see those notes
38 if [ $1 = s ]; then
39     suffix=$2 ; shift ; shift
40
41     if [ $1 = ]; then
42         echo "no files given"
43         exit 0
44     fi
45
46 for file in $*
47 do
48     mv ${file} $file$suffix
49 done
50
51 exit 0
52 fi
53
54 # check for the replacement rename
55 if [ $1 = r ]; then
56
57     shift
58
59 # i included this bit as to not damage any files if the user does not specify
```

```
60 # anything to be done
61 # just a safety measure
62
63 if [ $# -lt 3 ] ; then
64     echo "usage: renna r [expression] [replacement] files... "
65     exit 0
66 fi
67
68 # remove other information
69 OLD=$1 ; NEW=$2 ; shift ; shift
70
71 # this for loop iterates through all of the files that we give the program
72 # it does one rename per file given using the program 'sed'
73 # this is a single command line program that parses standard input and
74 # replaces a set expression with a give string
75 # here we pass it the file name ( as standard input) and replace the nessesary
76 # text
77
78 for file in $*
79 do
80     new='echo ${file} | sed s/${OLD}/${NEW}/g'
81     mv ${file} $new
82 done
83 exit 0
84 fi
85
86 # if we have reached here then nothing proper was passed to the program
87 # so we tell the user how to use it
88 echo "usage;"
89 echo " renna p [prefix] files.."
90 echo " renna s [suffix] files.."
91 echo " renna r [expression] [replacement] files.."
92 exit 0
93
94 # done!
```

代码 3.11: bash 代码示例

## 3.4 其他格式要求

### 3.4.1 几个字体宏

有几个特别对象，要在文档里用特别的字体来，我们定义了几个字体的宏，如表 3.2 所列。文档中用到这样的对象时，请采用表中的字体宏。

表 3.2: 几类特别对象

对象	语法示例	效果示例
文件名	<code>\filename{directory/file.name}</code>	<code>directory/file.name</code>
命令名	<code>\command{xelatex main}</code>	<code>xelatex main</code>
参数名	<code>\parameter{Energy}</code>	<code>Energy</code>
用户输入	<code>\userinput{runApp}</code>	<code>runApp</code>
图形界面菜单	<code>\guimenu{File &gt; Open}</code>	<code>File ▶ Open</code>

### 3.4.2 Item 列表

对于三个 item 列表（`itemize`、`enumerate` 和 `description`），我们都改了间距。如果用户想在每个 item 里放一段落，于是想保留段落间距的话，可以用新的环境变量 `paraitem`。语法为：

```

1 \begin{paraitem}
2 \item 第一段。
3 \item 第二段。
4 .....
5 \end{paraitem}

```

代码 3.12: 保留段落间距的环境变量

## 第四章 对某些对象的排版要求

本章介绍的是关于几个常用对象的排版要求。这些要求的效果属于常见的排版效果，于是公司模板中不需要自定义的宏包，只在本章做些要求或建议。为帮助理解这些要求和建议，本章会对一些基本概念做简单的解释，而这些解释不足以向新手介绍清楚对象的用法。

### 4.1 用户自己增加宏包

如果用户需要增加新的宏包，可以在 `preamble` 区（`\documentclass` 语句之后，`\begin{document}` 语句之前）加入语句 `\RequirePackage{Package}`。不建议用 `\usepackage`。

### 4.2 交叉引用和参考文献

$\text{\LaTeX}$  排版的一个优点是交叉引用的逻辑清晰而严谨，以下做一个简单的介绍。

#### 4.2.1 交叉引用

所有有 `Label` 的表格、图片、代码、公式和章节都可以被引用，语法为 `\ref{Label}`。举例：如果想实现“如表 3.1 所示”字样， $\text{\LaTeX}$  语法为

如表~\ref{triline} 所示

中间的波浪线的功能是实现“表”字与编号 3.1 之间的留空。这个留空不用普通的空格来实现，是为了避免在这里换行。

如果想提到某一 `Label` 对象所在的页码，语法为 `\pageref{Label}`。如果想给某些章节加 `Label`，方法是在章节名如 `\chapter{Chapter}` 后加一句 `\label{Label}`。`Label` 可能会非常多，作者自己会记不清。对此可参考的建议有：

- 不需要引用的就不用给 `Label`，等到需要的时候再给也不迟。
- 分类取名，比如表格都用 `tab:` 开头，图片都用 `fig:` 开头。
- 用比较清晰的英文简称作为 `Label`。

#### 4.2.2 参考文献及引用

建议参考文献用 `bibtex` 来编译。参考文献集中在文件 `main.bib` 中，文献格式示例：

```
1 @article{katz1979history,  
2   title={The History of Stokes's Theorem},  
3   author={Victor J. Katz},  
4   journal={Mathematics Magazine},  
5   volume={52},  
6   pages={146-156},  
7   year={1979},
```

8 }

代码 4.1: 参考文献格式示例

Label 在其中第一行。

这篇文献的引用方法为：

```
1 \cite{\katz1979history\}
```

### 4.3 数学环境

$\text{\LaTeX}$  是数学系的至爱，因为其数学公式排版漂亮。 $\text{\LaTeX}$  有三种数学环境：

1. `math`：行内数学环境，像在正文行内出现  $E = mc^2$  这样。
2. `displaymath`：独立占行，不参与公式编号，无 `Label`。
3. `equation`：独立占行，参与公式编号，可以设 `Label` 来引用。

我们要求所有的数学内容都用数学环境，包括物理量。比如提到深度变量的时候用  $d$  而不是  $d$ ，前者变用了 `math` 环境。排出漂亮的公式需要不少数学语法，可以准备网页版的参考手册，也可以考虑 `lyx` 或其他软件来生成。

- `math` 环境的语法为两端用 `\`(和 `\`) 括起来（或者用 `$` 括起来）。例如质能方程的语法为 `\(E=mc^2\)`，也可以是 `$E=mc^2$`。
- `display` 环境的语法是两端用 `\[`和`\]`括起来。（也可以用 `$$` 括，但据介绍可能会与其他宏包冲突，不推荐。）举例：

$$\int_0^{\infty} e^{-x} dx$$

其语法为：

```
1 \[ \int_0^{\infty} \mathrm{e}^{-x}\,,\mathrm{d}x \]
```

- `equation` 环境来表达同一个积分式，效果为：

$$\int_0^{\infty} e^{-x} dx \quad (4.1)$$

式 4.1 的语法为：

```
1 \begin{equation}\label{eq:integral}
2 \int_0^{\infty} \mathrm{e}^{-x}\,,\mathrm{d}x
3 \end{equation}
```

需要注意的是，在数学模式中，一般的字母会被解释为变量，于是变成了斜体，当你不需要斜体的时候，就要作特别声明，如 `\mathrm`。也有宏包做其他的定义方式，`dx` 中的 `d` 可以用 `\diff`；`r = 2.5cm` 中的 `2.5cm` 可以用 `\unit{2.5cm}`。

## 4.4 算法

有些宏包可以帮助 L<sup>A</sup>T<sub>E</sub>X 排版算法（伪代码）。这里举一个例子：

```
if  $i \geq \textit{maxval}$  then  
   $i \leftarrow 0$   
else  
  if  $i + k \leq \textit{maxval}$  then  
     $i \leftarrow i + k$   
  end if  
end if
```

这个例子的代码为（用了 `algpseudocode` 宏包）：

```
1 \begin{algorithmic}  
2 \If {$i \geq \textit{maxval}$}  
3   \State $i \gets 0$  
4 \Else  
5   \If {$i+k \leq \textit{maxval}$}  
6     \State $i \gets i+k$  
7   \EndIf  
8 \EndIf  
9 \end{algorithmic}
```

## 参考文献

- [1] T. Oetiker, H. Partl, I. Hyna, and E. Schlegl, “The not so short introduction to L<sup>A</sup>T<sub>E</sub>X2<sub>ε</sub>.” available online: <http://www.tex.ac.uk/tex-archive/info/lshort>, 1995.