

My T_EX & Asymptote note

redspid@gmail.com

November 21, 2011

“[T]he T_EX research project that I embarked on was driven by two major goals. The first goal was quality: we wanted to produce documents that were *not just nice, but actually the best.*”

“I never intended to have a system that would be universal and used by everybody. I always wanted to write a system that would be used for just the *finest* books.”

“The current version number for T_EX is 3.1, and for METAFONT it is 2.7. If corrections are necessary, the next versions of T_EX will be 3.14, 3.141, then 3.14159, . . . , converging to the ratio of a circle’s circumference to its diameter; for METAFONT the sequence will be 2.71, 2.718, . . . , converging to the base of natural logarithms. I intend to be fully responsible for all changes to these systems for the rest of my life.”

Donald E. Knuth
Digital Typography (1999)

Contents

1	什么是文件名数据库？怎样刷新？	4
2	如何安装宏包	4
2.1	生成样式文件并装入系统	4
2.2	生成说明文档	4
3	建立自己的包	5
4	在 CJK 中如何调整汉字的间距	5
5	首段缩进 和 段首缩进	5
6	L ^A T _E X-Suite 安装	5
6.1	GNU/Linux、Unix类	5
6.2	Windows	6
7	字体相对大小	6
8	L ^A T _E X 画线	6
9	纸张尺寸设置	6
10	图形和表格目录	6
11	图文混排	6
11.1	wrapfig 宏包	6
11.2	picins 宏包	7
12	堆叠符号	8
13	分隔符	8
14	存储盒子的用法	8
15	文字、背景颜色	8
16	Asymptote 指示箭头	8
17	用 L ^A T _E X-Suite 导入模板	9
18	Asymptote 带边框的文字	9
19	Asymptote labelpath	9
20	L ^A T _E X-Suite hotkey	10
20.1	智能扩展字符：	10
20.2	重复键入自动扩展字符	10
20.3	\前缀扩展字符：	10
20.4	希腊字母：	10

21 Asymptote: 轴对称变换	11
22 Asymptote: 标记角度	11
23 Asymptote 中的中文	11
24 Asymptote 的线条连接风格	12
25 Asymptote: 导入图像文件	12
26 L ^A T _E X 中局部行距设置	12
27 Asymptote: cut() 和 struct slice	13
28 Asymptote 颜色名称	13
29 Asymptote: 字体大小, Arrow、UnFill 大小 和 xtick	13
30 Asymptote: size()函数	14
31 特殊符号	14
31.1 带外圈的符号	14
31.2 货币符号	15
32 Beamer 文档类	15
33 自定义颜色	15
34 单/双栏混排	15
35 Asymptote: 多行文本的 label	16

List of Figures

1	8
2	9
3	9
4	11
5	11
6	12
7	12
8	12
9	13
10 Asymptote 颜色名称	13
11 颜色由深到浅的修饰词	13
12	14
13	16

1 什么是文件名数据库？怎样刷新？

T_EX 系统通过这个数据库记录了所需要的各种文件的名称和具体目录位置。每次向 T_EX 系统中添加文件后，都要记得刷新这个数据库，否则 T_EX 有可能找不到需要的文件。

刷新文件名数据库的具体命令取决于当前的 T_EX 系统。t_EX 和 f_PT_EX 的命令是：texhash
MiK_T_EX 系统可以在命令行窗口输入 initexmf --update-fndb

2 如何安装宏包

如果得到的宏包是已经编译过的，也就是已经有 .sty 或者 .cls 文件，只需把这些文件放在 localtexmf\tex\latex\ 下的某个子目录中，可以自己建一个。然后还要让 T_EX 知道这个新的宏包的存在，需要刷新 T_EX 系统的文件名数据库。现在你可以使用这个宏包了。

也可以把这些宏包文件和要编辑的 T_EX 源文件放在同一目录下，这种情况下不需要刷新 T_EX 系统的文件名数据库，但只有同一个目录下的文件可以使用这些宏包。

如果得到的是宏包的源文件，一般来说都包含两个文件：一个扩展名为 .ins，另一个扩展名为 .dtx。此外，通常会有一个 readme.txt 对宏包进行简要的说明。现在需要两个步骤将它安装到 T_EX 系统中：

2.1 生成样式文件并装入系统

1. 对 .ins 运行 L^AT_EX 命令。这将会产生一个或若干个 .sty 文件
2. 把 .sty 文件移到系统能找到的地方。通常是 .../localtexmf/tex/latex 子目录下(Windows 或者 OS/2 用户应该改变斜线为反斜线)
3. 刷新系统的文件名数据库。参照上一节的说明

2.2 生成说明文档

1. 对 .dtx 文件运行 L^AT_EX 命令。这将会产生一个 .dvi 文件。注意：可能需要多次运行 L^AT_EX 来正确处理交叉引用
2. 检查一下 L^AT_EX 是否产生了 .dvi 文件。如果没有发现这个文件，就可以执行第 5 步了。
3. 为了生成索引，键入命令：makeindex -s gind.ins name (这里 name 表示不带扩展名的主文件名)
4. 再次对 .dtx 文件运行 L^AT_EX 命令。
5. 最后一步不是必需的，生成 .ps 文件或者 .pdf 文件以方便阅读。

有时会发现生成了一个 .glo(glossary) 文件。在第 4 步和第 5 步之间运行下面的命令

```
1 | makeindex -s gglo.ist -o name.gls name.glo
```

确认在执行第 5 步前最后对 .dtx 文件运行一遍 L^AT_EX 命令。

3 建立自己的包

如果建立了很多自己的环境和命令，文档导言部分将变得很长，在这种情况下，建立一个新的 L^AT_EX 包来存放所有自定义的命令和环境是一个好的处理方式。可以在文档中使用 `\usepackage` 命令来引入中定义宏包中的环境和命令。

写一个宏包的基本工作就是将文档导言复制到一个分离的文件中去，这个文件要以 `.sty` 结尾。还需要一个命令：

```
1 \ProvidesPackage{package_name}
```

这个命令应该在包定义源文件的最前面使用。它用于告诉 L^AT_EX 宏包的名称从而允许 L^AT_EX 在你尝试两次引入同一个宏包的时候给出一个良好的错误信息。

4 在 CJK 中如何调整汉字的间距

CJK 会在两个汉字间插入一个 `\CJKglue`，因此我们修改 `\CJKglue` 的定义就可以调整汉字的间距：`\renewcommand{CJKglue}{\hskipplus<p>minus<m>}`

其中 ``、`<p>` 和 `<m>` 都是 T_EX 长度，如 `0.05pt`。执行以上命令后，汉字的标准间距就是 ``，在需要的时候可以增加或减少，调整幅度分别不超过 `<p>` 和 `<m>`。

CJK 的默认值是 `\hskip 0pt plus 0.08\baselineskip`。

5 首段缩进 和 段首缩进

`\usepackage{indentfirst}` 宏包可指定首段缩进。

article 文档类默认首段缩进 2em，从正文看可能比两个字符的宽度小一点，因为两个字符间有个间距，可在导言区用 `\parindent` 指令自定义段首缩进。

6 L^AT_EX-Suite 安装

6.1 GNU/Linux、Unix类

1. 创建 `~/vimrc/` 目录
2. 将 L^AT_EX-Suite 压缩包中内容复制到其中
3. 在 `~/vimrc/.vimrc` 中添加下面代码(引号开头的是注释行，可以删除，建议保留)

```
" -----begin latex-suite -----
" REQUIRED. This makes vim invoke latex-suite when you open a tex file.
" filetype plugin on

" let vim invoke latex-suite when you a file which has postfix .tex
let g:tex_flavor="tex"

" IMPORTANT: win32 users will need to have 'shellslash' set so that latex
" can be called correctly.
set shellslash

" IMPORTANT: grep will sometimes skip displaying the file name if you
" search in a single file. This will confuse latex-suite. Set your grep
" program to always generate a file-name.
set grepprg=grep\ -nH\ $*
```

```
" OPTIONAL: This enables automatic indentation as you type
filetype indent on
" ----- end latex-suite -----
```

6.2 Windows

1. 将 L^AT_EX-Suite 压缩包中内容复制到 ...\vim\vimfiles\ 中
2. 在 ...\vim_vimrc\ 中添加代码(与上小节相同)

L^AT_EX-Suite 中给出的配置文件预设状态是让 Vim 以打开文件的内容来判断是否 T_EX 文件(上面的代码我已做了修改, 以文件名判断打开文件的类型), 若是, 则调用 L^AT_EX-Suite。

7 字体相对大小

当前 normalsize 大小为 12pt。

tiny scriptsize footnotesize small normalsize large Large LARGE huge Huge

8 L^AT_EX 画线

`\rule[lift]{width}{height}`

9 纸张尺寸设置

`\paperheight=26cm`

`\paperwidth=20cm`

但是当通过 dvipdfm 或 dvipdfmx 将 dvi 文件转换为 pdf 文件的时候这两条指令不起作用。此时可在导言区用 `\special{papersize=10cm,20cm}` 这样的指令来设置。

注意当前的 `\textheight` 和 `\textwidth` 设置, 应在纸张大小的范围内。

10 图形和表格目录

`\listoffigures` 指令排版图形目录, `\listfigurename` 为图形目录标题。

`\listoftables` 指令排版表格目录, `\listtablename` 为表格目录标题。

11 图文混排

11.1 wrapfig 宏包

用法: `\begin{wrapfigure}{行数}[位置][超出长度]{宽度}<图形>\end{wrapfigure}`

行数是指图形高度所占的文本行的数目。如果不给出此选项, wrapfig 会自动计算。位置是指图形相对于文本的位置, 须给定下面四项的一个。

- r,R 表示图形位于文本的右边
- l,L 表示图形位于文本的左边

- **i,I** 表示图形位于页面靠里的一边(用在双面格式里)
- **o,O** 表示图形位于页面靠外的一边

超出长度是指图形超出文本边界的长度, 缺省为 *0pt*。宽度则指图形的宽度。wrapfig 会自动计算图形的高度。不过, 我们也可设定图形的高度, 具体可见 wrapfig.sty 内的说明。

例:

```
1 \usepackage{graphicx}
2 \usepackage{wrapfig}
3 ...
4 \begin{wrapfigure}{r}{4.5cm}
5 \includegraphics [width=4cm,clip]{yourfigure.eps}
6 \end{wrapfigure}
```

11.2 picins 宏包

用法: `\parpic(宽度,高度)(水平偏移,垂直偏移)[选项][位置]{图形}`

位置只能为下面两个中的一个

- l 将图形置于文本段落的左方(这也是缺省值)
- r 将图形置于文本段落的右方

外观只能为下面五个中的一个, 可与上述位置项配合使用

- f 将图形置于一个实框盒子中
- d 将图形置于一个虚框盒子中
- o 将图形置于一个圆角框盒子中
- s 将图形置于一个具有阴影效果的盒子中
- x 将图形置于一个具有立体效果的盒子中

位置仅当给定的宽度和高度与图形的实际大小相差很大的情况下才起作用。若税票活垂直偏移已给出, 那么此项也不起作用。缺省位置是将图形置于盒子的中央。也可以取以下的值:

- l 将图形置盒子的左方
- r 将图形置于盒子的右方
- t 将图形置于盒子的上方
- b 将图形置于盒子的下方
- x 将图形置于一个具有立体效果的盒子中

例如图1 “Asymptote 指示箭头” 一节中使用的插图代码:

```
\parpic[r]{\includegraphics[scale=0.7]{arrow_show.eps}}
```

用下面的代码为混排的图片增加 caption 和 label :

```
1 \piccaption{yourcaption\label{yourlabel}}
2 \parpic[r]{\includegraphics{show.eps}}
```

可在 `\parpic` 指令后面的小括号内设置图片所占空间大小(必须宽、高同时设定, 不能缺项。这里不是图片大小, 图片大小用 `\includegraphics` 的 `width` 参数设定)。插入图2的代码如下:

```
1 \piccaption{Asymptote 文字外框\label{string_label}}
2 \parpic(5cm,3cm)[r]{\includegraphics{string_label.eps}}
```

12 堆叠符号

用`\stackrel{}{}`可将两个符号竖直堆叠起来。

例如: `\stackrel{y\rightarrow+\infty}{x\rightarrow+\infty}` 的效果: $x \xrightarrow{y \rightarrow +\infty} +\infty$

若要以相同大小堆叠, 可用`\atop`命令。

例如: `\stackrel{y\rightarrow+\infty}{x\rightarrow+\infty}` 的效果: $x \xrightarrow{y \rightarrow +\infty} +\infty$

13 分隔符

`\{x\mid x>0\}` $\{x \mid x > 0\}$

14 存储盒子的用法

```
1 \newsavebox{\mybox} %定义盒子名称
2 \savebox{\mybox}{测试存储盒子的用法} %定义盒子内容
3
4 \usebox{\mybox} %使用盒子, 出现的将是盒子内容
```

定义盒子的完整形式为: `\savebox{\boxname}[width][pos]{...}`

15 文字、背景颜色

彩色文字和背景需导入 `color` 宏包: `\usepackage{color}`, 中文环境需将这条指令置于 `\begin{CJK}` 下面。

例: `\textcolor{red}{红色的文字}`: 红色的文字

16 Asymptote 指示箭头

```
1 size(10cm,0);
2 texpreamble("\usepackage{amsmath}");
3
4 path p = (0,0)..(1,1)..(2,.5){dir(0)};
5 real al = arclength(p);
6 real l = length(p);
7 pair pt = point(p,l/2);
8 pair apt = arcpoint(p,al/2);
9
10 draw(p);
11 draw(subpath(p,0,arctime(p,al/2)),blue);
12
13 // 指向 pt 的箭头, 文字位于 pt 的 SSE, 箭头长度 1cm
14 dot(pt,red); dot(apt,blue);
15
16 // 指向 pt 的箭头, 文字位于 apt 的 SSW, 箭头长度 1cm
17 arrow("\frac{\text{length}(p)}{2}",pt,SSE,1cm);
18 arrow("\frac{\text{arclength}(p)}{2}",apt,SSW,1cm);
19 label("\small The sizes of the blue path and the black path are equal",
20 (1,.25));
```

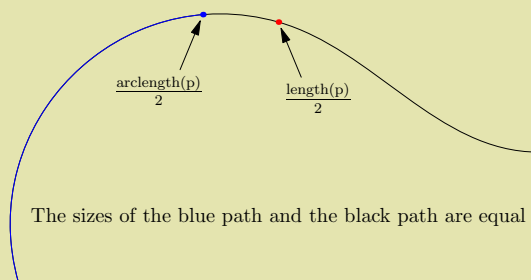


图 1: Asymptote 指示箭头演示图

17 用 L^AT_EX-Suite 导入模板

安装 LaTeX-Suite 后可以用 `:TTemplate` 命令列出已有的模板列表。也可以用 `:TTemplate tname` 导入名为 `tname` 的模板。`:TTemplate` 可按照 T_EX 规则简写为 `:TTe`。

(2008-7-22 12:15:21)模板导入时会自动缩进, 可能导致格式错乱。现在我在 `_vimrc` 中加入两个自动命令来解决:

```
1 autocmd BufReadPre
2 filetype indent off
3 autocmd BufReadPost
4 filetype indent on
```

这个方法在引入较复杂的, 例如有 `tabular` 环境的模板时仍会出现自动缩进的问题。

18 Asymptote 带边框的文字

`point(object, -1)` 为对象左连接点, `point(object, 1)` 为对象右连接点。

```
1 currentpen = linewidth(2bp);
2
3 // draw(string, path, pair, 文字与边界的margin);
4 object hello = draw("Hello", box, (0,15mm),1mm);
5 object world = draw("World", ellipse, (0,0), 2mm);
6
7 draw(point(hello, -1){SW} .. {W}point(world, 1), Arrow(TeXHead));
```

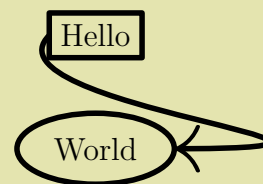


图 2: Asymptote 文字外框

19 Asymptote labelpath

这个模块使用 PSTricks `pstextpath` 宏使 labels 按照一个 path 排列(像示例文件 `curvedlabel.asy` 中一样适当紧密排列)。

```
1 函数原型:
2 void labelpath(picture pic=currentpicture, Label L, path g,
3               string justify=Centered, pen p=currentpen);
```

这里 `justify` 是 `LeftJustified`、`Centered` 或 `RightJustified` 其中的一个。The x component of a shift transform applied to the Label is interpreted as a shift along the curve, whereas Chapter 7: Base modules 91 the y component is interpreted as a shift away from the curve. All other Label transforms are ignored. This package requires the latex tex engine and inherits the limitations of the PSTricks `\pstextpath` macro.

Note: 源码文件命名的时候不能与代码中导入的 module 同名, 否则可能被编译器认为是递归调用 module。

图3代码如下:

```
1 size(4cm, 0);
2 import labelpath;
3
4 labelpath("This is a test of curved labels
5           in Asymptote.",
6           reverse(rotate(-90)*unitcircle));
```

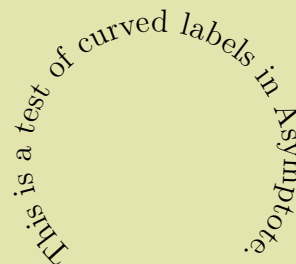


图 3: labelpath 示例

20 L^AT_EX-Suite hotkey

- <F5>键可自动扩展当前 word 为环境模式(在正文部分), 在导言部分可扩展当前 word 为`\usepackage{word}`
- <F7>键可自动扩展当前 word 为命令模式
- 和环境扩展以‘E’开通类似, 字体环境以‘F’开头。‘FEM’即扩展为: `\emph{<+>}`
- ALT-l 键扩展为: `\label{<+>}`

20.1 智能扩展字符:

```
1 ' __ ' -> ' _{<+>}<+> '          ' ( ) ' -> ' (<+>)<+> '
2 ' [] ' -> ' [<+>]<+> '          ' { } ' -> ' {<+>}<+> '
3 ' ^ ^ ' -> ' ^{<+>}<+> '          ' $ $ ' -> ' $<+>$<+> '
4 ' = = ' -> ' &= & '          ' ~ ~ ' -> ' &\approx & '
5 ' = ~ ' -> ' \approx '          ' : : ' -> ' \dots '
6 ' ( ( ' -> ' \left( <+> \right)<+> '
7 ' [ [ ' -> ' \left[ <+> \right]<+> '
8 ' { { ' -> ' \left\{ <+> \right\}<+> '
```

20.2 重复键入自动扩展字符

```
1 = -> \equiv          \ -> \setminus
2 '. ' -> \cdot         '* ' -> \times
3 '& ' -> \wedge        '- ' -> \bigcap
4 '+ ' -> \bigcup        'M' -> \sum_{<+>}^{<+>}<+>
5 'S' -> \sum_{<+>}^{<+>}<+>  '(' -> \subset
6 ')' -> \supset          '<' -> \le
7 '>' -> \ge            ', ' -> \nonumber
8 '~ ' -> \tilde{<+>}<+>  ';' -> \dot{<+>}<+>
9 ': ' -> \ddot{<+>}<+>
```

20.3 \前缀扩展字符:

```
1 ^ -> \hat{<+>}<+>      _ -> \bar{<+>}<+>
2 6 -> \partial          8 -> \infty
3 / -> \frac{<+>}{<+>}<+>  % -> \frac{<+>}{<+>}<+>
4 @ -> \circ            0 -> ^\circ
5 2 -> \sqrt{<+>}<+>      | -> \Big|
6 I -> \int_{<+>}^{<+>}<+>
```

20.4 希腊字母:

```
1 a -> \alpha          b -> \beta          c -> \chi
2 d -> \delta          e -> \varepsilon      f -> \varphi
3 g -> \gamma          h -> \eta          k -> \kappa
4 l -> \lambda          m -> \mu          n -> \nu
5 p -> \pi             q -> \theta        r -> \rho
6 s -> \sigma          t -> \tau          u -> \upsilon
7 v -> \varsigma       w -> \omega        x -> \xi
8 y -> \psi            z -> \zeta
```

$\alpha \beta \chi \delta \varepsilon \varphi \gamma \eta \kappa \lambda \mu \nu \pi \theta \rho \sigma \tau \upsilon \varsigma \wedge \xi \psi \zeta$

不是所有的希腊字母在 L^AT_EX-Suite 中都有大写形式，

可参考 <http://www.giss.nasa.gov/latex/ltx-405.html>

1	D -> \Delta	G -> \Gamma	F -> \Phi
2	L -> \Lambda	X -> \Xi	Q -> \Theta
3	Y -> \Psi	S -> \Sigma	U -> \Upsilon
4	W -> \Omega		

$\Delta \quad \Gamma \quad \Phi \quad \Lambda \quad \Xi \quad \Theta \quad \Psi \quad \Sigma \quad \Upsilon \quad \Omega$

21 Asymptote: 轴对称变换

```

1 size(4cm);
2 import graph;
3 pen drawpen = linewidth(bp);
4
5 real f(real t) { return t^2; }
6 path ff = graph(f, 0, 1.5, operator..);
7 draw(ff, drawpen+red);
8
9 draw(reflect((0,0),(1,1)) * ff, drawpen+blue+dashed);
10
11 draw((0,0)--(2,2), drawpen);

```

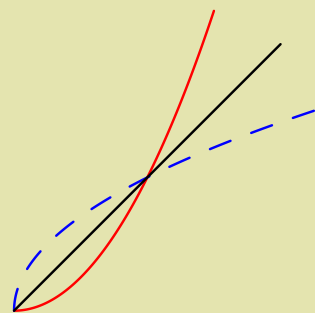


图 4: 轴对称变换

22 Asymptote: 标记角度

使用 `markers` 包中的 `markangle()` 函数为角度做标记。 `\angle`→ \angle 。

函数原型:

```

1 void markangle(picture pic=currentpicture, Label L="",
2               int n=1, real radius=0, real space=0,
3               pair A, pair O, pair B, arrowbar arrow=None,
4               pen p=currentpen, margin margin=NoMargin,
5               marker marker=nomarker);

```

示例(图5):

```

1 import markers;
2 pair A=(50,60), B=(60,10), origin=(0,0);
3 draw(A--origin--B, linewidth(2bp));
4 markangle("\angle BOA", B, origin, A, Arrow, linewidth(bp));

```

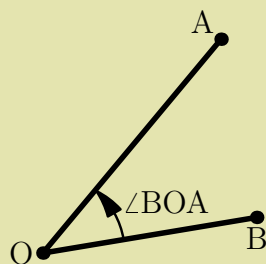


图 5: 标记角度

23 Asymptote 中的中文

Asymptote 使用 T_EX 处理中文，按图6中这样在生成的图像中使用中文:

```

1 import labelpath;
2 size(100);
3 texpreamble("\usepackage{CJK}\AtBeginDocument{
4     \begin{CJK}{GBK}{fs}}
5     \AtEndDocument{\clearpage\end{CJK}}");
6 labelpath("\scriptsize 现在可以像在 \LaTeX{}
7     中一样在 Asymptote 中使用中文",
8     reverse(rotate(-90)*unitcircle));

```



图 6: 在 Asymptote 中使用中文

24 Asymptote 的线条连接风格

```

1 pen squarecap = linecap(0);
2 pen roundcap = linecap(1);
3 pen extendcap = linecap(2);

```



图 7: linecap(0、1、2) 三种连接风格

25 Asymptote: 导入图像文件

`string graphic(string name, string options="")`

返回一个可以用来导入 EPS 文件的 string 变量。

这里 `name` 是被导入的 EPS 文件名。 `option` 是一个包含下面选项的用逗号分割的参数列表。

- (bb=llx lly urx ury) bounding box
- (width=value) width
- (height=value) height
- (angle=value) rotation
- (scale=factor) scaling
- (clip=bool) clipping
- (draft=bool) draft mode

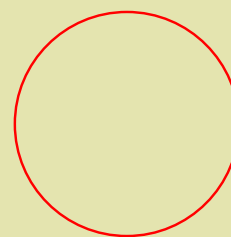


图 8: Asymptote 导入图像文件

`void layer()` 函数可以将后面添加的对象放在导入的图像上层。

图8的代码:

```

1 label(graphic("hb.eps", "width=5cm, clip=true"),
2     position=(0,0));
3 layer();
4 draw(circle((0,-13), 42), linewidth(bp)+red);

```

26 L^AT_EX 中局部行距设置

可用 `setspace` 宏包设置局部行距。如下所示:

```

1 \usepackage{setspace}
2 \begin{spacing}{1.8}
3 .....
4 \end{spacing}

```

27 Asymptote: cut() 和 struct slice

`slice cut(path p, path knife, int n);`

以 `struct slice { path before, after; }` 形式返回路径 `p` 被与路径 `knife` 第 `n` (从0开始计数) 个交点分隔开的前、后两部分(如果两条路径没有交点, 整个路径 `p` 都被认为是交点‘前’的部分)。

参数 `n` 会对交点数取模(`n%numOfIntersection`)。

`slice firstcut(path p, path knife);` 等价于 `cut(p, knife, 0);` 。

`slice lastcut(path p, path knife);` 等价于 `cut(p, knife, -1);` 。

示例代码(图9):

```
1 size(150); import graph; defaultpen(2bp);
2
3 path s = graph(sin, 0, 2pi);
4 path h = (0,0) -- (6.3,0);
5 draw(h);
6
7 slice k = cut(s, h, 1);
8 draw("$sin(x)$before", k.before, N, blue);
9 draw("$sin(x)$after", k.after, S, red);
```

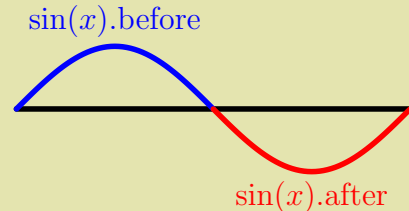


图 9: cut 函数和 struct slice 示例

28 Asymptote 颜色名称



图 10: Asymptote 颜色名称

颜色由深到浅的修饰词: dark→deep→heavy→*→medium→light→pale。

如图11所示:

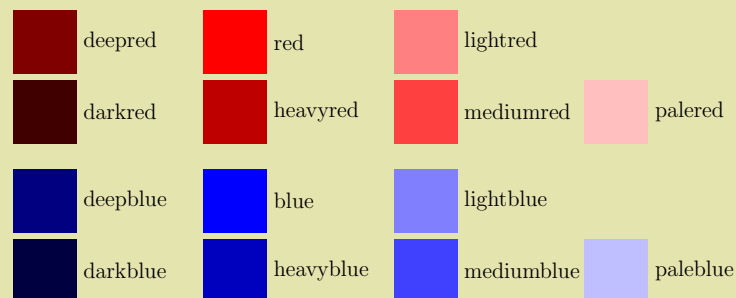


图 11: 颜色由深到浅的修饰词

29 Asymptote: 字体大小, Arrow、UnFill 大小 和 xtick

下面代码第 3行设置定义默认字体大小, 也可用`defaultpen(fontsize(12pt))`形式。

第 14、15 行代码方位标志 **S** 前的数字表示 label 与标记点的距离，`UnFill(real)` 中参数定义空白大小 (**Note**: 不是 dot 大小，dot 大小由 `linewidth()` 或 `dotfactor` 设定)。

图 12 的 a 标记点周围有空白，这就是 `UnFill(real)` 的设置过大。

```

1 import graph;
2 size(200,0);
3 defaultpen(1bp+red); //设定默认pen直径和颜色
4 defaultpen(fontcommand("\large"));
5
6 xlimits(0,10);
7 xaxis("$x$", Arrow(10pt)); //Arrow(real) 中参数可以设定大小
8 xtick("$0$", 1); //第二个参数也可以用 pair
9 pair a=(3,0), b=(7,0), rise=(0,0.5);
10
11 draw("$ (a, b) $", a--(a+rise)--(b+rise)--b,
12     N, blue+fontsize(16pt)); //最后一个参数定义此处的pen和字体大小
13
14 dot("$a$", a, 2S, UnFill(3pt));
15 dot("$b$", b, 2S, UnFill);

```

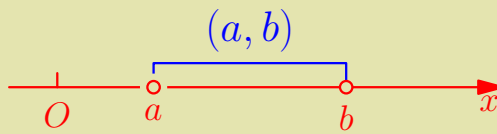


图 12: 字体 `Arrow.UnFill` 大小设置

30 Asymptote: `size()` 函数

`size(200);` // $x=200, y=200$

`size(200,0);` // $x=200$, y 不限制












`size(0,200);` // x 不限制, $y=200$

`size(0,0);` // 直接采用 PostScript 的坐标，最终不做任何图形尺寸的控制，这时作图等价于 MetaPost 的手工控制最终图形的尺寸。


31 特殊符号


宏包导入太多命令会冲突，`ifsym1` 里面有好多漂亮的符号就没法在这里 show 了，我找找有没有导入局部命令的方法²。特殊符号参考文档 `symbols`。`bbding` 宏包一定要在 `wasysym` 之前导入：

导入：`\usepackage{bbding, wasysym}`

<code>%</code>	<code>wasysym</code>	<code>\permil</code>		<code>bbding</code>	<code>\HandLeft</code>
	<code>bbding</code>	<code>\HandRight</code>		<code>bbding</code>	<code>\HandLeftUp</code>
	<code>bbding</code>	<code>\HandRightUp</code>		<code>bbding</code>	<code>\HandPencilLeft</code>
	<code>bbding</code>	<code>\Peace</code>		<code>bbding</code>	<code>\Phone</code>
	<code>bbding</code>	<code>\PhoneHandset</code>		<code>bbding</code>	<code>\Envelope</code>
	<code>bbding</code>	<code>\Plane</code>		<code>bbding</code>	<code>\Tape</code>

31.1 带外圈的符号

`\textcircled{A}` 

汉字比英文字母大一点会跑到圈外面：`\textcircled{字}` 

¹ 导入天气符号需要用 `weather` 选项

² 似乎目前还没有好的办法可以完美解决这个问题，关于汉字注音的笔记由于和希腊字母命令冲突，只能放在 `additional.tex` 里面了

31.2 货币符号

`\$` `$` `\pounds` `£`
textcomp 宏包: `\textcent` `¢` `\textyen` `¥` `\textsterling` `£`
eurosym 宏包: `\geneuro` `€` `\geneuronarrow` `€` `\geneurowide` `€` `\officialeguro` `€`

32 Beamer 文档类

- 在`\section{}`参数中用汉字，需在导入 beamer 文档类时加 CJK 参数：
`\documentclass[CJK]{beamer}`
- `\begin{CJK}{GBK}{fs}`要放在 `\begin{document}` 下面。
- 需使用`\verb`命令或 verbatim 环境的 frame 要带参数 `fragile:\begin{frame}[fragile]` 并且此时不可直接在 frame 环境参数中写 `frametitle`，而应用 `\frametitle{}` 命令。
- 导言区加入`\setbeamertheme{navigation symbols}`取消导航条。
- `\includegraphics[]{}{}`定义宽度或高度参数需明确写：“`[width=]`”或“`[height=]`”。

33 自定义颜色

使用 `color` 宏包的 `\definecolor{name}{mode}{color-spec}` 命令可以自定义颜色。这个笔记中 **强调文本** 的颜色就是用下面的代码定义的：

```
1 \definecolor{Mzise}{RGB}{109,0,219}
2 \newcommand{\Memph}[1]{\textcolor{Mzise}{#1}}
```

除了用 `RGB` 模式定义颜色外，还可以用 `cmymk` 或其它模式，目前 `RGB` 颜色已经够用了。

34 单/双栏混排

`cuted`宏包用于在双栏选项下提供 Full width 的单栏环境。

```
1 \documentclass[a4paper,twocolumn,10pt]{article}
2 \usepackage{cuted}
3 \begin{document}
4 % 此处为双栏...
5 \begin{strip}
6     % 此处单栏...
7 \end{strip}
8 \end{document}
```

35 Asymptote: 多行文本的 label

需要在图中使用多行文本，可以用 `minipage` 函数：

```
1 pair O=(0,0);  
2 string l = "I want to write many lines of comments  
3           for a component, just like this.";   
4 draw( circle(O, 37mm), 1bp+darkgreen );  
5 dot(O);  
6 label( minipage(1, 30mm), (O,O), SE, blue);
```

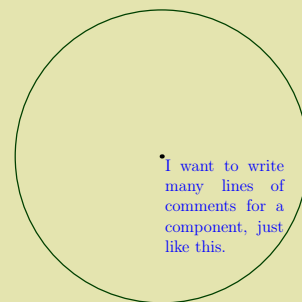


图 13: Asymptote 导入图像文件