geometry宏包使用说明

目录

摘要

这个宏包提供了一个能够方便灵活地管理页面规格的接口。用户能够利用直观的参数来改变页面的布局。比如说：如果你想让文章边缘和纸张边缘的距离为 2 厘米，你只需要输入如下命令：`\usepackage[margin=2cm]{geometry}`。页面的布局可以利用`\newgeometry` 命令在文章的任意位置来修改。

第一章　第五版前言

- **能够在文章中改变页面样式**

  这里要利用两个新的命令：`\newgeometry{···}` 和`\restoregeometry`。这两个新的命令使得用户能够在文章中改变页面的布局。 `\newgeometry` 这个命令使得在之前声明的所有关于页面布局的选项无效化，并将忽略一切与纸张大小相关的参数，这些参数包括：`landscape`，`portrait` 和纸张大小的参数（比如说`papersize`，`paper=a4paper` 等）。在其他的方面`\newgeometry` 和`\geometry` 命令相同。

- **加入了一些新的表示布局区域的参数**

  一些新的参数在计算页面区域和布局的时候可以用到，这些参数有：`layout`，`layoutsize`，`layoutwidth`，`layoutheight` 等等。这些参数可以帮助我们在不同的纸张大小下打印特定的页面布局。举例来说，在`a4paper` 和`layout=a5paper` 这两个命令的作用下，geometry宏包会在'A4' 的纸张上使用'A5' 的布局来计算边界。

- **一个新的驱动选项——xetex**

  在第五版中新加入了一个驱动选项`xetex`。程序自动检测驱动系统已经做了完善，以避免'undefined control sequences' 这种错误。在 TEX Live 中的'geometry.cfg' 的文件中，它禁止了程序自动检测驱动和设置`pdftex` 的功能，虽然这个文件在第五版的宏包中仍然存在，但是他已经是不必要的，也不会引起任何错误的了。我们强烈建议在 XELATEX 中使用`xetex` 选项。

- **为 JIS B-series 和 ISO C-series 设计的新的纸张大小**

  新加入了满足 JIS(Japanese Industrial Standards)B-series 的纸张大小（`b0j` 到`b6j`）和满足 ISO C-series(v5.4∼) 的纸张大小（`c0paper` 到`c6paper`）。

- **改变了没有被设置的边距的默认值**

  在之前的版本中，如果只有一个边距被改变了，比如说`bottom=1cm` 这个命令，geometry会通过一个比例设置它的另一边的边距（在垂直的排版中，这个比例是 1:1），也就是说会使得同时执行`top=1cm` 这个命令。第五版中设置了这个比例为 0.7，并以此来确定这些没有被设置的边距。（6.5）

- **参数 showframe和 showcrop在每一页都起作用**

  在参数`showframe` 的作用下，页边框在每一页都会显示。另外，一个新的参数`showcrop` 在每一页的每个角上打印 crop 标记。注意如果不设置页面布局尺寸小于纸张尺寸时，这个标记是不可见的。Version 5.4 introduced a new`shipout` overloading process using atbegshi package，所以 atbegshi宏包需要在 showframe 和 showcrop 参数调用时被使用。

- **在处理类之前读取 geometry.cfg 文件** 之前的版本是在处理类之后才读取的 geometry.cfg 文件。现在因为在处理类之前就读取了配置文件，所以用户可以在`\documentclass`、`\usepackage` 和`\geometry` 中加入参数以改变在 geometry.cfg中使用的一些设置。

- **删除了一些参数：compat2 和 twosideshift**

  第五版不兼容之前的版本，因此为求简洁，compat2和 twosideshift这两个参数被删除了。

第二章　简介

在 LaTeX 中设置页面的布局不是一件容易的事。用户需要适应几种 LaTeX 原生的尺寸来将文字区域放置在期望的位置。如果用户想要将文字区域放在使用的纸的中心，举例来说，你需要按如下的方法来设置页面尺寸：

```
\usepackage{calc}
\setlength\textwidth{7in}
\setlength\textheight{10in}
\setlength\oddsidemargin{(\paperwidth-\textwidth)/2-1in}
\setlength\topmargin{(\paperheight-\textheight
                      -\headheight-\headsep-\footskip)/2-1in}
```

如果不使用 calc 宏包，上面的例子需要更多复杂的设置，geometry宏包提供了一个设置页面布局参数的简单的方法。利用 geometry 宏包，你所做的只是敲入如下命令：

```
\usepackage[text={7in,10in},centering]{geometry}
```

除开文章居中的问题，设置页边距也很麻烦。但是 geometry宏包也让它变得容易实现了。如果你想设置每个页边距为 1.5in，你可以输入：

```
\usepackage[margin=1.5in]{geometry}
```

因此，geometry宏包具有“自动完成设置”的功能，也就是说，没有指定的一些布局都可以被自动设置。geometry宏包还可以在你需要设置极为严格的页面布局时变得十分有用，比如说：

> 文字区域的尺寸是 6.5 英尺宽和 8.75 英尺高。每页顶部的页边距应该是 1.2 英尺。左边的页边距是 0.9 英尺，脚注和页码必须在文字区域的底部

在这种情况下，使用 geometry 宏包，你可以输入如下命令：

```
\usepackage[total={6.5in,8.75in}],
          top=1.2in,left=0.9in,incluefoot]{geometry}
```

在文件制作系统中设置一个文字区域和在一个视窗系统中放置一个窗口类似，'geometry' 这个名字来自于在 **X Windows**系统中用于设置窗口大小和位置的参数 **-geometry**。

第三章　页面 geometry

图 1表示了在 geometry 宏包中定义的页面布局。这个页面布局包括： *total body*（可打印区域）和 *margins*（页边距）。*total body* 是由 *body*（文字区域）和一个可修改的 *header*,*footer* 和页边批注组成。一共有四类页边距：*left*,*right*,*top* 和 *bottom*。对双面打印的文章来说，水平的页边距通常被叫做 *inner* 和 *outer*。

paper : *total body* 和 *margins*
total body : *body*（文字区域）（可以修改的有：*head*,*foot*,*marginpar*）
margins : *left*(*inner*),*right*(*outer*),*top*和*bottom*

每一个页边距都是从它对应纸张的边上开始测量的。例如，左页边距（内页边距）是指从纸张的最左边到*total body*的水平距离。因此，左和上页边距和原来的页面定义 `\leftmargin` 和 `\topmargin`

是不同的。*body*（文字区域）的大小可以通过命令 `\textwidth` 和 `\textheight` 来修改。纸张的尺寸，*total body* 和 *margin* 有如下的关系：

$$\texttt{paperwidth} = \texttt{left} + \texttt{width} + \texttt{right} \tag{1}$$

$$\texttt{paperheight} = \texttt{top} + \texttt{height} + \texttt{bottom} \tag{2}$$

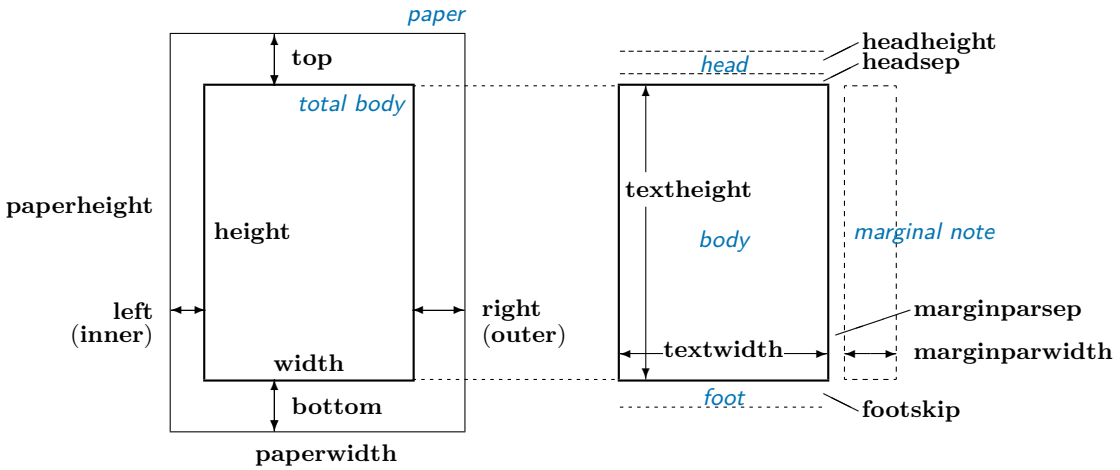


图 1: 在 geometry宏包中使用了一些独有的表示方法：
比如：*width* = *textwidth* 和 *height* = *textheight*。而 *left*， *right*， *top* 和 *bottom* 这四个参数用来表示页边距。如果文章通过 *twoside* 这个参数修改过，那么可以使用 *left*(*inner*) 和 *right*(*outer*) 来确定靠里页面和靠外页面的页边距。

*total body*的宽度和高度定义如下：

$$width = textwidth(+marginparsep + marginparwidth) \tag{3}$$

$$height = textheight(+headheight + headsep + footskip) \tag{4}$$

当 `marginparsep` 和 `marginparwidth` 在 `width` 的范围之内，也即 `includemp` 这个参数被指定为 `true` 时，等式 (3) 是系统在水平方向默认的设置。等式 (4) 是系统在垂直方向默认的设置。但是当 `includehead` 这个参数被设置为 `true` 时，`headheight` 和 `headsep` 则被包括在了 `height` 中。同样，`includefoot` 这个参数将 `footskip` 包含在了 `height` 中。

图 2表示了这些参数是怎样在垂直的方向起作用的。

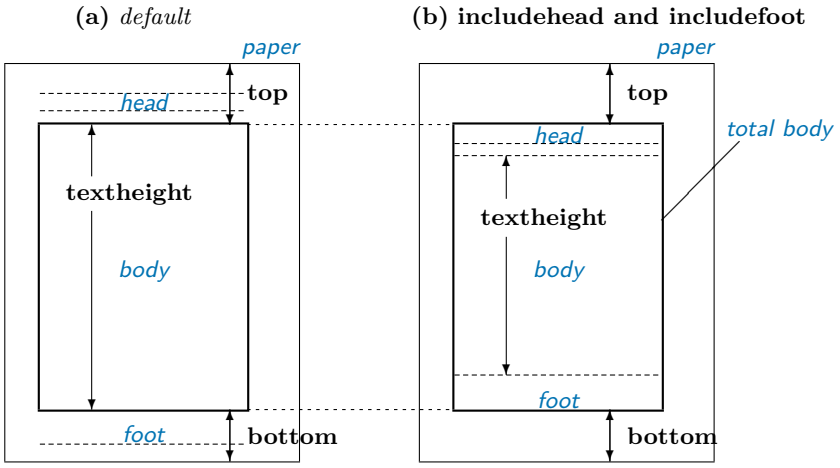


图 2: *includehead* 和 *includefoot* 分别将 *head* 和 *foot* 包括到了 *total body* 中。**(a)**图表示了 *height* = *textheight*（默认值）。**(b)**图表示了如果设置了 *includehead*和 *includefoot*，那么 *height* = *textheight* + *headheight* + *headsep* + *footskip*。显然，如果定义了 *top* 和 *bottom* 参数 *inludehead* 和 *includefoot* 会导致更短的 *textheight*。

这样，这个页面的布局在每个方向上就包括了三个部分的长度：一个主体 (body) 和两个边距 (margin)。如果他们中的两个被设置了，那么第三个就确定了，也就不用再单独地设置了。

图 3表示了一个简单的页面尺寸关系的模型。当长度L 给定并且分成 *body* b，边距 a 和 c，显然有如下的等式成立：

$$L = a + b + c \tag{5}$$

这个等式表明了当 a,b,c 其中的两个参数确定之后，另外一个参数就可以解出来。如果两个或两个以上的参数没有被确定，那么等式 (5) 就不能在没有两者其他关系的情况下被解出来。如果所有的参数都是确定的，那么则需要检查他们是否能够满足等式 (5)。 geometry宏包具有的自动完成的功

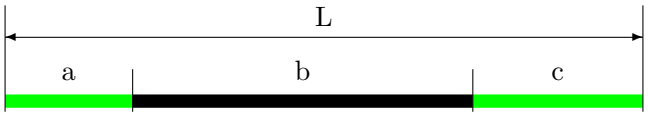


图 3: 一个关于页面尺寸关系的简单的模型

能可以帮助用户确定页面的尺寸关系。比如说，你可以在 A4 的纸张大小系输入如下命令：

```
\usepackage[width=14cm,left=3cm]{geometry}
```

这种情况下，你不需要输入右边距的命令。关于自动完成这个命令将会在第 6.5章中详细被讨论。

第四章　用户接口

geometry宏包提供了如下的命令：

- `\geometry{`<*options*>`}`
- `\newgeometry{`<*options*>`}` 和 `\restoregeometry`
- `\savegeometry{`<*name*>`}` 和 `loadgeometry{`<*name*>`}`

\geometry{<*options*>} 通过 *options*中参数的来改变页面的布局。这个命令一般需要放置在引言区（在\begin{document} 之前）。

geometry 宏包可能会被当做类或者用户在文档中使用的其他宏包的一部分来使用，也就是说 \geometry 命令可能会覆盖引言区得其他的一些设定。用户可以多次地使用 \geometry 命令，相关的设置都会起作用。在 geometry 宏包加载之前，用户可以通过 \uspackage{<*option*>} 来代替单独的命令 \geometry

\savegeometry{<*name*>} 能够保存页面的尺寸信息为 <*name*>，而\loadgeometry{<*name*>} 能够读取页面的尺寸信息为 <*name*> 的设置。在第 **??**会有更多的介绍。

4.1　可修改的参数

geometry宏包采用 keyval接口：'<*key*>=<*value*>'。这个接口格式适用于\usepackage,\geometry 和\newgeometry。

这个参数包括了一系列的命令分隔的 keyval选项，并具有一些基本的规则：

- 支持多行命令，但是不支持空白行命令
- 单词之间的所有空白将被忽略
- 参数之间的顺序不会影响结果（但是有一些例外，具体参见第 6.2章）

比如以下命令：

```
\usepackage[ a5paper,  hmargin = {3cm,
                  .8in }, height
          =   10in   ]{geometry}
```

和：

```
\usepackage[a5paper,height=10in,hmargin={3cm,0.8in}]{geometry}
```

是等同的。

一些参数可能会有下一层的参数列表，比如说{3cm,0.8in}。在下一层的参数列表中，各个参数的顺序是至关重要的。上面的例子和下面的例子的作用也是一样的：

```
\usepackage{geometry}
\geomegry{height=10in,a5paper,hmargin={3cm,0.8in}}
```

或者：

```
\usepackage[a5paper]{geometry}
\geometry{hmargin={3cm,0.8in},height=8in}
\geometry{height=10in}
```

通常来说，多次使用\geometry 命令只是用来定义附加的参数。

geometry支持 *calc*宏包[1]。调用方法如下：

```
\usepackage{calc}
\usepackage[textheight=20\baselineskip+10pt]{geometry}
```

[1] CTAN: macros/latex/required/tools

4.2　参数类型

geometry参数被分成四类：

1. **Bollen** 型

   表示一个 boolean 型参数的值（`true` 或 `false`）。如果没有定义，默认为 `true`。

   <*key*>=ture | false
   <*key*> 如果没有赋值则相当于: <*key*>=ture

   一些定义的例子： `verbose=ture`, `includehead`, `twoside=false`。
   使用的纸张的名字是个例外。选择的纸张必须被设置为没有赋值。不管赋给这类参数什么值都会被忽略。比如说：`a4paper=XXX` 和 `a4paper` 的作用是相同的。

2. **Single-valued** 型

   这类参数必须要赋一个值。

   <*key*>=<*value*>

   例子： `width=7in`, `left=1.25in`, `footskip=1cm`,`height=.86\paperheight`。

3. **Double-valued** 型

   在大括号中含有一对以逗号分隔的参数值。当某个参数相同的时候可以只写一个参数。

   <*key*>={<*value1*>, <*value2*>}.
   <*key*>=<*value*> 和 <*key*>={<*value1*>,<*value1*>} 是等价的

   例子： `hmargin={1.5in,1in}`, `scal=0.8`, `body={7in,10in}`.

4. **Triple-valued** 型

   在大括号中含有三个通过逗号分隔并且必须定义的参数值

   <*key*>={<*value1*>, <*value2*>, <*value3*>}

   每个参数的值必须是数或者是 null 。当用户赋给了参数空值或者'*' 时，这意味着用户让系统的自动完成功能为这个参数选择一个合适的值。；用户需要赋予至少一个参数的值，一般是两个参数。用户可以给所有参数不赋值，但是这个不会起任何作用。
   例如： `hdivide={2cm,*,1cm}`, `vdivide={3cm,19cm, }`, `divide={1in,*,1in}`.

第五章　参数的详细介绍

这一章描述了在 geometry 宏包中可以使用的所有参数。前面有 † 符号的参数表示不能再`\newgeometry` 中使用（参见第 七章）

5.1　页面大小

下面介绍的选项主要用于设置页面的大小和方向

†`paper` | `papername`

用名字来定义纸张大小：`paper=`<*paper-name*>。方便起见，用户可以省去 `paper` 来定义纸张大小。比如：`a4paper` 和 `paper=a4paper` 作用相同。

† a0paper, a1paper, a2paper, a3paper, a4paper, a5paper, a6paper,
b0paper, b1paper, b2paper, b3paper, b4paper, b5paper, b6paper,
c0paper, c1paper, c2paper, c3paper, c4paper, c5paper, c6paper,
b0j, b1j, b2j, b3j, b4j, b5j, b6j,
ansiapaper, ansibpaper, ansicpaper, ansidpaper, ansiepaper,
letterpaper, executivepaper, legalpaper
定义纸张的名字。赋值的部分总是会省略。比如说，接下来的这些例子拥有相同的排版效果： a5paper, a5paper=true, a5paper=false 等等。 a[0-6]paper, b[0-6]paper 和 c[0-6]paper 分别是 ISO A, B 和 C 的纸张大小标准系列。JIS(Japanese Industrial Standards) A 系列和 ISO A 系列是相同的，但是 JIS B 系列和 ISO B 系列有所差别。b[0-6]j 专门用于 JIS B 系列。

† screen　一个宽度和高度分别为 225mm 和 180mm 的特别的纸张大小，在 PC 和放映机上显示时，利用"screen,centering" 和'slide' 的\documentclass 会得到比较好的显示效果。

† paperwidth　表示纸张的宽度，paperwidth=<*length*>

† paperheight　表示纸张的高度，paperheight=<*height*>

† papersize　表示纸张的高度和宽度。paperwidth={<*width*>, <*length*>} 或者 paperwidth=<*length*>

† landscape　表示纸张的方向，当设置了这个参数时，表示纸张为水平放置

† portrait　表示纸张的方向，当设置了这个参数时，表示纸张为竖直放置。和 landscap=false 作用相同。

表示纸张大小（比如：a4paper）和纸张方向的参数（portrait 和 landscape）可以在 \documenclass 中设置。比如，用户可以用 \documentclass[a4paper, landscap]{article} 命令，这个命令和 a4paper 和 landscape 参数在 geometry 中设置的作用是相同的。同样，twoside 和 twocolumn 这两个参数也可以这样设置。

5.2　布局大小

用户可以利用在本节中提到的参数来定义纸张的布局，这个布局不管在什么大小的纸张下都是生效的。这些参数可以帮助用户在不同的纸张大小下定义布局。比如说：利用 a4paper 和 layout=a5paper 这两个参数，geometry 宏包会使用'A5' 纸张的布局在'A4' 的纸张上来计算页边距。layout 这个参数默认和纸张大小相同。并且 layout 这个参数可以在 \newgeometry 中被定义，因此，用户可以在文档中改变布局的大小，但是纸张大小是不能改变的。图 4说明了 layout 和 paper 的区别。

layout　通过纸张的名字定义页面的布局。layout=<*paper-name*>。所有在 geometry 宏包中定义的纸张名字都是可用的。第 5.1有详细的介绍。

layoutwidth　表示页面布局的宽度。layoutwidth=<*length*>。

layoutheight　表示页面布局的高度。layoutheight=<*length*>。

layoutsize　表示页面布局的宽度和高度。layoutsize={<*width*>, <*height*>}，或者layoutsize=<*length*>。

layouthoffset　表示页面布局相对于纸张左边的水平偏移。layoutoffset=<*length*>。

layoutvoffset　表示页面布局相对于纸张顶边的垂直偏移。layoutvoffset=<*length*>。

`layoutoffset` 表示页面布局的水平和垂直偏移。`layoutoffset={<hoffset>, <voffset>}`，或者`layoutoffset=<length>`。

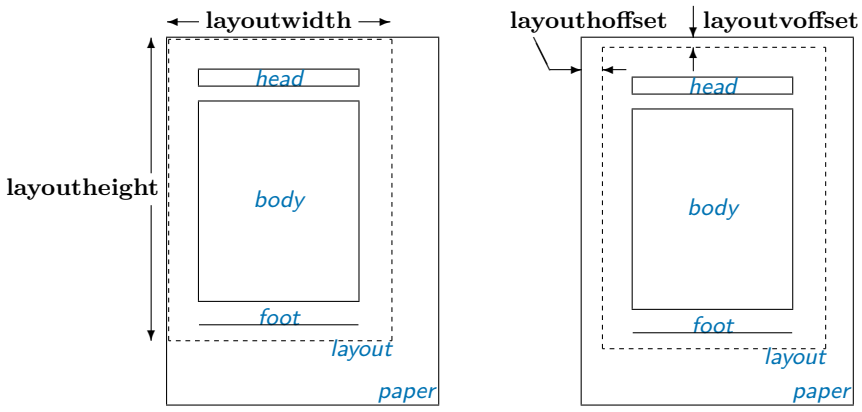


图 4: 页面的尺寸和页面布局大小相关。注意，页面布局的大小默认和纸张大小一致，因此用户在绝大部分情况下都不用单独地给页面布局相关的参数赋值。

5.3 *body* 的尺寸

这节所列出的参数主要用于改变 *total body* 的尺寸。

`hscale` *total body* 的宽度和 `\paperwidth` 的比例。`hscal=<h-scale>`。比如：`hscale=0.8` 和 `\width=0.8\paperwidth` 作用相同。这个参数的默认值是`0.7`

`vscale` *total body* 的高度和 `\paperheight` 的比例。`vscale=<v-scale>`。比如：`vscale=0.9` 和 `height=0.9\paperheight` 的作用相同。这个参数的默认值是`0.7`。

`scale` *totale body* 和纸张大小的比例。`scale={<h-scale>, <v-scale>}` 或者 `scale=<\emph{scale}>`。这个参数的默认值为`0.7`。

`width | totalwidth`
total body 的宽度。`width=<length>` 或者`totalwidth=<length>`。这个参数默认与 `textwidth` 相同。但是如果 `includemp` 被设置为 `true` 时，`width≥ textwidth`。因为这时 `width` 还包括了页边批注的宽度。如果 `textwidth` 和 `width` 同时被赋值，那么`textwidth` 的值会比`width` 具有更高的优先级。

`height | totalheight`
total body 的高度，默认地包括了页眉和页脚的高度。如果`includehead` 或者 `includefoot` 被设置后，`height` 和 `textheight` 都会包括页眉和页脚的高度。`height=<length>` 或者 `totalheight=<length>`。如果 `textheight` 和 `height` 同时被赋值的话，`height` 这个参数会被忽略。

`total` 表示 *total body* 的宽度和高度。
`total={<width>, <height>}` 或者`total=<length>`。

`textwidth` 设置`\textwidth`。表示 *body*（文字区域）的宽度。`textwidth=<length>`。

`textheight` 设置`\textheight`。表示 *body*（文字区域）的高度。`textlength=<length>`。

`text | body` 表示 *body*（文字区域）的宽度和高度。`body={<width>, <height>}` 或者 `text=<length>`。

`lines` 允许让用户通过行数来定义 `textheight` 这个参数。`lines=< 整数 >`。

includehead　将页面的页眉，\headheight 和 \headsep 包括进了*total body*。这个参数默认值为 false，作用和ignorehead 作用相反。参看图 2和图 5。

includefoot　将页脚和 \footskip 包括进了*total body*，作用和ignorefoot 相反。这个参数的默认值为 false。参看图 2 和图 5。

includeheadfoot
　同时设置 includehead 和 includefoot 为 true，这个参数的作用和ignoreheadfoot 作用相反。参看图 2 和图 5。

includemp　在计算水平参数时，将页边批注，\marginparwidth 和\marginparsep 都加入*body*。

includeall　同时将 includeheadfoot 和 includemp 设置为 true。参看图 5。

ignorehead　在计算垂直布局时不考虑页眉，headheight 和 headsep，但是并不改变这些参数的值。这个参数和 includehead=false 作用相同。ignorehead 默认为 true。参看includehead。

ignorefoot　在计算垂直布局时不考虑页脚和 footskip。但是并不改变这两个参数的值。这个参数和 includefoot=false 作用相同。ignorefoot 默认为 true。参看includefoot。

ignoreheadfoot
　同时设置 ignorehead 和 ignorefoot 为 true。参看includeheadfoot。

ignoremp　在计算水平页边距是不考虑页边批注，这个参数默认为 true。如果页边批注的范围超过了页面的范围，当 verbose=true 时，系统会显示警告信息。参见 includemp 和图 5。

ignoreall　同时设置 ignoreheadfoot 和 ignoremp 为 true。参见 includeall。

heightrounded
　这个选项使得 textheight 成为 n倍（n为整数）的 \baselineskip 加上\topskip，以避免在垂直方向上空间的剩余。比如说：如果 \textheight 是 486pt，\baselineskip 是 12pt，\topskip 是 10pt，那么：

$$(39 \times 12\text{pt} + 10\text{pt} =)\ 478\text{pt} < 486\text{pt} < 490\text{pt}\ (= 40 \times 12\text{pt} + 10\text{pt}),$$

　结果 \textheight 被自动优化成了 490pt。

　heightrounded 默认值为 false。

图 5 展示了在不同的页面布局模式下的各种页面布局。页眉和页脚可以通过nohead 和 nofoot 模式来控制，这个模式设置了页眉页脚的高度为 0pt。另一方面，具有 ignore 前缀的选项并不改变响应的尺寸。 下面几个参数可以利用三个在大括号里用逗号分开的值同时设置 *body* 和 *margins*。

hdivide　水平方向的参数（left,width,right） hdivide={<*left margin*>, <*width*>, <*right margin*>} 注意，用户并不需要设置所有的三个参数。使用这个参数最好的方法是设置其中的两个，然后设置另一个为 null，或者'*'。比如，当用户设置hdivide={2cm,15cm, } 时，右边距就会被自动设置成paperwidth-2cm-15cm。

vdivide　垂直方向的参数（top,height,bottom） vdivide={<*top margin*>, <*height*>, <*bottom margin*>}。

divide　divide={<*A*>, <*B*>, <*C*>} 和 hdivide={<*A*>, <*B*>, <*C*>} 和 vdivide={<*A*>, <*B*>, <*C*>} 两者合起来作用相同。

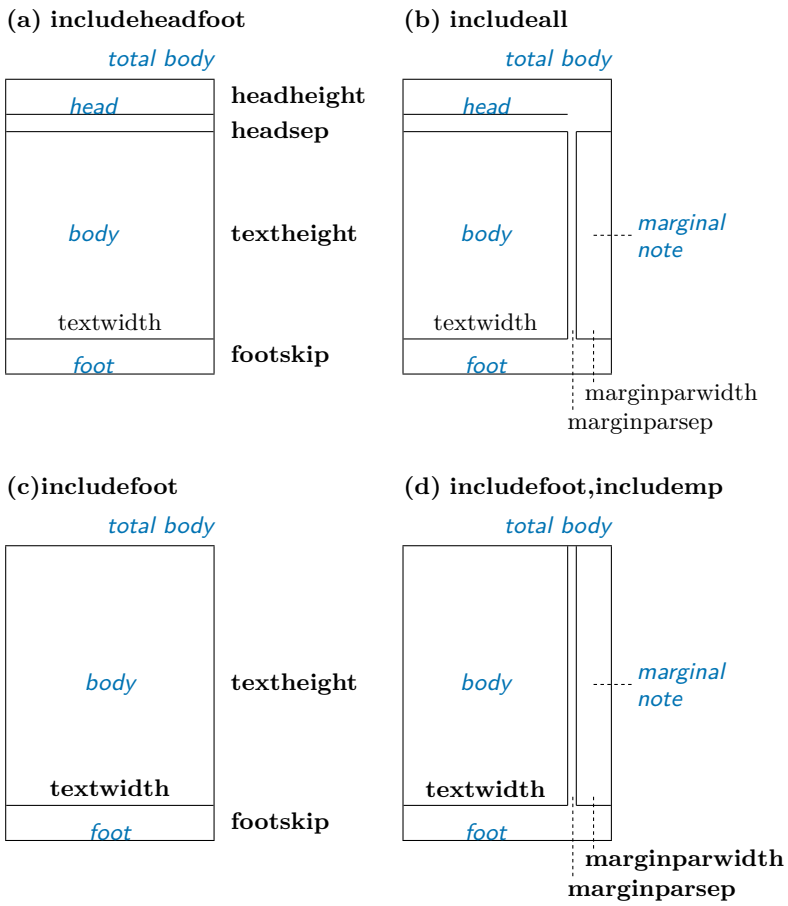


图 5: 不同选项下的 *total body* 的不同尺寸的示例。(a) **includeheadfoot**, (b) **includeall**, (c) **includefoot** and (d) **includefoot,includemp**. 如果 **reversemp** 设置为 **true**，页边批注的位置将会在每一页交换方向。参数 |twoside| 交换页边距和页边批注在反面页上的位置。注意如果有页边批注的话，不管是有参数 **ignoremp** 或者 **includemp=false**，还是超出了页面的范围，页边批注都会被打印出来。

5.4　页边距大小

定义页边距的参数将在下表中列出：

`left | lmargin | inner`
: 表示了 *total body* 的左页边距（一面打印时）或内边距（双面打印时）。换句话说，也就是表示从纸张的最左边到 *total body* 最左边的距离。`left=`<*length*>

`right | rmargin | outer`
: 表示了 *total body* 的右边距或者外边距。`right=`<*length*>。

`top | tmargin`
: 表示了页面的顶部边距。`top=`<*length*>。注意，这个参数和本来的尺寸 `\topmargin` 没有关系。

`bottom | bmargin`
: 表示了页面的底部边距。`bottom=`<*length*>

`hmargin`
: 表示了左边距和右边距。`hmargin=`{<*left margin*>, <*right margin*>} 或者 `hmargin=`<*length*>。

`vmargin`
: 表示了页面的顶部和底部边距。 `vmargin=`{<*top margin*>, <*bottom margin*>} 或者 `vmargin=`<*length*>。

`margin`
: `margin=`{<*A*>, <*B*>} 和 `hmargin=`{<*A*>, <*B*>} 和 `vmargin=`{<*A*>, <*B*>} 联合起来的作用相同。

`hmarginratio`
: 表示水平方向的页边距 `left`(inner) 和 `right`(outer) 之比。<*ratio*> 中的两个数必须用冒号分隔开来。每个值必须为小于 100 的整数。比如说：`2:3` 是正确的，而`1:1.5` 是错误的。默认的比例是：`1:1`（单面印刷），`2:3`（双面印刷）。

`vmarginratio`
: 表示垂直方向的页边距 `top` 和 `bottom` 的比例。默认值是`2:3`。

`marginration | ratio`
: 表示了水平方向和垂直方向各自页边界的比例。`marginratio=`{<*horizontal ratio*>, <*vertical ratio*>} 或 `marginratio=`<*ratio*>。

`hcentering`
: 设置水平方向自动居中，和 `hmarginratio=1:1` 等价。在单面打印（`oneside`）时，这个参数默认为 `true`。参见 `hmarginratio`。

`vcentering`
: 设置垂直方向自动居中，和 `vmarginratio=1:1` 等价。默认为 `false`，参见 `vmarginratio`。

`centering`
: 设置自动居中，和 `marginratio=1:1` 等价。默认值为 `false`。参见 `marginratio`。

`twoside`
: 选择两面印刷模式，这时左边距和右边距的设置会翻页互换。这个参数设置了 `\@twoside` 和 `\@mparswitch`。参见`asymmetric`。

`asymmetric`
: 完成了一个两面印刷的布局，在这个布局里面左边距和右边距不会因翻页而互换（通过设置 `\oddsidemargin=\evensidemargin+bindingoffset` 实现奇偶页边距的不同）。在这个模式下，页边批注一直出现一侧。这个参数可以用作一个较为灵活的两面引述参数。参见 `twoside`。

`bindingoffset`
: 额外加入了对于单面印刷页面左侧的空间，或者是双面印刷页面内侧的空间（装订线）。`bindingoffset=`<*length*>。这个参数在if pages are bound by a press binding . . . 很有用。参看图 6。

`hdivide`
: 参看第 5.3章。

`vdivide`
: 参看第 5.3章。

`divide`
: 参看第 5.3章。

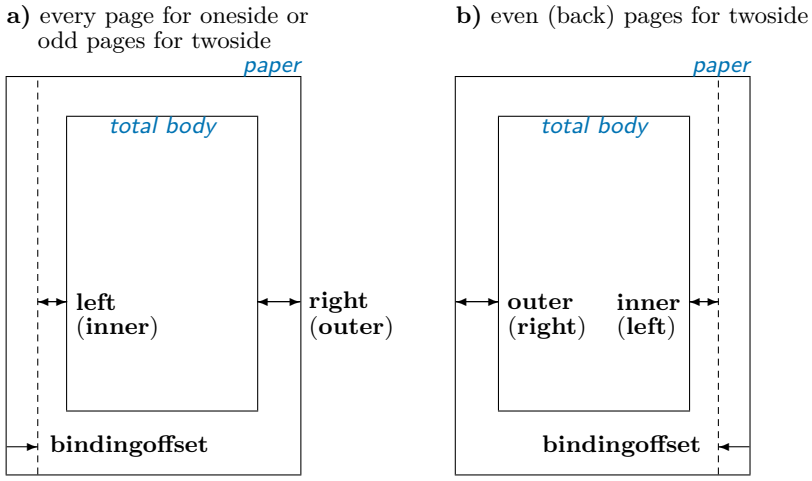


图 6: 参数 **bindingoffset** 给内页边距加入了确定的长度。注意，**twoside** 这个参数会交换水平页边距和页边批注还有 **bindingoffset** 产生的空间（参见 **b**），但是 **asymmetric** 这个参数不交换页边距和页边批注的位置，但是会交换 **bindingoffset** 产生的空间

5.5 原始尺寸

以下的这些参数会覆盖 LaTeX 的原始的尺寸来改变页面布局（参见图 1右边部分）。

`headheight` | `head`
: 改变表示页眉的高度 `\headheight` 这个参数。`headheigt=`<*length*> 或 `head=`<*length*>。

`headsep`
: 改变参数 `\headsep` 的值，这个参数表示了页眉和文字 (*body*) 之间的距离。`headsep=`<*length*>。

`footskip` | `foot`
: 改变参数 `\footskip` 的值，这个参数表示了最后一行文字” 基线” 和页脚的” 基线” 之间的距离。 `footskip=`<*length*> 或者 `foot=`<*length*>。

`nohead`
: 完全不考虑页眉空间。这个参数设置为真的作用和同时设置 `\headheight=0pt` 和 `\headsep=0pt` 作用相同。

`nofoot`
: 完全不考虑页脚空间。这个参数设置为真的作用和同时设置 `\footskip=0pt` 相同。

`noheadfoot`
: 这个参数设置为真的作用和 `nofoot` 和 `nohead` 同时设置作用相同。

`footnotesep`
: 这个参数改变 `\skip` 和 `\footins`，表示了脚注的顶端和 *text body* 的距离。

`marginparwidth` | `marginpar`
: 改变 `\marginparwidth` 参数的值，表示页边批注的宽度。`marginparwidth=`<*length*>。

`marginparsep`
: 改变参数 `\marginparsep` 的值，表示 *body* 和页边批注的距离。`marginparsep=`<*length*>

`nomarginpar`
: 将页边批注的距离减小为 0pt。这个命令和同时设置 `\marginparwidth=0pt` 和 `\marginparsep=0pt` 作用相同。

`columnsep`
: 改变参数 `\columnsep` 的值。表示在 `twocolumn` 模式下两列之间的距离。

`hoffset`
: 改变参数 `\hoffset` 的值。`hoffset=`<*length*>。

`voffset`
: 改变参数 `\voffset` 的值。`voffset=`<*length*>。

`twocolumn`　利用参数 `\@twocolumntrue` 设置 `twocolumn` 模式。`twocolumn=false` 表示单列模式，也可以用参数 `\@twocolumnfalse` 设置。用户也可以设置 `onecolumn`（默认为 `true`）来代替设置 `twocolumn=false`。

`onecolumn`　和 `twocolumn=false` 作用相同。另一方面，`onecolumn=false` 和 `twocolumn` 作用相同。

`twoside`　同时设置参数 `\@twosidetrue` 和 `@mparswitchtrue` 为真。参见第 5.4节。

`textwidth`　直接设置参数 `\textwidth`。参见第 5.3节。

`textheight`　直接设置参数 `\textheight`。参见第 5.3节。

`reversemp` | `reversemarginpar`
让页边批注一直在页面的左侧（内侧）。这个参数并不会改变 `includemp` 模式。默认为 `false`。

5.6　驱动

geometry 宏包支持 `divps`, `divpdfm`, `pdftex`, `xetex` 和 `vtex`。用户通过设置 `dvipdfm` 来使用 dvipdfmx 和 xdvipdfmx，设置 `pdftex` 来使用 pdflatex，设置 `vtex` 来使用 VTEX 环境。这些关于相关驱动的参数和之前的有些差别。驱动可以通过 `driver=`<*driver name*> 来设置，也可以直接通过驱动的名字来设置比如 `pdftex`。geometry 宏包默认这些驱件是正确支持这个系统的。因此，用户在绝大部分情况下都不用设置这些参数。但是，如果用户需要使用 `dvipdfm`，需要明确地声明这个变量。

† `driver`　用 `driver=`<*driver name*> 来声明需要使用的驱动。可以使用的 *driver name* 有：`dvips`, `dvipdfm`, `pdftex`, `vtex`, `xetex`, `auto` 和 `none`。除了 `auto` 和 `none`，其他的参数都可以不通过 `driver=` 来直接设置。 `driver=auto` 让系统自动来决定当前使用的驱动。 `driver=none` 不让系统自动决定使用的驱动，并不设置驱动。这个命令在当你想让其他宏包不再当前的驱动软件设定下工作时比较有用。比如：如果用户想在 geometry 下用 crop 宏包，用户需要在载入 |crop| 之前调用 `\usepackage[driver=none]{geometry}` 命令。

† `dvips`　使用 `\special` 宏命令在 dvi 输出中打印纸张大小。如果用户把 *dvips* 当做一个 DVI-to-PS 的驱动，比如说使用 `\geometry{a3paper, landscape}` 在 A3 之上横向打印一篇文章，用户不需要在 *dvips* 中设置"`-t a3 -t landscape`"。

† `dvipdfm`　和 `dvips` 工作方式相似，不过 `dvipdfm` 进行了水平方向的校正。当用户使用 dvipdfmx 和 xdvipdfmx 来进行 dvi 输出时可以设置这个参数。

† `pdftex`　会自动设置 `\pdfpagewidth` 和 `\pdfpageheight` 这两个参数。

† `xetex`　和 `pdftex` 的作用相同，不过 `xetex` 忽略了在 XeLaTeX 中没有定义的 `\pdf{h,v}origin`。这个参数推荐在第五版中设置。注意'geometry.cfg' 在 TeX Live 中已经不再需要了。如果用户想用一些在 XeTeX 下 dvipdfm 的特殊的功能，除了使用 `xetex`，也可以在 XeLaTeX 下使用参数 `dvipdfm`。

† `vtex`　设置 VTEX 下 `\mediawidth` 和 `\mediaheight` 的参数值。当这个参数被设置的时候，geometry 会自动地决定在 VTEX 中用哪种输出模式（DVI, PDF 或 PS），并进行相应的合适的设置。

如果设置的驱动和使用的排版程序不相符，那么系统将会选择默认的驱动 `dvips`。

5.7 Other options

其他有用的设置在下面列出：

† verbose 在终端上显示参数配置。默认为 verbose=false，将这些参数的结果放入 log 文件中。

† reset 在 geometry 被在入之前设置页面布局和开关等设置回默认值。在 geometry.cfg 文件中的参数设置也会被清除。注意，若设置了 truedimen，这个命令不能重置 pass 和 mag。默认为 reset=false，reset 没有作用但是也不能关闭之前的 reset=true 命令。比如，当用户输入：

```
\documentclass[landscape]{article}
\usepackage[twoside,reset,left=2cm]{geometry}
```

并在 geometry.cfg 设置了 ExecuteOptions{scale=0.9}。这样产生的结果是：landscape 和 left=2cm 仍然有效，scale=0.9 和 twoside 不起作用。

† mag 设置放大倍数 \mag 并根据放大倍数自动设置 \hoffset 和 \voffset 的值。mag=<*value*>。注意，meg=<*value*> 必须是一个整数，1000 是正常的值。比如，在 a4paper 下设置 mag=1414，会提供一个在 a3 大小的纸张上被放大的打印效果。打印的记过会 $1.414(=\sqrt{2})$ 倍于 a4 纸张。字体的放大需要额外的磁盘空间。**注意，设置 mag 必须在任何有'true' 的前缀的命令之前，比如 1.5truein, 2truecm 等等。**参见 truedimen 参数。

† truedimen 改变所有内部声明的尺寸值为 true 尺寸。比如 1in 会改为 1truein。通常这个选项会和 mag 选项一起使用。注意，这个参数对外部声明的尺寸值没有作用。比如，当用户设置了"mag=1440, margin=10pt, truedimen"，页边距并没有变成'true' 而且被放大了。如果用户想设置精确的页边距，则需要像这样设置：
"mag=1440,margin=10truept,truedimen"。

† pass 让所有除 verbose 和 showframe 之外的 geometry 选项和计算结果无效。这是一个独立的命令，可以用来检查又其他宏包和手动设置的的页面布局。

† showframe 为文字区域和页面显示可见的外框和页眉页脚的横线。

† showcrop 在用户自定义的页面布局区域打印剪切记号。

第六章 进程选项

6.1 读取顺序

geometry 将会首先读取 TEX 能够找到的 geometry.cfg 文件。比如在 geometry.cfg 文件中用户定义了 \ExecuteOptions{a4paper}，即设置了 A4 的纸张为默认的。一般说来，用户可以使用在 geometry 中定义的参数加上 \ExecuteOptions{} 来在 geometry.cfg 中使用。

在用户文档的引言区读取参数的顺序如下：

1. geometry.cfg（如果存在的话）
2. 在 \documentclass[<*options*>]{...} 中定义的参数
3. 在 \usepackage[<*options*>]{geometry} 中定义的参数
4. 在 \geometry{<*options*>} 中定义的参数，这个命令可以被多次地调用（reset 命令会取消之前在 \usepackage{geometry} 和 \geometry）中的参数设置。

6.2　参数的顺序

geometry 的参数的定义是和顺序相关的，后来的参数会覆盖之前的或者相同的设置。比如：

`[left=2cm, right=3cm]` 和：`right=3cm, left=2cm` 作用相同。

多次调用的参数会覆盖之前的参数，比如：

`[verbose=true, verbose=false]` 会导致 `verbose=false`

`[hmargin={3cm,2cm}, left=1cm]` 和 `hmargin={1cm,2cm}` 作用相同，这个命令中的左边距是被 `left=1cm` 定义的。

`reset` 和 `mag` 是两个例外。`reset` 参数会取消它之前所有的 geometry 参数设置（除了 `pass`）。如果用户定义

```
\documentclass[landscape]{article}
\usepackage[margin=1cm,twoside]{geometry}
\geometry{a5paper, reset, left=2cm}
```

那么 `margin=1cm`, `twoside` 和 `a5paper` 三个设置将不会起作用，实际上它的排版效果和

```
\documentclass[landscape]{article}
\usepackage[left=2cm]{geometry}
```

相同。`mag` 参数必须设置在所有有'true' 长度之前，比如 `left=1.5truecm`, `width=5truein` 等等。`\mag` 可以设置在这宏包调用之前。

6.3　优先级

有几个方法可以定义 *body* 的尺寸：`scale`, `total`, `text` 和 `lines`。geometry 宏包给了更具体的定义更高的优先级。一下是对 *body* 优先级的定义：

$$\text{priority:} \qquad \text{low} \quad \longrightarrow \quad \text{high}$$

$$\left\{\begin{array}{l}\texttt{hscale}\\ \texttt{vscale}\\ \texttt{scale}\end{array}\right\} < \left\{\begin{array}{l}\texttt{width}\\ \texttt{height}\\ \texttt{total}\end{array}\right\} < \left\{\begin{array}{l}\texttt{textwidth}\\ \texttt{textheight}\\ \texttt{text}\end{array}\right\} < \texttt{lines}.$$

比如说

```
\usepackage[hscale=0.8, textwidth=7in, width=18cm]{geometry}
```

和 `\usepackage[textwidth=7in]{geometry}` 有相同的排版效果。另外：

```
\usepackage[lines=30, scale=0.8, text=7in]{geometry}
```

会导致 `[lines=30, textwidth=7in]` 的排版。

6.4　默认参数

这一节总和了在自动完成系统中所有的默认的设置：

默认的垂直的页边距之比是 2/3，也就是说：

$$\text{top} : \text{bottom} = 2 : 3 \qquad \textit{default}. \tag{6}$$

对于水平页边距的比例，默认值由文档是单面印刷还是双面印刷决定的。

$$\text{left (inner)} : \text{right (outer)} = \begin{cases} 1:1 & \textit{default for oneside,} \\ 2:3 & \textit{default for twoside.} \end{cases} \tag{7}$$

显然，在单面印刷中默认的水平页边距是” 居中” 的。

geometry 宏包对单面印刷的文档有以下的默认设置：

- `sclae=0.7` (*body* = 0.7 × *paper*)
- `marginratio={1:1, 2:3}`（水平方向比例 1:1，垂直方向比例 2:3）
- `ignoreall` （在计算 *body* 大小的时候，页眉页脚和页边注释都不包括在内）

对于使用 `twoside` 参数的 *twosided* 文档来说，默认的设置和 *onesided* 文档相同，只不过水平的页边距比例变也成了 `2:3`。

6.5　自动完成

图 7 说明了有多少参数存在，并说明了怎么解决参数定义的含糊不清的问题。每个坐标轴表示了明确地给 *body* 和页边距定义长度的次数。S(m,b) 用一组数来表示，(*margin*, *body*) = (m, b)。

比如，命令 `width=14cm, left=3cm` 应该被分到 S(1,1) 类中，这是一个完整的定义。如果用户加上了 `right=4cm`，那么组命令将会分到 S(2,1) 类中，也就会产生重复定义的情况。如果仅仅定义了 `width=4cm`，那么将会被分入 S(1,0) 类中，也就是并没有完全定义。

geometry 宏包具有自动完成的机制，利用这个机制，如果页面布局参数没有完全定义或者重复定义了，geometry 将会根据默认值或者其他关系来确定没有定义的参数。这就是参数定义和自动完成的规则。

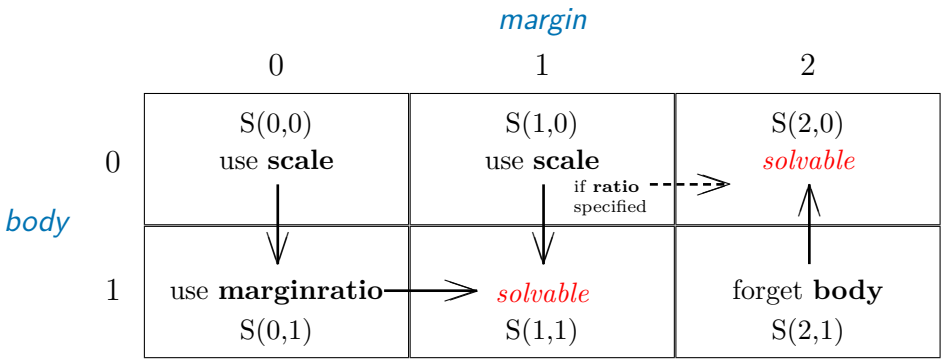


图 7: 定义了 S(0,0) 到 S(2,1) 各种类别和参数的自动完成的规则（箭头方向）。行列的数字表示了单独给页边距和 *body* 定义了长度的次数。S(m,b) 表示页边距的长度定义了 m 次，而 *body* 定义了 b 次。

S(0,0) 没有参数被定义，geometry 宏包设置 *body* 为默认值：`scale=0.7`。
比如说，`width` 被设置为 0.7 × `layoutwidth`。注意，默认的 `layoutwidth` 和 `layoutheight` 和 `\paperwidth` 和 `\paperheight` 相等。然后，S(0,0) 转到 S(0,1)，参见 S(0,1).

S(0,1) 只有 *body* 被声明，比如 `width=7in`, `lines=20`, `body={20cm,24cm}`, `scale=0.9` 等等。 geometry 宏包利用页边距的比例设置页边距。如果页边距的比例没有被声明，那么将会使用默认值。默认的垂直方向的页边距定义如下：

$$\texttt{top} : \texttt{bottom} = 2 : 3 \qquad \textit{default} \tag{8}$$

对于水平方向的页边距来说，默认值取决于文档是单面打印还是双面打印。

$$\texttt{left (inner)} : \texttt{right (outer)} = \begin{cases} 1:1 & \textit{default for oneside,} \\ 2:3 & \textit{default for twoside.} \end{cases} \tag{9}$$

比如，在 A4 纸张中定义了 `height=22cm`，geometry 宏包按如下的方式计算 `top`：

$$\begin{aligned} \texttt{top} &= (\texttt{layoutheight} - \texttt{height}) \times 2/5 \\ &= (29.7 - 22) \times 2/5 = 3.08(\text{cm}) \end{aligned} \tag{10}$$

因此，`top` 页边距和 *body* 的 `height` 已经被决定了，也就是说垂直方向的参数确定了，因此进入 S(1,1) 类别，所有的参数都能够确定了。

S(1,0) 只有页边距确定了，比如 `bottom=2cm`, `left=1in`, `top=3cm` 等等。

- **如果页边距比例没有确定**，geometry 宏包设置 *body* 为默认的`scale`$(= 0.7)$。比如，如果 `to=2.4cm` 确定了，geometry 宏包设置

  $$\texttt{height} = 0.7 \times \texttt{layoutheight} \quad (\text{默认} = \texttt{0.7\paperheight})$$

  这样 S(1,0) 状态就会进入到 S(1,1) 状态，`bottom` 就可以由 `layoutheight` $-$ (`height` $+$ `top`) 计算，并最终在 A4 纸张上得到 6.51cm，如果页面布局大小和纸张大小相同的话。

- **如果页边距的比例被定义了**，比如 `hamerginration={1:2}`, `vratio={3:4}` 等等。geometry 会通过已经设置的页边距的比例来设定其他的页边距。比如，定义了"`top=2.4cm, vratio={3:4}`"， geometry 通过计算设置 `bottom` 为 `3.2cm`。

  $$\texttt{bottom} = \texttt{top}/3 \times 4 = 3.2\text{cm}$$

  从而进入下一个状态 S(2,0)。

注意，第四版之前的版本曾经利用页边距比例来设置也边距。在第五版中，在相同的设置下，将会出现和第四版之前不同的排版效果。比如说，如果只定义了 `top=2.4cm`，在第四版之前的版本中将会得到 `bottom=2,4cm` 的排版效果，但是在第五版中会得到 `bottom=6.51cm` 的排版效果。

S(2,1) *body* 和两个 *margins* 都被声明了，比如 `vdivide={1in,8in,1.5in}`,"`left=3cm,width=13cm,right=4cm`" 等等。因为 geometry 宏包给予 *margins* 更高的优先级，所以如果参数被重复定义，geometry 将会忽略并重置 *body* 的定义。比如，如果用户定义了 "

```
\usepackage[a4paper,left=3cm,width=13cm,right=4cm]{geometry}
```

`width` 将会被设置为默认值 14cm，因为 A4 纸张的宽度是 21cm。

第七章　在文档中改变页面布局

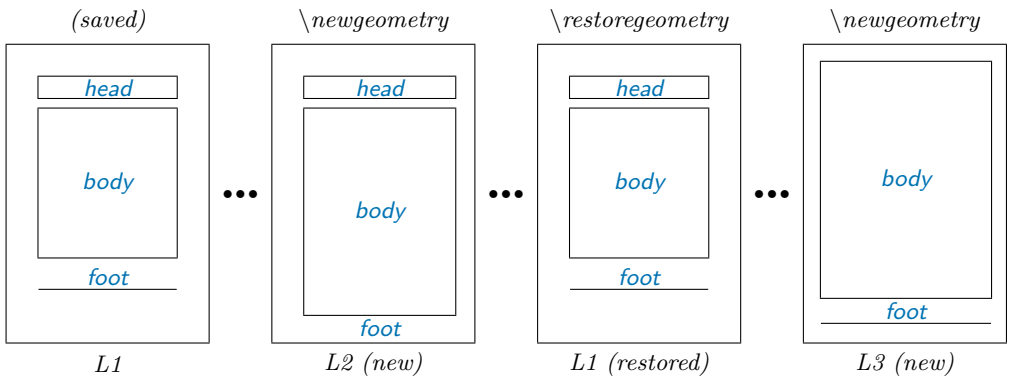
第五版提供了两个新的命令 \newgeometry{···} 和 \restoregeometry，这两个命令允许了用户在文档中部改变页面尺寸。和引言区的 \geometry 不同，\newgeometry 命令只在 \begin{document} 之后有效，并会重置除和纸张布局相关的参数之外的所有参数。和纸张布局相关的参数包括：landscape, portrait 和纸张大小的参数 (papersize, paper=a4paper 等等)，这些参数不能通过 \newgeometry 来改变。

\restoregeometry 命令将会恢复在引言区 (\begin{document} 之前) 用\usepackage{geometry} 和 \geometry 定义的和页面布局相关参数。

注意，\newgeometry 和 \restoregeometry 将会在他们被调用的地方插入 \clearpage。

以下是一个在文档中部改变页边布局的例子。L1 布局通过 hmargin=3cm 定义（left 和 right 页边距是 3cm），然后会在文档中改成 L2 布局，L2 通过 left=3cm, right=1cm 和 bottom=0.1m 定义。L1 布局通过 \restoregeometry 被恢复。

```
\usepackage[hmargin=3cm]{geometry}
\begin{document}
    Layout L1
\newgeometry{left=3cm,right=1cm,bottom=0.1cm}
    Layout L2 (new)
\restoregeometry
    Layout L1 (restored)
\newgeometry{margin=1cm,includefoot}
    Layout L3 (new)
\end{document}
```



\savegeometry{<*name*>} 和 loadgeometry{<*name*>} 可以使用户在文档中使用更多不同的页面布局。比如：

```
\usepackage[hmargin=3cm]{geometry}
\begin{document}
    L1
\newgeometry{left=3cm,right=1cm,bottom=0.1cm}
\savegeometry{L2}
    L2 (new, saved)
```

```
\restoregeometry
      L1 (restored)
\newgeometry{margin=1cm,includefoot}
      L3 (new)
\loadgeometry{L2}
      L2 (loaded)
\end{document}
```

第八章　例子

1. 一个单面印刷的布局，文字区域在纸张中央。下面的例子有相同的排版结果，因为水平方向的页边距比例对单面印刷默认为 1:1。

   - centering
   - marginratio=1:1
   - vcentering

2. 一个双面印刷的布局，内侧用于装订线的位移为 1cm。

   - twoside,bindingoffset=1cm

   在这个情况中，textwidth 比默认的双面文档要短 $0.7 \times \mathbf{1cm}$ ($= 0.7$cm) 因为默认的 *body* 的宽度的比例是 scale=0.7，也就是说 width $= 0.7 \times$ layoutwidth (默认 $= 0.7$\paperwidth)

3. 一个左，右和顶部页边距为 3cm, 2cm 和 2.5in 的页面布局。文档高度为 40 行，页眉页脚包含在 *total body* 中。

   - left=3cm, right=2cm, lines=40, top=2.5in, includeheadfoot
   - hmargin={3cm,2cm}, tmargin=2.5in, lines=40, includeheadfoot

4. 一个 *total body* 为 10in ，底部页边距为 2cm，默认宽度的布局。顶部的页边距将会自动计算。以下的每个结果会获得相同的页面布局。

   - vdivide={*, 10in, 2cm}
   - bmargin=2cm, height=10in
   - bottom=2cm, textheight=10in

   注意，*head* 和 *foot* 并没有包含在 *total body* 的 height 里面。如果加上 includefoot 会使得 \footskip 也包含在 totalheight 中。因此，在这种情况当中，由于 \footskip 被包括进了 total body，\textheight 在前一种页面布局中会比后面一种短（准确地说是短 10in）。也就是说，当 includefoot=true 时：height = textheight + footskip。

   - bmargin=2cm, height=10inm includefoot
   - bottom=2cm, textheight=10in, includefoot

5. 一个 textwidth 和 textheight 占据纸张的 90%，并且 *body* 中央放置的页面布局。只要 layoutwidth 和 layoutheight 没有被改变，下面的每个处理结果都会获得相同的页面布局：

- `scale=0.9, centering`
- `text={.0\paperwidth,.9paperheight}, ratio=1:1`
- `width=.9\paperwidth, vmargin=.05\paperheight, marginratio=1:1`
- `hdivide={*,0.9\paperwidth,*}, vdivide={*,0.9\paperheighit,*}`（对于单面印刷的文档来说）
- `margin={0.5\paperwidth,0.05\paperheight}`

用户可以加上 `heightrounded` 来避免"underfull vbox warning"，比如：

```
Underfull \vbox (badnes 10000) has occurred while \output is active
```

关于 `heightrounded`, 请参见第 5.3章。

6. 一个页边批注宽度为 `3cm` 并包括在 *total body* 中的页面布局。以下的例子都会获得相同的排版效果：
   - `marginparwidth=3cm, includemp`
   - `marginpar=3cm, ignoremp=false`
7. 一个 *body* 占据 A5 纸张所有页面的的横版排版。以下的例子具有相同的排版效果：
   - `a5paper, landscape, scale=1.0`
   - `landscape=TRUE, paper=a5paper, margin=0pt`
8. 一个适应屏幕 PC 和录像放映机大小的页面排版

```
\documenclass{slide}
\usepackage[screen,margin=0.8in]{geometry}
...
\begin{slide}
   ...
\end{slide}
```

9. 一个字体和排版空间从 A4 扩展到 A3 的页面排版。以下的情况中，会获得 A3 纸张的排版。
   - `a4paper, mag=1414`

   如果如果用户希望获得两倍大的字体但是不要改变页面的大小的排版，可以输入：
   - `letterpaper, mag=2000, truedimen`

   用户可以加入 `dvips` 参数，这个参数对使用 `dviout` 和 `xdvi` 用正确的纸张大小对页面进行预览很有用。
10. 在载入 geometry 宏包之前改变首页的页面布局，并让其他页为默认值。使用 `pass` 参数，`\newgeometry` 和 `\restoregeometry`。

```
 \documentclass{book}
 \usepackage[pass]{geometry}
  % 'pass' 会忽视宏包的布局
  % 所以原始的 'book' 布局会被使用和记录
\begin{document}
\newgemetry{margin=1cm}% 盖面第一页的布局
      Page 1
\restoregeometry % 恢复原始的 'book' 页面布局
      Page 2 等
\end{document}
```

11. 一个复合的页面布局

```
\usepackage[a5paper, landscape, twocolumn, twoside,
left=2cm, hmarginratio=2:1, includemp, marginparwidth=43pt, bottom=1cm,
 foot=.7cm, includefoot, textheight=11cm, heightrounded, columnsep=1cm,
 dvips, verbos]{geometry}
```

试试输入这些参数，自己来看看结果。:-)

第九章　已知的问题

- 利用 mag $\neq$ 1000 和 truedimen，paperwidth 和 paperheight 在 verbose 模式下会显示和在 PDF 中真实出现的尺寸不同。PDF 中显示的是正确的尺寸。
- 利用 mag $\neq$ 1000，无 truedimen 和 hyperref，hyperref 必须在 geometry 宏包之前载入，否则在 PDF 中会显示错误的结果。
- 利用 crop 宏包和 mag $\neq$ 1000，center 参数在 crop 中会存在问题。