The `Swiss Knife` package

Vincent Hugot

January 19, 2006

0 The purpose of the `SKnife` package

0.1 A generalist package...

A Swiss knife is a chameleon tool, meant to serve manifold purposes. Although one must admit it serves none of them as efficiently as a specialized tool would, its versatility made it reliable enough to cause countless men (and women) through the ages to buy one and carry it along with them in their exotic travels as well as their daily lunch hours. The Swiss knife didn't invent anything ; it merely packed well-known tools which had nothing but their usefulness in common in a most convenient package. The spirit of the `SKnife` package is very much alike : it doesn't claim to provide any sharp arcane solution to some very specific and undreamed-of puzzle. Its one and only aim is that, once it is loaded, people starting a blank document won't waste hours fine-tuning the layout, wondering which package they should load, what kind of purely TEXnical commands they should define and so forth and so on. The `SKnife` package automatically loads *most* widespread packages and provides *most* people with default layouts and macros meeting their needs.

0.2 ... turned towards mathematics

Then again, it seems difficult to satisfy everyone. Besides, this is not my aim either. Having to type mathematics, I hand-crafted these macros in order to make *my* life easier than it was, and I must confess I didn't really have the concerns of chess or music lovers in mind at that time. So if you use LATEX for typesetting chess boards or scales and scores, or even Chemistry or... whatever, you won't find anything useful in here and should be looking for a specialized package. Some such 'specialized' packages *are* loaded by `SKnife`, as the `crosswrd` package for typesetting crosswords grids; however, I won't document them here but content myself with explaining how I altered the way they work. This necessary warning coming to an end, let's get to the heart of the matter.

1 General advice

I think that the best way to build LATEX files is :

1. Forget about PDF-LATEX: it does not understand any PostScript instruction, which is a severe drawback as the most interesting macros of this package make an heavy use of PostScript. And the final result with PDF-LATEX is not even optimal anyway: the files are much larger (in terms of computer memory) than they could be. And finally, when you print from PDF, the fraction bars appear to be too thick.

2. Instead, use DVI→PS (command line "-P pdf -I c"), and print from GSView.

3. If you need a PDF, use GSView to convert PS→PDF. Play with the options to get exactly what you want.

4. The only case when you could use PDF-LaTeX is when you have included a bitmap in your file, want a PDF, and have no PostScript at all. But this can be done just as well with a PS→PDF conversion with the proper compression options.

5. Keep in mind also that the `hyperref` package loaded with `SKnife` is set for DVI→PS→PDF. I you use PDF-LaTeX the links won't work. You would then have to use the `nohyperref` option and load the `hyperref` package yourself.

2 The bundle of packages

The following packages are automatically loaded when `SKnife` itself is loaded :

[*dvips*]graphics [*ansinew*]inputenc amsmath calc amssymb lastpage fancyhdr eurosym stmaryrd pifont mathrsfs verbatim moreverb hhline array color graphicx bm slashbox fancybox url multicol picins ifthen upquote shortvrb crosswrd [*normalem*]ulem pst-3d pst-char pst-coil pst-eps pst-fill pst-grad pst-node pst-plot pst-text pst-tree eucal mathrsfs (hyperref)

If you want to know what these do, see their own documentation. They are all part of most standard LaTeX distributions. Please notice that some other packages might be loaded as well, depending upon the options you pass to the `SKnife` package. See the appropriate section for more information about them.

3 The available options

alcal Provides an alternative *mathcal* typeface. The command `\mathcal` is redefined.

defhf Initializes the header and foot of the page with default values.

deflayout Changes the layout of the page to default values.

english The typographical conventions of the English language will be respected. This option causes the [*english*]`babel` package to be loaded. In absence of this option, the French conventions will prevail, therefore the [*french*]`babel` package will be loaded. This, however, can be avoided. See the `nobabel` option. (It also sets `crosswrd` package text in french but this is an epiphenomenon.)

hugematrices The maximum number of columns in the `matrix` environnement is set to 20, as opposed to 10, which is the default value.

index Generates an index at the end of the document, seeing that index entries have been created with the `\index` command.

nobabel Prevents the `babel` package to be loaded at all. This option overruns any language option that might be passed to the package. Use this to avoid conflicts if you intend to load `babel` with another language option in your file, or if you use a competing package (such as `french`).

nohyperref Disables the links in the document. Prevents the `hyperref` package from being loaded. Use this if you intend only to print your document and dont want the link's colors on your paper, or if you want to load the `hyperref` package yourself, with your own options.

sanserif Changes the default typeface into this one. This goes for the equation mode too. Additionally, the behaviours of the `\textsf` and `\textrm` functions are changed so that the first now switches to the classic Roman typeface, and the latter to a sans serif one. This is meant to preserve the `\textrm` function as the indicator of the "default" typeface. This option causes the `cmbright` package to be loaded.

noshortverb When loading `SKnife`, the symbol `|` becomes a special shortcut for the `\verb` command. With this option, `|` recovers it's usual meaning. See the `shortvrb` documentation and look for the `\MakeShortVerb` command if you are eager to know more about that.

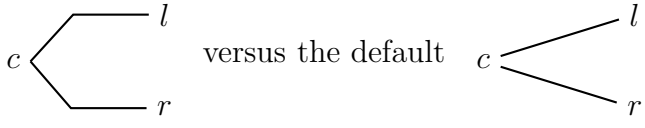
vrule Adds a thin vertical rule between columns of text, whether they are created by the standard `twocolumn` class option or the `SKnife \twocol` command.

4 Micro-changes in LaTeX behaviour

When the package loads, it alters slightly the general behaviour of LaTeX (for good reasons, of course). The changes that might result from the choice of an option (or the absence thereof) are not listed here.

- ◇ The title page no longer yields a page number. If you want it back, you may use the `\formermaketitle` command.
- ◇ The list symbols of the `itemize` environment are changed into nicer yet as classic-looking ones.
- ◇ All sectioning commands, from `\part` to `\subparagraph`, now yield numbers.
- ◇ All sections, from parts to subparagraphs, now appear in the table of contents.
- ◇ The `enumerate` environment now yields bold numbers for improved legibility.
- ◇ The footnote numbers appear now surrounded by parenthesis, to prevent them from being mistaken for mathematical powers (a^2). An appropriate space is also automatically added before the number so you don't have to worry about that anymore[(1)].
- ◇ New default values are set for the `pstricks` package, mostly when trees and their spacing are involved.
- ◇ A new `pstricks` edge is created for trees. It is `\armedge` and works thus :

```
\pstree[edge=\armedge]{\Tr{c}}{\Tr{l}\Tr{r}}
```



- ◇ Some commands are redefined : `\r` etc.

[(1)] A footnote example

5 Generalist macros

5.1 Text-related macros

5.1.1 Text-positioning commands

\centre ⟨*text*⟩ Centers text.

\evenodd ⟨*number*⟩ ⟨*if even*⟩ ⟨*if odd*⟩ Yields a different result if the number is odd or even. Useful for page headers.

\noi The following paragraph is not indented. Shortcut for `\noindent`.

\rem Yields a negative space of five millimeters. Useful when LaTeX adds much too much space after an equation as it sometimes does.

\T ⟨*spec*⟩⟨*body*⟩ Shortcut to a tabular environment.

\tsb ⟨*bellow*⟩ **\tss** ⟨*above*⟩ Superscripts and subscripts in text mode : $\text{thing}_{\text{bellow}}$ and $\text{thing}^{\text{above}}$. `\tss` is a mere shortcut for `\textsuperscript`.

\twocol ⟨*text*⟩ Puts the text in two balanced columns.

5.1.2 Miscellaneous text-related macros

\exhaustfont ⟨*fontname*⟩ Prints all the symbols in a font. If the font name isn't valid the standard roman font is displayed instead. Valid font names are "wasy" and "ding" etc. Of course, it depends on your local LaTeX installation. For technical reasons, this function cannot be used more than once in a document.

\latex Yields LaTeX. Shortcut to `\LaTeX`.

\man \woman \permil \check \nocheck Yield ♀ ♂ ‰ ✓ ✗, respectively.

\mlc Shortcut for the `\multicolumn` command used in tabular environnements.

5.2 Mathematical macros

5.2.1 Added default functions

The tree hyperbolical functions ch, sh and th are now available.

5.2.2 The delimiter commands

5.2.2.1 Adjusted pairs The syntax is **\p_** ⟨*math*⟩. They yield an auto-adjusting pair of delimiters surrounding the mathematical expression.

They are the commands : `\p \pd \pdd \pc \pcc \pa \pb \pco \pf`

For instance,

`\[ \p{-\pd{-\pdd{-\pc{-\pcc{-\pa{-\pb{-\pco{-\pf{-\f12}}}}}}}}}\]` yields

$$\left(-\left|-\left\|-\left[-\left[\!\left[-\left\{-\left\langle-\right]-\left\lfloor-\frac{1}{2}\right\rfloor\left[\right\rangle\right\}\right]\!\right]\right]\right\|\right|\right)$$

Please note that there is a 'shortcut' for `\pd` (although there are more letters in it's name than in the original function...) : `\abs`. It obviously stands for the mathematical function abs : $x \longmapsto |x|$.

5.2.2.2 Stand alone left and right adjusted delimiters The only difference with the pair commands is that the names of the right delimiters get an "r" and those of the left ones a "l". You just can't make simpler than that ! The only thing to remember is that `\pf`, unlike the other commands, hasn't got any left or right version. This stems from the fact that this macro is meant to stand for the *floor* mathematical function and therefore should'nt be used as a common delimiter.
`\lp \lpd \lpdd \lpc \lpcc \lpa \lpb \lpco` and
`\rp \rpd \rpdd \rpc \rpcc \rpa \rpb \rpco`
With the example above :

$$\Big(-\Big|-\Big\|-\Big[-\Big[\!\Big[-\Big\{-\Big\langle-\Big]--\frac{1}{2}$$

$$---------\frac{1}{2}\Big[\Big\rangle\Big\}\Big]\!\Big]\Big\|\Big|\Big)$$

5.2.2.3 Nesting aids When it comes to typesetting mathematics, doing things rightly is not quite enough. For instance, this deeply nested mathematical expression

```
\[  P\p X= a_0+x\p{a_1 + x\p{a_2 + x\p{\cdots + x a_n}}} \]
```

$$P\left(X\right)=a_0+x\left(a_1+x\left(a_2+x\left(\cdots+xa_n\right)\right)\right)$$

is 'right' in so far as the idea is there and the expression is mathematically correct. Yet it doesn't look too good. And in LaTeX, everything *must* look good![(2)] In such a case it is always worthwhile to show the nesting more accurately ; the `\gi \gii \giii \giv \gv \gvi \gvii \gviii \gix` and `\gx` macros are here to make this much easier : although invisible, they force all the surrounding auto-adjusting delimiters to look as if they enclosed vertically larger expressions. Let's demonstrate :

```
\[  P\p X= a_0+x\p{a_1 + x\p{a_2 + x\p{\cdots + x a_n}\gi}\gii} \]
```

$$P\left(X\right)=a_0+x\left(a_1+x\left(a_2+x\left(\cdots+xa_n\right)\right)\right)$$

This works by using ghosts of fixed sizes. These macros can also be used in arrays and tabular environments to equalize the spacing in equations systems for instance.
For those needing still finer tuning, the `\g` macro stands as a shortcut for `\vphantom`, the mother of each and every one of these nesting aids. The exact sizes of the `\gi` commands are shown bellow :

```
\[()\p{\gi}\p{\gii}\p{\giii}\p{\giv}\p{\gv}\p{\gvi}\p{\gvii}
\p{\gviii}\p{\gix}\p{\gx}\]
```

$$()\,()\,()\,()\,()\,()\,()\,()\,()\,()\,()$$

[(2)] There I shall only quote Donald E. Knuth himself : "with TeX, the goal is to produce the *finest* quality; this requires more attention to details...". (*The TeXbook* page 1)

Please notice that they make the \bigl and C^ie commands obsolete :

```
\[\Bigl\{\iff\lpa{\gii} \quad \Bigl( a \Bigr)\iff\p{a\gii}\]
```

$$\left\{ \iff \left\{ \quad \left(a\right) \iff \left(a\right)$$

Imagine what the code of a very deeply nested, very long expression would look like without the \g.. commands... With these, you save typing time and avoid ludicrous mistakes like pairs of parenthesis facing each other yet of different sizes. Even if you aren't truly sensitive to aesthetical considerations, and see them as a waste of time, just keep in mind that mathematical expressions are often complex enough without added difficulty from typographical deficiencies. Beautiful pages are, in most cases, easier to read and thus simpler to understand. To put it in a nutshell : *don't you neglect the nesting aids !*

5.2.3 Display commands

rectresult ⟨*math*⟩ Displays a centered, boxed mathematical expression in bold typeface. This is most useful when you want to emphasize some very important result, hence the name.

```
\rectresult{ \sumn0\infi u_n(x)= \f1{1-a}}
```

$$\boxed{\sum_{n=0}^{\infty} u_n(x) = \frac{1}{1-a}}$$

result ⟨*math*⟩ The same as \rectresult, except that the edges of the box are smoothed.

```
\result{ \sumn0\infi u_n(x)= \f1{1-a}}
```

$$\boxed{\sum_{n=0}^{\infty} u_n(x) = \frac{1}{1-a}}$$

sys(b) ⟨*lcr...*⟩⟨*math tabular*⟩ Shortcut for an array environment with a left brace (\sys) or a left bar (it's variant \sysb)

```
\[ f\p x = \sys{lcr}{0 & \text{if} & x>0\\1 & \text{if} & x\< 0}\]
```

$$f\left(x\right) = \left\{ \begin{array}{lll} 0 & \text{if} & x > 0 \\ 1 & \text{if} & x \leqslant 0 \end{array} \right.$$

```
\[ f\p x = \sysb{lcr}{0 & \text{if} & x>0\\1 & \text{if} & x\< 0}\]
```

$$f\left(x\right) = \left| \begin{array}{lll} 0 & \text{if} & x > 0 \\ 1 & \text{if} & x \leqslant 0 \end{array} \right.$$

ds **ts** Shortcuts for \displaystyle and \textstyle.

aligne ⟨*math*⟩ Shortcut for the align environment.

```
\aligne{P(x) &= \sumn0d a_n X^n \\ &= a_0+a_1x+a_2x^2+\cdots+a_dx^d}
```

$$\begin{aligned} P(x) &= \sum_{n=0}^{d} a_n X^n \\ &= a_0 + a_1 x + a_2 x^2 + \cdots + a_d x^d \end{aligned}$$

A ⟨***lcr...***⟩ ⟨***math tabular***⟩ Shortcut for an array. Works exactly the same way `\sys` does.

build ⟨***arg***⟩_⟨***arg bellow***⟩^⟨***arg above***⟩ Yields $\overset{\text{arg above}}{\underset{\text{arg bellow}}{\text{arg}}}$ Beware : unlike that of the conventional sub and superscripts, the order in the arguments can **not** be changed.

5.2.3.1 Shortcuts and symbols Here are various simple shortcuts : `\r` for `\sqrt`, `\f` for `\frac`, `\v` for `\overrightarrow` (used for vectors : $\overrightarrow{v} = \overrightarrow{AB}$), `\im` for the implication $\Longrightarrow$, `\mi` for the reciprocal implication $\Longleftarrow$, `\ortho` for $\perp$. The common set symbols are also available, and work in text mode as well as in math mode. The command names are as simple as one can imagine : for instance `\N` yields $\mathbb{N}$. $\mathbb{Z}$, $\mathbb{Q}$, $\mathbb{R}$, $\mathbb{C}$, $\mathbb{K}$ and $\mathbb{P}$ work the same way. There are also `\lto` for $\longrightarrow$, `\assoc` for $\longmapsto$.

Those shortcuts are for derivatives : `f(x) \d x` produces $f(x)\,\mathrm{d}x$. Note that the d is displayed in a roman typeface and preceded by a space, just as legibility and conventions want it to be. The same thing goes for `\dr` : $f(x,y)\,\partial x$. They are used in all the following commands :

```
\[  \dern fx,\derp fx,\dernp fx, \derpp fx  \]
```

$$\frac{\mathrm{d}f}{\mathrm{d}x}, \frac{\partial f}{\partial x}, \left(\frac{\mathrm{d}f}{\mathrm{d}x}\right), \left(\frac{\partial f}{\partial x}\right)$$

There is also a shortcut for the n^{th} derivative of a function : `f\der n (x)`$\Longrightarrow f^{(n)}(x)$.

Sums and products are very common objects. Half a second spared each time you type one means hours spared each year...

```
\[  \sumn0\infi\sump0n\sumk0n\sumi0n\sumj0n \]
```

$$\sum_{n=0}^{\infty}\sum_{p=0}^{n}\sum_{k=0}^{n}\sum_{i=0}^{n}\sum_{j=0}^{n}$$

```
\[ \prodn0\infi\prodp0n\prodk0n\prodi0n\prodj0n \]
```

$$\prod_{n=0}^{\infty}\prod_{p=0}^{n}\prod_{k=0}^{n}\prod_{i=0}^{n}\prod_{j=0}^{n}$$

Let's go back to some miscellaneous shortcuts : `\<, \>, \infi`$\Longrightarrow \leqslant, \geqslant, \infty$ `\x` for `\text`. Also two hand-crafted symbols for circular integrals :

```
\[  \iinto_D f(x,y) \iiinto_D f(x,y,z) \]
```

$$\oiint_D f(x,y) \oiiint_D f(x,y,z)$$

application [***0/1***] ⟨***from set***⟩⟨***to set***⟩⟨x⟩⟨$f(x)$⟩ (If the optional argument is omitted, the default value is one) A clear-cut example will do more good than a long speech:

```
\[f:\application \C\R x{\abs x}\ f:\application[0] \C\R x{\abs x}\]
```

$$f : \begin{vmatrix} \mathbb{C} & \longrightarrow & \mathbb{R} \\ x & \longmapsto & |x| \end{vmatrix} \quad f : \left| \begin{smallmatrix} \mathbb{C} \longrightarrow \mathbb{R} \\ x \longmapsto |x| \end{smallmatrix} \right.$$

pr [***B***] ⟨***A***⟩ Yields the probability $\mathbb{P}_B(A)$

limi [>]⟨***x***⟩⟨***a***⟩ ⟨***f(x)***⟩ `\limi[<]xa{f(x)}`$\Longrightarrow \lim_{\substack{x\to a\\<}} f(x)$

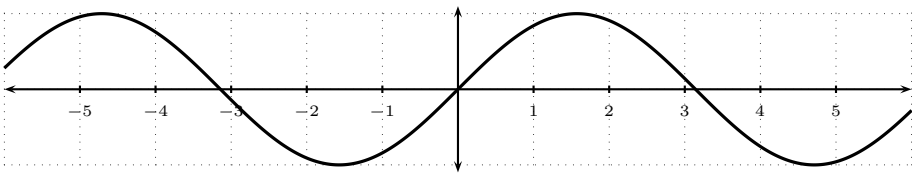
6 Drawing graphs

These functions provide efficient means to draw the curves of monadic functions, such as $f : x \longmapsto \sin x$. They are heavily based upon the `PsTricks` package, therefore its parameters may influence the appearance of the curves and PDF-LaTeX won't work anymore once a curve is drawn.

6.1 Drawing functions, explicit syntax

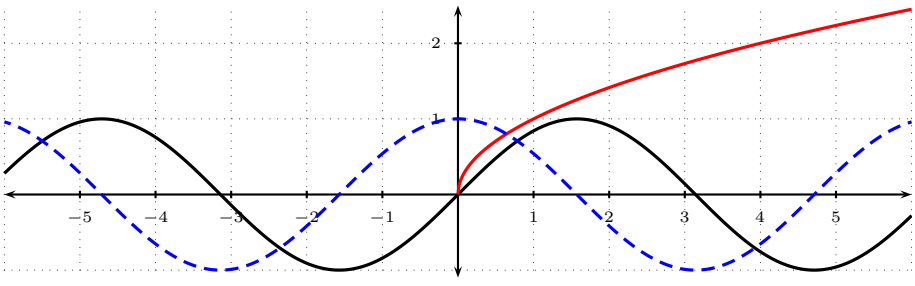
\graph f(x)=⟨*expr*⟩,x=⟨*xmin*⟩..⟨*xmax*⟩,y=⟨*ymin*⟩ ..⟨*ymax*⟩(x=⟨*xunit*⟩,y=⟨*yunit*⟩)
The syntax, peculiar though it is, seems quite transparent. But be careful : *expr* is a PostScript operation of the variable x. Therefore you must learn some PostScript in order to draw graphs. Be careful : you will encounter a PostScript error if there are points between x_{min} and x_{max} for which $f(x)$ does not exist. For instance, anyone thoughtlessly trying to draw $\sqrt{x}$ between -5 and 5 will have a nice surprise when they read the DVI file...

```
\graph f(x)=\pssin x,x=-6..6,y=-1.1..1.1(x=1cm,y=1cm)
```



\graphs ⟨*graphs list*⟩,x=⟨*xmin*⟩..⟨*xmax*⟩,y=⟨*ymin*⟩ ..⟨*ymax*⟩(x=⟨*xunit*⟩,y=⟨*yunit*⟩)
This function is far more flexible than `\graph`. It creates a virgin grid upon which the graphs of the graphs list are drawn. The graphs list consists of `\addgraph` and `\addrgraph` commands, as well as any instruction meant to change the behaviour of the graphs, such as `\graphcolor` or any PostScript or `PsTricks` instruction.

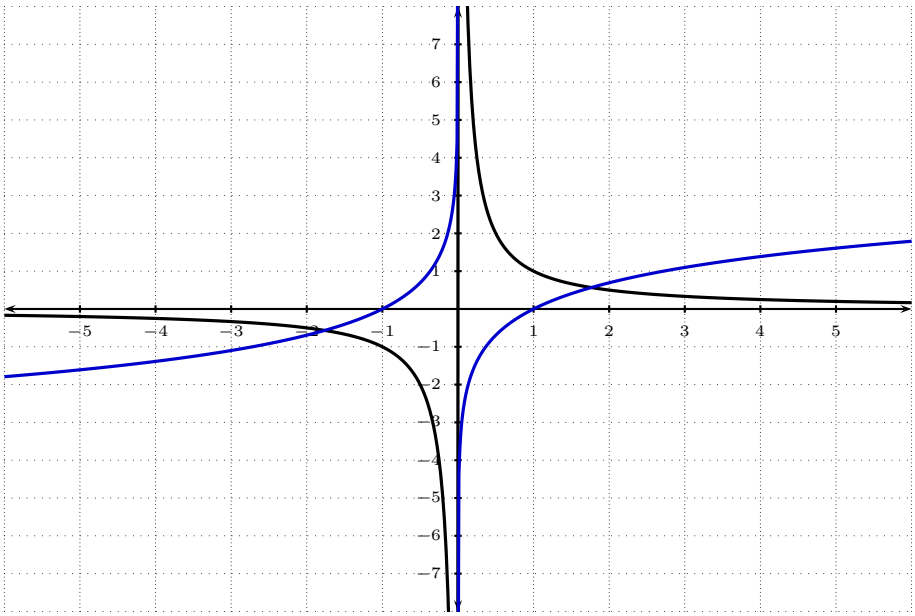
```
\graphs{%
\addgraph f(x)=\pssin x..
\graphcolor{red}
\addrgraph f(x)=x sqrt,x=0..6..
\psset{linestyle=dashed}\graphcolor{blue}
\addgraph f(x)=\pscos x..
},x=-6..6,y=-1.1..2.5(x=1cm,y=1cm)
```



\addgraph f(x)=⟨*expr*⟩.. In the first argument of a `\graphs` function, draws a curve on the whole interval. This function cannot work outside of a `\graphs` function.

\addrgraph f(x)=⟨*expr*⟩,x=⟨*xmin*⟩..⟨*xmax*⟩ Works the same way `\addgraph` does, except that you need to tell it on which interval the curve must be drawn. This is useful when you want to draw curves defined by several functions or not defined for some values of x. Although nothing prevents you from doing so[(3)], you shouldn't use it outside `\graphs`. If you want to draw a "naked" curve, there are other ways.

```
\graphs{
\addgraph f(x)=1 x div..
\graphcolor{ink}
\addrgraph f(x)=x abs ln,x=0.00001..6..
\addrgraph f(x)=x neg ln neg,x=-6..-.00001..
},x=-6..6,y=-8..8(x=1cm,y=.5cm)
```

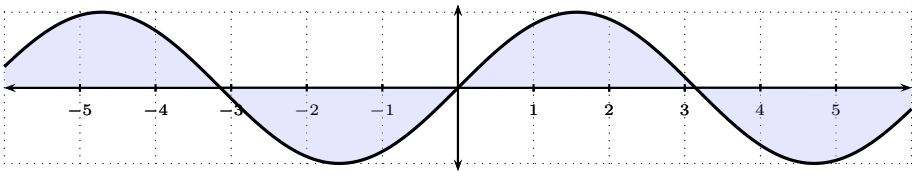


Now, if you are gifted with some common (mathematical) sense, you will frown and wonder : $x \longmapsto \frac{1}{x}$ is undefined for $x = 0$. How come this code does not cause a cataclysmic crash ? The answer is : it *should*. But `\graphpoints` is set to the default value of 500. This number is even. As the points used to draw the graph are equally spaced in respect to the x-axis, none of them is such as $x = 0$. Therefore no attempt is made to compute $1/0$, which is fortunate. However, should `\graphpoint` become odd, the same code would yield a dreadful PostScript error. Then there would be two ways to bypass this : either change `\graphpoint` (locally) to an even value, or use `\addrgraph` the way I did for ln. The second way is obviously far more satisfying.

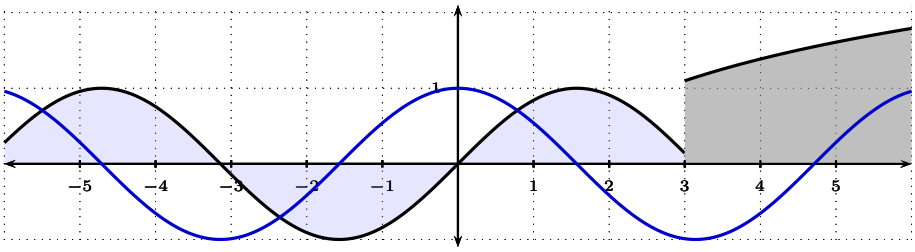
\addgraphshade f(x)=⟨*expr*⟩.., \addrgraphshade f(x)=⟨*expr*⟩.. Similar to `\addgraph` and `\addgraphshade`, but a shade is added between the curve and the x-axis.

```
\graphs{ \addgraphshade f(x)=\pssin x..
},x=-6..6,y=-1.1..1.1(x=1cm,y=1cm)
```

(3) As a matter of fact, this function is nothing more than the `psplot` function.




```
\graphs{
    \addrgraphshade f(x)=\pssin x,x=-6..3..
    \graphshadecolor{lightgray}\addrgraphshade f(x)=x ln,x=3..6..
    \graphcolor{ink}\addgraph f(x)=\pscos x..
    },x=-6..6,y=-1.1..2.1(x=1cm,y=1cm)
```



For technical reasons, if you want your ordinary curves to cohabit peacefully with the shaded ones, the latter should appear first in the graphs list. Otherwise the filling of the shade might erase part of the ordinary curves.

verticalline ⟨***x***⟩ Draws a vertical line

regraphaxes If, for any reason, you want the axes and the grid to be redrawn, for instance if you find that your curves have erased too much of them, this command, when put at the very end of the graphs list, will make them reappear.

6.2 Drawing functions, standard syntax

Not yet available.

6.3 Additional functions

graphpoint [***display ?***] ⟨***x***⟩ ⟨***y***⟩ ⟨***f(x)***⟩ ⟨***caption x***⟩ ⟨***caption y***⟩ ⟨***caption f(x)***⟩ What it does is pretty obvious. The optional argument can take the values h, v, $hv = vh$, *none*. The default value is hv, meaning that both the vertical and the horizontal lines are drawn.

```
{\def\graphticks{0}
\graphs{
\addgraph f(x)=0.0869 x mul x mul x mul 0.4571 x mul
               x mul sub 1.401 x mul sub 2.772 add..
{\color{linkcolor}
    \graphpoint 11{f(x)}{x}{y}
    \graphpoint {-2.4}{2.3}{f(x)}{x}{y}
    \graphpoint {-4}{-4.5}{f(x)}{x}{y}
    \graphpoint {3}{-3.2}{f(x)}{x}{y}
    \graphpoint 00{0}{}{}}
},x=-5..6,y=-5..4(x=1cm,y=.7cm)}
```

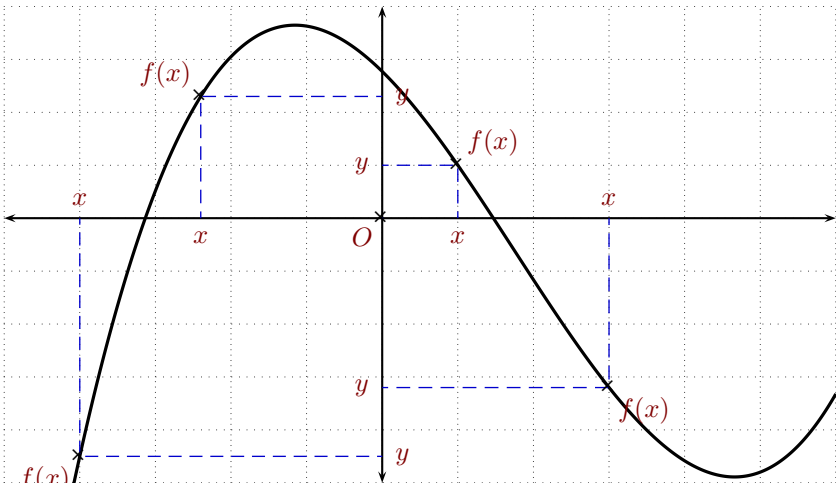


Figure 1: Curve of $f(x) = \frac{1117}{12852}x^3 - \frac{16}{35}x^2 - \frac{90047}{64260}x + \frac{29683}{10710}$

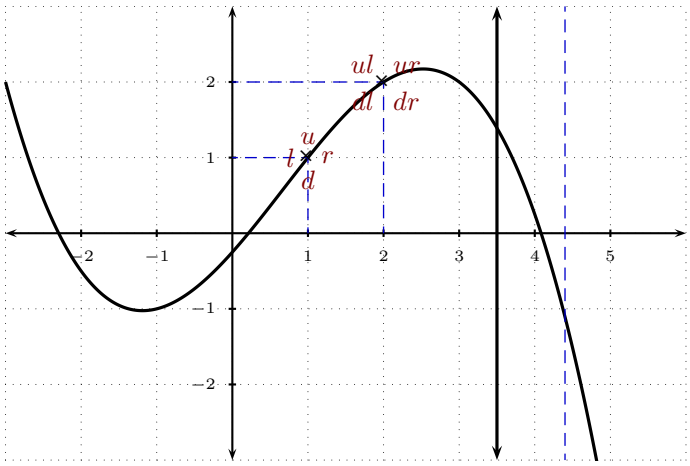
Undoubtedly, you will have noticed that the captions are always placed at the *right* position, which depends upon the coordinates of the point. [(4)]

\graphvert [<-> *etc*] ⟨***x***⟩ Draws a vertical line in standard style. If an optional argument is given, it must be a valid pstricks arrowhead symbol, such as `<-> -> - -o` etc. Also, the line will be influenced by the surrounding pstricks parameters pertaining to lines color, style and so on if and only if an optional argument is given.

\graphlabel ⟨***caption (math)***⟩ ⟨***position***⟩ ⟨***x***⟩ ⟨***y***⟩ Puts a label close to a point. The position argument can be u, d, r, l, or ur, ul, dl, dr. They mean up, down, right, left, up and right, etc...

```
\graphs{
    \addgraph f(x)=x x mul x mul 1 8 div neg mul 1 4 div x x mul
                mul add 9 8 div x mul add 1 4 div sub..
    {\graphcolor{black}
    \graphpoint 11{}{}{}
    \graphpoint 22{}{}{}
    \graphvert{4.4}
    \graphvert[<->]{3.5}
    }\color{linkcolor}
    \graphlabel uu11
    \graphlabel rr11
    \graphlabel dd11
    \graphlabel ll11
    \graphlabel {ur}{ur}22
    \graphlabel {ul}{ul}22
    \graphlabel {dr}{dr}22
    \graphlabel {dl}{dl}22
    },x=-3..6,y=-3..3(x=1cm,y=1cm)
```

(4) Just as a little anecdote, you may wonder why I chose such a strange function for this graph. Well, I didn't : I placed the four points on the graph and then I built the polynomial which would incorporate them, by means of Lagrange's interpolation formula.



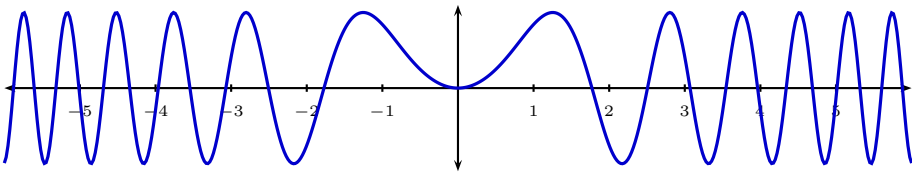
6.4 Parameters for graphs

These functions (actually most of them are constants) change the overall behaviour of the drawing functions. Be careful when you use them, for they have effect on everything that follows them. To apply them locally, you must enclose them and the graphs they should influence between braces. The default values appear in bold in the following descriptions.

graphgrid :

0 $\Longrightarrow$ The grid is not displayed
1 $\Longrightarrow$ The grid is displayed.

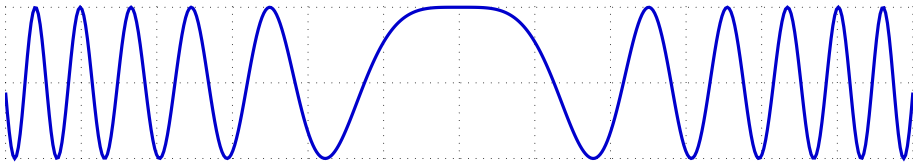
```
{\def\graphgrid{0}
\graph f(x)=\pssin{x x mul},x=-6..6,y=-1.1..1.1(x=1cm,y=1cm)}
```



graphaxes :

0 $\Longrightarrow$ The axes are not displayed
1 $\Longrightarrow$ The axes are displayed.

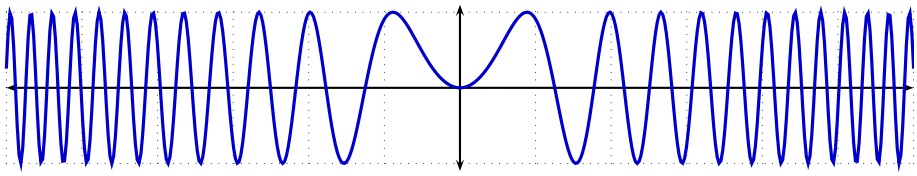
```
{\def\graphaxes{0}
\graph f(x)=\pssin{x x mul},x=-6..6,y=-1.1..1.1(x=1cm,y=1cm)}
```



graphticks :

0 $\Longrightarrow$ The ticks are not displayed but the axes remain
1 $\Longrightarrow$ The ticks are displayed **if** there are axes to put them on.

```
{\def\graphticks{0}
\graph f(x)=\pssin{2 x mul x mul},x=-6..6,y=-1.1..1.1(x=1cm,y=1cm)}
```

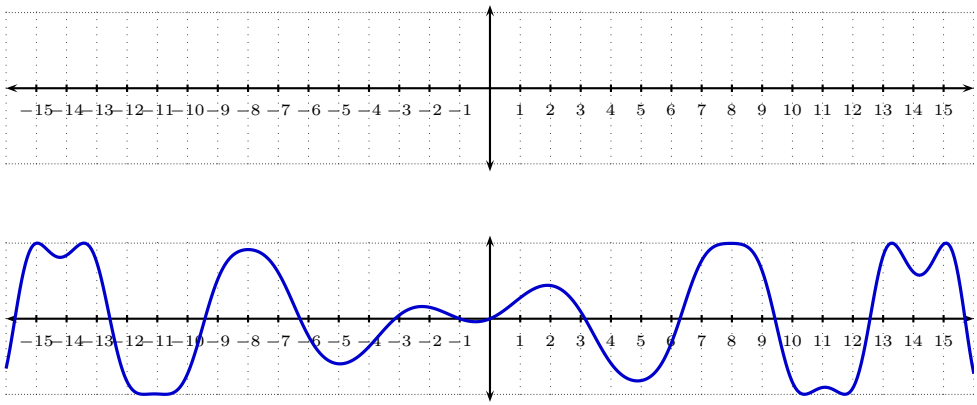


\graphplot :

0 $\Longrightarrow$ The plot is not displayed

1 $\Longrightarrow$ The plot is displayed.

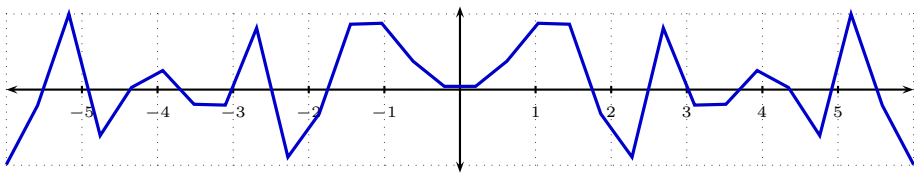
```
{\def\graphplot{0}
\graph f(x)=\pssin{x 6 div 1 6 div add \pssin x mul}
,x=-16..16,y=-1.1..1.1(x=.4cm,y=1cm)} {\def\graphplot{1}
\graph f(x)=\pssin{x 6 div 1 6 div add \pssin x mul}
,x=-16..16,y=-1.1..1.1(x=.4cm,y=1cm)}
```



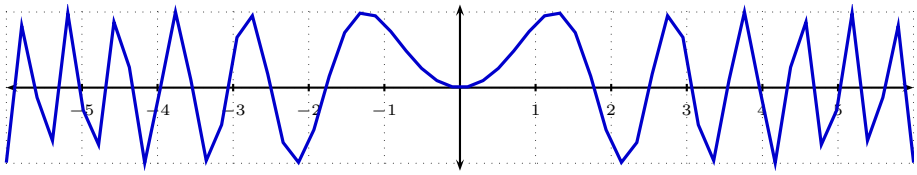
\graphpoints :

Indicates how many points are computed in order to draw the curve. The higher this value, the better the quality. However, the building time of the resulting files, as well as their size, are increased with this value. The default is 500 and should not be changed.

```
{\def\graphpoints{30}
\graph f(x)=\pssin{x x mul},x=-6..6,y=-1.1..1.1(x=1cm,y=1cm)}
```



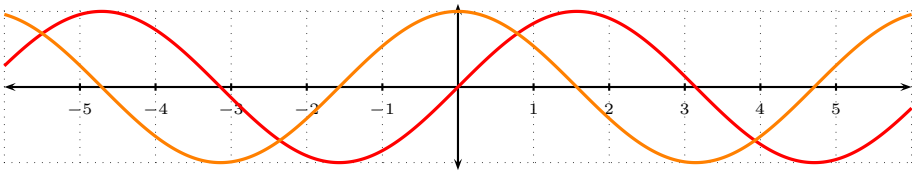

```
{\def\graphpoints{60}
\graph f(x)=\pssin{x x mul},x=-6..6,y=-1.1..1.1(x=1cm,y=1cm)}
```



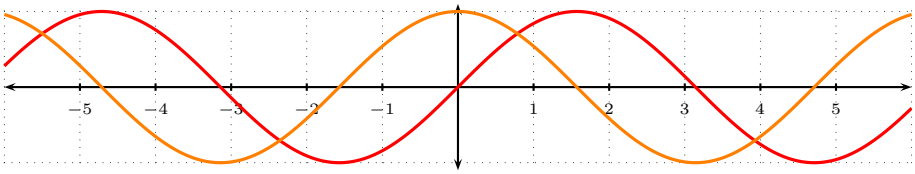
\graphcolor ⟨***new default color***⟩ :
Changes the color of the curves to come. For instance, I put `\graphcolor{ink}` at the beginning of this section, and `\graphcolor{black}` at its end. *ink* is a color defined by the `SKnife` package.

Do not use this instruction as the first one of a `\graphs` functions lest it yields a bug which creates an unwanted offset :

```
\graphs{\graphcolor{red}\addgraph f(x)=\pssin x..
\graphcolor{orange}\addgraph f(x)=\pscos x..
},x=-6..6,y=-1.1..1.1(x=1cm,y=1cm)
```




```
{\graphcolor{red}\graphs{\addgraph f(x)=\pssin x..
\graphcolor{orange}\addgraph f(x)=\pscos x..
},x=-6..6,y=-1.1..1.1(x=1cm,y=1cm)}
```

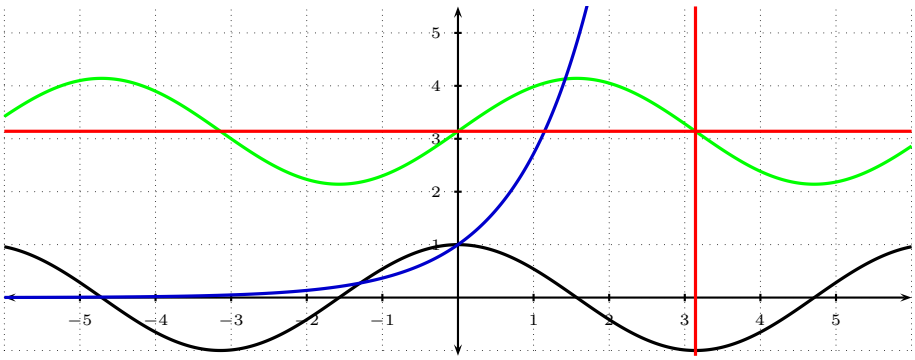


\graphshadecolor ⟨***color***⟩ Works the same way `\graphcolor` does, except that, exactly as the name tells, it is the default color of the shades that is changed and not that of the curves. Please note that the *curves* drawn by the shading function aren't influenced in any extent by this function.

6.5 Special functions dedicated to graphs

\pssin ⟨***angle (radian)***⟩, **\pscos, \psexp, \pspi** These functions stand for the PostScript code of sin, cos, exp and $x \longmapsto \pi$. Beware : their syntax is not that of PostScript, but their own arguments as well as everything surrounding them must be. An empty argument **must** be passed to `\pspi`.

```
\graphs{\addgraph f(x)=\pscos x..
\graphcolor{green}\addgraph f(x)=\pssin x \pspi{} add..
\graphcolor{ink}\addgraph f(x)=\psexp x..
\graphcolor{red}\addgraph f(x)=\pspi{}..
\verticalline{\pspi{}}
},x=-6..6,y=-1.1..5.5(x=1cm,y=.7cm)
```



6.6 Graphs, PostScript and PsTricks

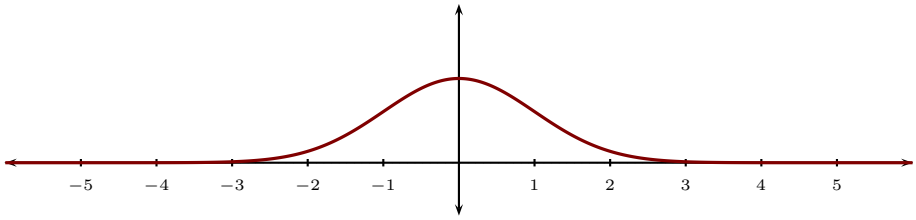
6.6.1 Quick glance to PostScript

This most specialized programming language is stack-based. Here are some operators you can use for graphs:

a dup $\Longrightarrow$	$a\ a$ Duplicates the element on top of the stack.
$a\ b$ add $\Longrightarrow$	$a+b$; standard addition.
$a\ b$ sub $\Longrightarrow$	$a-b$; standard substraction.
$a\ b$ mul $\Longrightarrow$	$a\times b$; standard multiplication.
$a\ b$ div $\Longrightarrow$	a/b ; standard division.
a neg $\Longrightarrow$	$-a$; negative.
a ceiling $\Longrightarrow$	$\lceil a\rceil$.
a floor $\Longrightarrow$	$\lfloor a\rfloor$.
a round $\Longrightarrow$	nearest integer to a.
a truncate $\Longrightarrow$	$\begin{cases}\lceil a\rceil & \text{if } x<0\\ \lfloor a\rfloor & \text{if } x\geqslant 0\end{cases}$
θ cos $\Longrightarrow$	$\cos\left(\frac{\pi}{180}\theta\right)$. Use the special functions instead.
θ sin $\Longrightarrow$	$\sin\left(\frac{\pi}{180}\theta\right)$. Use the special functions instead.
$a\ x$ $\Longrightarrow$	a^x. for $x\longmapsto\exp(x)$, use the special function.
$a\geqslant 0$ sqrt $\Longrightarrow$	$\sqrt{a}$.
$a>0$ ln $\Longrightarrow$	$\ln a$.
$a>0$ log $\Longrightarrow$	$\log a$. (base 10)

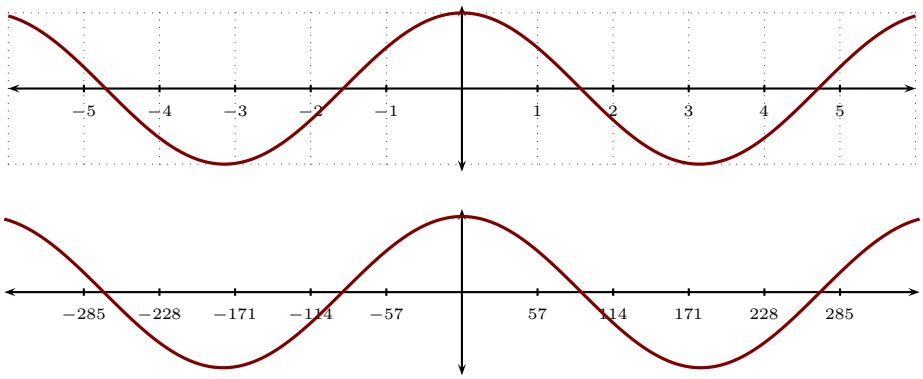
Therefore, the function $f:x\longmapsto\frac{1}{2\pi}e^{-\frac{t^2}{2}}$ would become :

```
\graph f(x)=1 2 \pspi{} mul div \psexp{x x mul 2 div neg} mul%
,x=-6..6,y=-.1...3(x=1cm,y=7cm)
```



And here is why the PostScript cos and sin functions are not immediately suitable for mathematics :

```
\graph f(x)=\pscos x,x=-6..6,y=-1.1..1.1(x=1cm,y=1cm)
{\def\graphgrid{0}\psset{Dx=57}
\graph f(x)=x cos,x=-345..345,y=-1.1..1.1(x=0.01754cm,y=1cm)}
```



Who on earth uses degrees, anyway ?

Work in Progress

Contents