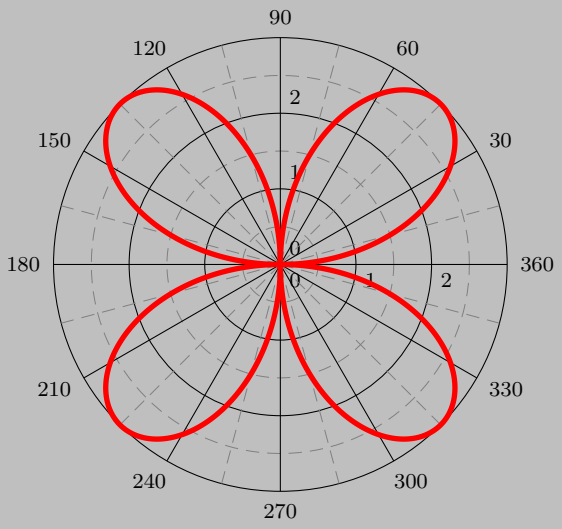
PSTricks

pst-plot
plotting data and functions

v.1.69

August 17, 2014



Documentation by
Herbert Voß

Package author(s):
Timothy Van Zandt
Herbert Voß

This version of pst-plot uses the extended keyval handling of pst-xkey and has a lot of the macros which were recently in the package pstricks-add. This documentation describes only the new and changed stuff. For the default behaviour look into the documentation part of the base pstricks package. You find the documentation here: http://mirrors.ctan.org/graphics/pstricks/base/doc/.

Thanks to: Guillaume van Baalen; Stefano Baroni; Martin Chicoine; Gerry Coombes; Ulrich Dirr; Christophe Fourey; Hubert Gäßlein; Jürgen Gilg; Denis Girou; Peter Hutnick; Christophe Jorssen; Uwe Kern; Alexander Kornrumpf; Manuel Luque; Patrice Mégret; Jens-Uwe Morawski; Tobias Nähring; Rolf Niepraschk; Martin Paech; Alan Ristow; Christine Römer; Arnaud Schmittbuhl

Contents

Part I.
Basic commands, connections and labels

1. Introduction

The plotting commands described in this part are defined in the very first version of `pst-plot.tex` and available for all new and ancient versions.

The `\psdots`, `\psline`, `\pspolygon`, `\pscurve`, `\psecurve` and `\psccurve` graphics objects let you plot data in a variety of ways. However, first you have to generate the data and enter it as coordinate pairs x,y. The plotting macros in this section give you other ways to get and use the data.

To parameter `plotstyle=style` determines what kind of plot you get. Valid styles are `dots`, `line`, `polygon`, `curve`, `ecurve`, `ccurve`. E ,g., if the `plotstyle` is `polygon`, then the macro becomes a variant of the `\pspolygon` object.

You can use arrows with the plot styles that are open curves, but there is no optional argument for specifying the arrows. You have to use the `arrows` parameter instead.

No PostScript error checking is provided for the data arguments. There are system-dependent limits on the amount of data TEX and PostScript can handle. You are much less likely to exceed the PostScript limits when you use the `line`, `polygon` or `dots` plot style, with `showpoints=false`, `linearc=0pt`, and no arrows.

Note that the lists of data generated or used by the plot commands cannot contain units. The values of `\psxunit` and `\psyunit` are used as the unit.

2. Plotting data records

```
\fileplot [Options] {file}
\psfileplot [Options] {file}
\dataplot [Options] {\⟨macro⟩}
\psdataplot [Options] {\⟨macro⟩}
\savedata{\⟨macro⟩}[data]
\readdata{\⟨macro⟩}{file}
\psreadColumnData{colNo}{delimiter}{\⟨macro⟩}{filename}
\listplot{data}
\pslistplot{data}
```

The macros with a preceeding `ps` are equivalent to those without.

`\fileplot` is the simplest of the plotting functions to use. You just need a file that contains a list of coordinates (without units), such as generated by Mathematica or other mathematical packages. The data can be delimited by curly braces { }, parentheses ( ), commas, and/or white space. Bracketing all the data with square brackets [ ] will significantly speed up the rate at which the data is read, but there are system-dependent

limits on how much data TEX can read like this in one chunk. (The [*must* go at the beginning of a line.) The file should not contain anything else (not even \endinput), except for comments marked with %.

\fileplot only recognizes the line, polygon and dots plot styles, and it ignores the arrows, linearc and showpoints parameters. The \listplot command, described below, can also plot data from file, without these restrictions and with faster TEX processing. However, you are less likely to exceed PostScript's memory or operand stack limits with \fileplot.

If you find that it takes TEX a long time to process your \fileplot command, you may want to use the \PSTtoEPS command described on page **??**. This will also reduce TEX's memory requirements.

\dataplot is also for plotting lists of data generated by other programs, but you first have to retrieve the data with one of the following commands: *data* or the data in *file* should conform to the rules described above for the data in \fileplot (with \savedata, the data must be delimited by [ ], and with \readdata, bracketing the data with [ ] speeds things up). You can concatenate and reuse lists, as in

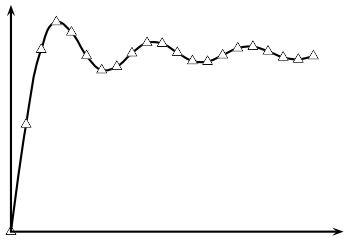
```
\readdata{\foo}{foo.data}
\readdata{\bar}{bar.data}
\dataplot{\foo\bar}
\dataplot[origin={0,1}]{\bar}
```

The \readdata and \dataplot combination is faster than \fileplot if you reuse the data. \fileplot uses less of TEX's memory than \readdata and \dataplot if you are also use \PSTtoEPS.

Here is a plot of $\int \sin(x)\mathrm{d}x$. The data was generated by Mathematica, with

```
Table[{x,N[SinIntegral[x]]},{x,0,20}]
```

and then copied to this document.



```
\pspicture(4,3) \psset{xunit=.2cm,yunit=1.5cm}
  \savedata{\mydata}[
    {{0, 0}, {1., 0.946083}, {2., 1.60541}, {3., 1.84865}, {4., 1.7582},
    {5., 1.54993}, {6., 1.42469}, {7., 1.4546}, {8., 1.57419},
    {9., 1.66504}, {10., 1.65835}, {11., 1.57831}, {12., 1.50497},
    {13., 1.49936}, {14., 1.55621}, {15., 1.61819}, {16., 1.6313},
    {17., 1.59014}, {18., 1.53661}, {19., 1.51863}, {20., 1.54824}}]
  \dataplot[plotstyle=curve,showpoints,dotstyle=triangle]{\mydata}
  \psline{<->}(0,2)(0,0)(22,0)
\endpspicture
```

\listplot is yet another way of plotting lists of data. This time, <list> should be a list of data (coordinate pairs), delimited only by white space. *list* is first expanded by

TEX and then by PostScript. This means that *list* might be a PostScript program that leaves on the stack a list of data, but you can also include data that has been retrieved with `\readdata` and `\dataplot`. However, when using the `line`, `polygon` or `dots` plotstyles with `showpoints=false`, `linearc=0pt` and no arrows, `\dataplot` is much less likely than `\listplot` to exceed PostScript's memory or stack limits. In the preceding example, these restrictions were not satisfied, and so the example is equivalent to when `\listplot` is used:

```
 ...
\listplot[plotstyle=curve,showpoints=true,dotstyle=triangle]{\mydata}
 ...
```

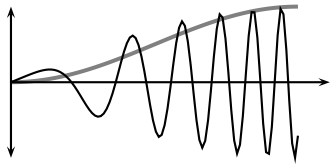
3. Plotting mathematical functions

`\psplot` `[Options]` {$x_!\min$@}{$x_!\max$@}{*function*}
`\parametricplot` `[Options]` {$t_!\min$@}{$t_!\max$@}{*x(t) y(t)*}

`\psplot` can be used to plot a function $f(x)$, if you know a little PostScript. *function* should be the PostScript or algebraic code for calculating $f(x)$. Note that you must use x as the dependent variable.

```
 \psplot[plotpoints=200]{0}{720}{x sin}
```

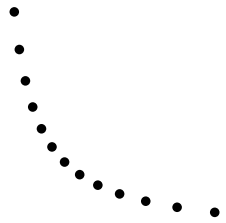
plots $\sin(x)$ from 0 to 720 degrees, by calculating $\sin(x)$ roughly every 3.6 degrees and then connecting the points with `\psline`. Here are plots of $\sin(x)\cos((x/2)^2)$ and $\sin^2(x)$:



```
\pspicture(0,-1)(4,1)
\psset{xunit=1.2pt}
\psplot[linecolor=gray,linewidth=1.5pt,plotstyle=curve]{0}{90}{x sin dup mul}
\psplot[plotpoints=100]{0}{90}{x sin x 2 div 2 exp cos mul}
\psline{<->}(0,-1)(0,1) \psline{->}(100,0)
\endpspicture
```

`\parametricplot` is for a parametric plot of $(x(t), y(t))$. *function* is the PostScript code or algebraic expression for calculating the pair $x(t)$ $y(t)$.

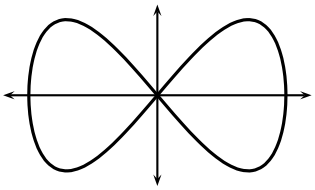
For example,



```
\pspicture(3,3)
\parametricplot[plotstyle=dots,plotpoints=13]%
   {-6}{6}{1.2 t exp 1.2 t neg exp}
\endpspicture
```

plots 13 points from the hyperbola $xy = 1$, starting with $(1.2^{-6}, 1.2^{6})$ and ending with $(1.2^{6}, 1.2^{-6})$.

Here is a parametric plot of $(\sin(t), \sin(2t))$:



```
\pspicture(-2,-1)(2,1)
  \psset{xunit=1.7cm}
  \parametricplot[linewidth=1.2pt,plotstyle=ccurve]%
    {0}{360}{t sin t 2 mul sin}
  \psline{<->}(0,-1.2)(0,1.2)
  \psline{<->}(-1.2,0)(1.2,0)
\endpspicture
```

The number of points that the \psplot and \parametricplot commands calculate is set by the plotpoints=<value> parameter. Using "curve" or its variants instead of "line" and increasing the value of plotpoints are two ways to get a smoother curve. Both ways increase the imaging time. Which is better depends on the complexity of the computation. (Note that all PostScript lines are ultimately rendered as a series (perhaps short) line segments.) Mathematica generally uses "lineto" to connect the points in its plots. The default minimum number of plot points for Mathematica is 25, but unlike \psplot and \parametricplot, Mathematica increases the sampling frequency on sections of the curve with greater fluctuation.

Part II.
New commands

4. Extended syntax for \psplot, \psparametricplot, and \psaxes

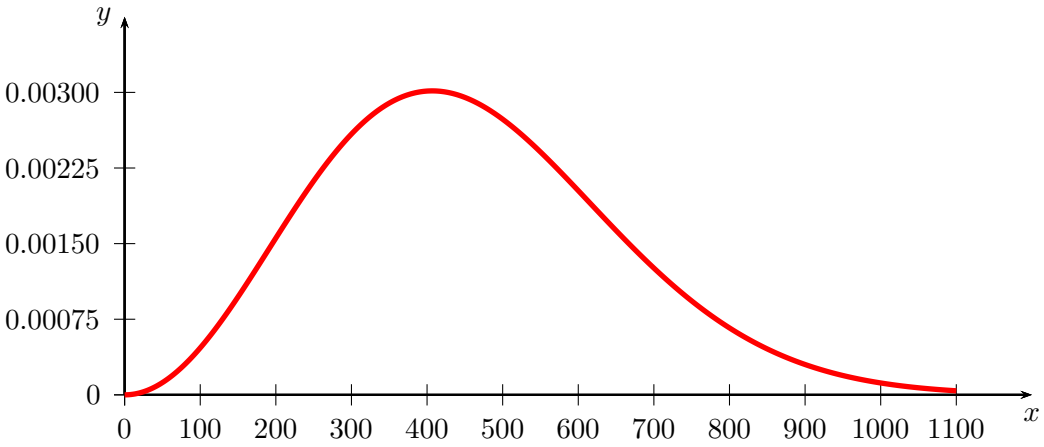
There is now a new optional argument for \psplot and \psparametricplot to pass additional PostScriptcommands into the code. This makes the use of \pstVerb in most cases superfluous.

```
\psplot [Options] {x0}{x1} [PS commands] {function}
\psparametricplot [Options] {t0}{t1} [PS commands] {x(t) y(t)}
\psaxes [Options] {arrows} (x0,y0)(x1,y1)(x2,y2) [Xlabel,Xangle] [Ylabel,Yangle]
```

The macro \psaxes has now four optional arguments, one for the setting, one for the arrows, one for the x-label and one for the y-label. If you want only a y-label, then leave the x one empty. A missing y-label is possible. The following examples show how it can be used.

```
\begin{pspicture}(-1,-0.5)(12,5)
  \psaxes[Dx=100,dx=1,Dy=0.00075,dy=1]{->}(0,0)(12,5)[$x$,-90][$y$,180]
  \psplot[linecolor=red, plotstyle=curve,linewidth=2pt,plotpoints=200]{0}{11}%
    [ /const1 3.3 10 8 neg exp mul def
```

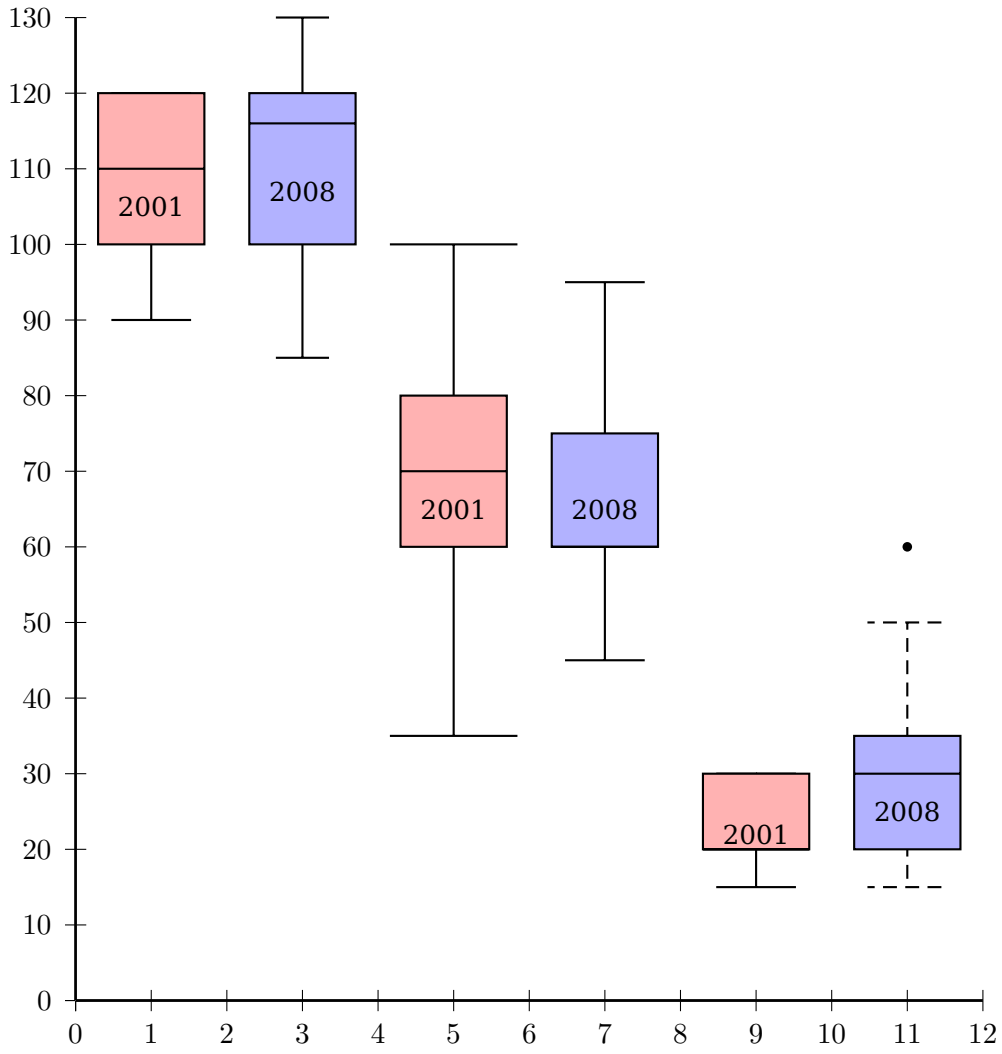
```
    /s 10 def
    /const2 6.04 10 6 neg exp mul def ] % optional PS commands
  { const1 x 100 mul dup mul mul Euler const2 neg x 100 mul dup mul mul exp
      mul 2000 mul}
\end{pspicture}
```



5. New Macro \psBoxplot

A box-and-whisker plot (often called simply a box plot) is a histogram-like method of displaying data, invented by John. Tukey. The box-and-whisker plot is a box with ends at the quartiles Q_1 and Q_3 and has a statistical median M as a horizontal line in the box. The "'whiskers"* are lines to the farthest points that are not outliers (i.e., that are within 3/2 times the interquartile range of Q_1 and Q_3). Then, for every point more than 3/2 times the interquartile range from the end of a box, is a dot.

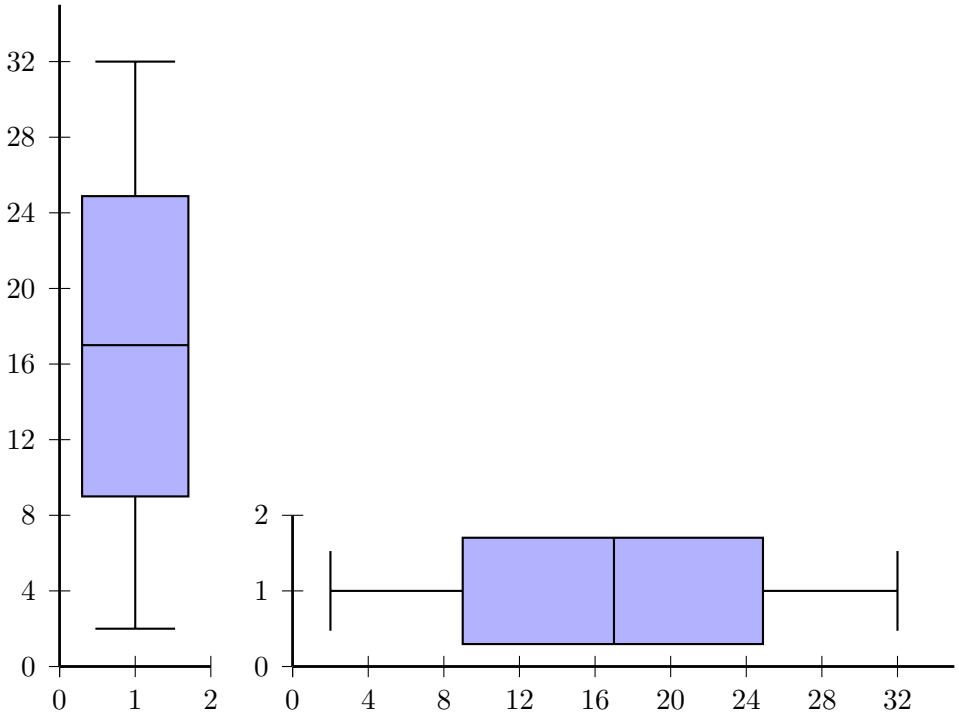
The only special optional arguments, beside all other which are valid for drawing lines and filling areas, are `IQLfactor`, `barwidth`, and `arrowlength`, where the latter is a factor which is multiplied with the barwidth for the line ends. The `IQLfactor`, preset to 1.5, defines the area for the outliners. The outliners are plotted as a dot and take the settings for such a dot into account, eg. `dotstyle`, `dotsize`, `dotscale`, and `fillcolor`. The default is the black dot.




```
\begin{pspicture}(-1,-1)(12,14)
```

```
\psset{yunit=0.1,fillstyle=solid}
\savedata{\data}[100 90 120 115 120 110 100 110 100 90 100 100 120 120 120]
\rput(1,0){\psBoxplot[fillcolor=red!30]{\data}}
\rput(1,105){2001}
\savedata{\data}[90 120 115 116 115 110 90 130 120 120 120 85 100 130 130]
\rput(3,0){\psBoxplot[arrowlength=0.5,fillcolor=blue!30]{\data}}
\rput(3,107){2008}
\savedata{\data}[35 70 90 60 100 60 60 80 80 60 50 55 90 70 70]
\rput(5,0){\psBoxplot[barwidth=40pt,arrowlength=1.2,fillcolor=red!30]{\data}}
\rput(5,65){2001}
\savedata{\data}[60 65 60 75 75 60 50 90 95 60 65 45 45 60 90]
\rput(7,0){\psBoxplot[barwidth=40pt,fillcolor=blue!30]{\data}}
\rput(7,65){2008}
\savedata{\data}[20 20 25 20 15 20 20 25 30 20 20 20 30 30 30]
\rput(9,0){\psBoxplot[fillcolor=red!30]{\data}}
\rput(9,22){2001}
\savedata{\data}[20 30 20 35 35 20 20 60 50 20 35 15 30 20 40]
\rput(11,0){\psBoxplot[fillcolor=blue!30,linestyle=dashed]{\data}}
\rput(11,25){2008}
\psaxes[dy=1cm,Dy=10](0,0)(12,130)
\end{pspicture}
```

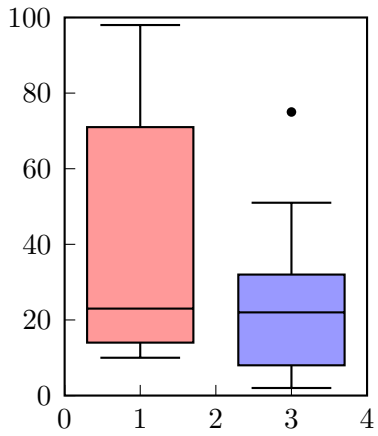
The next example uses an external file for the data, which must first be read by the macro `\readdata`. The next one creates a horizontal boxplot by rotating the output with -90 degrees.




```
\readdata{\data}{boxplot.data}
```

```
\begin{pspicture}(-1,-1)(2,10)
\psset{yunit=0.25,fillstyle=solid}
\savedata{\data}[2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32]
\rput(1,0){\psBoxplot[fillcolor=blue!30]{\data}}
\psaxes[dy=1cm,Dy=4](0,0)(2,35)
\end{pspicture}
%
\begin{pspicture}(-1,-1)(11,2)
\psset{xunit=0.25,fillstyle=solid}
\savedata{\data}[2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32]
\rput{-90}(0,1){\psBoxplot[yunit=0.25,fillcolor=blue!30]{\data}}
\psaxes[dx=1cm,Dx=4](0,0)(35,2)
\end{pspicture}
```

It is also possible to read a data column from an external file:

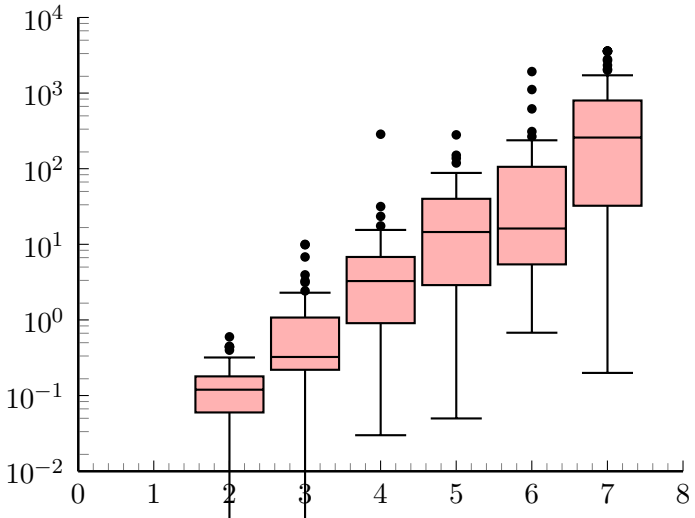



```
\begin{filecontents*}{Data.dat}
98, 32
20, 11
79, 26
14, 9
23, 22
21, 10
58, 25
13, 8
19, 5
53, 29
41, 37
11, 2
83, 25
71, 51
10, 7
89, 17
10, 6
  , 41
  , 75
```

```
\end{filecontents*}

\begin{pspicture}(-1,-1)(5,6)
\psaxes[axesstyle=frame,dy=1cm,Dy=20,ticksize=4pt 0](0,0)(4,5)
\psreadDataColumn{1}{,}{\data}{Data.dat}
\rput(1,0){\psBoxplot[fillcolor=red!40,yunit=0.05]{\data}}
\psreadDataColumn{2}{,}{\data}{Data.dat}
\rput(3,0){\psBoxplot[fillcolor=blue!40,yunit=0.05]{\data}}
\end{pspicture}
```

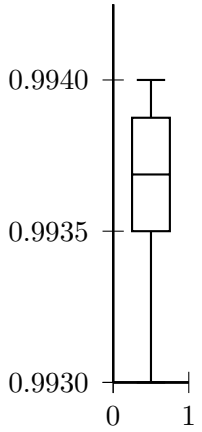
With the optional argument `postAction` one can modify the y value of the boxplot, e.g. for an output with a vertical axis in logarithm scaling:




```
\begin{pspicture}(-1,-3)(9,5)
\psset{fillstyle=solid}
\psaxes[ylogBase=10,Oy=-2,logLines=y,ticksize=0 4pt, subticks=5](1,-2)(9,4)
\rput(3,0){\psBoxplot[fillcolor=red!30,barwidth=0.9cm,postAction=Log]{
 0.09 0.44 0.12 0.06 0.32 0.23 0.44 0.02 0.15 0.18 0 0.29 0 0.11 0.26 0.11
 0 0.45 0.04 0.14 0.03 0.12 0.14 0.31 0.06 0.06 0.11 0.12 0.12 0.12 0.13
 0.01 0.40 0.01 0.03 0.17 0 0.10 0.15 0.16 0.06 0.10 0.01 0.60 0.26 0.11
 0.15 0.22 0.14 0.01 }}
 \rput(4,0){\psBoxplot[fillcolor=red!30,barwidth=0.9cm,postAction=Log]{
0.07 0.49 0.34 0.20 0.02 1.08 6.83 0.31 0.54 0.02 0.29 0.18 0.60 0.09 0.61
1.37 0.26 0.03 2.30 0.09 3.15 0.13 0.29 0.27 1.30 0.73 0.63 0.24 10.03 0
0.26 0.18 3.29 2.43 1.94 0.22 0.23 0.60 1.69 0.35 3.96 0.56 9.90 0.10 0.43
0.22 0.26 0.31 0.29 0.79 }}
 \rput(5,0){\psBoxplot[fillcolor=red!30,barwidth=0.9cm,postAction=Log]{
12.70 1.34 0.68 0.51 1.77 0.04 3.79 287.05 1.35 5.41 15.56 3.13 0.91 7.48
2.40 1.04 3.53 0.58 31.71 7.89 4.90 2.61 0.89 0.03 3.78 8.11 4.82 1.02 5.57
8.85 0.15 17.59 0.21 8.10 2.15 3.43 6.44 1.65 6.83 23.54 0.52 1.47 0.75
3.54 3.59 5.56 0.33 8.58 1.90 0.78 }}
 \rput(6,0){\psBoxplot[fillcolor=red!30,barwidth=0.9cm,postAction=Log]{
55.72 14.91 14.95 6.01 6.53 88.30 281.50 40.15 13.41 0.91 1.65 44.32 13.41
7.33 3.51 3.44 70.40 0.75 58.20 54.88 26.45 33.76 0.70 0.05 0.29 57.12
```

```
14.30 31.11 18.56 0.48 21.33 1.15 2.22 3.88 1.78 151.25 7.77 137.92 0.50
3.01 1.99 23.18 119.59 17.50 15.87 13.63 21.85 23.53 68.72 2.90 }}
 \rput(7,0){\psBoxplot[fillcolor=red!30,barwidth=0.9cm,postAction=Log]{
1.19 1.94 13.40 7.40 267.30 5.94 11.05 6.51 2.94 5.45 5.24 231 4.48 0.68
311.29 77.47 621.20 139.08 1933.59 2.52 100.96 11.02 153.43 26.67 83.84
4.31 106.34 15.90 1118.59 9.49 131.48 48.92 5.85 3.74 1.05 32.03 5.69
45.10 12.43 238.56 28.75 1.01 119.29 12.09 31.18 16.60 29.67 138.55
17.42 0.83 }}
 \rput(8,0){\psBoxplot[fillcolor=red!30,barwidth=0.9cm,postAction=Log]{
2077.45 762.10 469 143.60 685 3600 20.20 249.60 269 0.30 0.20 779.40 1.80
146.80 1.30 32.50 137 2016.40 2.30 33.90 801.60 2.20 646.90 3600 1184 627
500.50 238.30 477.40 3600 17.80 1726.80 2 316.70 174.50 2802.70 335.30
201.20 1.10 247.10 2705.10 156.90 5.10 2342.50 3600 3600 72.70 47.40
301.20 1.60 }}
\end{pspicture}
```

It uses the PostScript function `Log` instead of `log`. The latter cannot handle zero values. The next examples shows how a very small intervall can be handled:




```
\psset{yunit=0.5cm}
\begin{pspicture}(-2,-1)(2,11)
\savedata{\data}[0.9936 0.9937 0.9934 0.9936 0.9937 0.9938 0.9934 0.9933 0.9930
    0.9935]
\psaxes[Oy=0.9930,Dy=0.0005,dy=2cm](0,0)(1,10)
\rput(.5,0){\psBoxplot[barwidth=.5\psxunit,postAction=0.993 sub 1e4 mul]{\data
   }}
\end{pspicture}
```

6. The psgraph environment

This new environment psgraph does the scaling, it expects as parameter the values (without units!) for the coordinate system and the values of the physical width and height (with units!). The syntax is:

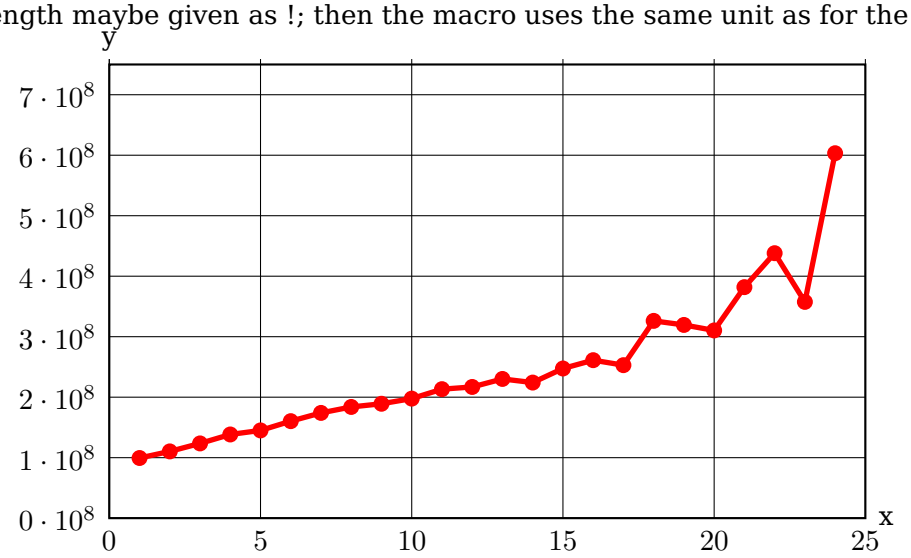
```
\psgraph [Options] {<arrows>}%
    (xOrig,yOrig)(xMin,yMin)(xMax,yMax){xLength}{yLength}
...
\endpsgraph

\begin{psgraph} [Options] {<arrows>}%
    (xOrig,yOrig)(xMin,yMin)(xMax,yMax){xLength}{yLength}
...
\end{psgraph}
```

where the options are valid **only** for the the \psaxes macro. The first two arguments have the usual PSTricks behaviour.

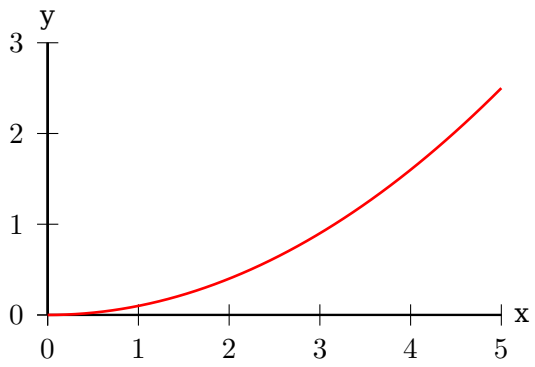
- if (xOrig,yOrig) is missing, it is substituted to (xMin,xMax);
- if (xOrig,yOrig) **and** (xMin,yMin) are missing, they are both substituted to (0,0).

The y-length maybe given as !; then the macro uses the same unit as for the x-axis.

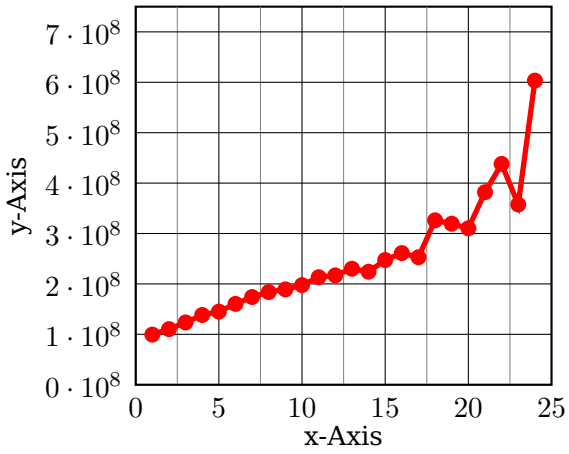



```
\readdata{\data}{demo1.data}
\pstScalePoints(1,1e-08){}{}% (x,y){additional x operator}{y op}
\psset{llx=-1cm,lly=-1cm}
\begin{psgraph}[axesstyle=frame,xticksize=0 7.59,yticksize=0 25,%
   subticks=0,ylabelFactor=\cdot 10^8,
   Dx=5,dy=1\psyunit,Dy=1](0,0)(25,7.5){10cm}{6cm} % parameters
  \listplot[linecolor=red,linewidth=2pt,showpoints=true]{\data}
\end{psgraph}
```

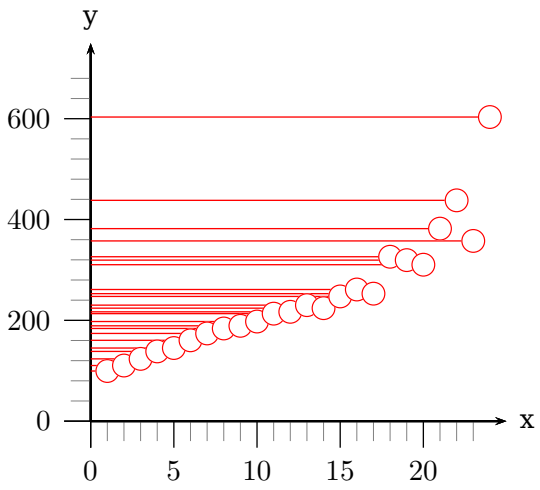
In the following example, the y unit gets the same value as the one for the x-axis.



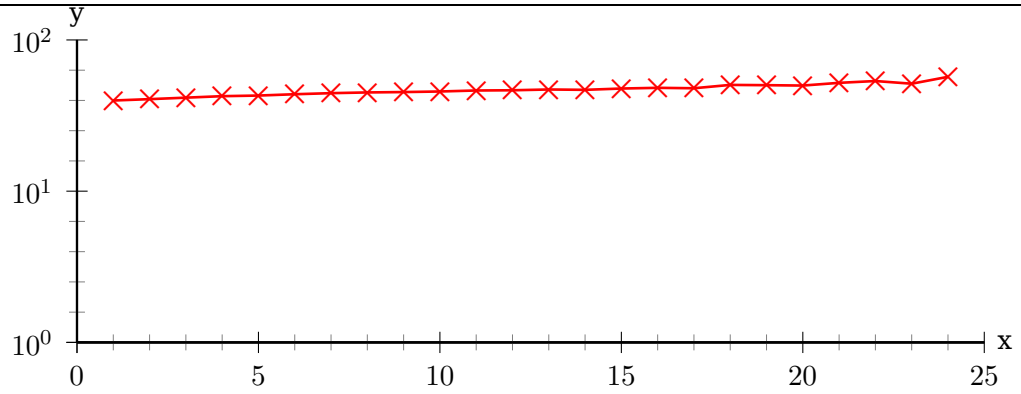

```
\psset{llx=-1cm,lly=-0.5cm,ury=0.5cm}
\begin{psgraph}(0,0)(5,3){6cm}{!} % x-y-axis with same unit
  \psplot[linecolor=red,linewidth=1pt]{0}{5}{x dup mul 10 div}
\end{psgraph}
```



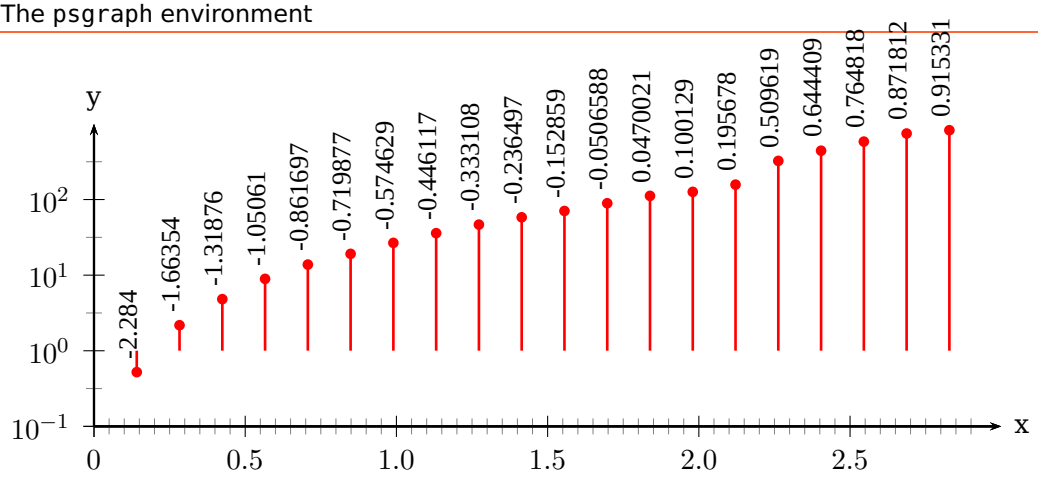

```
\readdata{\data}{demo1.data}
\psset{xAxisLabel=x-Axis,yAxisLabel=y-Axis,llx=-.5cm,lly=-1cm,ury=0.5cm,
  xAxisLabelPos={c,-1},yAxisLabelPos={-7,c}}
\pstScalePoints(1,0.00000001){}{}
\begin{psgraph}[axesstyle=frame,xticksize=0 7.5,yticksize=0 25,subticksize=1,
    ylabelFactor=\cdot 10^8,Dx=5,Dy=1,xsubticks=2](0,0)(25,7.5){5.5cm}{5cm}
 \listplot[linecolor=red, linewidth=2pt, showpoints=true]{\data}
\end{psgraph}
```



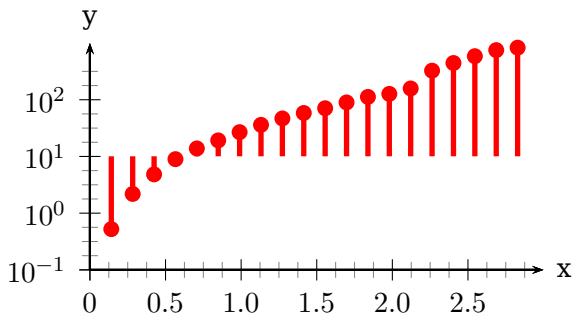

```
\readdata{\data}{demo1.data}
\psset{llx=-0.5cm,lly=-1cm}
\pstScalePoints(1,0.000001){}{}
\psgraph[arrows=->,Dx=5,dy=200\psyunit,Dy=200,subticks=5,ticksize=-10pt 0,
   tickwidth=0.5pt,subtickwidth=0.1pt](0,0)(25,750){5.5cm}{5cm}
\listplot[linecolor=red,linewidth=0.5pt,showpoints=true,dotscale=3,
 plotstyle=LineToYAxis,dotstyle=o]{\data}
\endpsgraph
```



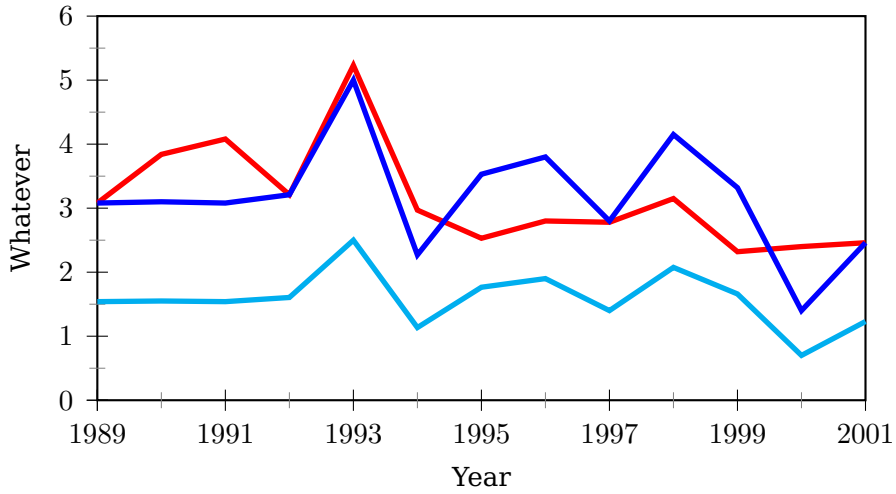

```
\readdata{\data}{demo1.data}
\pstScalePoints(1,0.2){}{log}
\psset{lly=-0.75cm}
\psgraph[ylogBase=10,Dx=5,Dy=1,subticks=5](0,0)(25,2){12cm}{4cm}
  \listplot[linecolor=red,linewidth=1pt,showpoints,dotstyle=x,dotscale=2]{\data
     }
\endpsgraph
```



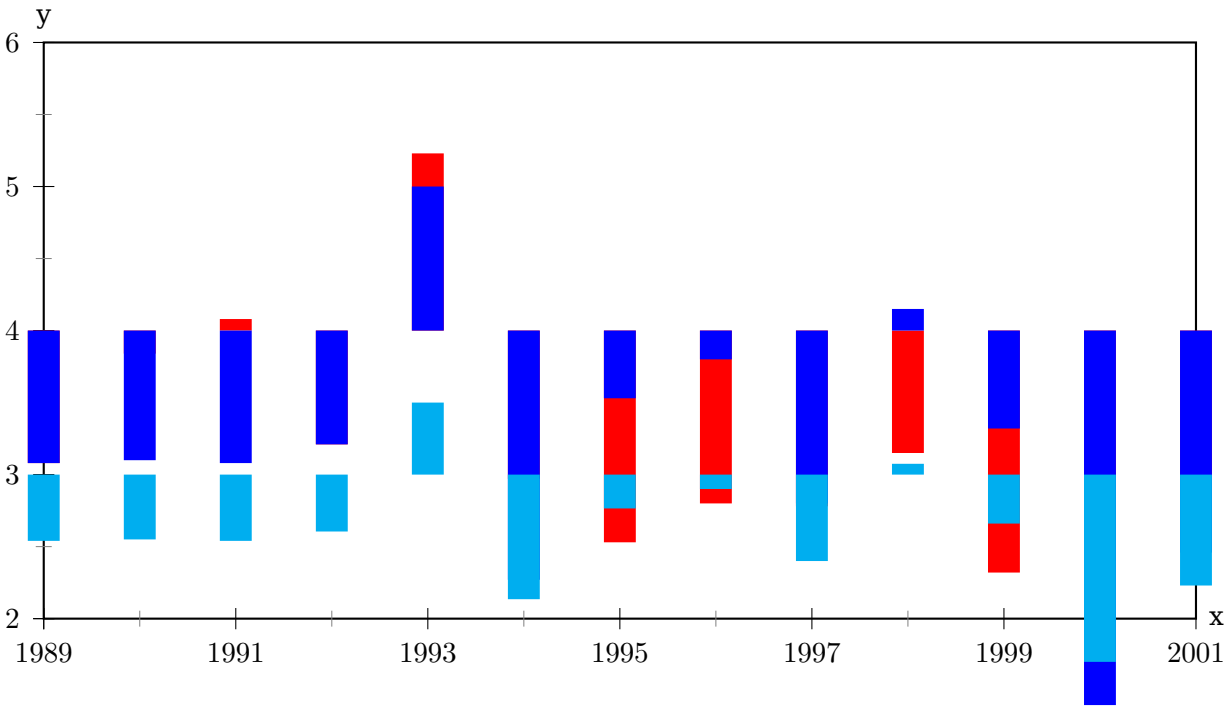

```
\readdata{\data}{demo0.data}
\psset{lly=-0.75cm,ury=0.5cm}
\pstScalePoints(1,1){}{log}
\begin{psgraph}[arrows=->,Dx=0.5,ylogBase=10,Oy=-1,xsubticks=10,%
    ysubticks=2](0,-3)(3,1){12cm}{4cm}
  \psset{Oy=-2}% must be global
  \listplot[linecolor=red,linewidth=1pt,showpoints=true,
    plotstyle=LineToXAxis]{\data}
  \listplot[plotstyle=values,rot=90]{\data}
\end{psgraph}
```




```
\psset{lly=-0.75cm,ury=0.5cm}
\readdata{\data}{demo0.data}
\pstScalePoints(1,1){}{log}
\psgraph[arrows=->,Dx=0.5,ylogBase=10,Oy=-1,subticks=4](0,-3)(3,1){6cm}{3cm}
  \listplot[linecolor=red,linewidth=2pt,showpoints=true,plotstyle=LineToXAxis]{\
     data}
\endpsgraph
```



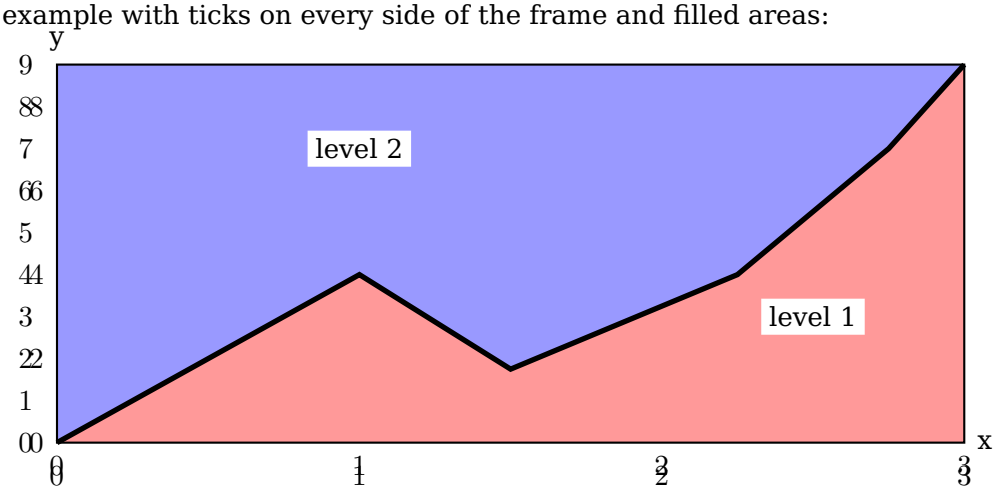

```
\readdata{\data}{demo2.data}%
\readdata{\dataII}{demo3.data}%
\pstScalePoints(1,1){1989 sub}{}
\psset{llx=-0.5cm,lly=-1cm, xAxisLabel=Year,yAxisLabel=Whatever,%
    xAxisLabelPos={c,-0.4in},yAxisLabelPos={-0.4in,c}}
\psgraph[axesstyle=frame,Dx=2,Ox=1989,subticks=2](0,0)(12,6){4in}{2in}%
  \listplot[linecolor=red,linewidth=2pt]{\data}
  \listplot[linecolor=blue,linewidth=2pt]{\dataII}
  \listplot[linecolor=cyan,linewidth=2pt,yunit=0.5]{\dataII}
\endpsgraph
```




```
\readdata{\data}{demo2.data}%
\readdata{\dataII}{demo3.data}%
\psset{llx=-0.5cm,lly=-0.75cm,plotstyle=LineToXAxis}
```

```
\pstScalePoints(1,1){1989 sub}{2 sub}
\begin{psgraph}[axesstyle=frame,Dx=2,Ox=1989,Oy=2,subticks=2](0,0)(12,4){6in}{3
    in}
  \listplot[linecolor=red,linewidth=12pt]{\data}
  \listplot[linecolor=blue,linewidth=12pt]{\dataII}
  \listplot[linecolor=cyan,linewidth=12pt,yunit=0.5]{\dataII}
\end{psgraph}
```

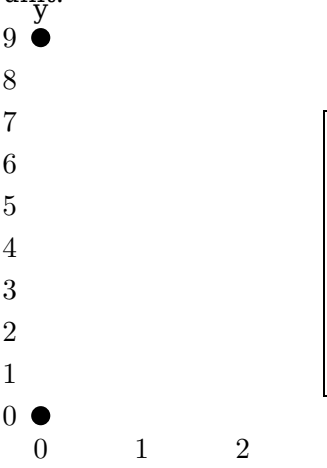
An example with ticks on every side of the frame and filled areas:




```
\def\data{0 0 1 4 1.5 1.75 2.25 4 2.75 7 3 9}
\psset{lly=-0.5cm}
\begin{psgraph}[axesstyle=none,ticks=none](0,0)(3.0,9.0){12cm}{5cm}
  \pscustom[fillstyle=solid,fillcolor=red!40,linestyle=none]{%
    \listplot{\data}
    \psline(3,9)(3,0)}
  \pscustom[fillstyle=solid,fillcolor=blue!40,linestyle=none]{%
    \listplot{\data}
    \psline(3,9)(0,9)}
  \listplot[linewidth=2pt]{\data}
  \psaxes[axesstyle=frame,ticksize=0 5pt,xsubticks=20,ysubticks=4,
    tickstyle=inner,dy=2,Dy=2,tickwidth=1.5pt,subtickcolor=black](0,0)(3,9)
  \rput*(2.5,3){level 1}\rput*(1,7){level 2}
\end{psgraph}
```

6.1. Coordinates of the psgraph area

The coordinates of the calculated area are saved in the four macros \psgraphLLx, \psgraphLLy, \psgraphURx, and \psgraphURy, which is LowerLeft, UpperLeft, LowerRight, and UpperRight. The values have no dimension but are saved in the current unit.

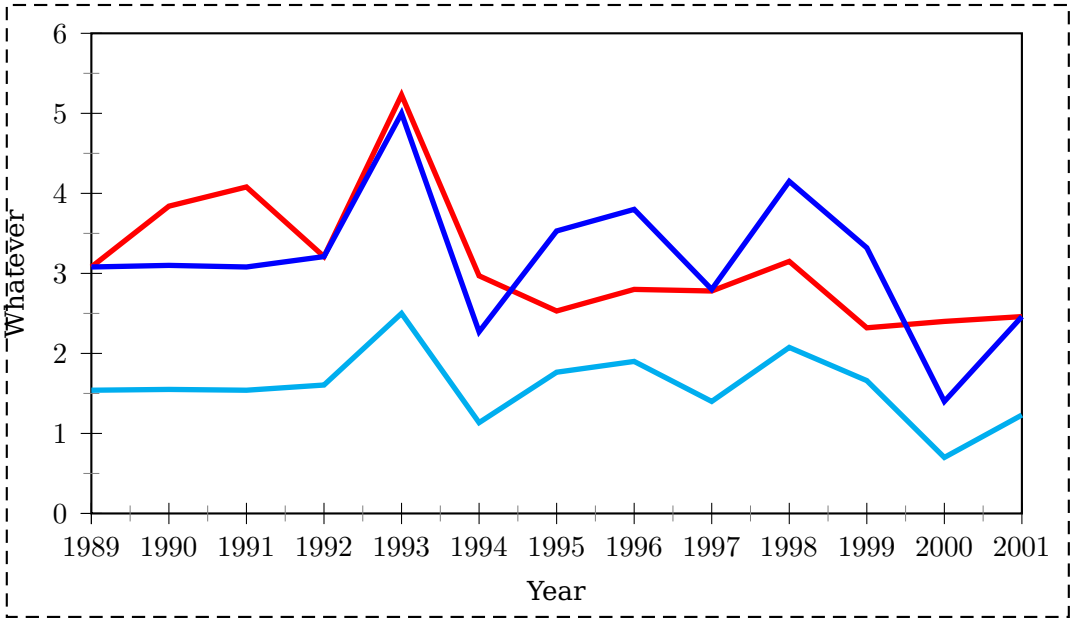



```
\psset{llx=-5mm,lly=-1cm}
\begin{psgraph}[axesstyle=none,ticks=none](0,0)(3.0,9.0)
   {4cm}{5cm}
  \psdot[dotscale=2](\psgraphLLx,\psgraphLLy)
  \psdot[dotscale=2](\psgraphLLx,\psgraphURy)
  \psdot[dotscale=2](\psgraphURx,\psgraphLLy)
  \psdot[dotscale=2](\psgraphURx,\psgraphURy)
\end{psgraph}
```

6.2. The new options for psgraph

name	default	meaning
xAxisLabel	x	label for the x-axis
yAxisLabel	y	label for the y-axis
xAxisLabelPos	{}	where to put the x-label
yAxisLabelPos	{}	where to put the y-label
xlabelsep	5pt	labelsep for the x-axis labels
ylabelsep	5pt	labelsep for the x-axis labels
llx	0pt	trim for the lower left x
lly	0pt	trim for the lower left y
urx	0pt	trim for the upper right x
ury	0pt	trim for the upper right y
axespos	bottom	draw axes first (bottom or last (top)

There is one restriction in using the trim parameters, they must been set **before** \psgraph is called. They are redundant when used as parameters of \psgraph itself. The xAxisLabelPos and yAxisLabelPos options can use the letter c for centering an x-axis or y-axis label. The c is a replacement for the x or y value. When using values with units, the position is always measured from the origin of the coordinate system, which can be outside of the visible pspicture environment.

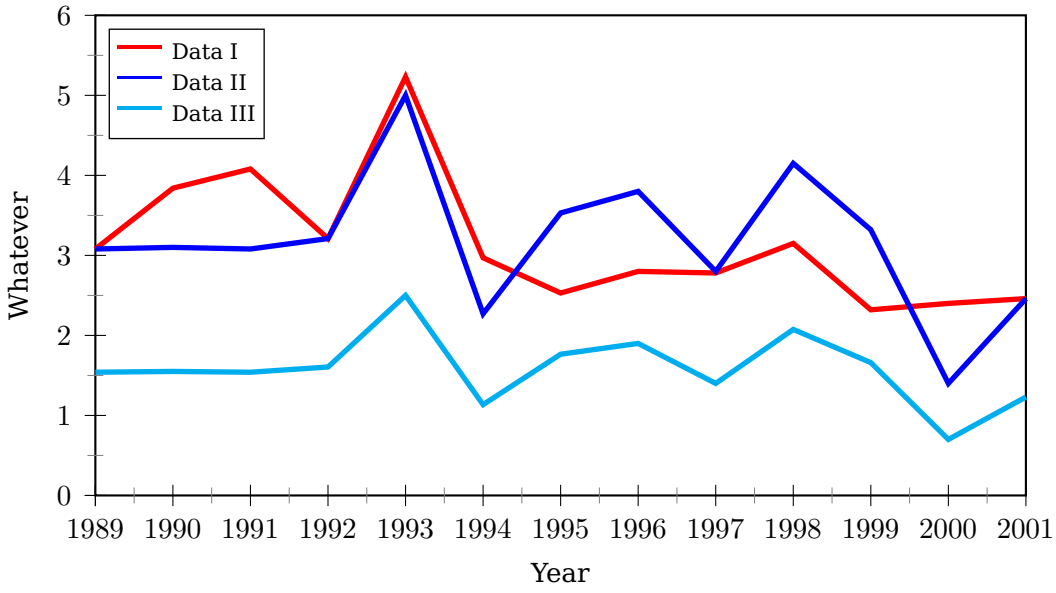



```
\readdata{\data}{demo2.data}%
\readdata{\dataII}{demo3.data}%
\psset{llx=-1cm,lly=-1.25cm,urx=0.5cm,ury=0.1in,xAxisLabel=Year,%
  yAxisLabel=Whatever,xAxisLabelPos={c,-0.4in},%
  yAxisLabelPos={-0.4in,c}}
\pstScalePoints(1,1){1989 sub}{}
\psframebox[linestyle=dashed,boxsep=false]{%
\begin{psgraph}[axesstyle=frame,Ox=1989,subticks=2](0,0)(12,6){0.8\linewidth
   }{2.5in}%
 \listplot[linecolor=red,linewidth=2pt]{\data}%
 \listplot[linecolor=blue,linewidth=2pt]{\dataII}%
 \listplot[linecolor=cyan,linewidth=2pt,yunit=0.5]{\dataII}%
\end{psgraph}%
}
```

6.3. The new macro \pslegend for psgraph

\pslegend [Reference] (*xOffset,yOffset*) {*Text*}

The reference can be one of the `lb`, `lt`, `rb`, or `rt`, where the latter is the default. The values for `xOffset` and `yOffset` must be multiples of the unit pt. Without an offset the value of `\pslabelsep` are used. The legend has to be defined *before* the environment `psgraph`.




```
\readdata{\data}{demo2.data}%
\readdata{\dataII}{demo3.data}%
\psset{llx=-1cm,lly=-1.25cm,urx=0.5cm,ury=0.1in,xAxisLabel=Year,%
  yAxisLabel=Whatever,xAxisLabelPos={c,-0.4in},%
  yAxisLabelPos={-0.4in,c}}
\pstScalePoints(1,1){1989 sub}{}
\pslegend[lt]{\red\rule[1ex]{2em}{1pt} & Data I\\
          \blue\rule[1ex]{2em}{1pt} & Data II\\
          \cyan\rule[1ex]{2em}{1pt} & Data III}
\begin{psgraph}[axesstyle=frame,Ox=1989,subticks=2](0,0)(12,6){0.8\linewidth
   }{2.5in}%
 \listplot[linecolor=red,linewidth=2pt]{\data}%
 \listplot[linecolor=blue,linewidth=2pt]{\dataII}%
 \listplot[linecolor=cyan,linewidth=2pt,yunit=0.5]{\dataII}%
\end{psgraph}%
```

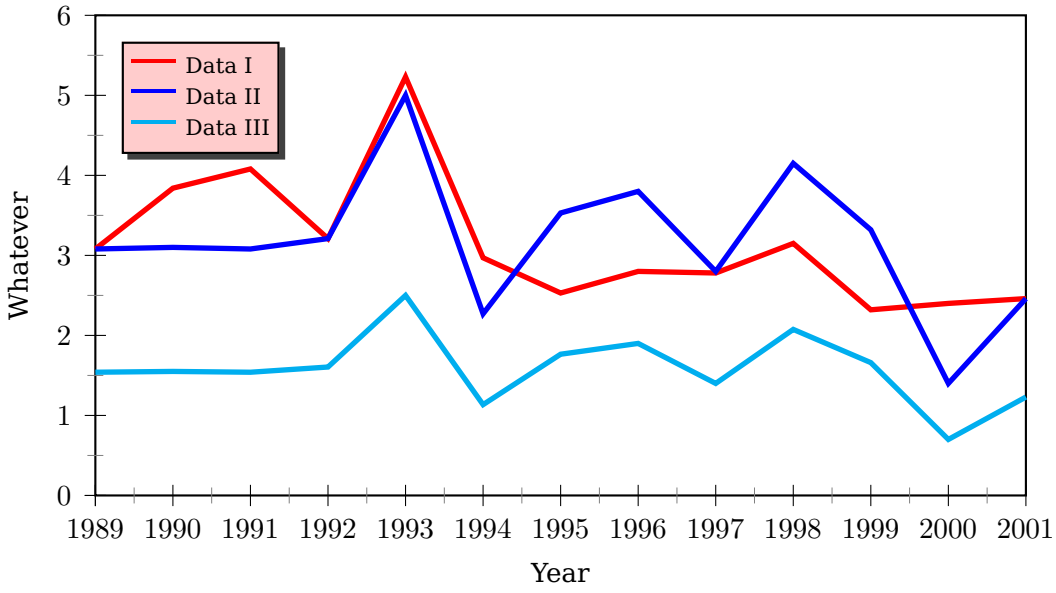
- `\pslegend` uses the commands `\tabular` and `\endtabular`, which are only available when running LaTeX. With TeX you have to redefine the macro `\pslegend@ii`:

```
\def\pslegend@ii[#1](#2){\rput[#1](!#2){\psframebox[style=legendstyle]{%
 \footnotesize\tabcolsep=2pt%
 \tabular[t]{@{}ll@{}}\pslegend@text\endtabular}}\gdef\pslegend@text{}}
```

- The fontsize can be changed locally for each cell or globally, when also redefining the macro `\pslegend@ii`.
- If you want to use more than two columns for the table or a shadow box, then redefine `\pslegend@ii`.

The macro `\psframebox` uses the style `legendstyle` which is preset to `fillstyle=solid` , `fillcolor=white` , and `linewidth=0.5pt` and can be redefined by

```
\newpsstyle{legendstyle}{fillstyle=solid,fillcolor=red!20,shadow=true}
```

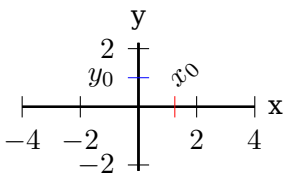



```
\readdata{\data}{demo2.data}%
\readdata{\dataII}{demo3.data}%
\psset{llx=-1cm,lly=-1.25cm,urx=0.5cm,ury=0.1in,xAxisLabel=Year,%
  yAxisLabel=Whatever,xAxisLabelPos={c,-0.4in},%
  yAxisLabelPos={-0.4in,c}}
\pstScalePoints(1,1){1989 sub}{}
\newpsstyle{legendstyle}{fillstyle=solid,fillcolor=red!20,shadow=true}
\pslegend[lt](10,10){\red\rule[1ex]{2em}{1pt} & Data I\\
                \blue\rule[1ex]{2em}{1pt} & Data II\\
                \cyan\rule[1ex]{2em}{1pt} & Data III}
\begin{psgraph}[axesstyle=frame,Ox=1989,subticks=2](0,0)(12,6){0.8\linewidth
   }{2.5in}%
 \listplot[linecolor=red,linewidth=2pt]{\data}%
 \listplot[linecolor=blue,linewidth=2pt]{\dataII}%
 \listplot[linecolor=cyan,linewidth=2pt,yunit=0.5]{\dataII}%
\end{psgraph}%
```

7. \psxTick and \psyTick

Single ticks with labels on an axis can be set with the two macros \psxTick and \psyTick. The label is set with the macro \pshlabel, the setting of mathLabel is taken into account.

\psxTick [Options] {*rotation*} (*x value*){*label*}
\psyTick [Options] {*rotation*} (*y value*){*label*}



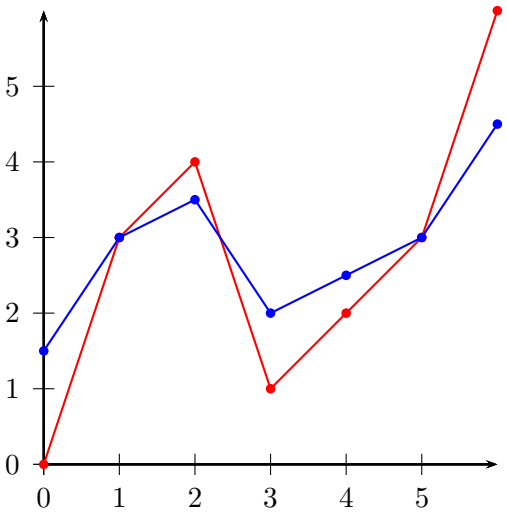

```
\begin{psgraph}[Dx=2,Dy=2](0,0)(-4,-2.2)
    (4,2.2){.5\textwidth}{!}
  \psxTick[linecolor=red,labelsep=-20pt
     ]{45}(1.25){x_0}
  \psyTick[linecolor=blue](1){y_0}
\end{psgraph}
```

8. \pstScalePoints

The syntax is

\pstScalePoints(*xScale,xScale*){*xPS*}{*yPS*}

xScale,yScale are decimal values used as scaling factors, the xPS and yPS are additional PostScript code applied to the x- and y-values of the data records. This macro is only valid for the \listplot macro!




```
\def\data{%
  0 0 1 3 2 4 3 1
  4 2 5 3 6 6 }
\begin{pspicture}(-0.5,-1)(6,6)
  \psaxes{->}(0,0)(6,6)
  \listplot[showpoints=true,%
    linecolor=red]{\data}
  \pstScalePoints(1,0.5){}{3 add}
  \listplot[showpoints=true,%
    linecolor=blue]{\data}
\end{pspicture}
```

\pstScalePoints(*1,0.5*){}{*3 add*} means that **first** the value 3 is added to the y values and **second** this value is scaled with the factor 0.5. As seen for the blue line for $x = 0$ we get $y(0) = (0 + 3) \cdot 0.5 = 1.5$.

Changes with \pstScalePoints are always global to all following \listplot macros. This is the reason why it is a good idea to reset the values at the end of the pspicture environment.

9. New or extended options

9.1. Introduction

The option `tickstyle=full |top|bottom` no longer works in the usual way. Only the additional value `inner` is valid for `pst-plot`, because everything can be set by the `ticksize` option. When using the `comma` or `trigLabels` option, the macros `\pshlabel` and `\psvlabel` shouldn't be redefined, because the package does it itself internally in these cases. However, if you need a redefinition, then do it for `\pst@@@hlabel` and `\pst@@@vlabel` with

```
\makeatletter
\def\pst@@@hlabel#1{...}
\def\pst@@@vlabel#1{...}
\makeatother
```

Table 1: All new parameters for `pst-plot`

name	*type*	*default*	*page*
`axesstyle`	`none|axes|frame|polar|inner`	`axes`	32
`barwidth`	length	0.25cm	74
`ChangeOrder`	boolean	`false`	72
`comma`	boolean	`false`	38
`decimals`	integer	-1[1]	80
`decimalSeparator`	char	.	38
`fontscale`	real	10	80
`ignoreLines`	integer	0	64
`labelFontSize`	macro	{}	36
`labels`	`all|x|y|none`	`all`	34
`llx`	length	0pt	21
`lly`	length	0pt	21
`logLines`	`none|x|y|all`	`none`	52
`mathLabel`	boolean	`false`	36
`nEnd`	integer or empty	{}	68
`nStart`	integer	0	67
`nStep`	integer	1	65
`plotNo`	integer	1	69
`plotNoMax`	integer	1	69
`plotstyle`	style	`line`	28
`polarplot`	boolean	`false`	82
`PSfont`	PS font	`Times-Romasn`	80
`psgrid`	boolean	`false`	63

continued ...

1 A negative value plots all decimals

... continued

name	*type*	*default*	*page*
quadrant	integer	4	32
subtickcolor	color	darkgray	51
subticklinestyle	solid|dashed|dotted|none	solid	52
subticks	integer	0	50
subticksize	real	0.75	50
subtickwidth	length	0.5\pslinewidth	59
tickcolor	color	black	51
ticklinestyle	solid|dashed|dotted|none	solid	52
ticks	all|x|y|none	all	47
ticksize	length [length]	-4pt 4pt	48
tickstyle	full|top|bottom|inner	full	48
tickwidth	length	0.5\pslinewidth	59
trigLabelBase	integer	0	39
trigLabels	boolean	false	39
urx	length	0pt	21
ury	length	0pt	21
valuewidth	integer	10	80
xAxis	boolean	true	34
xAxisLabel	literal	{\@empty}	21
xAxisLabelPos	(x,y) or empty	{\@empty}	21
xDecimals	integer or empty	{}	39
xEnd	integer or empty	{}	68
xLabels	list	{\empty}	29
xlabelFactor	anything	{\@empty}	37
xlabelFontSize	macro	{}	36
xlabelOffset	length	0	29
xlabelPos	bottom,axis,top	bottom	35
xLabelsRot	angle	0	29
xlogBase	integer or empty	{}	57
xmathLabel	boolean	false	36
xticklinestyle	solid|dashed|dotted|none	solid	52
xStart	integer or empty	{}	67
xStep	integer	0	65
xsubtickcolor	color	darkgray	51
xsubticklinestyle	solid|dashed|dotted|none	solid	52
xsubticks	integer	0	50
xsubticksize	real	0.75	50
xtickcolor	color	black	51
xticksize	length [length]	-4pt 4pt	48
xtrigLabels	boolean	false	44
xyAxes	boolean	true	34

continued ...

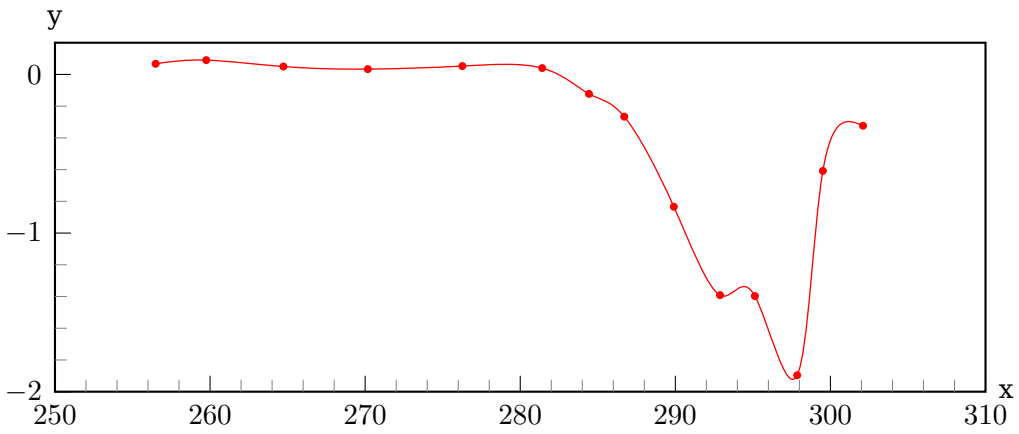
... continued

name	*type*	*default*	*page*
xyDecimals	integer or empty	{}	39
xylogBase	integer or empty	{}	54
yAxis	boolean	true	34
yAxisLabel	literal	{\@empty}	21
yAxisLabelPos	(x,y) or empty	{\@empty}	21
yDecimals	integer or empty	{}	39
yEnd	integer or empty	{}	69
yLabels	list	{\empty}	29
ylabelFactor	literal	{\empty}	37
ylabelFontSize	macro	{}	36
ylabelOffset	length	0	29
ylabelPos	left|axis|right	left	35
xLabelsRot	angle	0	29
ylogBase	integer or empty	{}	55
ymathLabel	boolean	false	36
yMaxValue	real	1.e30	30
yMinValue	real	-1.e30	30
yStart	integer or empty	{}	69
yStep	integer	0	65
ysubtickcolor	<color>	darkgray	51
ysubticklinestyle	solid|dashed|dotted|none	solid	52
ysubticks	integer	0	50
ysubticksize	real	0.75	50
ytickcolor	color>	black	51
yticklinestyle	solid|dashed|dotted|none	solid	52
yticksize	length [length]	-4pt 4pt	48
ytrigLabels	boolean	false	44

9.2. Option plotstyle (Christoph Bersch)

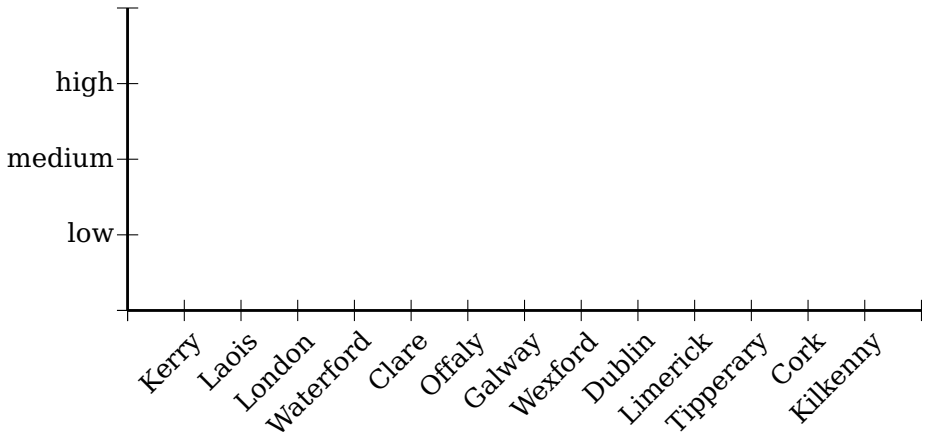
There is a new value cspline for the plotstyle to interpolate a curve with cubic splines.

```
\readdata{\foo}{data1.dat}
\begin{psgraph}[axesstyle=frame,ticksize=6pt,subticks=5,ury=1cm,
  Ox=250,Dx=10,Oy=-2,](250,-2)(310,0.2){0.8\linewidth}{0.3\linewidth}
\listplot[plotstyle=cspline,linecolor=red,linewidth=0.5pt,showpoints]{\foo}
\end{psgraph}
```



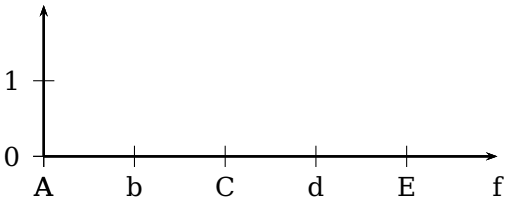
9.3. Option xLabels, yLabels, xLabelsrot, and yLabelsrot

```
\psset{xunit=0.75}
\begin{pspicture}(-2,-2)(14,4)
\psaxes[xLabels={,Kerry,Laois,London,Waterford,Clare,Offaly,Galway,Wexford,%
 Dublin,Limerick,Tipperary,Cork,Kilkenny},xLabelsRot=45,
 yLabels={,low,medium,high},mathLabel=false](14,4)
\end{pspicture}
```

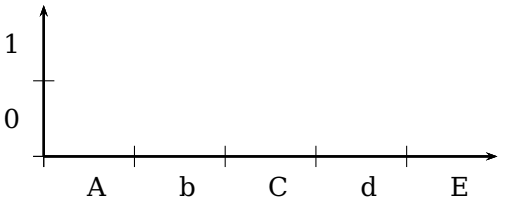


The values for xlabelsep and ylabelsep are taken into account.

9.4. Option xLabelOffset and ylabelOffset




```
\psset{xAxisLabel=,yAxisLabel=,
  llx=-5mm,urx=1cm,lly=-5mm,
  mathLabel=false,xlabelsep=-5pt,
  xLabels={A,b,C,d,E,f}}
\begin{psgraph}{->}(5,2){6cm}{2cm}
\end{psgraph}
```

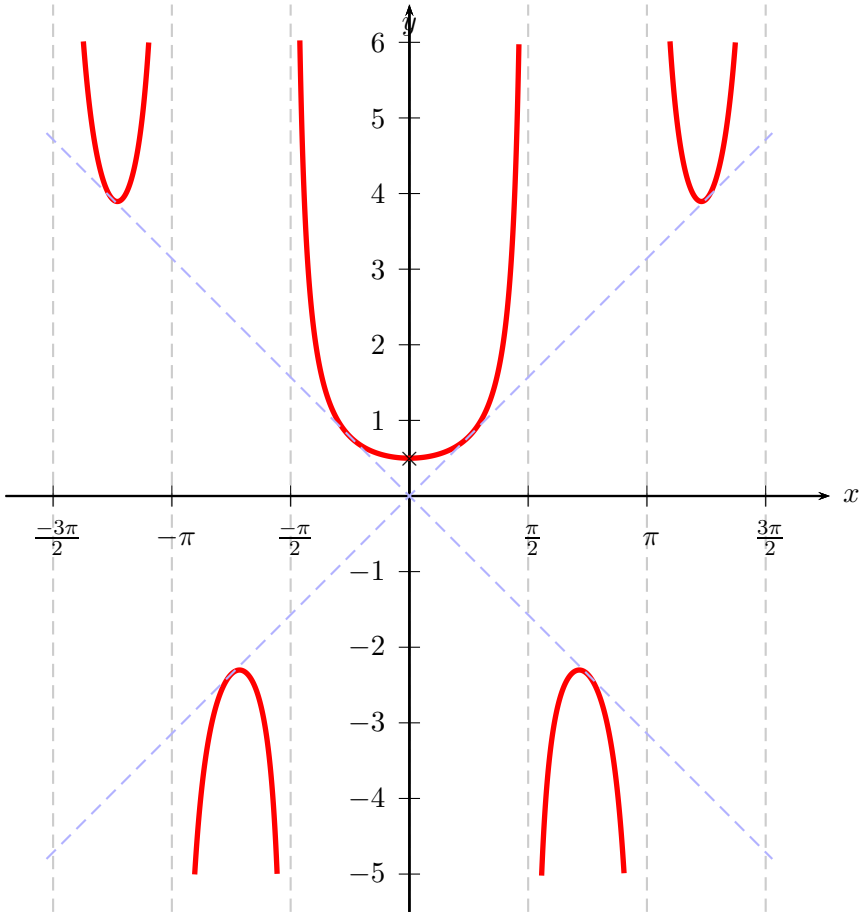


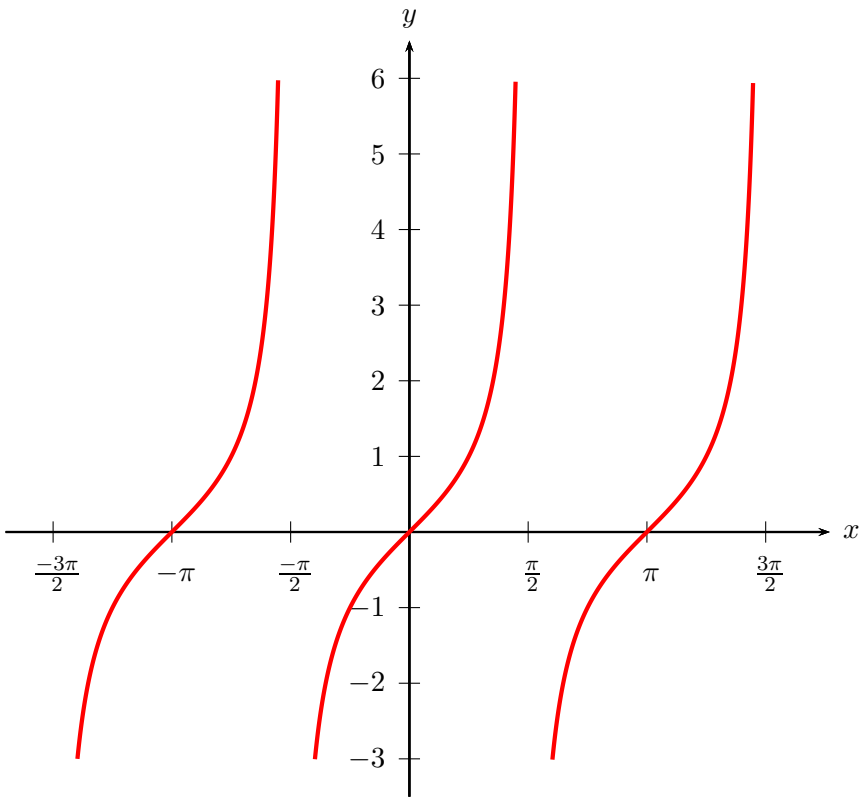

```
\psset{xAxisLabel=,yAxisLabel=,
  llx=-5mm,urx=1cm,lly=-5mm,
  mathLabel=false,xlabelsep=-5pt,
  xLabels={,A,b,C,d,E},
  xlabelOffset=-0.5,
  ylabelOffset=0.5}
\begin{psgraph}{->}(5,2){6cm}{2cm}
\end{psgraph}
```

9.5. Option yMaxValue and yMinValue

With the new optional arguments yMaxValue and yMinValue one can control the behaviour of discontinuous functions, like the tangent function. The code does not check that yMaxValue is bigger than yMinValue (if not, the function is *not* plotted at all). All four possibilities can be used, i.e. one, both or none of the two arguments yMaxValue and yMinValue can be set.

```
\begin{pspicture}(-6.5,-6)(6.5,7.5)
\multido{\rA=-4.71239+\psPiH}{7}{%
  \psline[linecolor=black!20,linestyle=dashed](\rA,-5.5)(\rA,6.5)}
\psset{algebraic,plotpoints=10000,plotstyle=line}
\psaxes[trigLabelBase=2,dx=\psPiH,xunit=\psPi,trigLabels]
  {->}(0,0)(-1.7,-5.5)(1.77,6.5)[$x$,0][$y$,-90]
\psclip{\psframe[linestyle=none](-4.55,-5.5)(5.55,6.5)}
  \psplot[yMaxValue=6,yMinValue=-5,linewidth=2pt,linecolor=red]{-4.55}{4.55}{(x
      )/(sin(2*x))}
\endpsclip
\psplot[linestyle=dashed,linecolor=blue!30]{-4.8}{4.8}{x}
\psplot[linestyle=dashed,linecolor=blue!30]{-4.8}{4.8}{-x}
\rput(0,0.5){$\times$}
\end{pspicture}
```


y
x
6
5
4
3
2
1
−1
−2
−3
−4
−5
$\frac{-3\pi}{2}$
$-\pi$
$\frac{-\pi}{2}$
$\frac{\pi}{2}$
π
$\frac{3\pi}{2}$




```
\begin{pspicture}(-6.5,-4)(6.5,7.5)
\psaxes[trigLabelBase=2,dx=\psPiH,xunit=\psPi,trigLabels]%
  {->}(0,0)(-1.7,-3.5)(1.77,6.5)[$x$,0][$y$,90]
\psplot[yMaxValue=6,yMinValue=-3,linewidth=1.6pt,plotpoints=2000,
  linecolor=red,algebraic]{-4.55}{4.55}{tan(x)}
\end{pspicture}
```

9.6. Option axesstyle

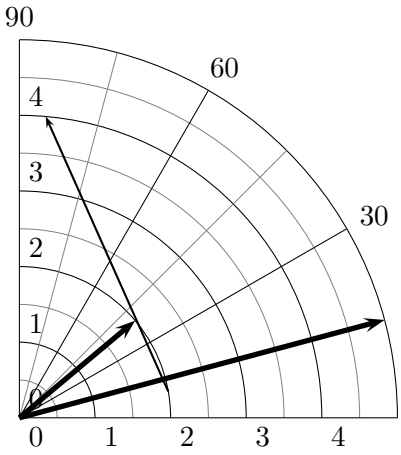
There is a new axes style `polar` which plots a polar coordinate system.

Syntax:

```
\psplot[axesstyle=polar](Rx,Angle)
\psplot[axesstyle=polar](...)(Rx,Angle)
\psplot[axesstyle=polar](...)(...)(Rx,Angle)
```

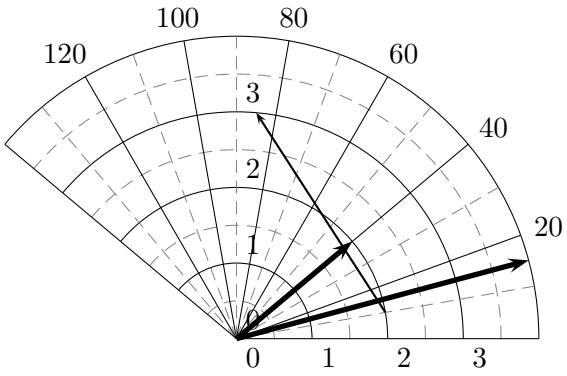
Important is the fact, that only one pair of coordinates is taken into account for the radius and the angle. It is *always* the last pair in a sequence of allowed coordinates for the `\psaxes` macro. The other ones are ignored; they are not valid for the polar coordinate system. However, if the angle is set to 0 it is changed to 360 degrees for a full circle.

```
\begin{pspicture}(-1,-1)(5.75,5.75)
\psaxes[axesstyle=polar,
  subticks=2](5,90)
\psline[linewidth=2pt]{->}(5;15)
\psline[linewidth=2pt]{->}(2;40)
\psline{->}(2;10)(4;85)
\end{pspicture}
```

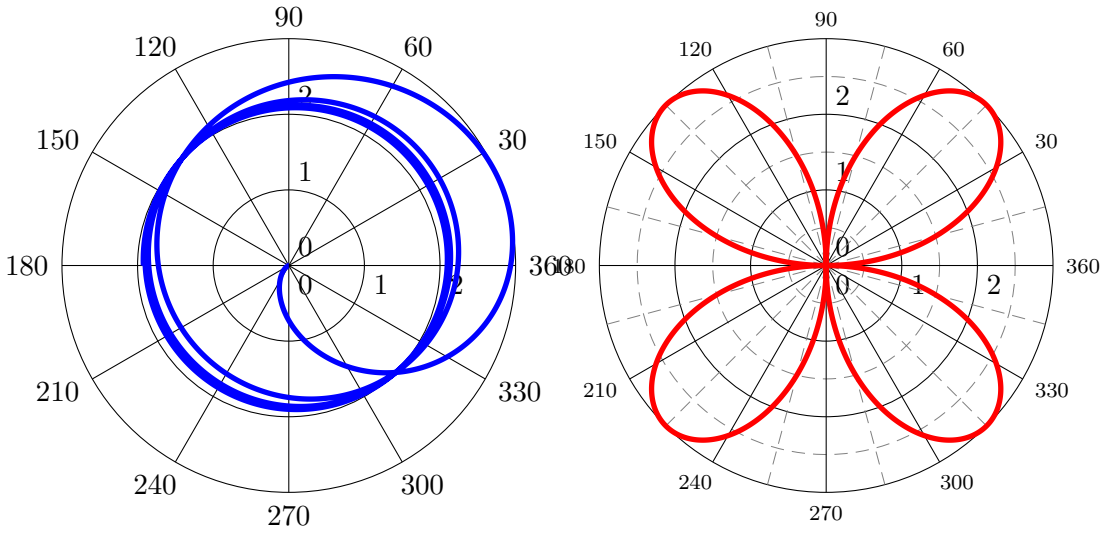


All valid optional arguments for the axes are also possible for the polar style, if they make sense ... :-) Important are the `Dy` option, it defines the angle interval and `subticks`, for the intermediate circles and lines. The number can be different for the circles (`ysubticks`) and the lines (`xsubticks`).

```
\begin{pspicture}(-3,-1)(4.5,4.5)
\psaxes[axesstyle=polar,
  subticklinestyle=dashed,
  subticks=2,Dy=20](4,140)
\psline[linewidth=2pt]{->}(4;15)
\psline[linewidth=2pt]{->}(2;40)
\psline{->}(2;10)(3;85)
\end{pspicture}
```




```
\begin{pspicture}(-3.5,-3.5)(3.5,3.5)
\psaxes[axesstyle=polar](3,0)
\psplot[polarplot,algebraic,linecolor=blue,linewidth=2pt,
   plotpoints=2000]{0}{TwoPi 4 mul}{2*(sin(x)-x)/(cos(x)+x)}
\end{pspicture}
%
\begin{pspicture}(-3.5,-3.5)(3.5,3.5)
\psaxes[axesstyle=polar,subticklinestyle=dashed,subticks=2,
 xlabelFontSize=\scriptstyle](3,360)
\psplot[polarplot,algebraic,linecolor=red,linewidth=2pt,
   plotpoints=2000]{0}{TwoPi}{6*sin(x)*cos(x)}
\end{pspicture}
```

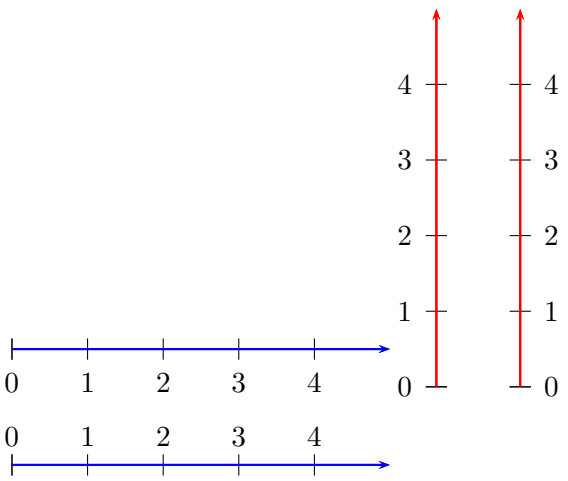


9.7. Option xyAxes, xAxis and yAxis

Syntax:

```
xyAxes=true|false
xAxis=true|false
yAxis=true|false
```

Sometimes there is only a need for one axis with ticks. In this case you can set one of the preceding options to false. The xyAxes only makes sense when you want to set both x and y to true with only one command, back to the default, because with xyAxes=falseyou get nothing with the \psaxes macro.




```
\begin{pspicture}(5,1)
\psaxes[yAxis=false,linecolor=blue
    ]{->}(0,0.5)(5,0.5)
\end{pspicture}
\begin{pspicture}(1,5)
\psaxes[xAxis=false,linecolor=red
    ]{->}(0.5,0)(0.5,5)
\end{pspicture}
\begin{pspicture}(1,5)
\psaxes[xAxis=false,linecolor=red,
   ylabelPos=right]{->}(0.5,0)(0.5,5)
\end{pspicture}\\[0.5cm]
\begin{pspicture}(5,1)
\psaxes[yAxis=false,linecolor=blue,
   xlabelPos=top]{->}(0,0.5)(5,0.5)
\end{pspicture}
```

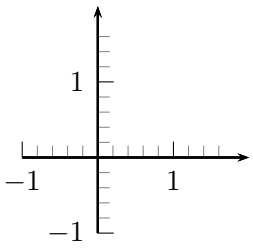
As seen in the example, a single y axis gets the labels on the left side. This can be changed with the option ylabelPos or with xlabelPos for the x-axis.

9.8. Option labels

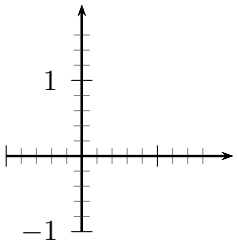
Syntax:

```
labels=all|x|y|none
```

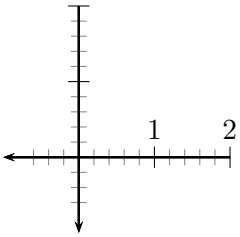
This option was already in the pst-plot package and only mentioned here for completeness.



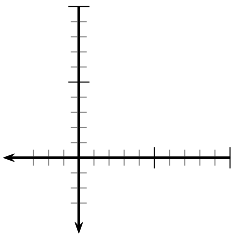

```
\psset{ticksize=6pt}
\begin{pspicture}(-1,-1)(2,2)
\psaxes[labels=all,subticks=5]{->}(0,0)(-1,-1)(2,2)
\end{pspicture}
```




```
\begin{pspicture}(-1,-1)(2,2)
\psaxes[labels=y,subticks=5]{->}(0,0)(-1,-1)(2,2)
\end{pspicture}
```




```
\begin{pspicture}(-1,-1)(2,2)
\psaxes[labels=x,subticks=5]{->}(0,0)(2,2)(-1,-1)
\end{pspicture}
```



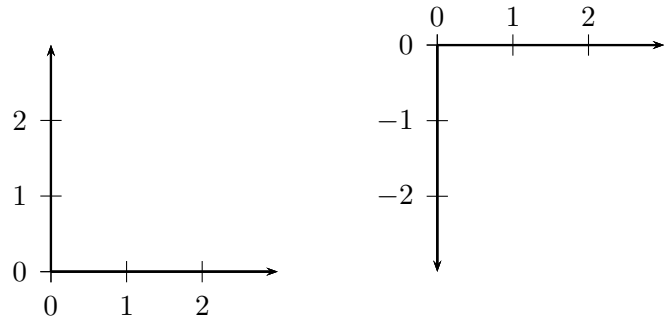
```
\begin{pspicture}(-1,-1)(2,2)
\psaxes[labels=none,subticks=5]{->}(0,0)(2,2)(-1,-1)
\end{pspicture}
```

9.9. Options xlabelPos and ylabelPos

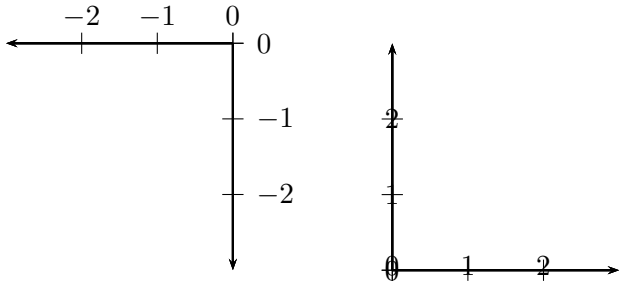
Syntax:

```
xlabelPos=bottom|axis|top
ylabelPos=left|axis|right
```

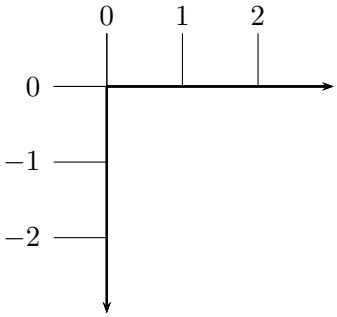
By default the labels for ticks are placed at the bottom (x axis) and left (y-axis). If both axes are drawn in the negative direction the default is top (x axis) and right (y axis). It be changed with the two options xlabelPos and ylabelPos. With the value axis the user can place the labels depending on the value of labelsep, which is taken into account for axis.




```
\begin{pspicture}(3,3)
\psaxes{->}(3,3)
\end{pspicture}\hspace{2cm}
\begin{pspicture}(3,-3)
\psaxes[xlabelPos=top
    ]{->}(3,-3)
\end{pspicture}
```




```
\begin{pspicture}(-3,-3)
\psaxes{->}(-3,-3)
\end{pspicture}\hspace{2cm}
\begin{pspicture}(3,3)
\psaxes[labelsep=0pt,
   ylabelPos=axis,
   xlabelPos=axis]{->}(3,3)
\end{pspicture}
```



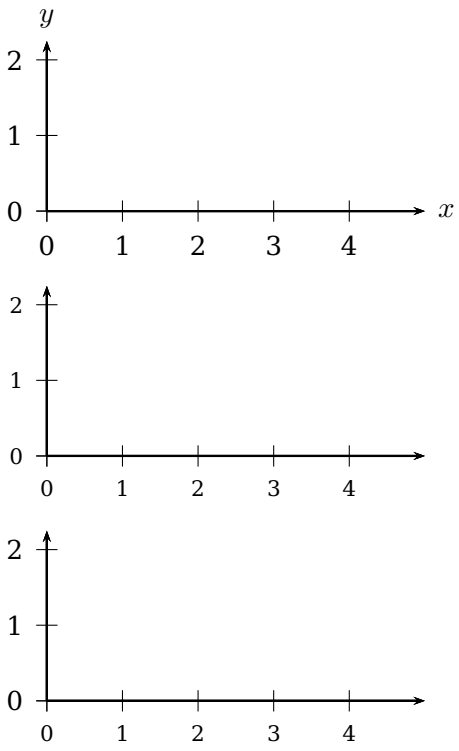

```
\begin{pspicture}(-1,1)(3,-3)
\psaxes[xlabelPos=top,
   xticksize=0 20pt,
   yticksize=-20pt 0]{->}(3,-3)
\end{pspicture}
```

9.10. Options x|ylabelFontSize and x|ymathLabel

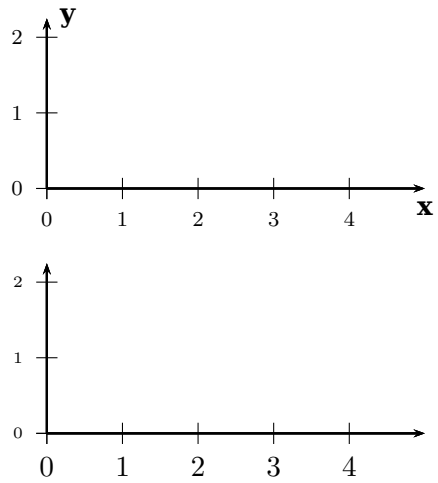
This option sets the horizontal **and** vertical font size for the labels depending on the option `mathLabel` (`xmathLabel`/`ymathLabel`) for the text or the math mode. It will be overwritten when another package or a user defines

```
\def\pshlabel#1{\xlabelFontSize ...}
\def\psvlabel#1{\ylabelFontSize ...}
\def\pshlabel#1{$\xlabelFontSize ...$}% for mathLabel=true (default)
\def\psvlabel#1{$\ylabelFontSize ...$}% for mathLabel=true (default)
```

in another way. Note that for `mathLabel=true`the font size must be set by one of the mathematical styles `\textstyle`, `\displaystyle`, `\scriptstyle`, or `\scriptscriptstyle`.




```
\psset{mathLabel=false}
\begin{pspicture}(-0.25,-0.25)(5,2.25)
\psaxes{->}(5,2.25)[$x$,0][$y$,90]
\end{pspicture}\\[20pt]
\begin{pspicture}(-0.25,-0.25)(5,2.25)
\psaxes[labelFontSize=\footnotesize
    ]{->}(5,2.25)
\end{pspicture}\\[20pt]
\begin{pspicture}(-0.25,-0.25)(5,2.25)
\psaxes[xlabelFontSize=\footnotesize
    ]{->}(5,2.25)
\end{pspicture}\\[20pt]
```

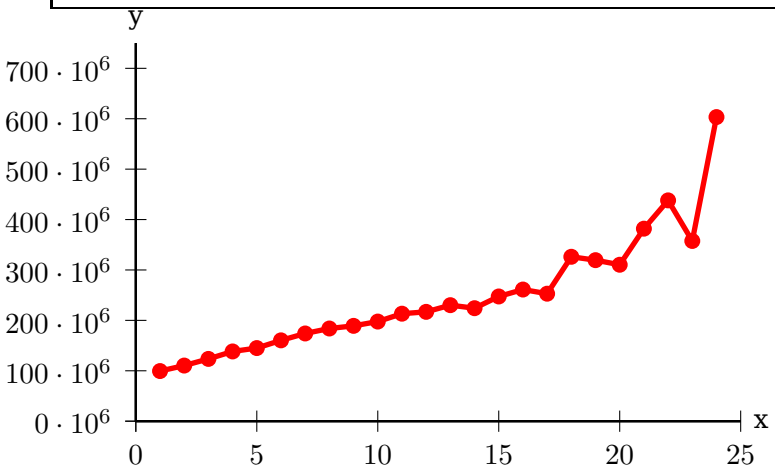



```
\begin{pspicture}(-0.25,-0.25)(5,2.25)
\psaxes[labelFontSize=\scriptstyle]{->}(5,2.25)
    [\textbf{x},-90][\textbf{y},0]
\end{pspicture}\\[20pt]
\psset{mathLabel=true}
\begin{pspicture}(-0.25,-0.25)(5,2.25)
\psaxes[ylabelFontSize=\scriptscriptstyle
    ]{->}(5,2.25)
\end{pspicture}\\[20pt]
```

9.11. Options xlabelFactor and ylabelFactor

When having big numbers as data records then it makes sense to write the values as $< number > \cdot 10^{<exp>}$. These new options allow you to define the additional part of the value, but it must be set in math mode when using math operators or macros like \cdot!

```
\readdata{\data}{demo1.data}
\pstScalePoints(1,0.000001){}{}% (x,y){additional x operator}{y op}
\psset{llx=-1cm,lly=-1cm}
\psgraph[ylabelFactor=\cdot 10^6,Dx=5,Dy=100](0,0)(25,750){8cm}{5cm}
  \listplot[linecolor=red, linewidth=2pt, showpoints=true]{\data}
\endpsgraph
\pstScalePoints(1,1){}{}% reset
```

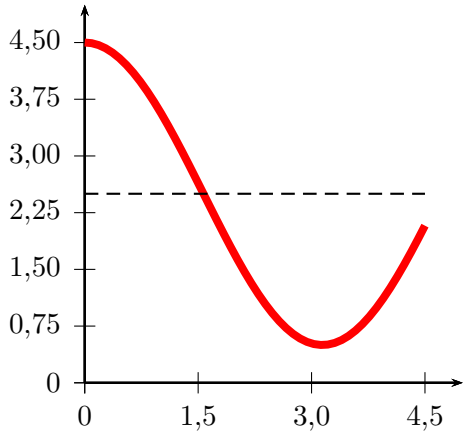


9.12. Options decimalSeparator and comma

Syntax:

```
comma=false|true
decimalSeparator=<charactor>
```

Setting the option comma to true gives labels with a comma as a decimal separator instead of the default dot. comma and comma=true is the same. The optional argument decimalSeparator allows an individual setting for languages with a different character than a dot or a comma. The character has to be set into braces, if it is an active one, e. g. decimalSeparator={,}.



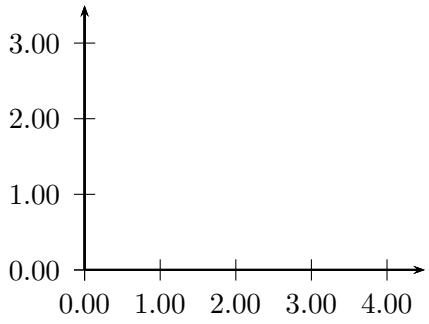

```
\begin{pspicture}(-0.5,-0.5)(5,5.5)
\psaxes[Dx=1.5,comma,Dy=0.75,dy=0.75]{->}(5,5)
\psplot[linecolor=red,linewidth=3pt]{0}{4.5}%
  {x RadtoDeg cos 2 mul 2.5 add}
\psline[linestyle=dashed](0,2.5)(4.5,2.5)
\end{pspicture}
```

9.13. Options xyDecimals, xDecimals and yDecimals

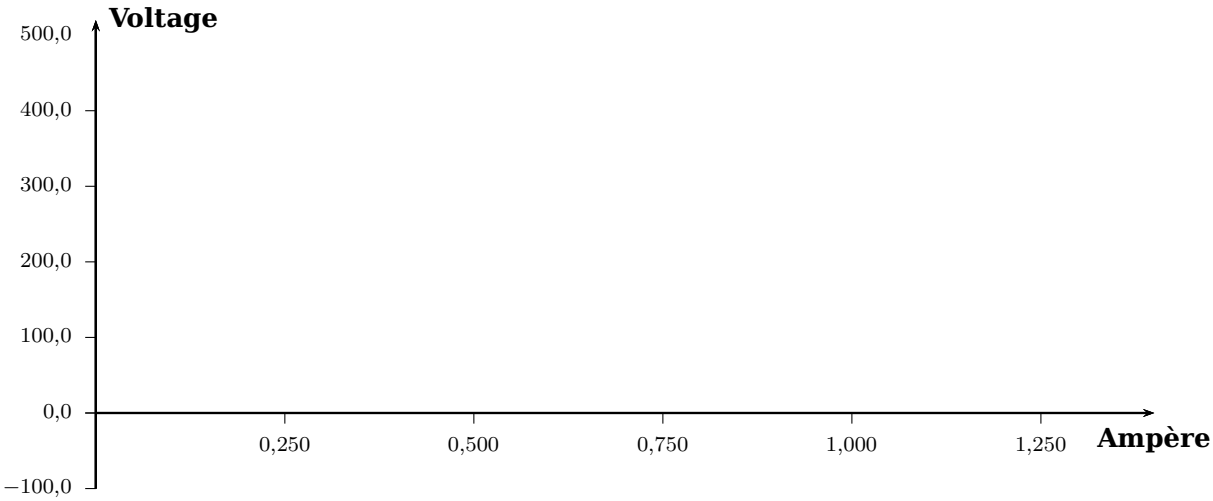
Syntax:

```
xyDecimals=<number>
xDecimals=<any>
yDecimals=<any>
```

By default the labels of the axes get numbers with or without decimals, depending on the numbers itself. With these options it is possible to determine the decimals, where the option `xyDecimals` sets this identical for both axes. `xDecimals` only for the x and `yDecimals` only for the y axis. The default setting `{}` means, that you'll get the standard behaviour.




```
\begin{pspicture}(-1.5,-0.5)(5,3.75)
  \psaxes[xyDecimals=2]{->}(0,0)(4.5,3.5)
\end{pspicture}
```




```
\psset{xunit=10cm,yunit=0.01cm,labelFontSize=\scriptstyle}
\begin{pspicture}(-0.1,-150)(1.5,550.0)
  \psaxes[Dx=0.25,Dy=100,ticksize=-4pt 0,comma,xDecimals=3,yDecimals=1]{->}%
    (0,0)(0,-100)(1.4,520)[\textbf{Amp\'ere},-90][\textbf{Voltage},0]
\end{pspicture}
```

9.14. Options trigLabels, xtrigLabels, ytrigLabels, and trigLabelBase for an axis with trigonmetrical units

With the option `trigLabels=true` *only* the labels on the x axis are trigonometrical ones. It is the same than setting `xtrigLabels=true`. The option `trigLabelBase` sets

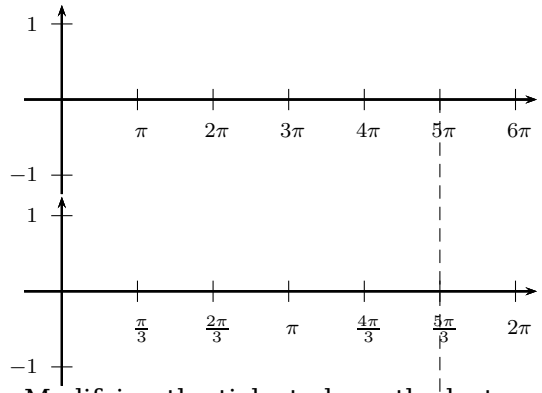
the denominator of fraction. The default value of 0 is the same as no fraction. The following constants are defined in the package:

```
\def\psPiFour{12.566371}
\def\psPiTwo{6.283185}
\def\psPi{3.14159265}
\def\psPiH{1.570796327}
\newdimen\pstRadUnit
\newdimen\pstRadUnitInv
\pstRadUnit=1.047198cm % this is pi/3
\pstRadUnitInv=0.95493cm % this is 3/pi
```

Because it is a bit complicated to set the right values, we show some more examples here. For **all** following examples in this section we did a global

```
\psset{trigLabels,labelFontSize=\scriptstyle}
```

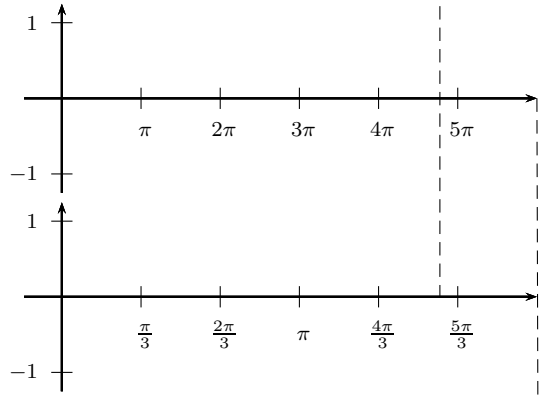
Translating the decimal ticks to trigonometrical ones makes no real sense, because every 1 xunit (1cm) is a tick and the last one is at 6cm.




```
\begin{pspicture}(-0.5,-1.25)(6.5,1.25)%
  \pnode(5,0){A}%
  \psaxes{->}(0,0)(-.5,-1.25)(\psPiTwo,1.25)
\end{pspicture}
```

```
\begin{pspicture}(-0.5,-1.25)(6.5,1.25)%
  \psaxes[trigLabelBase=3]{->}(0,0)(-0.5,-1.25)(\
     psPiTwo,1.25)
\end{pspicture}
```

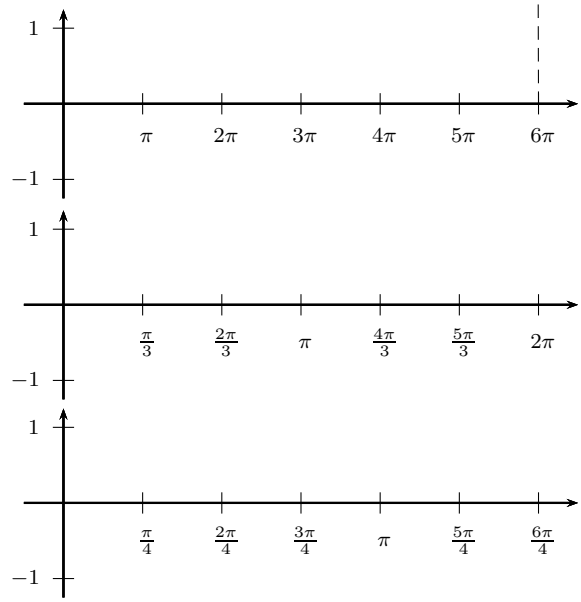
Modifying the ticks to have the last one exactly at the end is possible with a different dx value ($\frac{\pi}{3} \approx 1.047$):




```
\begin{pspicture}(-0.5,-1.25)(6.5,1.25)\pnode(\
   psPiTwo,0){C}%
  \psaxes[dx=\pstRadUnit]{->}(0,0)(-0.5,-1.25)(\
     psPiTwo,1.25)
\end{pspicture}%
```

```
\begin{pspicture}(-0.5,-1.25)(6.5,1.25)\pnode
   (5,0){B}%
  \psaxes[dx=\pstRadUnit,trigLabelBase=3]
     {->}(0,0)(-0.5,-1.25)(\psPiTwo,1.25)
\end{pspicture}%
```

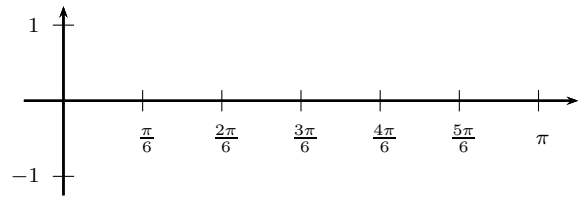
Set everything globally in radian units. Now 6 units on the x-axis are 6π. Using `trigLabelBase=3` reduces this value to 2π, a.s.o.




```
\psset{xunit=\pstRadUnit}%
\begin{pspicture}(-0.5,-1.25)(6.5,1.25)\pnode
   (6,0){D}%
  \psaxes{->}(0,0)(-0.5,-1.25)(6.5,1.25)%
\end{pspicture}%
```

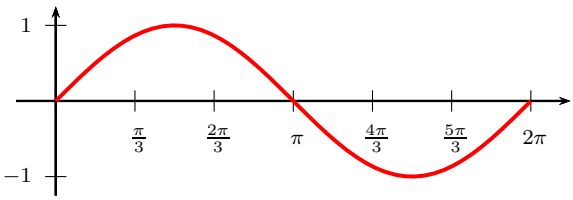
```
\psset{xunit=\pstRadUnit}%
\begin{pspicture}(-0.5,-1.25)(6.5,1.25)
  \psaxes[trigLabelBase=3]{->}(0,0)(-0.5,-1.25)
     (6.5,1.25)
\end{pspicture}%
```

```
\psset{xunit=\pstRadUnit}%
\begin{pspicture}(-0.5,-1.25)(6.5,1.25)
  \psaxes[trigLabelBase=4]{->}(0,0)(-0.5,-1.25)
     (6.5,1.25)
\end{pspicture}%
```

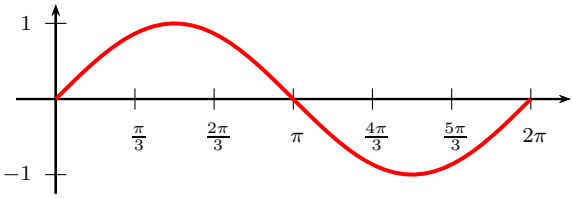



```
\psset{xunit=\pstRadUnit}%
\begin{pspicture}(-0.5,-1.25)(6.5,1.25)
  \psaxes[trigLabelBase=6]{->}(0,0)(-0.5,-1.25)
      (6.5,1.25)
\end{pspicture}%
```

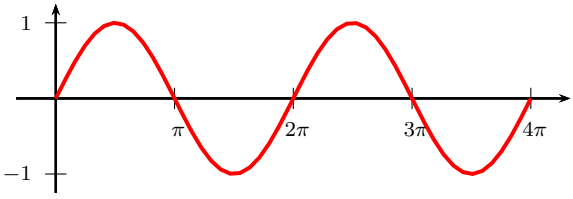
The best way seems to be to set the x-unit to `\pstRadUnit`. Plotting a function doesn't consider the value for `trigLabelBase`, it has to be done by the user. The first example sets the unit locally for the `\psplot` back to 1cm, which is needed, because we use this unit on the PostScript side.



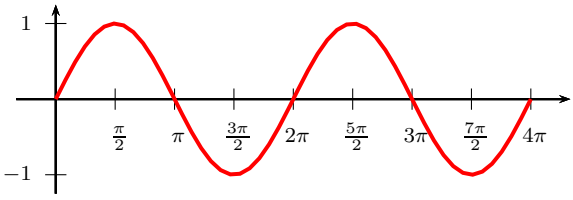

```
\psset{xunit=\pstRadUnit}%
\begin{pspicture}(-0.5,-1.25)(6.5,1.25)
  \psaxes[trigLabelBase=3]{->}(0,0)(-0.5,-1.25)
      (6.5,1.25)
  \psplot[xunit=1cm,linecolor=red,linewidth=1.5pt
      ]{0}{\psPiTwo}{x RadtoDeg sin}
\end{pspicture}
```




```
\psset{xunit=\pstRadUnit}%
\begin{pspicture}(-0.5,-1.25)(6.5,1.25)
  \psaxes[trigLabelBase=3]{->}(0,0)(-0.5,-1.25)
      (6.5,1.25)
  \psplot[linecolor=red,linewidth=1.5pt]{0}{6}{x
      Pi 3 div mul RadtoDeg sin}
\end{pspicture}
```

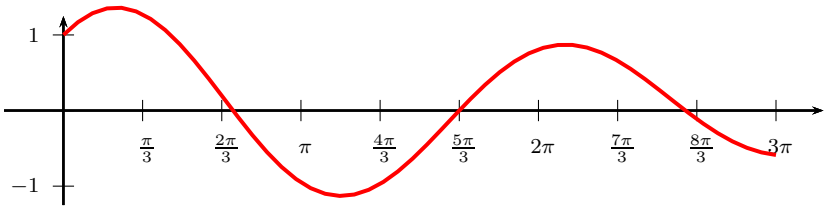



```
\psset{xunit=\pstRadUnit}%
\begin{pspicture}(-0.5,-1.25)(6.5,1.25)
  \psaxes[dx=1.5]{->}(0,0)(-0.5,-1.25)(6.5,1.25)
  \psplot[xunit=0.5cm,linecolor=red,linewidth=1.5
      pt]{0}{\psPiFour}{x RadtoDeg sin}
\end{pspicture}
```

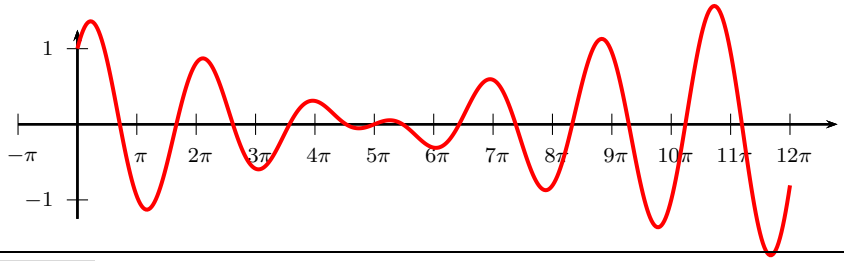



```
\psset{xunit=\pstRadUnit}%
\begin{pspicture}(-0.5,-1.25)(6.5,1.25)
  \psaxes[dx=0.75,trigLabelBase=2]{->}(0,0)
      (-0.5,-1.25)(6.5,1.25)
  \psplot[xunit=0.5cm,linecolor=red,linewidth=1.5
      pt]{0}{\psPiFour}{x RadtoDeg sin}
\end{pspicture}
```

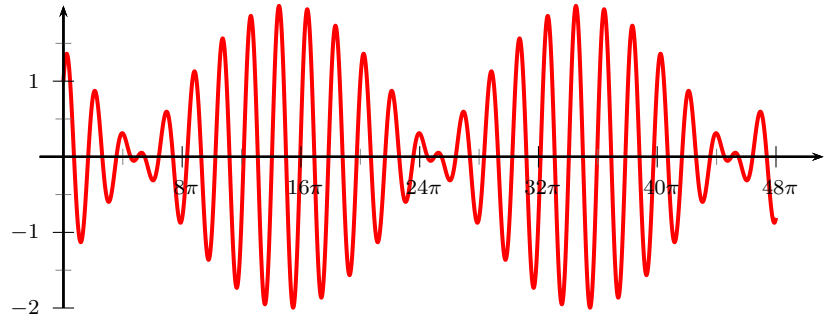
It is also possible to set the x unit and dx value to get the labels right. But this needs some more understanding as to how it really works. A `xunit=1.570796327` sets the unit to $\pi/2$ and a `dx=0.666667` then puts at every $2/3$ of the unit a tick mark and a label. The length of the x-axis is 6.4 units which is $6.4 \cdot 1.570796327cm \approx 10cm$. The function then is plotted from 0 to $3\pi = 9.424777961$.



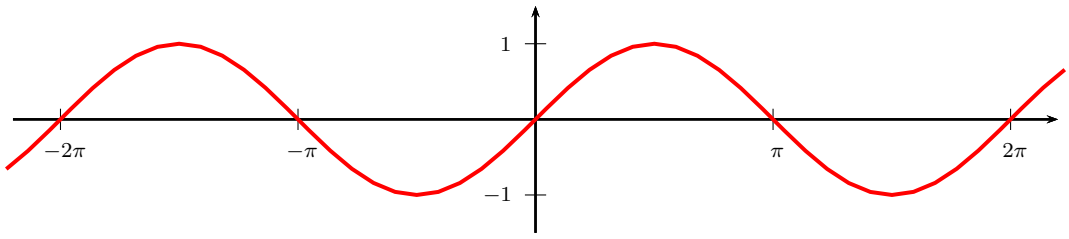

```
\begin{pspicture}(-0.5,-1.25)(10,1.25)
  \psaxes[xunit=\psPiH,trigLabelBase=3,dx=0.666667]{->}(0,0)(-0.5,-1.25)
      (6.4,1.25)
  \psplot[linecolor=red,linewidth=1.5pt]{0}{9.424777961}{%
  x RadtoDeg dup sin exch 1.1 mul cos add}
\end{pspicture}
```



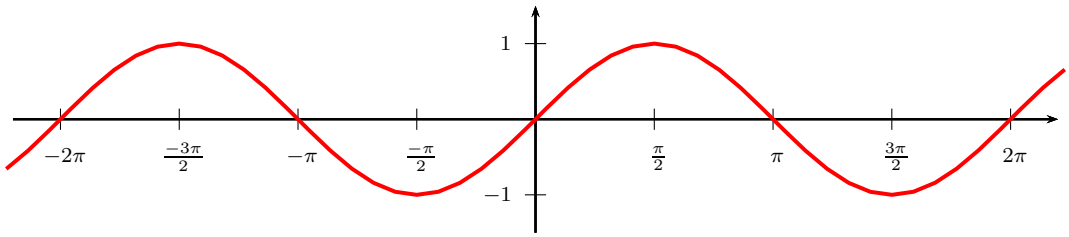

```
\psset{unit=1cm}
  \psplot[xunit=0.25,plotpoints=500,linecolor=red,linewidth=1.5pt]{0}{37.70}{%
  x RadtoDeg dup sin exch 1.1 mul cos add}
\end{pspicture}
```




```
\psset{unit=1cm}
\begin{pspicture}(-0.5,-1.25)(10,1.25)
  \psplot[xunit=0.0625,linecolor=red,linewidth=1.5pt,%
     plotpoints=5000]{0}{150.80}%
  {x RadtoDeg dup sin exch 1.1 mul cos add}
  \psaxes[xunit=\psPi,dx=0.5,Dx=8]{->}(0,0)(-0.25,-1.25)(3.2,1.25)
\end{pspicture}
```

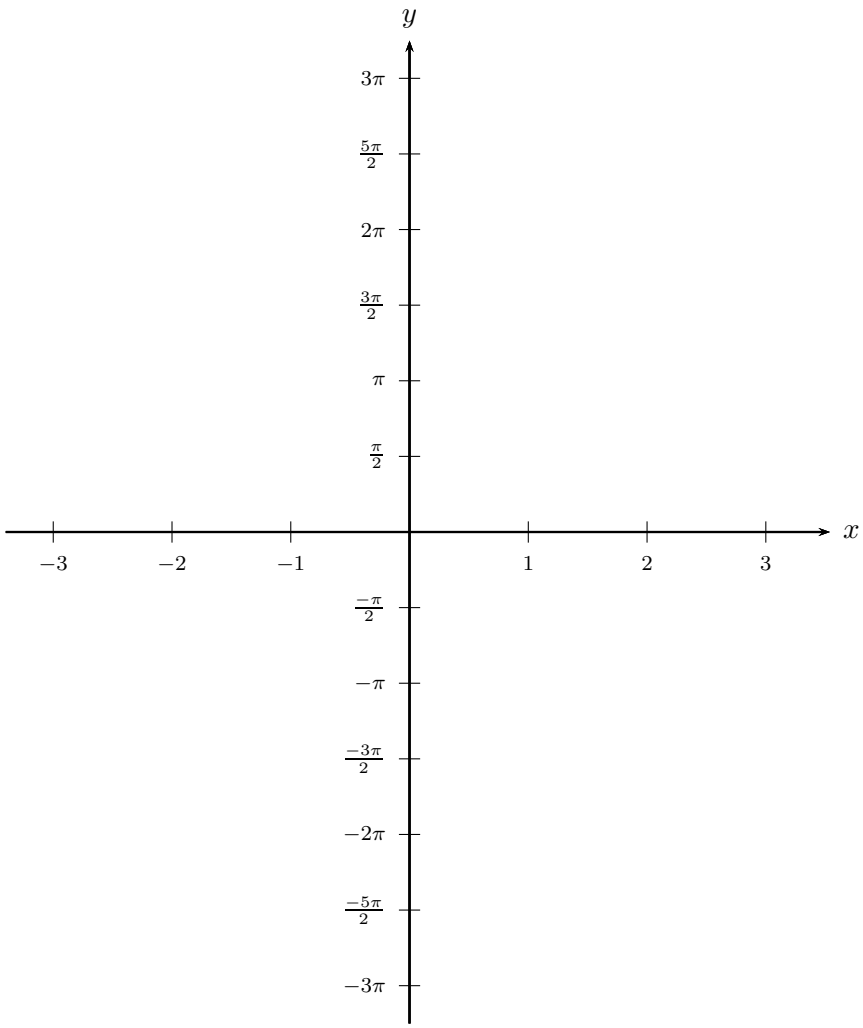


```
\begin{pspicture}(-7,-1.5)(7,1.5)
  \psaxes[trigLabels=true,xunit=\psPi]{->}(0,0)(-2.2,-1.5)(2.2,1.5)
  \psplot[linecolor=red,linewidth=1.5pt]{-7}{7}{x RadtoDeg sin}
\end{pspicture}
```



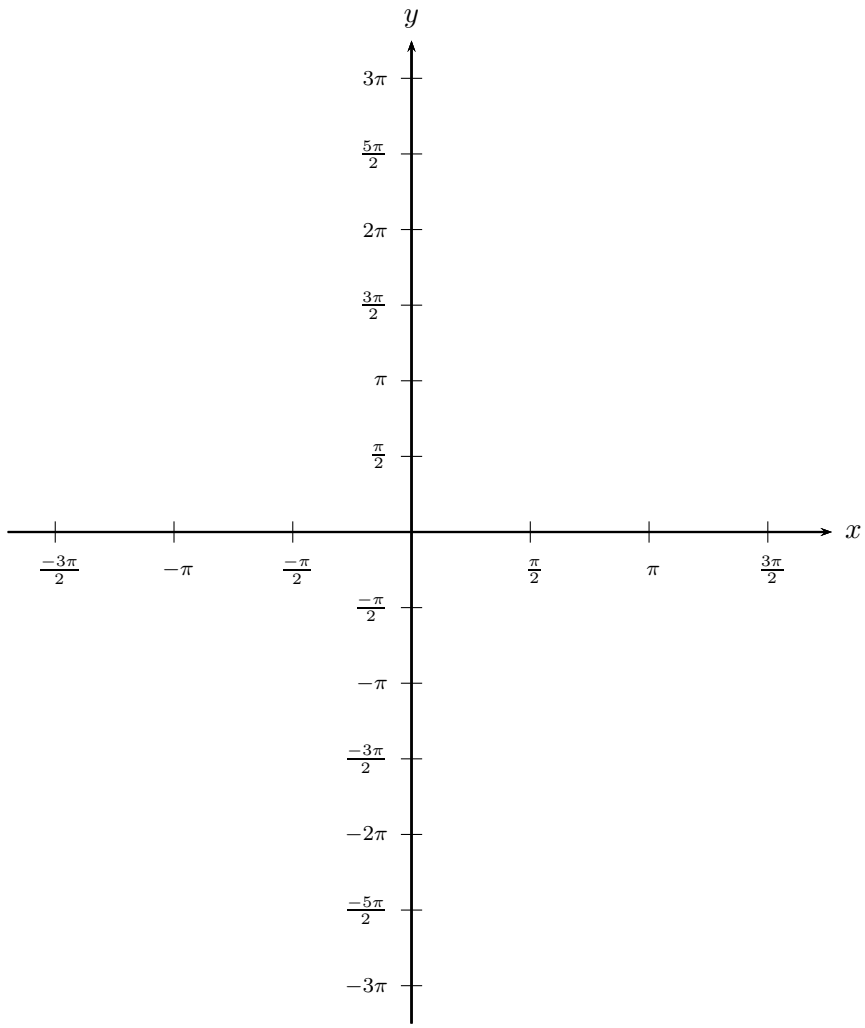

```
\begin{pspicture}(-7,-1.5)(7,1.5)
  \psaxes[trigLabels=true,
  trigLabelBase=2,dx=\psPiH,xunit=\psPi]{->}(0,0)(-2.2,-1.5)(2.2,1.5)
  \psplot[linecolor=red,linewidth=1.5pt]{-7}{7}{x RadtoDeg sin}
\end{pspicture}
```

The setting of trigonometrical labels with `ytriglabels=true`for the y axis is the same as for the x axis.




```
\psset{unit=1cm}
\begin{pspicture}(-6.5,-7)(6.5,7.5)
\psaxes[trigLabelBase=2,dx=\psPiH,xunit=\psPi,ytrigLabels]
  {->}(0,0)(-1.7,-6.5)(1.77,6.5)[$x$,0][$y$,90]
\end{pspicture}
```

Also setting labels for the x axis is possible with `trigLabels=true`or alternatively with `ytrigLabels=true`.



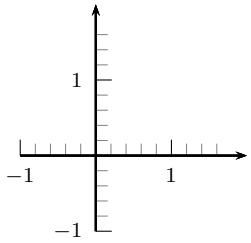

```
\psset{unit=1cm}
\begin{pspicture}(-6.5,-7)(6.5,7.5)
\psaxes[trigLabelBase=2,dx=\psPiH,xunit=\psPi,xtrigLabels,ytrigLabels]
  {->}(0,0)(-1.7,-6.5)(1.77,6.5)[$x$,0][$y$,90]
\end{pspicture}
```

9.15. Option ticks

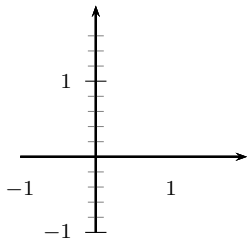
Syntax:

```
ticks=all|x|y|none
```

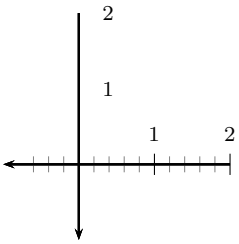
This option was already in the pst-plot package and only mentioned here for some completeness.



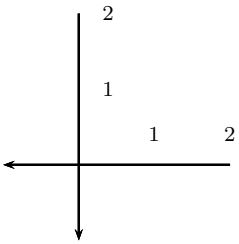

```
\psset{ticksize=6pt}
\begin{pspicture}(-1,-1)(2,2)
\psaxes[ticks=all,subticks=5]{->}(0,0)(-1,-1)(2,2)
\end{pspicture}
```




```
\begin{pspicture}(-1,-1)(2,2)
\psaxes[ticks=y,subticks=5]{->}(0,0)(-1,-1)(2,2)
\end{pspicture}
```




```
\begin{pspicture}(-1,-1)(2,2)
\psaxes[ticks=x,subticks=5]{->}(0,0)(2,2)(-1,-1)
\end{pspicture}
```



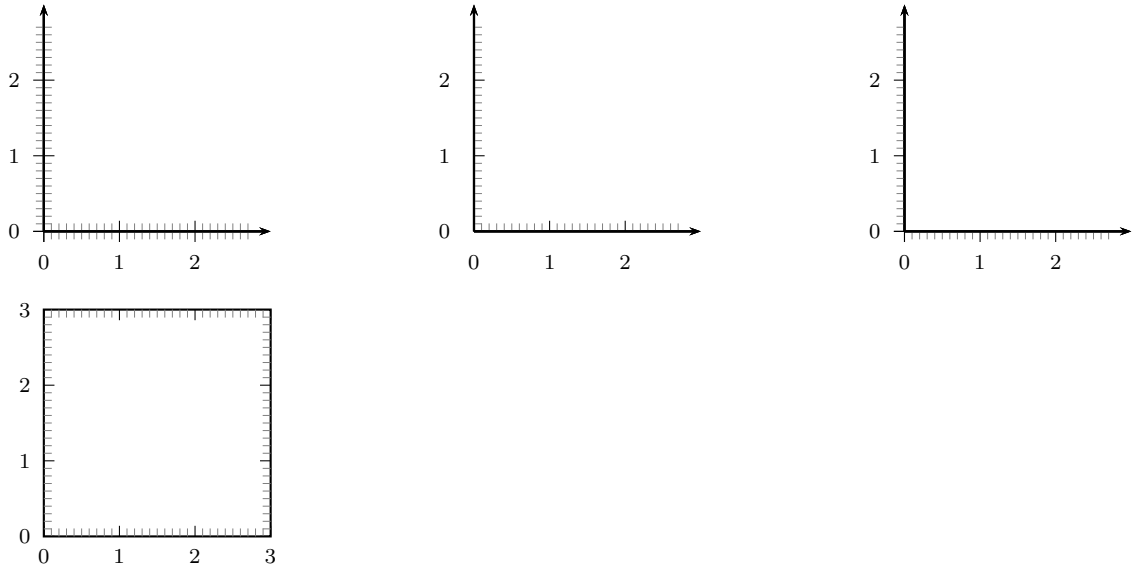

```
\begin{pspicture}(-1,-1)(2,2)
\psaxes[ticks=none,subticks=5]{->}(0,0)(2,2)(-1,-1)
\end{pspicture}
```

9.16. Option tickstyle

Syntax:

```
tickstyle=full|top|bottom|inner
```

The value inner is only possible for the axes style frame.




```
\psset{subticks=10}
\begin{pspicture}(-1,-1)(3,3) \psaxes[tickstyle=full]{->}(3,3) \end{pspicture}
\begin{pspicture}(-1,-1)(3,3) \psaxes[tickstyle=top]{->}(3,3) \end{pspicture}
\begin{pspicture}(-1,-1)(3,3) \psaxes[tickstyle=bottom]{->}(3,3)\end{pspicture}
\begin{pspicture}(-1,-1)(3,3)
  \psaxes[axesstyle=frame, tickstyle=inner, ticksize=0 4pt]{->}(3,3)
\end{pspicture}
```

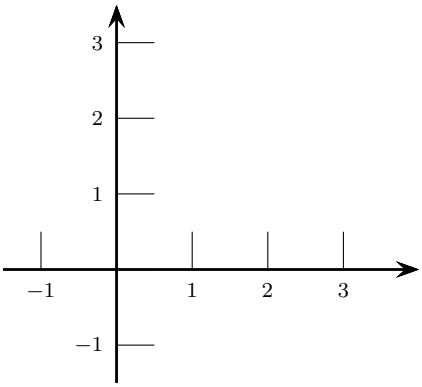
9.17. Options ticksize, xticksize, yticksize

With this new option the recent tickstyle option of pst-plot is obsolete and no longer supported by pstricks-add.

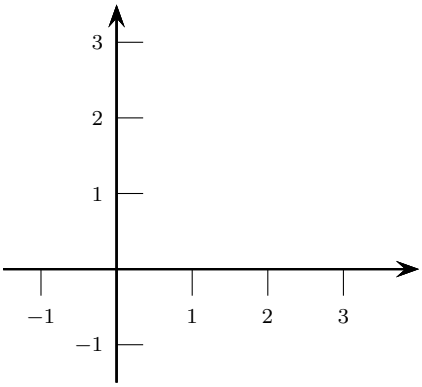
Syntax:

```
ticksize=value[unit]
ticksize=value[unit] value[unit]
xticksize=value[unit]
xticksize=value[unit] value[unit]
yticksize=value[unit]
yticksize=value[unit] value[unit]
```

ticksize sets both values. The first one is left/below and the optional second one is right/above of the coordinate axis. The old setting tickstyle=bottom is now easy to realize, e. g.: ticksize=-6pt 0, or vice versa, if the coordinates are set from positive to negative values.

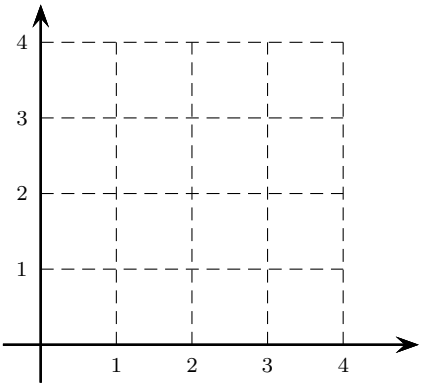



```
\psset{arrowscale=2}
\begin{pspicture}(-1.5,-1.5)(4,3.5)
 \psaxes[ticksize=0.5cm]{->}(0,0)(-1.5,-1.5)
     (4,3.5)
\end{pspicture}
```




```
\psset{arrowscale=2}
\begin{pspicture}(-1.5,-1.5)(4,3.5)
 \psaxes[xticksize=-10pt 0,yticksize=0 10pt]%
   {->}(0,0)(-1.5,-1.5)(4,3.5)
\end{pspicture}
```

A grid is also possible by setting the values to the max/min coordinates.



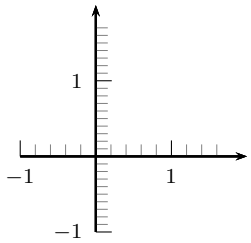

```
\psset{arrowscale=2}
\begin{pspicture}(-.5,-.5)(5,4.5)
 \psaxes[ticklinestyle=dashed,
   ticksize=0 4cm]{->}(0,0)(-.5,-.5)(5,4.5)
\end{pspicture}
```

9.18. Options subticks, xsubticks, and ysubticks

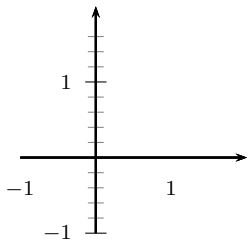
Syntax:

```
subticks=<number>
xsubticks=<number>
ysubticks=<number>
```

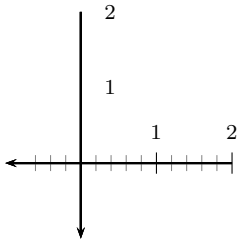
By default `subticks` cannot have labels.



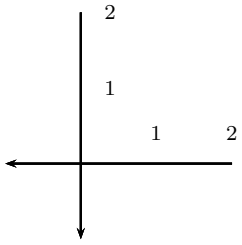

```
\psset{ticksize=6pt}
\begin{pspicture}(-1,-1)(2,2)
\psaxes[ticks=all,xsubticks=5,
  ysubticks=10]{->}(0,0)(-1,-1)(2,2)
\end{pspicture}
```




```
\begin{pspicture}(-1,-1)(2,2)
\psaxes[ticks=y,subticks=5]{->}(0,0)(-1,-1)(2,2)
\end{pspicture}
```




```
\begin{pspicture}(-1,-1)(2,2)
\psaxes[ticks=x,subticks=5]{->}(0,0)(2,2)(-1,-1)
\end{pspicture}
```



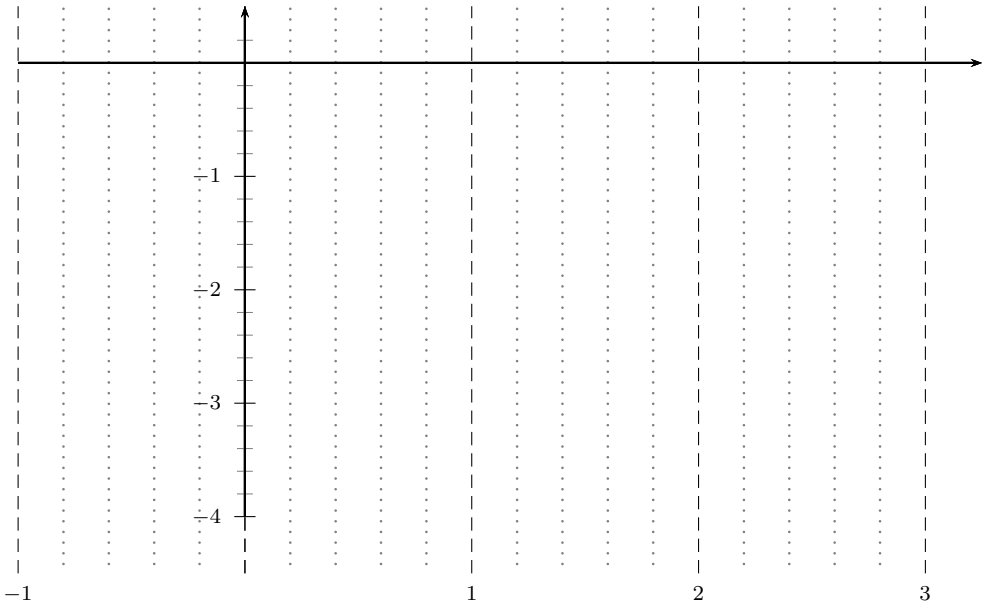

```
\begin{pspicture}(-1,-1)(2,2)
\psaxes[ticks=none,subticks=5]{->}(0,0)(2,2)(-1,-1)
\end{pspicture}
```

9.19. Options subticksize, xsubticksize, ysubticksize

`subticksize` sets both values, `xsubticksize` only for the x-axis and `ysubticksize` only for the y-axis, which must be relative to the ticksize length and can have any number. 1 sets it to the same length as the main ticks.

Syntax:

```
subticksize=value
xsubticksize=value
ysubticksize=value
```



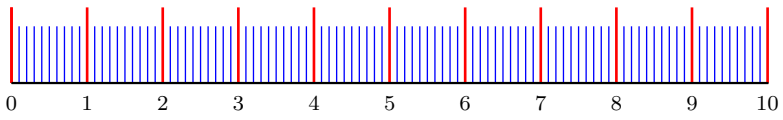

```
\psset{yunit=1.5cm,xunit=3cm}
\begin{pspicture}(-1.25,-4.75)(3.25,.75)
  \psaxes[xticksize=-4.5 0.5,ticklinestyle=dashed,subticks=5,xsubticksize=1,%
    ysubticksize=0.75,xsubticklinestyle=dotted,xsubtickwidth=1pt,
    subtickcolor=gray]{->}(0,0)(-1,-4)(3.25,0.5)
\end{pspicture}
```

9.20. Options tickcolor, xtickcolor, ytickcolor, subtickcolor, xsubtickcolor, and ysubtickcolor

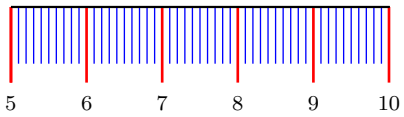
Syntax:

```
tickcolor=<color>
xtickcolor=<color>
ytickcolor=<color>
subtickcolor=<color>
xsubtickcolor=<color>
ysubtickcolor=<color>
```

`tickcolor` and `subtickcolor` set both for the x- and the y-Axis.




```
\begin{pspicture}(0,-0.75)(10,1)
\psaxes[yAxis=false,labelFontSize=\scriptstyle,ticksize=0 10mm,subticks=10,
    subticksize=0.75,
  tickcolor=red,subtickcolor=blue,tickwidth=1pt,subtickwidth=0.5pt](10.01,0)
\end{pspicture}
```



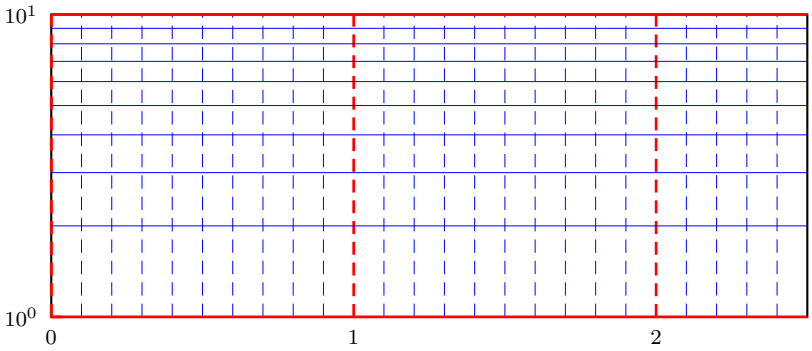

```
\begin{pspicture}(5,-0.75)(10,1)
\psaxes[yAxis=false,labelFontSize=\scriptstyle,
    ticksize=0 -10mm,subticks=10,subticksize=0.75,
  tickcolor=red,subtickcolor=blue,tickwidth=1pt,
     subtickwidth=0.5pt,Ox=5](5,0)(5,0)(10.01,0)
\end{pspicture}
```

9.21. Options ticklinestyle, xticklinestyle, yticklinestyle, subticklinestyle, xsubticklinestyle, and ysubticklinestyle

Syntax:

```
ticklinestyle=solid|dashed|dotted|none
xticklinestyle=solid|dashed|dotted|none
yticklinestyle=solid|dashed|dotted|none
subticklinestyle=solid|dashed|dotted|none
xsubticklinestyle=solid|dashed|dotted|none
ysubticklinestyle=solid|dashed|dotted|none
```

`ticklinestyle` and `subticklinestyle` set both values for the x and y axis. The value `none` doesn't really makes sense, because it is the same as `[sub]ticklines=0`



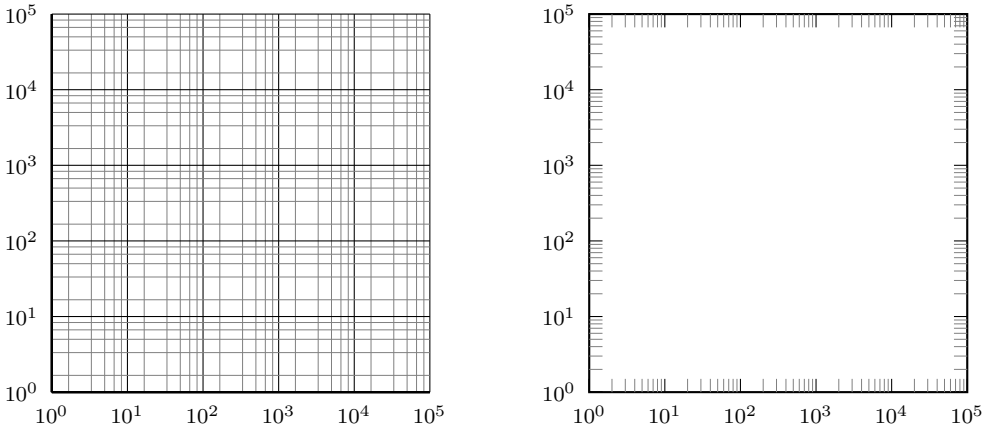

```
\psset{unit=4cm}
\pspicture(-0.15,-0.15)(2.5,1)
  \psaxes[axesstyle=frame,logLines=y,xticksize=0 1,xsubticksize=1,ylogBase=10,
    tickcolor=red,subtickcolor=blue,tickwidth=1pt,subticks=9,xsubticks=10,
    xticklinestyle=dashed,xsubticklinestyle=dashed](2.5,1)
\endpspicture
```

9.22. logLines

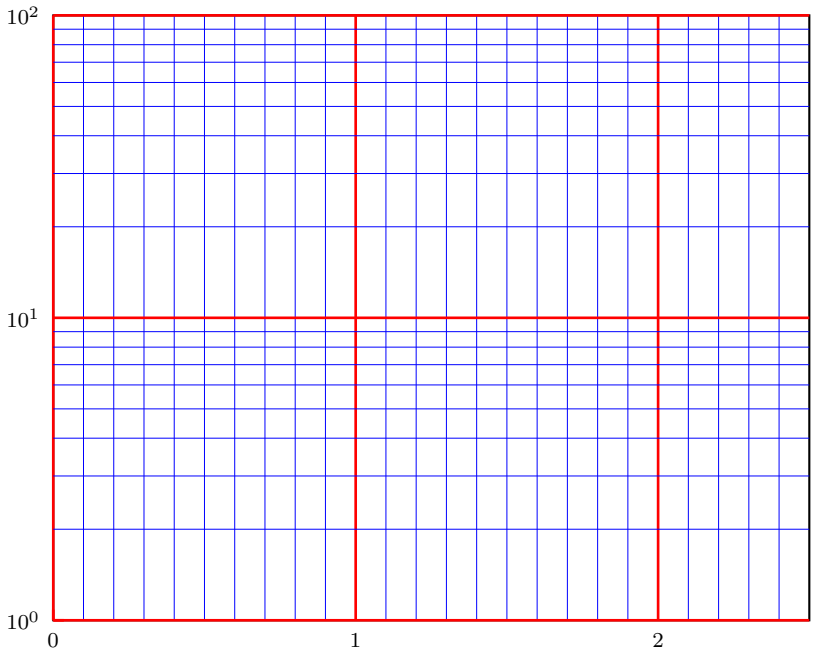
Syntax:

```
logLines=all|x|y
```

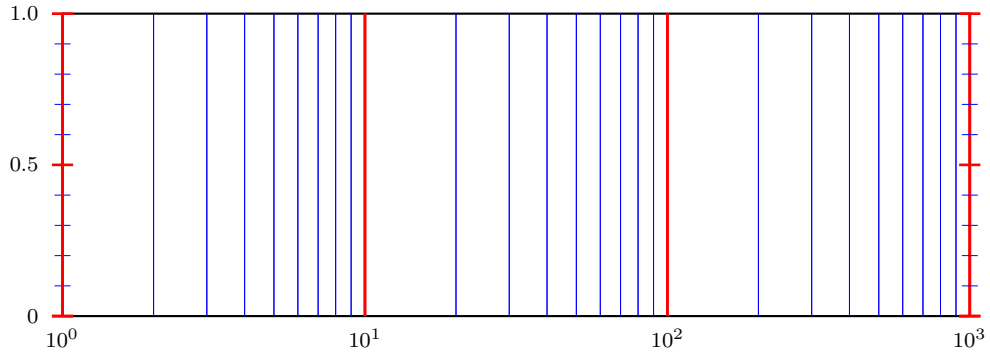
By default the option `logLines` sets the ticksize to the maximal length for x, y, or both. It can be changed, when *after* the option `logLines` the ticksize is set.




```
\pspicture(-1,-1)(5,5)
  \psaxes[subticks=5,xylogBase=10,logLines=all](5,5)
\endpspicture\hspace{1cm}
\pspicture(-1,-1)(5,5)
  \psaxes[subticks=9,axesstyle=frame,xylogBase=10,logLines=all,
    ticksize=0 5pt,tickstyle=inner](5,5)
\endpspicture
```




```
\psset{unit=4cm}
\pspicture(-0.15,-0.15)(2.5,2)
  \psaxes[axesstyle=frame,logLines=y,xticksize=max,xsubticksize=1,ylogBase=10,
    tickcolor=red,subtickcolor=blue,tickwidth=1pt,subticks=9,xsubticks=10](2.5,2)
\endpspicture
```



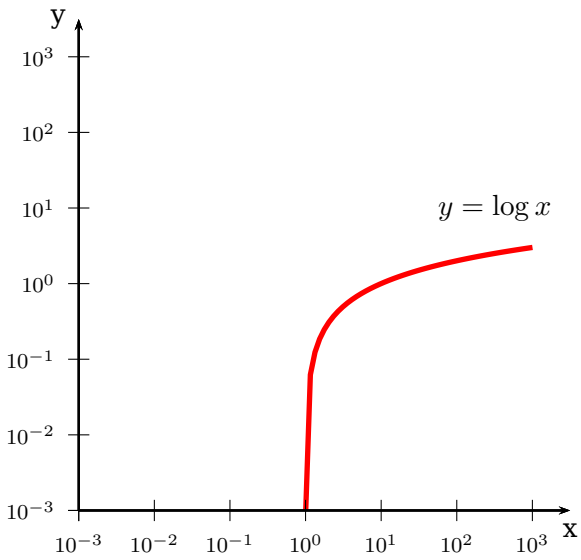

```
\psset{unit=4}
\pspicture(-0.5,-0.3)(3,1.2)
  \psaxes[axesstyle=frame,tickstyle=inner,logLines=x,xlogBase=10,Dy=0.5,
      tickcolor=red,
    subtickcolor=blue,tickwidth=1pt,ysubticks=5,xsubticks=9](3,1)
\endpspicture
```

9.23. xylogBase, xlogBase and ylogBase

There are additional options `xylogBase`, `xlogBase`, `ylogBase` to get one or both axes with logarithmic labels. For an interval of $[10^{-3}...10^{2}]$ choose a `PSTricks`interval of [-3,2]. `PSTricks`takes 0 as the origin of this axes, which is wrong if we want to have a logarithmic axes. With the options `Oy` and `Ox` we can set the origin to -3, so that the first label gets 10^{-3}. If this is not done by the user then `pst-plot` does it by default. An alternative is to set these parameters to empty values `Ox={},Oy={}`, in this case the package does nothing.

9.24. xylogBase

This mode in math is also called double logarithmic. It is a combination of the two foregoing modes and the function is now $y = \log x$ and is shown in the following example.




```
\begin{pspicture}(-3.5,-3.5)(3.5,3.5)
  \psplot[linewidth=2pt,linecolor=red
      ]{0.001}{3}{x log}
  \psaxes[xylogBase=10,Oy=-3,Ox
      =-3]{->}(-3,-3)(3.5,3.5)
  \uput[-90](3.5,-3){x}
  \uput[180](-3,3.5){y}
  \rput(2.5,1){$y=\log x$}
\end{pspicture}
```

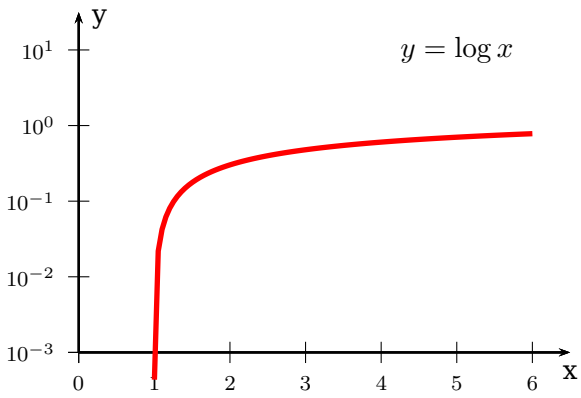
9.25. ylogBase

The values for the \psaxes y-coordinate are now the exponents to the base 10 and for the right function to the base e: $10^{-3} \ldots 10^{1}$ which corresponds to the given y-interval $-3 \ldots 1.5$, where only integers as exponents are possible. These logarithmic labels have no effect on the internally used units. To draw the logarithm function we have to use the math function

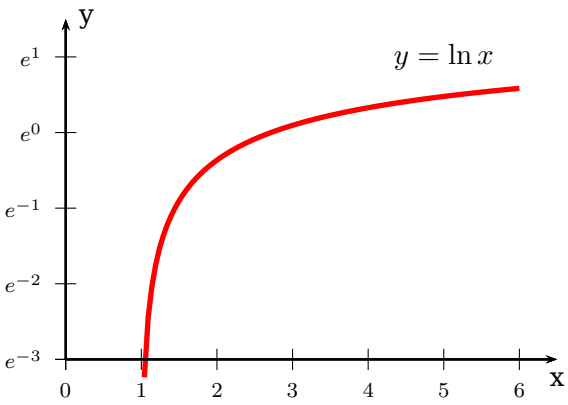
$$y = \log\{\log x\}$$

$$y = \ln\{\ln x\}$$

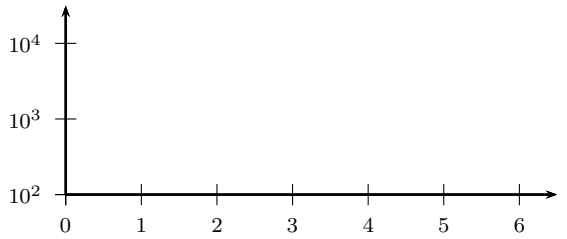
with an drawing interval of $1.001 \ldots 6$.



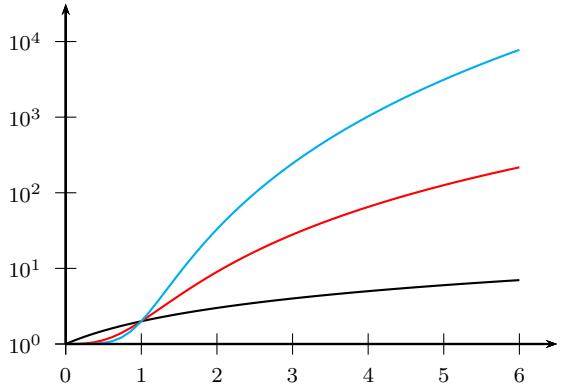

```
\begin{pspicture}(-0.5,-3.5)(6.5,1.5)
  \psaxes[ylogBase=10,Oy=-3]{->}(0,-3)
      (6.5,1.5)
  \uput[-90](6.5,-3){x}
  \uput[0](0,1.4){y}
  \rput(5,1){$y=\log x$}
  \psplot[linewidth=2pt,%
  plotpoints=100,linecolor=red]{1.001}{6}{
      x log log} % log(log(x))
\end{pspicture}
```



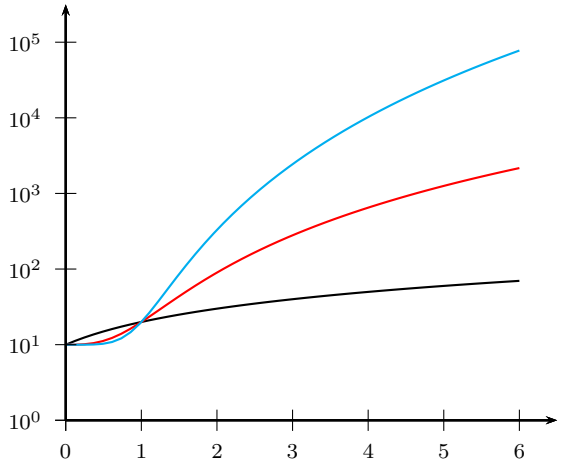

```
\begin{pspicture}(-0.5,-3.5)(6.5,1.5)
 \psplot[linewidth=2pt,plotpoints=100,
    linecolor=red]%
  {1.04}{6}[ /ln {log 0.4343 div} def ]{x
      ln ln} % log(x)
 \psaxes[ylogBase=e,Oy=-3]{->}(0,-3)
    (6.5,1.5)
 \uput[-90](6.5,-3){x}
 \uput[0](0,1.5){y}
 \rput(5,1){$y=\ln x$}
\end{pspicture}
```



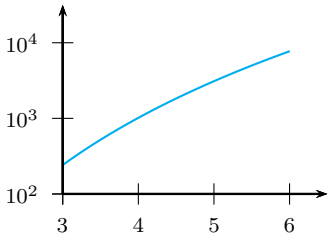

```
 \begin{pspicture}(-0.5,1.75)(6.5,4.5)
  \psaxes[ylogBase=10,Oy=2]{->}(0,2)(0,2)
     (6.5,4.5)
 \end{pspicture}
```




```
 \begin{pspicture}(-0.5,-0.25)(6.5,4.5)
  \psplot{0}{6}{x x cos add log}
              % x  + cox(x)
  \psplot[linecolor=red]{0}{6}{x 3 exp x
     cos add log} % x^3 + cos(x)
  \psplot[linecolor=cyan]{0}{6}{x 5 exp x
      cos add log} % x^5 + cos(x)
  \psaxes[ylogBase=10]{->}(6.5,4.5)
 \end{pspicture}
```



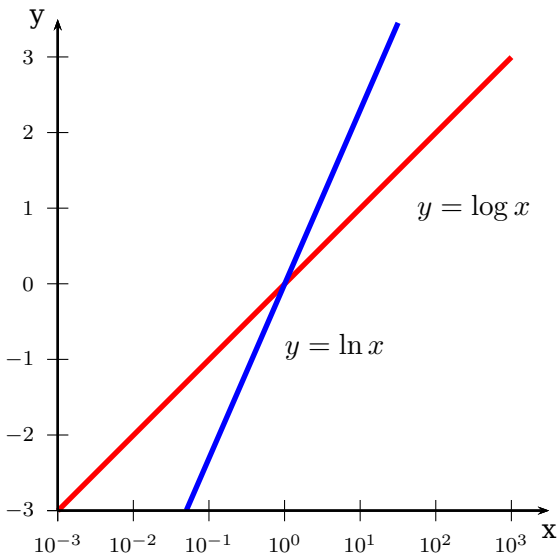

```
\begin{pspicture}(-0.5,-1.25)(6.5,4.5)
 \psplot{0}{6}{x x cos add log}          %
     x  + cox(x)
 \psplot[linecolor=red]{0}{6}{x 3 exp x
    cos add log} % x^3 + cos(x)
 \psplot[linecolor=cyan]{0}{6}{x 5 exp x
    cos add log} % x^5 + cos(x)
 \psaxes[ylogBase=10]{->}(0,-1)(0,-1)
    (6.5,4.5)
\end{pspicture}
```



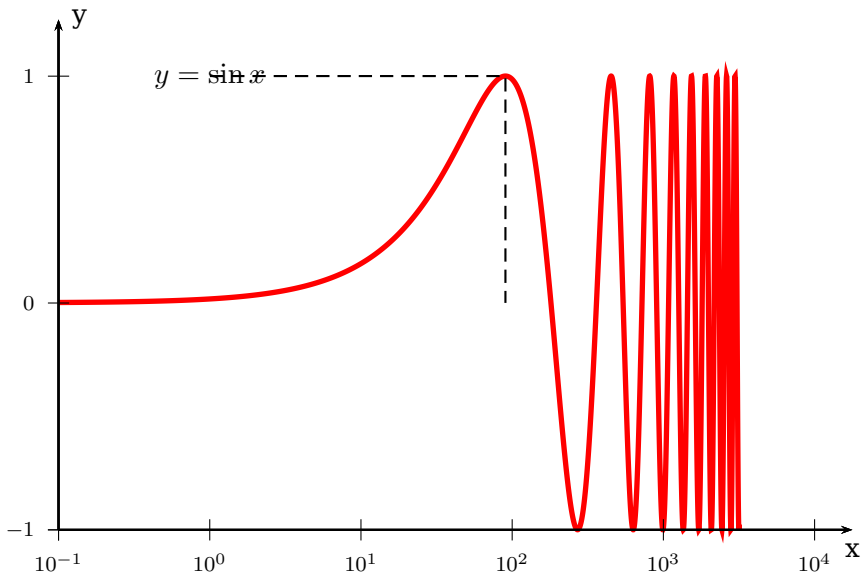

```
\begin{pspicture}(2.5,1.75)(6.5,4.5)
  \psplot[linecolor=cyan]{3}{6}{x 5 exp x cos add log} % x
      ^5 + cos(x)
  \psaxes[ylogBase=10,Ox=3,Oy=2]{->}(3,2)(3,2)(6.5,4.5)
\end{pspicture}
```

9.26. xlogBase

Now we have to use the easy math function $y = x$ because the x axis is still $\log x$.

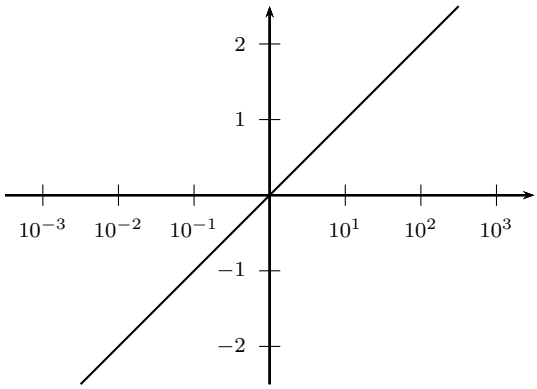



```
\begin{pspicture}(-3.5,-3.5)(3.5,3.5)
  \psplot[linewidth=2pt,linecolor=red
      ]{-3}{3}{x} % log(x)
  \psplot[linewidth=2pt,linecolor=blue
      ]{-1.3}{1.5}{x 0.4343 div} % ln(x)
  \psaxes[xlogBase=10,Oy=-3,Ox
      =-3]{->}(-3,-3)(3.5,3.5)
  \uput[-90](3.5,-3){x}
  \uput[180](-3,3.5){y}
  \rput(2.5,1){$y=\log x$}
  \rput[lb](0,-1){$y=\ln x$}
\end{pspicture}
```

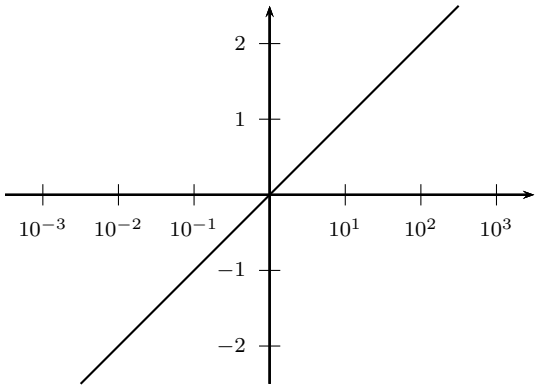



```
\psset{yunit=3cm,xunit=2cm}
\begin{pspicture}(-1.25,-1.25)(4.25,1.5)
  \uput[-90](4.25,-1){x}
```

```
  \uput[0](-1,1.25){y}
  \rput(0,1){$y=\sin x$}
  \psplot[linewidth=2pt,plotpoints=5000,linecolor=red]{-1}{3.5}{10 x exp sin }
  \psaxes[xlogBase=10,Ox=-1,Oy=-1]{->}(-1,-1)(4.25,1.25)
   \psline[linestyle=dashed](-1,0)(4,0)
   \psline[linestyle=dashed](!-1 1)(!90 log 1)(!90 log -1)
   \psline[linestyle=dashed](!90 log 1)(!180 log 1)(!180 log -1)
\end{pspicture}
```



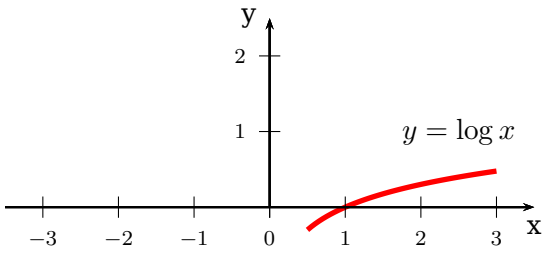

```
\begin{pspicture}(-3.5,-2.5)(3.5,2.5)
  \psaxes[xlogBase=10]{->}(0,0)(-3.5,-2.5)
      (3.5,2.5)
  \psplot{-2.5}{2.5}{10 x exp log}
\end{pspicture}
```




```
\begin{pspicture}(-3.5,-2.5)(3.5,2.5)
  \psaxes[xlogBase=10,Ox={},Oy
      ={}]{->}(0,0)(-3.5,-2.5)(3.5,2.5)
  \psplot{-2.5}{2.5}{10 x exp log}
\end{pspicture}
```

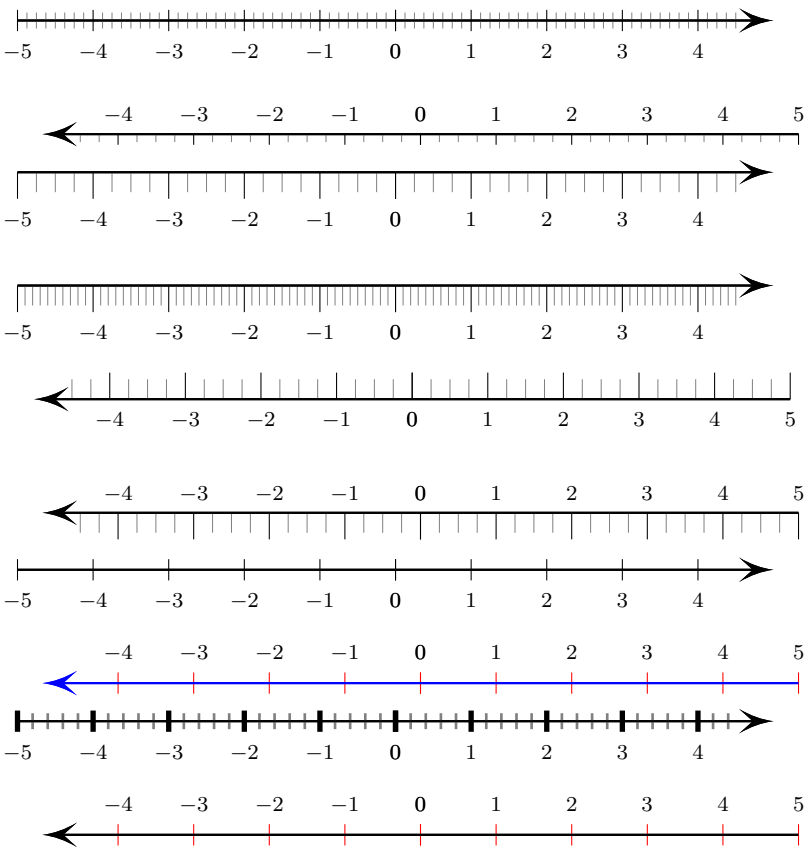
9.27. No logstyle (xylogBase={})

This is only a demonstration that the default option ={} still works ... :-)

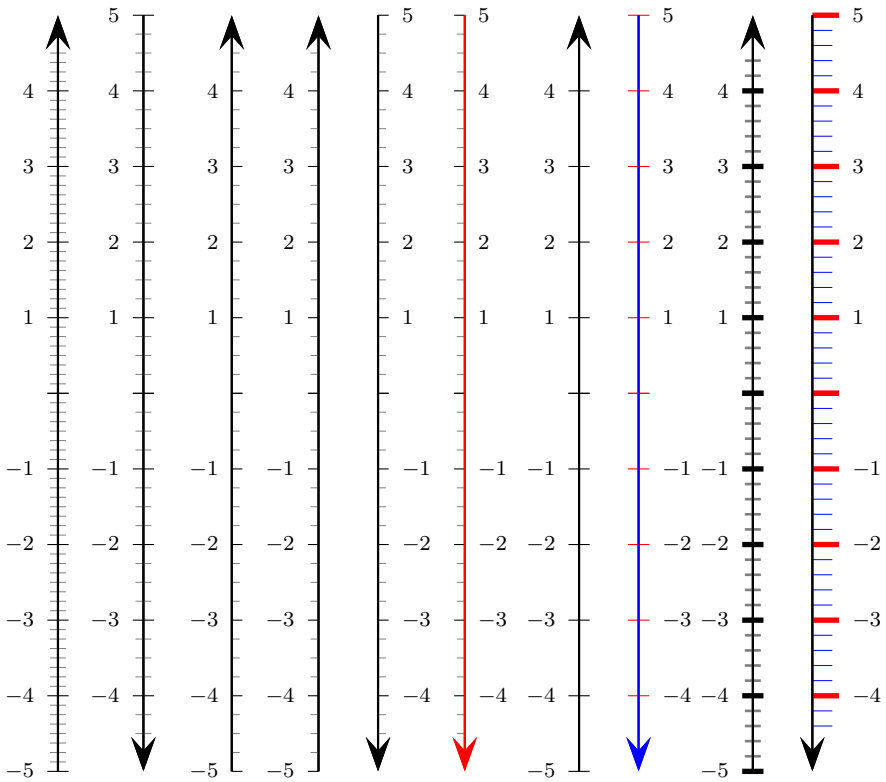



```
\begin{pspicture}(-3.5,-0.5)(3.5,2.5)
  \psplot[linewidth=2pt,linecolor=red,
      xylogBase={}]{0.5}{3}{x log} % log(x)
  \psaxes{->}(0,0)(-3.5,0)(3.5,2.5)
  \uput[-90](3.5,0){x}
  \uput[180](0,2.5){y}
  \rput(2.5,1){$y=\log x$}
\end{pspicture}
```

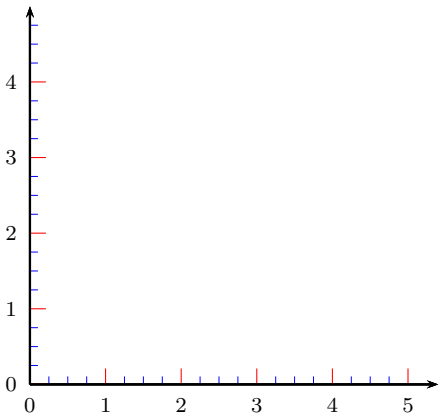
9.28. Option tickwidth and subtickwidth



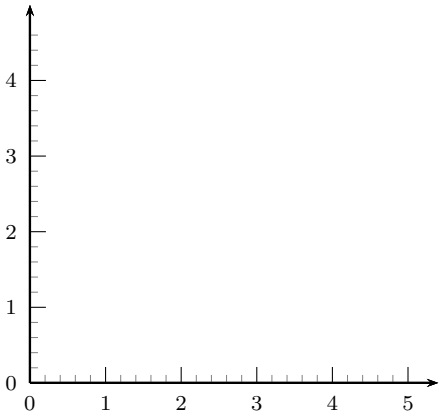

```
\psset{arrowscale=3,arrows=-D>,yAxis=false}
 \psaxes[subticks=8](0,0)(-5,-1)(5,1)\\[1cm]
 \psaxes[subticks=4,ticksize=-4pt 0,xlabelPos=top](0,0)(5,1)(-5,-1)\\
 \psaxes[subticks=4,ticksize=-10pt 0](0,0)(-5,-5)(5,5)\\[1cm]
 \psaxes[subticks=10,ticksize=0 -10pt](0,0)(-5,-5)(5,5)\\[1cm]
 \psaxes[subticks=4,ticksize=0 10pt,xlabelPos=bottom](0,0)(5,5)(-5,-5)\\[1cm]
 \psaxes[subticks=4,ticksize=0 -10pt,xlabelPos=top](0,0)(5,5)(-5,-5)\\[0.25cm]
 \psaxes[subticks=0](0,0)(-5,-5)(5,5)\\[1cm]
 \psaxes[subticks=0,tickcolor=red,linecolor=blue,xlabelPos=top](0,0)(5,5)(-5,-5)\\
 \psaxes[subticks=5,tickwidth=2pt,subtickwidth=1pt](0,0)(-5,-5)(5,5)\\[1cm]
 \psaxes[subticks=0,tickcolor=red,xlabelPos=top](0,0)(5,5)(-5,-5)}
```



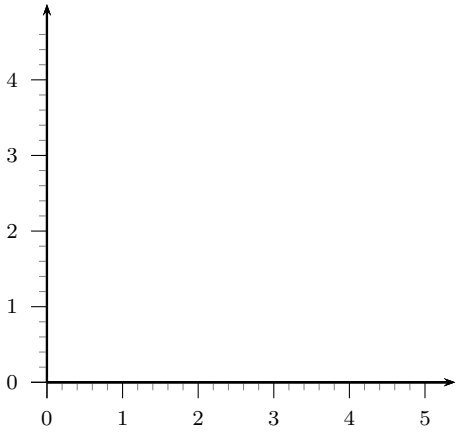
```
\psset{arrowscale=3,xAxis=false}
\psaxes[subticks=8]{->}(0,0)(-5,-5)(5,5)\hspace{2em}
\psaxes[subticks=4,ylabelPos=right,ylabelPos=left]{->}(0,0)(5,5)(-5,-5)\hspace{4em}
\psaxes[subticks=4,ticksize=0 4pt]{->}(0,0)(-5,-5)(5,5)\hspace{3em}
\psaxes[subticks=4,ticksize=-4pt 0]{->}(0,0)(-5,-5)(5,5)\hspace{1em}
\psaxes[subticks=4,ticksize=0 4pt,ylabelPos=right]{->}(0,0)(5,5)(-5,-5)\hspace{3em}
\psaxes[subticks=4,ticksize=-4pt 0,linecolor=red,ylabelPos=right]{->}(0,0)(5,5)(-5,-5)\
    hspace{5em}
\psaxes[subticks=0]{->}(0,0)(-5,-5)(5,5)\hspace{1em}
\psaxes[subticks=0,tickcolor=red,linecolor=blue,ylabelPos=right]{->}(0,0)(5,5)(-5,-5)\
    hspace{5em}
\psaxes[subticks=5,tickwidth=2pt,subtickwidth=1pt]{->}(0,0)(-5,-5)(5,5)\hspace{1em}
\psaxes[subticks=5,tickcolor=red,tickwidth=2pt,%
 ticksize=10pt,subtickcolor=blue,subticksize=0.75,ylabelPos=right]{->}(0,0)(5,5)(-5,-5)
```



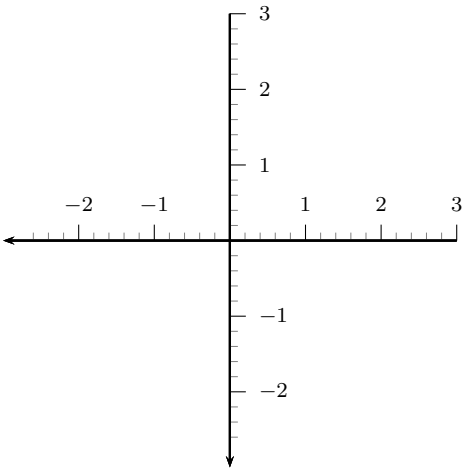
```
\pspicture(5,5.5)
\psaxes[subticks=4,ticksize=6pt,subticksize=0.5,%
  tickcolor=red,subtickcolor=blue]{->}(5.4,5)
\endpspicture
```



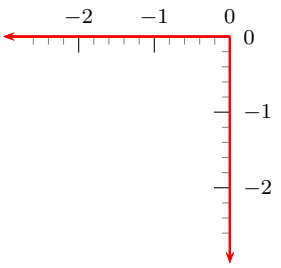

```
\pspicture(5,5.5)
  \psaxes[subticks=5,ticksize=0 6pt,subticksize
     =0.5]{->}(5.4,5)
\endpspicture
```



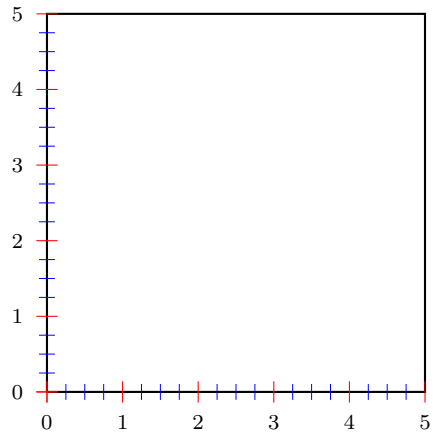

```
\pspicture(5,5.5)
  \psaxes[subticks=5,ticksize=-6pt 0,subticksize
     =0.5]{->}(5.4,5)
\endpspicture
```



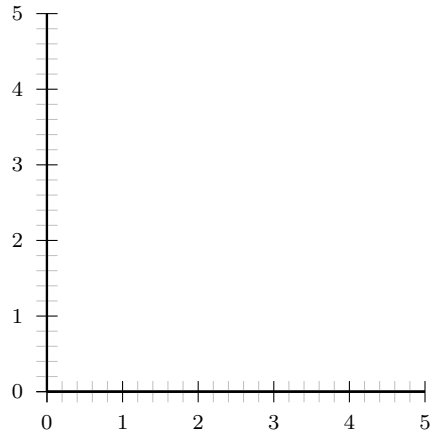

```
\pspicture(-3,-3)(3,3.5)
  \psaxes[subticks=5,ticksize=0 6pt,
     subticksize=0.5]{->}(0,0)(3,3)(-3,-3)
\endpspicture
```



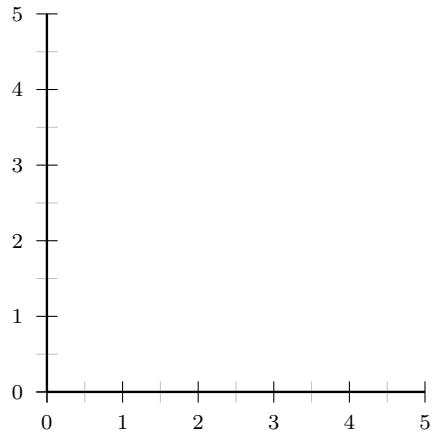

```
\pspicture(0,0.5)(-3,-3)
  \psaxes[subticks=5,ticksize=-6pt 0,
     subticksize=0.5,linecolor=red
     ]{->}(-3,-3)
\endpspicture
```



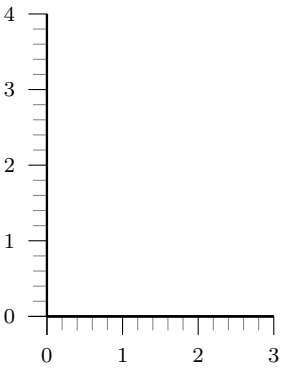

```
\psset{axesstyle=frame}
\pspicture(5,5.5)
  \psaxes[subticks=4,tickcolor=red,subtickcolor=
     blue](5,5)
\endpspicture
```



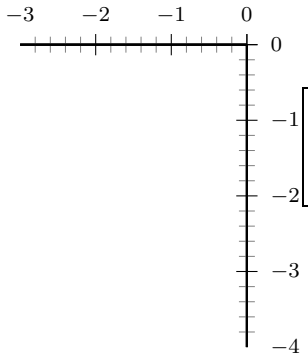

```
\pspicture(5,5.5)
  \psaxes[subticks=5,subticksize=1,subtickcolor=
     lightgray](5,5)
\endpspicture
```



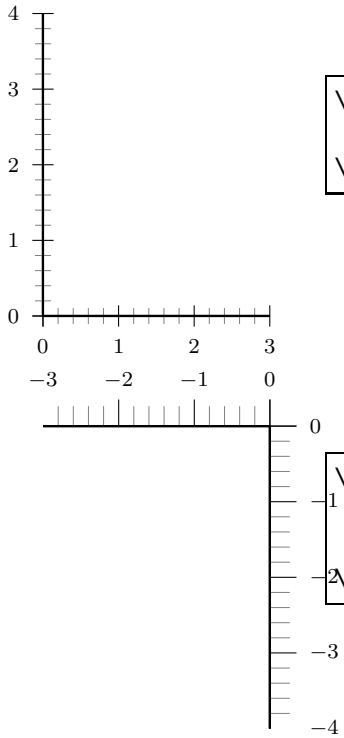

```
\pspicture(5,5.5)
  \psaxes[subticks=2,subticksize=1,subtickcolor=
     lightgray](5,5)
\endpspicture
```




```
\pspicture(3,4.5)
  \psaxes[subticks=5,ticksize=-7pt 0](3,4)
\endpspicture
```




```
\pspicture(0,1)(-3,-4)
  \psaxes[subticks=5](-3,-4)
\endpspicture
```



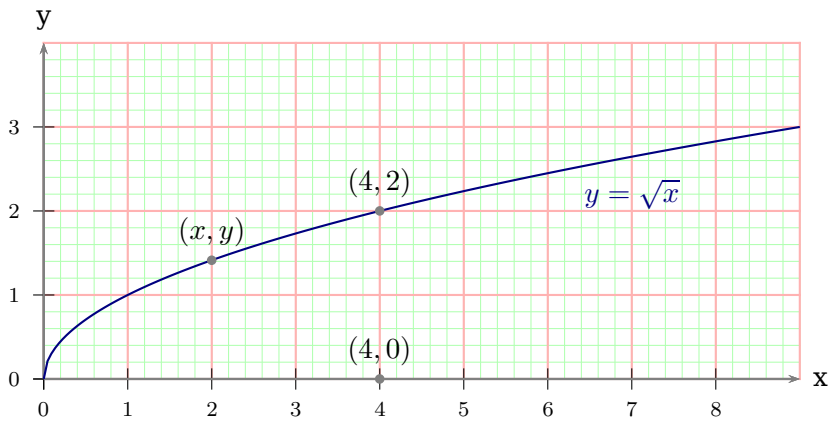

```
\pspicture(3,4.5)
  \psaxes[axesstyle=axes,subticks=5](3,4)
\endpspicture
```

```
\pspicture(0,1)(-3,-4)
  \psaxes[axesstyle=axes,subticks=5,%
    ticksize=0 10pt](-3,-4)
\endpspicture
```

9.29. Option psgrid, gridcoor, and gridpara

A simple grid can be set with the optional argument `psgrid` which uses the setting of `gridpara` and `gridcoor`. `gridpara` is preset to

```
\gridpara={gridlabels=0pt,gridcolor=red!30,subgridcolor=green!30,
  subgridwidth=0.5\pslinewidth,subgriddiv=5},...
```




```
\usepackage{pst-plot}
\psset{llx=-5mm,lly=-5mm,urx=5mm,ury=5mm,labelFontSize=\scriptstyle,
   algebraic,plotpoints=200,psgrid,gridcoor={(0,0)(9,4)}}
\begin{psgraph}[linecolor=gray]{->}(0,0)(9,4){10cm}{!}
\psplot[linecolor=NavyBlue]{0}{9}{sqrt(x)}% needs dvipsnames
\psdots(*2 {sqrt(2)})(4,2)(4,0)
\uput[90](*2 {sqrt(2)}){$(x,y)$}\uput[90](4,2){$(4,2)$}\uput[90](4,0){$(4,0)$}
\rput(7,2.2){\textcolor{NavyBlue}{$y=\sqrt{x}$}}
\end{psgraph}
```

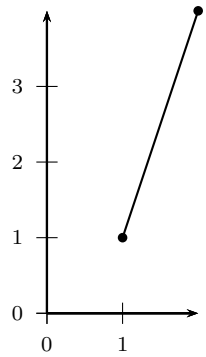
10. New options for \readdata

By default the macro \readdata reads every data record, which could be annoying when you have some text lines at top of your data files or when there are more than 10000 records to read.

pst-plot defines two additional keys ignoreLines and nStep, which allows you to ignore preceeding lines, e. g. ignoreLines=2, or to read only a selected part of the data records, e. g. nStep=10, only every 10^{th} record is saved.

```
\readdata[ignoreLines=2]{\dataA}{stressrawdata.data}
\readdata[nStep=10]{\dataA}{stressrawdata.data}
```

The default value for ignoreLines is 0 and for nStep is 1. the following data file has two text lines which shall be ignored by the \readdata macro:



```
\begin{filecontents*}{pstricks-add-data9.data}
some nonsense in this line ---time forcex forcey
0 0.2
1 1
2 4
\end{filecontents*}
\readdata[ignoreLines=2]{\data}{pstricks-add-data9.data}
\pspicture(2,4)
  \listplot[showpoints]{\data}
  \psaxes{->}(2,4)
\endpspicture
```

11. New options for \listplot

By default the plot macros \dataplot, \fileplot and \listplot plot every data record. There are noew additional keys nStep, nStart, nEnd, and xStep, xStart, xEnd, which allows to plot only a selected part of the data records, e. g. nStep=10. These "'n"' options mark the number of the record to be plotted $(0, 1, 2, ...)$ and the "'x"' ones the x-values of the data records.

The new options are only available for the \listplot macro, which is not a real limitation, because all data records can be read from a file with the \readdata macro (see example files or [**?**]):

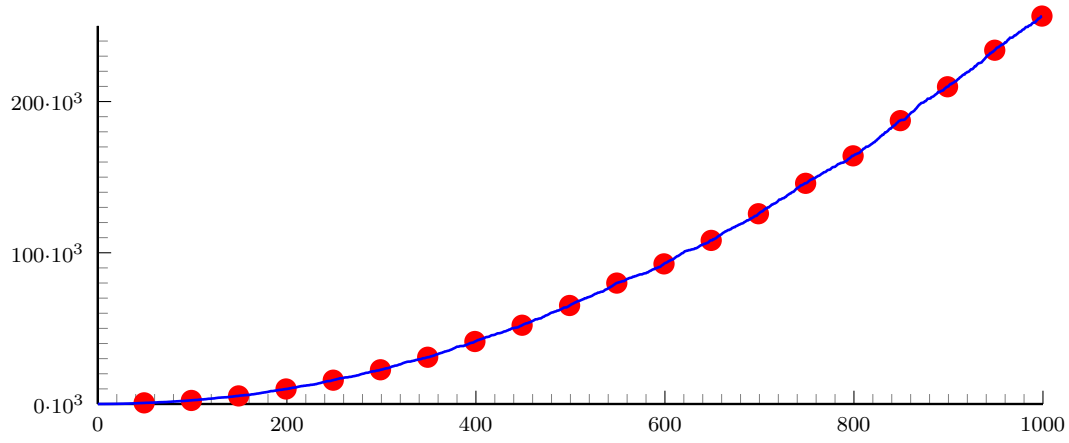
```
\readdata[nStep=10]{\data}{/home/voss/data/data1.data}
```

The use nStep and xStep options only make real sense when also using the option plotstyle=dots . Otherwise the coordinates are connected by a line as usual. Also the xStep option needs increasing x values. Note that nStep can be used for \readdata and for \listplot. If used in both macros then the effect is multiplied, e.g. \readdata with nStep=5 and \listplot with nStep=10 means, that only every 50^{th} data record is read and plotted.

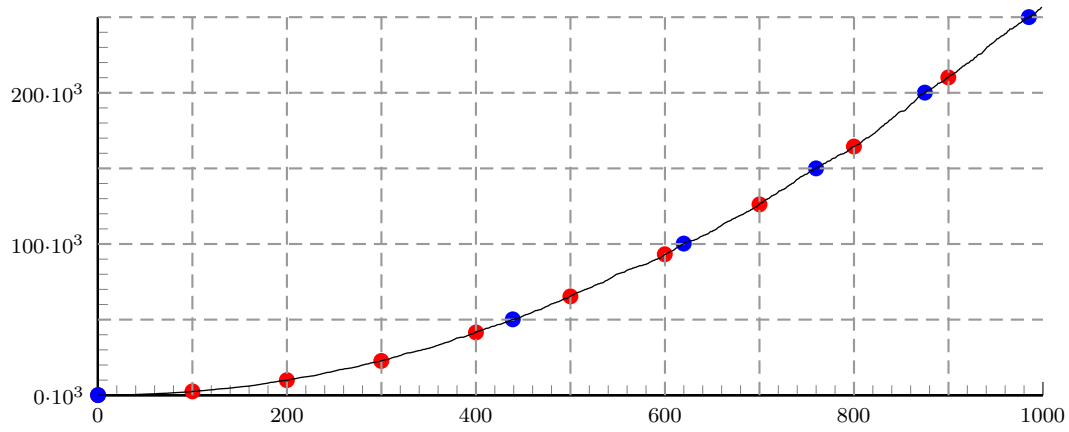
When both, x/yStart/End are defined then the values are also compared with both values.

11.1. Options nStep, xStep, and yStep

The datafile data.data contains 1000 data records. The thin blue line is the plot of all records with the plotstyle option curve.

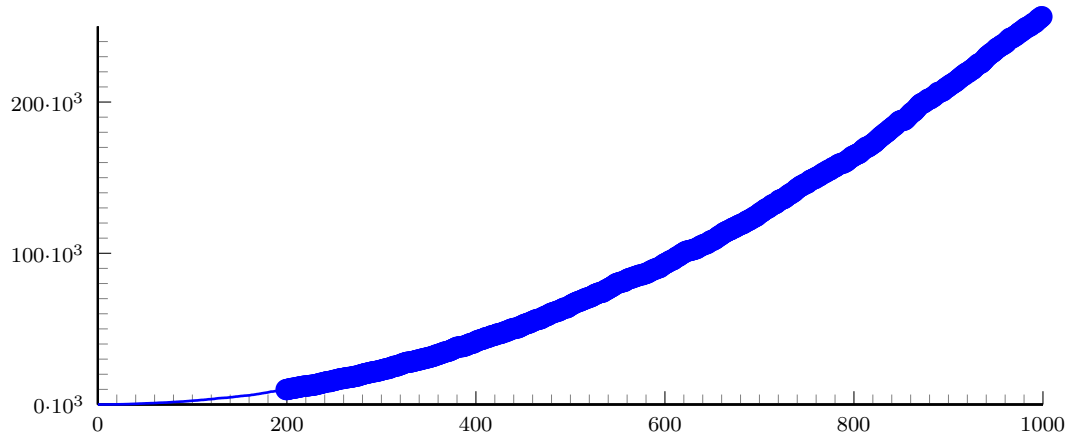



```
\readdata{\data}{data.data}
\psset{xunit=12.5cm,yunit=0.2mm}
\begin{pspicture}(-0.080,-30)(1,270)
\pstScalePoints(1,1){1000 div}{1000 div}
\psaxes[Dx=200,dx=2.5cm,Dy=100,ticksize=0 5pt,tickstyle=inner,
   subticks=10,ylabelFactor=\cdot10^3,dy=2cm](0,0)(1,250)
\listplot[nStep=50,linewidth=3pt,linecolor=red,plotstyle=dots]{\data}
\listplot[linewidth=1pt,linecolor=blue]{\data}
\end{pspicture}
```



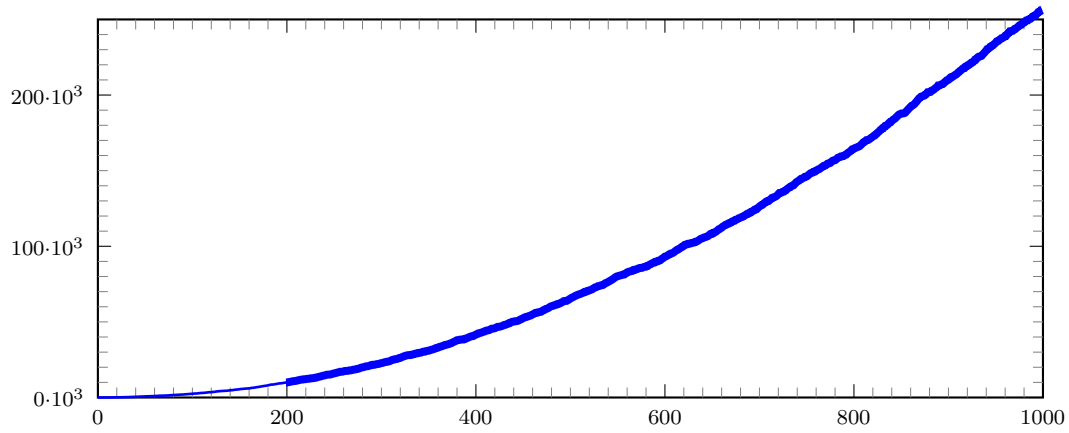

```
\readdata{\data}{data.data}
\psset{xunit=12.5cm,yunit=0.2mm}
\begin{pspicture}(-0.080,-30)(1,270)
\pstScalePoints(1,1){1000 div}{1000 div}
\psaxes[Dx=200,dx=2.5cm,Dy=100,ticksize=0 5pt,tickstyle=inner,
   subticks=10,ylabelFactor=\cdot10^3,dy=2cm](0,0)(1,250)
\listplot[xStep=100,linewidth=2pt,linecolor=red,plotstyle=dots]{\data}
\multido{\rA=0.1+0.1}{9}{%
  \psline[linecolor=black!40,linestyle=dashed](\rA,0)(\rA,250)}
\listplot[yStep=50000,linewidth=2pt,linecolor=blue,plotstyle=dots]{\data}
\multido{\nA=50+50}{5}{%
  \psline[linecolor=black!40,linestyle=dashed](0,\nA)(1,\nA)}
\listplot[linewidth=0.5pt]{\data}
\end{pspicture}
```

11.2. Options nStart and xStart



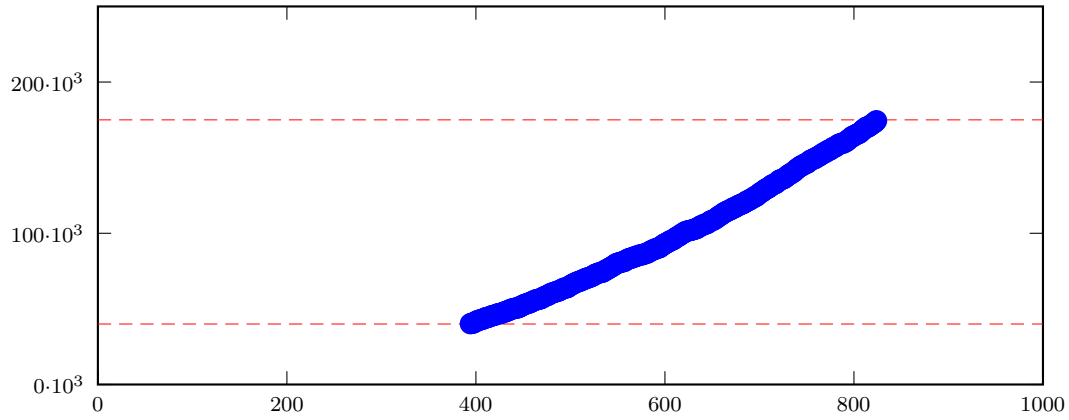

```
\readdata{\data}{data.data}
\psset{xunit=12.5cm,yunit=0.2mm}
\begin{pspicture}(-0.080,-30)(1,270)
\pstScalePoints(1,1){1000 div}{1000 div}
\psaxes[Dx=200,dx=2.5cm,Dy=100,ticksize=0 5pt,tickstyle=inner,
   subticks=10,ylabelFactor=\cdot10^3,dy=2cm](0,0)(1,250)
\listplot[nStart=200,linewidth=3pt,
   linecolor=blue,plotstyle=dots]{\data}
\listplot[linewidth=1pt,linecolor=blue]{\data}
\end{pspicture}
```

11.3. Options nEnd and xEnd




```
\readdata{\data}{data.data}
\psset{xunit=12.5cm,yunit=0.2mm}
\begin{pspicture}(-0.080,-30)(1,270)
\pstScalePoints(1,1){1000 div}{1000 div}
\psaxes[axesstyle=frame,Dx=200,dx=2.5cm,Dy=100,ticksize=0 5pt,tickstyle=inner,
   subticks=10,ylabelFactor=\cdot10^3,dy=2cm](0,0)(1,250)
\listplot[nStart=200,linewidth=3pt,
   linecolor=blue]{\data}
\listplot[linewidth=1pt,linecolor=blue]{\data}
\end{pspicture}
```

11.4. Options yStart and yEnd




```
\readdata{\data}{data.data}
\psset{xunit=12.5cm,yunit=0.2mm}
\begin{pspicture}(-0.080,-30)(1,270)
\pstScalePoints(1,1){1000 div}{1000 div}
\psaxes[axesstyle=frame,Dx=200,dx=2.5cm,Dy=100,ticksize=0 5pt,tickstyle=inner,
   ylabelFactor=\cdot10^3,dy=2cm](0,0)(1,250)
  \psset{linewidth=0.1pt, linestyle=dashed,linecolor=red}
  \psline(0,40)(1,40)
  \psline(0,175)(1,175)
  \listplot[yStart=40000, yEnd=175000,linewidth=3pt,linecolor=blue,plotstyle=dots
     ]{\data}
\end{pspicture}
```

11.5. Options plotNo, plotNoX, and plotNoMax

By default the plot macros expect `x|y` data records, but when having data files with multiple values for y, like:

```
x y1 y2 y3 y4 ... yMax
x y1 y2 y3 y4 ... yMax
...
```

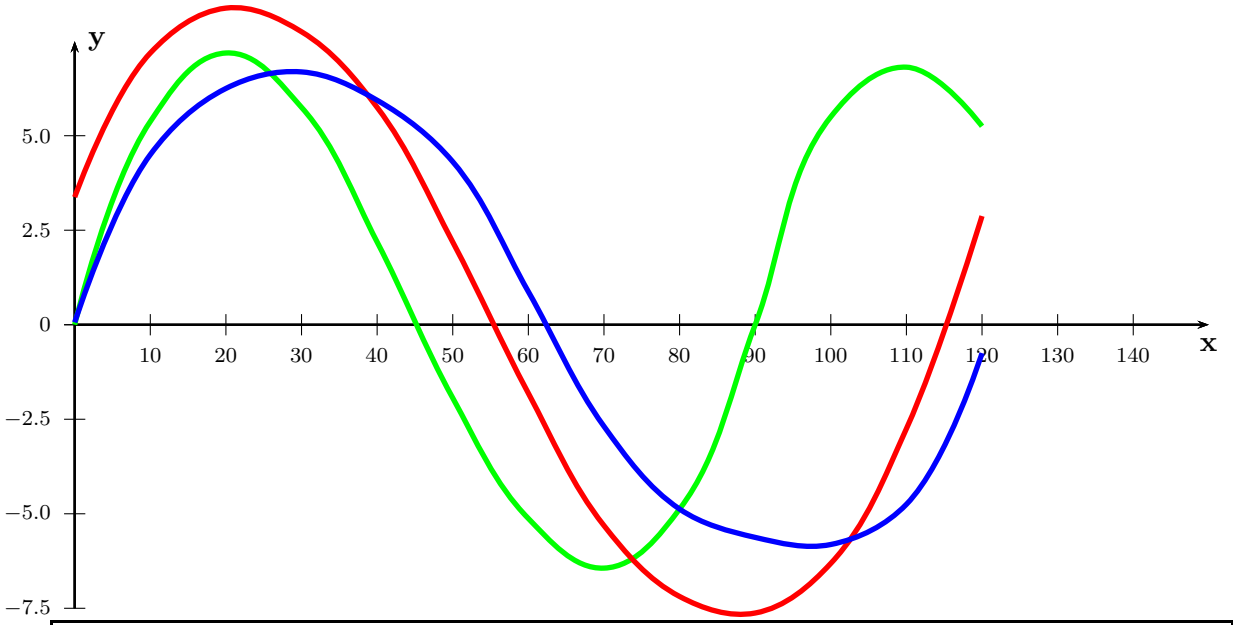
you can select the y value which should be plotted. The option `plotNo` marks the plotted value (default 1) and the option `plotNoMax` tells `pst-plot` how many y values are present. There are no real restrictions in the maximum number for `plotNoMax`.

We have the following data file:

```
[% file data.data
0   0   3.375  0.0625
10   5.375  7.1875  4.5
20   7.1875  8.375  6.25
30   5.75  7.75  6.6875
40   2.1875  5.75  5.9375
50   -1.9375 2.1875  4.3125
60   -5.125  -1.8125 0.875
```

```
70   -6.4375  -5.3125  -2.6875
80   -4.875   -7.1875  -4.875
90   0    -7.625   -5.625
100   5.5   -6.3125  -5.8125
110   6.8125  -2.75   -4.75
120   5.25   2.875   -0.75
]%
```

which holds data records for multiple plots (`x y1 y2 y3`). This can be plotted without any modification to the data file:

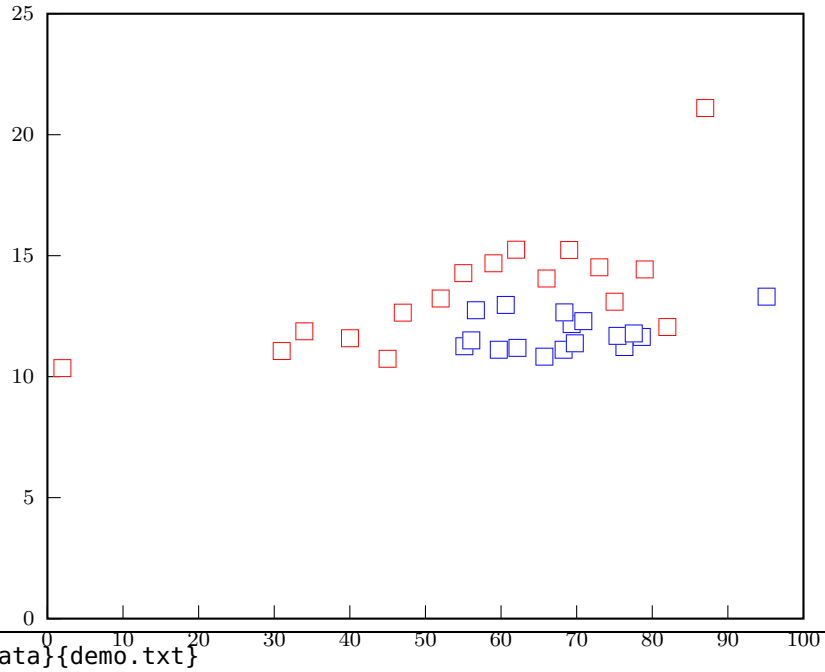



```
\readdata\Data{dataMul.data}
\psset{xunit=0.1cm, yunit=0.5cm,lly=-0.5cm}
\begin{pspicture}(0,-7.5)(150,10)
\psaxes[Dx=10,Dy=2.5]{->}(0,0)(0,-7.5)(150,7.5)[$\mathbf{x}$,-90][$\mathbf{y
    }$,0]
\psset{linewidth=2pt,plotstyle=curve}
\listplot[linecolor=green,plotNo=1,plotNoMax=3]{\Data}
\listplot[linecolor=red,plotNo=2,plotNoMax=3]{\Data}
\listplot[linecolor=blue,plotNo=3,plotNoMax=3]{\Data}
\end{pspicture}
```

It is also possible to select another column for the x-value. Suppose we have a data base with records like `x y y x y`, then it is by default a record with one x value and four possible y values. We still have to define `plotNoMax=4`. However, it is possible to select the forth value as new x value by setting `plotNoX=4` (it is preset to 1). Then the forth value is taken as x. The example uses the the following data set.

```
% X1 X2    Y1    Y2
  2   55.1500 10.35 11.26
 31   59.7167 11.06 11.11
```

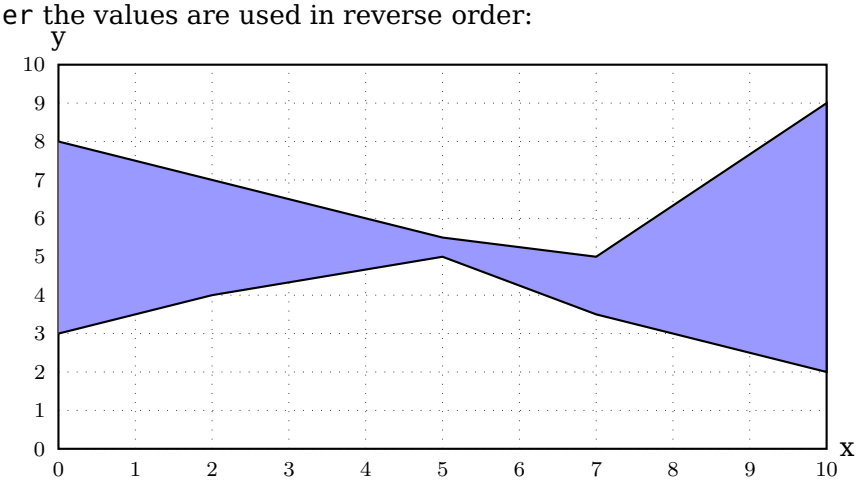
```
34  65.7167 11.87 10.83
40  62.1833 11.59 11.19
45  56.0500 10.74 11.50
47  68.2667 12.65 11.11
52  69.7500 13.23 11.38
55  76.3333 14.28 11.22
59  75.4000 14.69 11.69
62  78.6000 15.25 11.64
66  69.3167 14.06 12.17
69  77.5500 15.24 11.79
73  70.8833 14.52 12.29
75  60.6167 13.10 12.97
79  68.3833 14.43 12.66
82  56.6833 12.05 12.75
87  95.1333 21.10 13.31
```




```
\readdata{\data}{demo.txt}
\psset{xAxisLabel={},yAxisLabel={},llx=-5mm}
\begin{psgraph}[axesstyle=frame,Dy=5,Dx=10,ticksize=5pt 0](0,0)(100,25){10cm}{8
    cm}
\psset{dotstyle=square,dotscale=1.5,linewidth=1.5pt}
\listplot[plotNoMax=3,plotNo=2,linecolor=red,plotstyle=dots]{\data}
\listplot[plotNoMax=3,plotNoX=2,plotNo=3,linecolor=blue,plotstyle=dots]{\data}
\end{psgraph}
```

11.6. Option changeOrder

It is only possible to fill the region between two listplots with \pscustom if one of them has the values in reverse order. Otherwise we do not get a closed path. With the option ChangeOrder the values are used in reverse order:

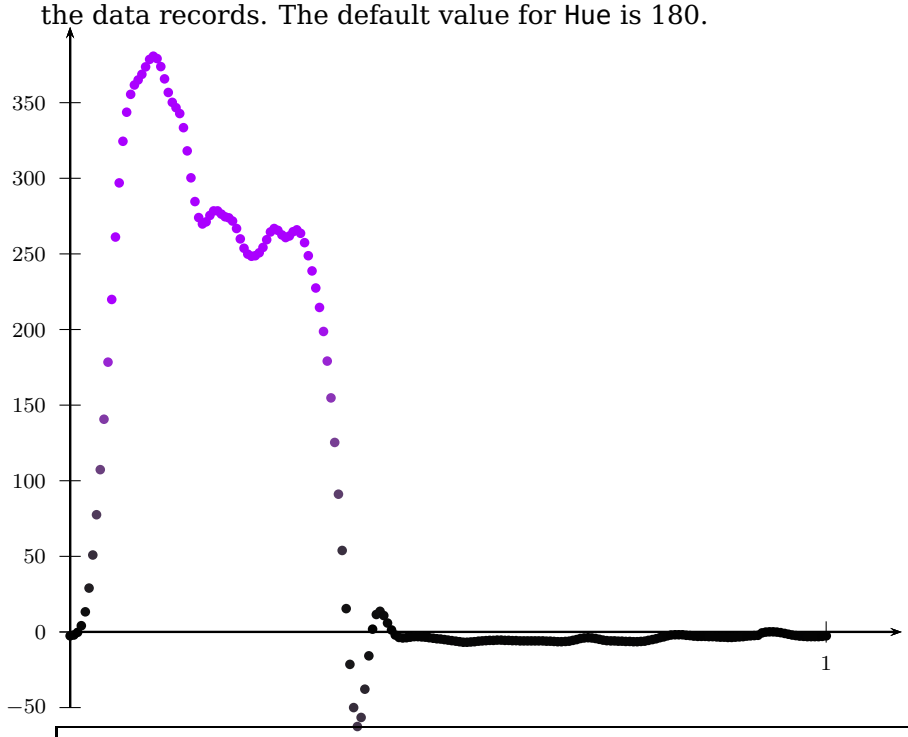



```
\begin{filecontents*}{test.data}
  0 3 8
  2 4 7
  5 5 5.5
  7 3.5 5
  10 2 9
\end{filecontents*}
\psset{lly=-.5cm}
\begin{psgraph}[axesstyle=frame,ticklinestyle=dotted,ticksize=0 10](0,0)(10,10)
    {4in}{2in}%
\readdata{\data}{test.data}%
\pscustom[fillstyle=solid,fillcolor=blue!40]{%
  \listplot[plotNo=2,plotNoMax=2]{\data}%
  \listplot[plotNo=1,plotNoMax=2,ChangeOrder]{\data}}
\end{psgraph}
```

12. New plot styles

12.1. Plot style `colordot` and option `Hue`

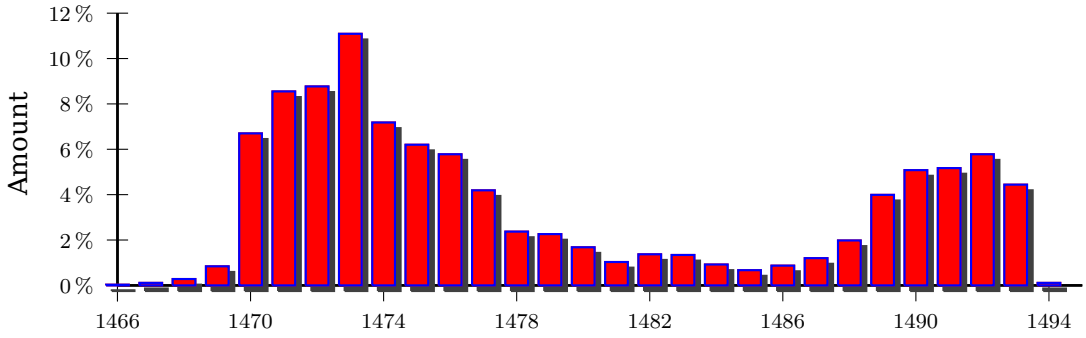
The plotted dots can be colored with the HSB color model, where Hue is set by an angle (0...360) and the values of Saturation and Brightnes are set by the relative y value of the data records. The default value for `Hue` is 180.



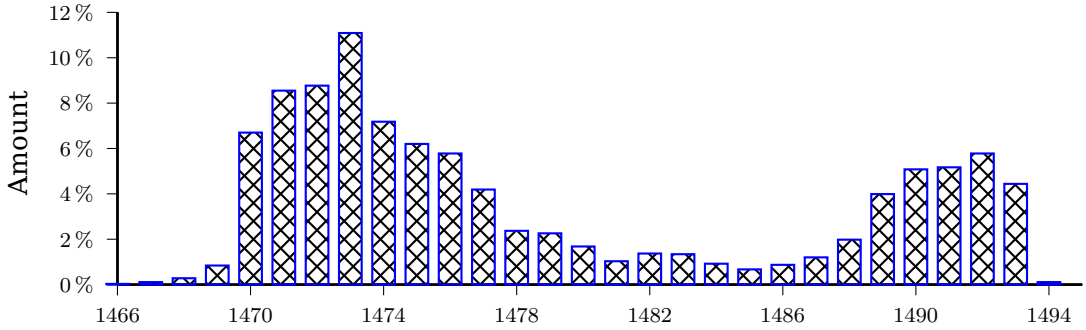

```
\readdata{\data}{data3.data}
\psset{xunit=10,yunit=0.02}
\begin{pspicture}(0,-50)(1.1,400)
  \psaxes[dy=1cm,Dy=50]{->}(0,0)(0,-50)(1.1,400)
  \listplot[Hue=280,plotstyle=colordots]{\data}
\end{pspicture}
```

12.2. Plot style bar and option barwidth

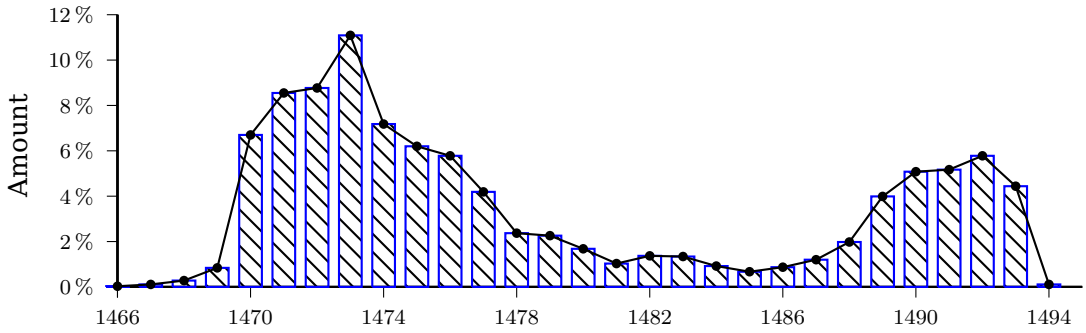
This option allows you to draw bars for the data records. The width of the bars is controlled by the option barwidth, which is set by default to value of 0.25cm, which is the total width.




```
\psset{xunit=.44cm,yunit=.3cm}
\begin{pspicture}(-2,-3)(29,13)
  \psaxes[axesstyle=axes,Ox=1466,Oy=0,Dx=4,Dy=2,xticksize=-6pt 0,
    ylabelFactor={\,\%}]{-}(29,12)
  \listplot[shadow=true,linecolor=blue,plotstyle=bar,barwidth=0.3cm,
    fillcolor=red,fillstyle=solid]{\barData}
  \rput{90}(-3,6.25){Amount}
\end{pspicture}
```



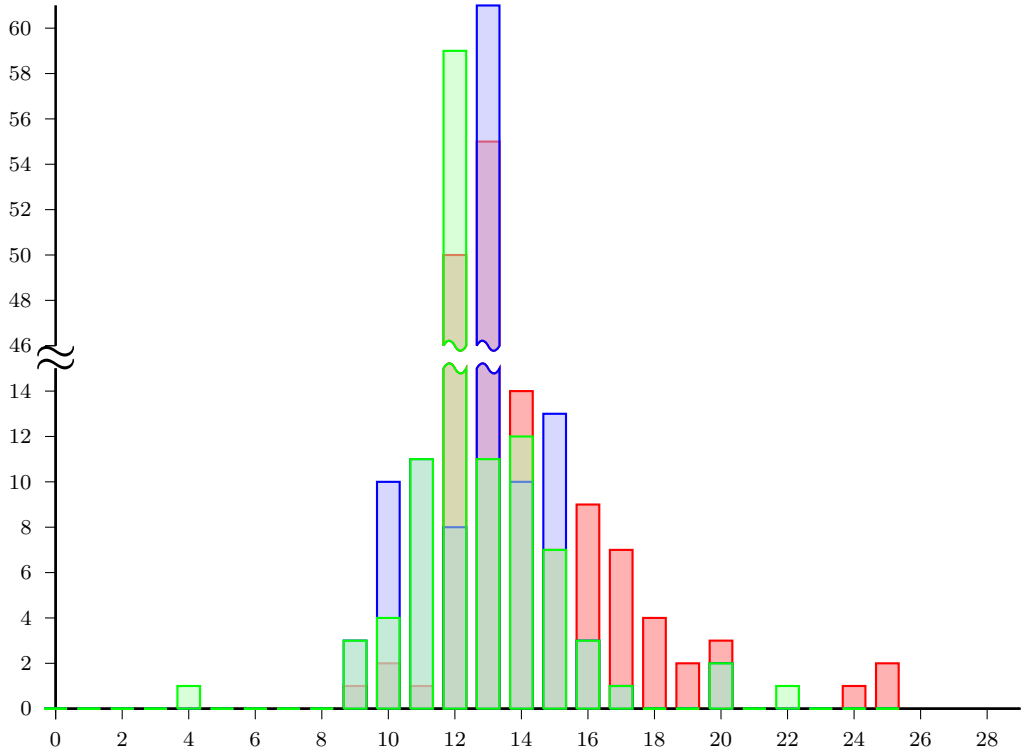

```
\psset{xunit=.44cm,yunit=.3cm}
\begin{pspicture}(-2,-3)(29,13)
  \psaxes[axesstyle=axes,Ox=1466,Oy=0,Dx=4,Dy=2,ticksize=-4pt 0,
    ylabelFactor={\,\%}]{-}(29,12)
  \listplot[linecolor=blue,plotstyle=bar,barwidth=0.3cm,
    fillcolor=red,fillstyle=crosshatch]{\barData}
  \rput{90}(-3,6.25){Amount}
\end{pspicture}
```




```
\psset{xunit=.44cm,yunit=.3cm}
\begin{pspicture}(-2,-3)(29,13)
  \psaxes[axesstyle=axes,Ox=1466,Oy=0,Dx=4,Dy=2,ticksize=-4pt 0,
    ylabelFactor={\,\%}]{-}(29,12)
  \listplot[linecolor=blue,plotstyle=bar,barwidth=0.3cm,
    fillcolor=red,fillstyle=vlines]{\barData}
  \listplot[showpoints=true]{\barData}
  \rput{90}(-3,6.25){Amount}
\end{pspicture}
```

Interrupted bar plot

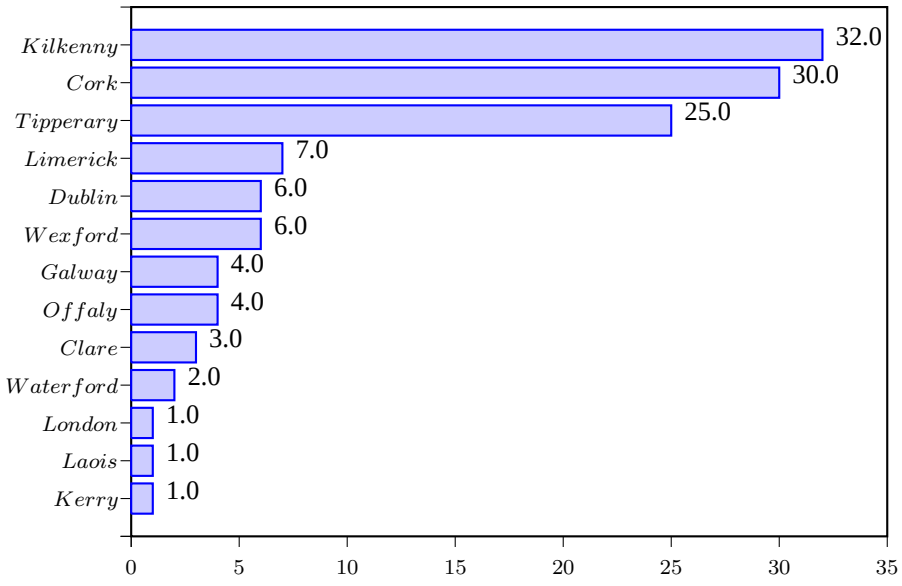
The new keywords `interrupt` takes three comma separated values: the value, when the interrupted y axis is interruptes, the separation for the drawn tilde and the value for the inrrupted section, e.g. `interrupt={15,1,30}`.



```
\psset{xunit=.44cm,yunit=.3cm}
\begin{pspicture}(-2,-3)(29,32)
\psaxes[axesstyle=axes,ticksize=-4pt 0,Dy=2,Dx=2](29,15)
\rput(0,15.4){\textbf{\huge$\approx$}}
\rput(0,16){\psaxes[xAxis=false,ticksize=-4pt 0,
  Dy=2,Oy=46,Dx=2](29,15)}
\psset{interrupt={15,1,30}}
\listplot[linecolor=red,plotstyle=bar,barwidth=0.3cm,
  fillcolor=red!30,fillstyle=solid]{
0 0 1 0 2 0 3 0 4 0 5 0 6 0 7 0 8 0 9 1 10 2
11 1 12 50 13 55 14 14 15 7 16 9 17 7 18 4
19 2 20 3 21 0 22 0 23 0 24 1 25 2 % 1st example
}
\listplot[linecolor=blue,plotstyle=bar,barwidth=0.3cm,
  fillcolor=blue!30,fillstyle=solid,opacity=0.5]{
0 0 1 0 2 0 3 0 4 0 5 0 6 0 7 0 8 0 9 3 10 10
11 11 12 8 13 61 14 10 15 13 16 3 17 1 18 0
19 0 20 2 21 0 22 0 23 0 24 0 25 0 % 2nd exa
}
\listplot[linecolor=green,plotstyle=bar,barwidth=0.3cm,
  fillcolor=green!30,fillstyle=solid,opacity=0.5]{
0 0 1 0 2 0 3 0 4 1 5 0 6 0 7 0 8 0 9 3 10 4
11 11 12 59 13 11 14 12 15 7 16 3 17 1 18 0
19 0 20 2 21 0 22 1 23 0 24 0 25 0 % 3rd exa
}
\end{pspicture}
```

12.3. Plot style ybar

With the setting `plotstyle=ybar` the graph is set with horizontal bars instead of vertical. For `yLabels` see section **??**. .



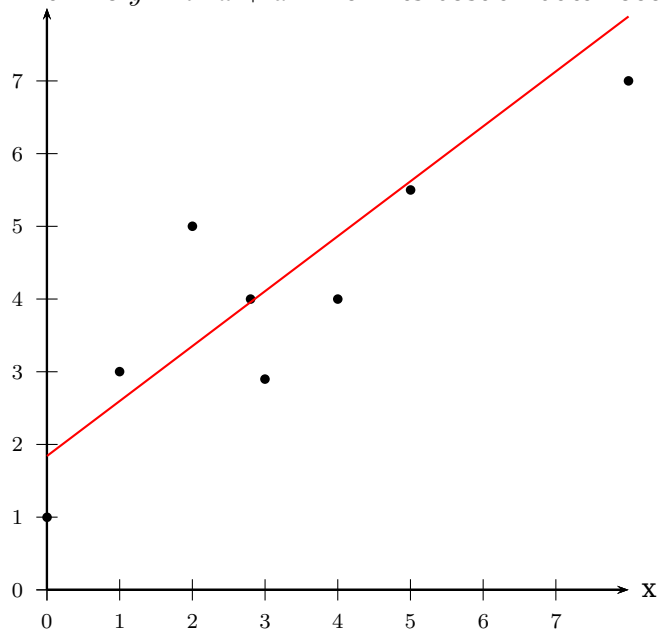
Frequency of Winning the Final

```
\savedata{\data}[1 1 1 2 1 3 2 4 3 5 4 6 4 7 6 8 6 9 7 10 25 11 30 12 32 13]

\psset{llx=-1.5cm,lly=-1.5cm,xAxisLabel=\textbf{Frequency of Winning the Final
    },
  xAxisLabelPos={c,-1cm},yAxisLabel=,yLabels={,Kerry,Laois,London,Waterford,
      Clare,Offaly,
  Galway,Wexford,Dublin,Limerick,Tipperary,Cork,Kilkenny}}
\begin{psgraph}[axesstyle=frame,labels=x,ticksize=-4pt 0,Dx=5](0,0)(35,14){10cm
    }{7cm}
  \listplot[plotstyle=ybar,fillcolor=blue!20,linecolor=blue,barwidth=4mm,
      fillstyle=solid]{\data}
  \listplot[plotstyle=xvalues,labelsep=5pt]{\data}
\end{psgraph}
```

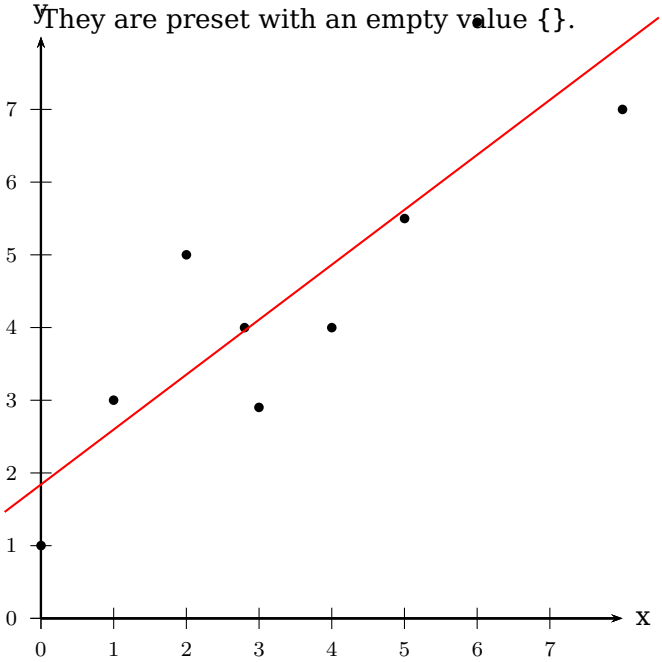
12.4. Plotstyle LSM

With the setting `plotstyle=LSM` (**L**east **S**quare **M**ethod) the data records are not printed in the usual way as dots or a line, the `\listplot` macro calculates the values for a line $y = v \cdot x + u$ which fits best all data records.




```
\begin{filecontents*}{LSM.data}
0 1 1 3 2.8 4 3 2.9 2 5 4 4 5 5.5 6 8.2 8 7
\end{filecontents*}
\psset{lly=-.5cm}
\readdata{\data}{LSM.data}
\begin{psgraph}[arrows=->](0,0)(0,0)(8,8){.5\textwidth}{!}
  \listplot[plotstyle=dots]{\data}
  \listplot[plotstyle=LSM,linecolor=red]{\data}
\end{psgraph}
```

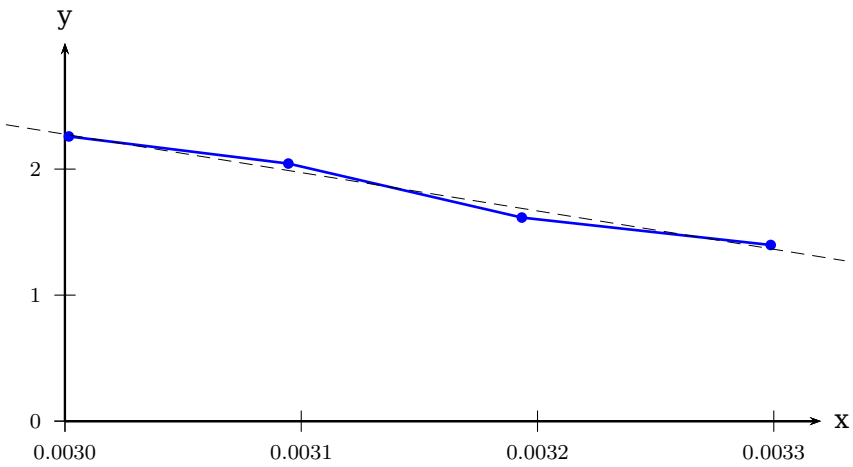
The macro looks for the lowest and biggest x-value and draws the line for this interval. It is possible to pass other values to the macro by setting the `xStart` and/or `xEnd` options. They are preset with an empty value {}.



y=0.755679 x+1.84105

```
\begin{filecontents*}{LSM.data}
0 1 1 3 2.8 4 3 2.9 2 5 4 4 5 5.5 6 8.2 8 7
\end{filecontents*}
\readdata{\data}{LSM.data}
\psset{lly=-1.75cm}
\begin{psgraph}[arrows=->](0,0)(0,0)(8,8){.5\textwidth}{!}
  \listplot[plotstyle=dots]{\data}
  \listplot[PstDebug,plotstyle=LSM,xStart=-0.5,xEnd=8.5,linecolor=red]{\data}
\end{psgraph}
```

With `PstDebug` one gets the equation $y = v \cdot x + u$ printed, beginning at the position (0|-50pt). This cannot be changed, because it is only for some kind of debugging. Pay attention for the correct `xStart` and `xEnd` values, when you use the `\pstScalePoints` Macro. In the following example we use an x-interval from 0 to 3 to plot the values; first we subtract 0.003 from all x-values and then scale them with 10000. This is not taken into account for the `xStart` and `xEnd` values.

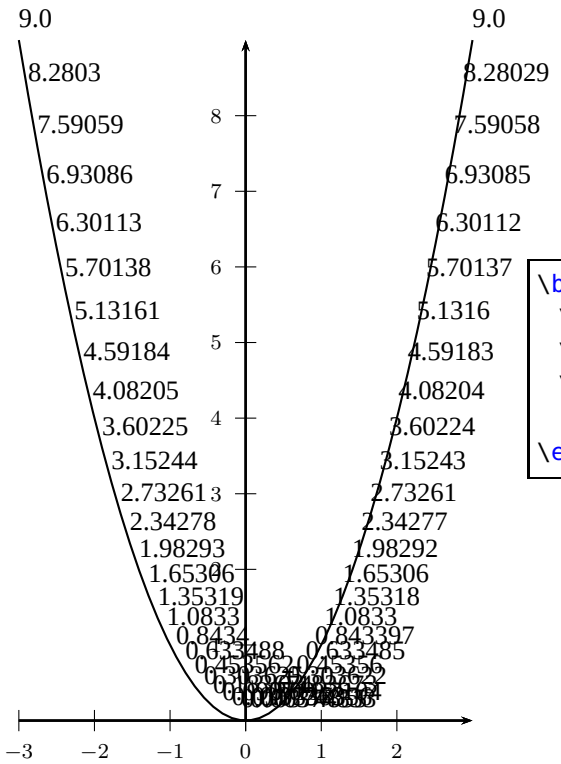


y=-0.304095 x+2.27634

```
\begin{filecontents*}{LSM.data}
0.003298697 1.397785583
0.003193358 1.615489564
0.003094538 2.044019006
0.003001651 2.259240127
\end{filecontents*}
\readdata{\data}{LSM.data}
\pstScalePoints(10000,1){ 0.003 sub }{}
\psset{lly=-1.75cm}
\psgraph[arrows=->,Ox=0.0030,Dx=0.0001,dx=\psxunit](0,0)(3.2,3){10cm}{5cm}
 \listplot[showpoints=true,linewidth=1pt,linecolor=blue]{\data}
 \listplot[PstDebug=1,plotstyle=LSM,linewidth=0.1pt,linestyle=dashed,%
   xStart=-0.25,xEnd=3.3]{\data}
\endpsgraph
```

12.5. Plotstyles values and values∗

Instead of plotting the curve with the setting `plotstyle=values` the y-values are printed at the current point.

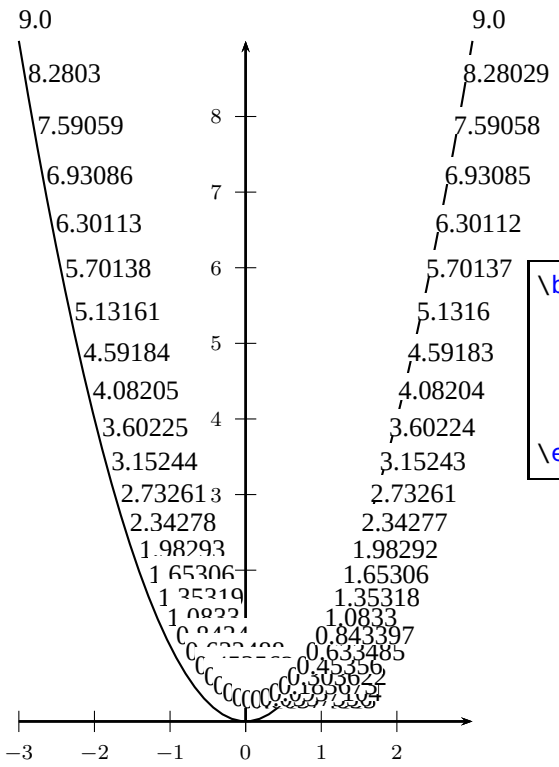



```
\begin{pspicture}(-3.5,-.5)(3.5,9.5)
  \psaxes{->}(0,0)(-3,0)(3,9)
  \psplot{-3}{3}{x dup mul }
  \psplot[plotstyle=values]{-3}{3}{x dup
      mul }
\end{pspicture}
```

The possible optional arguments are `PSfont`,`valuewidth`, `fontscale`, and `decimals`. The default setting is:

```
\psset[pst-plot]{PSfont=Times-Roman,fontscale=10,valuewidth=10,decimals=-1}
```

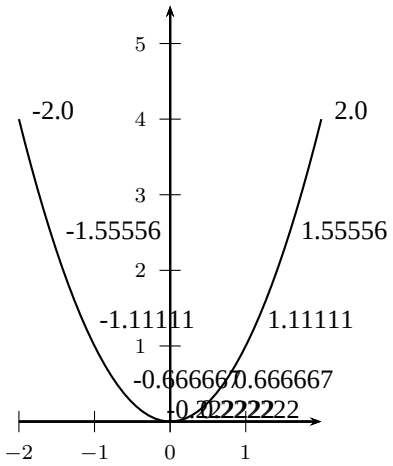
The optional argument `rot` from the base package `pstricks` is also taken into account. With the star version `plotstyle=values∗` the box of the printed value isn't transparent, everything behind this box is not seen.




```
\begin{pspicture}(-3.5,-.5)(3.5,9.5)
  \psaxes{->}(0,0)(-3,0)(3,9)
  \psplot{-3}{3}{x dup mul }
  \psplot[plotstyle=values∗]{-3}{3}{x dup
      mul }
\end{pspicture}
```

12.6. Plotstyles xvalues and xvalues∗

This is similiar to the options `values`, except that it plots the x-values instead of the y-values. This maybe useful when also using the plotstyle `ybar` (see Section 12.3 on page 76).




```
\begin{pspicture}(-2.5,-.5)(2.5,5.5)
  \psaxes{->}(0,0)(-2,0)(2,5.5)
  \psplot{-2}{2}{x dup mul }
  \psplot[plotstyle=xvalues,
    plotpoints=10]{-2}{2}{x dup mul }
\end{pspicture}
```

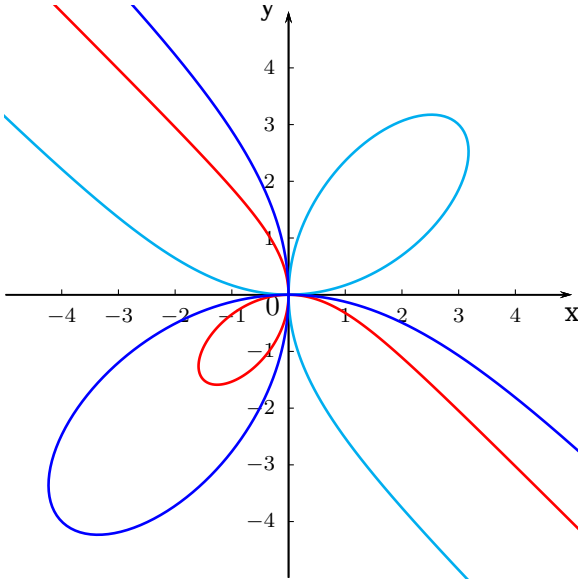
13. Polar plots

With the option `polarplot=false|true` it is possible to use any plot command in polar mode:

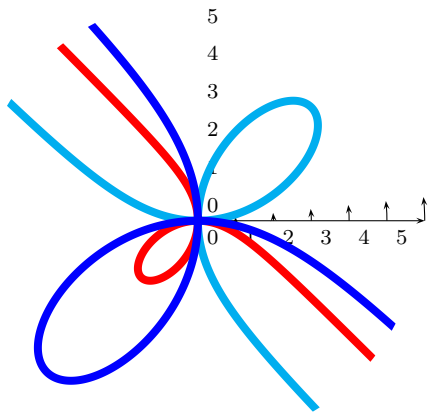
```
\ps????plot [polarplot,...] {<start angle>}{<end angle>}%
 [PS command] {<r(alpha)>}
```

The equation in PostScript code is interpreted as a function $r = f(\alpha)$, e.g. for the circle with radius 1 as $r = \sqrt{\sin^2 x + \cos^2 x}$, or $r = a * \frac{sin(x) * cos(x)}{(sin(x)^3 + cos(x)^3)}$ for the following examples:

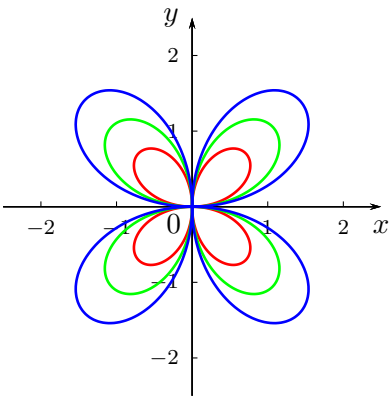
```
x sin dup mul x cos dup mul add sqrt
```



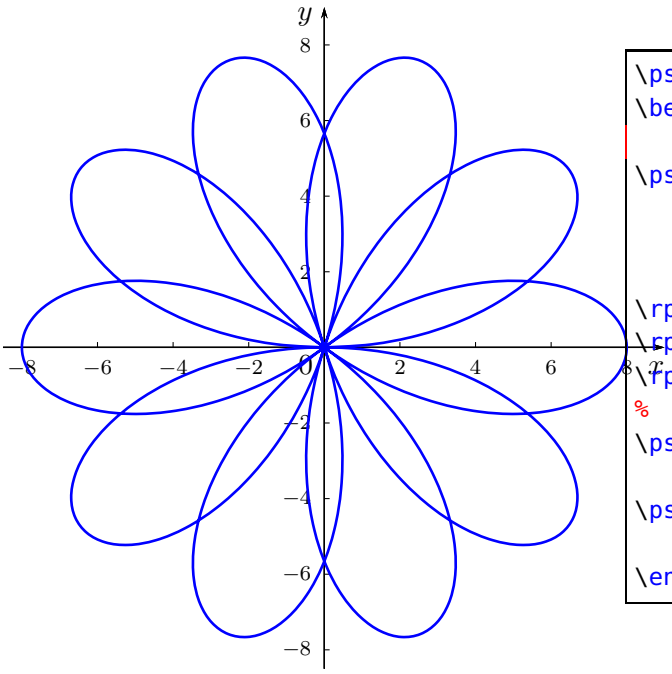

```
\psset{plotpoints=200,unit=0.75}
\begin{pspicture*}(-5,-5)(5.1,5.1)
  \psaxes[arrowlength=1.75,ticksize=2pt,labelFontSize=\scriptstyle,
    linewidth=0.2mm]{->}(0,0)(-4.99,-4.99)(5,5)[x,-90][y,180]
  \rput[Br](-.15,-.35){$0$} \psset{linewidth=.35mm,polarplot}
  \psplot[linecolor=red]{140}{310}{3 neg x sin mul x cos mul x sin 3 exp x cos 3
      exp add div}
  \psplot[linecolor=cyan]{140}{310}{6 x sin mul x cos mul x sin 3 exp x cos 3 exp
       add div}
  \psplot[linecolor=blue,algebraic]{2.44}{5.41}{-8*sin(x)*cos(x)/(sin(x)^3+cos(x)
      ^3)}
\end{pspicture*}
```




```
\psset{unit=0.5cm}
\begin{pspicture}(-6,-6)(6,6)
\psaxes[axesstyle=polar,labelFontSize=\scriptstyle,linewidth=0.2mm]{->}(6,6)
\psset{linewidth=3pt,polarplot,plotpoints=500,plotstyle=curve}
\psclip{\pscircle[linestyle=none]{6}}
  \psplot[linecolor=red]{140}{310}{3 neg x sin mul x cos mul x sin 3 exp x cos 3
      exp add div}
  \psplot[linecolor=cyan]{140}{310}{6 x sin mul x cos mul x sin 3 exp x cos 3 exp
       add div}
  \psplot[linecolor=blue,algebraic]{2.44}{5.41}{-8*sin(x)*cos(x)/(sin(x)^3+cos(x)
      ^3)}
\endpsclip
\end{pspicture}
```




```
\psset{plotpoints=200,unit=1}
\begin{pspicture}(-2.5,-2.5)(2.5,2.5)% Ulrich Dirr
 \psaxes[arrowlength=1.75,%
   ticksize=2pt,linewidth=0.17mm]{->}%
     (0,0)(-2.5,-2.5)(2.5,2.5)[$x$,-90][$y$,180]
  \rput[Br](-.15,-.35){$0$}
  \psset{linewidth=.35mm,plotstyle=curve,polarplot}
  \psplot[linecolor=red]{0}{360}{x cos 2 mul x sin
      mul}
  \psplot[linecolor=green]{0}{360}{x cos 3 mul x sin
      mul}
  \psplot[linecolor=blue]{0}{360}{x cos 4 mul x sin
      mul}
\end{pspicture}
```

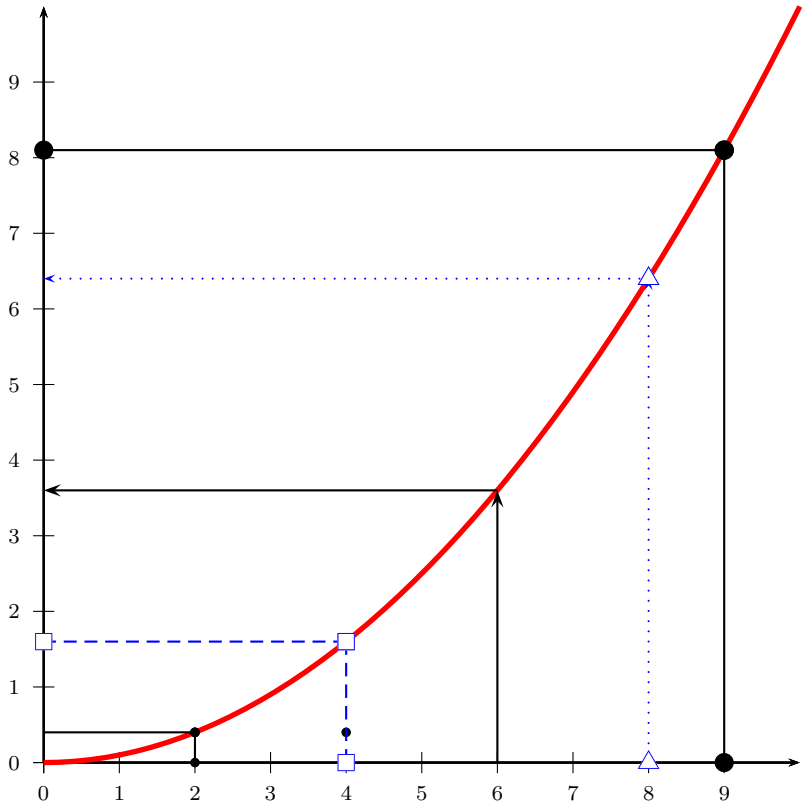



```
\psset{plotpoints=200,unit=0.5}
\begin{pspicture}(-8.5,-8.5)(9,9)%
    Ulrich Dirr
\psaxes[Dx=2,dx=2,Dy=2,dy=2,
    arrowlength=1.75,
   ticksize=2pt,linewidth=0.17mm
       ]{->}(0,0)(-8.5,-8.5)(9,9)
\rput[Br](9,-.7){$x$}
\rput[tr](-.3,9){$y$}
\rput[Br](-.3,-.7){$0$}
%
\psset{linewidth=.35mm,plotstyle=
    curve,polarplot}
\psplot[linecolor=blue]{0}{720}{8
    2.5 x mul sin mul}
\end{pspicture}
```

14. New macros

14.1. \psCoordinates

\psCoordinates [Options] (x,y)

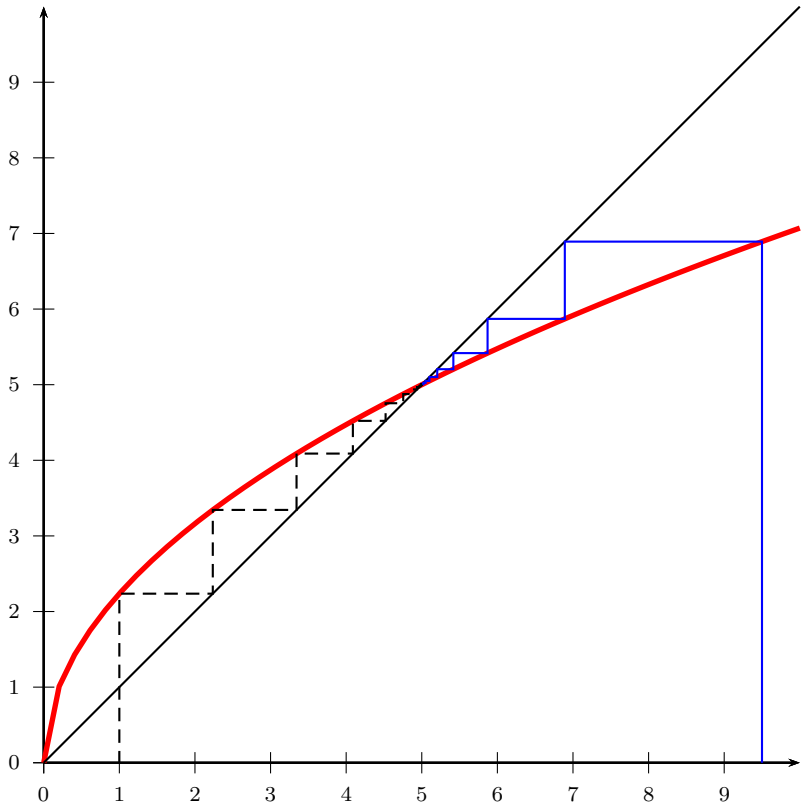



```
\begin{pspicture}(-5mm,-1cm)(10,10)
  \psaxes{->}(10,10)
  \psplot[algebraic,linecolor=red,linewidth=2pt]{0}{10}{x^2/10}
  \psCoordinates(*2 {x^2/10})
  \psCoordinates[linecolor=blue,linestyle=dashed,
    dotstyle=square,dotscale=2](*4 {x^2/10})
  \psCoordinates[arrowscale=1.5,arrows=->,showpoints=false](*6 {x^2/10})
  \psCoordinates[arrows=->,linecolor=blue,linestyle=dotted,
    dotstyle=triangle,dotscale=2](*8 {x^2/10})
  \psCoordinates[dotscale=2](*9 {x^2/10})
\end{pspicture}
```

14.2. \psFixpoint

\psFixpoint [Options] {x_0}{$f(x)$}{n}

x_0 is the start value of the iteration, $f(x)$ the function, which can either be in postfix or algebraic notation, for the latter it needs the optional argument `algebraic`. The number of the iteration is given by n.

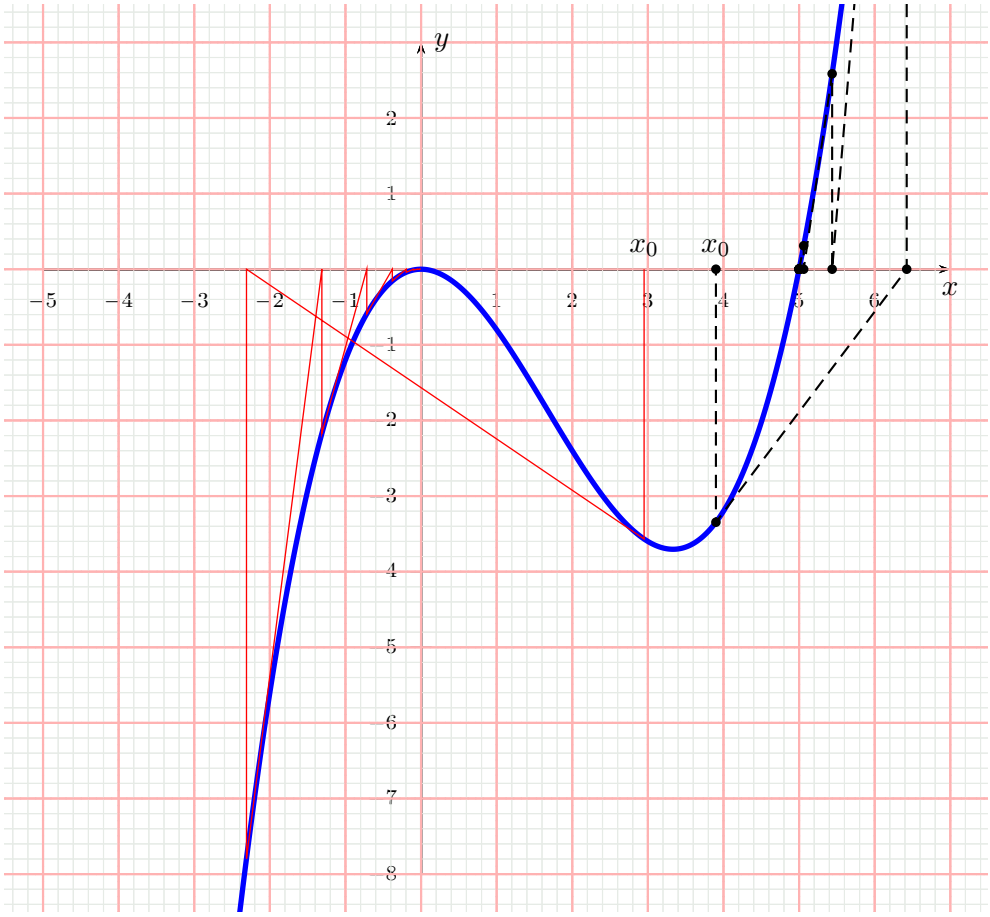



```
\begin{pspicture}[algebraic](-5mm,-1cm)(10,10)
  \psaxes{->}(10,10)
  \psplot[linecolor=red,linewidth=2pt]{0}{10}{sqrt(5*x)}
  \psline(10,10)
  \psFixpoint[linecolor=blue]{9.5}{sqrt(5*x)}{20}
  \psFixpoint[linestyle=dashed]{1}{sqrt(5*x)}{20}
\end{pspicture}
```

14.3. \psNewton

\psNewton [Options] {x_0}{$f(x)$} [$f'(x)$] {n}

If the optional derivation of the function $f(x)$ is missing, then the macro itself calculates the derivation with an interval of ± 0.01. It can be changed by setting the optional argument `VarStepEpsilon` to another value. If the derivation is also given as a function, it is used without any check for the values.




```
\def\f{1/5*x^3-x^2}
\psset{plotpoints=2000,algebraic}
%
\begin{pspicture*}[showgrid](-5.5,-8.5)(7.5,3.5)
\psaxes{->}(0,0)(-5,-8)(7,3)[$x$,270][$y$,0]
\psplot[linewidth=2pt,linecolor=blue]{-5}{8}{\f}
\uput[90](2.95,0){$x_0$}\uput[90](3.9,0){$x_0$}
\psNewton[linecolor=red,linewidth=0.5pt]{2.95}{\f}{10}
\psNewton[showpoints,linestyle=dashed]{3.9}{\f}{8}
\end{pspicture*}
```

x_0 is the start value of the iteration, $f(x)$ the function, which can either be in postfix or algebraic notation, for the latter it needs the optional argument `algebraic`. The number of the iteration is given by n. All defined plotstyles can be used, but there

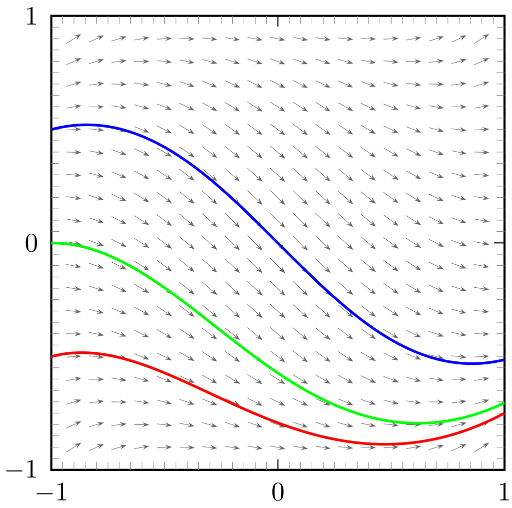
maybe PostScript errors for `plotstyle=values` if the number of steps is too big. In such a case decrease the number of steps.


```
\def\f{-(1/192)*x^3-(1/12)*x-(1/192)*Pi*x^2-(1/12)*Pi+2}
\def\fDerive{-(3/192)*x^2-(1/12)-(2/192)*Pi*x}
\psset{plotpoints=2000,unit=0.5,algebraic}
%
\begin{pspicture*}[showgrid](-16,-5)(8.5,18.5)
\psaxes[Dx=6,Dy=4]{->}(0,0)(-16,-5)(8,18)[$x$,270][$y$,0]
\psplot[algebraic,linewidth=2pt,linecolor=blue]{-20}{8}{\f}
\psxTick(-15){x_0}
\psNewton[linecolor=red,linewidth=0.5pt]{-15}{\f}{12}
\psNewton[linecolor=red,linewidth=0.5pt,plotstyle=xvalues,showDerivation=false
    ]{-15}{\f}{6}
%
%-15, -9.567466932, -4.903526029, 3.026073041, 6.688396612, 5.580230655 (Made by
     Maple)
\end{pspicture*}
```

14.4. \psVectorfield

\psVectorfield [Options] $(x_0, y_0)(x_1, y_1)\{f'(x,y)\}$

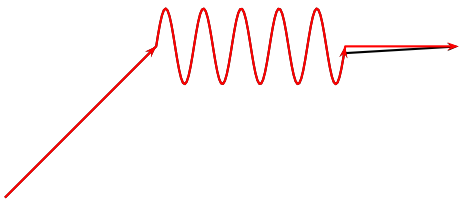
$f'(x,y)$ can be in Postfix notation or with option algebraic in Infix notation. The Δx and Δy are given by Dx and Dy and preset to 0.1, the length of the arrow lines is relative and internally set by 1/Ox with a preset of Ox=3.




```
%\usepackage{pst-ode}
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%solve dy/dx=x^2 + y^2 - 1 numerically for different initial values of y in the
%interval x=[-1.1,1.1]; store resulting data points as tables into Postscript
%objects
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
\psset{unit=3cm}
\begin{pspicture}(-1.2,-1.2)(1.1,1.1)
\psaxes[ticksize=0 4pt,axesstyle=frame,tickstyle=inner,subticks=20,
      Ox=-1,Oy=-1](-1,-1)(1,1)
\psset{arrows=->,algebraic}
\psVectorfield[linecolor=black!60](-0.9,-0.9)(0.9,0.9){ x^2+y^2-1 }
%y0_a=-0.5
\pstODEsolve[algebraicOutputFormat]{y0_a}{t | x[0]}{-1}{1}{100}{-0.5}{t^2+x
    [0]^2-1}
%y0_b=0.0
\pstODEsolve[algebraicOutputFormat]{y0_b}{t | x[0]}{-1}{1}{100}{0.0}{t^2+x
    [0]^2-1}
%y0_c=0.5
\pstODEsolve[algebraicOutputFormat]{y0_c}{t | x[0]}{-1}{1}{100}{0.5}{t^2+x
    [0]^2-1}
\psset{arrows=-}%
\listplot[linecolor=red, linewidth=1pt]{y0_a}
\listplot[linecolor=green,linewidth=1pt]{y0_b}
\listplot[linecolor=blue, linewidth=1pt]{y0_c}
\end{pspicture}
```

15. Internals

The last pair of coordinates of `\psplot` and `\psparametricplot` is saved in a PostScript array and can be used as `FinalState` inside PostScript code.



```
\psset{unit=2}
\begin{pspicture}(0,-1)(3,0.5)
  \pscustom[linejoin=1,arrows=->]{%
    \psline(0,-1)(1,0)
    \psplot[algebraic,plotpoints=100]{1}{2.25}{.25*sin(2*Pi*x/.25)}
    \psline(3,0)
  }
%
  \pscustom[linejoin=1,arrows=->,linecolor=red]{%
    \psline(0,-1)(1,0)
    \psplot[algebraic,plotpoints=100]{1}{2.25}{.25*sin(2*Pi*x/.25)}
    \psline(! FinalState aload pop )(3,0)
  }
\end{pspicture}
```

16. List of all optional arguments for pst-plot

Key	Type	Default
ignoreLines	ordinary	0
Hue	ordinary	180
barwidth	ordinary	0.25cm
interrupt	ordinary	
IQLfactor	ordinary	[none]
postAction	ordinary	
plotstyle	ordinary	line
plotpoints	ordinary	50
yMaxValue	ordinary	1.e30
yMinValue	ordinary	-1.e30
PSfont	ordinary	Times-Roman
valuewidth	ordinary	10
fontscale	ordinary	10
decimals	ordinary	-1
xlabelsep	ordinary	5pt
ylabelsep	ordinary	5pt
xyValues	boolean	true
ChangeOrder	boolean	true
polarplot	boolean	true
VarStep	boolean	true
PlotDerivative	ordinary	[none]
VarStepEpsilon	ordinary	[none]
method	ordinary	[none]
ticks	ordinary	all
labels	ordinary	all
Ox	ordinary	0
Dx	ordinary	1
dx	ordinary	0
Oy	ordinary	0
Dy	ordinary	1
dy	ordinary	0
showorigin	boolean	true
labelFontSize	ordinary	
xlabelFontSize	ordinary	
ylabelFontSize	ordinary	
mathLabel	boolean	true
xmathLabel	boolean	true
ymathLabel	boolean	true
xAxis	boolean	true
yAxis	boolean	true
xyAxes	boolean	true

Continued on next page

Continued from previous page

Key	Type	Default
xlabelPos	ordinary	b
ylabelPos	ordinary	l
xyDecimals	ordinary	
xDecimals	ordinary	
yDecimals	ordinary	
xlogBase	ordinary	
ylogBase	ordinary	
xylogBase	ordinary	
trigLabelBase	ordinary	0
xtrigLabels	boolean	true
ytrigLabels	boolean	true
trigLabels	boolean	true
logLines	ordinary	none
ylabelFactor	ordinary	\relax
xlabelFactor	ordinary	\relax
showOriginTick	boolean	true
ticksize	ordinary	-4pt 4pt
xticksize	ordinary	[none]
yticksize	ordinary	[none]
tickstyle	ordinary	full
subticks	ordinary	1
xsubticks	ordinary	1
ysubticks	ordinary	1
subticksize	ordinary	0.75
xsubticksize	ordinary	0.75
ysubticksize	ordinary	0.75
tickwidth	ordinary	0.5\pslinewidth
xtickwidth	ordinary	0.5\pslinewidth
ytickwidth	ordinary	0.5\pslinewidth
subtickwidth	ordinary	0.25\pslinewidth
xsubtickwidth	ordinary	0.25\pslinewidth
ysubtickwidth	ordinary	0.25\pslinewidth
labelOffset	ordinary	0pt
xlabelOffset	ordinary	0pt
ylabelOffset	ordinary	0pt
frameOffset	ordinary	0pt
tickcolor	ordinary	black
xtickcolor	ordinary	black
ytickcolor	ordinary	black
subtickcolor	ordinary	gray
xsubtickcolor	ordinary	gray
ysubtickcolor	ordinary	gray
xticklinestyle	ordinary	solid

Continued on next page

Continued from previous page

Key	Type	Default
xsubticklinestyle	ordinary	solid
yticklinestyle	ordinary	solid
ysubticklinestyle	ordinary	solid
ticklinestyle	ordinary	solid
subticklinestyle	ordinary	solid
nStep	ordinary	1
nStart	ordinary	0
nEnd	ordinary	
xStep	ordinary	0
yStep	ordinary	0
xStart	ordinary	
xEnd	ordinary	
yStart	ordinary	
yEnd	ordinary	
plotNoX	ordinary	1
plotNo	ordinary	1
plotNoMax	ordinary	1
axesstyle	ordinary	axes
xLabels	ordinary	
xLabelsRot	ordinary	0
yLabels	ordinary	
yLabelsRot	ordinary	0
xAxisLabel	ordinary	x
yAxisLabel	ordinary	y
xAxisLabelPos	ordinary	
yAxisLabelPos	ordinary	
llx	ordinary	\z@
lly	ordinary	\z@
urx	ordinary	\z@
ury	ordinary	\z@
psgrid	boolean	true
gridpara	ordinary	
gridcoor	ordinary	\relax
axespos	ordinary	bottom
showDerivation	boolean	true

References

[1] Victor Eijkhout. *TEX by Topic – A TEXnician Reference*. DANTE – lehmanns media, Heidelberg/Berlin, 1 edition, 2014.

[2] Denis Girou. Présentation de PSTricks. *Cahier GUTenberg*, 16:21–70, April 1994.

[3] Michel Goosens, Frank Mittelbach, Sebastian Rahtz, Dennis Roegel, and Her-

bert Voß. *The LaTeX Graphics Companion*. Addison-Wesley Publishing Company, Boston, Mass., second edition, 2007.

[4] Nikolai G. Kollock. *PostScript richtig eingesetzt: vom Konzept zum praktischen Einsatz*. IWT, Vaterstetten, 1989.

[5] Herbert Voß. *PSTricks – Grafik für TeX und LaTeX*. DANTE – Lehmanns, Heidelberg/Hamburg, 6. edition, 2010.

[6] Herbert Voß. *PSTricks – Graphics and PostScript for LaTeX*. UIT, Cambridge – UK, 1. edition, 2011.

[7] Herbert Voß. *LaTeX quick reference*. UIT, Cambridge – UK, 1. edition, 2012.

[8] Herbert Voß. *Presentations with LaTeX*. DANTE – Lehmanns Media, Heidelberg/Berlin, 1. edition, 2012.

[9] Timothy Van Zandt. *multido.tex - a loop macro, that supports fixed-point addition*. CTAN:/macros/generic/multido.tex, 1997.

[10] Timothy Van Zandt and Denis Girou. Inside PSTricks. *TUGboat*, 15:239–246, September 1994.

Index