

# pstricks-add

## additional Macros for pstricks\*

v.2.18

Herbert Voß

October 16, 2004

### Abstract

This version of `pstricks-add` needs `pstricks.tex` version 1.04 from June 2004, otherwise the additional macros may not work as expected. The ellipsis material and the option `asolid` (renamed to `eofill`) are now part of the new `pstricks.tex` package, available at CTAN or at <http://perce.de/LaTeX/pstricks>.

## Contents

<b>I</b>	<b>pstricks</b>	<b>5</b>
<b>1</b>	<b>Numeric functions</b>	<b>5</b>
1.1	<code>\pst@divide</code> . . . . .	5
1.2	<code>\pst@mod</code> . . . . .	5
1.3	<code>\pst@max</code> . . . . .	6
1.4	<code>\pst@maxdim</code> . . . . .	6
1.5	<code>\pst@abs</code> . . . . .	6
1.6	<code>\pst@absdim</code> . . . . .	6
<b>2</b>	<b>Dashed Lines</b>	<b>7</b>
<b>3</b>	<b><code>\rmultiput</code>, a multiple <code>\rput</code></b>	<b>7</b>

---

\*This document was written with Kile: 1.61 (Qt: 3.1.1; KDE: 3.2.1; <http://sourceforge.net/projects/kile/>) and the PDF output was build with VTeX/Free (<http://www.micropress-inc.com/linux>)

<b>4</b>	<b>\pslineII Colored lines</b>	<b>8</b>
4.1	The options . . . . .	8
4.2	Examples . . . . .	9
<b>5</b>	<b>\pslineIII Variable linewidth</b>	<b>10</b>
5.1	The options . . . . .	11
5.2	Examples . . . . .	11
<b>6</b>	<b>\psbrace</b>	<b>12</b>
6.1	Syntax . . . . .	12
6.2	Options . . . . .	12
6.3	Examples . . . . .	13
<b>7</b>	<b>Arrows</b>	<b>17</b>
7.1	Definition . . . . .	17
7.2	Multiple arrows . . . . .	17
7.3	hookrightarrow and hookleftarrow . . . . .	18
7.4	ArrowInside Option . . . . .	19
7.5	ArrowFill Option . . . . .	20
7.6	Examples . . . . .	21
7.6.1	\psline . . . . .	21
7.6.2	\pspolygon . . . . .	23
7.6.3	\psbezier . . . . .	25
7.6.4	\pcline . . . . .	27
7.6.5	\pccurve . . . . .	28
<b>8</b>	<b>\psFormatInt</b>	<b>28</b>
<b>II</b>	<b>pst-node</b>	<b>29</b>
<b>9</b>	<b>\nclineII</b>	<b>29</b>
9.1	The options . . . . .	29
9.2	Examples . . . . .	29
<b>10</b>	<b>\pclineII</b>	<b>30</b>

11 <code>\ncdiag</code> and <code>\pcdiag</code>	30
12 <code>\ncdiagg</code> and <code>\pcdiagg</code>	32
13 <code>\ncbarr</code>	34
14 <code>\psLNode</code> and <code>\psLCNode</code>	34

### III `pst-plot` 36

15 New options	36
15.1 <code>algebraic</code> . . . . .	37
15.2 <code>comma</code> . . . . .	39
15.3 <code>xyAxes</code> , <code>xAxis</code> and <code>yAxis</code> . . . . .	40
15.4 <code>xyDecimals</code> , <code>xDecimals</code> and <code>yDecimals</code> . . . . .	41
15.5 <code>xyLabel</code> , <code>xLabel</code> and <code>yLabel</code> . . . . .	42
15.6 <code>tickstyle</code> . . . . .	42
15.7 <code>ticks</code> . . . . .	43
15.8 <code>labels</code> . . . . .	44
15.9 <code>ticksize</code> , <code>xticksize</code> , <code>yticksize</code> . . . . .	45
15.10 <code>subticks</code> . . . . .	46
15.11 <code>subticksize</code> , <code>xsubticksize</code> , <code>ysubticksize</code> . . . . .	47
15.12 <code>tickcolor</code> , <code>subtickcolor</code> . . . . .	48
15.13 <code>ticklinestyle</code> and <code>subticklinestyle</code> . . . . .	49
15.14 <code>loglines</code> . . . . .	49
15.15 <code>xylogBase</code> , <code>xlogBase</code> and <code>ylogBase</code> . . . . .	51
15.15.1 <code>xylogBase</code> . . . . .	51
15.15.2 <code>ylogBase</code> . . . . .	52
15.15.3 <code>xlogBase</code> . . . . .	54
15.15.4 No logstyle ( <code>xylogBase={}</code> ) . . . . .	55
15.16 <code>subticks</code> , <code>tickwidth</code> and <code>subtickwidth</code> . . . . .	56
15.17 <code>xlabelFactor</code> and <code>ylabelFactor</code> . . . . .	61
15.18 Plot style <code>bar</code> and option <code>barwidth</code> . . . . .	62
15.19 New options for <code>\readdata</code> . . . . .	63

<b>16 New options for <code>\listplot</code></b>	<b>63</b>
16.1 Example for <code>nStep/xStep</code> . . . . .	64
16.2 Example for <code>nStart/xStart</code> . . . . .	65
16.3 Example for <code>nEnd/xEnd</code> . . . . .	66
16.4 Example for all new options . . . . .	66
16.5 Example for <code>xStart</code> . . . . .	67
16.6 Example for <code>yStart/yEnd</code> . . . . .	68
16.7 Example for <code>plotNo/plotNoMax</code> . . . . .	69
<b>17 Polar plots</b>	<b>70</b>
<b>18 New commands and environments</b>	<b>72</b>
18.1 <code>\pstScalePoints</code> . . . . .	72
18.2 <code>psgraph</code> environment . . . . .	73
18.2.1 The new options . . . . .	78
18.3 <code>\resetOptions</code> . . . . .	79
<b>19 Credits</b>	<b>80</b>
<b>20 Change log</b>	<b>80</b>

# Part I

## pstricks

### 1 Numeric functions

All macronames contain a @ in their name, because they are only for internal use, but it is no problem to use it as the other macros. One can define another name without an @:

```
\makeatletter
\let\pstdivide\pst@divide
\makeatother
```

or put the macro inside of the \makeatletter – \makeatother sequence.

#### 1.1 \pst@divide

pstricks itself has its own divide macro, called \pst@divide which can divide two lengths and saves the quotient as a floating point number:

```
\pst@divide{<dividend>}{<divisor>}{<result as a macro>}
```

```
5.66666
-0.17647
1 \makeatletter
2 \pst@divide{34pt}{6pt}\quotient \quotient\
3 \pst@divide{-6pt}{34pt}\quotient \quotient
4 \makeatother
```

this gives the output 5.66666. The result is not a length!

#### 1.2 \pst@mod

pstricks-add defines an additional numeric function for the modulus:

```
\pst@mod{<integer>}{<integer>}{<result as a macro>}
```

```
4
1
2 \makeatletter
3 \pst@mod{34}{6}\modulo \modulo\
4 \pst@mod{25}{-6}\modulo \modulo
5 \makeatother
```

this gives the output 4. Using this internal numeric functions in documents requires a setting inside the makeatletter and makeatother environment. It makes some sense to define a new macroname in the preamble to use it throughou, e.g. \let\modulo\pst@mod.

### 1.3 \pst@max

`\pst@max{<integer>}{<integer>}{<result as count register>}`

```
-6      1 \newcount\maxNo
11      2 \makeatletter
        3 \pst@max{-34}{-6}\maxNo \the\maxNo\\
        4 \pst@max{0}{11}\maxNo \the\maxNo
        5 \makeatother
```

### 1.4 \pst@maxdim

`\pst@maxdim{<dimension>}{<dimension>}{<result as dimension register>}`

```
1234.0pt 1 \newdimen\maxDim
967.39369pt 2 \makeatletter
           3 \pst@maxdim{34cm}{1234pt}\maxDim \the\maxDim\\
           4 \pst@maxdim{34cm}{123pt}\maxDim \the\maxDim
           5 \makeatother
```

### 1.5 \pst@abs

`\pst@abs{<integer>}{<result as a count register>}`

```
34      1 \newcount\absNo
4        2 \makeatletter
        3 \pst@abs{-34}\absNo \the\absNo\\
        4 \pst@abs{4}\absNo \the\absNo
        5 \makeatother
```

### 1.6 \pst@absdim

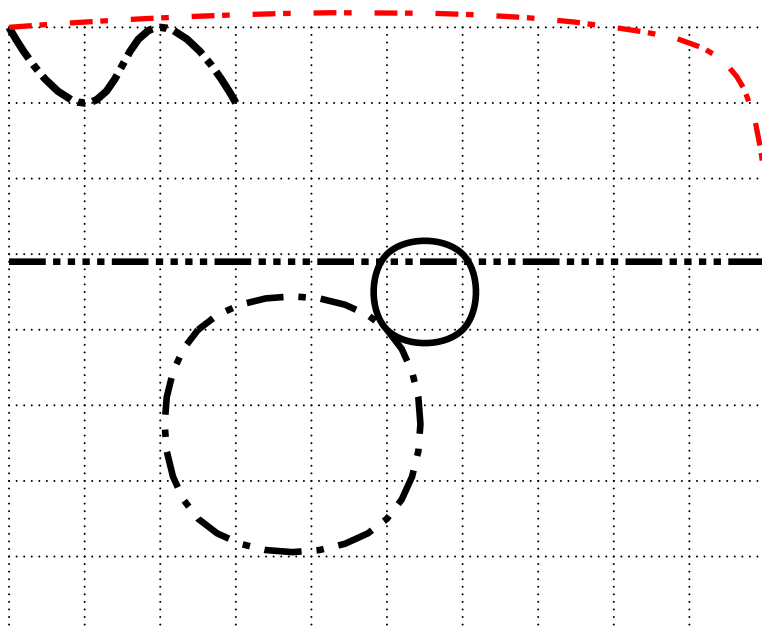
`\pst@absdim{<dimension>}{<result as a dimension register>}`

```
967.39369pt 1 \newdimen\absDim
0.00006pt 2 \makeatletter
          3 \pst@absdim{-34cm}\absDim \the\absDim\\
          4 \pst@absdim{4sp}\absDim \the\absDim
          5 \makeatother
```

## 2 Dashed Lines

Tobias Nähring implemented an enhanced feature for dashed lines. The number of arguments is no more limited.

`dash=value1[unit] value2[unit] ...`



```

1 \begin{pspicture}(-5,-4)(5,4)
2   \psset{linewidth=2.5pt}
3   \psgrid[subgriddiv=0,griddots=10,gridlabels=0pt]
4   \psset{linestyle=dashed}
5   \pscurve[dash=5mm 1mm 1mm 1mm,linewidth=0.1](-5,4)(-4,3)(-3,4)(-2,3)
6   \psline[dash=5mm 1mm 1mm 1mm 1mm 1mm 1mm 1mm 1mm 1mm](-5,0.9)(5,0.9)
7   \psccurve[linestyle=solid](0,0)(1,0)(1,1)(0,1)
8   \psccurve[linestyle=dashed,dash=5mm 2mm 0.1 0.2,linetype=0](0,0)(-2.5,0)
9     (-2.5,-2.5)(0,-2.5)
10  \pscurve[dash=3mm 3mm 1mm 1mm,linecolor=red,linewidth=2pt](5,-4)(5,2)
    (4.5,3.5)(3,4)(-5,4)
11 \end{pspicture}

```

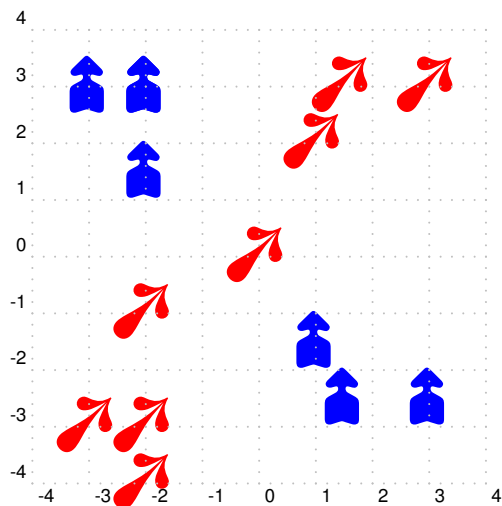
## 3 \rmultiput, a multiple \rput

PSTricks already knows a `multirput`, which puts a box  $n$  times with a difference of  $dx$  and  $dy$  relative to each other. It is not possible to put it with a different distance from one point to the next one. This is possible with `rmultiput`:

```

\rmultiput[<options>]{<any material>}(x1,y1)(x2,y2) ... (xn,yn)
\rmultiput*[<options>]{<any material>}(x1,y1)(x2,y2) ... (xn,yn)

```



```

1 \psset{unit=0.75}
2 \begin{pspicture}(-4,-4)(4,4)
3 \rmultitup[rot=45]{\red\psscalebox{3}{\ding
  {250}}}%
4 (-2,-4)(-2,-3)(-3,-3)(-2,-1)(0,0)(1,2)
  (1.5,3)(3,3)
5 \rmultitup[rot=90,ref=1C]{\blue\psscalebox
  {2}{\ding{253}}}%
6 (-2,2.5)(-2,2.5)(-3,2.5)(-2,1)(1,-2)
  (1.5,-3)(3,-3)
7 \psgrid[subgriddiv=0,gridcolor=lightgray]
8 \end{pspicture}

```

## 4 \pslineII Colored lines

The dashed lines are by default black and white lines. The new macro `\pslineII` offers two-color lines and has the same syntax as `\psline`.



```

1 \begin{pspicture}(0,-0.5)(12,0.5)
2 \pslineII[linewidth=5pt,arrowscale=2]{o-o}(0,0)(12,0)
3 \end{pspicture}

```

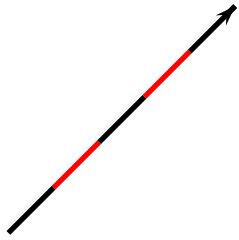
### 4.1 The options

name	meaning
<code>dashColorI</code>	first color, default is black
<code>dashColorII</code>	second color, default is red
<code>dashNo</code>	the difference in per cent of the colored lines, default is 0.2
<code>linecap</code>	how two lines are connected. 0: no modification 1: rounded edges 2: an additional half square at both ends

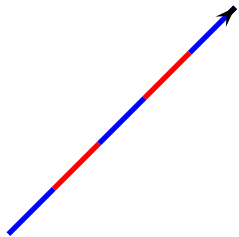
`dashNo` can have values greater than 1. In this case the value will be taken as an absolute width in the pt unit. Only this unit is possible!



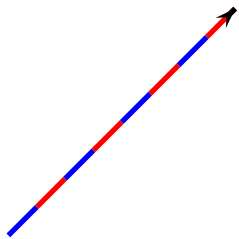
## 4.2 Examples



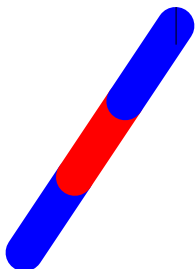
```
1 \psset{linewidth=2pt}
2 \begin{pspicture}(3,3)
3   \pslineII{->}(0,0)(3,3)
4 \end{pspicture}
```



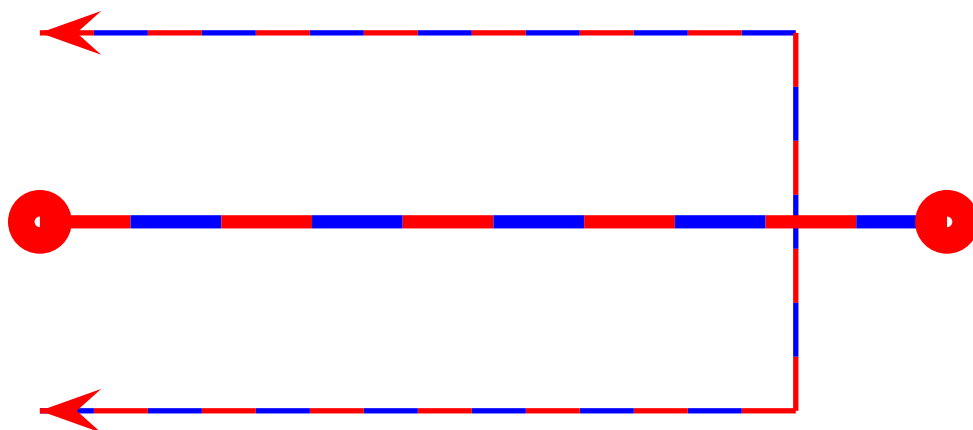
```
1 \psset{linewidth=2pt}
2 \begin{pspicture}(3,3)
3   \pslineII[dashColorI=blue]{->}(0,0)(3,3)
4 \end{pspicture}
```



```
1 \psset{linewidth=2pt}
2 \begin{pspicture}(3,3)
3   \pslineII[dashColorI=blue,dashNo=15]{->}(0,0)(3,3)
4 \end{pspicture}
```



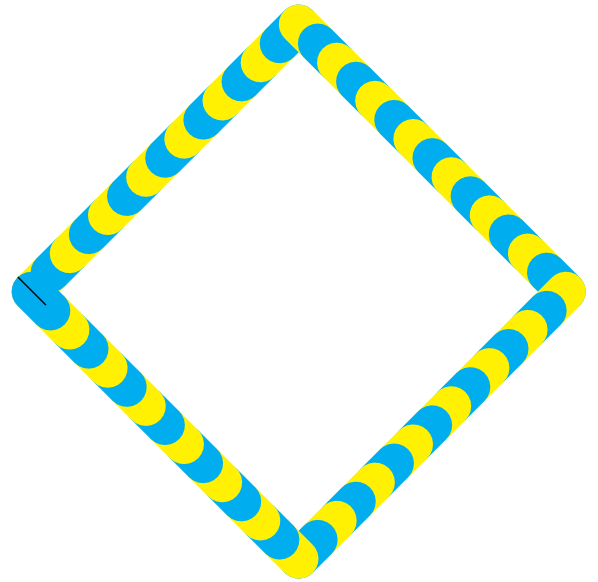
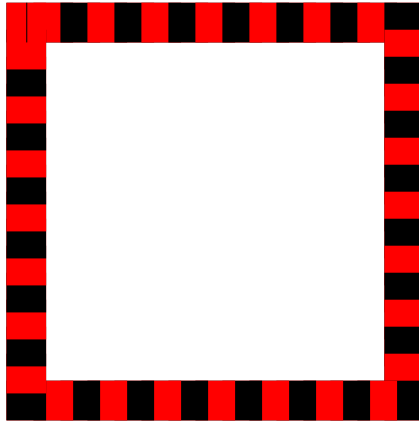
```
1 \psset{linewidth=2pt}
2 \begin{pspicture}(3,3)
3   \pslineII[dashColorI=blue,linecap=1,%
4     dashNo=0.3,linewidth=0.5](0,0)(2,3)
5 \end{pspicture}
```



```

1 \psset{linecolor=red,arrowscale=3}
2 \psset{dashColorI=red,dashColorII=blue,dashNo=20,linewidth=2pt}
3 \begin{pspicture}(0,0)(12,-5)
4 \pslineII{<->}(0,0)(10,0)(10,-5)(0,-5)
5 \pslineII[linewidth=5pt,%
6   dashNo=0.1,arrowscale=2]{o-o}(0,-2.5)(12,-2.5)
7 \end{pspicture}

```



```

1 \psset{linewidth=15pt,dashNo=10}
2 \begin{pspicture}(0,1)(10,-6)
3   \pslineII[linecap=2](0,0)(5,0)(5,-5)(0,-5)(0,0)
4   \rput{45}(7,-2.5){%
5     \pslineII[linecap=1,dashColorI=yellow,%
6       dashColorII=cyan](0,0)(5,0)(5,-5)(0,-5)(0,0)%
7   }
8 \end{pspicture}

```

## 5 \pslineIII Variable linewidth

By default all lines have a fixed width. `\pslineIII` allows to define the start and the end width of a line. It has the same syntax as `\psline`.



```

1 \pslineIII[wBegin=1cm,wEnd=0.3cm,linecolor=cyan](0,0)(12,0)

```

## 5.1 The options

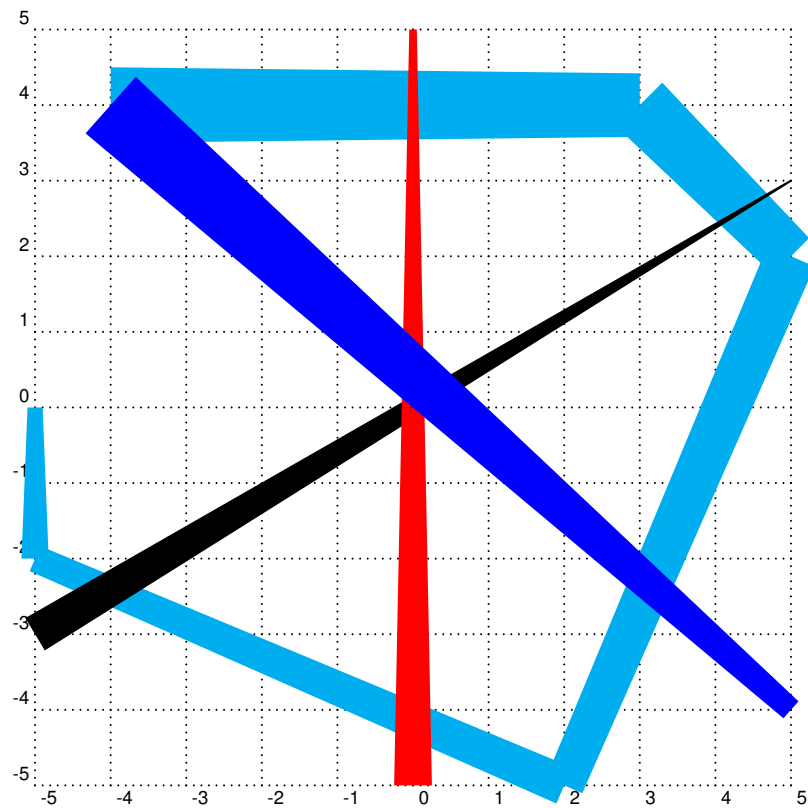
name	meaning
wBegin	first width, default is <code>\pslinewidth</code>
wEnd	last width, default is <code>\pslinewidth</code>

It is also possible to use `pslineIII` with more than two coordinates, like



```
1 \pslineIII[wBegin=1cm,wEnd=0.1cm,linecolor=cyan](0,0)(0,1.5)(12,1.5)
   (12,0)
```

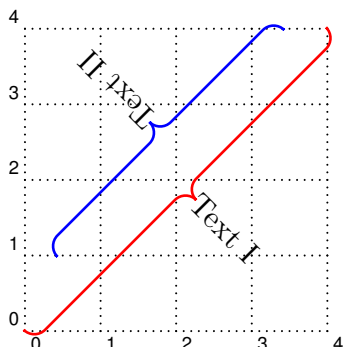
## 5.2 Examples



## 6 \psbrace

### 6.1 Syntax

`\psbrace[<options>](<A>)(<B>){<text>}`



```

1 \begin{pspicture}(4,4)
2 \psgrid[subgriddiv=0,griddots=10]
3 \pnode(0,0){A}
4 \pnode(4,4){B}
5 \psbrace[linecolor=red,ref=lC](A)(B){Text I}
6 \psbrace[linecolor=blue,ref=lC](3,4)(0,1){Text II}
7 \end{pspicture}

```

The option `\specialCoor` is enabled, so that all types of coordinates are possible, (nodename),  $(x,y)$ ,  $(nodeA|nodeB)$ , ...

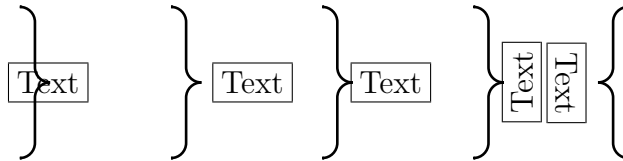
### 6.2 Options

Additional to all other available options from `pstricks` or the other related packages, there are two new option, named `braceWidth` and `bracePos`. All important ones are shown in the following table.

name	meaning
<code>braceWidth</code>	default is 0.35
<code>bracePos</code>	relative position (default is 0.5)
<code>lineararc</code>	absolute value for the arcs (default is 2mm)
<code>nodesepA</code>	x-separation (default is 0pt)
<code>nodesepB</code>	y-separation (default is 0pt)
<code>rot</code>	additional rotating for the text (default is 0)
<code>ref</code>	reference point for the text (default is c)

By default the text is written perpendicular to the brace line and can be changed with the `pstricks` option `rot=...`. The text parameter can take any object and may also be empty. The reference point can be any value of the combination of **l** (left) or **r** (right) and **b** (bottom) or **B** (Baseline) or **C** (center) or **t** (top), where the default is **c**, the center of the object.

## 6.3 Examples



```

1 \begin{pspicture}(8,2.5)
2 \psbrace(0,0)(0,2){\fbox{Text}}%
3 \psbrace[nodesepA=20pt](2,0)(2,2){\fbox{Text}}
4 \psbrace[ref=lC](4,0)(4,2){\fbox{Text}}
5 \psbrace[ref=lt,rot=90,nodesepB=-15pt](6,0)(6,2){\fbox{Text}}
6 \psbrace[ref=lt,rot=90,nodesepA=-5pt,nodesepB=15pt](8,2)(8,0){\fbox{Text}}
7 \end{pspicture}

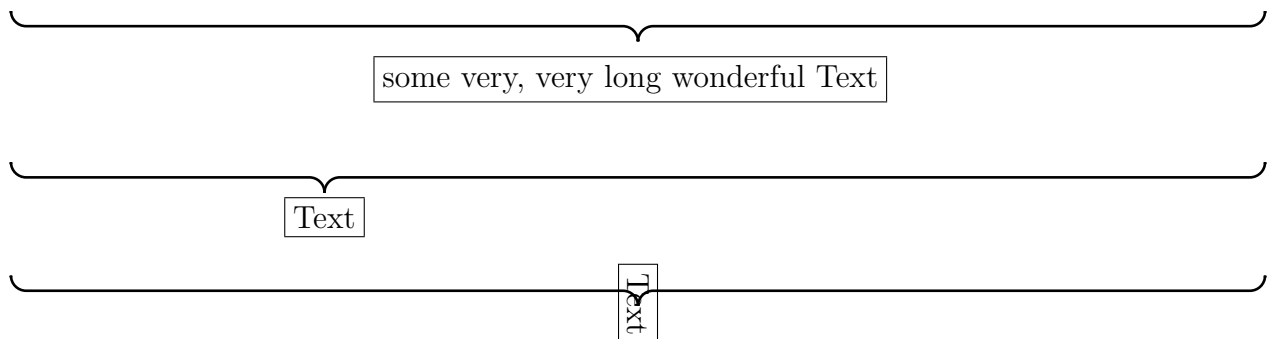
```

$$\left. \int_1^{\infty} \frac{1}{x^2} dx = 1 \right\} \quad \left. \int_1^{\infty} \frac{1}{x^2} dx = \int_1^{\infty} \frac{1}{x^2} dx = \int_1^{\infty} \frac{dx}{x^2} = \int_1^{\infty} \frac{1}{x^2} dx \right\}$$

```

1 \def\someMath{\int\limits_1^{\infty}\frac{1}{x^2}\,dx=1$}
2 \begin{pspicture}(8,2.5)
3 \psbrace(0,0)(0,2){\someMath}%
4 \psbrace[nodesepA=30pt](2,0)(2,2){\someMath}
5 \psbrace[ref=lC](4,0)(4,2){\someMath}
6 \psbrace[ref=lt,rot=90,nodesepB=-30pt](6,0)(6,2){\someMath}
7 \psbrace[ref=lt,rot=90,nodesepB=30pt](8,2)(8,0){\someMath}
8 \end{pspicture}

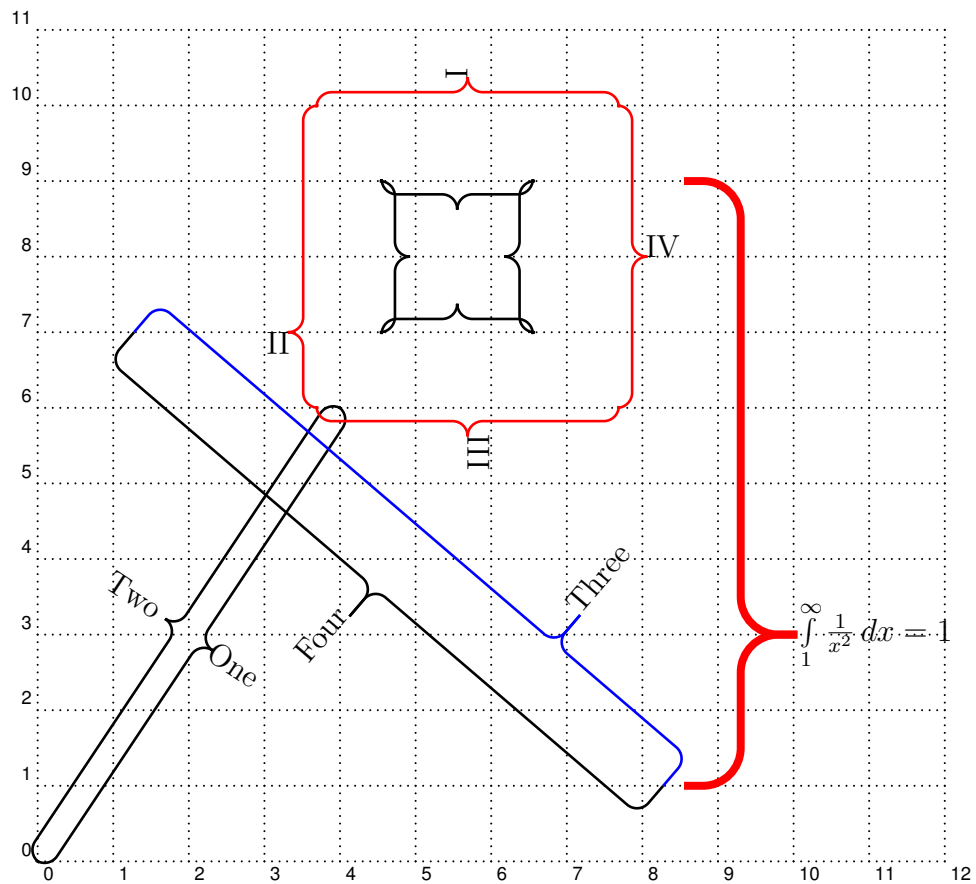
```



```

1 \begin{pspicture}(\linewidth,5)
2 \psbrace(0,0.5)(\linewidth,0.5){\fbox{Text}}%
3 \psbrace[bracePos=0.25,nodesepB=-10pt,rot=90](0,2)(\linewidth,2){\fbox{
  Text}}
4 \psbrace[ref=lC,nodesepA=-3.5cm,nodesepB=-15pt,rot=90](0,4)(\linewidth,4){
  %
  \fbox{some very, very long wonderful Text}}
5 \end{pspicture}
6

```



```

1 \def\someMath{\int\limits_1^{\infty}\frac{1}{x^2}\,,dx=1$}
2 \begin{pspicture}(12,11)
3 \psgrid[subgriddiv=0,griddots=10]
4 \pnode(0,0){A}
5 \pnode(4,6){B}
6 \psbrace[ref=lC](A)(B){One}
7 \psbrace[rot=180,nodesepA=-5pt,ref=rb](B)(A){Two}
8 \psbrace[linecolor=blue,bracePos=0.25,braceWidth=1,ref=lB](8,1)(1,7){Three
9 }
10 \psbrace[braceWidth=-1,rot=180,ref=rB](8,1)(1,7){Four}
11 \psbrace[lineararc=0.5,linecolor=red,linewidth=3pt,braceWidth=1.5,%
12 bracePos=0.25,ref=lC](8,1)(8,9){\someMath}
13 \psbrace(4,9)(6,9){}
14 \psbrace(6,9)(6,7){}
15 \psbrace(6,7)(4,7){}
16 \psbrace(4,7)(4,9){}
17 \psset{linecolor=red}
18 \psbrace[ref=lB](7,10)(3,10){I}
19 \psbrace[ref=lB,bracePos=0.75](3,10)(3,6){II}
20 \psbrace[ref=lB](3,6)(7,6){III}
21 \psbrace[ref=lB](7,6)(7,10){IV}
22 \end{pspicture}

```

It is also possible to put a vertical brace around a default paragraph. This works with setting two invisible nodes at the beginning and the end of the paragraph. Indentation is possible with a minipage.

Some nonsense text, which is nothing more than nonsense. Some nonsense text, which is nothing more than nonsense.

Some nonsense text, which is nothing more than nonsense. Some nonsense text, which is nothing more than nonsense. Some nonsense text, which is nothing more than nonsense. Some nonsense text, which is nothing more than nonsense. Some nonsense text, which is nothing more than nonsense. Some nonsense text, which is nothing more than nonsense.

Some nonsense text, which is nothing more than nonsense. Some nonsense text, which is nothing more than nonsense.

Some nonsense text, which is nothing more than nonsense. Some nonsense text, which is nothing more than nonsense. Some nonsense text, which is nothing more than nonsense. Some nonsense text, which is nothing more than nonsense. Some nonsense text, which is nothing more than nonsense. Some nonsense text, which is nothing more than nonsense.

```

1 \begin{framed}
2 Some nonsense text, which is nothing more than nonsense.
3 Some nonsense text, which is nothing more than nonsense.
4
5 \noindent\rnode{A}{}
6
7 \vspace*{-1ex}
8 Some nonsense text, which is nothing more than nonsense.
9 Some nonsense text, which is nothing more than nonsense.
10 Some nonsense text, which is nothing more than nonsense.
11 Some nonsense text, which is nothing more than nonsense.
12 Some nonsense text, which is nothing more than nonsense.
13 Some nonsense text, which is nothing more than nonsense.
14 Some nonsense text, which is nothing more than nonsense.
15 Some nonsense text, which is nothing more than nonsense.
16
17 \vspace*{-2ex}
18 \noindent\rnode{B}{}\psbrace[linecolor=red](A)(B){}
19
20 Some nonsense text, which is nothing more than nonsense.
21 Some nonsense text, which is nothing more than nonsense.
22
23 \medskip
24 \hfill\begin{minipage}{0.95\linewidth}
25 \noindent\rnode{A}{}
26
27 \vspace*{-1ex}
28 Some nonsense text, which is nothing more than nonsense.
29 Some nonsense text, which is nothing more than nonsense.
30 Some nonsense text, which is nothing more than nonsense.
31 Some nonsense text, which is nothing more than nonsense.
32 Some nonsense text, which is nothing more than nonsense.
33 Some nonsense text, which is nothing more than nonsense.
34 Some nonsense text, which is nothing more than nonsense.
35 Some nonsense text, which is nothing more than nonsense.
36
37 \vspace*{-2ex}
38 \noindent\rnode{B}{}\psbrace[linecolor=red](A)(B){}
39 \end{minipage}
40 \end{framed}



















```



## 7 Arrows

### 7.1 Definition

`pstricks-add` defines the following "arrows":

Value	Example	Name
-		None
<->		Arrowheads.
>-<		Reverse arrowheads.
<<->>		Double arrowheads.
>>-<<		Double reverse arrowheads.
-		T-bars, flush to endpoints.
* -   *		T-bars, centered on endpoints.
[ - ]		Square brackets.
] - [		Reversed square brackets.
( - )		Rounded brackets.
) - (		Reversed rounded brackets.
o - o		Circles, centered on endpoints.
* - *		Disks, centered on endpoints.
oo - oo		Circles, flush to endpoints.
** - **		Disks, flush to endpoints.
<->		T-bars and arrows.
>-<		T-bars and reverse arrows.
H - H		left/right hook arrows.

You can also mix and match, e.g., `->`, `*->` and `[->` are all valid values of the `arrows` parameter. The parameter can be set with

```
\psset{arrows=<type>}
```

or for some macros with a special option, like

```
\psline[<general options>]{<arrow type>}(A)(B)
```

```
\psline[linecolor=red,linewidth=2pt]{|->}(0,0)(0,2)
```



### 7.2 Multiple arrows

There are two new options which are only valid for the arrow type `<<` or `>>`. `nArrow` sets both, the `nArrowA` and the `nArrowB` parameter. The meaning is declared in the following tables. Without setting one of these parameters the behaviour is like the one described in the old PSTricks manual.

Value	Meaning
->>	-A
<<->>	A-A
<<-	A-
>>-	B-
-<<	-B
>>-<<	B-B
>>->>	B-A
<<-<<	A-B

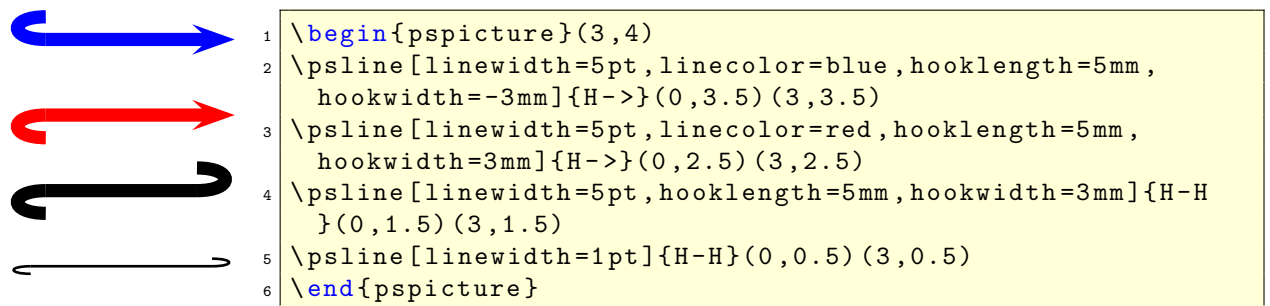
Value	Example
<code>\psline{-&gt;&gt;}(0,1ex)(2.3,1ex)</code>	
<code>\psline[nArrowsA=3]{-&gt;&gt;}(0,1ex)(2.3,1ex)</code>	
<code>\psline[nArrowsA=5]{-&gt;&gt;}(0,1ex)(2.3,1ex)</code>	
<code>\psline{&lt;&lt;-}(0,1ex)(2.3,1ex)</code>	
<code>\psline[nArrowsA=3]{&lt;&lt;-}(0,1ex)(2.3,1ex)</code>	
<code>\psline[nArrowsA=5]{&lt;&lt;-}(0,1ex)(2.3,1ex)</code>	
<code>\psline{&lt;&lt;-&gt;&gt;}(0,1ex)(2.3,1ex)</code>	
<code>\psline[nArrowsA=3]{&lt;&lt;-&gt;&gt;}(0,1ex)(2.3,1ex)</code>	
<code>\psline[nArrowsA=5]{&lt;&lt;-&gt;&gt;}(0,1ex)(2.3,1ex)</code>	
<code>\psline{&lt;&lt;- }(0,1ex)(2.3,1ex)</code>	
<code>\psline[nArrowsA=3]{&lt;&lt;-&lt;&lt;}(0,1ex)(2.3,1ex)</code>	
<code>\psline[nArrowsA=5]{&lt;&lt;-o}(0,1ex)(2.3,1ex)</code>	
<code>\psline[nArrowsA=3,nArrowsB=4]{&lt;&lt;-&lt;&lt;}(0,1ex)(2.3,1ex)</code>	
<code>\psline[nArrowsA=3,nArrowsB=4]{&gt;&gt;-&gt;&gt;}(0,1ex)(2.3,1ex)</code>	
<code>\psline[nArrowsA=1,nArrowsB=4]{&gt;&gt;-&gt;&gt;}(0,1ex)(2.3,1ex)</code>	

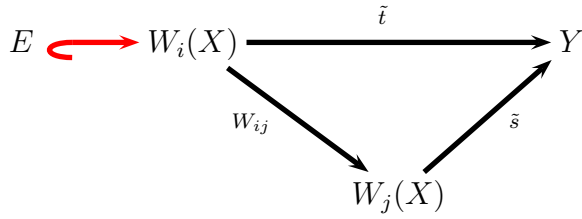
### 7.3 hookrightarrow and hookleftarrow

This is another type of an arrow and abbreviated with H. The length and width of the hook is set by the new options `hooklength` and `hookwidth`, which are by default set to

```
\psset{hooklength=3mm,hookwidth=1mm}
```

If the line begins with a right hook then the line ends with a left hook and vice versa:





```


1 $\begin{psmatrix}
2 E&W_i(X)&&Y\\
3 &&W_j(X)
4 \psset{arrows=->,nodesep=3pt,linewidth
5 =2pt}
6 \everypsbox{\scriptstyle}
7 \ncline[linecolor=red,arrows=H->,%
8 hooklength=4mm,hookwidth=2mm
9 ]{1,1}{1,2}
10 \ncline{1,2}{1,4}~{\tilde{t}}
11 \ncline{1,2}{2,3}<{W_{ij}}
12 \ncline{2,3}{1,4}>{\tilde{s}}
13 \end{psmatrix}$

```

## 7.4 ArrowInside Option

It is now possible to have arrows inside the lines and not only at the beginning or the end. The new defined options

Name	Example	Output
ArrowInside	<code>\psline[ArrowInside=-&gt;](0,0)(2,0)</code>	
ArrowInsidePos	<code>\psline[ArrowInside=-&gt;,% ArrowInsidePos=0.25](0,0)(2,0)</code>	
ArrowInsidePos	<code>\psline[ArrowInside=-&gt;,% ArrowInsidePos=10](0,0)(2,0)</code>	
ArrowInsideNo	<code>\psline[ArrowInside=-&gt;,% ArrowInsideNo=2](0,0)(2,0)</code>	
ArrowInsideOffset	<code>\psline[ArrowInside=-&gt;,% ArrowInsideNo=2,% ArrowInsideOffset=0.1](0,0)(2,0)</code>	
ArrowInside	<code>\psline[ArrowInside=-&gt;]{-&gt;}(0,0)(2,0)</code>	
ArrowInsidePos	<code>\psline[ArrowInside=-&gt;,% ArrowInsidePos=0.25]{-&gt;}(0,0)(2,0)</code>	
ArrowInsidePos	<code>\psline[ArrowInside=-&gt;,% ArrowInsidePos=10]{-&gt;}(0,0)(2,0)</code>	
ArrowInsideNo	<code>\psline[ArrowInside=-&gt;,% ArrowInsideNo=2]{-&gt;}(0,0)(2,0)</code>	
ArrowInsideOffset	<code>\psline[ArrowInside=-&gt;,% ArrowInsideNo=2,% ArrowInsideOffset=0.1]{-&gt;}(0,0)(2,0)</code>	
ArrowFill	<code>\psline[ArrowFill=false,% arrowinset=0]{-&gt;}(0,0)(2,0)</code>	
ArrowFill	<code>\psline[ArrowFill=false,% arrowinset=0]{&lt;&lt;-&gt;}(0,0)(2,0)</code>	

Name	Example	Output
ArrowFill	<pre>\psline[ArrowInside=-&gt;,%   arrowinset=0,%   ArrowFill=false,%   ArrowInsideNo=2,%   ArrowInsideOffset=0.1]{-&gt;}(0,0)(2,0)</pre>	

Without the default arrow definition there is only the one inside the line, defined by the type and the position. The position is relative to the length of the whole line. 0.25 means at 25% of the line length. The peak of the arrow gets the coordinates which are calculated by the macro. If you want arrows with an absolute position difference, then choose a value greater than 1, e.g. 10 which places an arrow every 10 pt. The default unit pt cannot be changed.

## 7.5 ArrowFill Option

By default all arrows are filled polygons. With the option **ArrowFill=false** there are "white" arrows. Only for the beginning/end arrows they are empty, the inside arrows are overpainted with the line.



```
1 \psset{arrowscale=3}
2 \psline[linecolor=red,arrowinset=0]{<->}(0,0)(3,0)
```



```
1 \psset{arrowscale=3}
2 \psline[linecolor=red,arrowinset=0,ArrowFill=false]
  {<->}(0,0)(3,0)
```



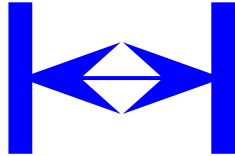
```
1 \psset{arrowscale=3}
2 \psline[linecolor=red,arrowinset=0,arrowsize=0.2,
  ArrowFill=false]{<->}(0,0)(3,0)
```



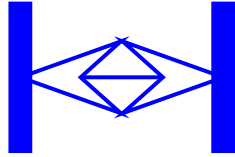
```
1 \psset{arrowscale=3}
2 \psline[linecolor=blue,arrowscale=6,ArrowFill=true]
  {>>->>}(0,0)(3,0)
```



```
1 \psset{arrowscale=3}
2 \psline[linecolor=blue,arrowscale=6,ArrowFill=false]
  {>>->>}(0,0)(3,0)
3 \rule{3cm}{0pt}\\[30pt]
```



```
1 \psset{arrowscale=3}
2 \psline[linecolor=blue,arrowscale=6,ArrowFill=true]
  {>|->|}(0,0)(3,0)
```

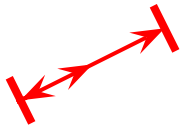


```
1 \psset{arrowscale=3}
2 \psline[linecolor=blue,arrowscale=6,ArrowFill=false]
  {>|->|}(0,0)(3,0) %
```

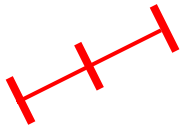
## 7.6 Examples

All examples are printed with `\psset{arrowscale=2,linecolor=red}`.

### 7.6.1 `\psline`



```
1 \begin{pspicture}(2,2)
2 \psset{arrowscale=2,ArrowFill=true}
3 \psline[ArrowInside=->]{|<->|}(2,1)
4 \end{pspicture}
```



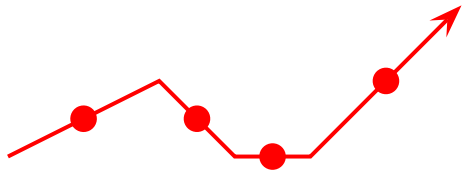
```
1 \begin{pspicture}(2,2)
2 \psset{arrowscale=2,ArrowFill=true}
3 \psline[ArrowInside=-|]{|-|}(2,1)
4 \end{pspicture}
```



```
1 \begin{pspicture}(2,2)
2 \psset{arrowscale=2,ArrowFill=true}
3 \psline[ArrowInside=->,ArrowInsideNo=2]{->}(2,1)
4 \end{pspicture}
```



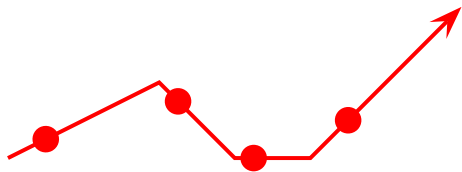
```
1 \begin{pspicture}(2,2)
2 \psset{arrowscale=2,ArrowFill=true}
3 \psline[ArrowInside=->,ArrowInsideNo=2,ArrowInsideOffset
  =0.1]{->}(2,1)
4 \end{pspicture}
```



```

1 \begin{pspicture}(6,2)
2 \psset{arrowscale=2,ArrowFill=true}
3 \psline[ArrowInside=-*]{->}(0,0)(2,1)(3,0)
4   (4,0)(6,2)
5 \end{pspicture}

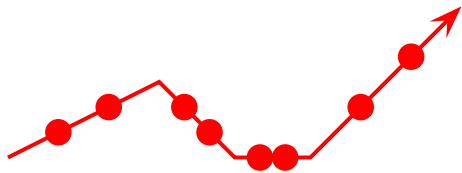
```



```

1 \begin{pspicture}(6,2)
2 \psset{arrowscale=2,ArrowFill=true}
3 \psline[ArrowInside=-*,ArrowInsidePos
4   =0.25]{->}(0,0)(2,1)(3,0)(4,0)(6,2)
5 \end{pspicture}

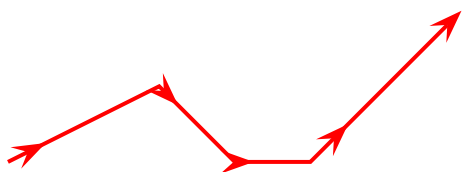
```



```

1 \begin{pspicture}(6,2)
2 \psset{arrowscale=2,ArrowFill=true}
3 \psline[ArrowInside=-*,ArrowInsidePos
4   =0.25,ArrowInsideNo=2]{->}%
5   (0,0)(2,1)(3,0)(4,0)(6,2)
6 \end{pspicture}

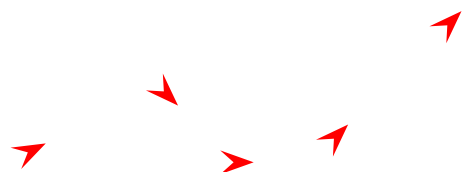
```



```

1 \begin{pspicture}(6,2)
2 \psset{arrowscale=2,ArrowFill=true}
3 \psline[ArrowInside=->,ArrowInsidePos
4   =0.25]{->}%
5   (0,0)(2,1)(3,0)(4,0)(6,2)
6 \end{pspicture}

```



```

1 \begin{pspicture}(6,2)
2 \psset{arrowscale=2,ArrowFill=true}
3 \psline[linestyle=None,ArrowInside=->,
4   ArrowInsidePos=0.25]{->}%
5   (0,0)(2,1)(3,0)(4,0)(6,2)
6 \end{pspicture}

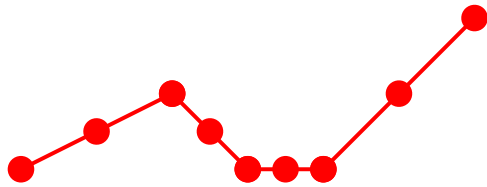
```



```

1 \begin{pspicture}(6,2)
2 \psset{arrowscale=2,ArrowFill=true}
3 \psline[ArrowInside=-<,ArrowInsidePos
4   =0.75]{->}%
5   (0,0)(2,1)(3,0)(4,0)(6,2)
6 \end{pspicture}

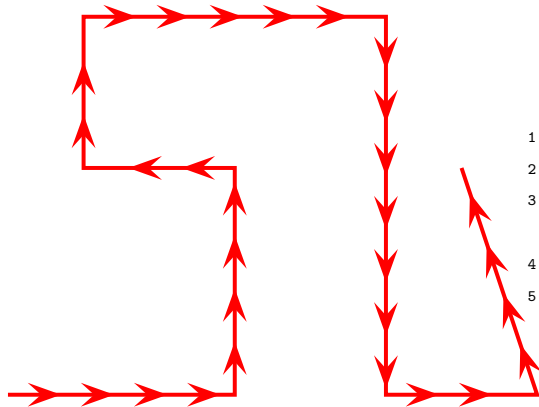
```



```

1 \begin{pspicture}(6,2)
2 \psset{arrowscale=2,ArrowFill=true,
   ArrowInside=-*}
3 \psline(0,0)(2,1)(3,0)(4,0)(6,2)
4 \psset{linestyle=none}
5 \psline[ArrowInsidePos=0](0,0)(2,1)(3,0)
   (4,0)(6,2)
6 \psline[ArrowInsidePos=1](0,0)(2,1)(3,0)
   (4,0)(6,2)
7 \end{pspicture}

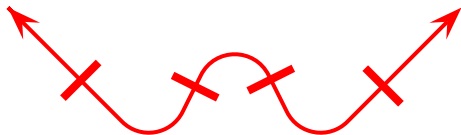
```



```

1 \begin{pspicture}(6,5)
2 \psset{arrowscale=2,ArrowFill=true}
3 \psline[ArrowInside=->,ArrowInsidePos
   =20](0,0)(3,0)%
4       (3,3)(1,3)(1,5)(5,5)(5,0)(7,0)(6,3)
5 \end{pspicture}

```

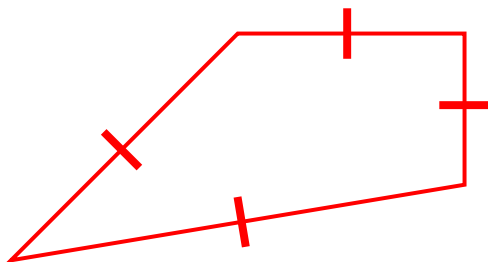


```

1 \begin{pspicture}(6,2)
2 \psset{arrowscale=2,ArrowFill=true}
3 \psline[lineararc=0.5,ArrowInside
   =-|]{<->}(0,2)(2,0)(3,2)(4,0)(6,2)
4 \end{pspicture}

```

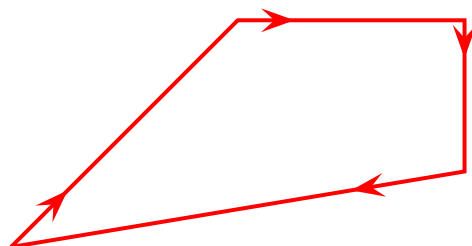
## 7.6.2 \pspolygon



```

1 \begin{pspicture}(6,3)
2 \psset{arrowscale=2}
3 \pspolygon[ArrowInside=-|](0,0)(3,3)(6,3)
   (6,1)
4 \end{pspicture}

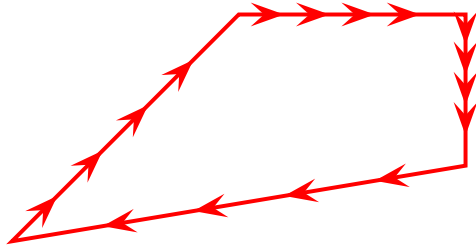
```



```

1 \begin{pspicture}(6,3)
2 \psset{arrowscale=2}
3 \pspolygon[ArrowInside=->,ArrowInsidePos
   =0.25](0,0)(3,3)(6,3)(6,1)
4 \end{pspicture}

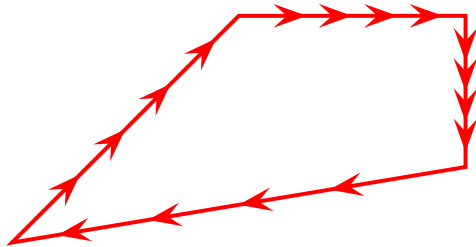
```



```

1 \begin{pspicture}(6,3)
2 \psset{arrowscale=2}
3 \pspolygon[ArrowInside=->,ArrowInsideNo=4]
4 % (0,0)(3,3)(6,3)(6,1)
5 \end{pspicture}

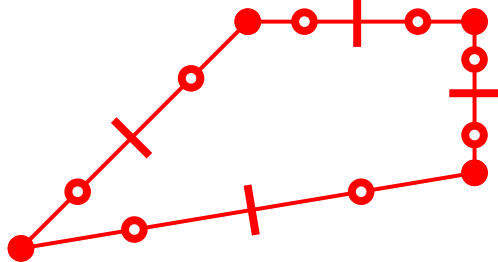
```



```

1 \begin{pspicture}(6,3)
2 \psset{arrowscale=2}
3 \pspolygon[ArrowInside=->,ArrowInsideNo=4,
4 % ArrowInsideOffset=0.1](0,0)(3,3)(6,3)
5 (6,1)
6 \end{pspicture}

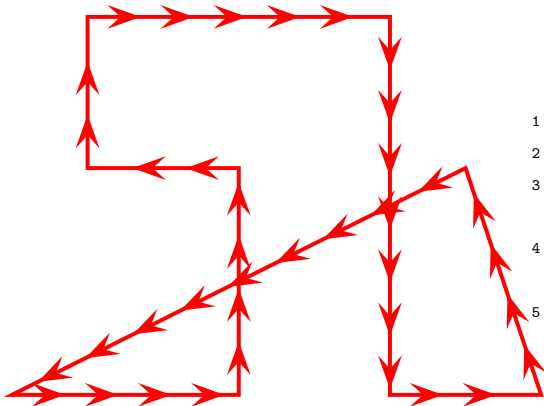
```



```

1 \begin{pspicture}(6,3)
2 \psset{arrowscale=2}
3 \pspolygon[ArrowInside=-|](0,0)(3,3)(6,3)
4 (6,1)
5 \psset{linestyle=none,ArrowInside=-*}
6 \pspolygon[ArrowInsidePos=0](0,0)(3,3)
7 (6,3)(6,1)
8 \pspolygon[ArrowInsidePos=1](0,0)(3,3)
9 (6,3)(6,1)
10 \psset{ArrowInside=-o}
11 \pspolygon[ArrowInsidePos=0.25](0,0)(3,3)
12 (6,3)(6,1)
13 \pspolygon[ArrowInsidePos=0.75](0,0)(3,3)
14 (6,3)(6,1)
15 \end{pspicture}

```



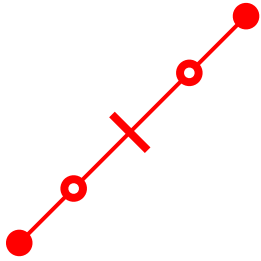
```

1 \begin{pspicture}(6,5)
2 \psset{arrowscale=2}
3 \pspolygon[ArrowInside=->,
4 ArrowInsidePos=20] % (0,0)(3,0)(3,3)(1,3)(1,5)(5,5)(5,0)
5 (7,0)(6,3)
6 \end{pspicture}

```



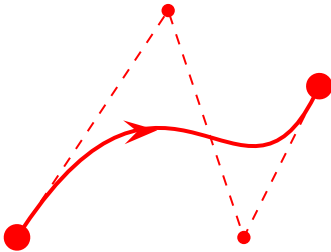
### 7.6.3 \psbezier



```

1 \begin{pspicture}(3,3)
2 \psset{arrowscale=2}
3 \psbezier[ArrowInside=-|](1,1)(2,2)(3,3)
4 \psset{linestyle=none,ArrowInside=-o}
5 \psbezier[ArrowInsidePos=0.25](1,1)(2,2)(3,3)
6 \psbezier[ArrowInsidePos=0.75](1,1)(2,2)(3,3)
7 \psset{linestyle=none,ArrowInside=-*}
8 \psbezier[ArrowInsidePos=0](1,1)(2,2)(3,3)
9 \psbezier[ArrowInsidePos=1](1,1)(2,2)(3,3)
10 \end{pspicture}

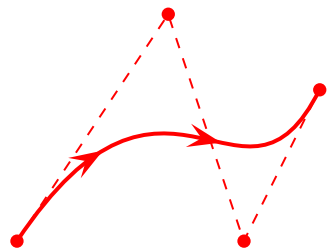
```



```

1 \begin{pspicture}(4,3)
2 \psset{arrowscale=2}
3 \psbezier[ArrowInside=->,showpoints=true]%
4 \psbezier[*-*](2,3)(3,0)(4,2)
5 \end{pspicture}

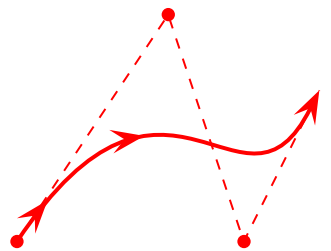
```



```

1 \begin{pspicture}(4,3)
2 \psset{arrowscale=2}
3 \psbezier[ArrowInside=->,showpoints=true,%
4 ArrowInsideNo=2](2,3)(3,0)(4,2)
5 \end{pspicture}

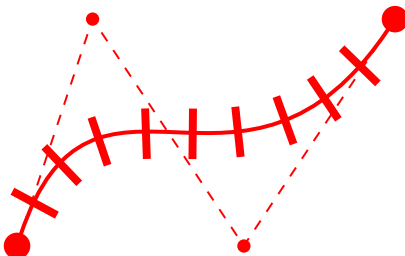
```



```

1 \begin{pspicture}(4,3)
2 \psset{arrowscale=2}
3 \psbezier[ArrowInside=->,showpoints=true,%
4 ArrowInsideNo=2,ArrowInsideOffset
5 =-0.2]{->}(2,3)(3,0)(4,2)
6 \end{pspicture}

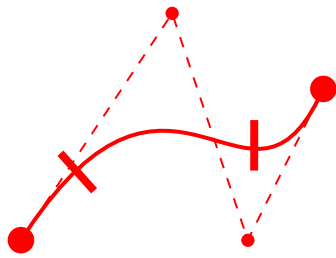
```



```

1 \begin{pspicture}(5,3)
2 \psset{arrowscale=2}
3 \psbezier[ArrowInsideNo=9,ArrowInside=-|,%
4 showpoints=true]{*-*}(1,3)(3,0)(5,3)
5 \end{pspicture}

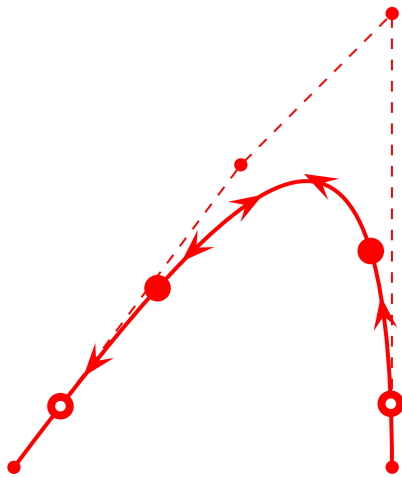
```



```

1 \begin{pspicture}(4,3)
2 \psset{arrowscale=2}
3 \psset{ArrowInside=-|}
4 \psbezier[ArrowInsidePos=0.25,showpoints=true]
5   [*-*](2,3)(3,0)(4,2)
6 \psset{linestyle=none}
7 \psbezier[ArrowInsidePos=0.75](2,3)(3,0)(4,2)
8 \end{pspicture}

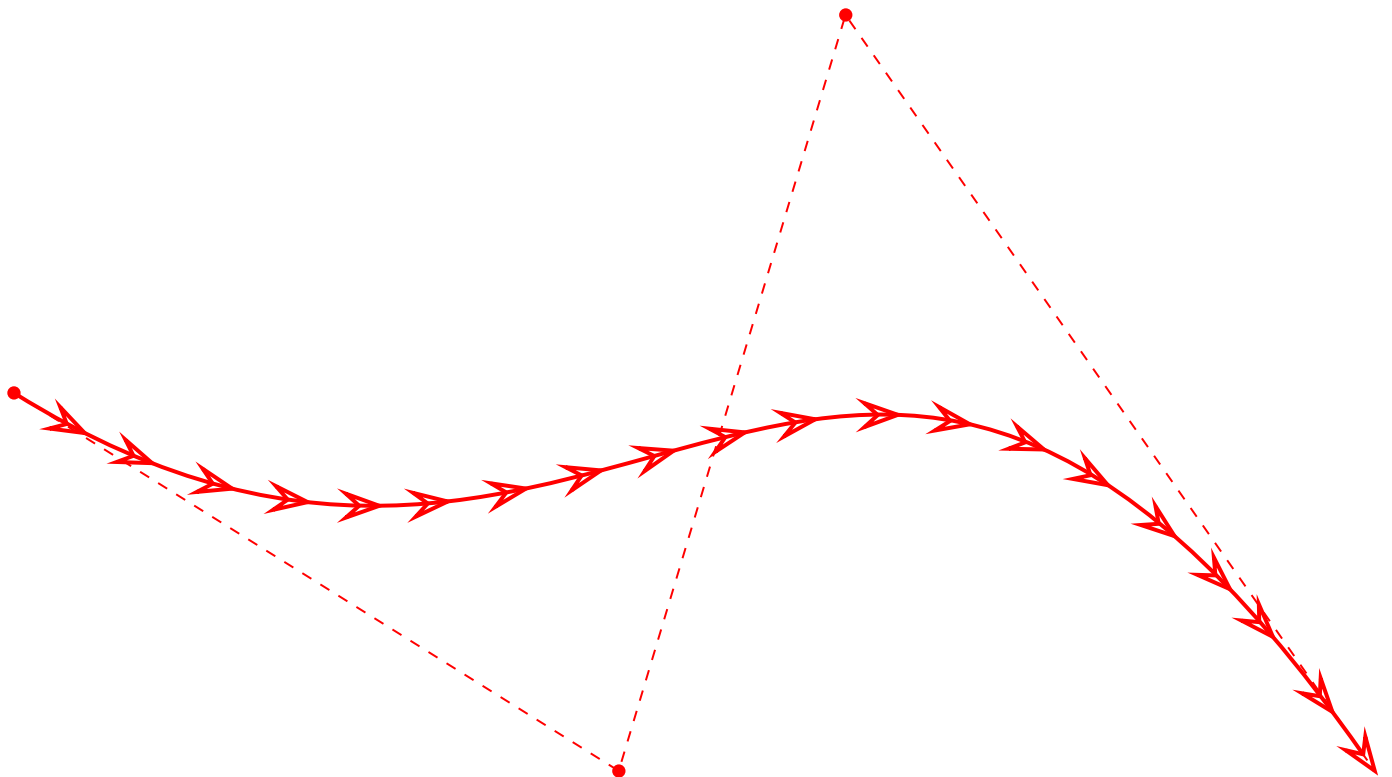
```



```

1 \begin{pspicture}(5,6)
2 \psset{arrowscale=2}
3 \pnode(3,4){A}\pnode(5,6){B}\pnode(5,0){C}
4 \psbezier[ArrowInside=->,%,
5   showpoints=true](A)(B)(C)
6 \psset{linestyle=none,ArrowInside=-<}
7 \psbezier[ArrowInsideNo=4](A)(B)(C)
8 \psset{ArrowInside=-o}
9 \psbezier[ArrowInsidePos=0.1](A)(B)(C)
10 \psbezier[ArrowInsidePos=0.9](A)(B)(C)
11 \psset{ArrowInside=-*}
12 \psbezier[ArrowInsidePos=0.3](A)(B)(C)
13 \psbezier[ArrowInsidePos=0.7](A)(B)(C)
14 \end{pspicture}

```



```

1 \begin{pspicture}(-3,-5)(15,5)
2   \psbezier[ArrowInsideNo=19,%
3     ArrowInside=->,ArrowFill=false,%
4     showpoints=true]{->}(-3,0)(5,-5)(8,5)(15,-5)
5 \end{pspicture}

```

## 7.6.4 \pcline

These examples need the package `pst-node`.



```

1 \begin{pspicture}(2,1)
2   \psset{arrowscale=2}
3   \pcline[ArrowInside=->](0,0)(2,1)
4 \end{pspicture}

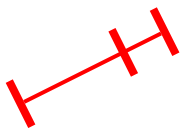
```



```

1 \begin{pspicture}(2,1)
2   \psset{arrowscale=2}
3   \pcline[ArrowInside=->]{<->}(0,0)(2,1)
4 \end{pspicture}

```



```

1 \begin{pspicture}(2,1)
2   \psset{arrowscale=2}
3   \pcline[ArrowInside=-|,ArrowInsidePos=0.75]{|-|}(0,0)(2,1)
4 \end{pspicture}

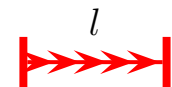
```



```

1 \psset{arrowscale=2}
2 \pcline[ArrowInside=->,ArrowInsidePos=0.65]{*-}(0,0)(2,0)
3 \naput[labelsep=0.3]{\large$g$}

```



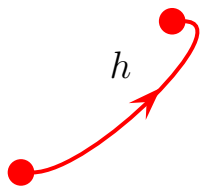
```

1 \psset{arrowscale=2}
2 \pcline[ArrowInside=->,ArrowInsidePos=10]{|-|}(0,0)(2,0)
3 \naput[labelsep=0.3]{\large$l$}

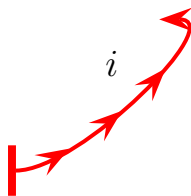
```

### 7.6.5 \pccurve

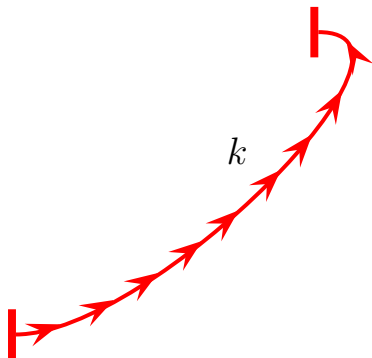
These examples also need the package `pst-node`.



```
1 \begin{pspicture}(2,2)
2 \psset{arrowscale=2}
3 \pccurve[ArrowInside=->,ArrowInsidePos=0.65,showpoints=true]
  {*-}(0,0)(2,2)
4 \naput[labelsep=0.3]{\large$h$}
5 \end{pspicture}
```



```
1 \begin{pspicture}(2,2)
2 \psset{arrowscale=2}
3 \pccurve[ArrowInside=->,ArrowInsideNo=3,showpoints=true]
  {|->}(0,0)(2,2)
4 \naput[labelsep=0.3]{\large$i$}
5 \end{pspicture}
```



```
1 \begin{pspicture}(4,4)
2 \psset{arrowscale=2}
3 \pccurve[ArrowInside=->,ArrowInsidePos=20]{|-|}(0,0)(4,4)
4 \naput[labelsep=0.3]{\large$k$}
5 \end{pspicture}
```

## 8 \psFormatInt

There exist some packages and a lot of code to format an integer like 1 000 000 or 1,234,567 (in Europe 1.234.567). But all packages expect a real number as argument and cannot handle macros as an argument. For this case `pstricks-add` has a macro `psFormatInt` which can handle both:

```
1,234,567 1 \psFormatInt{1234567}\\
1,234,567 2 \psFormatInt[intSeparator={,}]{1234567}\\
1.234.567 3 \psFormatInt[intSeparator=.]{1234567}\\
1-234.567 4 \psFormatInt[intSeparator=$\cdot$]{1234567}\\
965,432 5 \def\temp{965432}
6 \psFormatInt{\temp}
```

With the option `intSeparator` the symbol can be changed to any non-number character.

## Part II

# pst-node

## 9 \nclineII

The dashed lines are black and white by default. The new macro `\nclineII` offers two-color lines and has the same syntax as `\ncline`:

`\ncline[<options>]{<Node A>}{<Node B>}`



```
1 \circclenode[linecolor=blue,linewidth=2pt]{A}{A}%  
2 \hspace{9cm}\circclenode[linecolor=cyan,linewidth=2pt]{B}{B}  
3 \nclineII[linewidth=5pt]{A}{B}
```

### 9.1 The options

These options are all defined in the package `pstricks-add`.

name	meaning
<code>dashColorI</code>	first color, default is <b>black</b>
<code>dashColorII</code>	second color, default is <b>red</b>
<code>dashNo</code>	The ratio of <code>dashColorI</code> to <code>dashColorII</code> , the default is 0.2

`dashNo` can have values greater than 1. In this case the value will be taken as an absolute width in the pt unit. Only this unit is possible!

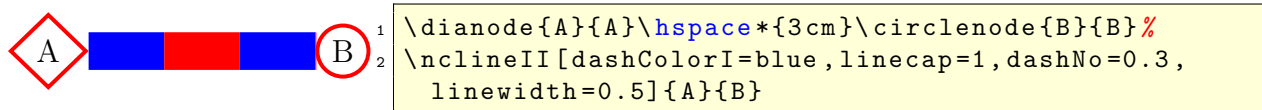
### 9.2 Examples



```
1 \circclenode{A}{A}\hspace*{3cm}\dianode{B}{B}%  
2 \nclineII[linewidth=8pt,dashColorI=blue]{A}{B}
```



```
1 \circclenode{A}{A}\hspace*{3cm}\circclenode{B}{B}%  
2 \nclineII[dashColorI=blue,linewidth=3pt,dashNo  
=15]{->}{A}{B}
```



## 10 \pclineII

This is nearly the same macro as `\psline` from the main `psstricks` package.

`\pcline[<options>](<Node A>)(<Node B>)`



```

1 \circlednode[linecolor=blue,linewidth=2pt]{A}{A}%
2 \hspace*{9cm}\circlednode[linecolor=cyan,linewidth=2pt]{B}{B}
3 \pclineII[linewidth=5pt](A)(B)

```

This macro makes only sense when connecting two "invisible" nodes, like this connection from here to the above word `psstricks`.

```

1 \raggedright This macro makes only sense when connecting two ''invisible
2 '' nodes,
3 like this connection from here\node{D}\pclineII{->}(D)(C){}
to the above word \verb|psstricks|.

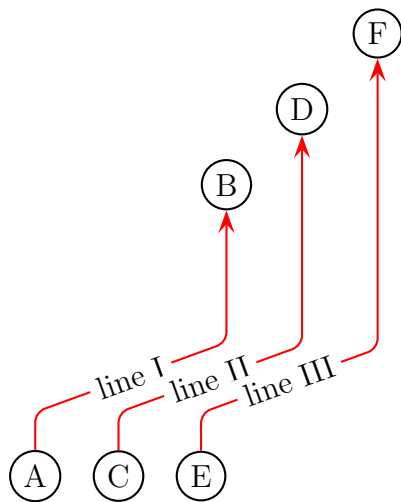
```

## 11 \ncdiag and \pcdiag

With the new option `lineAngle` the lines drawn by the `ncdiag` macro can now have a specified gradient. Without this option one has to define the two arms (which maybe zero) and PSTricks draws the connection between them. Now there is only a static `armA`, the second one `armB` is calculated when an angle `lineAngle` is defined. This angle is the gradient of the intermediate line between the two arms. The syntax of `ncdiag` is

`\ncdiag[<options>]{<Node A>}{<Node B>}`  
`\pcdiag[<options>](<Node A>)(<Node B>)`

name	meaning
<code>lineAngle</code>	angle of the intermediate line segment. Default is 0, which is the same than using <code>ncdiag</code> without the <code>lineAngle</code> option.

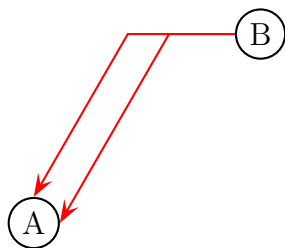


```

1 \begin{pspicture}(5,6)
2   \circnode{A}{A}\quad\circnode{C}{C}%
3   \quad\circnode{E}{E}
4   \rput(0,4){\circnode{B}{B}}
5   \rput(1,5){\circnode{D}{D}}
6   \rput(2,6){\circnode{F}{F}}
7   \psset{arrowscale=2,lineararc=0.2,%
8     linecolor=red,armA=0.5, angleA=90,angleB
9     =-90}
10  \ncdiag[lineAngle=20]{->}{A}{B}
11  \ncput*[nrot=:U]{line I}
12  \ncdiag[lineAngle=20]{->}{C}{D}
13  \ncput*[nrot=:U]{line II}
14  \ncdiag[lineAngle=20]{->}{E}{F}
15  \ncput*[nrot=:U]{line III}
16 \end{pspicture}

```

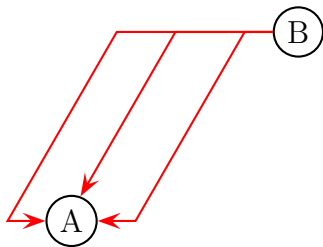
The `ncdiag` macro sets the `armB` dynamically to the calculated value. Any user setting of `armB` is overwritten by the macro. The `armA` could be set to a zero length:



```

1 \begin{pspicture}(4,3)
2   \rput(0.5,0.5){\circnode{A}{A}}
3   \rput(3.5,3){\circnode{B}{B}}
4   {\psset{linecolor=red,arrows=<-,arrowscale=2}
5     \ncdiag[lineAngle=60,%
6       armA=0,angleA=0,angleB=180]{A}{B}
7     \ncdiag[lineAngle=60,%
8       armA=0,angleA=90,angleB=180]{A}{B}}
9 \end{pspicture}

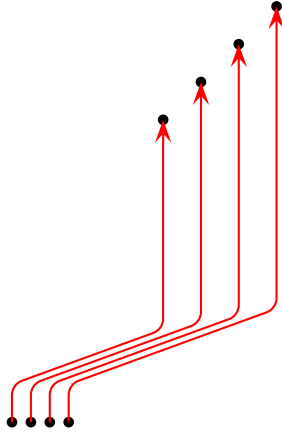
```



```

1 \begin{pspicture}(4,3)
2   \rput(1,0.5){\circnode{A}{A}}
3   \rput(4,3){\circnode{B}{B}}
4   {\psset{linecolor=red,arrows=<-,arrowscale=2}
5     \ncdiag[lineAngle=60,%
6       armA=0.5,angleA=0,angleB=180]{A}{B}
7     \ncdiag[lineAngle=60,%
8       armA=0,angleA=70,angleB=180]{A}{B}
9     \ncdiag[lineAngle=60,%
10      armA=0.5,angleA=180,angleB=180]{A}{B}}
11 \end{pspicture}

```



```

1 \begin{pspicture}(4,5.5)
2   \cnode*(0,0){2pt}{A}%
3   \cnode*(0.25,0){2pt}{C}%
4   \cnode*(0.5,0){2pt}{E}%
5   \cnode*(0.75,0){2pt}{G}%
6   \cnode*(2,4){2pt}{B}%
7   \cnode*(2.5,4.5){2pt}{D}%
8   \cnode*(3,5){2pt}{F}%
9   \cnode*(3.5,5.5){2pt}{H}%
10  {\psset{arrowscale=2,lineararc=0.2,%
11    linecolor=red,armA=0.5, angleA=90,angleB=-90}
12  \pcdiag[lineAngle=20]{->}{A}(B)
13  \pcdiag[lineAngle=20]{->}{C}(D)
14  \pcdiag[lineAngle=20]{->}{E}(F)
15  \pcdiag[lineAngle=20]{->}{G}(H)}
16 \end{pspicture}

```

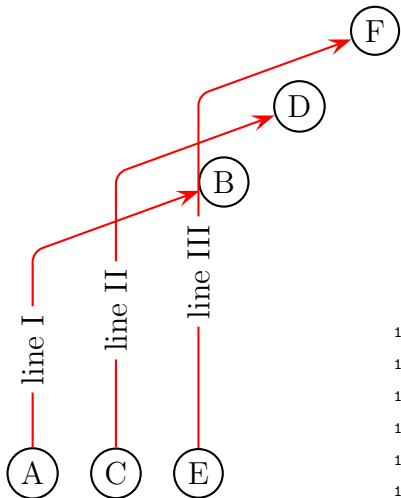
## 12 \ncdiag and \pcdiag

This is nearly the same than `\ncdiag` except that `armB=0` and the `angleB` value is computed by the macro, so that the line ends at the node with an angle like a `\pcdiag` line. The syntax of `ncdiag`/`pcdiag` is

```

\ncdiag[<options>]{<Node A>}{<Node B>}
\pcdiag[<options>](<Node A>)(<Node B>)

```

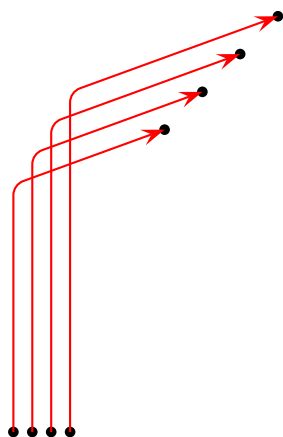


```

1 \begin{pspicture}(4,6)
2   \psset{linecolor=black}
3   \circlenode{A}{A}%
4   \quad\circlenode{C}{C}%
5   \quad\circlenode{E}{E}
6   \rput(0,4){\circlenode{B}{B}}
7   \rput(1,5){\circlenode{D}{D}}
8   \rput(2,6){\circlenode{F}{F}}
9   {\psset{arrowscale=2,lineararc=0.2,linecolor=red
10     ,armA=0.5, angleA=90}
11   \ncdiag[lineAngle=-160]{->}{A}{B}
12   \ncput*[nrot=:U]{line I}
13   \ncdiag[lineAngle=-160]{->}{C}{D}
14   \ncput*[nrot=:U]{line II}
15   \ncdiag[lineAngle=-160]{->}{E}{F}
16   \ncput*[nrot=:U]{line III}}
17 \end{pspicture}

```



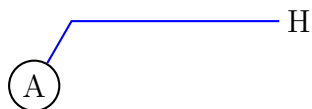


```

1 \begin{pspicture}(4,6)
2   \psset{linecolor=black}
3   \cnode*(0,0){2pt}{A}%
4   \cnode*(0.25,0){2pt}{C}%
5   \cnode*(0.5,0){2pt}{E}%
6   \cnode*(0.75,0){2pt}{G}%
7   \cnode*(2,4){2pt}{B}%
8   \cnode*(2.5,4.5){2pt}{D}%
9   \cnode*(3,5){2pt}{F}%
10  \cnode*(3.5,5.5){2pt}{H}%
11  {\psset{arrowscale=2,lineararc=0.2,linecolor=red
12    ,armA=0.5, angleA=90}
13  \pcdiagg[lineAngle=20]{->}(A)(B)
14  \pcdiagg[lineAngle=20]{->}(C)(D)
15  \pcdiagg[lineAngle=20]{->}(E)(F)
16  \pcdiagg[lineAngle=20]{->}(G)(H)}
17 \end{pspicture}

```

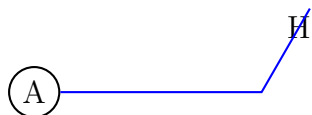
The only catch for `\ncdiagg` is, that you need the right value for `lineAngle`. If the node connection is on the wrong side of the second node, then choose the corresponding angle, e.g.: if  $20^\circ$  is wrong then take  $-160^\circ$ , the corresponding to  $180^\circ$ .



```

1 \begin{pspicture}(4,1.5)
2   \circlenode{a}{A}
3   \rput[l](3,1){\rnode{b}{H}}
4   \ncdiagg[lineAngle=60,angleA=180,armA=.5,nodesepA=3
5     pt,linecolor=blue]{b}{a}
6 \end{pspicture}

```



```

1 \begin{pspicture}(4,1.5)
2   \circlenode{a}{A}
3   \rput[l](3,1){\rnode{b}{H}}
4   \ncdiagg[lineAngle=60,armA=.5,nodesepB=3pt,
5     linecolor=blue]{a}{b}
6 \end{pspicture}

```



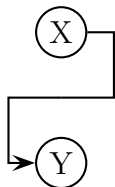
```

1 \begin{pspicture}(4,1.5)
2   \circlenode{a}{A}
3   \rput[l](3,1){\rnode{b}{H}}
4   \ncdiagg[lineAngle=-120,armA=.5,nodesepB=3pt,
5     linecolor=blue]{a}{b}
6 \end{pspicture}

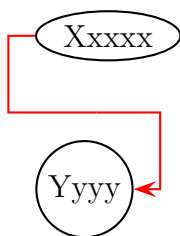
```

## 13 `\ncbarr`

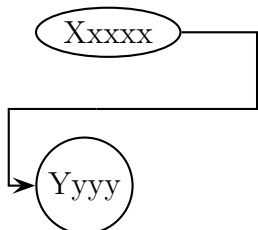
This has the same behaviour as `ncbar`, but has 5 segments and all are horizontal ones. This is the reason why `angleA` must be 0 or alternative 180. All other values are set to 0 by the macro. The intermediate horizontal line is symmetrical to the distance of the two nodes.



```
1 \psset{arrowscale=2}%
2 \circlenode{X}{X}\[1cm]
3 \circlenode{Y}{Y}
4 \ncbarr[angleA=0,arrows=->,arrowscale=2]{X}{Y}
```



```
1 \psset{arrowscale=2}%
2 \ovalnode{X}{Xxxxx}\[1cm]
3 \circlenode{Y}{Yyyy}
4 \ncbarr[angleA=180,arrows=->,arrowscale=2,linecolor=red]{X}{Y}
```

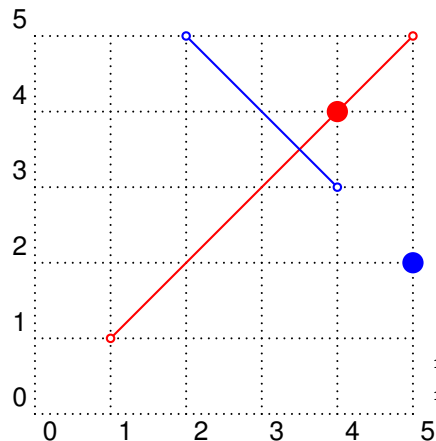


```
1 \psset{arrowscale=2}%
2 \ovalnode{X}{Xxxxx}\[1cm]
3 \circlenode{Y}{Yyyy}
4 \ncbarr[angleA=20,armA=1cm,arrows=->,arrowscale=2]{X}{Y}
```

## 14 `\psLNode` and `\psLCNode`

`\psLNode` interpolates the Line  $\overline{AB}$  by the given value and sets a node at this point. The syntax is

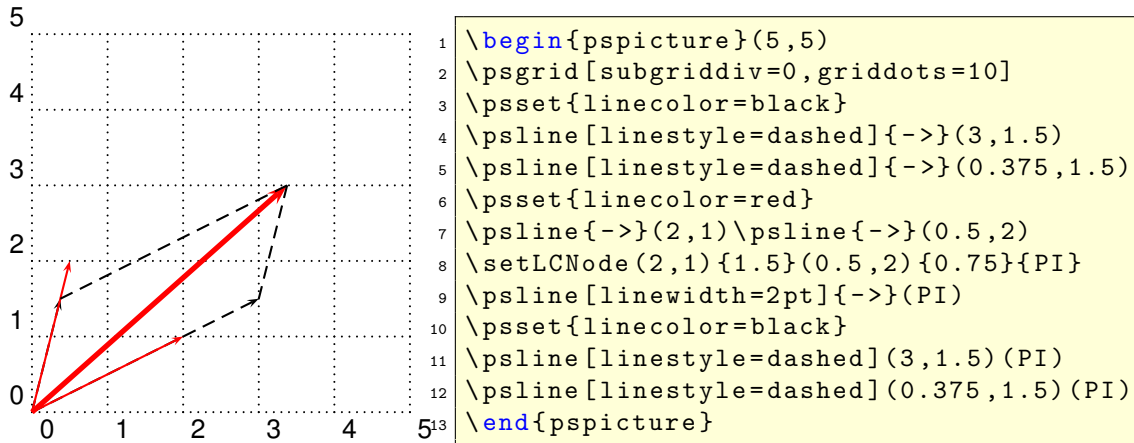
`\setLNode(P1)(P2){value}{Node name}`



```
1 \begin{pspicture}(5,5)
2 \psgrid[subgriddiv=0,griddots=10]
3 \psset{linecolor=red}
4 \psline{o-o}(1,1)(5,5)
5 \setLNode(1,1)(5,5){0.75}{PI}
6 \qdisk(PI){4pt}
7 \psset{linecolor=blue}
8 \psline{o-o}(4,3)(2,5)
9 \setLNode(4,3)(2,5){-0.5}{PII}
10 \qdisk(PII){4pt}
11 \end{pspicture}
```

The `\psLCNode` macro builds the linear combination of the two given vectors and stores the end of the new vector as a node. All vectors start at  $(0,0)$ , so a `\rput` maybe appropriate. The syntax is

`\setLCNode(P1){value 1}(P2){value 2}{Node name}`



## Part III

# pst-plot

## 15 New options

The option `tickstyle=full|top|bottom` is already present in the `pst-plot` package, but it is mentioned here for some completeness.

Table 2: All new parameters for `pst-plot`

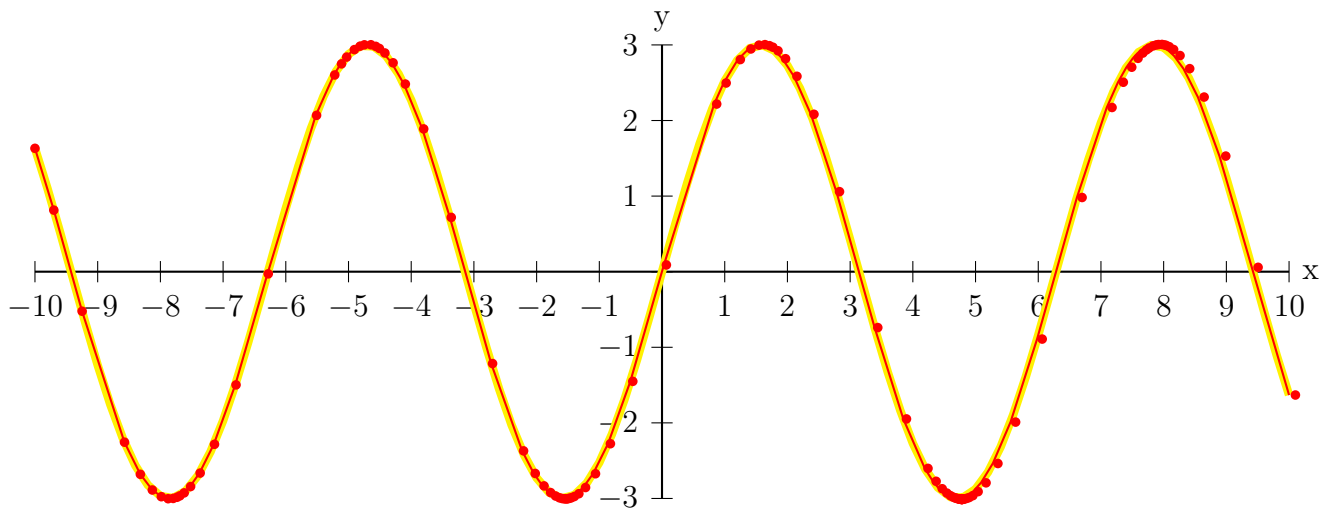
Name	Type	Default
<code>algebraic</code>	<code>false true</code>	<code>false</code>
<code>comma</code>	<code>false true</code>	<code>false</code>
<code>xAxis</code>	<code>false true</code>	<code>true</code>
<code>yAxis</code>	<code>false true</code>	<code>true</code>
<code>xyAxes</code>	<code>false true</code>	<code>true</code>
<code>xDecimals</code>	<code>&lt;number&gt; or empty</code>	<code>{}</code>
<code>yDecimals</code>	<code>&lt;number&gt; or empty</code>	<code>{}</code>
<code>xyDecimals</code>	<code>&lt;number&gt; or empty</code>	<code>{}</code>
<code>xLabel</code>	<code>&lt;anything&gt;</code>	<code>{}</code>
<code>yLabel</code>	<code>&lt;anything&gt;</code>	<code>{}</code>
<code>xyLabel</code>	<code>&lt;anything&gt;</code>	<code>{}</code>
<code>tickstyle</code>	<code>full top bottom</code>	<code>full</code>
<code>ticks</code>	<code>&lt;all x y none&gt;</code>	<code>all</code>
<code>labels</code>	<code>&lt;all x y none&gt;</code>	<code>all</code>
<code>subticks</code>	<code>&lt;number&gt;</code>	<code>0</code>
<code>xsubticks</code>	<code>&lt;number&gt;</code>	<code>0</code>
<code>ysubticks</code>	<code>&lt;number&gt;</code>	<code>0</code>
<code>ticksize</code>	<code>&lt;length [length]&gt;</code>	<code>-4pt 4pt</code>
<code>subticksize</code>	<code>&lt;number&gt;</code>	<code>0.75</code>
<code>tickwidth</code>	<code>&lt;length&gt;</code>	<code>0.5\pslinewidth</code>
<code>subtickwidth</code>	<code>&lt;length&gt;</code>	<code>0.25\pslinewidth</code>
<code>tickcolor</code>	<code>&lt;color&gt;</code>	<code>black</code>
<code>xtickcolor</code>	<code>&lt;color&gt;</code>	<code>black</code>
<code>ytickcolor</code>	<code>&lt;color&gt;</code>	<code>black</code>
<code>subtickcolor</code>	<code>&lt;color&gt;</code>	<code>darkgray</code>
<code>xsubtickcolor</code>	<code>&lt;color&gt;</code>	<code>darkgray</code>
<code>ysubtickcolor</code>	<code>&lt;color&gt;</code>	<code>darkgray</code>
<code>ticklinestyle</code>	<code>solid   dashed   dotted   none</code>	<code>solid</code>
<code>subticklinestyle</code>	<code>solid   dashed   dotted   none</code>	<code>solid</code>
<code>xlabelFactor</code>	<code>&lt;anything&gt;</code>	<code>{\ empty}</code>
<code>ylabelFactor</code>	<code>&lt;anything&gt;</code>	<code>{\ empty}</code>
<code>xlogBase</code>	<code>&lt;number&gt; or empty</code>	<code>{}</code>

Name	Type	Default
ylogBase	<number> or empty	{}
xylogBase	<number> or empty	{}
logLines	<none x y all>	none
nStep	<number>	1
nStart	<number>	0
nEnd	<number> or empty	{}
xStep	<number>	0
yStep	<number>	0
xStart	<number> or empty	{}
yStart	<number> or empty	{}
xEnd	<number> or empty	{}
yEnd	<number> or empty	{}
plotNo	<number>	1
plotNoMax	<number>	1
xAxisLabel	<anything>	{\ empty}
yAxisLabel	<anything>	{\ empty}
xAxisLabelPos	<(x,y)> or empty	{\ empty}
yAxisLabelPos	<(x,y)> or empty	{\ empty}
llx	<length>	0pt
lly	<length>	0pt
urx	<length>	0pt
ury	<length>	0pt
polarplot	false true	false

## 15.1 algebraic

By default the function of `\psplot` has to be described in Reversed Polish Notation. The option `algebraic` allows to do this in the common algebraic notation. E.g.:

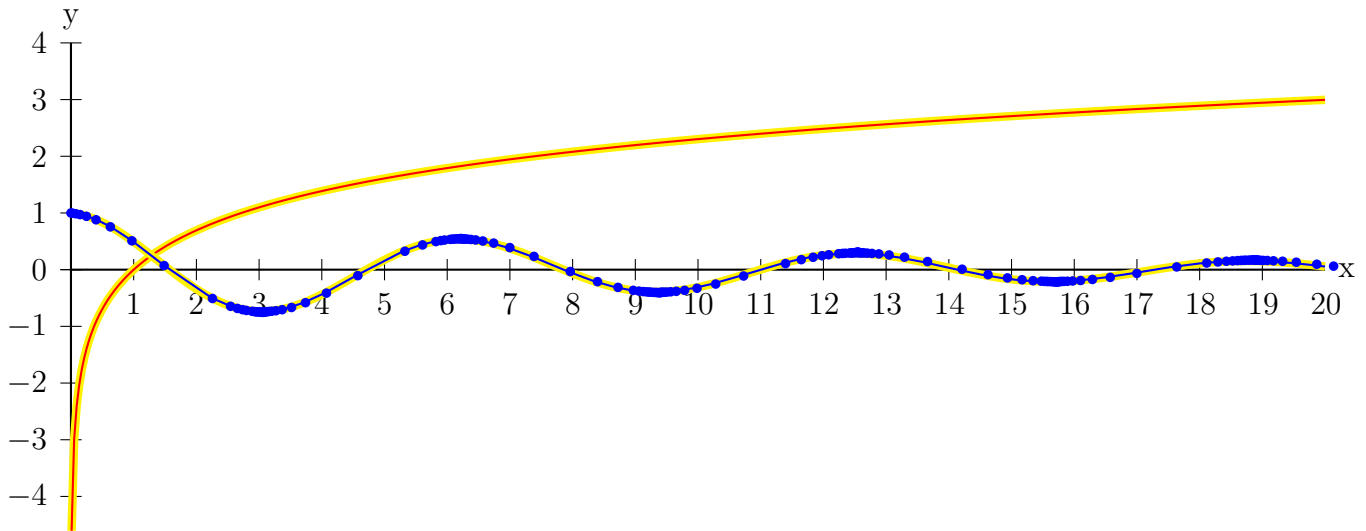
RPN	algebraic
x ln	ln(x)
x cos 2.71 x neg 10 div exp mul	cos(x)*2.71 <sup>(-x/10)</sup>
x cos 4 mul	4*cos(1/x)



```

1 \psgraph(-10,-3)(10,3){\linewidth}{6cm}
2 \psset{algebraic=true, plotpoints=101}
3 \psplot[linecolor=yellow, linewidth=4\pslinewidth]{-10}{10}{3*sin(x)}%
4 \psplot[linecolor=red, showpoints=true, VarStep=0.8]{-10}{10}{3*sin(x)}
5 \endpsgraph

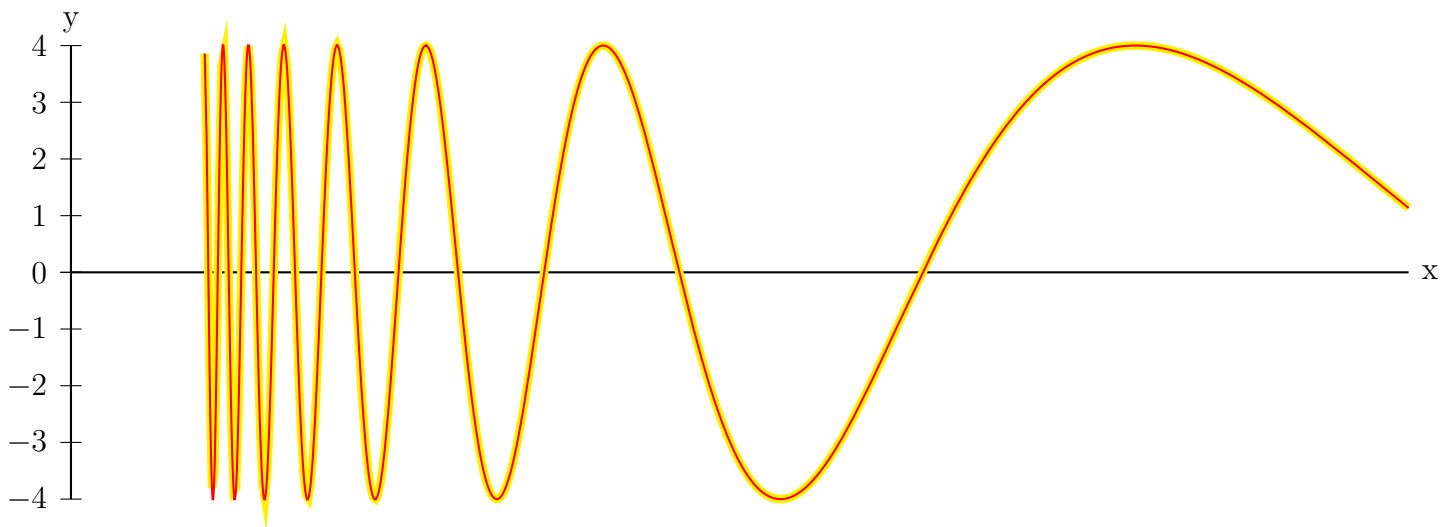
```



```

1 \psgraph(0,-4)(20,4){\linewidth}{6cm}
2 \psset{algebraic=true, plotpoints=501}
3 \psplot[linecolor=yellow, linewidth=4\pslinewidth]{0.01}{20}{ln(x)}%
4 \psplot[linecolor=red]{0.01}{20}{ln(x)}
5 \psplot[linecolor=yellow, linewidth=4\pslinewidth]{0}{20}{cos(x)*2.71^(-
  x/10)}
6 \psplot[linecolor=blue, showpoints=true, VarStep=0.8]{0}{20}{cos(x)
  *2.71^(-x/10)}
7 \endpsgraph

```



```

1 \begin{psgraph}(0,-4)(0.2,4){\linewidth}{6cm}
2   \psset{algebraic=true, plotpoints=501}
3   \psplot[linecolor=yellow, linewidth=4\pslinewidth]{0.02}{.2}{4*cos(1/x)}
4   %
5   \psplot[linecolor=red, VarStep=0.95]{.02}{.2}{4*cos(1/x)}%
6 \end{psgraph}

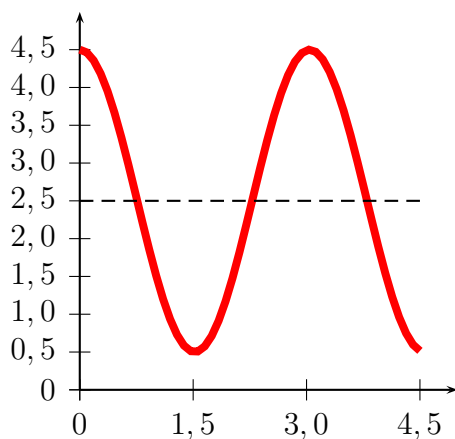
```

## 15.2 comma

Syntax:

comma=false|true

Setting this option to true gives labels with a comma as a decimal separator instead of the dot. comma and comma=true is the same.



```

1 \begin{pspicture}(-0.5,-0.5)(5,5.5)
2   \psaxes[Dx=1.5,Dy=0.5,comma]{->}(5,5)
3   \psplot[linecolor=red,linewidth=3pt]{0}{4.5}%
4     {x 180 mul 1.52 div cos 2 mul 2.5 add}
5   \psline[linestyle=dashed](0,2.5)(4.5,2.5)
6 \end{pspicture}

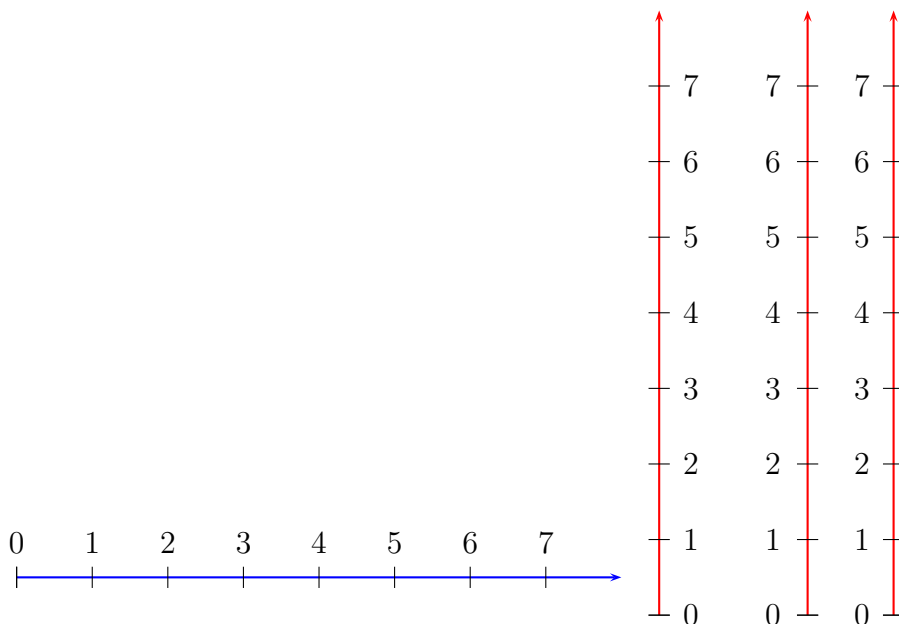
```

## 15.3 xyAxes, xAxis and yAxis

Syntax:

```
xyAxes=true|false
xAxes=true|false
yAxes=true|false
```

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



```
1 \begin{pspicture}(8,1)
2 \psaxes[yAxis=false,linecolor=blue]{->}(0,0.5)(8,0.5)
3 \end{pspicture}%
4 \begin{pspicture}(1,8)
5 \psaxes[xAxis=false,linecolor=red]{->}(0.5,0)(0.5,8)
6 \end{pspicture}\hspace{2em}
7 \begin{pspicture}(1,8)
8 \psaxes[xAxis=false,linecolor=red,labelsep=-20pt]{->}(0.5,0)(0.5,8)
9 \end{pspicture}
10 \begin{pspicture}(1,8)
11 \psaxes[xAxis=false,linecolor=red]{->}(0.5,0)(0.501,8)
12 \end{pspicture}%
```

As seen in the example, a single y axis gets the labels on the right side. This can be changed in two ways, first with the option `labelsep` and second with a very short and therefore invisible x-axis (right example).



## 15.4 xyDecimals, xDecimals and yDecimals

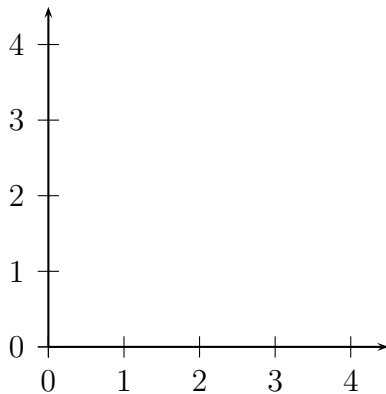
Syntax:

xyDecimals=<number>

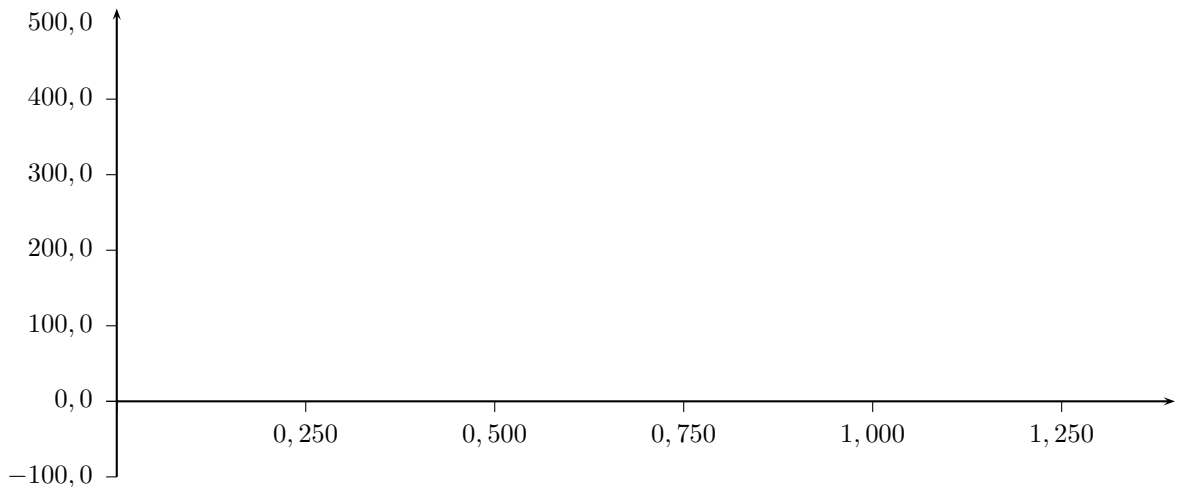
xDecimals=<any>

yDecimals=<any>

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



```
1 \begin{pspicture}(-1.5,-0.5)(5,4.75)
2   \psaxes[xyDecimals=2]{->}(0,0)(4.5,4.5)
3 \end{pspicture}
```



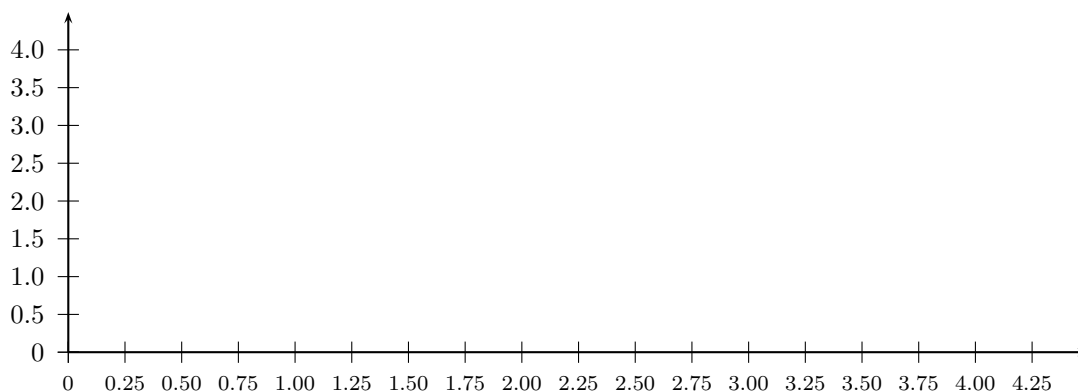
```
1 \psset{xunit=10cm, yunit=0.01cm}
2 \begin{pspicture}(-0.3,-150)(1.5,550.0)
3   \psaxes[Dx=0.25,Dy=100,tickstyle=bottom,xyLabel=\footnotesize,comma=true
4     ,%
5     xDecimals=3,yDecimals=1]{->}(0,0)(0,-100)(1.4,520)
6 \end{pspicture}
```

## 15.5 xyLabel, xLabel and yLabel

Syntax:

```
xyLabel=<any>  
xLabel=<any>  
yLabel=<any>
```

There are no special keywords to change the labelstyle for the `\psaxes` macro. With `xyLabel` it is possible to set both axes with the same command sequence. Unlike to the default `pst-plot` package the coordinates are not printed in mathmode. This makes it easier to choose other text styles.



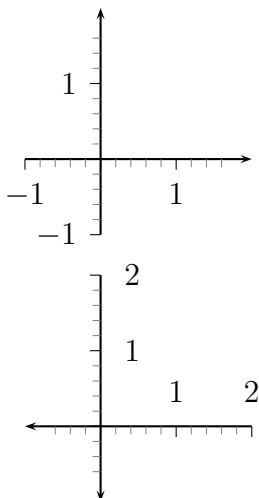
```
1 \psset{yunit=1cm,xunit=3cm}  
2 \begin{pspicture}(-0.3,-0.5)(5,4.75)  
3 \psaxes[xLabel={\scriptsize\itshape},yLabel={\sffamily\footnotesize},%  
4   Dy=0.5, Dx=0.25]{->}(0,0)(4.5,4.5)  
5 \end{pspicture}
```

## 15.6 tickstyle

Syntax:

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

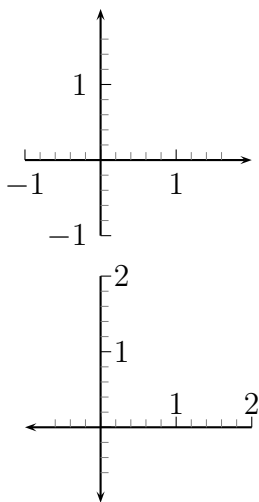
This option is already in the `pst-plot` package and only mentioned here for some completeness.



```

1 \begin{pspicture}(-1,-1)(2,2)
2 \psaxes[tickstyle=bottom,subticks=5]{->}(0,0)(-1,-1)
   (2,2)
3 \end{pspicture}\!\![0.5cm]
4 \begin{pspicture}(-1,-1)(2,2)
5 \psaxes[tickstyle=bottom,subticks=5]{->}(0,0)(2,2)
   (-1,-1)
6 \end{pspicture}

```



```

1 \begin{pspicture}(-1,-1)(2,2)
2 \psaxes[tickstyle=top,subticks=5]{->}(0,0)(-1,-1)(2,2)
3 \end{pspicture}\!\![0.5cm]
4 \begin{pspicture}(-1,-1)(2,2)
5 \psaxes[tickstyle=top,subticks=5]{->}(0,0)(2,2)(-1,-1)
6 \end{pspicture}

```

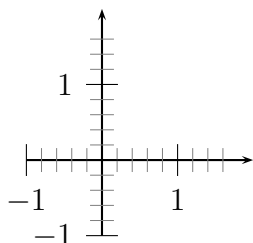
The `tickstyle` option changes the position of the labels by default. If you want the labels on the other side of an axis, then use the options `labelsep` or set the ticks with `ticksize`.

## 15.7 ticks

Syntax:

`ticks=all|x|y|none`

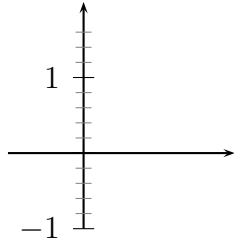
This option is also already in the `pst-plot` package and only mentioned here for some completeness.



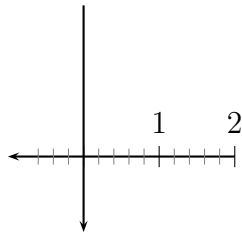
```

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

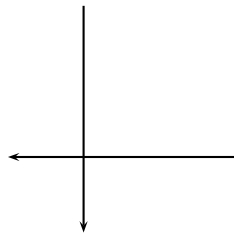
```



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



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



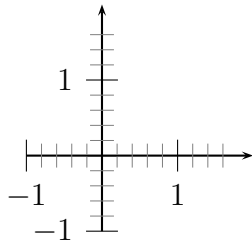
```
1 \begin{pspicture}(-1,-1)(2,2)
2 \psaxes[ticks=none,subticks=5]{->}(0,0)(2,2)(-1,-1)
3 \end{pspicture}
```

## 15.8 labels

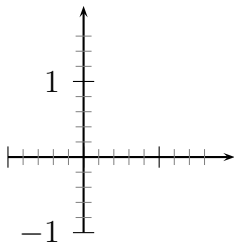
Syntax:

`labels=all|x|y|none`

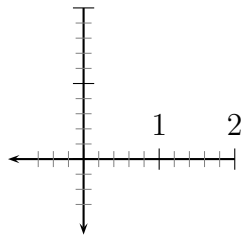
This option is also already in the `pst-plot` package and only mentioned here for some completeness.



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



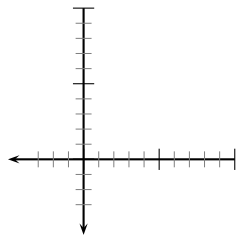
```
1 \begin{pspicture}(-1,-1)(2,2)
2 \psaxes[labels=y,subticks=5]{->}(0,0)(-1,-1)(2,2)
3 \end{pspicture}
```



```

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

```



```

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

```

## 15.9 ticksize, xticksize, yticksize

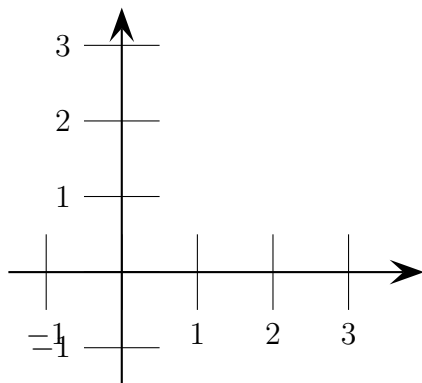
Syntax:

```

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

```

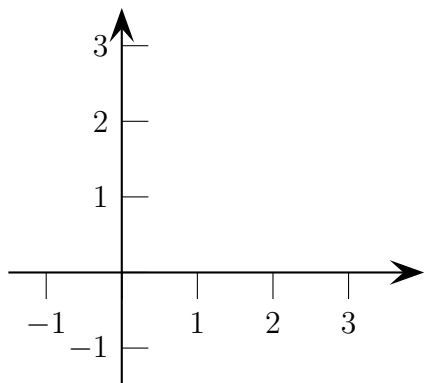
ticksize sets both values.



```

1 \psset{arrowscale=3}
2 \begin{pspicture}(-1.5,-1.5)(4,3.5)
3   \psaxes[ticksize=0.5cm]{->}(0,0)
4     (-1.5,-1.5)(4,3.5)
5 \end{pspicture}

```

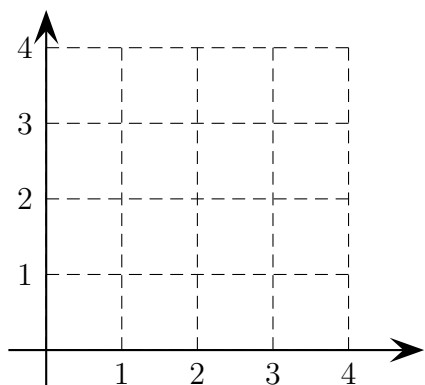


```

1 \psset{arrowscale=3}
2 \begin{pspicture}(-1.5,-1.5)(4,3.5)
3   \psaxes[xticks=-10pt 0,yticks=0 10pt
4     ]{->}(0,0)(-1.5,-1.5)(4,3.5)
5 \end{pspicture}

```

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



```

1 \psset{arrowscale=3}
2 \begin{pspicture}(-.5,-.5)(5,4.5)
3   \psaxes[ticklinestyle=dashed,ticks=0 4
4     cm]{->}(0,0)(-.5,-.5)(5,4.5)
5 \end{pspicture}

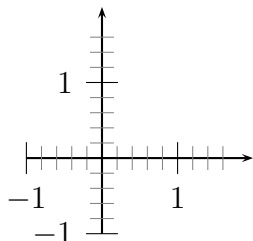
```

## 15.10 subticks

Syntax:

`subticks=<number>`

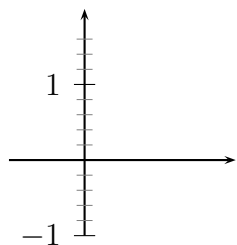
By default subticks cannot have labels.



```

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

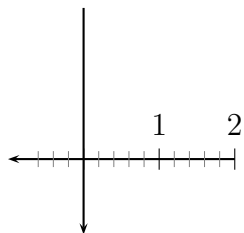
```



```

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

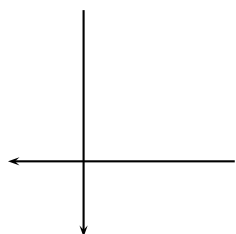
```



```

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

```



```

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

```

## 15.11 subticksize, xsubticksize, ysubticksize

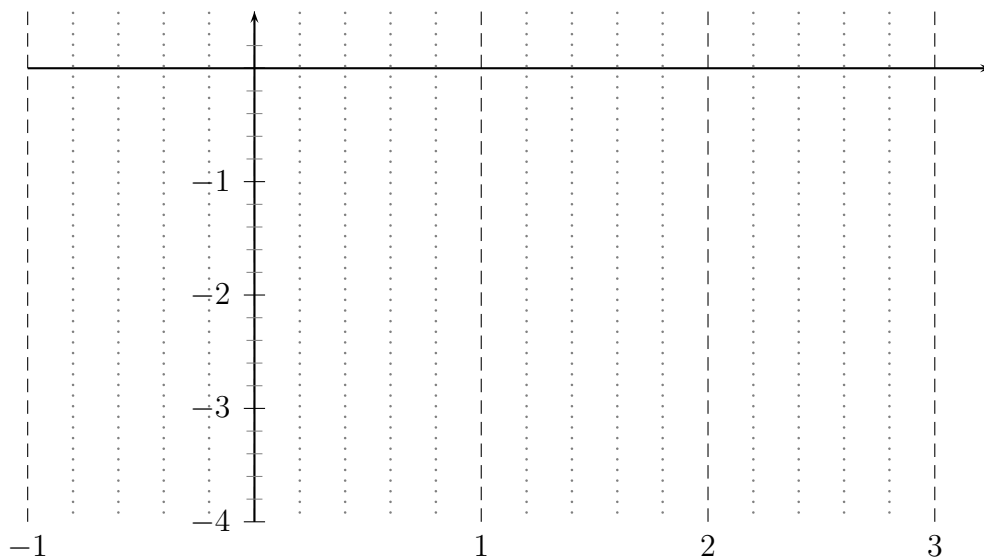
Syntax:

```

subticksize=value
xsubticksize=value
ysubticksize=value

```

`subticksize` sets both values, which are relative to the `ticksize` length and can have any number. 1 sets it to the same length as the main ticks.



```

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

```

## 15.12 tickcolor, subtickcolor

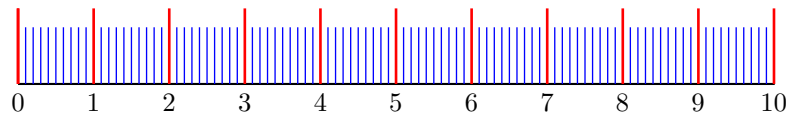
Syntax:

```

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

```

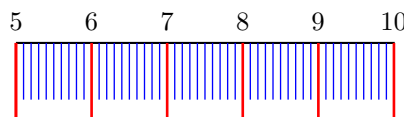
tickcolor and subtickcolor set both for the x- and the y-Axis.



```

1 \begin{pspicture}(0,-0.75)(10,1)
2 \psaxes[xLabel=\footnotesize,labelsep=2pt,yAxis=false,%
3   labelsep=-10pt,ticksize=0 10mm,subticks=10,subticksize=0.75,%
4   tickcolor=red,subtickcolor=blue,tickwidth=1pt,%
5   subtickwidth=0.5pt](10.01,0)
6 \end{pspicture}

```



```

1 \begin{pspicture}(5,-0.75)(10,1)
2 \psaxes[xLabel=\footnotesize,labelsep=2pt,yAxis=
3   false,%
4   labelsep=5pt,ticksize=0 -10mm,subticks=10,
5   subticksize=0.75,%
6   tickcolor=red,subtickcolor=blue,tickwidth=1pt,%
7   subtickwidth=0.5pt,0x=5](5,0)(5,0)(10.01,0)
\end{pspicture}

```

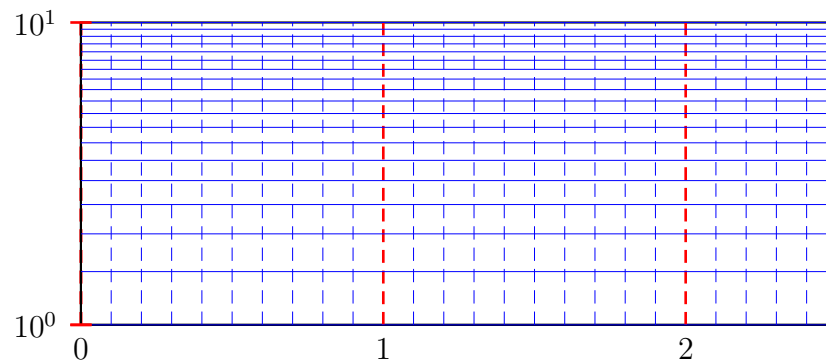


## 15.13 ticklinestyle and subticklinestyle

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 to `[sub]ticklines=0`

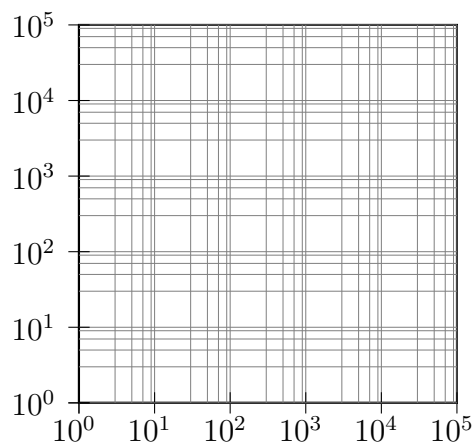


```
1 \psset{unit=4cm}
2 \pspicture(-0.15,-0.15)(2.5,1)
3   \psaxes[axesstyle=frame,logLines=y,xticks=0 1,xsubticks=1,%
4     ylogBase=10,tickcolor=red,subtickcolor=blue,tickwidth=1pt,%
5     subticks=20,xsubticks=10,xticklinestyle=dashed,%
6     xsubticklinestyle=dashed](2.5,1)
7 \endpspicture
```

## 15.14 loglines

Syntax:

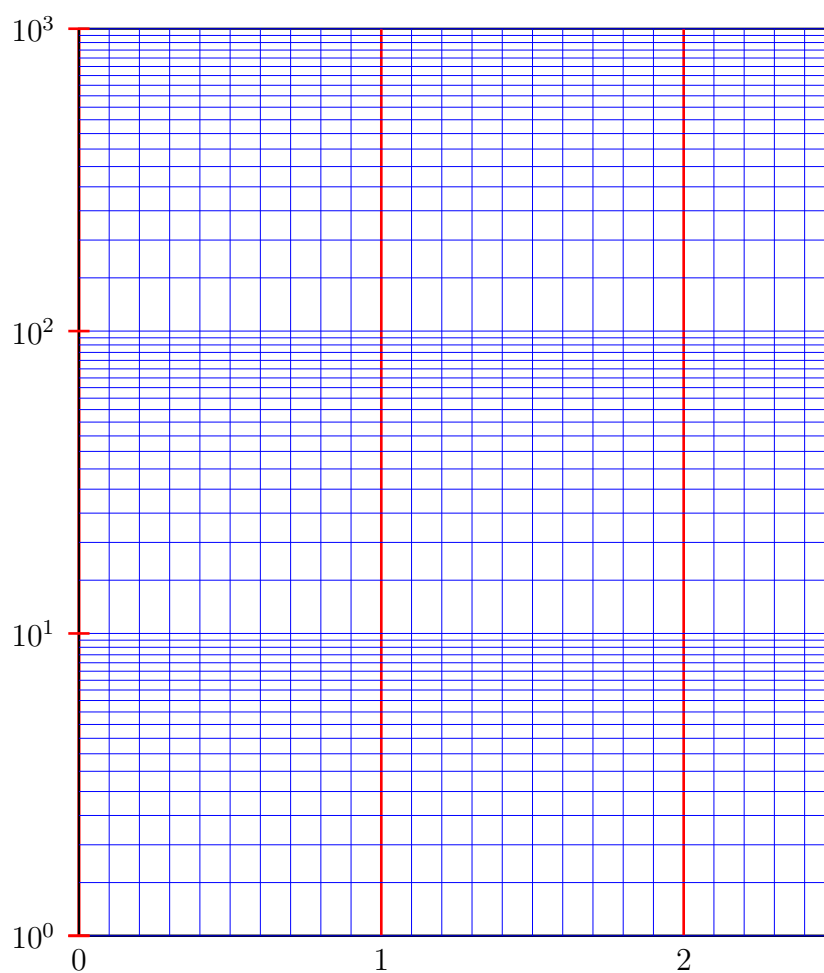
```
loglines=all|x|y
```



```

1 \pspicture(0,-1)(5,5)
2   \psaxes[subticks=5,axesstyle=frame,
3     xylogBase=10,logLines=all](5,5)
4 \endpspicture

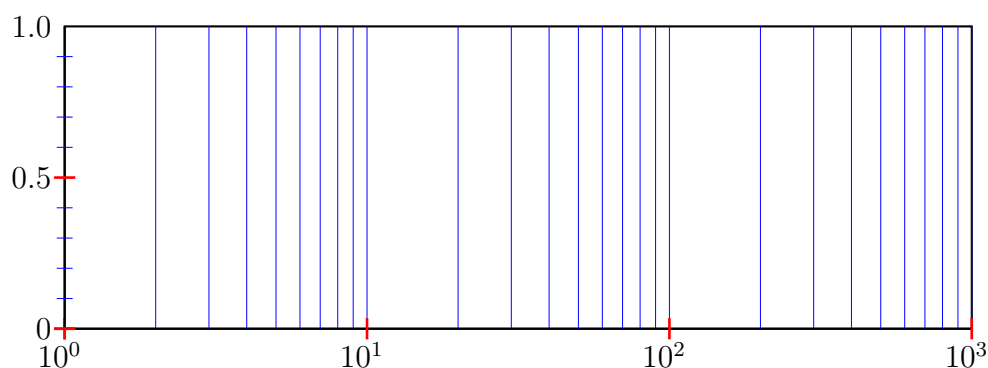
```



```

1 \psset{unit=4cm}
2 \pspicture(-0.15,-0.15)(2.5,3)
3   \psaxes[axesstyle=frame,logLines=y,xticksize=0.3,xsubticksize=1,%
4     ylogBase=10,tickcolor=red,subtickcolor=blue,tickwidth=1pt,%
5     subticks=20,xsubticks=10](2.5,3)
6 \endpspicture

```



```

1 \psset{unit=4}
2 \pspicture(0,-0.3)(3,1.2)
3   \psaxes[axesstyle=frame,logLines=x,xlogBase=10,Dy=0.5,%
4     tickcolor=red,subtickcolor=blue,tickwidth=1pt,ysubticks=5,xsubticks
5     =10](3,1)
6 \endpspicture

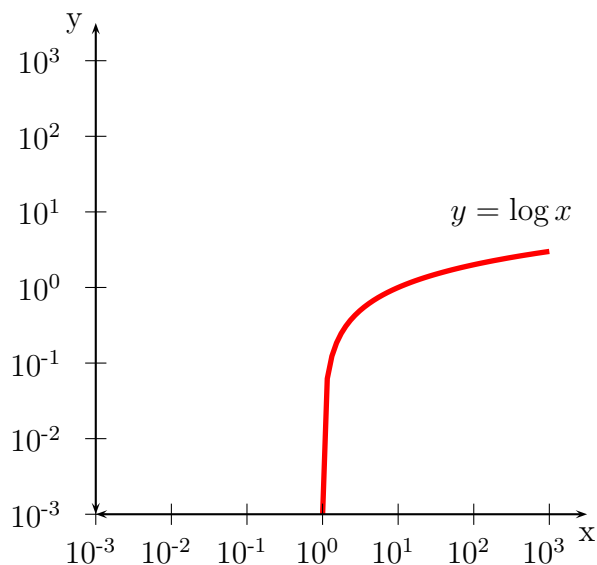
```

## 15.15 xylogBase, xlogBase and ylogBase

There are additional options `xylogBase` `xlogBase` | `ylogBase` to get one or both axes with logarithm 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 logarithm 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 `pstricks-add` does it by default. An alternative is to set these parameters to empty values `Ox={}`, `Oy={}`, in this case `pstricks-add` does nothing.

### 15.15.1 xylogBase

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



```

1 \begin{pspicture}(-3.5,-3.5)(3.5,3.5)
2   \psplot[linewidth=2pt,linecolor=red
3     ]{0.001}{3}{x log}
4   \psaxes[xylogBase=10,0y
5     =-3]{<->}(-3,-3)(3.5,3.5)
6   \uput[-90](3.5,-3){x}
7   \uput[180](-3,3.5){y}
8   \rput(2.5,1){$y=\log x$}
9 \end{pspicture}

```

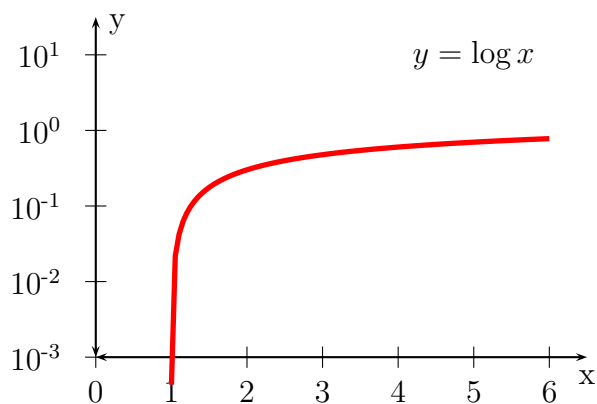
### 15.15.2 ylogBase

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

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

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

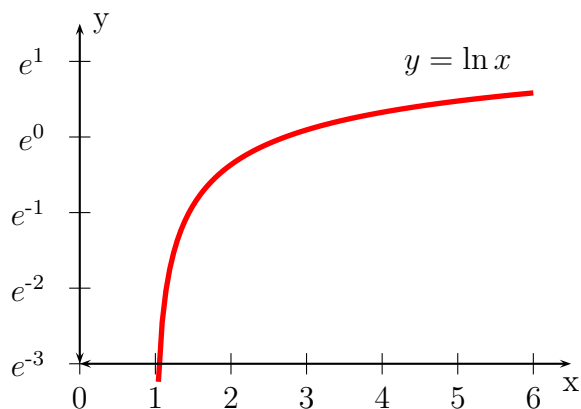
with an drawing intervall of  $1.001 \dots 6$ .



```

1 \begin{pspicture}(-0.5,-3.5)(6.5,1.5)
2   \psaxes[ylogBase=10]{<->}(0,-3)
3     (6.5,1.5)
4   \uput[-90](6.5,-3){x}
5   \uput[0](0,1.4){y}
6   \rput(5,1){$y=\log x$}
7   \psplot[linewidth=2pt,%
8     plotpoints=100,linecolor=red
9     ]{1.001}{6}{x log log} % log(x)
10 \end{pspicture}

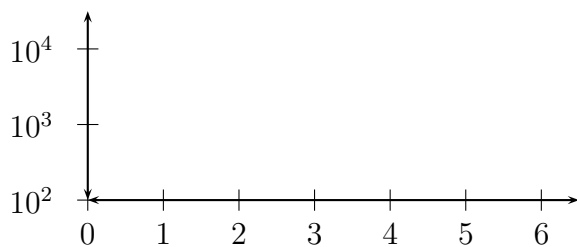
```



```

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

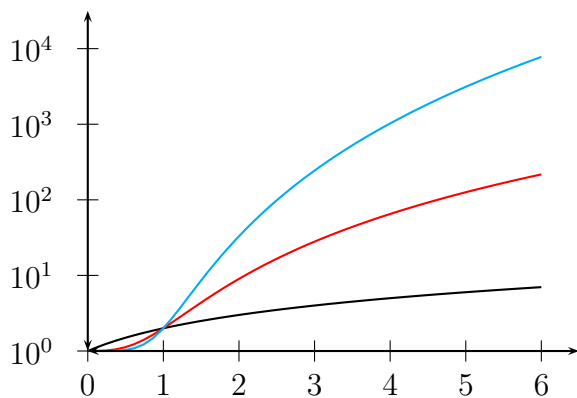
```



```

1 \begin{pspicture}(-0.5,1.75)(6.5,4.5)
2   \psaxes[ylogBase=10,0y=2]{<->}(0,2)
3     (0,2)(6.5,4.5)
4 \end{pspicture}

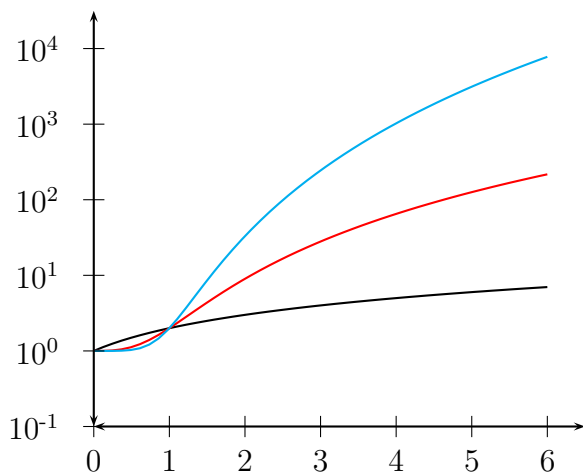
```



```

1 \begin{pspicture}(-0.5,-0.25)(6.5,4.5)
2   \psplot{0}{6}{x x cos add log}
3     % x^3 + cos(x)
4   \psplot[linewidth=red]{0}{6}{x 3 exp
5     x cos add log} % x^3 + cos(x)
6   \psplot[linewidth=cyan]{0}{6}{x 5
7     exp x cos add log} % x^5 + cos(x)
8   \psaxes[ylogBase=10]{<->}(6.5,4.5)
9 \end{pspicture}

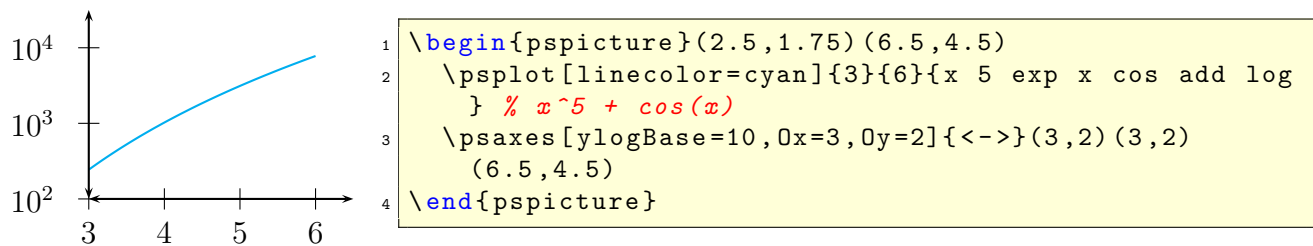
```



```

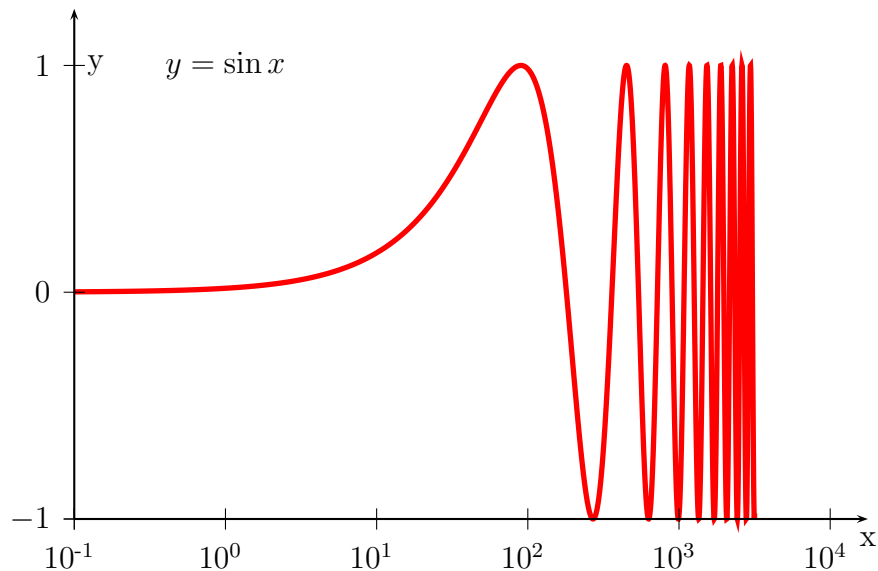
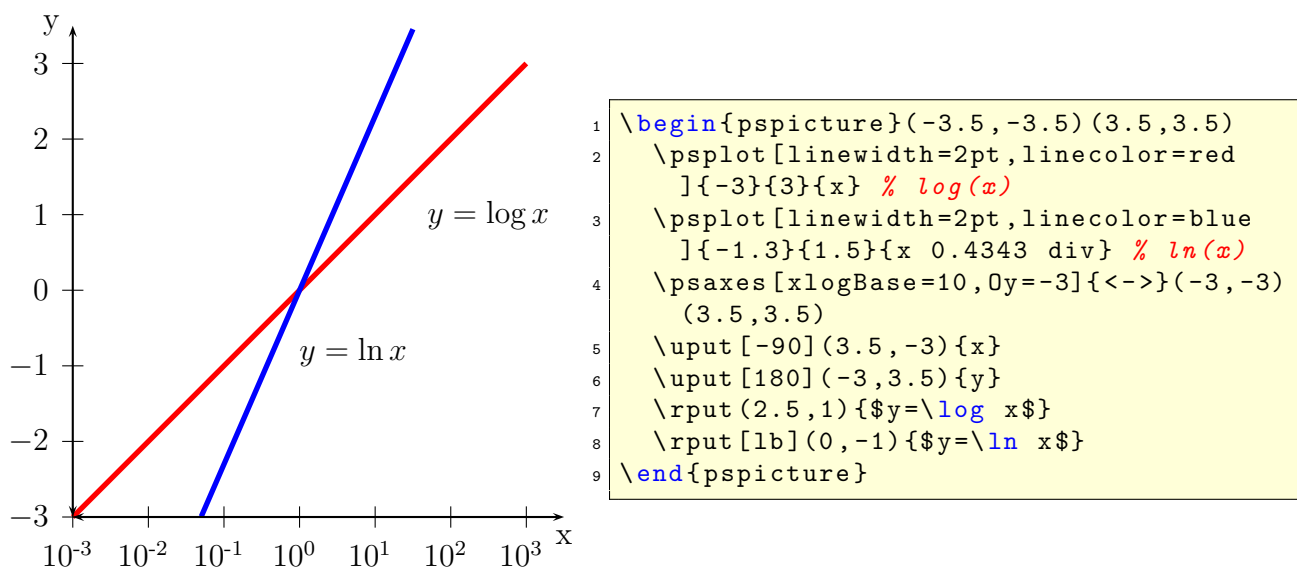
1 \begin{pspicture}(-0.5,-1.25)(6.5,4.5)
2   \psplot{0}{6}{x x cos add log}
3     % x^3 + cos(x)
4   \psplot[linewidth=red]{0}{6}{x 3 exp x
5     cos add log} % x^3 + cos(x)
6   \psplot[linewidth=cyan]{0}{6}{x 5 exp
7     x cos add log} % x^5 + cos(x)
8   \psaxes[ylogBase=10]{<->}(0,-1)(0,-1)
9     (6.5,4.5)
10 \end{pspicture}

```



### 15.15.3 xlogBase

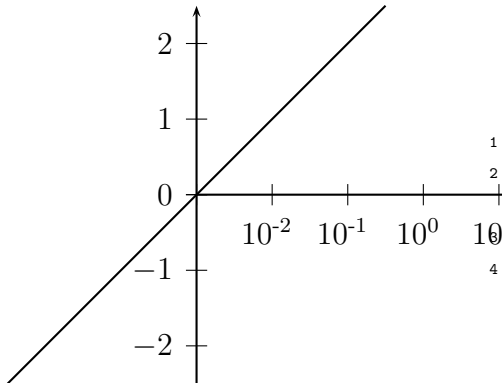
Now we have to use the easy math function  $y = x$  because the x axis is still  $\log x$ .



```

1 \psset{yunit=3cm,xunit=2cm}
2 \begin{pspicture}(-1.25,-1.25)(4.25,1.5)
3   \uput[-90](4.25,-1){x}
4   \uput[0](-1,1){y}
5   \rput(0,1){$y=\sin x$}
6   \psplot[linewidth=2pt,plotpoints=5000,linecolor=red]{-1}{3.5}{10 x exp
   sin }
7   \psaxes[xlogBase=10,0y=-1]{->}(-1,-1)(4.25,1.25)
8 \end{pspicture}

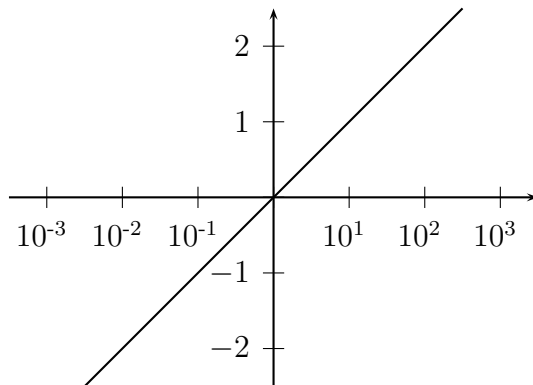
```



```

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

```



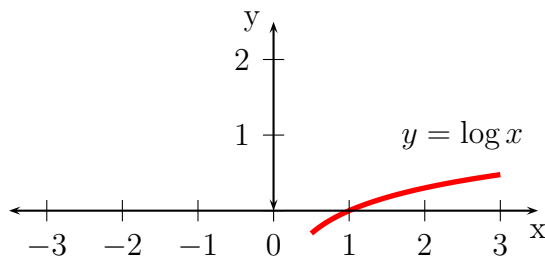
```

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

```

#### 15.15.4 No logstyle (xylogBase={})

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

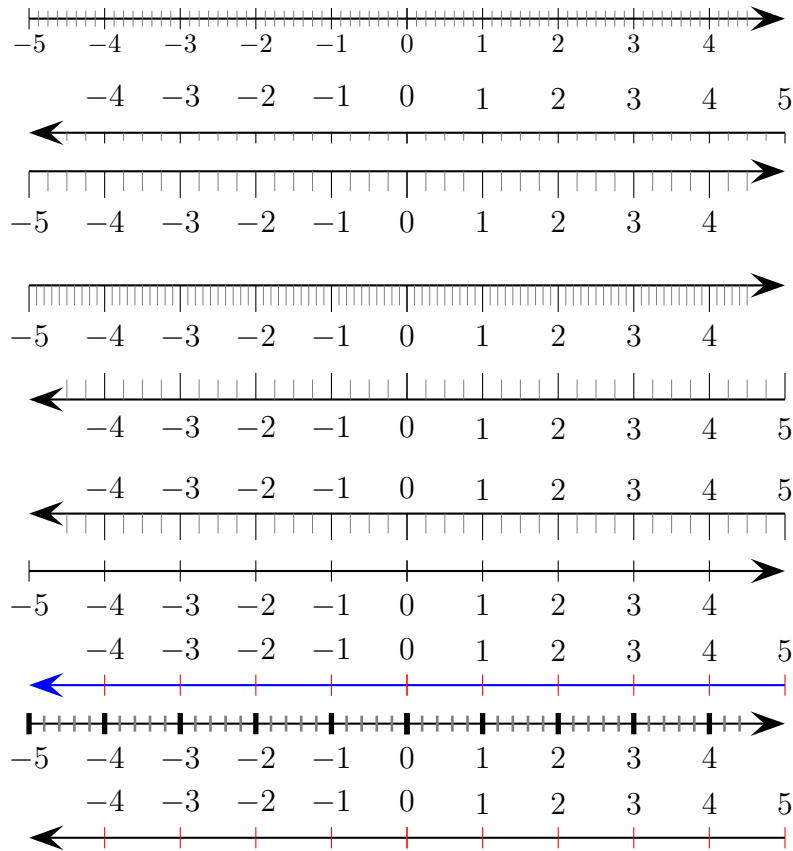


```

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

```

## 15.16 subticks, tickwidth and subtickwidth

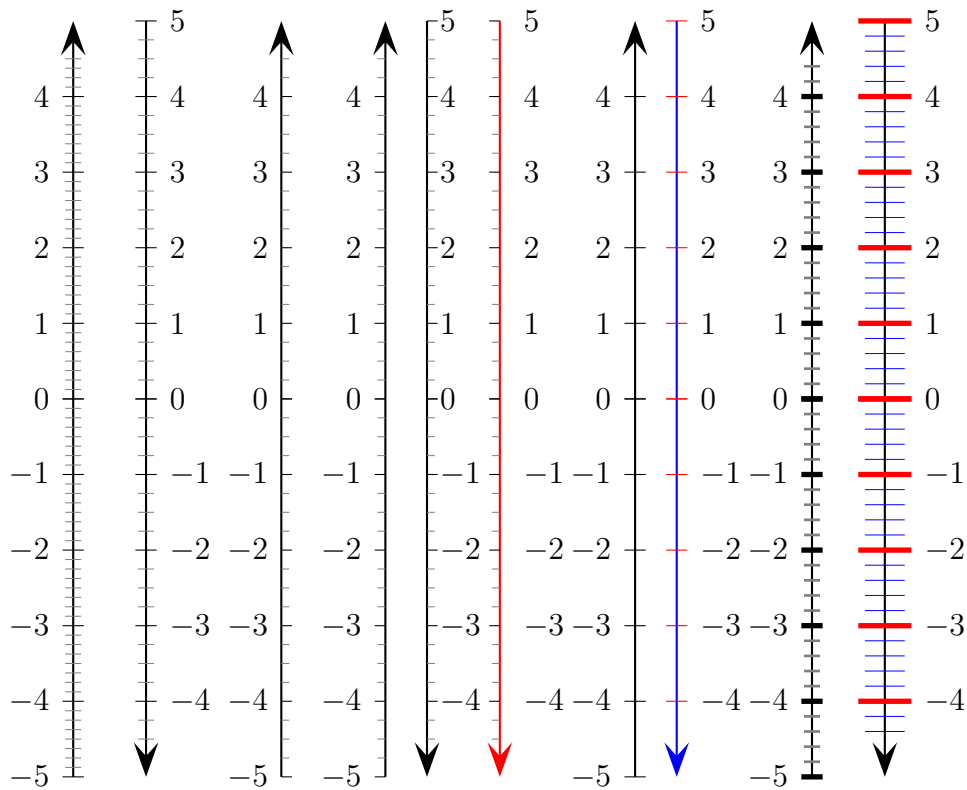


```

1 \psset{arrowscale=3}
2 \psaxes[xLabel=\footnotesize,labelsep=2pt,yAxis=false,subticks
3   =8]{->}(0,0)(-5,-1)(5,1)\[1cm]
4 \psaxes[yAxis=false,subticks=4,tickstyle=bottom]{->}(0,0)(5,1)(-5,-1)
5   \[1cm]
6 \psaxes[yAxis=false,subticks=4,ticks=-10pt 0]{->}(0,0)(-5,-5)(5,5)
7   \[1cm]
8 \psaxes[yAxis=false,subticks=10,ticks=0 -10pt,labelsep=15pt
9   ]{->}(0,0)(-5,-5)(5,5)\[1cm]
10 \psaxes[yAxis=false,subticks=4,ticks=0 10pt,labelsep=-15pt
11   ]{->}(0,0)(5,5)(-5,-5)\[1cm]
12 \psaxes[yAxis=false,subticks=4,ticks=0 -10pt]{->}(0,0)(5,5)(-5,-5)
13   \[0.25cm]
14 \psaxes[yAxis=false,subticks=0]{->}(0,0)(-5,-5)(5,5)\[1cm]
15 \psaxes[yAxis=false,subticks=0,tickcolor=red,linecolor=blue]{->}(0,0)
16   (5,5)(-5,-5)\[1cm]
17 \psaxes[yAxis=false,subticks=5,tickwidth=2pt,subtickwidth=1pt
18   ]{->}(0,0)(-5,-5)(5,5)\[1cm]
19 \psaxes[yAxis=false,subticks=0,tickcolor=red]{->}(0,0)(5,5)(-5,-5)

```

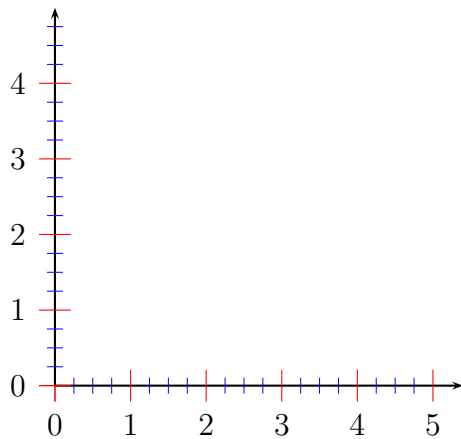




```

1 \psset{arrowscale=3}
2 \psaxes[xAxis=false,subticks=8]{->}(0,0)(-5,-5)(5,5)\hspace{2em}
3 \psaxes[xAxis=false,subticks=4]{->}(0,0)(5,5)(-5,-5)\hspace{4em}
4 \psaxes[xAxis=false,subticks=4,tickstyle=top]{->}(0,0)(-5,-5)(5,5)\hspace{3em}
5 \psaxes[xAxis=false,subticks=4,tickstyle=bottom]{->}(0,0)(-5,-5)(5,5)\hspace{1em}
6 \psaxes[xAxis=false,subticks=4,tickstyle=top]{->}(0,0)(5,5)(-5,-5)\hspace{2em}
7 \psaxes[xAxis=false,subticks=4,tickstyle=bottom,linecolor=red]{->}(0,0)(5,5)(-5,-5)\hspace{4em}
8 \psaxes[xAxis=false,subticks=0]{->}(0,0)(-5,-5)(5,5)\hspace{1em}
9 \psaxes[xAxis=false,subticks=0,tickcolor=red,linecolor=blue]{->}(0,0)(5,5)(-5,-5)\hspace{4em}
10 \psaxes[xAxis=false,subticks=5,tickwidth=2pt,subtickwidth=1pt]{->}(0,0)(-5,-5)(5,5)\hspace{2em}
11 \psaxes[xAxis=false,subticks=5,tickcolor=red,tickwidth=2pt,%
12 ticksize=10pt,subtickcolor=blue,subticksize=0.75]{->}(0,0)(5,5)(-5,-5)

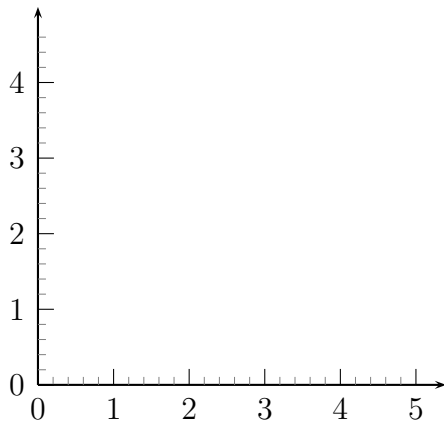
```



```

1 \pspicture(5,5.5)
2 \psaxes[subticks=4,ticks=6pt,subticks=
  =0.5,%
3   tickcolor=red,subtickcolor=blue]{->}(5.4,5)
4 \endpspicture

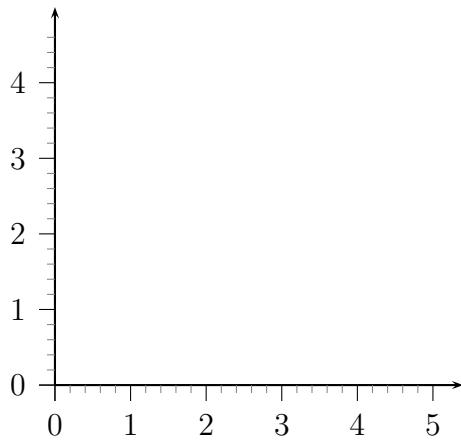
```



```

1 \pspicture(5,5.5)
2   \psaxes[subticks=5,ticks=6pt,subticks=
  =0.5,tickstyle=top]{->}(5.4,5)
3 \endpspicture

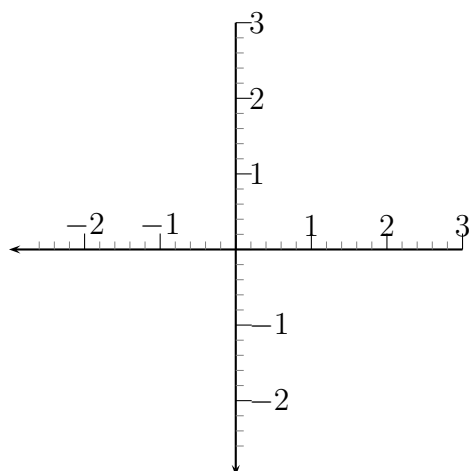
```



```

1 \pspicture(5,5.5)
2   \psaxes[subticks=5,ticks=6pt,subticks=
  =0.5,tickstyle=bottom]{->}(5.4,5)
3 \endpspicture

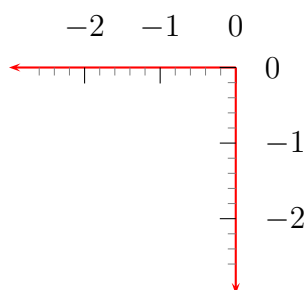
```



```

1 \pspicture(-3,-3)(3,3.5)
2   \psaxes[subticks=5,ticks=6pt,
3     subticksize=0.5,tickstyle=top
4     ]{->}(0,0)(3,3)(-3,-3)
5 \endpspicture

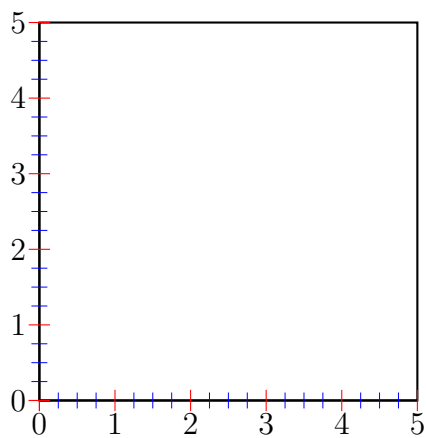
```



```

1 \pspicture(0,0.5)(-3,-3)
2   \psaxes[subticks=5,ticks=6pt,
3     subticksize=0.5,tickstyle=bottom,
4     linecolor=red]{->}(-3,-3)
5 \endpspicture

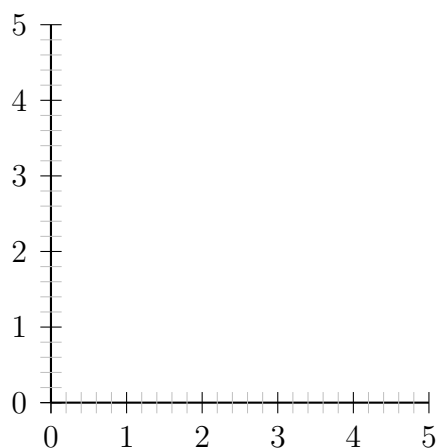
```



```

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

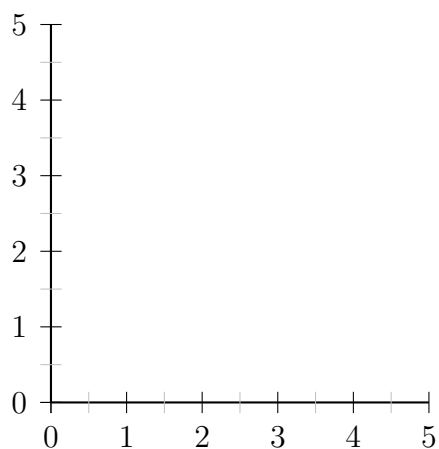
```



```

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

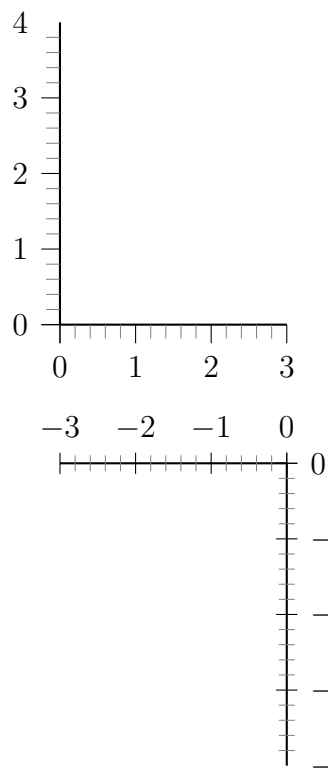
```



```

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

```



```

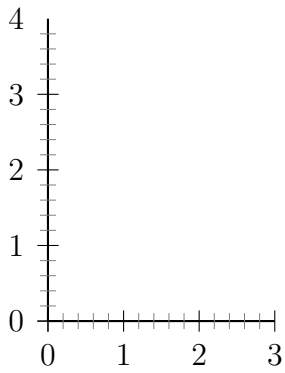
1 \pspicture(3,4.5)
2   \psaxes[subticks=5,ticksize=-7pt 0](3,4)
3 \endpspicture

```

```

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

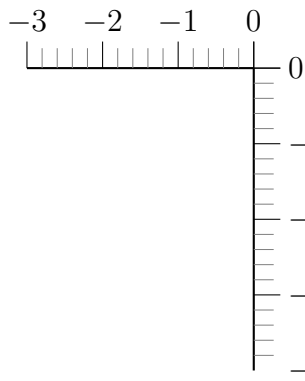
```



```

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

```



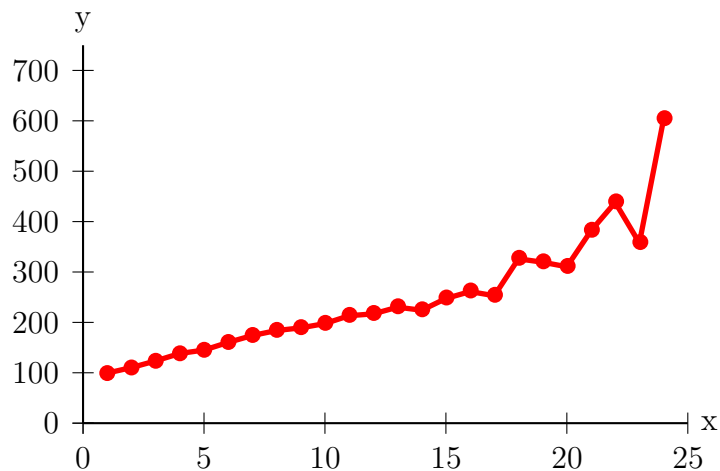
```

1 \pspicture(0,1)(-3,-4)
2   \psaxes[axesstyle=axes,subticks=5,%
3     ticksize=0 10pt,labelsep=13pt](-3,-4)
4 \endpspicture

```

## 15.17 xlabelFactor and ylabelFactor

When having big numbers as data records then it makes sense to write the values as  $\langle number \rangle \cdot 10^{\langle exp \rangle}$ . These new options allow to define the additional part of the value.



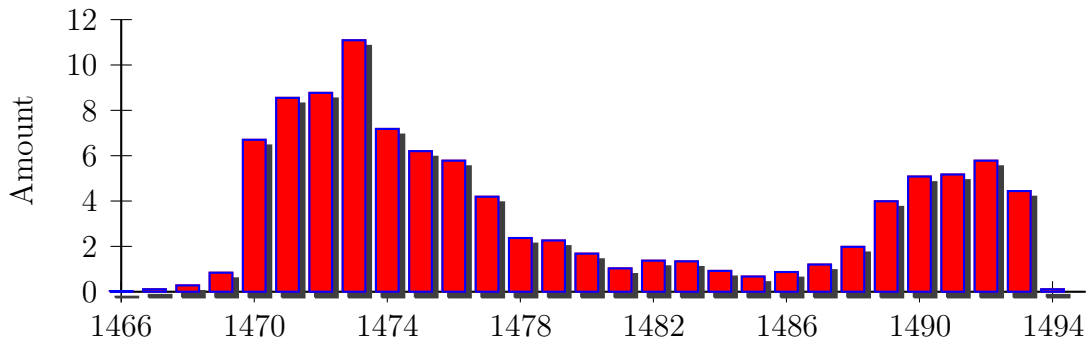
```

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

```

## 15.18 Plot style bar and option barwidth

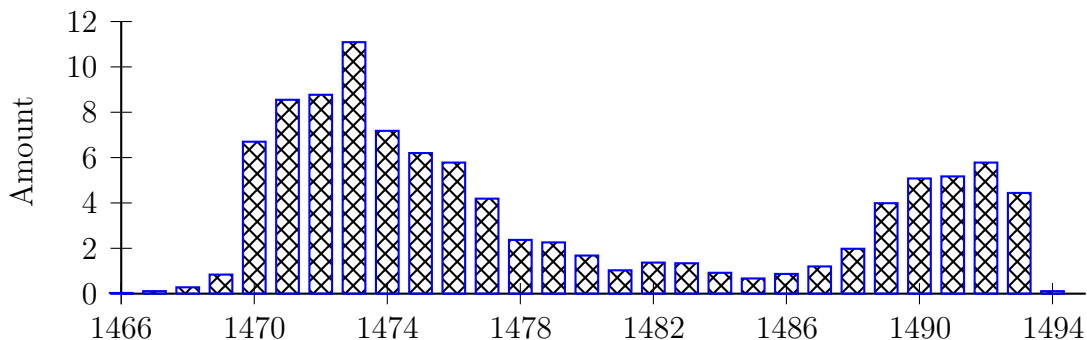
This option allows to draw bars for the data records. The width of the bars is controlled by the option `barwidth`, which is set by default to value of 0.25cm, which is the total width.



```

1 \psset{xunit=.44cm,yunit=.3cm}
2 \begin{pspicture}(-2,-1.5)(29,13)
3   \psaxes[axesstyle=axes, Ox=1466, Oy=0, Dx=4, Dy=2, %
4     ylabelFactor={\, \%}](-3)(29,12)
5   \listplot[shadow=true, linecolor=blue, plotstyle=bar, barwidth=0.3cm,
6     fillcolor=red, fillstyle=solid]{\barData}
7   \rput{90}(-3,6.25){Amount}
8 \end{pspicture}

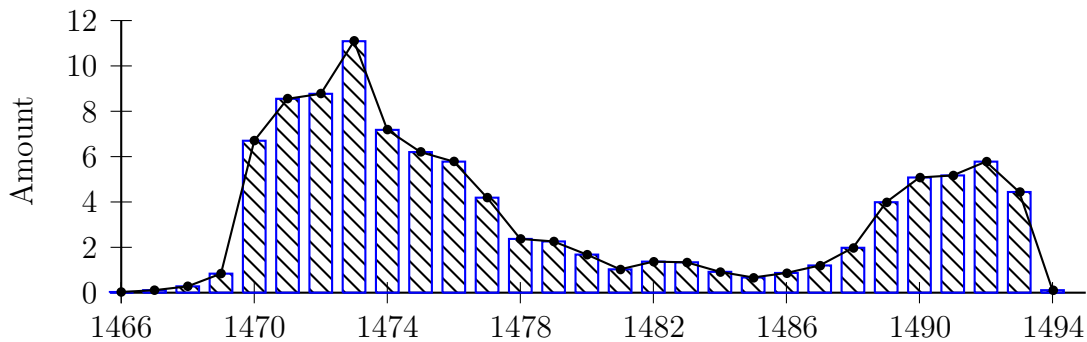
```



```

1 \psset{xunit=.44cm,yunit=.3cm}
2 \begin{pspicture}(-2,-1.5)(29,13)
3   \psaxes[axesstyle=axes,0x=1466,0y=0,Dx=4,Dy=2,%
4     ylabelFactor={\,\,\%}](-3,12)
5   \listplot[linecolor=blue,plotstyle=bar,barwidth=0.3cm,
6     fillcolor=red,fillstyle=crosshatch]{\barData}
7   \rput{90}(-3,6.25){Amount}
8 \end{pspicture}

```



```

1 \psset{xunit=.44cm,yunit=.3cm}
2 \begin{pspicture}(-2,-1.5)(29,13)
3   \psaxes[axesstyle=axes,0x=1466,0y=0,Dx=4,Dy=2,%
4     ylabelFactor={\,\,\%}](-3,12)
5   \listplot[linecolor=blue,plotstyle=bar,barwidth=0.3cm,
6     fillcolor=red,fillstyle=vlines]{\barData}
7   \listplot[showpoints=true]{\barData}
8   \rput{90}(-3,6.25){Amount}
9 \end{pspicture}

```

## 15.19 New options for \readdata

By default the macros `\readdata` reads every data record, which could be annoying when there are more than 10000 records to read. The package `pst-plot-add` defines an additional key `nStep`, which allows to read only a selected part of the data records, e.g. `nStep=10`, only every 10<sup>th</sup> records is saved.

```

1 \readdata[nStep=10]{\dataA}{stressrawdata.dat}

```

The default value for `nStep` is 1.

## 16 New options for \listplot

By default the plot macros `\dataplot`, `\fileplot` and `\listplot` plot every data record. The package `pst-plot-add` defines 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 plot (0, 1, 2, ...) and the "x" ones the x-values of the data records.

Name	Default setting
<code>nStart</code>	1
<code>nEnd</code>	{}
<code>nStep</code>	1
<code>xStart</code>	{}
<code>xEnd</code>	{}
<code>yStart</code>	{}
<code>yEnd</code>	{}
<code>xStep</code>	0
<code>plotNo</code>	1
<code>plotNoMax</code>	1

These 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 [\[3\]](#)):

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

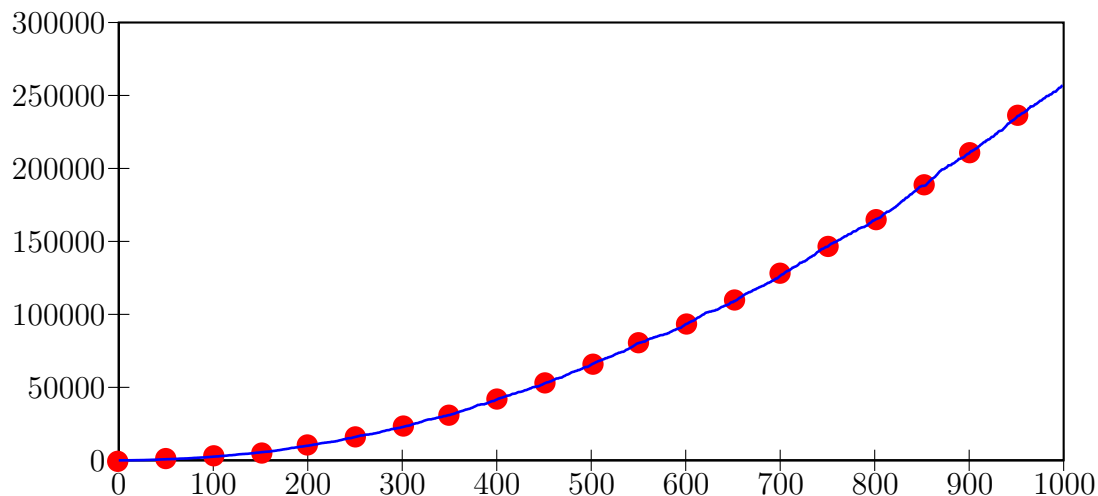
The use `nStep` and `xStep` options make only real sense when also using the option `plotstyle=dots`. Otherwise the coordinates are connected by a line as usual. Also the `xStep` option needs increasing x values. Pay attention that `nStep` can be used for `\readdata` and for `\listplot`. If used in both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

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

## 16.1 Example for `nStep/xStep`

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



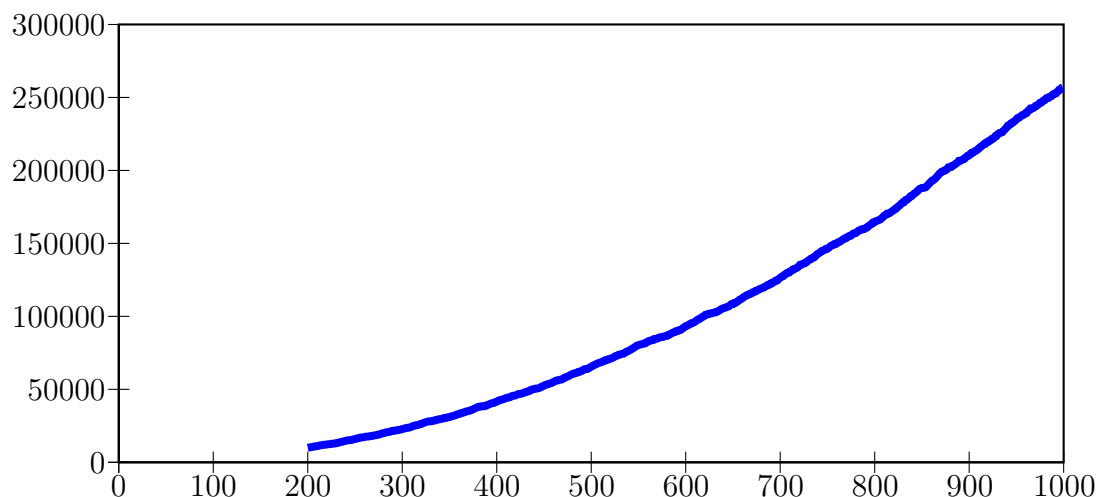


```

1 \readdata{\data}{examples/data.dat}
2 \psset{xunit=0.125mm,yunit=0.0002mm}
3 \begin{pspicture}(-80,-30000)(1000,310000)
4 \psaxes[axesstyle=frame,Dx=100,dx=100,Dy=50000,dy=50000](1000,300000)
5 \listplot[nStep=50,linewidth=3pt,linecolor=red,plotstyle=dots]{\data}
6 \listplot[linewidth=1pt,linecolor=blue]{\data}
7 \end{pspicture}

```

## 16.2 Example for nStart/xStart

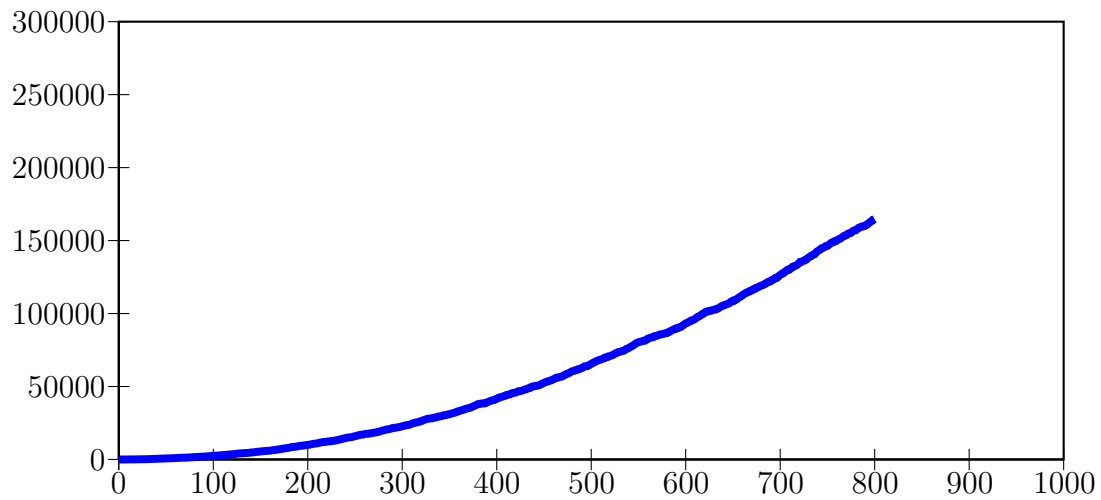


```

1 \readdata{\data}{examples/data.dat}
2 \psset{xunit=0.125mm,yunit=0.0002mm}
3 \begin{pspicture}(-80,-30000)(1000,310000)
4 \psaxes[axesstyle=frame,Dx=100,dx=100,Dy=50000,dy=50000](1000,300000)
5 \listplot[nStart=200,linewidth=3pt,linecolor=blue]{\data}
6 \end{pspicture}

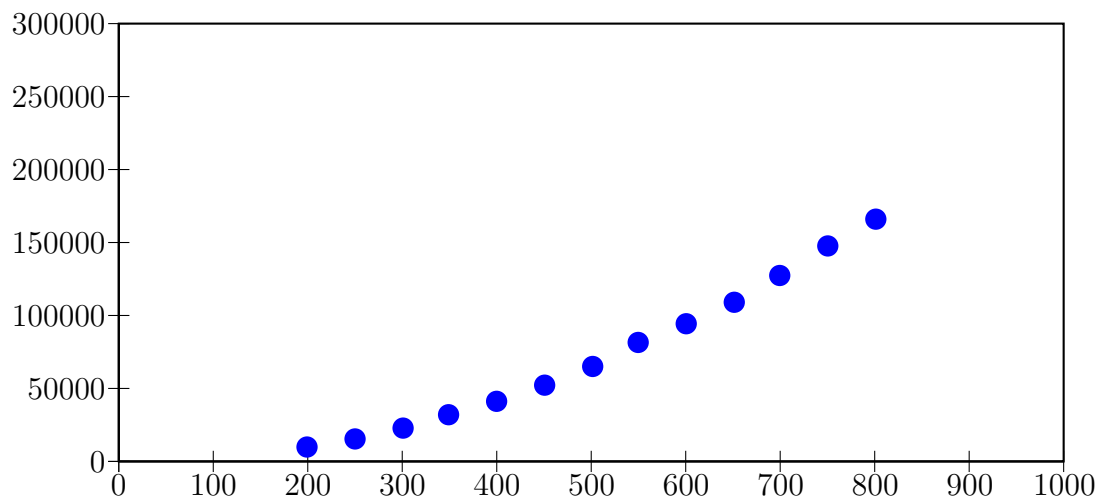
```

### 16.3 Example for nEnd/xEnd



```
1 \readdata{\data}{examples/data.dat}
2 \psset{xunit=0.125mm,yunit=0.0002mm}
3 \begin{pspicture}(-80,-30000)(1000,310000)
4 \psaxes[axesstyle=frame,Dx=100,dx=100,Dy=50000,dy=50000](1000,300000)
5 \listplot[nEnd=800,linewidth=3pt,linecolor=blue]{\data}
6 \end{pspicture}
```

### 16.4 Example for all new options



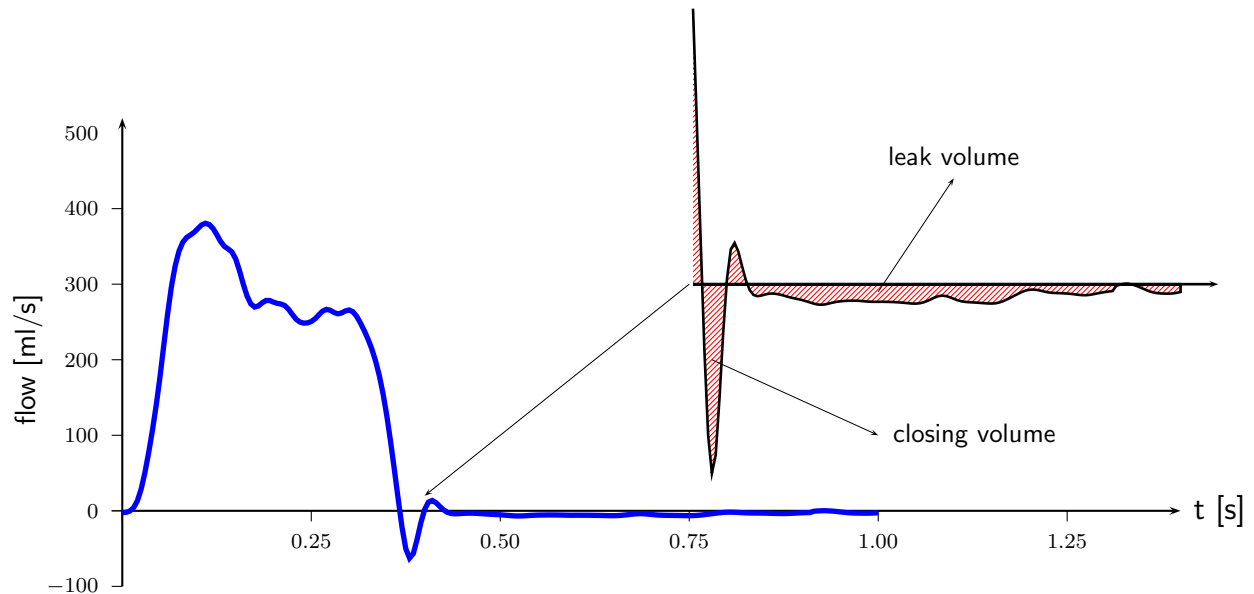
```

1 \readdata{\data}\examples\data.dat}
2 \psset{xunit=0.125mm,yunit=0.0002mm}
3 \begin{pspicture}(-80,-30000)(1000,310000)
4 \psaxes[axesstyle=frame,Dx=100,dx=100,Dy=50000,dy=50000](1000,300000)
5 \listplot[nStart=200,nEnd=800,nStep=50,linewidth=3pt,linecolor=blue,%
6   plotstyle=dots]{\data}
7 \end{pspicture}

```

## 16.5 Example for xStart

This example shows the use of the same plot with different units and different `xStart` value. The blue curve is the original plot of the data records. To show the important part of the curve there is another one plotted with a greater `yunit` and a start value of `xStart=0.35`. This makes it possible to have a kind of a zoom to the original graphic.

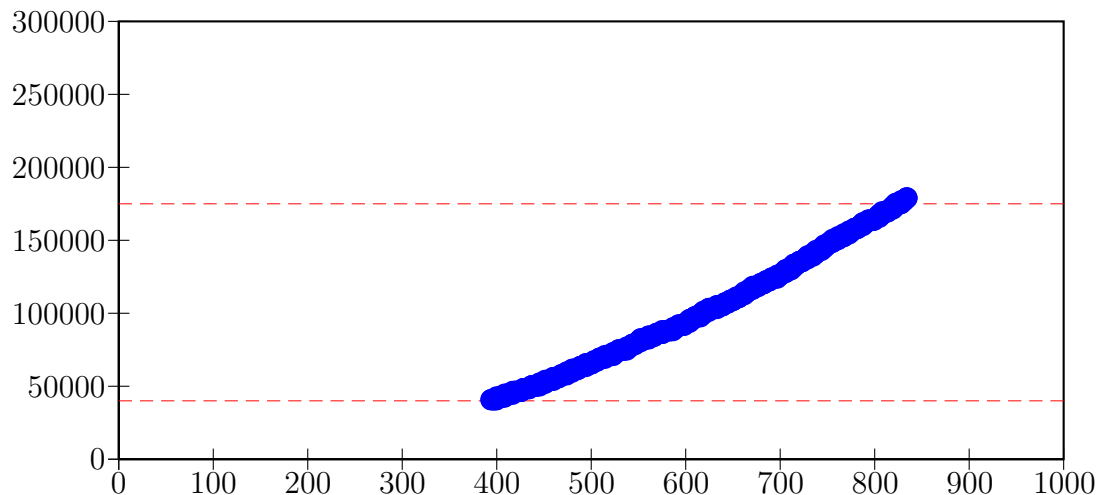


```

1 \psset{xunit=10cm, yunit=0.01cm,xLabel={\scriptsize\sffamily},yLabel={\scriptsize\sffamily}}
2 \readdata{\data}{examples/data3.dat}
3 \begin{pspicture}(-0.1,-100)(1.5,700.0)
4   \psaxes[Dx=0.25,Dy=100,dy=100\psyunit,tickstyle=bottom]{->}(0,0)(0,-100)(1.4,520)
5   \uput[0](1.4,0){\textsf{t [s]}}
6   \rput(-0.125,200){\rotateleft{\small\sffamily flow [ml/s]}}
7   \listplot[linewidth=2pt, linecolor=blue]{\data}
8   \rput(0.4,300){
9     \pscustom[yunit=0.04cm, linewidth=1pt]{%
10      \listplot[xStart=0.355]{\data}
11      \psline(1,-2.57)(1,0)(0.355,0)
12      \fill[fillstyle=hlines,fillcolor=gray,hatchwidth=0.4pt,hatchsep=1.5pt,hatchcolor=red]%
13      \psline[linewidth=0.5pt]{->}(0.7,0)(1.05,0)
14    }%
15  }
16  \psline[linewidth=.01]{->}(0.75,300)(0.4,20)
17  \psline[linewidth=.01]{->}(1,290)(1.1,440)
18  \rput(1.1,470){\footnotesize\sffamily leak volume}
19  \psline[linewidth=.01]{->}(0.78,200)(1,100)
20  \rput[1](1.02,100){\footnotesize\sffamily closing volume}
21 \end{pspicture}

```

## 16.6 Example for yStart/yEnd



```

1 \readdata{\data}{examples/data.dat}
2 \psset{xunit=0.125mm,yunit=0.0002mm}
3 \begin{pspicture}(-80,-30000)(1000,310000)
4   \psaxes[axesstyle=frame,Dx=100,dx=100,Dy=50000,dy=50000](1000,300000)
5   \psset{linewidth=0.1pt,linestyle=dashed,linecolor=red}
6   \psline(0,40000)(1000,40000)
7   \psline(0,175000)(1000,175000)
8   \listplot[yStart=40000,yEnd=175000,linewidth=3pt,linecolor=blue,
9     plotstyle=dots]{\data}
\end{pspicture}

```

## 16.7 Example for plotNo/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
...

```

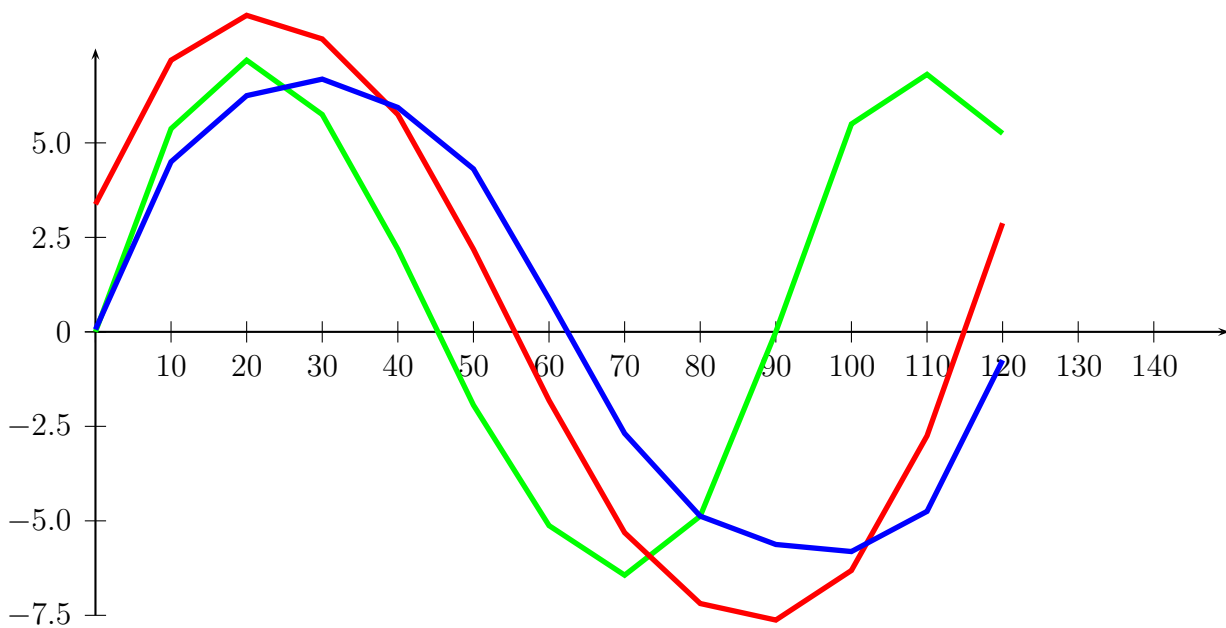
you can select the  $y$  value which should be plotted. The option `plotNo` marks the plotted value (default 1) and the option `plotNoMax` tells `pst-plot` how many  $y$  values are present. There are no real restrictions in the maximum number for `plotNoMax`.

We have the following data file:

```

[% file examples/data.dat
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 ( $x$   $y_1$   $y_2$   $y_3$ ). This can be plotted without any modification to the data file:



```

1 \readdata\Data{examples/dataMul.dat}
2 \psset{xunit=0.1cm, yunit=0.5cm}
3 \begin{pspicture}(0,-7.5)(150,10)
4 \psaxes[Dx=10,Dy=2.5]{->}(0,0)(0,-7.5)(150,7.5)
5 \psset{linewidth=2pt,plotstyle=line}
6 \listplot[linecolor=green,plotNo=1,plotNoMax=3]{\Data}
7 \listplot[linecolor=red,plotNo=2,plotNoMax=3]{\Data}
8 \listplot[linecolor=blue,plotNo=3,plotNoMax=3]{\Data}
9 \end{pspicture}

```

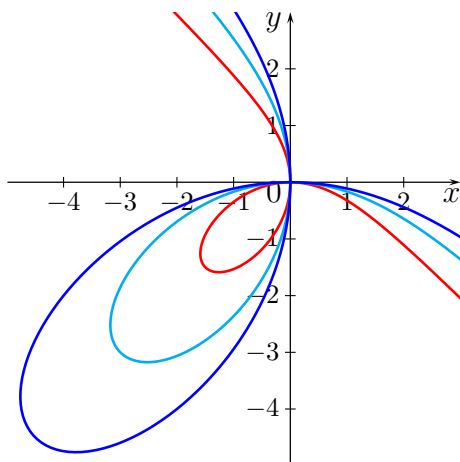
## 17 Polar plots

With the option `polarplot=false|true` it is possible to use `\psplot` in polar mode:

```
\psplot[polarplot=true,...]{<start angle>}{<end angle>}{<r(alpha)>}
```

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

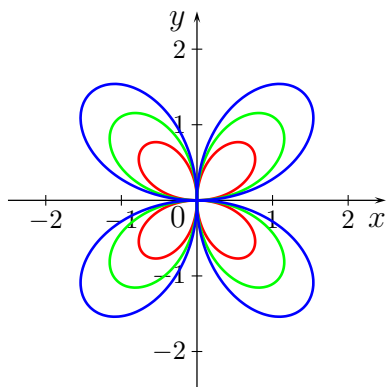
```
x sin dup mul x cos dup mul add sqrt
```



```

1 \resetOptions
2 \psset{plotpoints=200,unit=0.75}
3 \begin{pspicture}*(-5,-5)(3,3)
4   \psaxes[labelsep=.75mm,xyLabel=\
      footnotesize,
5     arrowlength=1.75,ticksiz=2pt,%
6     linewidth=0.17mm]{->}(0,0)(-4.99,-4.99)
7     (3,3)
8   \rput[Br](3,-.35){$x$}
9   \rput[tr](-.15,3){$y$}
10  \rput[Br](-.15,-.35){$0$}
11  \psset{linewidth=.35mm,polarplot=true}
12  \psplot[linecolor=red]{140}{310}{3 neg x
    sin mul x cos mul x sin 3 exp x cos 3
    exp add div}
13  \psplot[linecolor=cyan]{140}{310}{6 neg x
    sin mul x cos mul x sin 3 exp x cos 3
    exp add div}
14  \psplot[linecolor=blue]{140}{310}{9 neg x
    sin mul x cos mul x sin 3 exp x cos 3
    exp add div}
15 \end{pspicture}

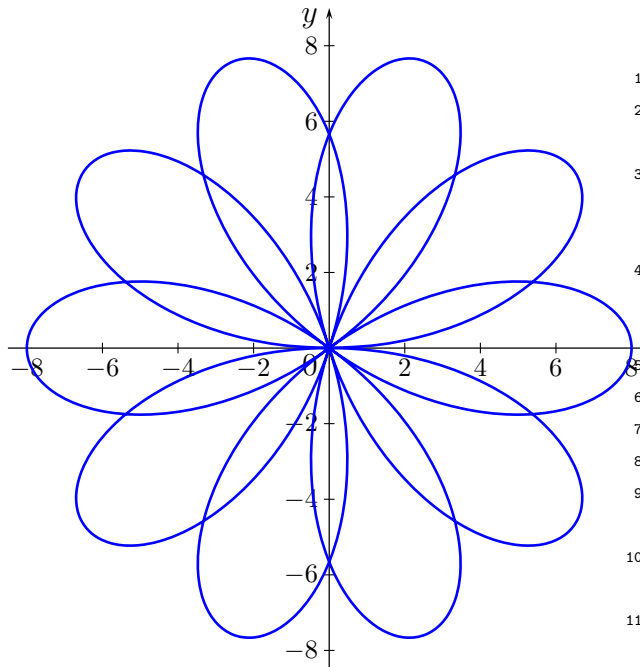
```



```

1 \resetOptions
2 \psset{plotpoints=200,unit=1}
3 \begin{pspicture}(-2.5,-2.5)(2.5,2.5)% Ulrich
4   \psaxes[labelsep=.75mm,xyLabel=\footnotesize,%
5     arrowlength=1.75,ticksiz=2pt,linewidth=0.17
6     mm]{->}(0,0)(-2.5,-2.5)(2.5,2.5)
7   \rput[Br](2.5,-.35){$x$}
8   \rput[tr](-.15,2.5){$y$}
9   \rput[Br](-.15,-.35){$0$}
10  \psset{linewidth=.35mm,plotstyle=curve,
11    polarplot=true}
12  \psplot[linecolor=red]{0}{360}{x cos 2 mul x
13    sin mul}
14  \psplot[linecolor=green]{0}{360}{x cos 3 mul x
15    sin mul}
16  \psplot[linecolor=blue]{0}{360}{x cos 4 mul x
17    sin mul}
18 \end{pspicture}

```



```

1 \psset{plotpoints=200,unit=0.5}
2 \begin{pspicture}(-8.5,-8.5)(9,9) %
   Ulrich Dirr
3 \psaxes[Dx=2,dx=2,Dy=2,dy=2,
   labelsep=.75mm,xyLabel=\
   footnotesize,%
4   arrowlength=1.75,ticksiz=2pt,
   linewidth=0.17mm]{->}(0,0)
   (-8.5,-8.5)(9,9)
5 \rput[Br](9,-.7){$x$}
6 \rput[tr](-.3,9){$y$}
7 \rput[Br](-.3,-.7){$0$}
8 %
9 \psset{linewidth=.35mm,plotstyle=
   curve,polarplot=true}
10 \psplot[linecolor=blue
   ]{0}{720}{8 2.5 x mul sin mul}
11 \end{pspicture}

```

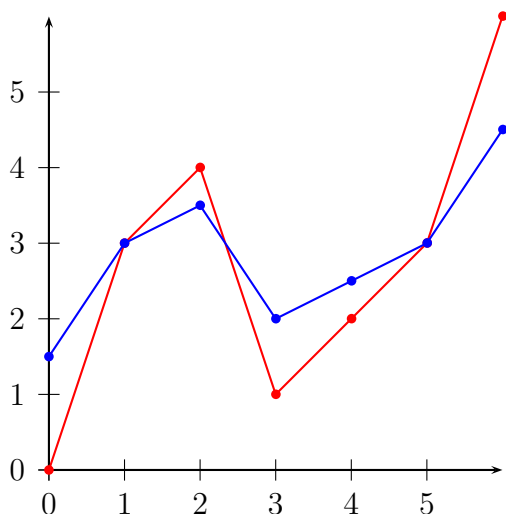
## 18 New commands and environments

### 18.1 \pstScalePoints

The syntax is

`\pstScalePoints(xScale,yScale){xPS}{yPS}`

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



```

1 \def\data{0 0
2 1 3
3 2 4
4 3 1
5 4 2
6 5 3
7 6 6}
8 \begin{pspicture}(6,6)
9 \psaxes{->}(6,6)
10 \listplot[showpoints=true,linecolor=red]{\
   data}
11 \pstScalePoints(1,0.5){}{3 add}
12 \listplot[showpoints=true,linecolor=blue]{\
   data}
13 \end{pspicture}

```



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.

```
\pstScalePoints(1,1){}{}
```

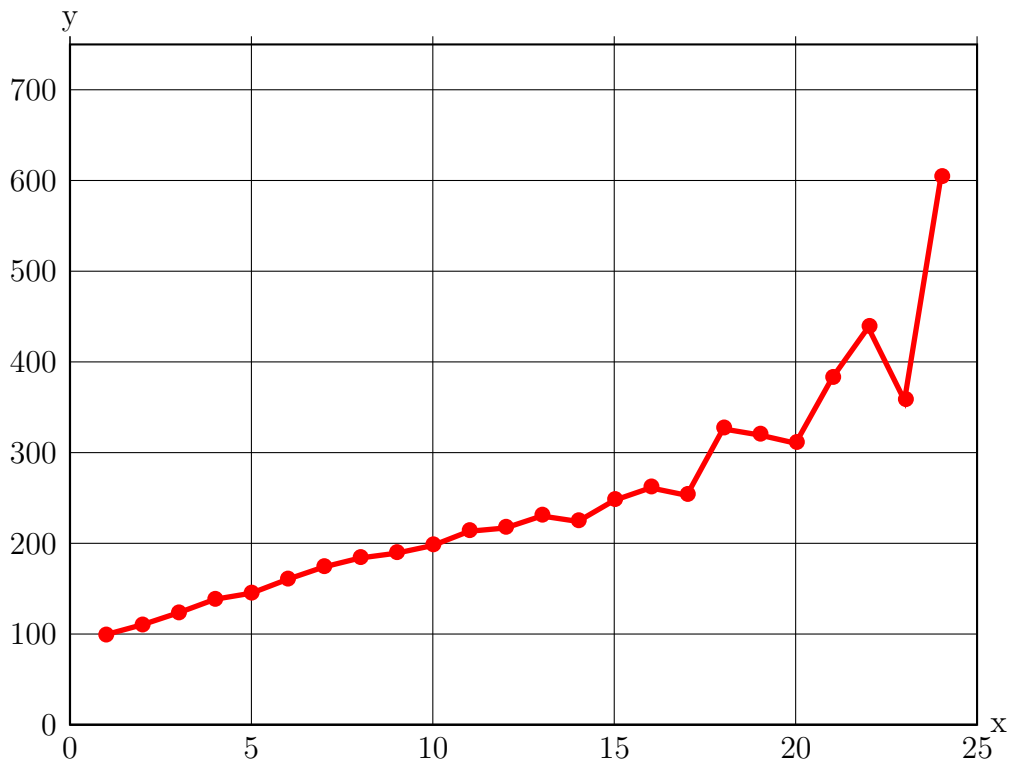
## 18.2 psgraph environment

This new environment 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>] (xMin,yMin) (xMax,yMax){xLength}{yLength}
...
\endpsgraph

\begin{psgraph}[<options>] (xMin,yMin) (xMax,yMax){xLength}{yLength}
...
\end{psgraph}
```

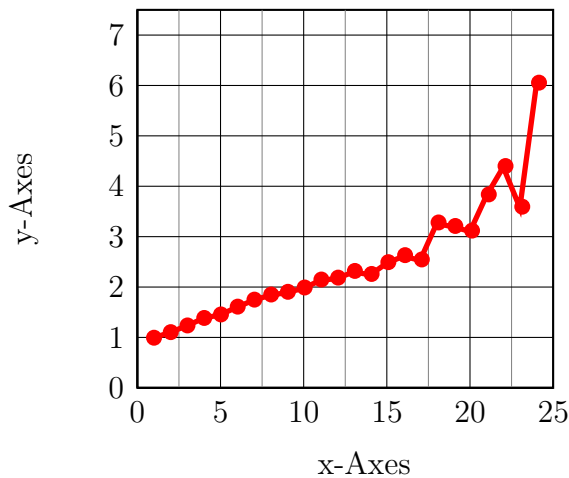
where the options are valid only for the the `\psaxes` macro.



```

1 \readdata{\data}{demo1.dat}
2 \pstScalePoints(1,0.000001){}{% (x,y){additional x operator}{y op}
3 \psset{llx=-0.5cm, lly=-1cm}
4 \psgraph[axesstyle=frame,xticks=0 759, yticks=0 25,%
5     subticks=0, ylabelFactor={\cdot 10^6},%
6     Dx=5, dy=100\psunit, Dy=100](0,0)(25,750){12cm}{9cm}
7     parameters
8 \listplot[linecolor=red, linewidth=2pt, showpoints=true]{\data}
\endpsgraph

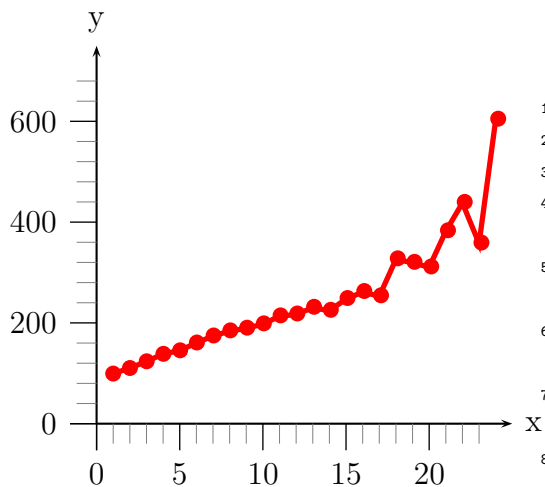
```



```

1 \readdata{\data}{demo1.dat}
2 \psset{xAxisLabel=x-Axes, yAxisLabel=y-
3     Axes, llx=-1cm,%
4     xAxisLabelPos={3cm, -1cm},
5     yAxisLabelPos={-1.5cm, 2.5cm}}
6 \pstScalePoints(1,0.00000001){}{%
7 \begin{psgraph}[axesstyle=frame,
8     xticks=0 7.5, yticks=0 25,
9     subticks=1,%
10     ylabelFactor={\cdot 10^8}, Dx=5, Dy
11     =1, xsubticks=2](0,0)(25,7.5){5.5
12     cm}{5cm}
13 \listplot[linecolor=red, linewidth=2pt
14     , showpoints=true]{\data}
15 \end{psgraph}

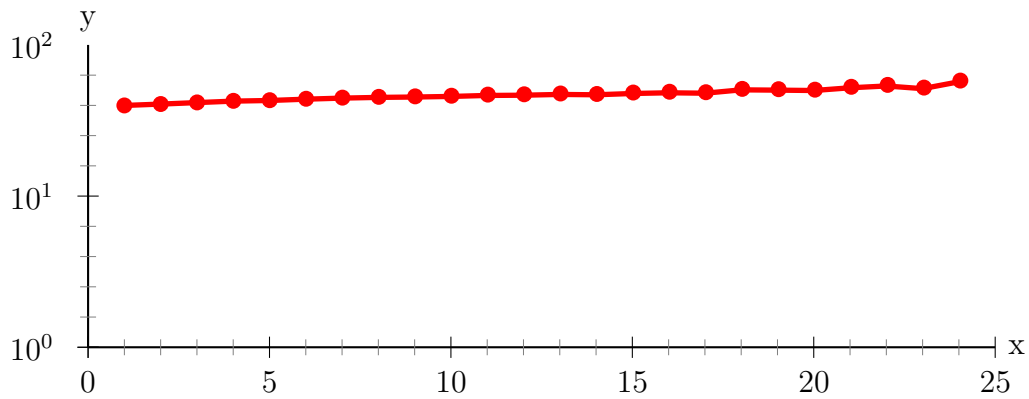
```



```

1 \readdata{\data}{demo1.dat}
2 \psset{llx=-0.5cm, lly=-1cm}
3 \pstScalePoints(1,0.000001){}{%
4 \psgraph[arrows=->, Dx=5, dy=200\psunit, Dy
5     =200,%
6     subticks=5, ticksize=-10pt 0, tickwidth
7     =0.5pt,%
8     subtickwidth=0.1pt](0,0)(25,750){5.5cm
9     }{5cm}
10 \listplot[linecolor=red, linewidth=2pt,
11     showpoints=true,]{\data}
12 \endpsgraph

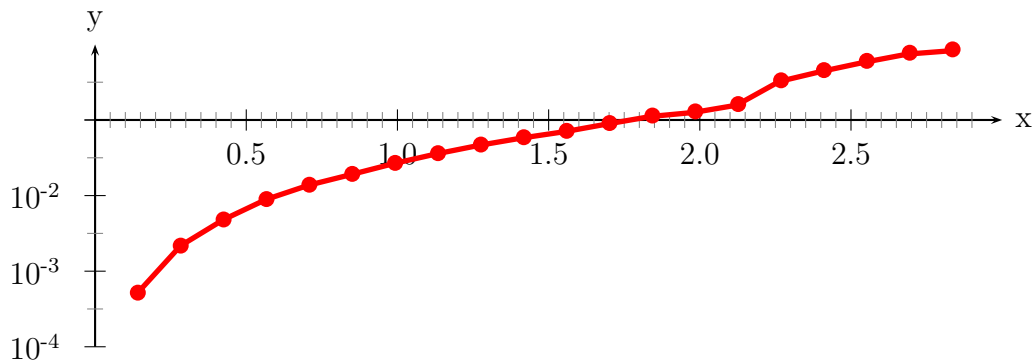
```



```

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

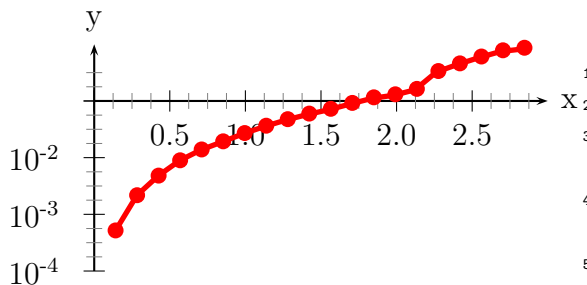
```



```

1 \readdata{\data}{demo0.dat}
2 \pstScalePoints(1,1){}{log}
3 \begin{psgraph}[arrows=->,Dx=0.5,ylogBase=10,Oy=-1,xsubticks=10,%
4   ysubticks=2](0,-3)(3,1){12cm}{4cm}
5   \listplot[linecolor=red,linewidth=2pt,showpoints=true]{\data}
6 \end{psgraph}

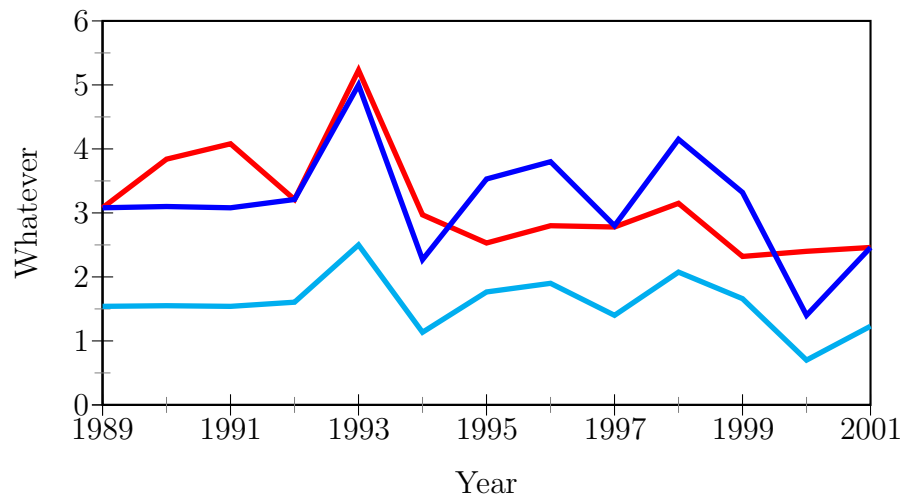
```



```

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

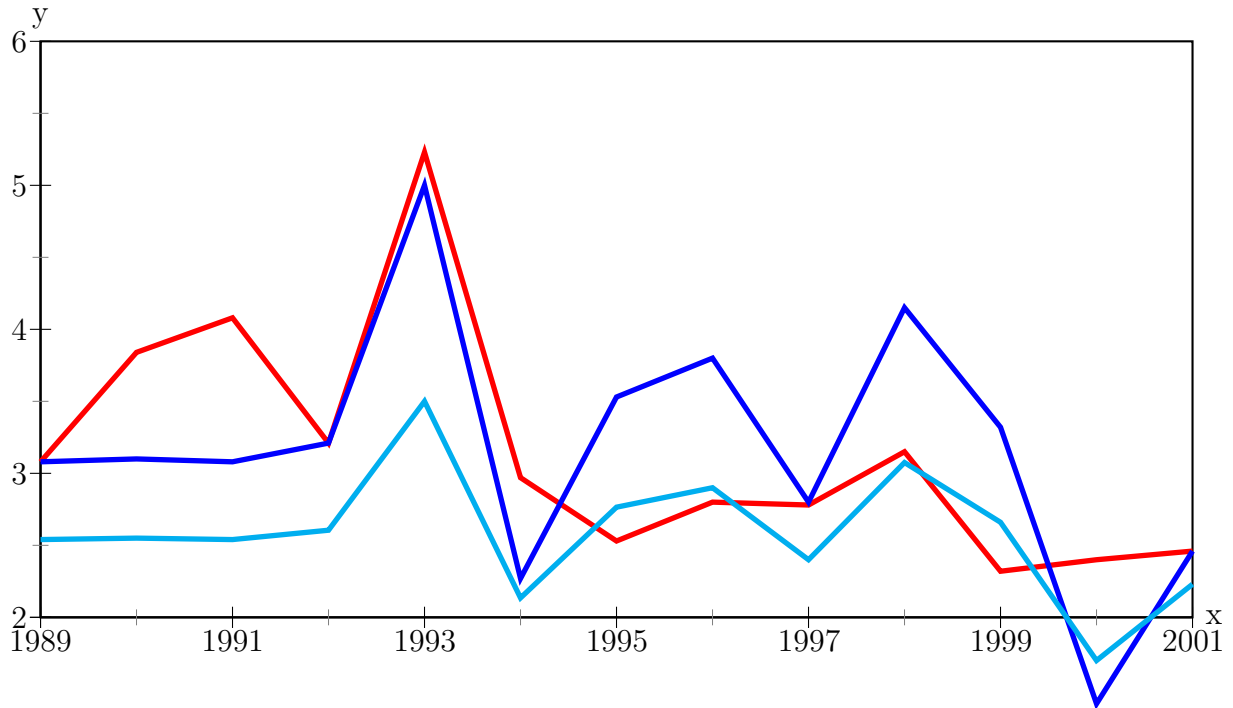
```



```

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

```

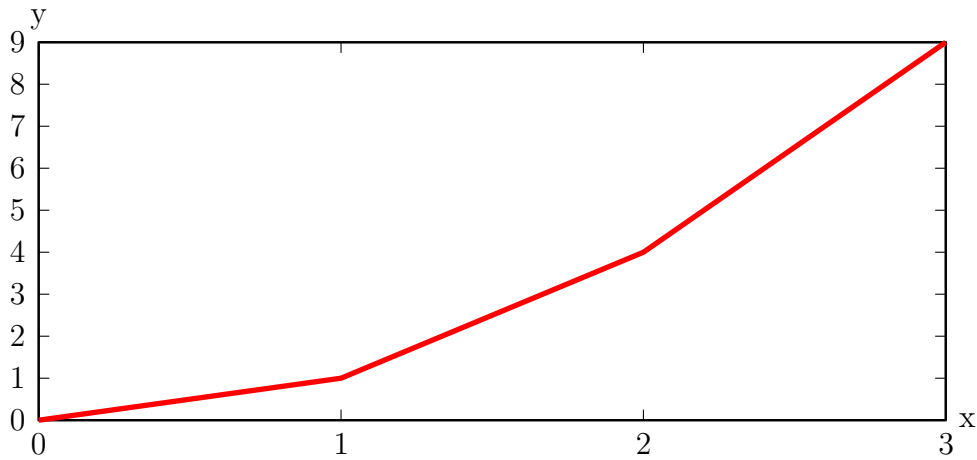


```

1 \psset{llx=-0.5cm,lly=-0.75cm}
2 \pstScalePoints(1,1){1989 sub}{2 sub}
3 \begin{psgraph}[axesstyle=frame,Dx=2,Ox=1989,Oy=2,subticks=2](0,0)(12,4){6
   in}{3in}%
4   \listplot[linecolor=red,linewidth=2pt]{\data}
5   \listplot[linecolor=blue,linewidth=2pt]{\dataII}
6   \listplot[linecolor=cyan,linewidth=2pt,yunit=0.5]{\dataII}
7 \end{psgraph}

```

An example with ticks on every side of the frame:



```

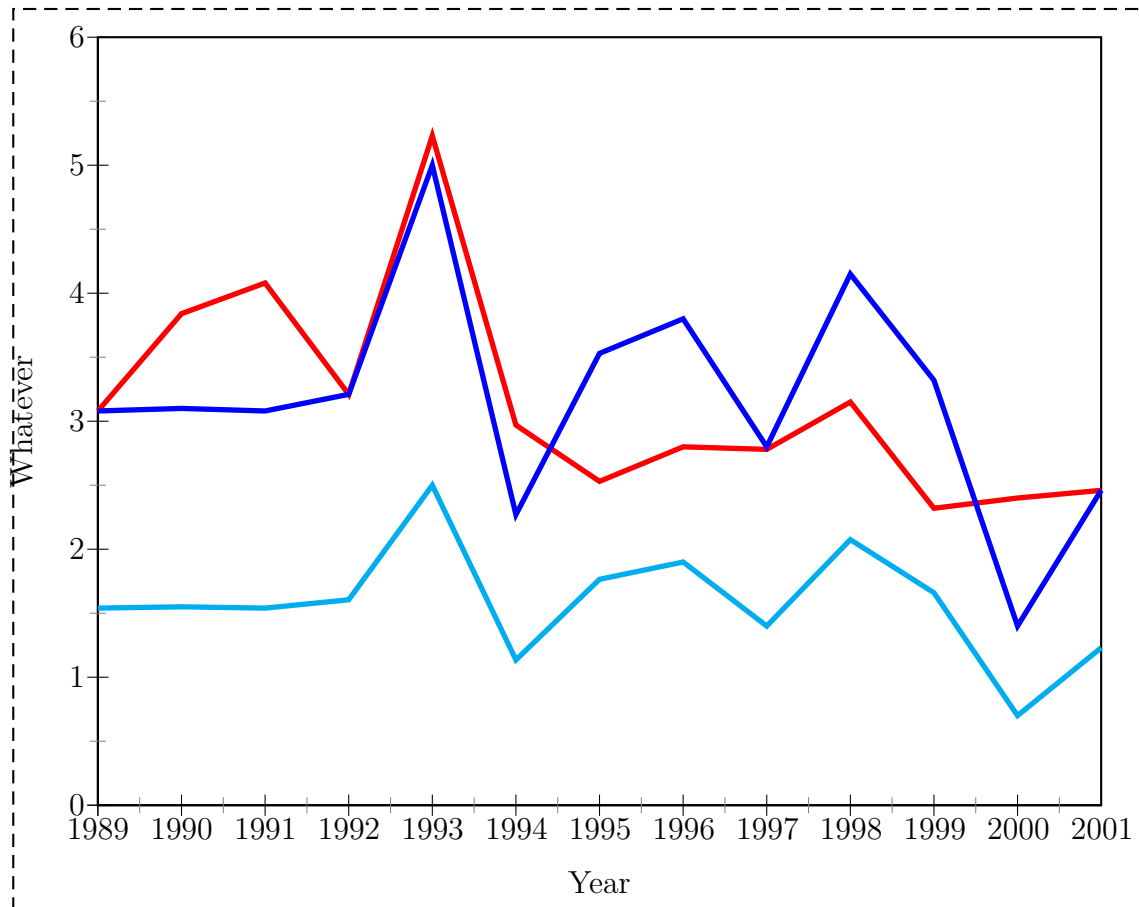
1 \def\data{0 0 1 1 2 4 3 9}
2 \begin{psgraph}[axesstyle=frame,tickstyle=top](0,0)(3.0,9.0){12cm}{5cm}
3   \psaxes[axesstyle=frame,labels=None,tickstyle=bottom](3,9)(0,0)(3,9)
4   \listplot[linecolor=red,linewidth=2pt]{\data}
5 \end{psgraph}

```

### 18.2.1 The new options

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
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

There is one restriction in using the trim parameters, they must be set **before** `psgraph` is called. They are senseless, when using as parameters of `psgraph` itself.



```

1 \psset{llx=-1cm,lly=-1.25cm,urx=0.5cm,ury=0.1in,xAxisLabel=Year,%
2   yAxisLabel=Whatever,xAxisLabelPos={.4\linewidth,-0.4in},%
3   yAxisLabelPos={-0.4in,2in}}
4 \pstScalePoints(1,1){1989 sub}{}
5 \psframebox[linestyle=dashed,boxsep=0pt]{%
6 \begin{psgraph}[axesstyle=frame,0x=1989,subticks=2](0,0)(12,6){0.8\
7   \linewidth}{4in}%
8   \listplot[linecolor=red,linewidth=2pt]{\data}%
9   \listplot[linecolor=blue,linewidth=2pt]{\dataII}%
10  \listplot[linecolor=cyan,linewidth=2pt,yunit=0.5]{\dataII}%
11 \end{psgraph}%

```

### 18.3 \resetOptions

Sometimes it is difficult to know what options which are changed inside a long document are different to the default one. With this macro all options depending to `pst-plot` can be reset. This depends to all options of the packages `pstricks`, `pst-plot` and `pst-node`.

## 19 Credits

Hendri Adrians | Ulrich Dirr | Hubert Gäßlein | Denis Girou | Peter Hutnick | Christophe Jorssen | Manuel Luque | Jens-Uwe Morawski | Tobias Nähring | Rolf Niepraschk | Dominique Rodriguez | Timothy Van Zandt

## References

- [1] Denis Girou. Présentation de PSTricks. *Cahier GUTenberg*, 16:21–70, April 1994.
- [2] Michel Goossens, Frank Mittelbach, and Alexander Samarin. *The L<sup>A</sup>T<sub>E</sub>X Graphics Companion*. Addison-Wesley Publishing Company, Reading, Mass., 1997.
- [3] Laura E. Jackson and Herbert Voß. Die plot-funktionen von `pst-plot`. *Die T<sub>E</sub>Xnische Komödie*, 2/02:27–34, June 2002.
- [4] Nikolai G. Kollock. *PostScript richtig eingesetzt: vom Konzept zum praktischen Einsatz*. IWT, Vaterstetten, 1989.
- [5] Herbert Voß. *Chaos und Fraktale selbst programmieren: von Mandelbrotmengen über Farbmanipulationen zur perfekten Darstellung*. Franzis Verlag, Poing, 1994.
- [6] Herbert Voß. Die mathematischen Funktionen von PostScript. *Die T<sub>E</sub>Xnische Komödie*, 1/02, March 2002.
- [7] Timothy van Zandt. *PSTricks - PostScript macros for generic T<sub>E</sub>X*. <http://www.tug.org/application/PSTricks>, 1993.
- [8] Timothy van Zandt. *multido.tex - a loop macro, that supports fixed-point addition*. [CTAN:/graphics/pstricks/generic/multido.tex](http://CTAN:/graphics/pstricks/generic/multido.tex), 1997.
- [9] Timothy van Zandt. *pst-plot: Plotting two dimensional functions and data*. [CTAN:/graphics/pstricks/generic/pst-plot.tex](http://CTAN:/graphics/pstricks/generic/pst-plot.tex), 1999.
- [10] Timothy van Zandt and Denis Girou. Inside PSTricks. *TUGboat*, 15:239–246, September 1994.

## 20 Change log

```
1 v 2.18 add \ncbarr
2 v 2.17 new multiple arrows
3 v 2.16 small bugfix
4 v 2.15 one more step forward with pst-xkey
5 v 2.14 fix bug with subticks=0
6 v 2.13 fix bug with comma
7 v 2.12 new plot option bar
```



```

8 v 2.11 small tweaks to psaxes for one dimension (showorigin)
9 v 2.10 modified dash (TN)
10 v 2.09 renaming \psvlabel and \pshlabel and fixing a bug in \psbrace
11 v 2.08 pspolarplot now also with algebraic option
12 v 2.07 adding \rmultiput
13 v 2.06 fixing a bug in \psbrace
14 v 2.05 adding the algebraic stuff
15 v 2.04 fixing the axes ends - first final version
16 v 2.03 fixing the bottom/top axes stuff
17 v 2.02 some more improvements and new options
18 v 2.01 improvements and new options
19 v 2.00 totally new psaxes macro
20 v 1.69 some more #5\spaces
21 v 1.68 revert some #5\spaces
22 v 1.67 use pst@divide instead of divide for psgraph
23 v 1.66 changing logBase to xylogBase
24 v 1.65 changing the loglines
25 v 1.64 bugfix for ticklines
26 v 1.63 some more modification to \psgraph
27 v 1.62 some modification to \psgraph
28 v 1.61 added [x|y]labelFactor
29 v 1.60 new macro \psgraph
30 v 1.51 delete bugfix for ghshb, now in the package pst-ghsb
31 v 1.50 new option polarplot
32 v 1.47 some modification in the logbase part
33 v 1.46 fix another bug with logbase
34 v 1.45 fix a bug with logbase
35 v 1.44 delete frame and yOffset option to fix clasjh with pst-fill
36 v 1.43 add \psLNode \psLCNode
37 v 1.42 add right/lefthookarrow
38 v 1.41 set lengths for psbrace to mm
39 v 1.40 new option ticklines and deleted the ellipsis stuff,
40     which is now part of pstricks.tex -- patch 15
41 v 1.30 bugfix for \preparePoints
42 v 1.29 bugfix for pst-3d and new macro tColor
43 v 1.28 some more bugfixes for pstricks and pst-tree
44 v 1.27 some more bugfixes for pstricks and this package
45 v 1.26 several bugfixes for pstricks
46 v 1.25 bugfix for a spurious blank in readdata, added new dash option
    with
47     four parameters
48 v 1.24 moved psplotPolynomial to pst-func
49 v 1.23 no need for loading pstcol.sty
50 v 1.22 small tweaks
51 v 1.21 make yOffset working
52 v 1.2 added psplotPolynomial
53 v 1.13 make nStep working for readdata, too
54 v 1.12 fixed bug with spurious blank in \endpspicture
55 v 1.11 added stuff for data files like x,y1,y2,y3,y4,...
56 v 1.10 new numerical macro \pst@mod
57 v 1.0h fix a bug in frame setting
58 v 1.0g added option frame
59 v 1.0f fixed a new introduced bug with xyAxes
60 v 1.0e fixed some bugs with xyAxes options

```

```

61 v 1.0d added more options for \resetOptions
62 v 1.0c added \resetPlotOptions
63 v 1.0b added options xAxis and yAxis
64 v 1.0a added \ncdiagg
65 v 1.0 initial version, which collects all the other new macros for
66     pst-plot and pst-node
67 v 0.9c add \psFormatInt
68 v 0.9b add \pslineIII
69 v 0.9a add relative dashNo
70 v 0.9 add \pslineII
71 v 0.8e add bracePos
72 v 0.8d fix bug in arrows
73 v 0.8c small tweaks to \@@rput@iv
74 v 0.8b now every object can be passed to psbrace
75 v 0.8a fix bug with arrow
76 v 0.8 ArrowFill added
77 v 0.7a adding option asolid as fillstyle
78 v 0.7 ArrowInsidePos>1 for all macros
79 v 0.6 ArrowInsidePos>1 sets now the arrows every n-th pt
80 v 0.5a small improvements to the code (use of Pyth)
81 v 0.5 new psbezier and pcline to get the arrows in the right place!
82 v 0.4 fix bug in psbezier and nccurve, to get the right arrow position

```