

PSTricks

News - 2010

**new macros and bugfixes for the basic packages pstricks,
pst-plot, and pst-node**

February 9, 2010

2010

Package author(s):
Herbert Voß

Contents

I. pstricks – package	3
1. General	3
2. pstricks.sty	3
3. pstricks.tex (2.0– 2010/01/01)	3
3.1. Option algebraic	3
4. Numeric functions	5
5. Numeric functions	5
5.1. \pstFPadd, \pstFPsub, \pstFPMul, and \pstFPdiv	5
5.2. \pstFPMul and \pstFPDiv	6
6. The PostScript header files	6
6.1. pstricks.pro	6
6.2. pst-algparser.pro	7
6.3. IfTE	8
6.4. Derive function	10
6.5. Special arrow option arrowLW	10
7. \psforeach and \psForeach	12
 II. pst-node – package	 13
8. pst-node.tex	13
 III. pst-plot – package	 13
9. pst-plot.tex	13

Part I.

pstricks – package

1. General

There exists a new document class `pst-doc` for writing PSTricks documentations, like this news document. It depends on the KOMA-Script document class `scrartcl`. `pst-doc` defines a lot of special macros to create a good index. Take one of the already existing package documentation and look into the source file. Then it will be easy to understand, how all these macros have to be used.

When running `pdflatex` the title page is created with boxes and inserted with the macro `\AddToShipoutPicture` from the package `eso-pic`. It inserts the background title page image `pst-doc-pdf` to use directly `pdflatex`. When running `latex` the title page is created with PSTricks macros. This allows to use the Perl script `pst2pdf` or the package `pst-pdf` or `auto-pst-pdf` or any other program/package which supports PostScript code in the document.

2. pstricks.sty

No changes

3. pstricks.tex (2.0– 2010/01/01)

3.1. Option algebraic

The option `algebraic` moved from the other packages into the main package `pstricks` to get rid of the dependencies.

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

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

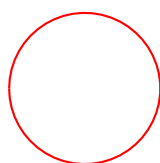
Setting the option `algebraic`, allow the user to describe all expression to be written in the classical algebraic notation (infix notation). The four arithmetic operations are obviously defined `+-*/`, and also the exponential operator `^`. The natural priorities are used : $3 + 4 \times 5^5 = 3 + (4 \times (5^5))$, and by default the computation is done from left to right. The following functions are defined :

sin, cos, tan, acos, asin	in radians
log, ln	
ceiling, floor, truncate, round	
sqrt	square root
abs	absolute value
fact	for the factorial
Sum	for building sums
IfTE	for an easy case structure

These options can be used with **all** plot macros.

Using the option algebraic implies that all angles have to be in radians!

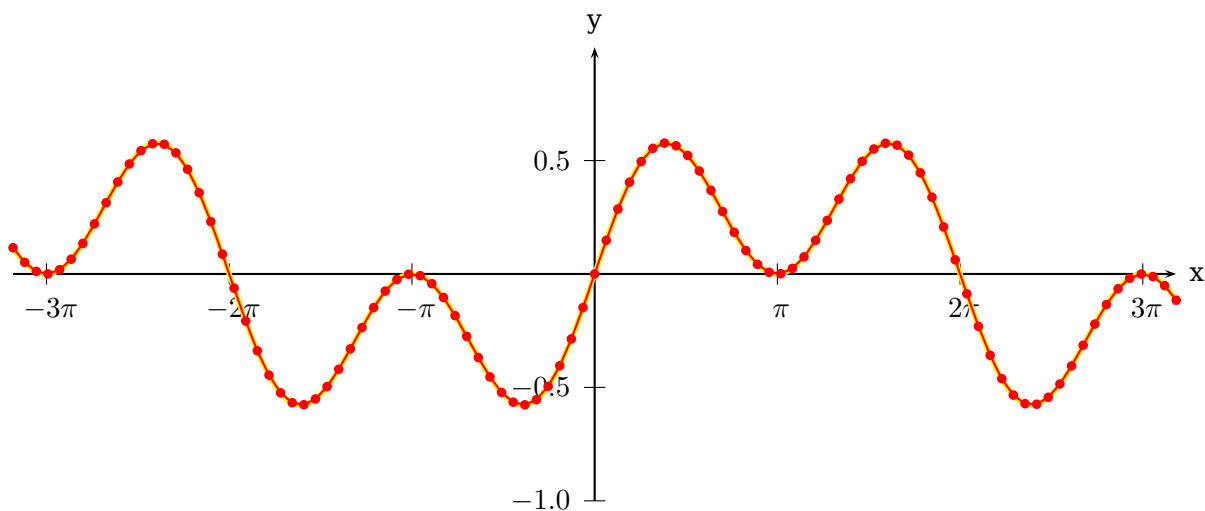
For the `\parametricplot` the two parts must be divided by the `|` character:



```

1 \begin{pspicture}(-0.5,-0.5)(0.5,0.5)
2 \parametricplot[algebraic,linecolor=red]{-3.14}{3.14}{cos(t)|sin(t)}
3 \end{pspicture}

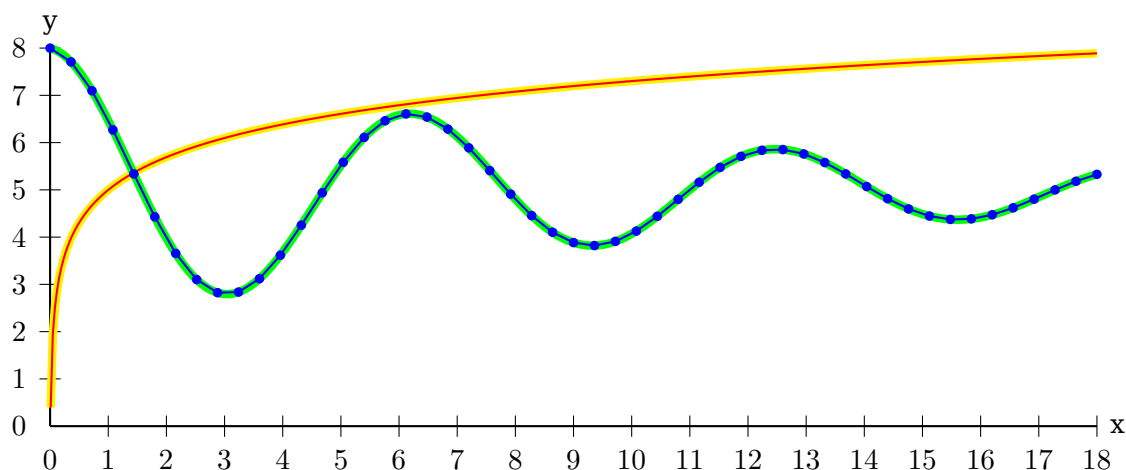
```



```

1 \psset{lly=-0.5cm}
2 \psgraph[trigLabels,dx=\psPi,dy=0.5,Dy=0.5]{->}(0,0)(-10,-1)(10,1){\linewidth}{6cm}
3 \psset{algebraic,plotpoints=1000}
4 \psplot[linecolor=yellow,linewidth=2pt]{-10}{10}{0.75*sin(x)*cos(x/2)}
5 \psplot[linecolor=red,showpoints=true,plotpoints=101]{-10}{10}{0.75*sin(x)*cos(x/2)}
6 \endpsgraph

```



```

1 \psset{lly=-0.5cm}
2 \psgraph(0,-5)(18,3){0.9\linewidth}{5cm}
3 \psset{algebraic,plotpoints=501}
4 \psplot[linecolor=yellow,linewidth=4\pslinewidth]{0.01}{18}{ln(x)}
5 \psplot[linecolor=red]{0.01}{18}{ln(x)}
6 \psplot[linecolor=yellow,linewidth=4\pslinewidth]{0}{18}{3*cos(x)*2.71^(-x
  /10)}
7 \psplot[linecolor=blue,showpoints=true,plotpoints=51]{0}{18}{3*cos(x)*2.71^(-
  x/10)}
8 \endpsgraph

```

4. Numeric functions

All macros have a @ in their name, because they are only for internal use, but it is no problem to use them like other macros. One can define another name without a @:

```

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

```

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

5. Numeric functions

By default PSTricks loads the file pst-fp which is derived from the fp package. It supports the following macros:

5.1. \pstFPadd, \pstFPsub, \pstFPMul, and \pstFPdiv

Multiplication and division:

```

\pstFPadd{result}{number}{number}
\pstFPsub{result}{number}{number}
\pstFPMul{result}{number}{number}
\pstFPdiv{result}{number}{number}

```

-0.079847250000000000	1	<code>\pstFPMul\Result{-3.405}{0.02345} \Result\quad</code>
-145.202558635394456289	2	<code>\pstFPdiv\Result{-3.405}{0.02345} \Result\\</code>
-0.079847250000000000	3	<code>\pstFPMul\Result{0.02345}{-3.405} \Result\quad</code>
-0.006886930983847283	4	<code>\pstFPdiv\Result{0.02345}{-3.405} \Result\\</code>
7726.059000000000000000	5	<code>\pstFPMul\Result{234.123}{33} \Result\quad</code>
7.0946363636363636	6	<code>\pstFPdiv\Result{234.123}{33} \Result\\</code>
267.123000000000000000	7	<code>\pstFPadd\Result{234.123}{33} \Result\quad</code>
201.123000000000000000	8	<code>\pstFPadd\Result{234.123}{-33} \Result\\</code>
201.123000000000000000	9	<code>\pstFPsub\Result{234.123}{33} \Result\quad</code>
-267.123000000000000000	10	<code>\pstFPsub\Result{-234.123}{33} \Result</code>

The zeros can be stripped with the macro `\pstFPstripZeros`. Expect always rounding errors, \TeX was not made for calculations ...

	1	<code>\pstFPMul\Result{-3.405}{0.02345}</code>
	2	<code>\pstFPstripZeros{\Result}{\Result}\Result\quad</code>
	3	<code>\pstFPdiv\Result{-3.405}{0.02345}</code>
	4	<code>\pstFPstripZeros{\Result}{\Result}\Result\\</code>
-0.07985 -145.20256	5	<code>\pstFPMul\Result{0.02345}{-3.405}</code>
-0.07985 -0.00688	6	<code>\pstFPstripZeros{\Result}{\Result}\Result\quad</code>
	7	<code>\pstFPdiv\Result{0.02345}{-3.405}</code>
	8	<code>\pstFPstripZeros{\Result}{\Result}\Result</code>

5.2. `\pstFPMul` and `\pstFPDiv`

Integer multiplication and division:

<code>\pstFPMul{result as a truncated integer}{number}{number}</code>
<code>\pstFPDiv{result as a truncated integer}{number}{number}</code>

-0 -145	1	<code>\makeatletter</code>
-79 -6	2	<code>\pstFPMul\Result{-34.05}{0.02345} \Result\quad</code>
7726 7	3	<code>\pstFPDiv\Result{-3.405}{0.02345} \Result\\</code>
	4	<code>\pstFPMul\Result{23.45}{-3.405} \Result\quad</code>
	5	<code>\pstFPDiv\Result{0.2345}{-0.03405} \Result\\</code>
	6	<code>\pstFPMul\Result{234.123}{33} \Result\quad</code>
	7	<code>\pstFPDiv\Result{234.123}{33} \Result</code>
	8	<code>\makeatother</code>

6. The PostScript header files

6.1. `pstricks.pro`

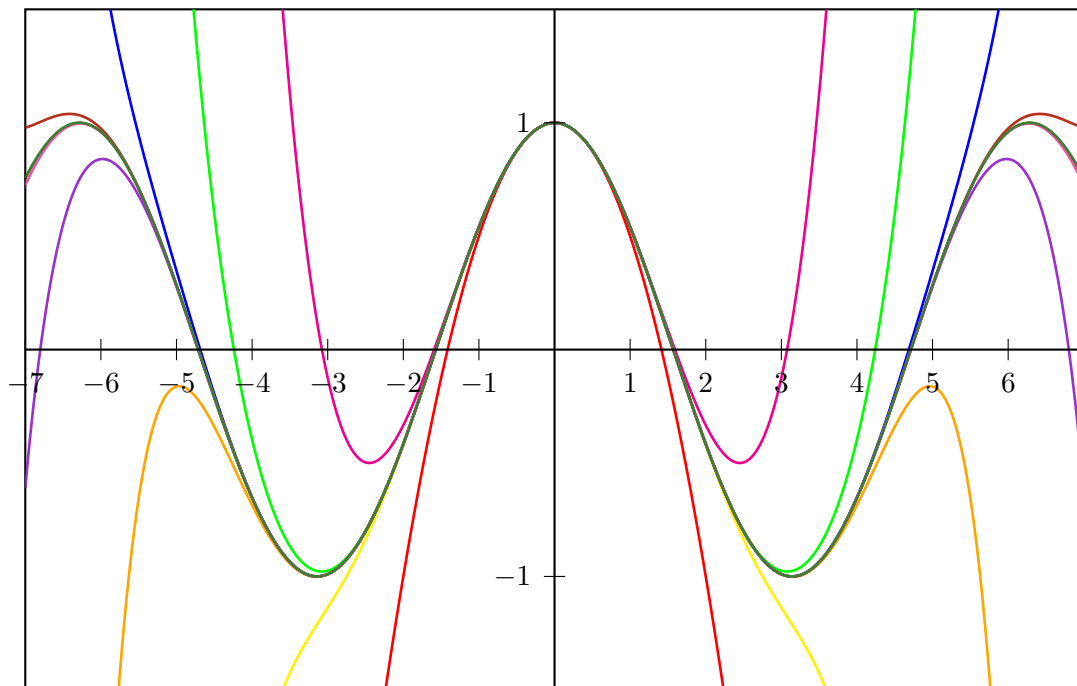
It contains now most of the stuff from `pstricks-add` and the new routines for plotting lines/curves with symbols.

6.2. pst-algparser.pro

Using the Sum function

`\Sum(<index name>,<start>,<step>,<end>,<function>)`

Let's plot the first development of cosine with polynomials: $\sum_{n=0}^{+\infty} \frac{(-1)^n x^{2n}}{n!}$.



```

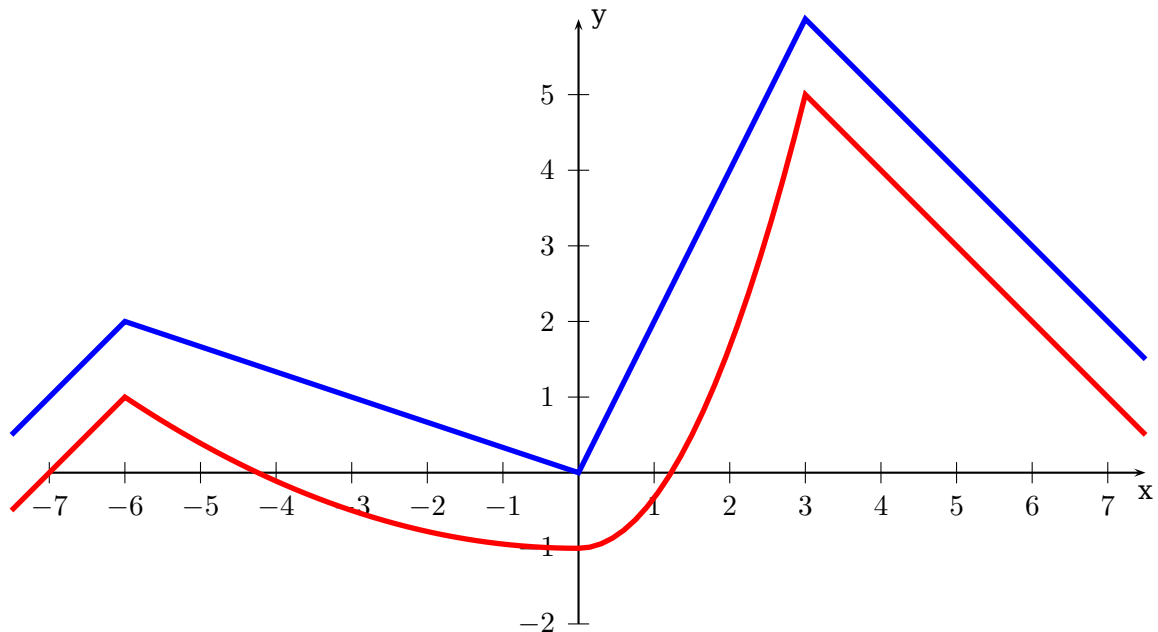
1 \psset{algebraic=true, plotpoints=501, yunit=3}
2 \def\getColor#1{\ifcase#1 black\or red\or magenta\or yellow\or green\or Orange\
  or blue\or
3   DarkOrchid\or BrickRed\or Rhodamine\or OliveGreen\fi}
4 \begin{pspicture}(-7,-1.5)(7,1.5)
5   \psclip{\psframe(-7,-1.5)(7,1.5)}
6   \psplot{-7}{7}{cos(x)}
7   \multido{\n=1+1}{10}{%
8     \psplot[linewidth=1pt, linecolor=\getColor{\n}]{-7}{7}{%
9       Sum(ijk,0,1,\n,(-1)^ijk*x^(2*ijk)/fact(2*ijk))}}
10  \endpsclip
11  \psaxes(0,0)(-7,-1.5)(7,1.5)
12 \end{pspicture}

```

6.3. The variable step algorithm together with the PostScript function IfTE

$\text{IfTE}(\langle \text{condition} \rangle, \langle \text{true part} \rangle, \langle \text{false part} \rangle)$

Nesting of several IfTE is possible and seen in the following examples. A classic example is a piece-wise linear function.

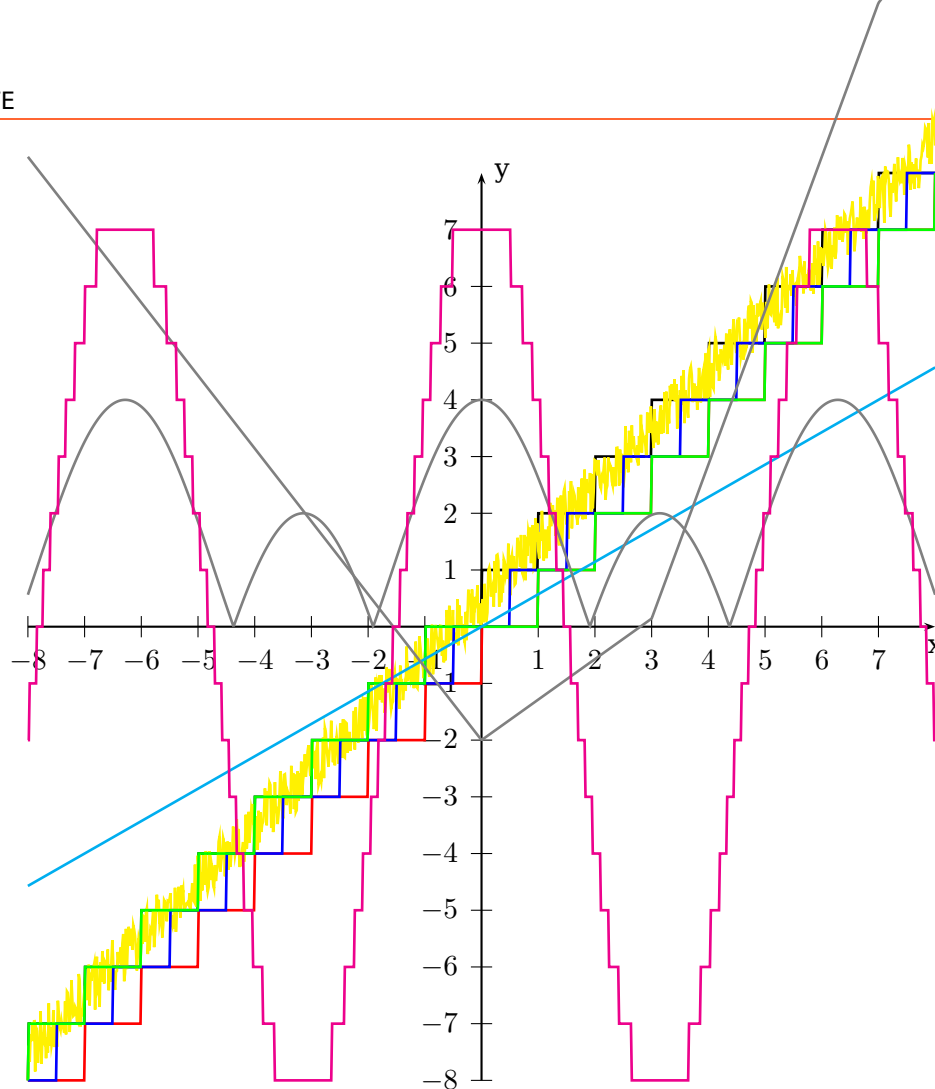


```

1 \psset{unit=1.5, algebraic, VarStep, showpoints, VarStepEpsilon=.001}
2 \begin{pspicture}[showgrid=true](-7,-2)(2,4)
3   \psplot{-7}{2}{IfTE(x<-5,-(x+5)^3/2,IfTE(x<0,0,x^2))}
4   \psplot{-7}{2}{5*x/9+26/9}
5   \psplot[linecolor=blue]{-7}{2}{(x+7)^30/9^30*4.5-1/2}
6   \psplot[linecolor=red]{-6.9}{2}
7     {IfTE(x<-6,ln(x+7),IfTE(x<-3,x+6,IfTE(x<0.1415926,sin(x+3)+3,3.1415926-x))
8     )}
9 \end{pspicture}

```

When you program a piece-wise defined function you must take care that a plotting point must be put at each point where the description changes. Use `showpoints=true` to see what's going on when there is a problem. You are on the safe side when you choose a big number for `plotpoints`.

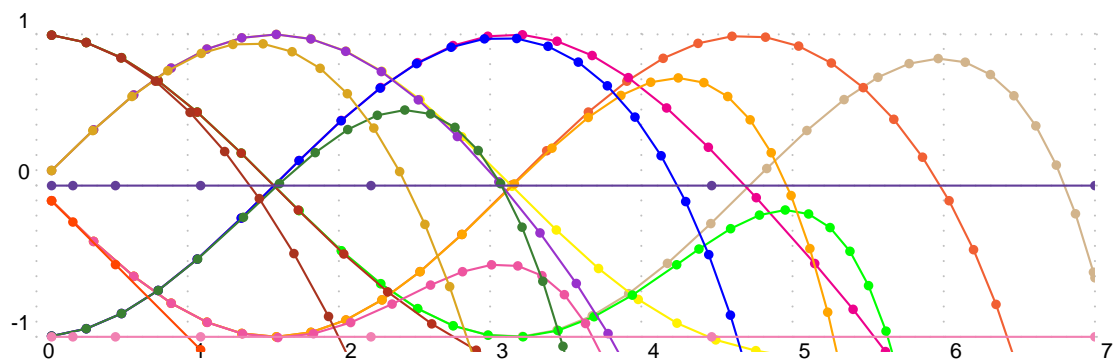


```

1 \psset{unit=0.75}
2 \begin{pspicture}(-8,-8)(8,8)
3   \psaxes{>}(0,0)(-8,-8)(8,8)[x,-90][y,0]
4   \psset{plotpoints=1000,linewidth=1pt}
5   \psplot[algebraic=true, linecolor=yellow]{-8}{8}{rand/(2^31-1)+x}
6   \psplot[algebraic=true]{-8}{8}{ceiling(x)}
7   \psplot[algebraic=true, linecolor=red]{-8}{8}{floor(x)}
8   \psplot[algebraic=true, linecolor=blue]{-8}{8}{round(x)}
9   \psplot[algebraic=true, linecolor=green]{-8}{8}{truncate(x)}
10  \psplot[algebraic=true, linecolor=cyan]{-8}{8}{div(mul(4,x),7)}
11  \psplot[algebraic=true, linecolor=gray]{-8}{8}{abs(x)+abs(x-3)-abs(5-5*x/7)}
12  \psplot[algebraic=true, linecolor=gray]{-8}{8}{abs(3*cos(x)+1)}
13  \psplot[algebraic=true, linecolor=magenta]{-8}{8}{floor(8*cos(x))}
14 \end{pspicture}

```

6.4. Successive derivatives of a polynomial with the PostScript function Derive



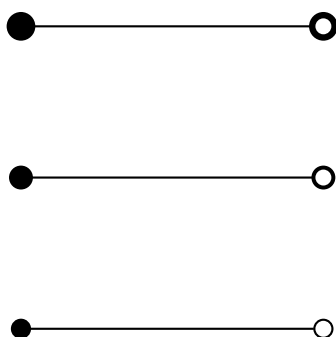
```

1 \psset{unit=2, algebraic=true, VarStep=true, showpoints=true, VarStepEpsilon
   =.001}
2 \def\getColor#1{\ifcase#1 Tan\or RedOrange\or magenta\or yellow\or green\or
   Orange\or blue\or
3   DarkOrchid\or BrickRed\or Rhodamine\or OliveGreen\or Goldenrod\or Mahogany\or
4   OrangeRed\or CarnationPink\or RoyalPurple\or Lavender\fi}
5 \begin{pspicture}[showgrid=true](0,-1.2)(7,1.5)
6   \psclip{\psframe[linestyle=none](0,-1.1)(7,1.1)}
7   \multido{\in=0+1}{16}{%
8     \psplot[algebraic=true, linecolor=\getColor{\in}]{0.1}{7}
9     {Derive(\in,Sum(i,0,1,7,(-1)^i*x^(2*i)/Fact(2*i)))}}
10  \endpsclip
11 \end{pspicture}

```

6.5. Special arrow option arrowLW

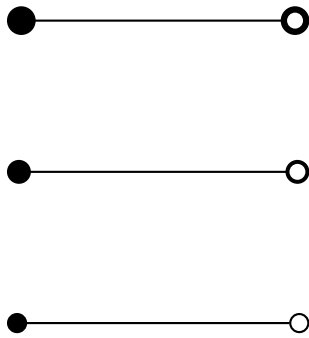
Only for the arrowtype o, oo, *, and ** it is possible to set the arrowlinewidth with the optional keyword arrowLW. When scaling an arrow by the keyword arrowScale the width of the borderline is also scaled. With the optional argument arrowLW the line width can be set separately and is not taken into account by the scaling value.



```

1 \begin{pspicture}(4,6)
2   \psline[arrowscale=3,arrows=-o](0,5)(4,5)
3   \psline[arrowscale=3,arrows=-o,
4     arrowLW=0.5pt](0,3)(4,3)
5   \psline[arrowscale=3,arrows=-o,
6     arrowLW=0.3333\pslinewidth](0,1)(4,1)
7 \end{pspicture}

```



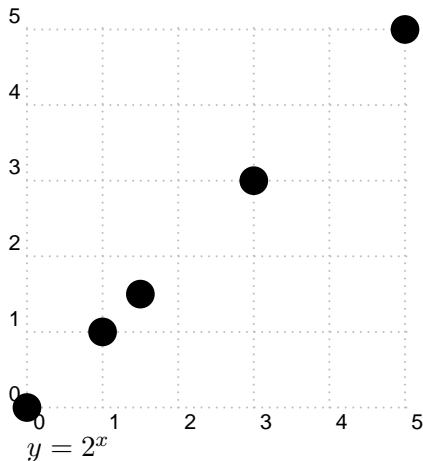
```
1 \begin{pspicture}(4,6)
2 \psline[arrowscale=3,arrows=**-oo](0,5)(4,5)
3 \psline[arrowscale=3,arrows=**-oo,
4   arrowLW=0.5pt](0,3)(4,3)
5 \psline[arrowscale=3,arrows=**-oo,
6   arrowLW=0.3333\pslinewidth](0,1)(4,1)
7 \end{pspicture}
```

7. \psforeach and \psForeach

The macro \psforeach allows a loop with an individual increment.

```
\psforeach{variable}{value list}{action}
\psForeach{variable}{value list}{action}
```

With \psforeach the *action* is done inside a group and for \psForeach not. This maybe useful when using the macro to create tabular cells, which are already grouped itself.

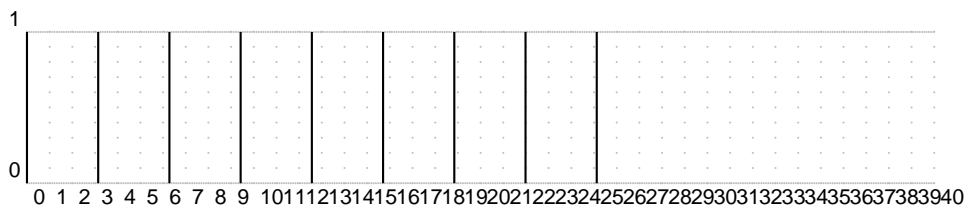


2	4	6	8	10	12
4.0	16.0	64.0	256.0	1024.0	4096.0

```
1 \begin{pspicture}[showgrid=true
   ](5,5)
2   \psforeach{\nA}{0, 1, 1.5, 3, 5}{%
3     \psdot[dotscale=3](\nA,\nA)}
4 \end{pspicture}
```

```
1 %\usepackage{pst-func}
2 \makeatletter
3 \newcommand*\InitToks{\toks@={}}
4 \newcommand\AddToks[1]{\toks@=\expandafter{\the\toks@ #1}}
5 \newcommand*\PrintToks{\the\toks@}
6 \newcommand*\makeTable}[4][5mm]{%
7   \begingroup
8     \InitToks%
9     \AddToks{\begin{tabular}{*{#2}>{\RaggedLeft}p{#1}}|@{}l@{}}\cline{1-#2}}
10    \psForeach{\iA}{#3}{\expandafter\AddToks\expandafter{\iA & }}
11    \AddToks{\\ \cline{1-#2}}%
12    \psForeach{\iA}{#3}{\expandafter\AddToks\expandafter{\expandafter%
13      \psPrintValue\expandafter{\iA\space /x ED #4} & }}
14    \AddToks{\\ \cline{1-#2}\end{tabular}}%
15    \PrintToks
16  \endgroup
17 }
18 \makeatother
19
20 \sffamily
21 \psset{decimals=2,valuewidth=7,xShift=-20}
22 $y=2^x$\\
23 \makeTable[1cm]{6}{2,4,6,8,10,12}{2 x exp}
```

The value List can also be given by the first two and the last value, e.g. 1,4,...,31, then PSTricks calculates all values with the distance given by the first two values.

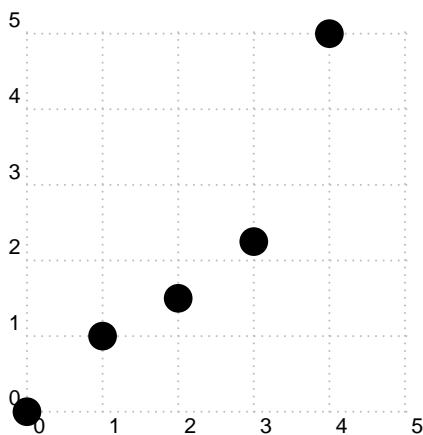


```

1 \psset{xunit=0.3cm,yunit=2cm}
2 \begin{pspicture}[showgrid](0,-5mm)(40,1)
3   \psforeach{\nA}{0, 3.14,...,25}{\psline(\nA,0)(\nA,1)}
4 \end{pspicture}

```

The internal counter for the steps is named `psLoopIndex` and can be used for own purposes.



```

1 \begin{pspicture}[showgrid=true]
2   (5,5)
3   \psforeach{\nA}{0, 1, 1.5, 2.25, 5}{
4     \psdot[dotsize=3](\the\psLoopIndex,\nA)}
5 \end{pspicture}

```

Part II.

pst-node – package

8. pst-node.tex

The package `pst-node` now uses the advanced key handling from `xkeyval`. The reason why it moved from the base into the contrib sections, where all packages uses `xkeyval`.

Part III.

pst-plot – package

9. pst-plot.tex

The package `pst-plot` now uses the advanced key handling from `xkeyval`. The reason why it moved from the base into the contrib sections, where all packages uses `xkeyval`.

Index

Symbole

`*`, 10

`**`, 10

A

`\AddToShipoutPicture`, 3

algebraic, 3, 4

`arrowLW`, 10

`arrowscale`, 10

`auto-pst-pdf`, 3

C

Class

– `pst-doc`, 3

– `scrartcl`, 3

E

`eso-pic`, 3

F

File

– `pst-doc-pdf`, 3

– `pst-fp`, 5

`fp`, 5

I

`IfTE`, 8

K

Keyword

– algebraic, 3, 4

– `arrowLW`, 10

– `arrowscale`, 10

– `plotpoints`, 8

– `showpoints`, 8

– `VarStep`, 8

– `VarStepEpsilon`, 8

L

`latex`, 3

M

Macro

– `\AddToShipoutPicture`, 3

– `\makeatletter`, 5

– `\makeatother`, 5

– `\parametricplot`, 4

– `\psForeach`, 12

– `\psforeach`, 12

– `\psplot`, 3

– `\pstFPAdd`, 5

– `\pstFPDiv`, 6

– `\pstFPdiv`, 5

– `\pstFPMul`, 6

– `\pstFPMul`, 5

– `\pstFPstripZeros`, 6

– `\pstFPsub`, 5

– `\Sum`, 7

`\makeatletter`, 5

`\makeatother`, 5

O

`o`, 10

`oo`, 10

P

Package

– `auto-pst-pdf`, 3

– `eso-pic`, 3

– `fp`, 5

– `pst-node`, 13

– `pst-pdf`, 3

– `pst-plot`, 13

– `pstricks`, 3

– `pstricks-add`, 6

– `xkeyval`, 13

`\parametricplot`, 4

`pdflatex`, 3

`plotpoints`, 8

PostScript

– `IfTE`, 8

Program

– `latex`, 3

– `pdflatex`, 3

– `pst2pdf`, 3

`\psForeach`, 12

`\psforeach`, 12

`psLoopIndex`, 13

\psplot, 3
pst-doc, 3
pst-doc-pdf, 3
pst-fp, 5
pst-node, 13
pst-pdf, 3
pst-plot, 13
pst2pdf, 3
\pstFPadd, 5
\pstFPDiv, 6
\pstFPdiv, 5
\pstFPMul, 6
\pstFPMul, 5
\pstFPstripZeros, 6
\pstFPsub, 5
pstricks, 3
pstricks-add, 6

S

scrartcl, 3
showpoints, 8
\Sum, 7

Syntax

– *, 10
– **, 10
– o, 10
– oo, 10

V

VarStep, 8
VarStepEpsilon, 8

X

xkeyval, 13