pst-func

Plotting special mathematical functions; v.0.95

Herbert Voß

June 6, 2020

Contents

`pst-func` loads by default the following packages: `pst-plot`, `pstricks-add`, `pst-math`, `pst-xkey`, and, of course `pstricks`. All should be already part of your local TEX installation. If not, or in case of having older versions, go to http://www.CTAN.org/ and load the newest version.

Thanks to
Rafal Bartczuk, Jean-Côme Charpentier, Martin Chicoine, Gerry Coombes, Denis Girou, John Frampton, Leon Free, Attila Gati, Horst Gierhardt, Jürgen Gilg, Christophe Jorssen, Lars Kotthoff, Buddy Ledger, Manuel Luque, Patrice Mégret, Svend Mortensen, Matthias Rüss, Thomas Söll, Jose-Emilio Vila-Forcen, Timothy Van Zandt, Michael Zedler, and last but not least http://mathworld.wolfram.com.

1 \psBezier#

This macro can plot a Bézier spline from order 1 up to 9 which needs (order+1) pairs of given coordinates.

Given a set of $n+1$ control points P_0, P_1, ..., P_n, the corresponding Bézier curve (or Bernstein-Bézier curve) is given by

$$C(t) = \sum_{i=0}^{n} P_i B_{i,n}(t) \tag{1}$$

where $B_{i,n}(t)$ is a Bernstein polynomial $B_{i,n}(t) = \binom{n}{i} t^i (1-t)^{n-i}$, and $t \in [0,1]$. The Bézier curve starts through the first and last given point and lies within the convex hull of all control points. The curve is tangent to $P_1 - P_0$ and $P_n - P_{n-1}$ at the endpoint. Undesirable properties of Bézier curves are their numerical instability for large numbers of control points, and the fact that moving a single control point changes the global shape of the curve. The former is sometimes avoided by smoothly patching together low-order Bézier curves.

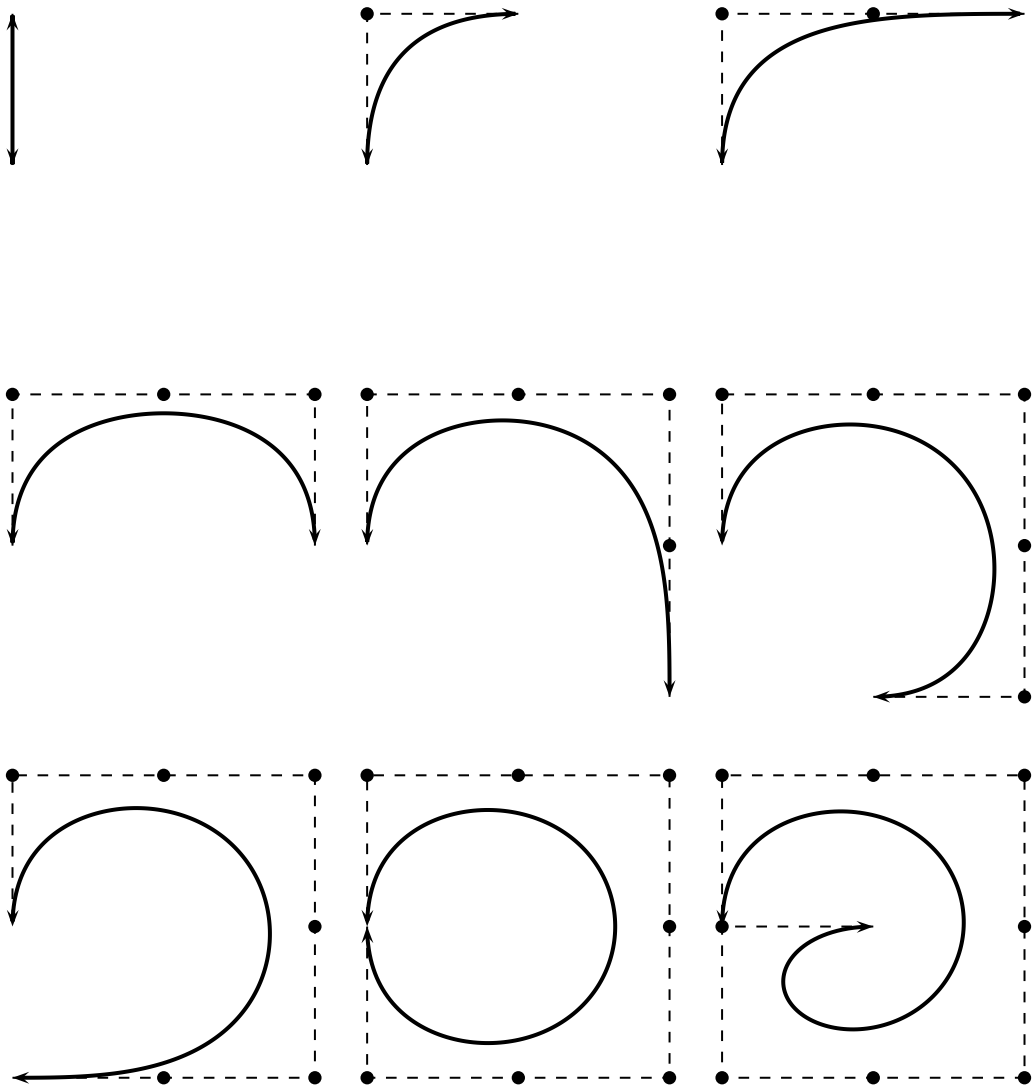
The macro `\psBezier` (note the upper case B) expects the number of the order and $n = \text{order} + 1$ pairs of coordinates:

`\psBezier#` `[Options]` $(x_0,y_0)(x_1,y_1)(x_n,y_n)$

The number of steps between the first and last control points is given by the keyword `plotpoints` and preset to 200. It can be changed in the usual way.

```
\psset{showpoints=true,linewidth=1.5pt}
\begin{pspicture}(-2,-2)(2,2)% order 1 -- linear
  \psBezier1{<->}(-2,0)(-2,2)
\end{pspicture}\qquad
%
\begin{pspicture}(-2,-2)(2,2)% order 2 -- quadratric
  \psBezier2{<->}(-2,0)(-2,2)(0,2)
\end{pspicture}\qquad
%
\begin{pspicture}(-2,-2)(2,2)% order 3 -- cubic
  \psBezier3{<->}(-2,0)(-2,2)(0,2)(2,2)
\end{pspicture}\qquad

\vspace{1cm}
\begin{pspicture}(-2,-2)(2,2)% order 4 -- quartic
  \psBezier4{<->}(-2,0)(-2,2)(0,2)(2,2)(2,0)
\end{pspicture}\qquad
%
\begin{pspicture}(-2,-2)(2,2)% order 5 -- quintic
  \psBezier5{<->}(-2,0)(-2,2)(0,2)(2,2)(2,0)(2,-2)
\end{pspicture}\qquad
%
\begin{pspicture}(-2,-2)(2,2)% order 6
  \psBezier6{<->}(-2,0)(-2,2)(0,2)(2,2)(2,0)(2,-2)(0,-2)
\end{pspicture}\qquad

\vspace{1cm}
\begin{pspicture}(-2,-2)(2,2)% order 7
  \psBezier7{<->}(-2,0)(-2,2)(0,2)(2,2)(2,0)(2,-2)(0,-2)(-2,-2)
\end{pspicture}\qquad
%
\begin{pspicture}(-2,-2)(2,2)% order 8
```

```
  \psBezier8{<->}(-2,0)(-2,2)(0,2)(2,2)(2,0)(2,-2)(0,-2)(-2,-2)(-2,0)
\end{pspicture}\qquad
%
\begin{pspicture}(-2,-2)(2,2)% order 9
  \psBezier9{<->}(-2,0)(-2,2)(0,2)(2,2)(2,0)(2,-2)(0,-2)(-2,-2)(-2,0)(0,0)
\end{pspicture}
```



2 Polynomials

2.1 Chebyshev polynomials

The polynomials of the first (`ChebyshevT`) kind are defined through the identity

$$T_n(\cos\theta) = \cos(n\theta)$$

They can be obtained from the generating functions

$$g_1(t,x) = \frac{1-t^2}{1-2xt+t^2} \tag{2}$$

$$= T_0(x) + 2\sum_{n=1}^{\infty} T_n(x)t^n \tag{3}$$

and

$$g_2(t,x) = \frac{1-xt}{1-2xt+t^2} \tag{4}$$

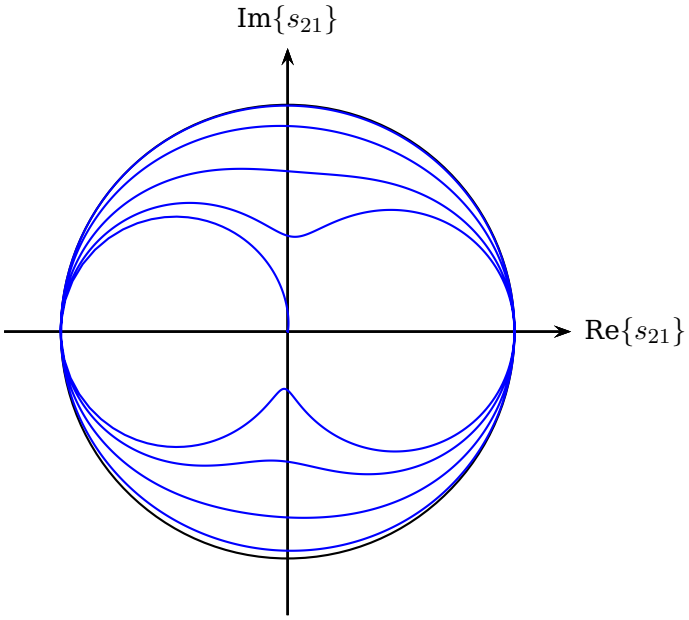
$$= \sum_{n=0}^{\infty} T_n(x)t^n \tag{5}$$

The polynomials of second kind (`ChebyshevU`) can be generated by

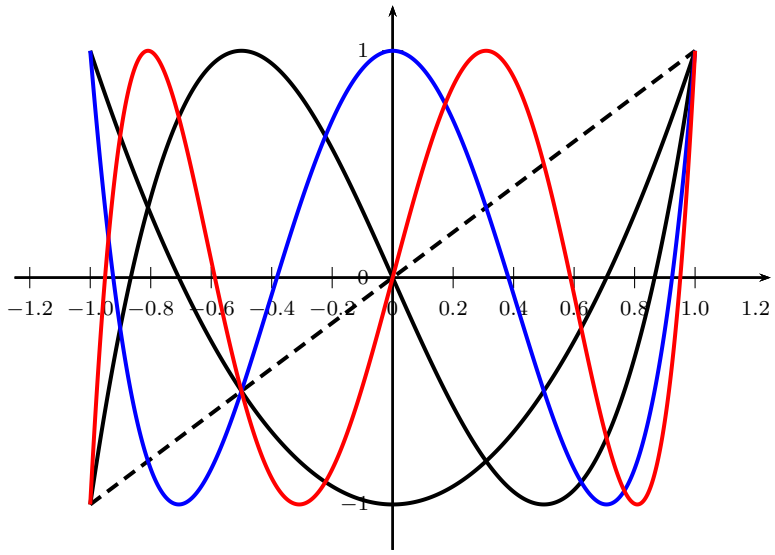
$$g(t,x) = \frac{1}{1-2xt+t^2} \tag{6}$$

$$= \sum_{n=0}^{\infty} U_n(x)t^n \tag{7}$$

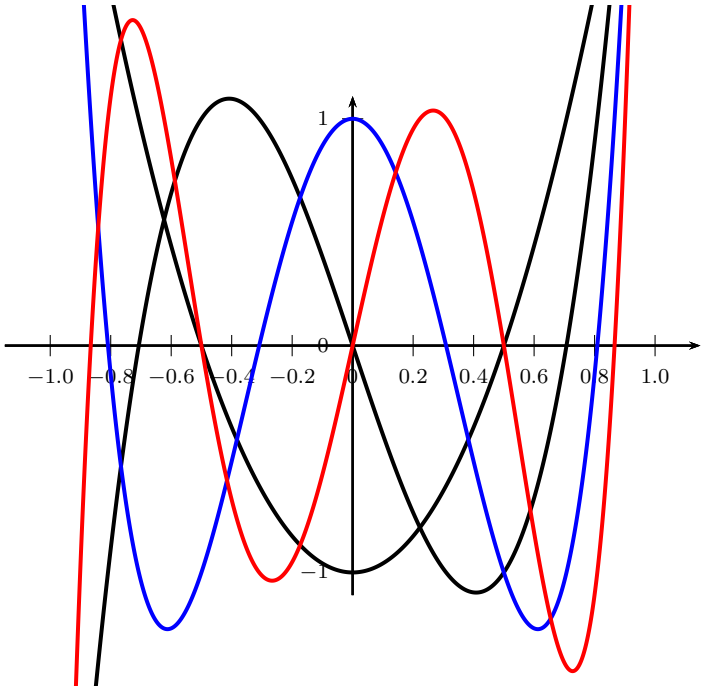
`pst-func` defines the TEX-macros `\ChebyshevT` for the first kind and `\ChebyshevU` for the second kind of Chebyshev polynomials. These TEX-macros cannot be used outside of PostScript, they are only wrappers for `tx@FuncDict begin ChebyshevT end` and the same for `\ChebyshevU`.



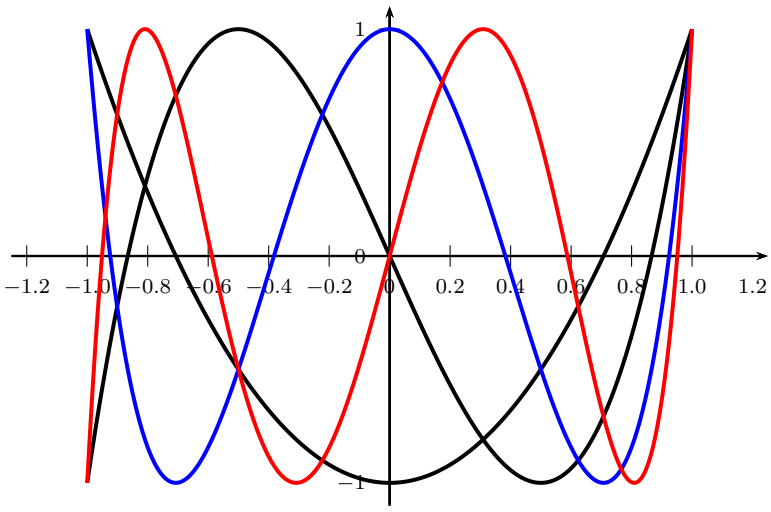

```
\psset{arrowscale=1.5,unit=3cm}
\begin{pspicture}(-1.5,-1.5)(1.5,1.5)
  \psaxes[ticks=none,labels=none]{->}(0,0)(-1.25,-1.25)(1.25,1.25)%
    [Re$\{s_{21}\}$,0][Im$\{s_{21}\}$,90]
  \pscircle(0,0){1}
  \parametricplot[linecolor=blue,plotpoints=10000]{0}{1.5}{
    /N 9 def
    /x 2 N mul t \ChebyshevT def
    /y 2 N mul 1 sub t \ChebyshevU def
    x x 2 exp y 2 exp add div
    y x 2 exp y 2 exp add div
  }
\end{pspicture}
```




```
\psset{xunit=4cm,yunit=3cm,plotpoints=1000}
\begin{pspicture}(-1.2,-2)(2,1.5)
  \psaxes[Dx=0.2]{->}(0,0)(-1.25,-1.2)(1.25,1.2)
  \psset{linewidth=1.5pt}
  \psplot[linestyle=dashed]{-1}{1}{1 x \ChebyshevT}
  \psplot[linecolor=black]{-1}{1}{2 x \ChebyshevT}
  \psplot[linecolor=black]{-1}{1}{3 x \ChebyshevT}
  \psplot[linecolor=blue]{-1}{1}{4 x \ChebyshevT }
  \psplot[linecolor=red]{-1}{1}{5 x \ChebyshevT }
\end{pspicture}
```




```
\psset{xunit=4cm,yunit=3cm,plotpoints=1000}
\begin{pspicture*}(-1.5,-1.5)(1.5,1.5)
  \psaxes[Dx=0.2]{->}(0,0)(-1.15,-1.1)(1.15,1.1)
  \psaxes[Dx=0.2]{->}(0,0)(-1.25,-1.2)(1.25,1.2)
  \psset{linewidth=1.5pt}
  \psplot[linecolor=black]{-1}{1}{2 x \ChebyshevU}
  \psplot[linecolor=black]{-1}{1}{3 x \ChebyshevU}
  \psplot[linecolor=blue]{-1}{1}{4 x \ChebyshevU}
  \psplot[linecolor=red]{-1}{1}{5 x \ChebyshevU}
\end{pspicture*}
```




```
\psset{xunit=4cm,yunit=3cm,plotpoints=1000}
\begin{pspicture}(-1.25,-1.2)(1.25,1.2)
  \psaxes[Dx=0.2]{->}(0,0)(-1.25,-1.2)(1.25,1.2)
  \psset{linewidth=1.5pt}
```

```
  \psplot[linecolor=black]{-1}{1}{x ACOS 2 mul RadtoDeg cos}
  \psplot[linecolor=black]{-1}{1}{x ACOS 3 mul RadtoDeg cos}
  \psplot[linecolor=blue]{-1}{1}{x ACOS 4 mul RadtoDeg cos}
  \psplot[linecolor=red]{-1}{1}{x ACOS 5 mul RadtoDeg cos}
\end{pspicture}
```

2.2 \psPolynomial

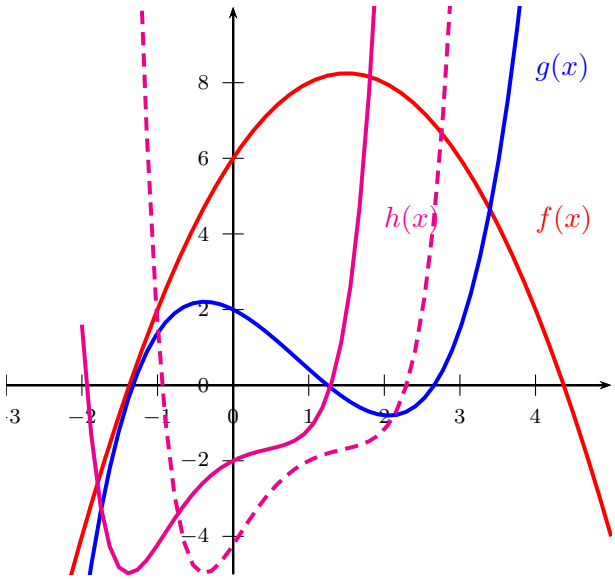
The polynomial function is defined as

$$f(x) = a_0 + a_1x + a_2x^2 + a_3x^3 + \ldots + a_{n-1}x^{n-1} + a_nx^n \tag{8}$$
$$f'(x) = a_1 + 2a_2x + 3a_3x^2 + \ldots + (n-1)a_{n-1}x^{n-2} + na_nx^{n-1} \tag{9}$$
$$f''(x) = 2a_2 + 6a_3x + \ldots + (n-1)(n-2)a_{n-1}x^{n-3} + n(n-1)a_nx^{n-2} \tag{10}$$

so `pst-func` needs only the coefficients of the polynomial to calculate the function. The syntax is

`\psPolynomial [Options] {xStart}{xEnd}`

With the option `xShift` one can do a horizontal shift to the graph of the function. With another than the predefined value the macro replaces x by $x - x\mathrm{Shift}$; `xShift=1` moves the graph of the polynomial function one unit to the right.




```
\psset{yunit=0.5cm,xunit=1cm}
\begin{pspicture*}(-3,-5)(5,10)
  \psaxes[Dy=2]{->}(0,0)(-3,-5)(5,10)
  \psset{linewidth=1.5pt}
  \psPolynomial[coeff=6 3 -1,linecolor=red]{-3}{5}
  \psPolynomial[coeff=2 -1 -1 .5 -.1 .025,linecolor=blue]{-2}{4}
  \psPolynomial[coeff=-2 1 -1 .5 .1 .025 .2 ,linecolor=magenta]{-2}{4}
  \psPolynomial[coeff=-2 1 -1 .5 .1 .025 .2 ,linecolor=magenta,xShift=1,linestyle=dashed]{-2}{4}
  \rput[lb](4,4){\textcolor{red}{$f(x)$}}
  \rput[lb](4,8){\textcolor{blue}{$g(x)$}}
  \rput[lb](2,4){\textcolor{magenta}{$h(x)$}}
\end{pspicture*}
```

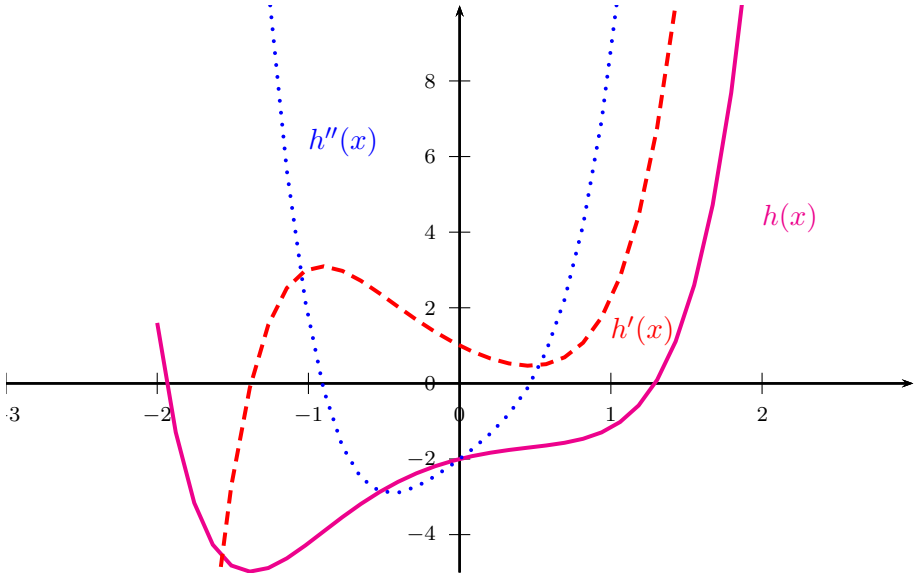
The plot is easily clipped using the star version of the `pspicture` environment, so that points whose coordinates are outside of the desired range are not plotted. The plotted polynomials are:

$$f(x) = 6 + 3x - x^2 \tag{11}$$
$$g(x) = 2 - x - x^2 + 0.5x^3 - 0.1x^4 + 0.025x^5 \tag{12}$$
$$h(x) = -2 + x - x^2 + 0.5x^3 + 0.1x^4 + 0.025x^5 + 0.2x^6 \tag{13}$$
$$h^*(x) = -2 + (x-1) - (x-1)^2 + 0.5(x-1)^3 + \\ + 0.1(x-1)^4 + 0.025(x-1)^5 + 0.2(x-1)^6 \tag{14}$$

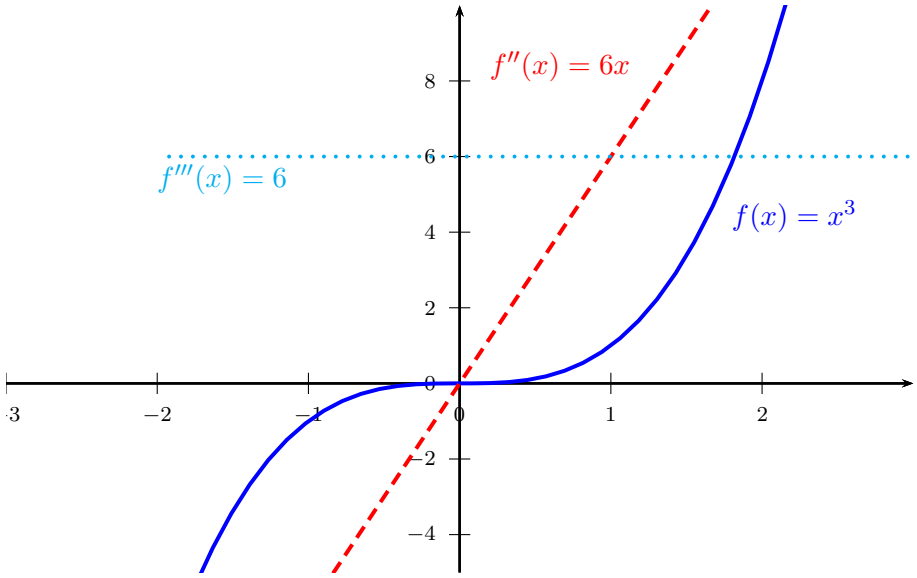
There are the following new options:

Name	Value	Default	
`coeff`	`a0 a1 a2 ...`	`0 0 1`	The coefficients must have the order $a_0\ a_1\ a_2 \ldots$ and be separated by **spaces**. The number of coefficients is limited only by the memory of the computer ... The default value of the parameter `coeff` is `0 0 1`, which gives the parabola $y = a_0 + a_1x + a_2x^2 = x^2$.
`xShift`	`<number>`	`0`	$(x - xShift)$ for the horizontal shift of the polynomial
`Derivation`	`<number>`	`0`	the default is the function itself
`markZeros`	`false|true`	`false`	dotstyle can be changed
`epsZero`	`<value>`	`0.1`	The distance between two zeros, important for the iteration function to test, if the zero value still exists
`dZero`	`<value>`	`0.1`	When searching for all zero values, the function is scanned with this step
`zeroLineTo`	`<number>`	`false`	plots a line from the zero point to the value of the zeroLineTo's Derivation of the polynomial function
`zeroLineStyle`	`<line style>`	`dashed`	the style is one of the for `PSTricks`valid styles.
`zeroLineColor`	`<color>`	`black`	any valid xolor is possible
`zeroLineWidth`	`<width>`	`0.5\pslinewidth`	

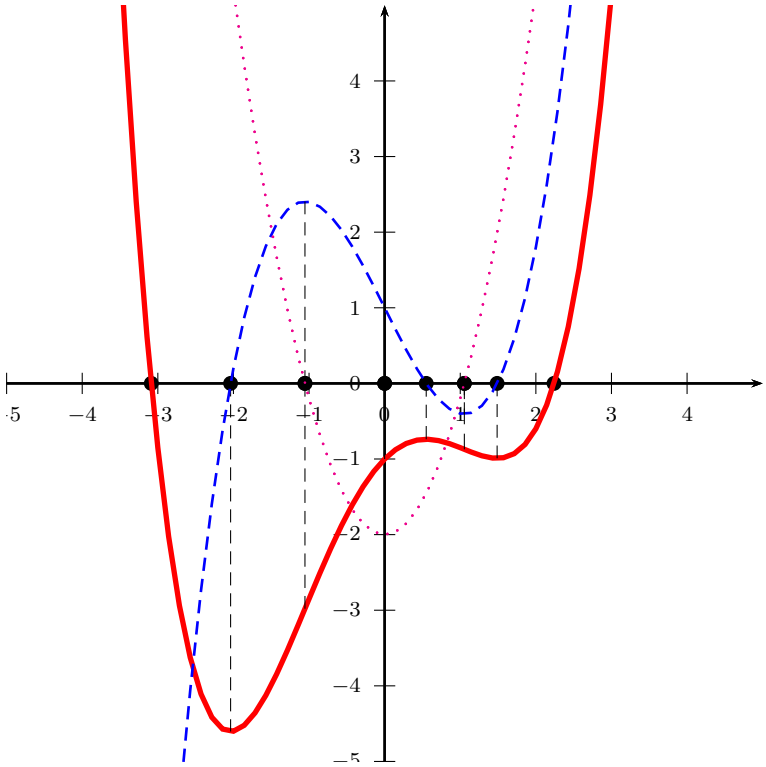
The above parameters are only valid for the `\psPolynomial` macro, except `x0`, which can also be used for the Gauss function. All options can be set in the usual way with `\psset`.



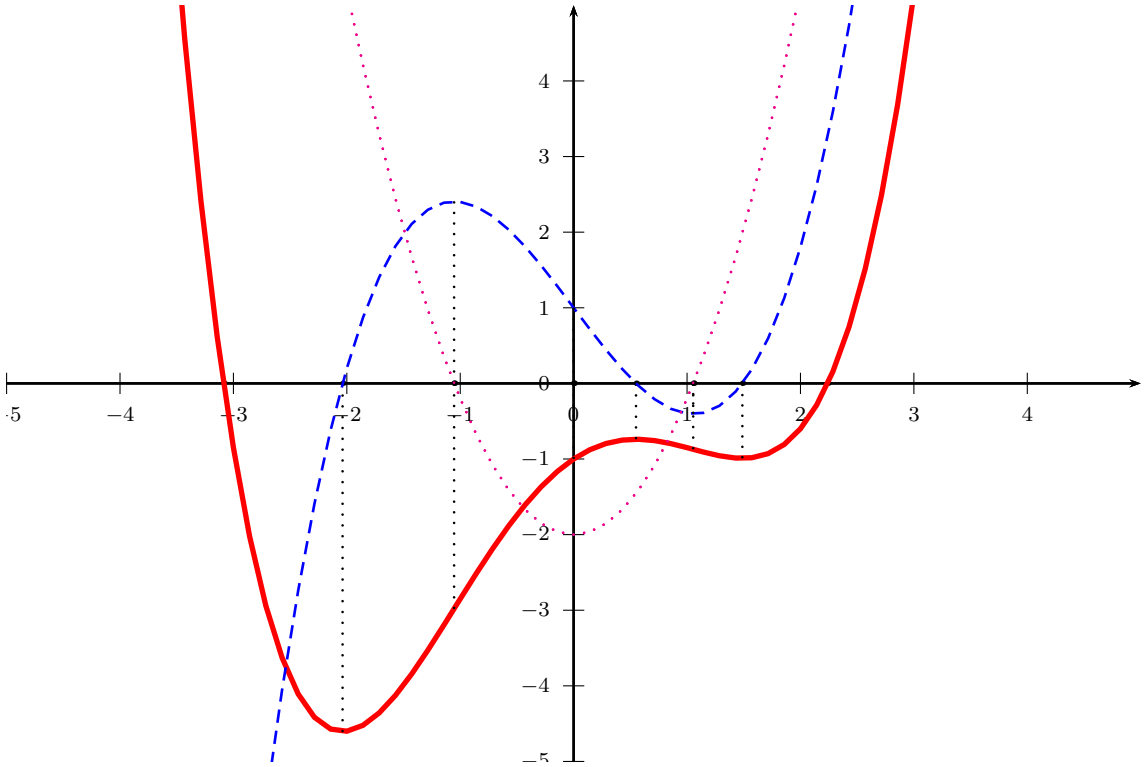

```
\psset{yunit=0.5cm,xunit=2cm}
\begin{pspicture*}(-3,-5)(3,10)
  \psaxes[Dy=2]{->}(0,0)(-3,-5)(3,10)
  \psset{linewidth=1.5pt}
  \psPolynomial[coeff=-2 1 -1 .5 .1 .025 .2 ,linecolor=magenta]{-2}{4}
  \psPolynomial[coeff=-2 1 -1 .5 .1 .025 .2 ,linecolor=red,%
    linestyle=dashed,Derivation=1]{-2}{4}
  \psPolynomial[coeff=-2 1 -1 .5 .1 .025 .2 ,linecolor=blue,%
    linestyle=dotted,Derivation=2]{-2}{4}
  \rput[lb](2,4){\textcolor{magenta}{$h(x)$}}
  \rput[lb](1,1){\textcolor{red}{$h^{\prime}(x)$}}
  \rput[lb](-1,6){\textcolor{blue}{$h^{\prime\prime}(x)$}}
\end{pspicture*}
```




```
\psset{yunit=0.5cm,xunit=2cm}
\begin{pspicture*}(-3,-5)(3,10)
  \psaxes[Dy=2]{->}(0,0)(-3,-5)(3,10)
  \psset{linewidth=1.5pt}
  \psPolynomial[coeff=0 0 0 1,linecolor=blue]{-2}{4}
  \psPolynomial[coeff=0 0 0 1,linecolor=red,%
    linestyle=dashed,Derivation=2]{-2}{4}
  \psPolynomial[coeff=0 0 0 1,linecolor=cyan,%
    linestyle=dotted,Derivation=3]{-2}{4}
  \rput[lb](1.8,4){\textcolor{blue}{$f(x)=x^3$}}
  \rput[lb](0.2,8){\textcolor{red}{$f^{\prime\prime}(x)=6x$}}
  \rput[lb](-2,5){\textcolor{cyan}{$f^{\prime\prime\prime}(x)=6$}}
\end{pspicture*}
```




```
\begin{pspicture*}(-5,-5)(5,5)
  \psaxes{->}(0,0)(-5,-5)(5,5)%
  \psset{dotscale=2}
  \psPolynomial[markZeros,linecolor=red,linewidth=2pt,coeff=-1 1 -1 0 0.15]{-4}{3}%
  \psPolynomial[markZeros,linecolor=blue,linewidth=1pt,linestyle=dashed,%
    coeff=-1 1 -1 0 0.15,Derivation=1,zeroLineTo=0]{-4}{3}%
  \psPolynomial[markZeros,linecolor=magenta,linewidth=1pt,linestyle=dotted,%
    coeff=-1 1 -1 0 0.15,Derivation=2,zeroLineTo=0]{-4}{3}%
  \psPolynomial[markZeros,linecolor=magenta,linewidth=1pt,linestyle=dotted,%
    coeff=-1 1 -1 0 0.15,Derivation=2,zeroLineTo=1]{-4}{3}%
\end{pspicture*}
```




```
\psset{xunit=1.5}
\begin{pspicture*}(-5,-5)(5,5)
  \psaxes{->}(0,0)(-5,-5)(5,5)%
  \psset{dotscale=2,dotstyle=x,zeroLineStyle=dotted,zeroLineWidth=1pt}
  \psPolynomial[markZeros,linecolor=red,linewidth=2pt,coeff=-1 1 -1 0 0.15]{-4}{3}%
  \psPolynomial[markZeros,linecolor=blue,linewidth=1pt,linestyle=dashed,%
    coeff=-1 1 -1 0 0.15,Derivation=1,zeroLineTo=0]{-4}{3}%
  \psPolynomial[markZeros,linecolor=magenta,linewidth=1pt,linestyle=dotted,%
    coeff=-1 1 -1 0 0.15,Derivation=2,zeroLineTo=0]{-4}{3}%
  \psPolynomial[markZeros,linecolor=magenta,linewidth=1pt,linestyle=dotted,%
    coeff=-1 1 -1 0 0.15,Derivation=2,zeroLineTo=1]{-4}{3}%
\end{pspicture*}
```

2.3 \psBernstein

The polynomials defined by

$$B_{i,n}(t) = \binom{n}{i} t^i (1-t)^{n-i}$$

where $\binom{n}{k}$ is a binomial coefficient are named Bernstein polynomials of degree n. They form a basis for the power polynomials of degree n. The Bernstein polynomials satisfy symmetry

$$B_{i,n}(t) = B_{n-i,n}(1-t)$$

positivity

$$B_{i,n}(t) \geq 0 \qquad \text{for } 0 \leq t \leq 1$$

normalization

$$\sum_{i=0}^{n} B_{i,n}(t) = 1$$

and $B_{i,n}$ with $i! = 0$, n has a single unique local maximum of

$$i^i n^{-n} (n-i)^{n-i} \binom{n}{i}$$

occurring at $t = \frac{i}{n}$. The envelope $f_n(x)$ of the Bernstein polynomials $B_{i,n}(x)$ for $i = 0, 1, \ldots, n$ is given by

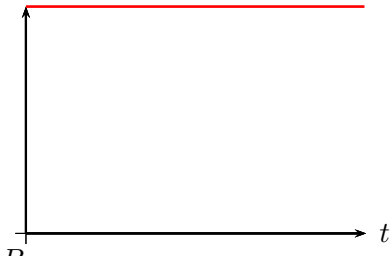
$$f_n(x) = \frac{1}{\sqrt{\pi n \cdot x(1-x)}}$$

illustrated below for $n = 20$.

`\psBernstein` `[Options]` `(`*tStart,tEnd*`)(`*i,n*`)`

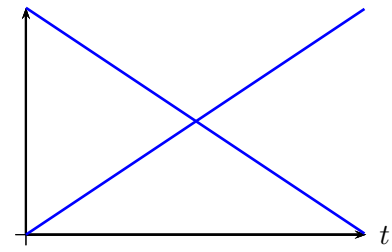
The (`tStart`, `tEnd`) are *optional* and preset by `(0,1)`. The only new optional argument is the boolean key `envelope`, which plots the envelope curve instead of the Bernstein polynomial.

$B_{0,0}$

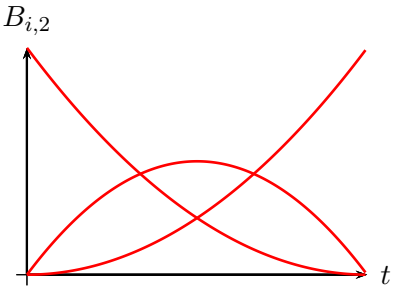



```
\psset{xunit=4.5cm,yunit=3cm}
\begin{pspicture}(1,1.1)
  \psaxes{->}(0,0)(1,1)[$t$,0][$B_{0,0}$,90]
  \psBernstein[linecolor=red,linewidth=1pt](0,0)
\end{pspicture}
```

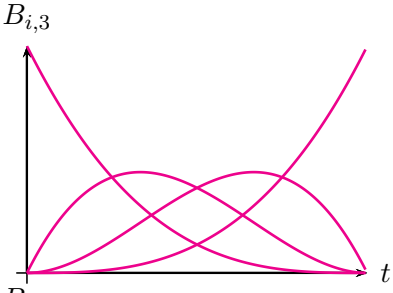
$B_{i,1}$



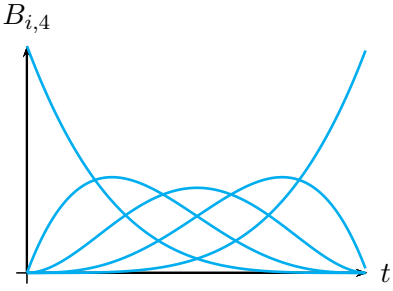

```
\psset{xunit=4.5cm,yunit=3cm}
\begin{pspicture}(1,1.1)
  \psaxes{->}(0,0)(1,1)[$t$,0][$B_{i,1}$,90]
  \psBernstein[linecolor=blue,linewidth=1pt](0,1)
  \psBernstein[linecolor=blue,linewidth=1pt](1,1)
\end{pspicture}
```



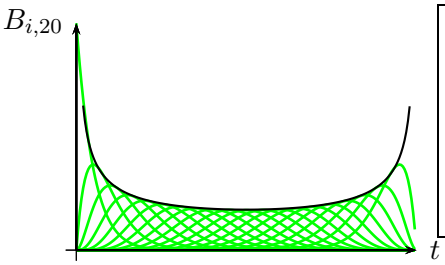

```
\psset{xunit=4.5cm,yunit=3cm}
\begin{pspicture}(1,1.1)
  \psaxes{->}(0,0)(1,1)[$t$,0][$B_{i,2}$,90]
  \multido{\i=0+1}{3}{\psBernstein[linecolor=red,
    linewidth=1pt](\i,2)}
\end{pspicture}
```



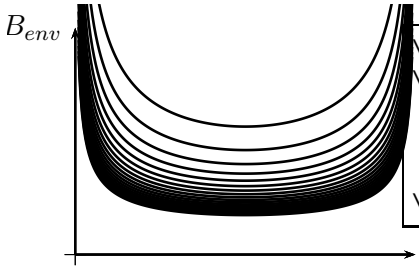

```
\psset{xunit=4.5cm,yunit=3cm}
\begin{pspicture}(1,1.1)
  \psaxes{->}(0,0)(1,1)[$t$,0][$B_{i,3}$,90]
  \multido{\i=0+1}{4}{\psBernstein[linecolor=magenta,
    linewidth=1pt](\i,3)}
\end{pspicture}
```




```
\psset{xunit=4.5cm,yunit=3cm}
\begin{pspicture}(1,1.1)
  \psaxes{->}(0,0)(1,1)[$t$,0][$B_{i,4}$,90]
  \multido{\i=0+1}{5}{\psBernstein[linecolor=cyan,
    linewidth=1pt](\i,4)}
\end{pspicture}
```




```
\psset{xunit=4.5cm,yunit=3cm}
\begin{pspicture}(-0.1,-0.05)(1.1,1.1)
  \multido{\i=0+1}{20}{\psBernstein[linecolor=green,
    linewidth=1pt](\i,20)}
  \psBernstein[envelope,linecolor=black](0.02,0.98)(0,20)
  \psaxes{->}(0,0)(1,1)[$t$,0][$B_{i,20}$,180]
\end{pspicture}
```

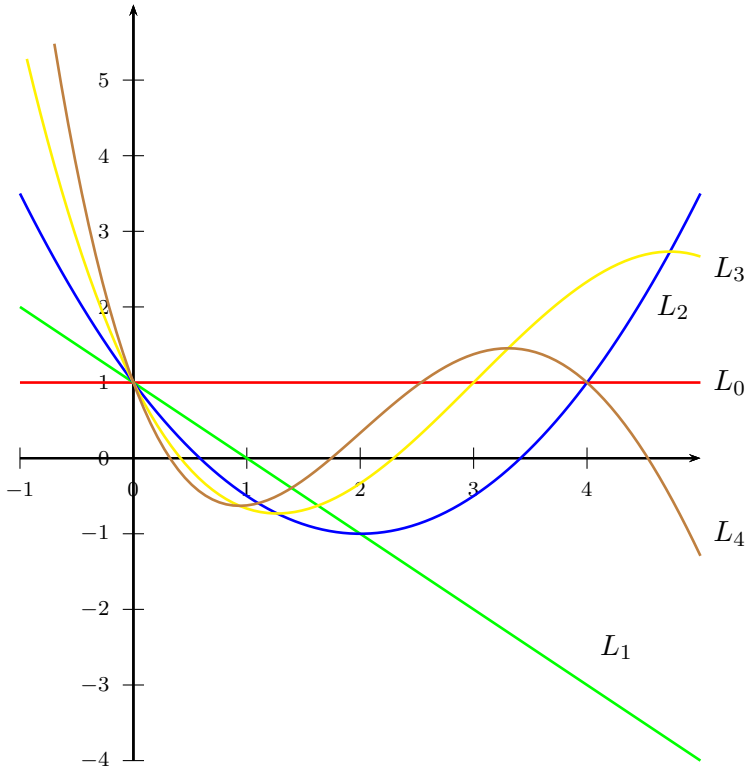



```
\psset{xunit=4.5cm,yunit=3cm}
\begin{pspicture*}(-0.2,-0.05)(1.1,1.1)
  \psaxes{->}(0,0)(1,1)[$t$,0][$B_{env}$,180]
  \multido{\i=2+1}{20}{\psBernstein[envelope,
    linewidth=1pt](0.01,0.99)(0,\i)}
\end{pspicture*}
```

2.4 Laguerre Polynomial

It is defined as

$$L_n(x) = \sum_{k=0}^{n} \frac{(-1)^k}{k!} \binom{n}{k} x^k$$




```
\psset{xunit=1.5}
\begin{pspicture}(-1,-4)(6,6)
\psaxes{->}(0,0)(-1,-4)(5,6)
\psset{yMaxValue=5,plotpoints=100,linewidth=1pt}
\psLaguerre[n=0,linecolor=red](-1,5)\uput[0](5,1){$L_0$}
\psLaguerre[n=1,linecolor=green](-1,5)\uput[0](4,-2.5){$L_1$}
\psLaguerre[n=2,linecolor=blue](-1,5)\uput[0](4.5,2){$L_2$}
\psLaguerre[n=3,linecolor=yellow](-1,5)\uput[0](5,2.5){$L_3$}
\psLaguerre[n=4,linecolor=brown](-1,5)\uput[0](5,-1){$L_4$}
\end{pspicture}
```

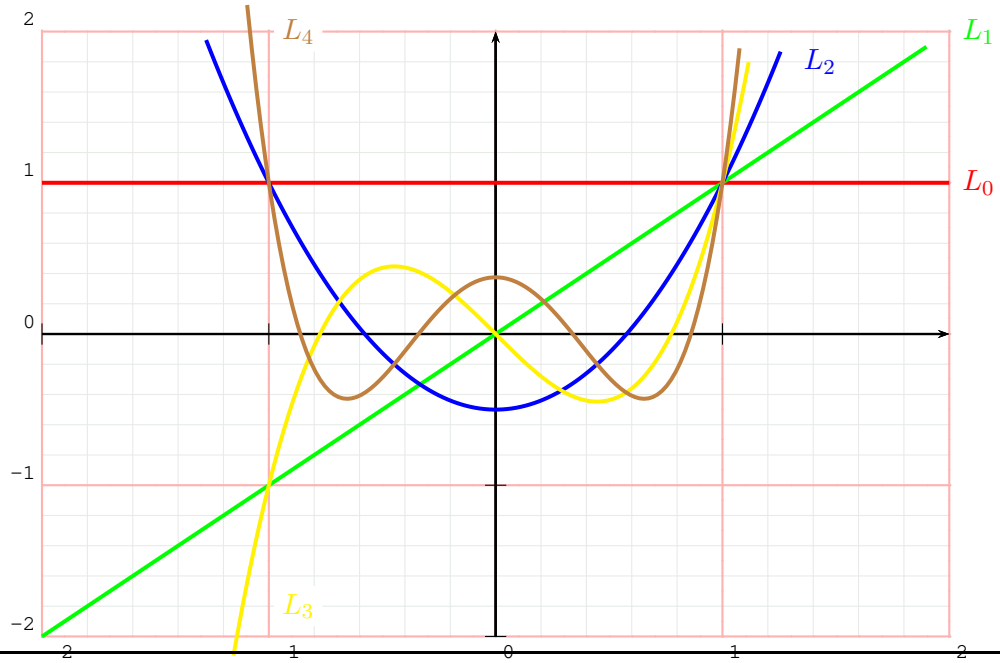
2.5 Legendre Polynomial

It is defined as an orthogonal system

$$\int_{-1}^{+1} P_m(x)(P_n(x)\mathrm{d}x = 0, \text{ with } n \neq m$$

For the computatíon we use a recursive definition:

$$\begin{aligned} P_0(x) &= 1 \\ P_1(x) &= x \\ n \cdot P_n(x) &= (2n-1)x \cdot P_{n-1}(x) - (n-1)P_{n-2}(x) \end{aligned}$$




```
\psset{xunit=3,yunit=2}
\begin{pspicture}[showgrid](-2,-2)(2,2)
\psaxes[labels=none]{->}(0,0)(-2,-2)(2,2)
\psset{yMaxValue=1.9,yMinValue=-2,plotpoints=200,linewidth=1.5pt}
\psLegendre[n=0,linecolor=red](-2,2)\uput[0](2,1){\textcolor{red}{$L_0$}}
\psLegendre[n=1,linecolor=green](-2,2)\uput[0](2,2){\textcolor{green}{$L_1$}}
\psLegendre[n=2,linecolor=blue](-2,2)\uput[0](1.3,1.8){\textcolor{blue}{$L_2$}}
\psLegendre[n=3,linecolor=yellow](-2,2)\uput*[0](-1,-1.8){\textcolor{yellow}{$L_3$}}
\psLegendre[n=4,linecolor=brown](-2,2)\uput*[0](-1,2){\textcolor{brown}{$L_4$}}
\end{pspicture}
```

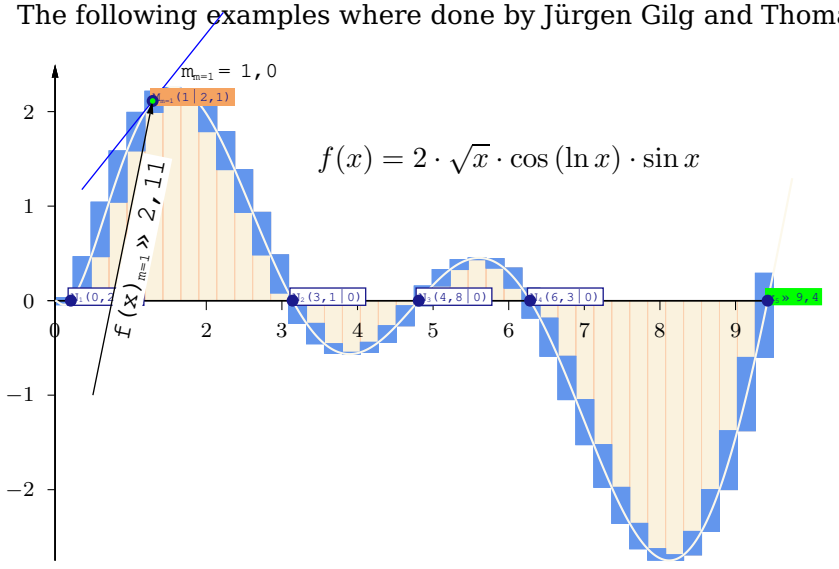
3 Calculating the zeros of a function or the the intermediate point of two function

`\psZero [Options] (`x_0, x_1`){`*functionA*`} [functionB] {`*node name*`}`

If the second function is not given the macro calculates and displays the zeros of the first function. If the second function is defined too, then the macro calculates the intermediate point of the two functions. The intervall is defined as $[x_0, x_1]$. Possible optional arguments are

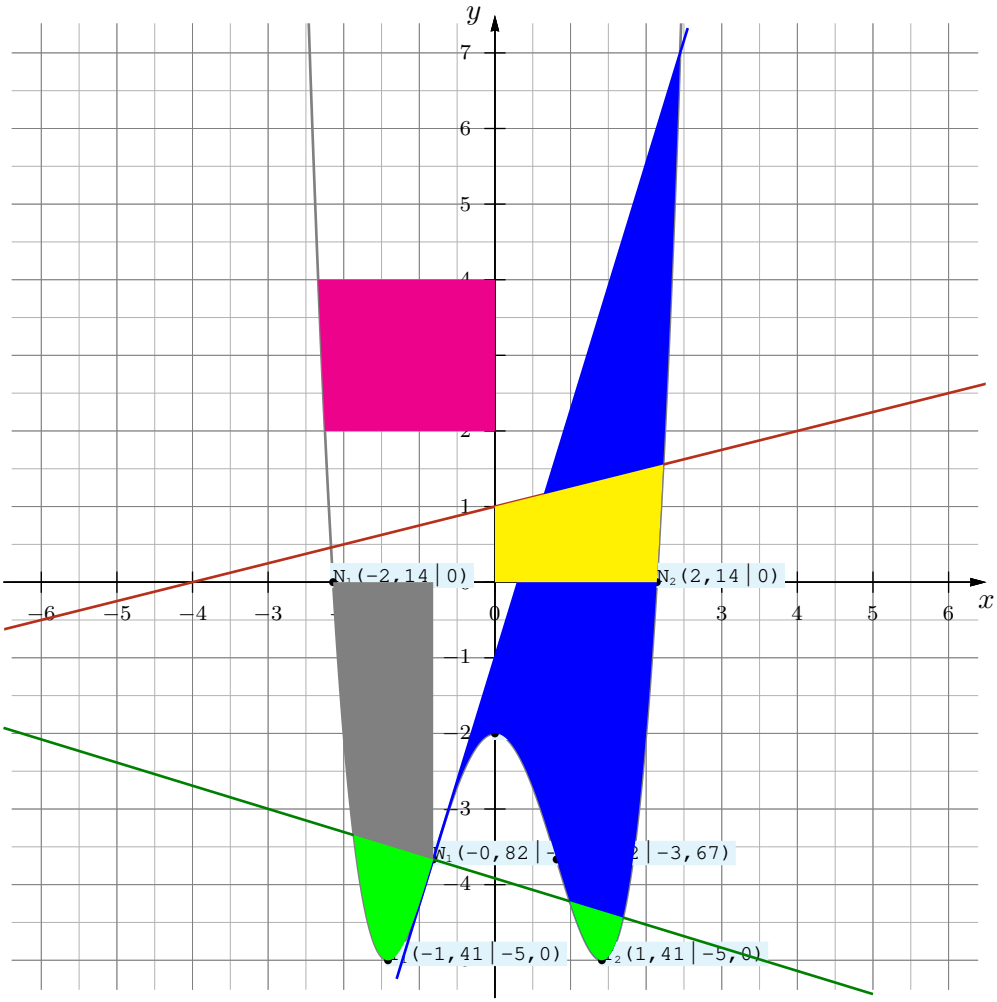
Name	*Default*	*Meaning*
markZeros	false	Mark the zeros/intermediate points with a symbol.
Newton	false	Use Newton method instead of the bisector one.
PrintCoord	false	Print the pair of coordinates of the zero/intermediate point, like $P(x\|y)$.
onlyNode	false	Calculate only the node, do not print anything, if markZeros = false.
onlyYVal	false	Print only the y-value.
xory	false	Print $x = x$-Value or, if onlyYVal = true, $y = y$-value.
approx	true	Change the =, if xory = true to $\approx$.
originV	false	Put the values without an offset.
Framed	false	Show a filled frame in backround, framesep, fillcolor, opacity or linestyle are options to show different frames.
PointName	I	The printed prefix for the calculated Points.
decimals	2	The decimals for the x value.
ydecimals	2	The decimals for the y value.
xShift	0	x move for the printed value.
yShift	0	y move for the printed value.

The following examples where done by Jürgen Gilg and Thomas Söll.




```
\definecolor{BeigeTS}{rgb}{0.98,0.95,0.87}
\definecolor{CornBlauTS}{rgb}{0.39,0.59,0.93}
\definecolor{SandBraun}{rgb}{0.96,0.64,0.38}
\psset{yunit=1.25cm,arrowinset=0.02,arrowlength=2,linewidth=0.5pt,saveNodeCoors,NodeCoorPrefix=n,
    comma}
\def\funkf{2*sqrt(x)*cos(ln(x))*sin(x)}
\begin{pspicture}[plotpoints=500,algebraic,fontscale=5,markZeros,PrintCoord,
  PointName=N,dotscale=0.7](-0.5,-3)(10,2.5)
\psStep[fillstyle=solid,fillcolor=BeigeTS,opacity=0.7,linewidth=0.3pt,
```

```
     linecolor=SandBraun!50](0.001,9.5){40}{\funkf}
\psStep[StepType=Riemann,fillstyle=solid,opacity=0.3,fillcolor=CornBlauTS,
     linecolor=CornBlauTS,linewidth=0.3pt](0.001,9.5){40}{\funkf}
\psaxes[labelFontSize=\scriptstyle,ticksize=-0.1 0]{->}(0,0)(0,-2.75)(10,2.5)
\psplot[linecolor=BeigeTS!60,linewidth=0.8pt]{0.001}{9.75}{\funkf}
\psplotTangent[linecolor=blue,Derive={Derive(1,\funkf)}]{1.29}{1.5}{\funkf}
\uput[90](6,1.2){$f(x)=2\cdot\sqrt{x}\cdot\cos{(\ln{x})}\cdot\sin{x}$}
{\psset{dotscale=1.5,linecolor=blue!50!black!90,ydecimals=0,Framed,opacity=0.8,decimals=1}
 \psZero[xShift=-0.2,yShift=0.15,postString=1,Newton](0.5,1){\funkf}{N1}
 \psZero[xShift=-0.05,yShift=0.15,postString=2](2,4){\funkf}{N2}
 \psZero[xShift=-0.45,yShift=0.15,postString=3](4,6){\funkf}{N3}
 \psZero[xShift=-0.45,yShift=0.15,postString=4](6,7){\funkf}{N4}
 \psZero[xShift=-0.45,yShift=0.15,postString=5](9,11){\funkf}{N5}
 \psZero[xShift=-1.15,yShift=0,PointName=M,decimals=0,linestyle=none,fillcolor=SandBraun,
  opacity=0.8,postString={m=1}](0.5,2){Derive(1,\funkf)-1+\funkf}[\funkf]{M}%
}
\pcline{->}(0.5,-1)(M)
\nbput[nrot=:U,labelsep=0.01]{%
  \scriptsize Steigung ist hier\phantom{i}
    \psPrintValueNew[PSfont=Palatino-Roman,decimals=0,round,fontscale=7]{nMx,{Derive(1,\funkf)}}}
\psdot[linecolor=green,strokeopacity=0.8](*{nMx} {\funkf})
\uput[90](*{nMx} {\funkf}){$m=$
  \psPrintValueNew[PSfont=Palatino-Roman,decimals=0,round,fontscale=8]{nMx,{Derive(1,\funkf)}}}
\end{pspicture}
```



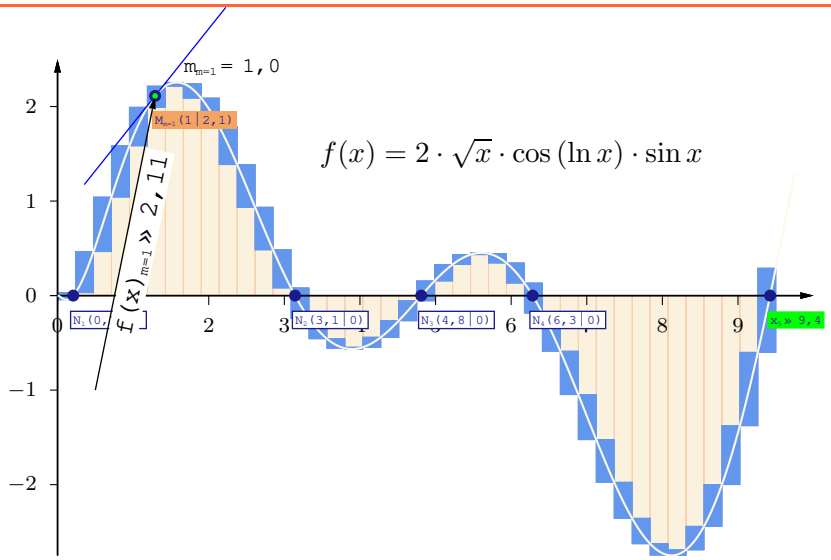
```
\psset{yunit=0.8,comma,decimals=2,algebraic=true,markZeros=true,plotpoints=500,saveNodeCoors,
    NodeCoorPrefix=n}
%----------------- FUNKTIONSDEFINITIONEN in "algebraic" -----------------
\def\funkf{0.75*x^4-3*x^2-2}
\def\funkg{0.25*x+1}

\begin{pspicture}(-6.5,-5.5)(6.5,8.5)
%------ Gitter im Hintergrund (CLIPPED) -----------------
\begin{psclip}%
{\psframe[linestyle=none](-6.4,-5.4)(6.4,7.4)}
\psgrid[subgriddiv=2,gridlabels=0,gridwidth=0.3pt,gridcolor=black!50,subgridwidth=0.2pt,subgridcolor
    =black!30](-6.5,-7.5)(6.5,8.5)
\end{psclip}
%--------- Achsen ------------
\psaxes[xDecimals=0, yDecimals=0,labelFontSize=\scriptstyle,arrowscale=1.3,arrowinset=0.05,
    arrowlength=1.9, Dy=1,dy=1,dx=1,Dx=1,subticks=0,comma,tickwidth=0.5pt]{->}(0,0)(-6.5,-5.5)
    (6.5,7.5)[$x$,-90][$y$,180]% Achsen
%----- Funktionsgraphen plotten (Clippen, damit sie nicht aus dem Gitter ragen) -----------------
\begin{psclip}%
{\psframe[linestyle=none](-6.5,-5.4)(6.5,7.4)}
\psplot[linewidth=1pt,linecolor=Gray]{-6.5}{6.5}{\funkf}%
\psplot[linewidth=1pt,linecolor=BrickRed]{-6.5}{6.5}{\funkg}%
\end{psclip}
%----------------- SPEZIELLE PUNKTE -----------------
{\psset{fontscale=8,PrintCoord=true,linestyle=none,opacity=0.8,Framed=true,fillcolor=cyan!10}
%----------------- NULLSTELLEN -----------------
\psZero[xShift=-0.9,yShift=0.15,PointName={N},postString={1},ydecimals=0](-3,-2){\funkf}[0]{N1}
\psZero[xShift=-0.9,yShift=0.15,PointName={N},postString={2},ydecimals=0](2,3){\funkf}[0]{N2}
%----------------- EXTREMWERTE -----------------
\psZero[xShift=-0.9,yShift=-0.25,PointName={T},postString={1}](-2,0){Derive(1,\funkf)+\funkf}[\funkf
    ]{T1}
\psZero[xShift=-0.9,yShift=0.25,PointName={H},postString={}](-1,1){Derive(1,\funkf)+\funkf}[\funkf]{
    H}
\psZero[xShift=-0.9,yShift=-0.25,PointName={T},postString={2}](0,2.5){Derive(1,\funkf)+\funkf}[\
    funkf]{T2}
%----------------- WENDEPUNKTE -----------------
\psZero[xShift=-1.2,yShift=-0.25,PointName={W},postString={1}](-1.5,-0.5){Derive(2,\funkf)+\funkf}[\
    funkf]{W1}
\psZero[xShift=-0.6,yShift=-0.25,PointName={W},postString={2}](0.5,1.5){Derive(2,\funkf)+\funkf}[\
    funkf]{W2}
\psZero[onlyNode=true,markZeros=false](-1.5,-0.5){Derive(2,\funkf)+Derive(1,\funkf)}[Derive(1,\funkf
    )]{mW1}%Steigung Wendepunkt 1 ist "nmW1y"
}
%----------------- GLEICHUNG WENDETANGENTE -----------------
\def\funkWende{nmW1y*(x-nW1x)+nW1y}
%----------------- GLEICHUNG WENDETANGENTE -----------------
\def\funkNormal{-1/nmW1y*(x-nW1x)+nW1y} %m_n=-1/m_t
%----------------- Tangente und Normale in W1 plotten ------------------
\psplot[linewidth=1pt,linecolor=blue]{-1.3}{2.55}{\funkWende}%
\psplot[linewidth=1pt,linecolor=Green]{-6.5}{5}{\funkNormal}%
%----------------- Punkte und Werte NICHT anzeigen
{\psset{onlyNode=true,markZeros=false}
%----------------- Schnittpunkt: Wendetangente in W1 mit f -------------
\psZero(0,4){\funkWende}[\funkf]{WS1}
%----------------- Schnittpunkte: Wendenormale in W1 mit f -------------
\psZero(-4,0){\funkNormal}[\funkf]{WN1}
\psZero(0,1.5){\funkNormal}[\funkf]{WN2}
\psZero(1.5,3){\funkNormal}[\funkf]{WN3}
```

```
%----------------- NULLSTELLE von g -----------------
\psZero(-3,3){\funkg}[0]{Ng1}
%----------------- SCHNITTPUNKTE f und g -----------------
\psZero(0,3){\funkg}[\funkf]{S1}
\psZero(-3,0){\funkg}[\funkf]{S2}
}
%----------------- FLÄCHE mit x-ACHSE -----------------
\pscustom[fillstyle=solid,opacity=0.3,fillcolor=gray,linestyle=none]{%
\psplot{nN1x}{nW1x}{\funkf}
\lineto(!nW1x 0)
\closepath
}
%----------------- FLÄCHE ZWISCHEN WENDETANGENTE UND KURVE f -----------------
\pscustom[fillstyle=solid,opacity=0.3,fillcolor=blue,linestyle=none]{%
\psplot{nW1x}{nWS1x}{\funkWende}
\psplot{nWS1x}{nW1x}{\funkf}
\closepath
}
%----------------- FLÄCHE ZWISCHEN WENDENORMALE UND KURVE f (Zwei FlÄchenstücke!!!) ----
%----------------- linke FLÄCHE -----------------
\pscustom[fillstyle=solid,opacity=0.3,fillcolor=green,linestyle=none]{%
\psplot{nWN1x}{nW1x}{\funkNormal}
\psplot{nW1x}{nWN1x}{\funkf}
\closepath
}
%----------------- rechte FLÄCHE -----------------
\pscustom[fillstyle=solid,opacity=0.3,fillcolor=green,linestyle=none]{%
\psplot{nWN2x}{nWN3x}{\funkNormal}
\psplot{nWN3x}{nWN2x}{\funkf}
\closepath
}
%----------------- FLÄCHE zwischen den KURVEN f und g und beiden KOORDINATEN-ACHSEN -----
\pscustom[fillstyle=solid,opacity=0.3,fillcolor=yellow,linestyle=none]{%
  \psplot{0}{nS1x}{\funkg}
  \psplot{nS1x}{nN2x}{\funkf}
  \lineto(0,0)
  \closepath}
% SPIELEREI: FLÄCHE mit f und PARALLELEN ZUR x-ACHSE
% Punkte und Werte NICHT anzeigen
{\psset{onlyNode=true,markZeros=false}
\psZero(-3,-2){\funkf}[2]{M1}
\psZero(-3,-2){\funkf}[4]{M2}}
\pscustom[fillstyle=solid,opacity=0.3,fillcolor=magenta,linestyle=none]{%
  \psplot{nM1x}{nM2x}{\funkf}
  \lineto(0,4)
  \lineto(0,2)
  \closepath}
\end{pspicture}
```

```
\definecolor{BeigeTS}{rgb}{0.98,0.95,0.87}
\definecolor{CornBlauTS}{rgb}{0.39,0.59,0.93}
\definecolor{SandBraun}{rgb}{0.96,0.64,0.38}
\psset{yunit=1.25cm,arrowinset=0.02,arrowlength=2,linewidth=0.5pt,saveNodeCoors,NodeCoorPrefix=n}
\def\funkf{2*sqrt(x)*cos(ln(x))*sin(x)}
\begin{pspicture}[plotpoints=500,algebraic,fontscale=5,markZeros,PrintCoord,
  PointName=N,dotscale=0.7](-0.5,-3)(10,2.5)
\psStep[fillstyle=solid,fillcolor=BeigeTS,opacity=0.7,linewidth=0.3pt,
     linecolor=SandBraun!50](0.001,9.5){40}{\funkf}
\psStep[StepType=Riemann,fillstyle=solid,opacity=0.3,fillcolor=CornBlauTS,
```

```
     linecolor=CornBlauTS,linewidth=0.3pt](0.001,9.5){40}{\funkf}
\psaxes[labelFontSize=\scriptstyle,ticksize=-0.1 0]{->}(0,0)(0,-2.75)(10,2.5)
\psplot[linecolor=BeigeTS!60,linewidth=0.8pt]{0.001}{9.75}{\funkf}
\psplotTangent[linecolor=blue,Derive={Derive(1,\funkf)}]{1.29}{1.5}{\funkf}
\uput[90](6,1.2){$f(x)=2\cdot\sqrt{x}\cdot\cos{(\ln{x})}\cdot\sin{x}$}
{\psset{dotscale=1.5,linecolor=blue!50!black!90,ydecimals=0}
 \psZero[xShift=-0.2,yShift=0.15,postString=1,Newton](0.5,1){\funkf}{N1}
 \psZero[xShift=-0.05,yShift=0.15,postString=2](2,4){\funkf}{N2}
 \psZero[xShift=-0.45,yShift=0.15,postString=3](4,6){\funkf}{N3}
 \psZero[xShift=-0.45,yShift=0.15,postString=4](6,7){\funkf}{N4}
 \psZero[xShift=-0.45,yShift=0.15,postString=5](9,11){\funkf}{N5}
 \psZero[xShift=-1.15,yShift=0,PointName=M,
   postString={m=1}](0.5,2){Derive(1,\funkf)-1+\funkf}[\funkf]{M}%
}
\pcline{->}(0.5,-1)(M)
\nbput[nrot=:U,labelsep=0.01]{%
  \scriptsize Steigung ist hier
    \psPrintValueNew[PSfont=Palatino-Roman,decimals=0,round,fontscale=7]{nMx,{Derive(1,\funkf)}}}
\psdot[linecolor=green,strokeopacity=0.8](*{nMx} {\funkf})
\uput[90](*{nMx} {\funkf}){$m=$
  \psPrintValueNew[PSfont=Palatino-Roman,decimals=0,round,fontscale=8]{nMx,{Derive(1,\funkf)}}}
\end{pspicture}
```

As an alternative the values of the zeros can be placed by using the optional arguments `labelangle` and `labeldistance`:




```
\definecolor{BeigeTS}{rgb}{0.98,0.95,0.87}
\definecolor{CornBlauTS}{rgb}{0.39,0.59,0.93}
\definecolor{SandBraun}{rgb}{0.96,0.64,0.38}
\psset{yunit=1.25cm,arrowinset=0.02,arrowlength=2,linewidth=0.5pt,saveNodeCoors,NodeCoorPrefix=n,comma
    }
\def\funkf{2*sqrt(x)*cos(ln(x))*sin(x)}
\begin{pspicture}[plotpoints=500,algebraic,fontscale=5,markZeros,
  PointName=N,dotscale=0.7](-0.5,-3)(10,2.5)
\psStep[fillstyle=solid,fillcolor=BeigeTS,opacity=0.7,linewidth=0.3pt,
     linecolor=SandBraun!50](0.001,9.5){40}{\funkf}
\psStep[StepType=Riemann,fillstyle=solid,opacity=0.3,fillcolor=CornBlauTS,
     linecolor=CornBlauTS,linewidth=0.3pt](0.001,9.5){40}{\funkf}
\psaxes[labelFontSize=\scriptstyle,ticksize=-0.1 0]{->}(0,0)(0,-2.75)(10,2.5)
\psplot[linecolor=BeigeTS!60,linewidth=0.8pt]{0.001}{9.75}{\funkf}
\psplotTangent[linecolor=blue,Derive={Derive(1,\funkf)}]{1.29}{1.5}{\funkf}
\uput[90](6,1.2){$f(x)=2\cdot\sqrt{x}\cdot\cos{(\ln{x})}\cdot\sin{x}$}
{\psset{dotscale=1.5,linecolor=blue!50!black!90,ydecimals=0,Framed,opacity=0.8,decimals=1,PrintCoord}
 \psZero[labelangle=-90,labeldistance=0.3,postString=1,Newton](0.5,1){\funkf}{N1}
 \psZero[labelangle=-90,labeldistance=0.3,postString=2](2,4){\funkf}{N2}
 \psZero[labelangle=-90,labeldistance=0.3,postString=3](4,6){\funkf}{N3}
 \psZero[labelangle=-90,labeldistance=0.3,postString=4](6,7){\funkf}{N4}
 \psZero[labelangle=-90,labeldistance=0.3,PointName=x,postString=5,xory,PrintCoord=false,
   linestyle=none,fillcolor=green,opacity=0.6](9,11){\funkf}{N5}
 \psZero[labelangle=-90,labeldistance=0.3,PointName=M,decimals=0,linestyle=none,fillcolor=SandBraun,
  ydecimals=1,opacity=0.8,postString={m=1}](0.5,2){Derive(1,\funkf)-1+\funkf}[\funkf]{M}%
}
\pcline{->}(0.5,-1)(M)
\nbput[nrot=:U,labelsep=0.3,npos=0.2]{%
  \scriptsize \psZero[originV=true,xory=true,onlyYVal=true,PointName=f(x),postString={m=1},Framed,
  opacity=0.8,linestyle=none,markZeros=false,fontscale=10](0.5,2){Derive(1,\funkf)-1+\funkf}[\funkf]{R
     }}
\psdot[linecolor=green,strokeopacity=0.8](M)
\uput{0.5}[40](M){\psZero[originV=true,approx=false,xory=true,onlyYVal=true,
 PointName=m,postString={m=1},markZeros=false,fontscale=8](0.5,2){Derive(1,\funkf)-1}[1]{R}}
\end{pspicture}
```

4 \psFourier

A Fourier sum has the form:

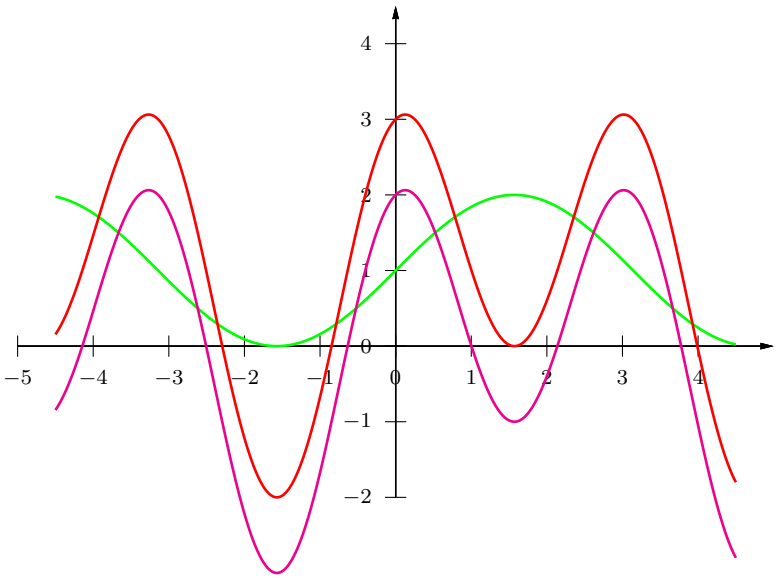
$$s(x) = \frac{a_0}{2} + a_1 \cos \omega x + a_2 \cos 2\omega x + a_3 \cos 3\omega x + \ldots + a_n \cos n\omega x \tag{15}$$

$$+ b_1 \sin \omega x + b_2 \sin 2\omega x + b_3 \sin 3\omega x + \ldots + b_m \sin m\omega x \tag{16}$$

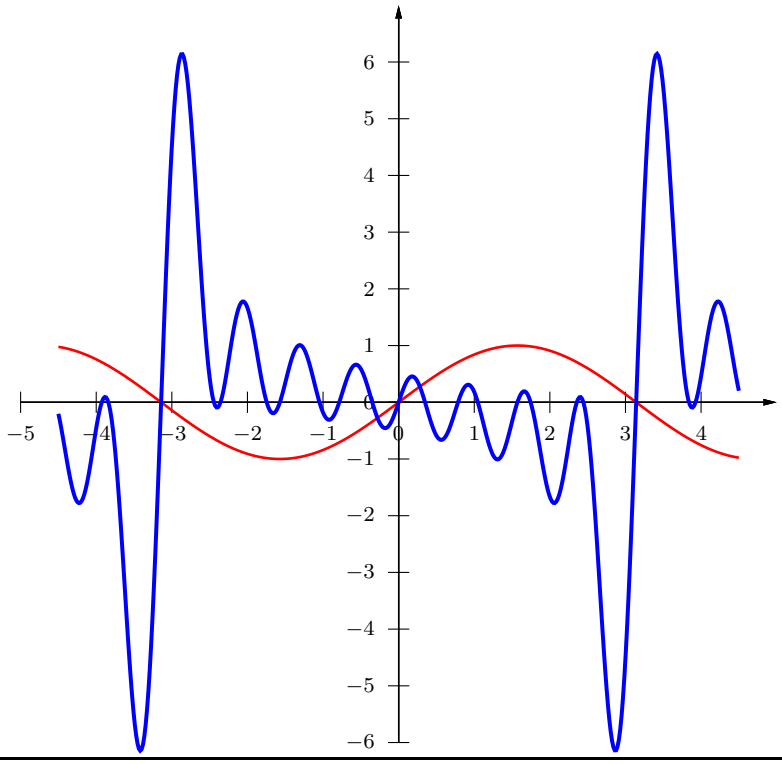
The macro `\psFourier` plots Fourier sums. The syntax is similiar to `\psPolynomial`, except that there are two kinds of coefficients:

`\psFourier [Options] {xStart}{xEnd}`

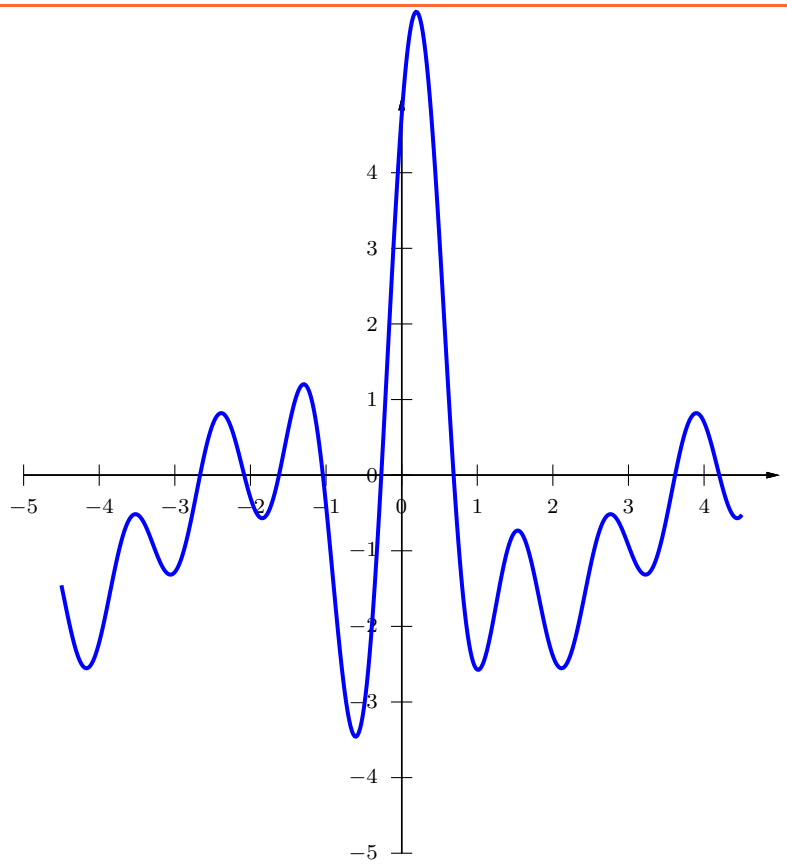
The coefficients must have the orders $cosCoeff = a_0\ a_1\ a_2\ \ldots$ and $sinCoeff = b_1\ b_2\ b_3\ \ldots$ and be separated by **spaces**. The default is `cosCoeff=0,sinCoeff=1`, which gives the standard `sin` function. Note that the constant value can only be set with `cosCoeff=a0`.




```
\begin{pspicture}(-5,-3)(5,5.5)
\psaxes{->}(0,0)(-5,-2)(5,4.5)
\psset{plotpoints=500,linewidth=1pt}
\psFourier[cosCoeff=2, linecolor=green]{-4.5}{4.5}
\psFourier[cosCoeff=0 0 2, linecolor=magenta]{-4.5}{4.5}
\psFourier[cosCoeff=2 0 2, linecolor=red]{-4.5}{4.5}
\end{pspicture}
```




```
\psset{yunit=0.75}
\begin{pspicture}(-5,-6)(5,7)
\psaxes{->}(0,0)(-5,-6)(5,7)
\psset{plotpoints=500}
\psFourier[linecolor=red,linewidth=1pt]{-4.5}{4.5}
\psFourier[sinCoeff= -1 1 -1 1 -1 1 -1 1,%
   linecolor=blue,linewidth=1.5pt]{-4.5}{4.5}
\end{pspicture}
```




```
\begin{pspicture}(-5,-5)(5,5.5)
\psaxes{->}(0,0)(-5,-5)(5,5)
\psset{plotpoints=500,linewidth=1.5pt}
\psFourier[sinCoeff=-.5 1 1 1 1 ,cosCoeff=-.5 1 1 1 1 1,%
   linecolor=blue]{-4.5}{4.5}
\end{pspicture}
```

5 \psBessel

The Bessel function of order n is defined as

$$J_n(x) = \frac{1}{\pi}\int_0^\pi \cos(x\sin t - nt)\,\mathrm{d}t \tag{17}$$

$$= \sum_{k=0}^{\infty} \frac{(-1)^k \left(\frac{x}{2}\right)^{n+2k}}{k!\Gamma(n+k+1)} \tag{18}$$

The syntax of the macro is

\psBessel [Options] {*order*}{*xStart*}{*xEnd*}

There are two special parameters for the Bessel function, and also the settings of many pst-plot or pstricks parameters affect the plot. These two “constants” have the following meaning:

$$f(t) = constI \cdot J_n + constII$$

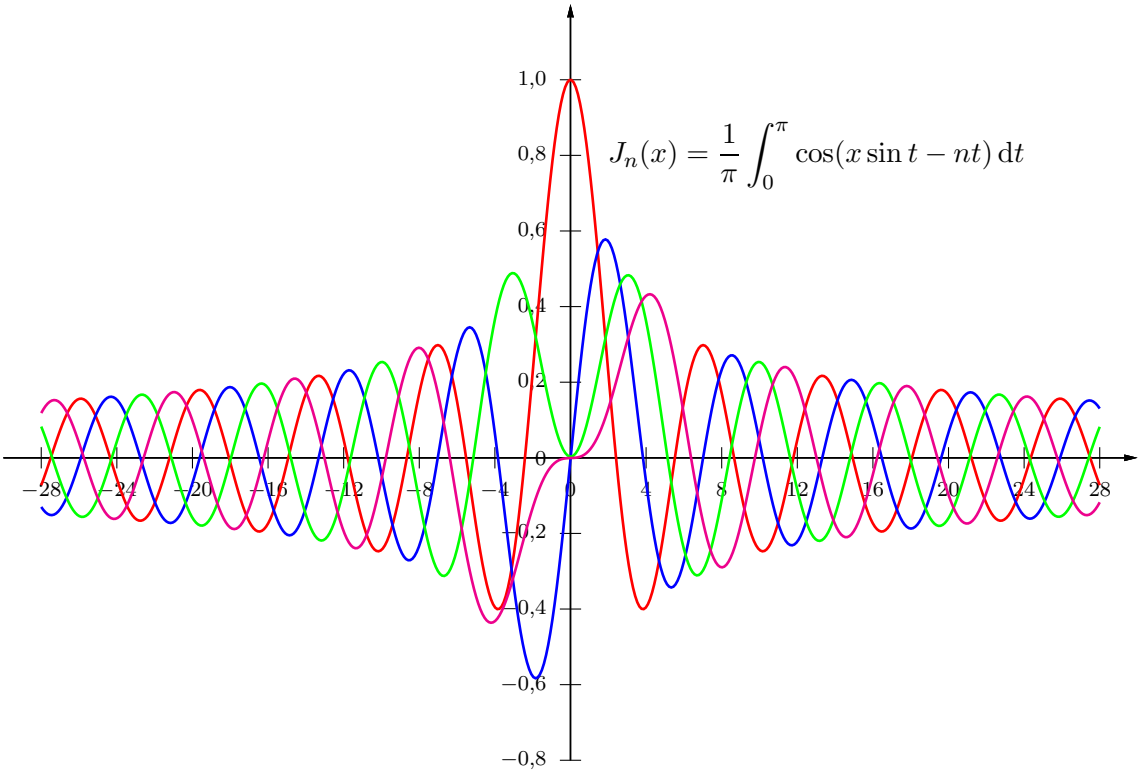
where constI and constII must be real PostScript expressions, e.g.

```
\psset{constI=2.3,constII=t k sin 1.2 mul 0.37 add}
```

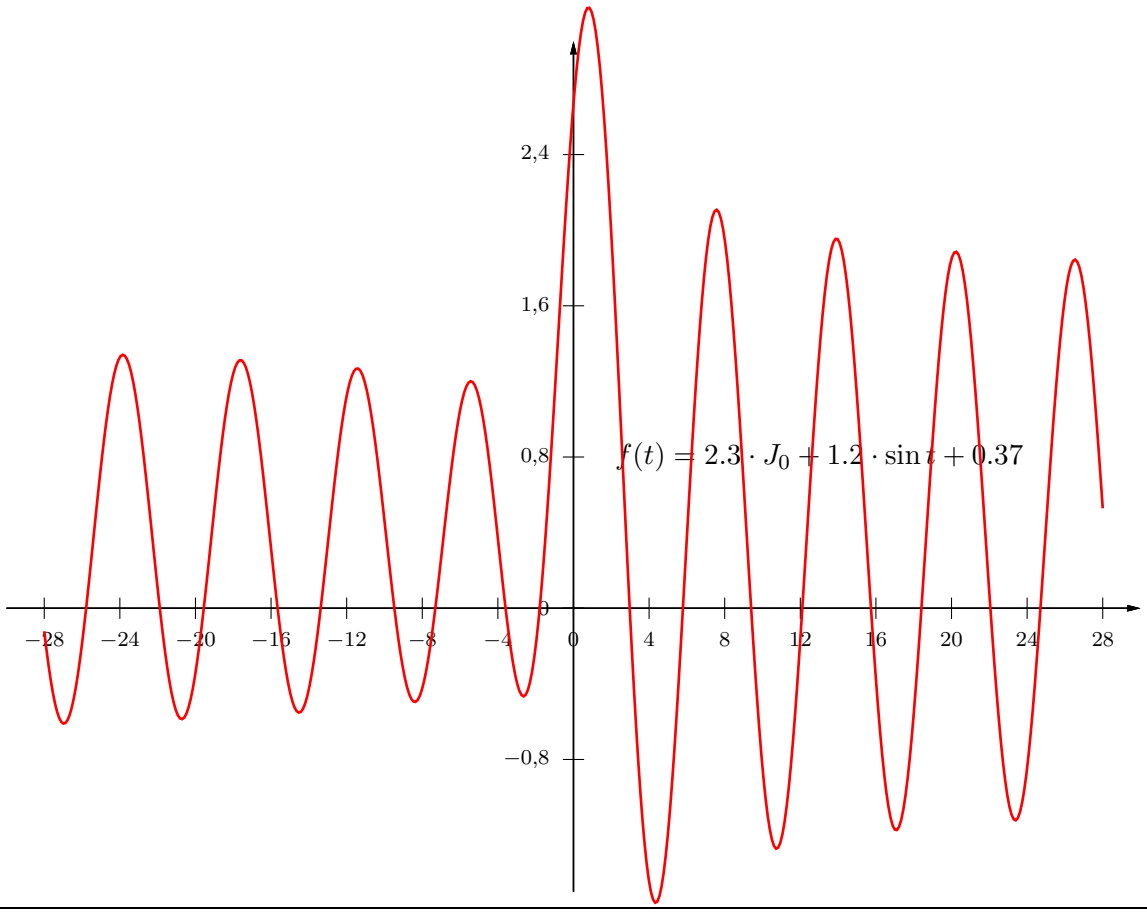
The Bessel function is plotted with the parametricplot macro, this is the reason why the variable is named t. The internal procedure k converts the value t from radian into degrees. The above setting is the same as

$$f(t) = 2.3 \cdot J_n + 1.2 \cdot \sin t + 0.37$$

In particular, note that the default for plotpoints is 500. If the plotting computations are too time consuming at this setting, it can be decreased in the usual way, at the cost of some reduction in graphics resolution.




```
{
\psset{xunit=0.25,yunit=5}
\begin{pspicture}(-13,-.85)(13,1.25)
\rput(13,0.8){%
   $\displaystyle J_n(x)=\frac{1}{\pi}\int_0^\pi\cos(x\sin t-nt)\dt$%
}
\psaxes[Dy=0.2,Dx=4]{->}(0,0)(-30,-.8)(30,1.2)
\psset{linewidth=1pt}
\psBessel[linecolor=red]{0}{-28}{28}%
\psBessel[linecolor=blue]{1}{-28}{28}%
\psBessel[linecolor=green]{2}{-28}{28}%
\psBessel[linecolor=magenta]{3}{-28}{28}%
\end{pspicture}
}
```




```
{
\psset{xunit=0.25,yunit=2.5}
\begin{pspicture}(-13,-1.5)(13,3)
\rput(13,0.8){%
   $\displaystyle f(t) = 2.3 \cdot J_0 + 1.2\cdot \sin t + 0.37$%
}
\psaxes[Dy=0.8,dy=2cm,Dx=4]{->}(0,0)(-30,-1.5)(30,3)
\psset{linewidth=1pt}
\psBessel[linecolor=red,constI=2.3,constII={t k sin 1.2 mul 0.37 add}]{0}{-28}{28}%
\end{pspicture}
}
```

6 Modfied Bessel function of first order

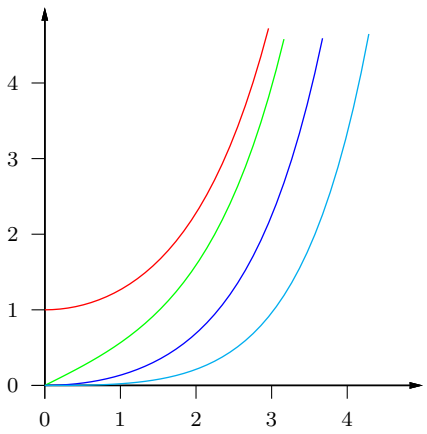
The modified Bessel function of first order is defined as

$$I_\nu(x) = \left(\frac{1}{2}x\right)^\nu \sum_{k=0}^{\infty} \frac{\left(\frac{1}{4}x^2\right)^k}{k!\Gamma(\nu+k+1)} \tag{19}$$

The syntax of the macro is

```
\psModBessel [Options] {xStart}{xEnd}
```

The only valid optional argument for the function is nue, which is preset to 0, it shows I_0.




```
\begin{pspicture}(0,-0.5)(5,5)
\psaxes[ticksize=-5pt 0]{->}(5,5)
\psModBessel[yMaxValue=5,nue=0,linecolor=red]{0}{5}
\psModBessel[yMaxValue=5,nue=1,linecolor=green]{0}{5}
\psModBessel[yMaxValue=5,nue=2,linecolor=blue]{0}{5}
\psModBessel[yMaxValue=5,nue=3,linecolor=cyan]{0}{5}
\end{pspicture}
```

7 \psSi, \pssi and \psCi

The integral sin and cosin are defined as

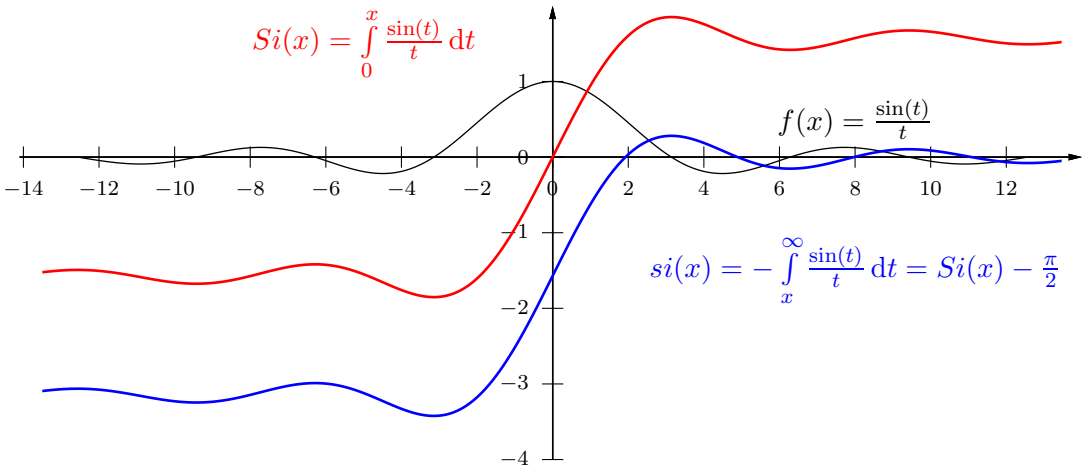
$$\mathrm{Si}(x) = \int_0^x \frac{\sin t}{t}\,\mathrm{d}t \tag{20}$$

$$\mathrm{si}(x) = -\int_x^\infty \frac{\sin t}{t}\,\mathrm{d}t = \mathrm{Si}(x) - \frac{\pi}{2} \tag{21}$$

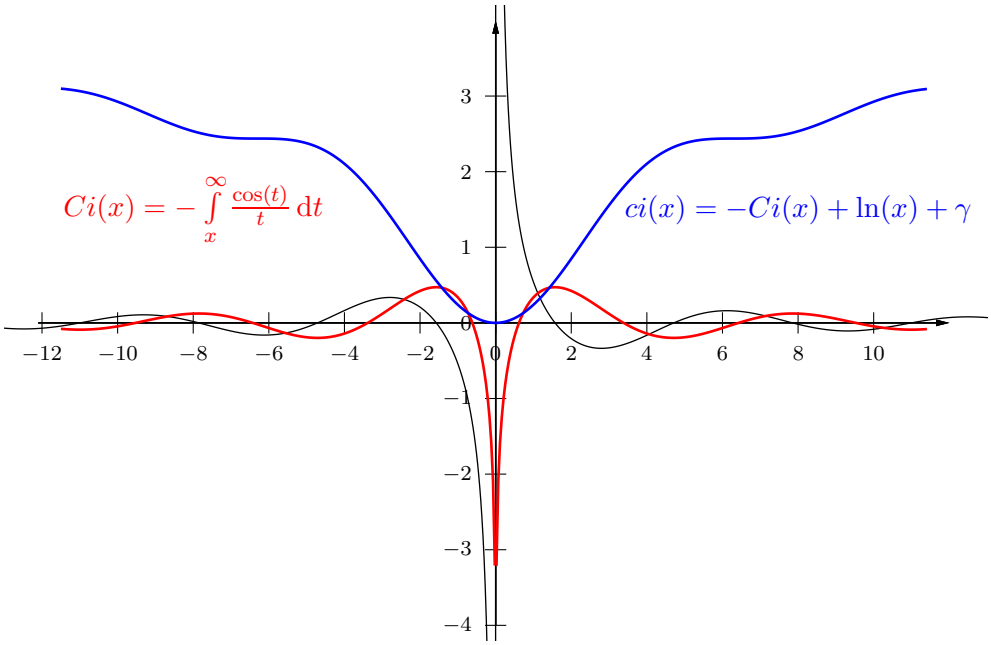
$$\mathrm{Ci}(x) = -\int_x^\infty \frac{\cos t}{t}\,\mathrm{d}t = \gamma + \ln x + \int_0^x \frac{\cos t - 1}{t}\,\mathrm{d}t \tag{22}$$

The syntax of the macros is

```
\psSi [Options] {xStart}{xEnd}
\pssi [Options] {xStart}{xEnd}
\psCi [Options] {xStart}{xEnd}
```




```
\psset{xunit=0.5}
\begin{pspicture}(-15,-4.5)(15,2)
  \psaxes[dx=1cm,Dx=2]{->}(0,0)(-14.1,-4)(14,2)
  \psplot[plotpoints=1000]{-12.5}{12.5}{ x RadtoDeg sin x div }
  \psSi[plotpoints=1500,linecolor=red,linewidth=1pt]{-13.5}{13.5}
  \pssi[plotpoints=1500,linecolor=blue,linewidth=1pt]{-13.5}{13.5}
  \rput(-5,1.5){\color{red}$Si(x)=\int\limits_{0}^x \frac{\sin(t)}{t}\dt$}
  \rput(8,-1.5){\color{blue}$si(x)=-\int\limits_{x}^{\infty} \frac{\sin(t)}{t}\dt=Si(x)-\frac{\pi
      }{2}$}
  \rput(8,.5){$f(x)= \frac{\sin(t)}{t}$}
\end{pspicture}
```




```
\psset{xunit=0.5}
\begin{pspicture*}(-13,-4.2)(13,4.2)
  \psaxes[dx=1cm,Dx=2]{->}(0,0)(-12.1,-4)(12,4)
  \psplot[plotpoints=1000]{-14.5}{14.5}{ x RadtoDeg cos x Div }
  \psCi[plotpoints=500,linecolor=red,linewidth=1pt]{-11.5}{11.5}
  \psci[plotpoints=500,linecolor=blue,linewidth=1pt]{-11.5}{11.5}
  \rput(-8,1.5){\color{red}$Ci(x)=-\int\limits_{x}^{\infty} \frac{\cos(t)}{t}\dt$}
  \rput(8,1.5){\color{blue}$ci(x)=-Ci(x)+\ln(x)+\gamma$}
\end{pspicture*}
```

8 \psIntegral, \psCumIntegral, and \psConv

These new macros[1] allows to plot the result of an integral using the Simpson numerical integration rule. The first one is the result of the integral of a function with two variables, and the integral is performed over one of them. The second one is the cumulative integral of a function (similar to \psGaussI but valid for all functions). The third one is the result of a convolution. They are defined as:

$$\texttt{\textbackslash psIntegral}(x) = \int_a^b f(x,t)\mathrm{d}t \tag{23}$$

$$\texttt{\textbackslash psCumIntegral}(x) = \int_{\text{xStart}}^{x} f(t)\mathrm{d}t \tag{24}$$

$$\texttt{\textbackslash psConv}(x) = \int_a^b f(t)g(x-t)\mathrm{d}t \tag{25}$$

In the first one, the integral is performed from a to b and the function f depends on two parameters. In the second one, the function f depends on only one parameter, and the integral is performed from the minimum value specified for x (xStart) and the current value of x in the plot. The third one uses the \psIntegral macro to perform an approximation to the convolution, where the integration is performed from a to b.

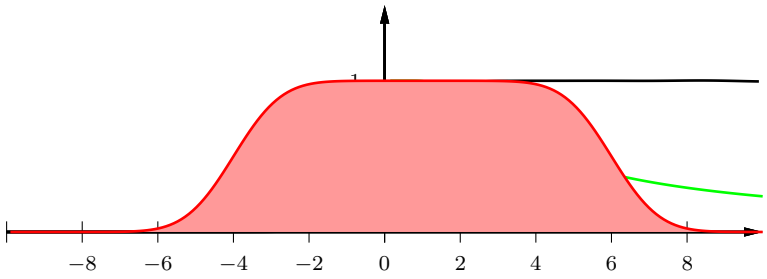
The syntax of these macros is:

```
\psIntegral [Options] {xStart}{xEnd}(a,b){ function }
\psCumIngegral [Options] {xStart}{xEnd}{ function }
\psConv [Options] {xStart}{xEnd}(a,b){ function f }{ function g }
```

In the first macro, the function should be created such that it accepts two values: <x t function> should be a value. For the second and the third functions, they only need to accept one parameter: <x function> should be a value.

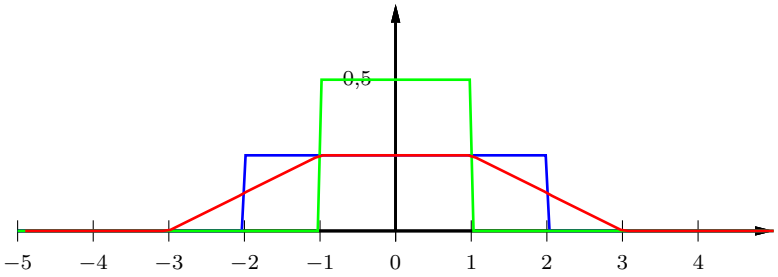
There are no new parameters for these functions. The two most important ones are plotpoints, which controls the number of points of the plot (number of divisions on x for the plot) and Simpson, which controls the precision of the integration (a larger number means a smallest step). The precision and the smoothness of the plot depend strongly on these two parameters.

1 Created by Jose-Emilio Vila-Forcen



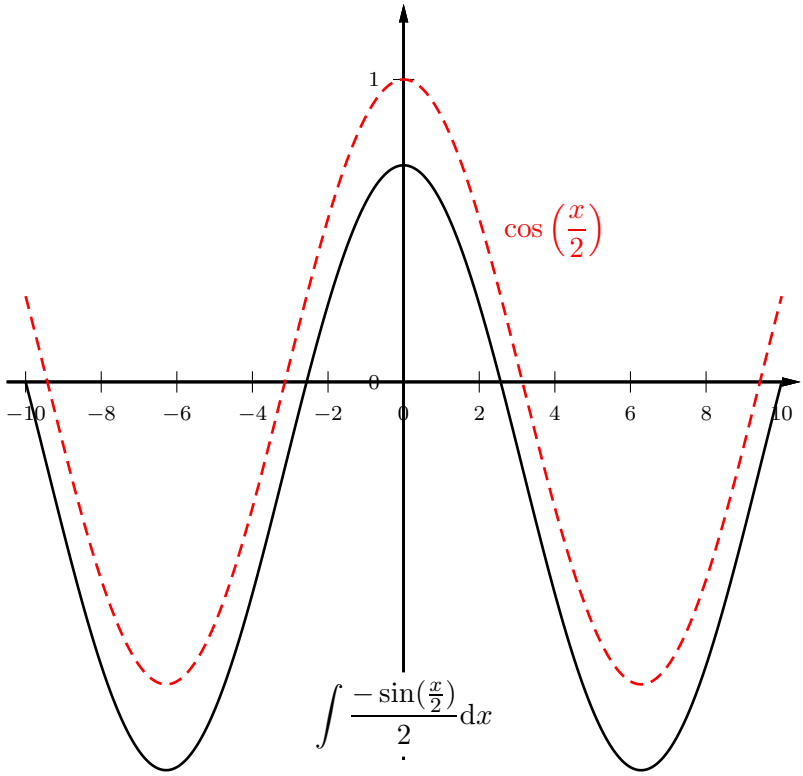

```
%\usepackage{pst-math}
\psset{xunit=0.5cm,yunit=2cm}
\begin{pspicture}[linewidth=1pt](-10,-.5)(10,1.5)
  \psaxes[dx=1cm,Dx=2]{->}(0,0)(-10,0)(10,1.5)
  \psCumIntegral[plotpoints=200,Simpson=10]{-10}{10}{0 1 GAUSS}
  \psIntegral[plotpoints=200,Simpson=100,linecolor=green]{.1}{10}(-3,3){0 exch GAUSS}
  \psIntegral[plotpoints=200,Simpson=10,linecolor=red,
    fillcolor=red!40,fillstyle=solid,opacity=0.5]{-10}{10}(-4,6){1 GAUSS}
\end{pspicture}
```

In the example, the cumulative integral of a Gaussian is presented in black. In red, a Gaussian is varying its mean from -10 to 10, and the result is the integral from -4 to 6. Finally, in green it is presented the integral of a Gaussian from -3 to 3, where the variance is varying from 0.1 to 10.




```
\psset{xunit=1cm,yunit=4cm}
\begin{pspicture}[linewidth=1pt](-5,-.2)(5,0.75)
  \psaxes[dx=1cm,Dx=1,Dy=0.5]{->}(0,0)(-5,0)(5,0.75)
  \psplot[linecolor=blue,plotpoints=200]{-5}{5}{x abs 2 le {0.25}{0} ifelse}
  \psplot[linecolor=green,plotpoints=200]{-5}{5}{x abs 1 le {.5}{0} ifelse}
  \psConv[plotpoints=100,Simpson=1000,linecolor=red]{-5}{5}(-10,10)%
    {abs 2 le {0.25}{0} ifelse}{abs 1 le {.5} {0} ifelse}
\end{pspicture}
```

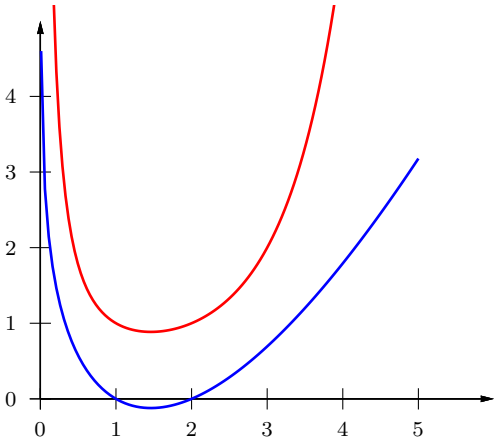
In the second example, a convolution is performed using two rectangle functions. The result (in red) is a trapezoid function.




```
\psset{xunit=0.5cm,yunit=4cm}
\begin{pspicture}[linewidth=1pt](-11,-1.5)(11,1.5)
  \psaxes[dx=1cm,Dx=2]{->}(0,0)(-10.5,-1.25)(10.5,1.25)
  \psCumIntegral[plotpoints=2000,Simpson=10,algebraic]{-10}{10}{-sin(x/2)/2}
  \psplot[plotpoints=2000,linestyle=dashed,linecolor=red,algebraic]{-10}{10}{cos(x/2)}
  \rput(4,0.5){\textcolor{red}{$\displaystyle\cos\left(\frac{x}2\right)$}}
  \rput*(0,-1.1){$\displaystyle\int\limits\frac{-\sin(\frac{x}2)}{2}\mathrm dx$}
\end{pspicture}
```

9 Distributions

All distributions which use the Γ- or $\ln\Gamma$-function need the `pst-math` package, it defines the PostScript functions `GAMMA` and `GAMMALN`. `pst-func` reads by default the PostScript file `pst-math.pro`. It is part of any TEX distribution and should also be on your system, otherwise install or update it from CTAN. It must be the latest version.




```
\begin{pspicture*}(-0.5,-0.5)(6.2,5.2)
 \psaxes{->}(0,0)(6,5)
 \psset{plotpoints=100,linewidth=1pt}
 \psplot[linecolor=red]{0.01}{4}{ x GAMMA }
 \psplot[linecolor=blue]{0.01}{5}{ x GAMMALN }
\end{pspicture*}
```

9.1 Normal distribution (Gauss)

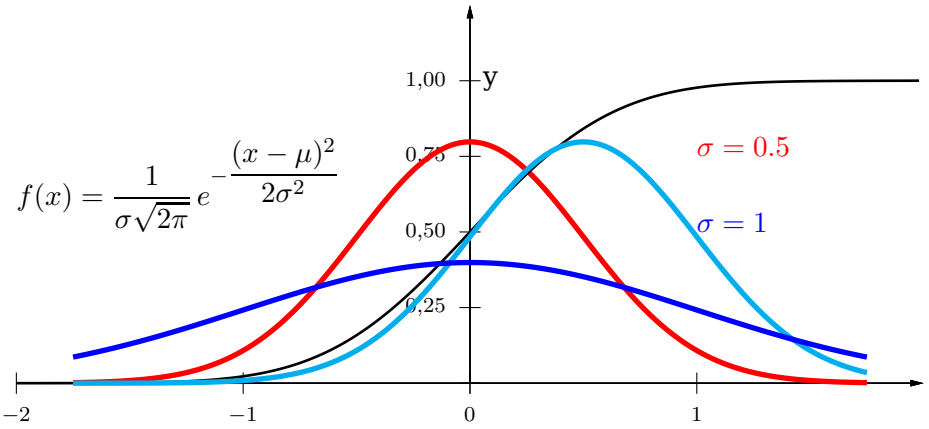
The Gauss function is defined as

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}}\, e^{-\frac{(x-\mu)^2}{2\sigma^2}} \tag{26}$$

The syntax of the macros is

```
\psGauss [Options] {xStart}{xEnd}
\psGaussI [Options] {xStart}{xEnd}
```

where the only new parameter are `sigma=<value>+` and `mue=<value>+` for the horizontal shift, which can also be set in the usual way with `\psset`. It is significant only for the `\psGauss` and `\psGaussI` macro. The default is `sigma=0.5` and `mue=0`. The integral is caclulated wuth the Simson algorithm and has one special option, called `Simpson`, which defines the number of intervals per step and is predefined with 5.




```
\psset{yunit=4cm,xunit=3}
\begin{pspicture}(-2,-0.2)(2,1.4)
% \psgrid[griddots=10,gridlabels=0pt, subgriddiv=0]
  \psaxes[Dy=0.25]{->}(0,0)(-2,0)(2,1.25)
  \uput[-90](6,0){x}\uput[0](0,1){y}
  \rput[lb](1,0.75){\textcolor{red}{$\sigma =0.5$}}
  \rput[lb](1,0.5){\textcolor{blue}{$\sigma =1$}}
  \rput[lb](-2,0.5){$f(x)=\dfrac{1}{\sigma\sqrt{2\pi}}\,e^{-\dfrac{(x-\mu)^2}{2\sigma{}^2}}$}
  \psGauss[linecolor=red, linewidth=2pt]{-1.75}{1.75}%
  \psGaussI[linewidth=1pt]{-2}{2}%
  \psGauss[linecolor=cyan, mue=0.5, linewidth=2pt]{-1.75}{1.75}%
  \psGauss[sigma=1, linecolor=blue, linewidth=2pt]{-1.75}{1.75}
\end{pspicture}
```

9.2 Binomial distribution

The following five macros plot binomial probability mass function `\psBinomial` and `\psBinomialC` in curve style, the normalized one is `\psBinomialN`. The cumulative distribution function F `\psBinomialF` and the complement of the cumulative distribution function $(1 - F)$ `\psBinomialFS` The vertical range for the plots is the y-Intervall $[0; 1]$. Rescaling other values can be done by setting the `yunit` option to any other value.

The binomial distribution `\psBinomial` gives the discrete probability distribution $P_p(n|N)$ n successes out of N Bernoulli trials (where the result of each Bernoulli trial is true with probability p and false with probability $q = 1 - p$). The binomial distribution is therefore given by

$$P_p(n|N) = \binom{N}{n} p^n q^{N-n} \tag{27}$$

$$= \frac{N!}{n!(N-n)!} p^n (1-p)^{N-n}, \tag{28}$$

where $(N; n)$ is a binomial coefficient and P the probability.

The syntax is:

```
\psBinomial [Options] {N}{probability p}
\psBinomial [Options] {m,N}{probability p}
\psBinomial [Options] {m,n,N}{probability p}
\psBinomialC [Options] {N}{probability p}
\psBinomialN [Options] {N}{probability p}
\psBinomialF [Options] {N}{probability p}
\psBinomialF [Options] {m,N}{probability p}
\psBinomialF [Options] {m,n,N}{probability p}
\psBinomialFS [Options] {N}{probability p}
\psBinomialFS [Options] {m,N}{probability p}
\psBinomialFS [Options] {m,n,N}{probability p}
```

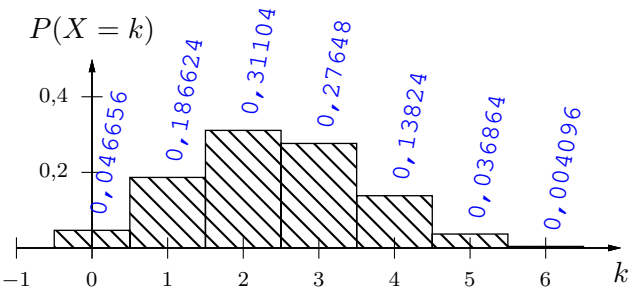
- with one argument N the sequence $0 \ldots N$ is calculated and plotted
- with two arguments m, N the sequence $0 \ldots N$ is calculated and the sequence $m \ldots N$ is plotted
- with three arguments m, n, N the sequence $0 \ldots N$ is calculated and the sequence $m \ldots n$ is plotted

Now `\psBinomial`, `\psBinomialF` and `\psBinomialFS` uses a new code, so the old restriction in using the value for N (old: $N < 100$) is no longer valid. A new limit vor N is not searched and it's not found. The valid options for the macros are `markZeros` to draw rectangles instead of a continous line and `printValue` for printing the y-values in the color LabelColor = color on top of the lines in distance labelsep and xlabelsep, rotated by labelangle = α. For this option all other options from section 1 for the macro `\psPrintValue` are valid, too. [6] Important is the keyword `valuewidth` which is preset to 15. If your value has more characters when converting into a string, it will not be printed or cause an GhostScript error.

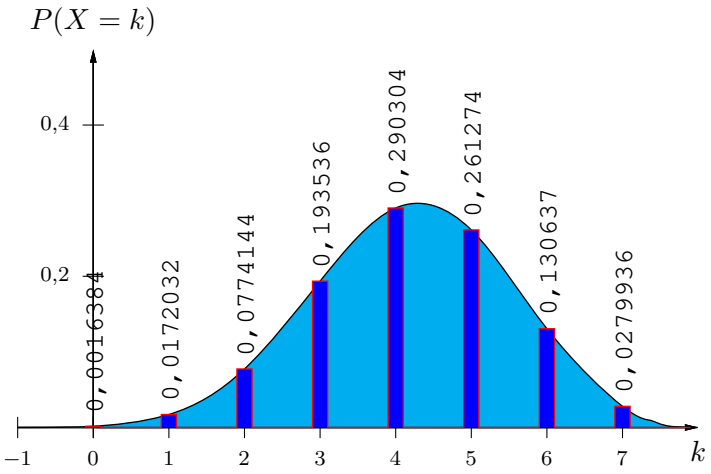
Special options are

- `barwidth`, which is a factor (no dimension) and set by default to 1. This option is not valid for the macro `\psBinomialN`!

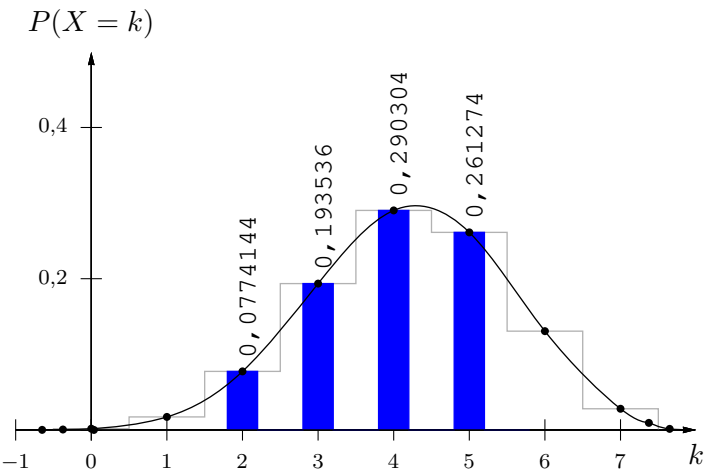
- `alternateColors` is a new fillstyle, so the colors alternates between `fillcolorA` and `fillcolorB`, only valid for `\psBinomial`.
- `fillcolorA` alternate color one.
- `fillcolorB` alternate color two.
- `labelangle` is the rotation of the printed values, default is 90°
- `xlabelsep` is the x-separation of the printed values, default is 0 (no dimension)
- `labelsep` is the y-separation of the printed values, default is 0.2 (no dimension)
- `LabelColor` is the color of the printed values, default is black
- `PrintVLimit` is the value limit for the printed values, default is $1e-64$, smaller values are not printed.
- `Switch2Log` is the value for N where the new calculation is used, default is 80.
- `LineEnding` this boolean is only valid for the macros `\psBinomialF` and `\psBinomialFS`, default is true. Draws circles at the end of the lines
- `VLines` this option is only valid for the macros `\psBinomialF` and `\psBinomialFS`, default is false. Draws the vertical lines dashed.
- `rightEnd`, this option is only valid for the macros `\psBinomialF` and `\psBinomialFS` when LineEnding=true (default) and $n = N$, default is 2
- `leftEnd`, this option is only valid for the macros `\psBinomialF` and `\psBinomialFS` when LineEnding=true (default) and $m = 0$, default is 1
- `radiusout`, this option is only valid for the macros `\psBinomialF` and `\psBinomialFS` when LineEnding=true (default) for the outer radius of the both dots left and right, default is 2
- `radiusinL`, this option is only valid for the macros `\psBinomialF` and `\psBinomialFS` when LineEnding=true (default) for the inner radius of the left dot, default is 0
- `radiusinR`, this option is only valid for the macros `\psBinomialF` and `\psBinomialFS` when LineEnding=true (default) for the inner radius of the right dot, default is 1.5
- `LineEndColorL` this option is only valid for the macros `\psBinomialF` and `\psBinomialFS` when LineEnding=true (default) for the color of the left dot, default is green
- `LineEndColorR` this option is only valid for the macros `\psBinomialF` and `\psBinomialFS` when LineEnding=true (default) for the inner radius of the right dot, default is red
- `LeftClipX` gives the left end of the clipping area for `\psBinomialC`, default is -1.
- `RightClipX` gives the distance to N for the right end of the clipping area for `\psBinomialC`, default is 1.



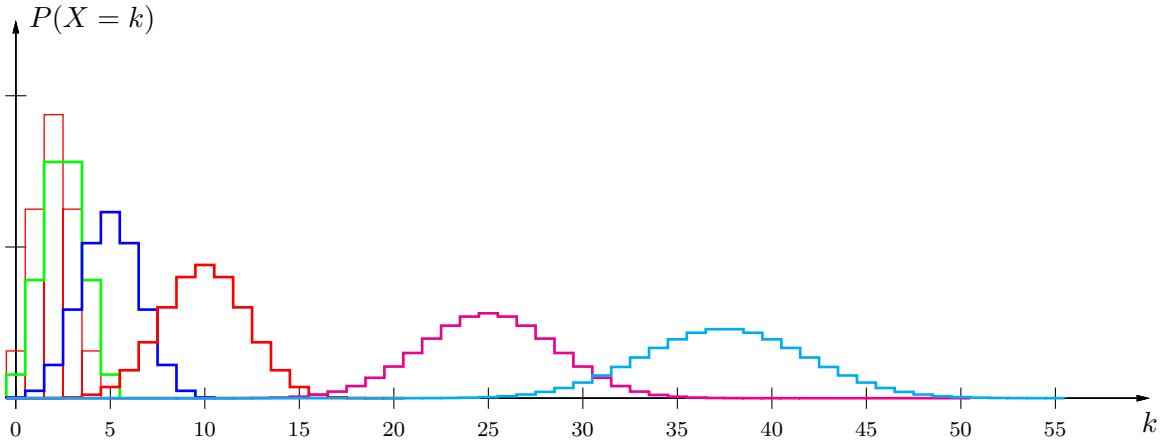

```
\psset{xunit=1cm,yunit=5cm}%
\begin{pspicture}(-1,-0.15)(7,0.6)%
\psaxes[Dy=0.2,dy=0.2\psyunit]{->}(0,0)(-1,0)(7,0.5)
\uput[-90](7,0){$k$} \uput[90](0,0.5){$P(X=k)$}
\psBinomial[markZeros,printValue,fillstyle=vlines,
labelangle=80,LabelColor=blue]{6}{0.4}
\end{pspicture}
```



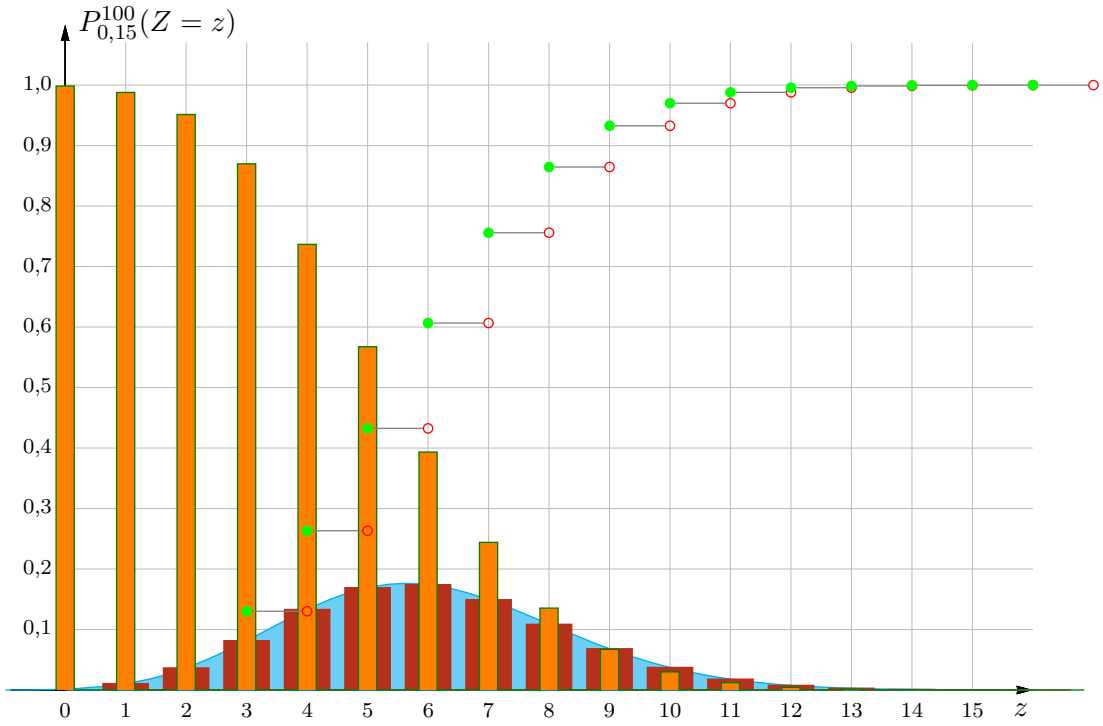

```
\psset{xunit=1cm,yunit=10cm}%
\begin{pspicture}(-1,-0.05)(8,0.6)%
\psaxes[Dy=0.2,dy=0.2\psyunit]{->}(0,0)(-1,0)(8,0.5)
\uput[-90](8,0){$k$} \uput[90](0,0.5){$P(X=k)$}
\psBinomialC[fillstyle=solid,opacity=0.5,fillcolor=cyan]{7}{0.6}
\psBinomial[linecolor=red,markZeros,printValue,fillstyle=solid,
   fillcolor=blue,barwidth=0.2,xlabelsep=-0.05]{7}{0.6}
\end{pspicture}
```



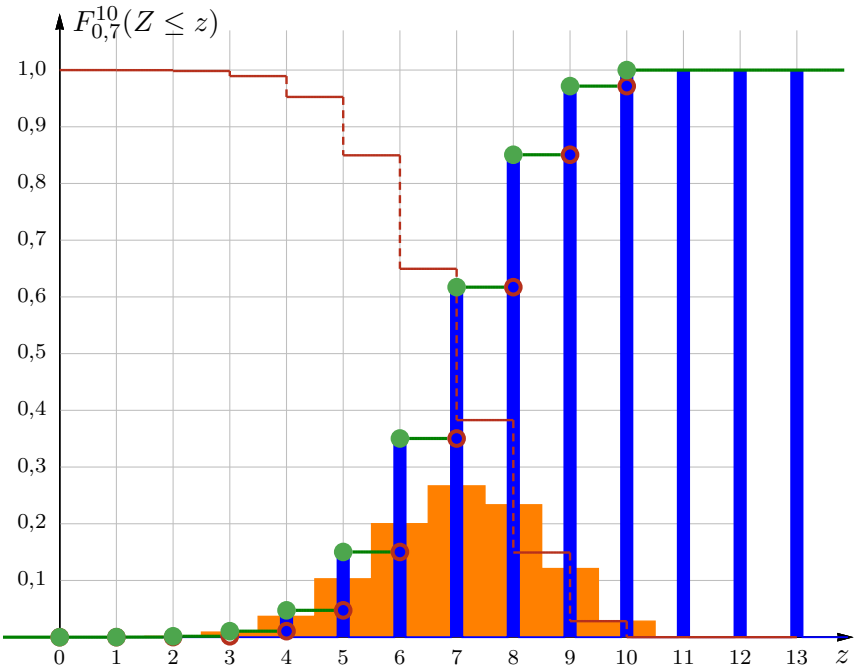

```
\psset{xunit=1cm,yunit=10cm}%
\begin{pspicture}(-1,-0.05)(8,0.6)%
\psaxes[Dy=0.2,dy=0.2\psyunit]{->}(0,0)(-1,0)(8,0.5)
\uput[-90](8,0){$k$} \uput[90](0,0.5){$P(X=k)$}
\psBinomial[linecolor=black!30]{0,7}{0.6}
\psBinomial[linecolor=blue,markZeros,printValue,fillstyle=solid,
   fillcolor=blue,barwidth=0.4]{2,5,7}{0.6}
\psBinomialC[showpoints=true]{7}{0.6}
\end{pspicture}
```



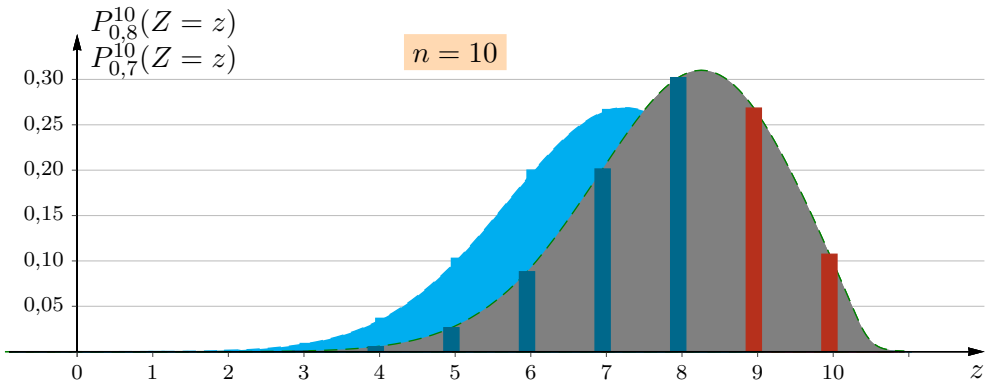

```
\psset{xunit=0.25cm,yunit=10cm}
\begin{pspicture*}(-1,-0.05)(61,0.52)
\psaxes[Dx=5,dx=5\psxunit,Dy=0.2,dy=0.2\psyunit]{->}(60,0.5)
\uput[-90](60,0){$k$} \uput[0](0,0.5){$P(X=k)$}
\psBinomial[markZeros,linecolor=red]{4}{.5}
\psset{linewidth=1pt}
\psBinomial[linecolor=green]{5}{.5} \psBinomial[linecolor=blue]{10}{.5}
\psBinomial[linecolor=red]{20}{.5} \psBinomial[linecolor=magenta]{50}{.5}
\psBinomial[linecolor=cyan]{0,55,75}{.5}
\end{pspicture*}
```



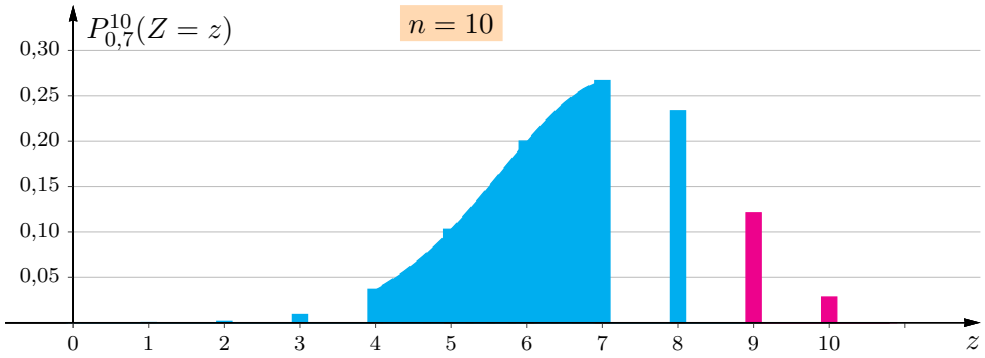

```
\psset{xunit=0.8cm,yunit=8cm}%
\begin{pspicture*}[showgrid=false](-1.5,-0.1)(16,1.2)%
\psset{arrowscale=1.3,arrowinset=0.05,arrowlength=1.9,comma}%
\psaxes[labelFontSize=\scriptstyle,xticksize=0 1.07,yticksize=0 16,tickcolor=gray!50,
      Dy=0.1,dy=0.1,Dx=1,dx=1,Ox=0]{->}(0,0)(-0.9,0)(16,1.1)
\uput[-90](15.8,0){$z$}\uput[0](0,1.1){$P_{0,15}^{100}(Z=z)$}
\psBinomialC[linecolor=cyan,fillstyle=solid,fillcolor=cyan!50,opacity=0.4]{40}{0.15}%
\psBinomial[markZeros,linecolor=BrickRed,fillstyle=solid,fillcolor=BrickRed,barwidth=0.75,opacity
    =0.6]{1,16,40}{0.15}%
\psBinomialFS[markZeros,linecolor=Green,fillstyle=solid,fillcolor=orange,barwidth=0.3,opacity
    =0.6]{0,16,40}{0.15}%
\psBinomialF[linecolor=gray,fillstyle=solid,fillcolor=yellow,barwidth=0.4,opacity=0.5]{3,16,40}{0.15}
\end{pspicture*}
```



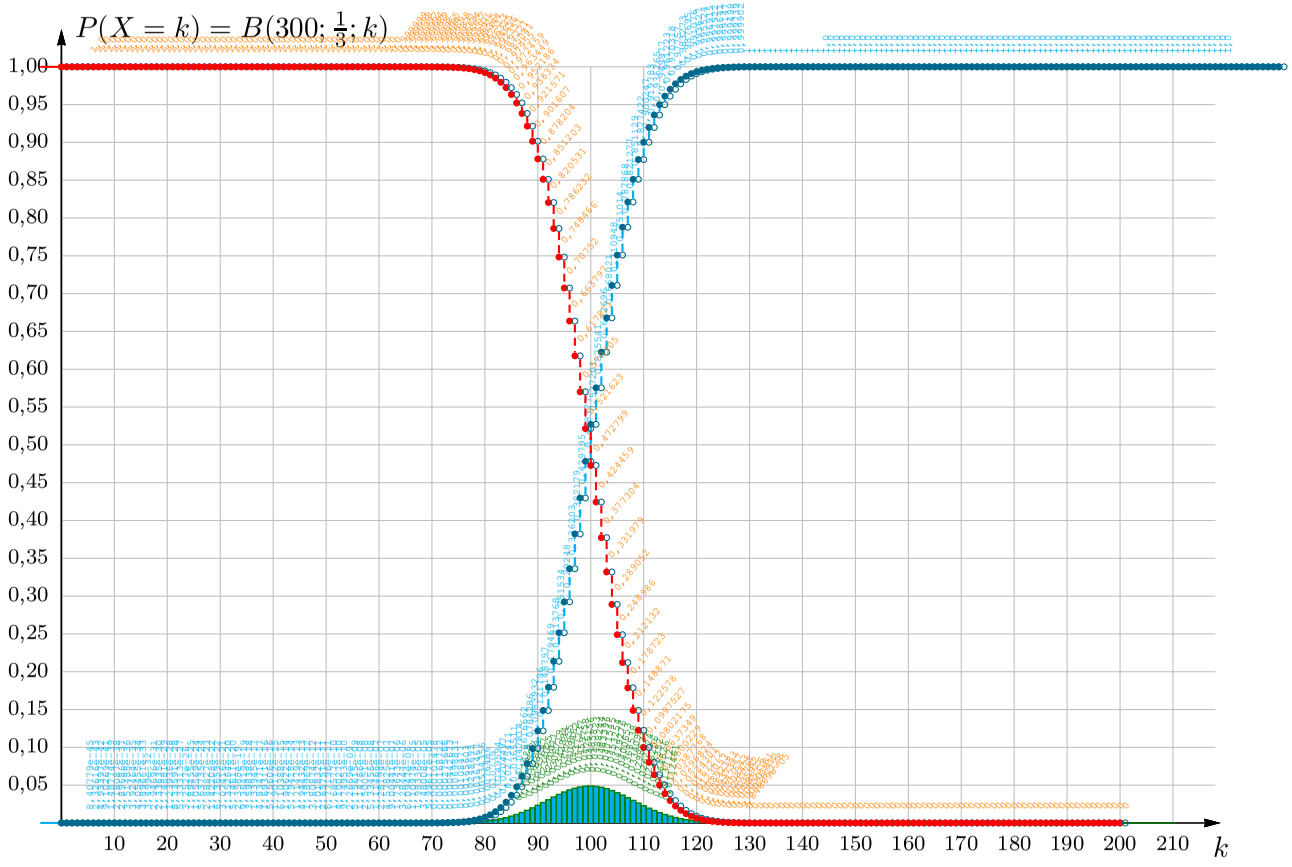

```
\psset{xunit=0.75cm,yunit=7.5cm}%
\begin{pspicture*}[showgrid=false](-1.3,-0.067)(14.67,1.13)%
\psset{arrowscale=1.3,arrowinset=0.05,arrowlength=1.9,comma}
\psaxes[labelFontSize=\scriptstyle,xticksize=0 1.07,yticksize=0 12,tickcolor=gray!50,Dy=0.1,dy=0.1,Dx
    =1,dx=1,Ox=0]{->}(0,0)(-0.9,0)(14,1.1)
\uput[-90](13.8,0){$z$} \uput[0](0,1.08){$F_{0,7}^{10}(Z\leq z)$}
\psBinomial[markZeros,linecolor=orange,fillstyle=solid,fillcolor=orange,barwidth=1,opacity
    =0.5]{0,10,10}{0.7}
\psBinomialF[markZeros,linecolor=blue,linewidth=0.7pt,barwidth=0.2,
opacity=0.5,fillstyle=solid,fillcolor=blue,valuewidth=15]{0,13,10}{0.7}
\psBinomialFS[LineEnding=false,linecolor=BrickRed,linewidth=0.9pt,VLines=true]{0,10,10}{0.7}
\psBinomialF[linecolor=Green,printValue=false,linewidth=1.2pt,LineEndColorR=BrickRed,LineEndColorL=
    Green!70,
radiusout=3.5,radiusinL=0,radiusinR=2,LineEnding=true,leftEnd=1,rightEnd=3]{0,10,10}{0.7}
\end{pspicture*}
```




```
\begin{pspicture}[showgrid=false](-.75,-1.8)(13.2,4.7)%
{\psset{xunit=1cm,yunit=12cm}%
\psset{plotpoints=500,arrowscale=1.3,arrowinset=0.05,arrowlength=1.9,comma}
\psaxes[labelFontSize=\scriptstyle,xticksize=0 0,yticksize=0 12,tickcolor=gray!50,Dy=0.05,dy=0.05,Dx
    =1,dx=1,Ox=0]{-}(0,0)(-0.9,0)(10.8,0.34)
\uput[-90](11.9,0){$z$} \uput[0](0,0.36){$P_{0,8}^{10}(Z=z)$}\uput[0](0,0.32){$P_{0,7}^{10}(Z=z)$}
\rput(-0.05,0){%
\psBinomialC[linecolor=Green,fillstyle=solid,fillcolor=gray,opacity=0.25,plotstyle=curve,linestyle=
    dashed]{10}{0.8}}
\rput(0.05,0){%
\psBinomialC[linecolor=cyan,fillstyle=solid,fillcolor=cyan,opacity=0.25,plotstyle=curve,linestyle=
    dashed]{10}{0.7}%
\psBinomial[markZeros,linecolor=cyan,fillstyle=solid,fillcolor=cyan,barwidth=0.2,opacity
    =0.85]{0,8,10}{0.7}%,printValue
\psBinomial[markZeros,linecolor=magenta,fillstyle=solid,fillcolor=magenta,barwidth=0.2,opacity
    =0.85]{9,10,10}{0.7}
}
\rput(-0.05,0){%
\psBinomialC[linecolor=Green,fillstyle=solid,fillcolor=gray,opacity=0.25,plotstyle=curve,linestyle=
    dashed]{10}{0.8}
\psBinomial[markZeros,linecolor=DeepSkyBlue4,fillstyle=solid,fillcolor=DeepSkyBlue4,barwidth=0.2,
    opacity=0.85]{0,8,10}{0.8}%,printValue
\psBinomial[markZeros,linecolor=BrickRed,fillstyle=solid,fillcolor=BrickRed,barwidth=0.2,opacity
    =0.85]{9,10,10}{0.8}
}
\psaxes[labels=none,xticksize=-2pt 0,yticksize=-2pt 0,tickcolor=black!70,Dy=0.05,dy=0.05\psyunit,Dx=1,
    dx=1\psxunit,Ox=0]{->}(0,0)(-0.9,0)(12,0.35)
\rput(5,0.33){\psframebox[fillstyle=solid,fillcolor=orange!30,linestyle=none]{$n=10$}}
}
\end{pspicture}
```



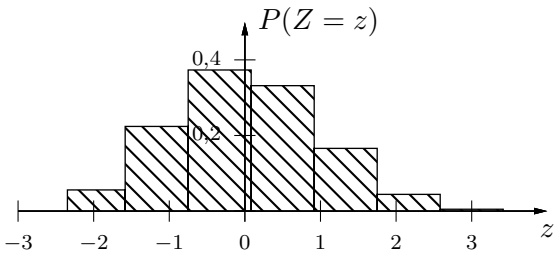

```
\begin{pspicture}[showgrid=false](-.75,-1.8)(13.2,4.7)%
{\psset{xunit=1cm,yunit=12cm}%
\psset{plotpoints=500,arrowscale=1.3,arrowinset=0.05,arrowlength=1.9,comma}
\psaxes[labelFontSize=\scriptstyle,xticksize=0 0,yticksize=0 12,tickcolor=gray!50,Dy=0.05,dy=0.05,Dx
    =1,dx=1,Ox=0]{-}(0,0)(-0.9,0)(10.8,0.34)
\uput[-90](11.9,0){$z$} \uput[0](0,0.32){$P_{0,7}^{10}(Z=z)$}
\psBinomialC[linecolor=cyan,fillstyle=solid,fillcolor=cyan,opacity=0.25,plotstyle=curve,linestyle=
    dashed,LeftClipX=4,RightClipX=-3]{10}{0.7}%
\psBinomial[markZeros,linecolor=cyan,fillstyle=solid,fillcolor=cyan,barwidth=0.2,opacity
    =0.85]{0,8,10}{0.7}%,printValue
\psBinomial[markZeros,linecolor=magenta,fillstyle=solid,fillcolor=magenta,barwidth=0.2,opacity
    =0.85]{9,10,10}{0.7}
\psaxes[labels=none,xticksize=-2pt 0,yticksize=-2pt 0,tickcolor=black!70,Dy=0.05,dy=0.05\psyunit,Dx=1,
    dx=1\psxunit,Ox=0]{->}(0,0)(-0.9,0)(12,0.35)
\rput(5,0.33){\psframebox[fillstyle=solid,fillcolor=orange!30,linestyle=none]{$n=10$}}
}
\end{pspicture}
```



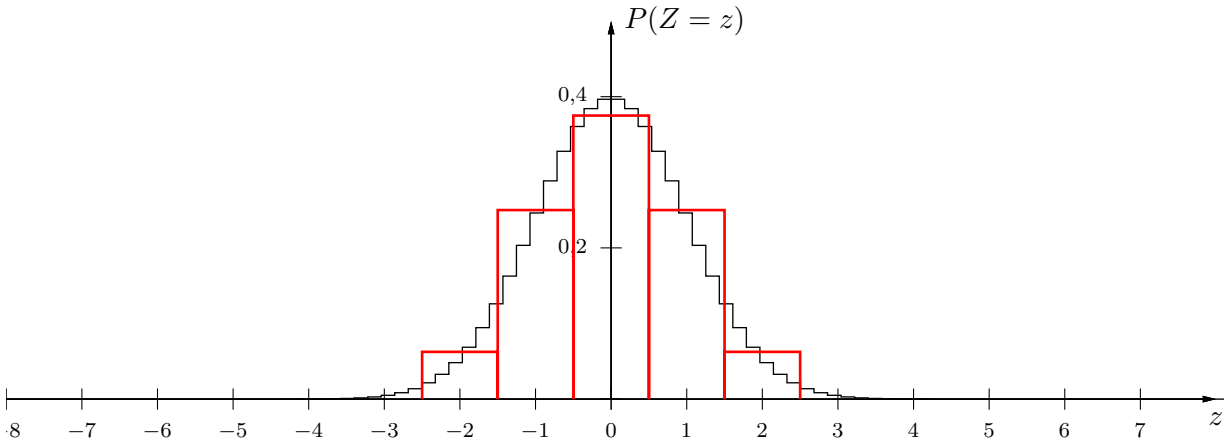

```
{\psset{xunit=0.07cm,yunit=10cm}%
\begin{pspicture}[showgrid=false](-6,-0.1)(220,1.1)%
\psset{plotpoints=500,arrowscale=1.3,arrowinset=0.05,arrowlength=1.9,comma}
\psaxes[labelFontSize=\scriptstyle,xticksize=0 1,yticksize=0 218,tickcolor=gray!50,Dy=0.05,dy=0.05,Dx
    =10,dx=10,showorigin=false]{->}(0,0)(219,1.05)
\uput[-90](219,0){$k$} \uput[0](0,1.05){$P(X=k)=B(300;\frac{1}{3};k)$}
\psBinomial[linecolor=Green,fillstyle=solid,fillcolor=cyan,opacity=0.5,printValue=true,markZeros,
    fontscale=4,xlabelsep=-0.175,LabelColor=Green,labelangle=80,PrintVLimit=0.01]{1,210,300}{1 3 div}%
    ,printValue
\psBinomialF[radiusout=1.3,radiusinR=0.9,linecolor=cyan,leftEnd=4,rightEnd=5,linewidth=0.8pt,
    LineEndColorR=DeepSkyBlue4,LineEndColorL=DeepSkyBlue4,VLines,printValue,fontscale=4,LabelColor=
    cyan]{0,230,300}{1 3 div}
\psBinomialFS[radiusout=1.3,radiusinR=0.9,linecolor=red,leftEnd=4,rightEnd=5,linewidth=0.8pt,
    LineEndColorR=DeepSkyBlue4,LineEndColorL=red,VLines,printValue,fontscale=4,labelangle=50,
    LabelColor=orange]{0,200,300}{1 3 div}
\end{pspicture}}
```

The default binomial distribution has the mean of $\mu = E(X) = N \cdot p$ and a variant of $\sigma^2 = \mu \cdot (1-p)$. The normalized distribution has a mean of 0. Instead of $P(X = k)$ we use $P(Z = z)$ with $Z = \frac{X - E(X)}{\sigma(X)}$ and $P \leftarrow P \cdot \sigma$. The macros use the recursive definition of the binomial distribution:

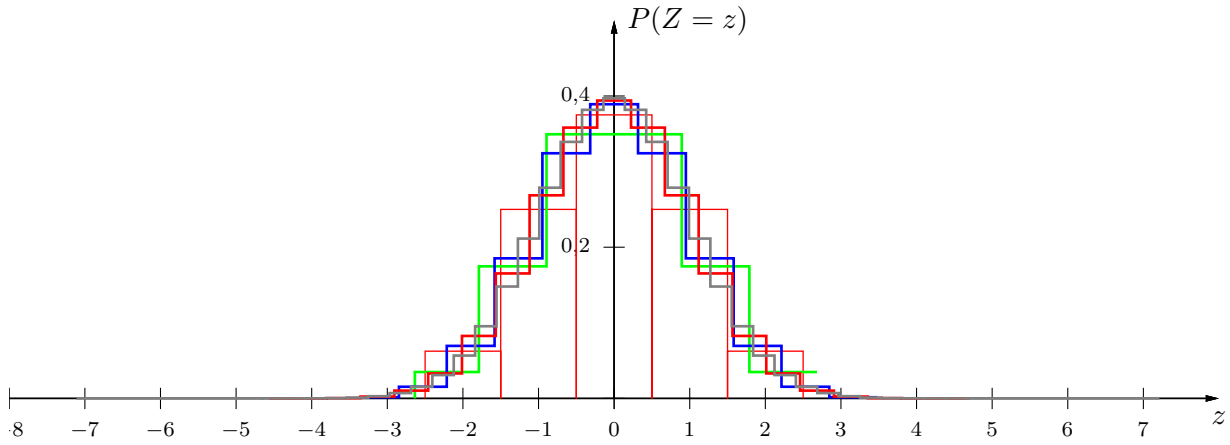
$$P(k) = P(k-1) \cdot \frac{N-k+1}{k} \cdot \frac{p}{1-p} \tag{29}$$




```
\psset{xunit=1cm,yunit=5cm}%
\begin{pspicture}(-3,-0.15)(4,0.55)%
\psaxes[Dy=0.2,dy=0.2\psyunit]{->}(0,0)(-3,0)(4,0.5)
\uput[-90](4,0){$z$} \uput[0](0,0.5){$P(Z=z)$}
\psBinomialN[markZeros,fillstyle=vlines]{6}{0.4}
\end{pspicture}
```

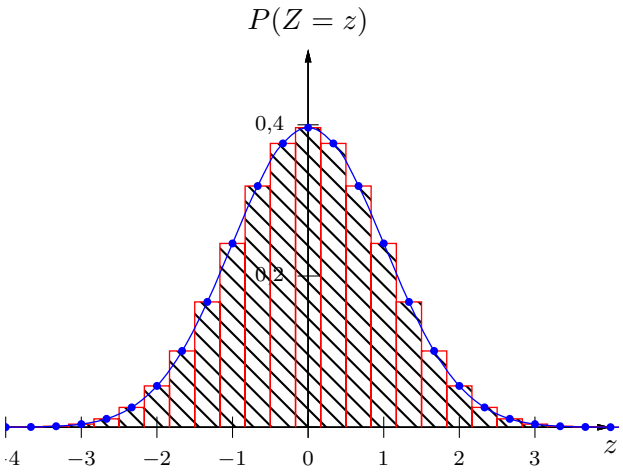



```
\psset{yunit=10}
\begin{pspicture*}(-8,-0.07)(8.1,0.55)
\psaxes[Dy=0.2,dy=0.2\psyunit]{->}(0,0)(-8,0)(8,0.5)
\uput[-90](8,0){$z$} \uput[0](0,0.5){$P(Z=z)$}
\psBinomialN{125}{.5}
\psBinomialN[markZeros,linewidth=1pt,linecolor=red]{4}{.5}
\end{pspicture*}
```

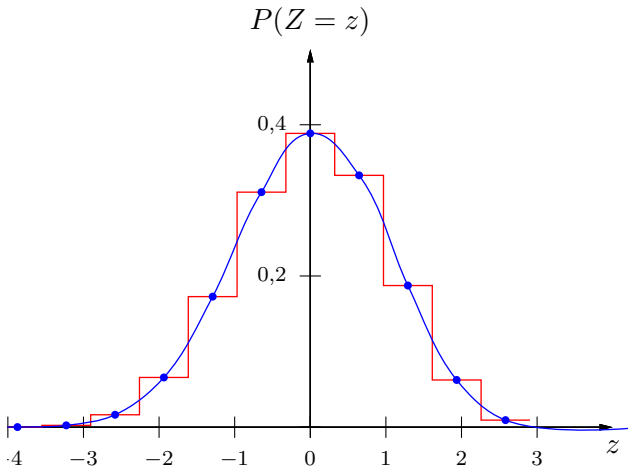



```
\psset{yunit=10}
\begin{pspicture*}(-8,-0.07)(8.1,0.52)
\psaxes[Dy=0.2,dy=0.2\psyunit]{->}(0,0)(-8,0)(8,0.5)
\uput[-90](8,0){$z$} \uput[0](0,0.5){$P(Z=z)$}
\psBinomialN[markZeros,linecolor=red]{4}{.5}
\psset{linewidth=1pt}
\psBinomialN[linecolor=green]{5}{.5}\psBinomialN[linecolor=blue]{10}{.5}
\psBinomialN[linecolor=red]{20}{.5} \psBinomialN[linecolor=gray]{50}{.5}
\end{pspicture*}
```

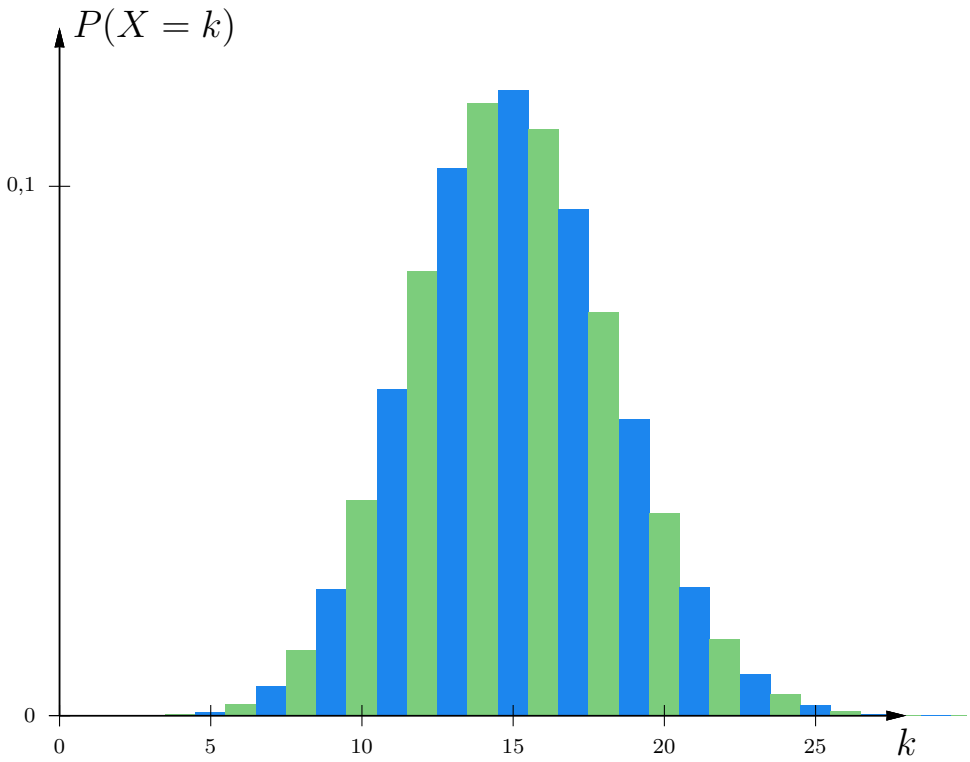
For the normalized distribution the plotstyle can be set to `curve` (`plotstyle=curve`), then the binomial distribution looks like a normal distribution. This option is only valid for `\psBinomialN`. The option `showpoints` is valid if `curve` was chosen.




```
\psset{xunit=1cm,yunit=10cm}%
\begin{pspicture*}(-4,-0.06)(4.1,0.57)%
\psaxes[Dy=0.2,dy=0.2\psyunit]{->}(0,0)(-4,0)(4,0.5)%
\uput[-90](4,0){$z$} \uput[90](0,0.5){$P(Z=z)$}%
\psBinomialN[linecolor=red,fillstyle=vlines,showpoints=true,markZeros]{36}{0.5}%
\psBinomialN[linecolor=blue,showpoints=true,plotstyle=curve]{36}{0.5}%
\end{pspicture*}
```




```
\psset{xunit=1cm,yunit=10cm}%
\begin{pspicture*}(-4,-0.06)(4.2,0.57)%
\psaxes[Dy=0.2,dy=0.2\psyunit]{->}(0,0)(-4,0)(4,0.5)%
\uput[-90](4,0){$z$} \uput[90](0,0.5){$P(Z=z)$}%
\psBinomialN[linecolor=red]{10}{0.6}%
\psBinomialN[linecolor=blue,showpoints=true,plotstyle=curve]{10}{0.6}%
\end{pspicture*}
```




```
\definecolor{A1}{RGB}{28, 134, 238}
\definecolor{A2}{RGB}{124, 205, 124}
\psset{xunit=4mm,yunit=70cm,arrowscale=1.5}%
\begin{pspicture*}(-2,-0.01)(30,0.25)
\psBinomial[fillstyle=alternateColors,
         fillcolorA=A1,fillcolorB=A2,
         markZeros]{60}{0.25}
\psaxes[Dx=5,dx=5\psxunit,Dy=0.1,dy=0.1\psyunit,
      arrows=D>]{->}(28,0.13)[\Large$k$,-90][\Large$P(X=k)$,0]
\end{pspicture*}%
```

9.3 Poisson distribution

Given a Poisson process[2], the probability of obtaining exactly n successes in N trials is given by the limit of a binomial distribution (see Section 9.2)

$$P_p(n|N) = \frac{N!}{n!(N-n)!} \cdot p^n (1-p)^{N-n} \tag{30}$$

Viewing the distribution as a function of the expected number of successes;

$$\lambda = N \cdot p \tag{31}$$

instead of the sample size N for fixed p, equation (2) then becomes (30);

$$P_{\frac{\lambda}{n}}(n|N) = \frac{N!}{n!(N-n)!} \frac{\lambda}{N}^n \frac{1-\lambda}{N}^{N-n} \tag{32}$$

Viewing the distribution as a function of the expected number of successes;

$$P_\lambda(X=k) = \frac{\lambda^k}{k!} e^{-\lambda}$$

. Letting the sample size become large ($N \to \infty$), the distribution then approaches (with $p = \frac{\lambda}{n}$):

$$\lim_{n\to\infty} P(X=k) = \lim_{n\to\infty} \frac{n!}{(n-k)!\,k!} \left(\frac{\lambda}{n}\right)^k \left(1-\frac{\lambda}{n}\right)^{n-k} \tag{33}$$

$$= \lim_{n\to\infty} \left(\frac{(n-k)! \cdot (n-k+1)\cdots(n-2)(n-1)n}{(n-k)!\,n^k}\right) \cdot \tag{34}$$

$$\left(\frac{\lambda^k}{k!}\right) \left(1-\frac{\lambda}{n}\right)^n \left(1-\frac{\lambda}{n}\right)^{-k} \tag{35}$$

$$= \frac{\lambda^k}{k!} \cdot \lim_{n\to\infty} \underbrace{\left(\frac{n}{n} \cdot \frac{n-1}{n} \cdot \frac{n-2}{n} \cdot \ldots \cdot \frac{n-k+1}{n}\right)}_{\to 1} \cdot \tag{36}$$

$$\underbrace{\left(1-\frac{\lambda}{n}\right)^n}_{\to e^{-\lambda}} \underbrace{\left(1-\frac{\lambda}{n}\right)^{-k}}_{\to 1} \tag{37}$$

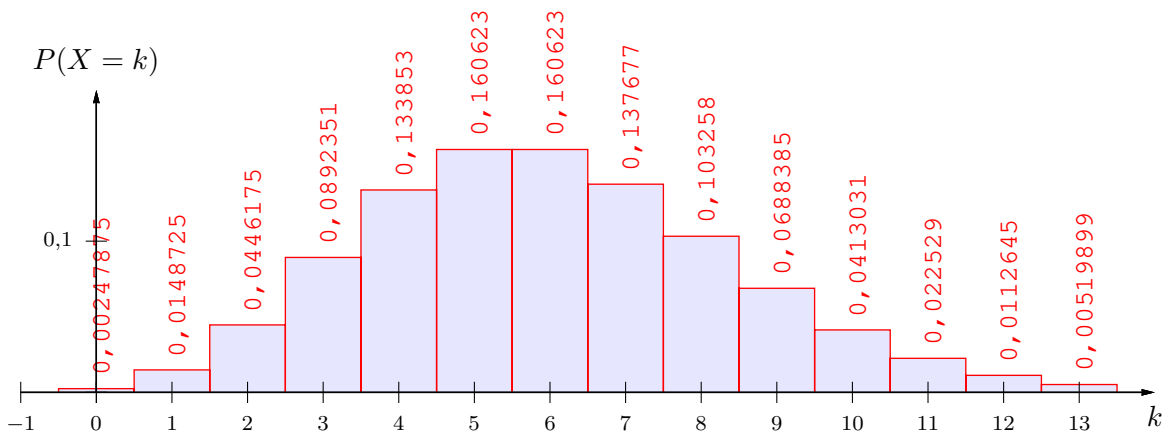
$$= \lambda^k e^{\frac{-\lambda}{k!}} \tag{38}$$

which is known as the Poisson distribution and has the follwing syntax:

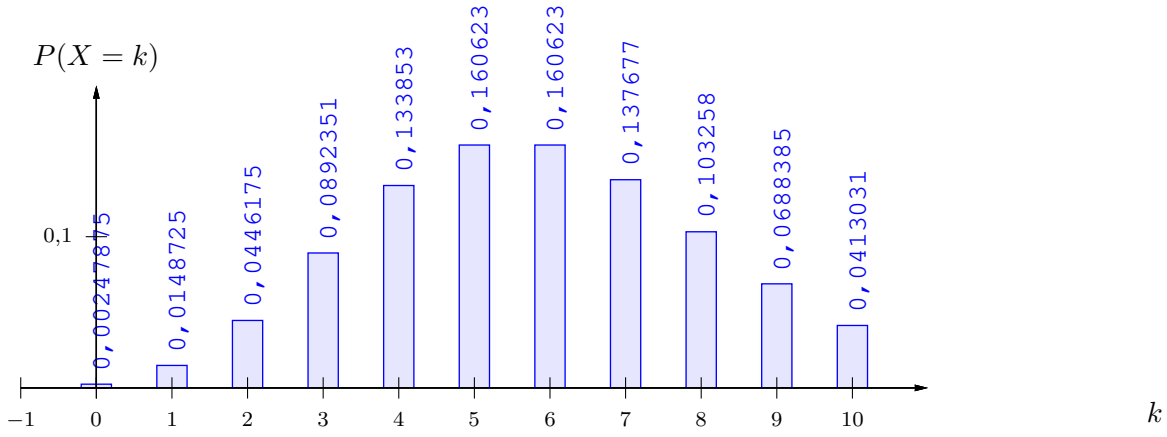
```
\psPoisson [Options] {N}{lambda}
\psPoisson [Options] {M,N}{lambda}
```

in which `M` is an optional argument with a default of 0.

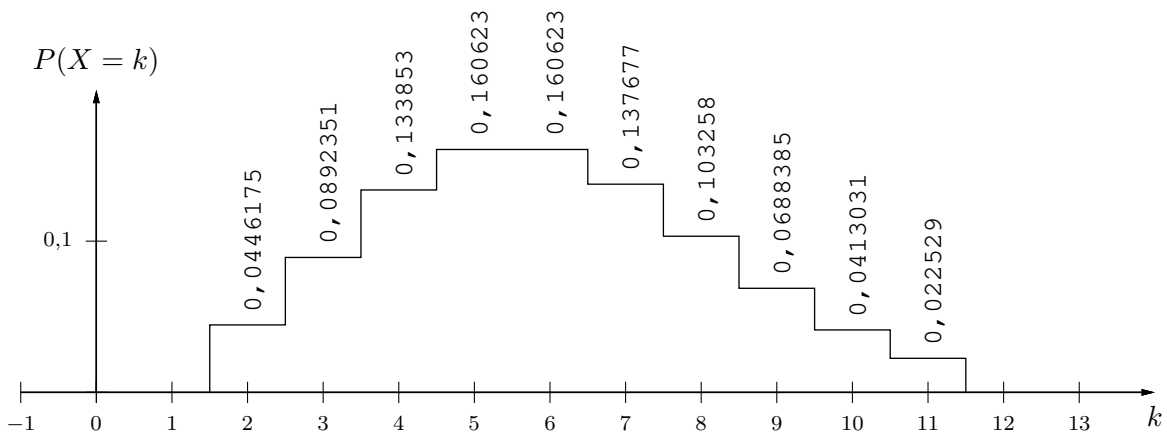
2 http://mathworld.wolfram.com/PoissonProcess.html




```
\psset{xunit=1cm,yunit=20cm}%
\begin{pspicture}(-1,-0.05)(14,0.25)%
\uput[-90](14,0){$k$} \uput[90](0,0.2){$P(X=k)$}
\psPoisson[linecolor=red,markZeros,fillstyle=solid,
   fillcolor=blue!10,printValue,valuewidth=20]{13}{6} % N lambda
\psaxes[Dy=0.1,dy=0.1\psyunit]{->}(0,0)(-1,0)(14,0.2)
\end{pspicture}
```




```
\psset{xunit=1cm,yunit=20cm}%
\begin{pspicture}(-1,-0.05)(14,0.25)%
\uput[-90](14,0){$k$} \uput[90](0,0.2){$P(X=k)$}
\psPoisson[linecolor=blue,markZeros,fillstyle=solid,barwidth=0.4,
   fillcolor=blue!10,printValue,valuewidth=20]{10}{6} % N lambda
\psaxes[Dy=0.1,dy=0.1\psyunit]{->}(0,0)(-1,0)(11,0.2)
\end{pspicture}
```




```
\psset{xunit=1cm,yunit=20cm}%
\begin{pspicture}(-1,-0.05)(14,0.25)%
\uput[-90](14,0){$k$} \uput[90](0,0.2){$P(X=k)$}
\psPoisson[printValue,valuewidth=20]{2,11}{6} % M,N lambda
\psaxes[Dy=0.1,dy=0.1\psyunit]{->}(0,0)(-1,0)(14,0.2)
\end{pspicture}
```

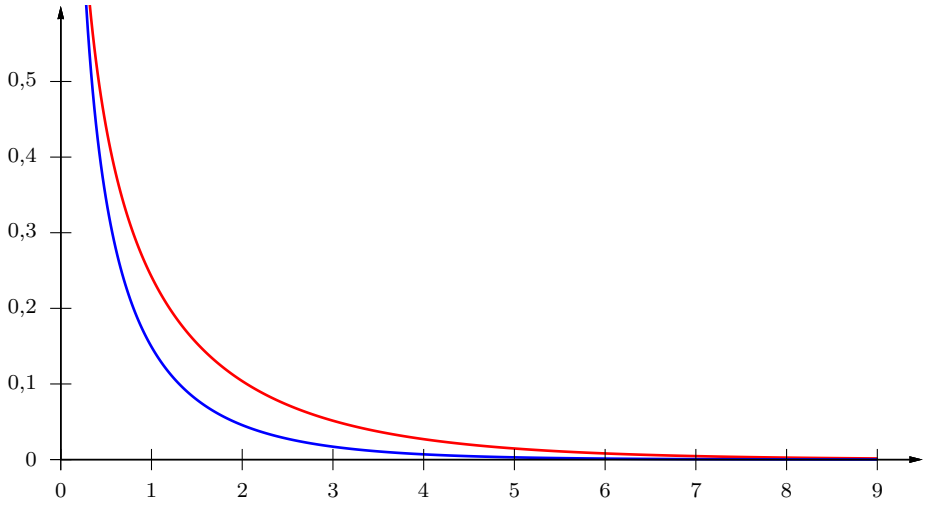
9.4 Gamma distribution

A gamma distribution is a general type of statistical distribution that is related to the beta distribution and arises naturally in processes for which the waiting times between Poisson distributed events are relevant. Gamma distributions have two free parameters, labeled α and β. It is defined as

$$f(x) = \frac{\beta(\beta x)^{\alpha-1}e^{-\beta x}}{\Gamma(\alpha)} \qquad \text{for } x > 0 \text{ and } \alpha,\, \beta > 0$$

and has the syntax

```
\psGammaDist[Options]{x0}{x1}
```



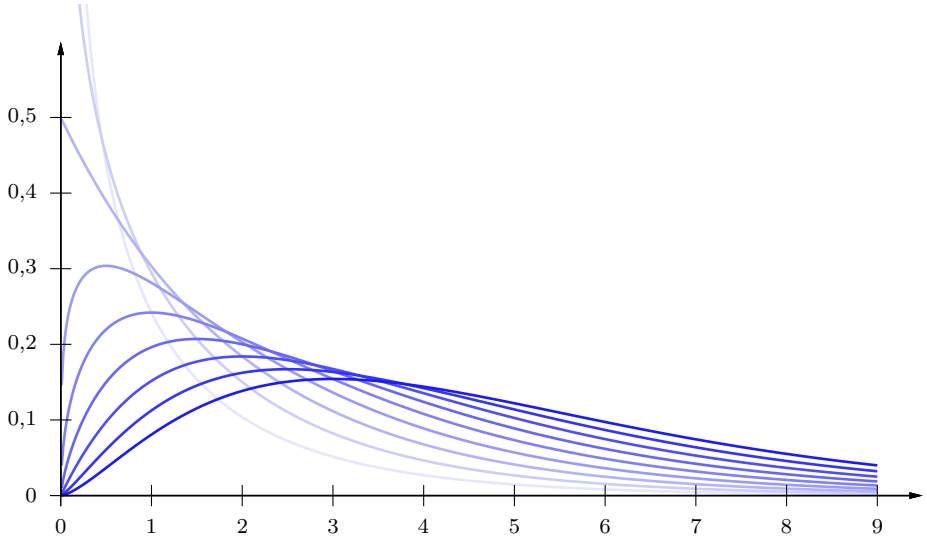

```
\psset{xunit=1.2cm,yunit=10cm,plotpoints=200}
\begin{pspicture*}(-0.75,-0.05)(9.5,0.6)
 \psGammaDist[linewidth=1pt,linecolor=red]{0.01}{9}
 \psGammaDist[linewidth=1pt,linecolor=blue,alpha=0.3,beta=0.7]{0.01}{9}
 \psaxes[Dy=0.1]{->}(0,0)(9.5,.6)
\end{pspicture*}
```

9.5 χ^2-distribution

The χ^2-distribution is a continuous probability distribution. It usually arises when a k-dimensional vector's orthogonal components are independent and each follow a standard normal distribution. The length of the vector will then have a χ^2-distribution.

The χ^2 with parameter ν is the same as a Gamma distribution with $\alpha = \nu/2$ and $\beta = 1/2$ and the syntax

```
\psChiIIDist [Options] {x0}{x1}
```




```
\psset{xunit=1.2cm,yunit=10cm,plotpoints=200}
\begin{pspicture*}(-0.75,-0.05)(9.5,.65)
 \multido{\rnue=0.5+0.5,\iblue=0+10}{10}{%
   \psChiIIDist[linewidth=1pt,linecolor=blue!\iblue,nue=\rnue]{0.01}{9}}
 \psaxes[Dy=0.1]{->}(0,0)(9.5,.6)
\end{pspicture*}
```

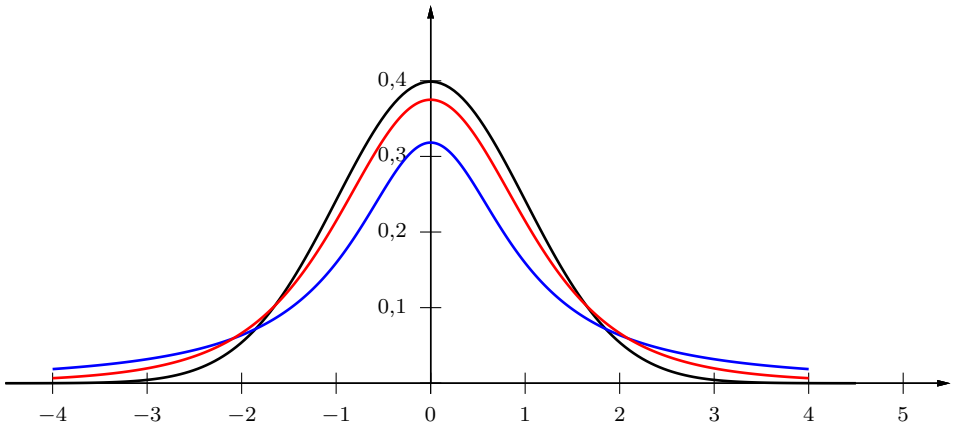
9.6 Student's t-distribution

A statistical distribution published by William Gosset in 1908 under his pseudonym "Student". The t-distribution with parameter ν has the density function

$$f(x) = \frac{1}{\sqrt{\nu\pi}} \cdot \frac{\Gamma[(\nu+1)/2]}{\Gamma(\nu/2)} \cdot \frac{1}{[1+(x^2/\nu)]^{(\nu+1)/2}} \qquad \text{for } -\infty < x < \infty \text{ and } \nu > 0$$

and the following syntax

```
\psTDist[Options]{x0}{x1}
```




```
\psset{xunit=1.25cm,yunit=10cm}
\begin{pspicture}(-6,-0.1)(6,.5)
 \psaxes[Dy=0.1]{->}(0,0)(-4.5,0)(5.5,0.5)
 \psset{linewidth=1pt,plotpoints=100}
 \psGauss[mue=0,sigma=1]{-4.5}{4.5}
 \psTDist[linecolor=blue]{-4}{4}
 \psTDist[linecolor=red,nue=4]{-4}{4}
\end{pspicture}
```

9.7 F-distribution

A continuous statistical distribution which arises in the testing of whether two observed samples have the same variance.

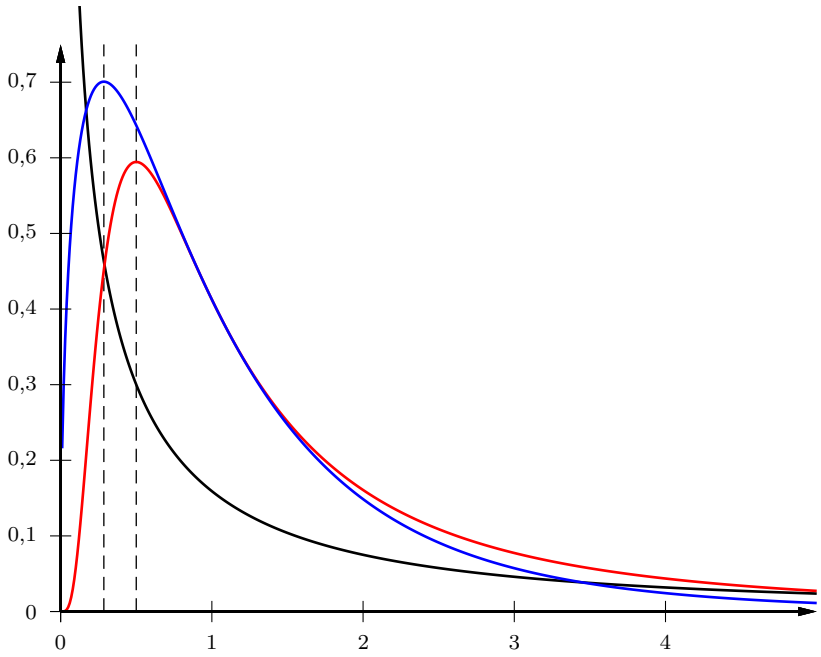
The F-distribution with parameters μ and ν has the probability function

$$f_{\mu,\nu}(x) = \frac{\Gamma[(\mu+\nu)/2]}{\Gamma(\mu/2)\Gamma(\nu/2)} \cdot (\mu/\nu)^{\mu/2} \frac{x^{(\mu/2)-1}}{[1+(\mu x/\nu)]^{(\mu+\nu)/2}} \quad \text{for } x > 0 \text{ and } \mu,\ \nu > 0$$

and the syntax

```
\psFDist[Options]{x0}{x1}
```

The default settings are $\mu = 1$ and $\nu = 1$.




```
\psset{xunit=2cm,yunit=10cm,plotpoints=100}
\begin{pspicture*}(-0.5,-0.07)(5.5,0.8)
 \psline[linestyle=dashed](0.5,0)(0.5,0.75)
 \psline[linestyle=dashed](! 2 7 div 0)(! 2 7 div 0.75)
 \psset{linewidth=1pt}
 \psFDist{0.1}{5}
 \psFDist[linecolor=red,nue=3,mue=12]{0.01}{5}
 \psFDist[linecolor=blue,nue=12,mue=3]{0.01}{5}
 \psaxes[Dy=0.1]{->}(0,0)(5,0.75)
\end{pspicture*}
```

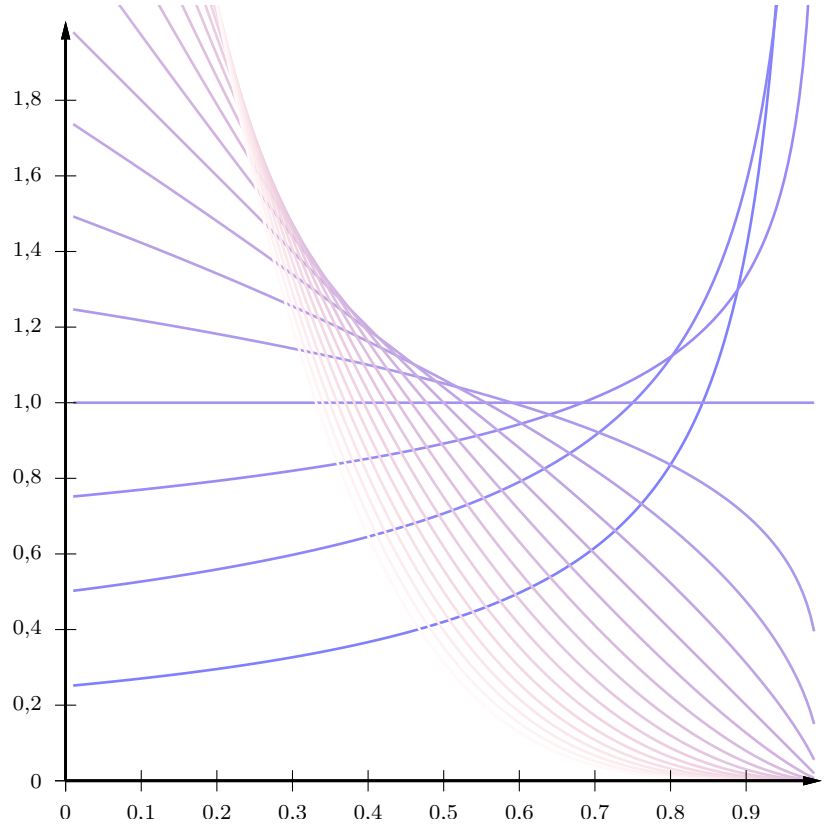
9.8 Beta distribution

A general type of statistical distribution which is related to the gamma distribution. Beta distributions have two free parameters, which are labeled according to one of two notational conventions. The usual definition calls these α and β, and the other uses $\beta' = \beta - 1$ and $\alpha' = \alpha - 1$. The beta distribution is used as a prior distribution for binomial proportions in Bayesian analysis. The domain is $[0, 1]$, and the probability function $P(x)$ is given by

$$P(x) = \frac{\Gamma(\alpha + \beta)}{\Gamma(\alpha)\Gamma(\beta)}(1 - x)^{\beta-1}x^{\alpha-1} \qquad \alpha, \beta > 0$$

and has the syntax (with a default setting of $\alpha = 1$ and $\beta = 1$):

```
\psBetaDist [Options] {x0}{x1}
```




```
\psset{xunit=10cm,yunit=5cm}
\begin{pspicture*}(-0.1,-0.1)(1.1,2.05)
 \psset{linewidth=1pt}
 \multido{\rbeta=0.25+0.25,\ired=0+5,\rblue=50.0+-2.5}{20}{%
   \psBetaDist[beta=\rbeta,linecolor=red!\ired!blue!\rblue]{0.01}{0.99}}
 \psaxes[Dy=0.2,Dx=0.1]{->}(0,0)(1,2.01)
\end{pspicture*}
```

9.9 Cauchy distribution

The Cauchy distribution, also called the Lorentz distribution, is a continuous distribution describing resonance behavior. It also describes the distribution of horizontal distances at which a line segment tilted at a random angle cuts the x-axis.

The general Cauchy distribution and its cumulative distribution can be written as

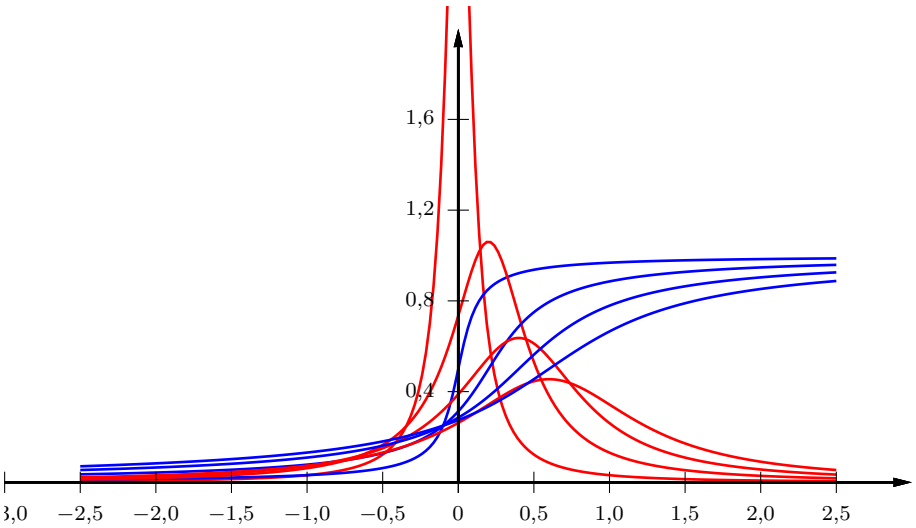
$$P(x) = \frac{1}{\pi}\frac{b}{(x-m)^2 + b^2} \tag{39}$$

$$D(x) = \frac{1}{2} + \frac{1}{\pi}\arctan\left(\frac{x-m}{b}\right) \tag{40}$$

where `b` is the half width at half maximum and `m` is the statistical median. The macro has the syntax (with a default setting of $m = 0$ and $b = 1$):

```
\psCauchy [Options] {x0}{x1}
\psCauchyI [Options] {x0}{x1}
```

`\psCauchyI` is the integral or the cumulative distribution and often named as $D(x)$.




```
\psset{xunit=2,yunit=3cm}
\begin{pspicture*}(-3,-0.3)(3.1,2.1)
\psset{linewidth=1pt}
\multido{\rb=0.1+0.2,\rm=0.0+0.2}{4}{%
  \psCauchy[b=\rb,m=\rm,linecolor=red]{-2.5}{2.5}
    \psCauchyI[b=\rb,m=\rm,linecolor=blue]{-2.5}{2.5}}
\psaxes[Dy=0.4,dy=0.4,Dx=0.5,dx=0.5]{->}(0,0)(-3,0)(3,2)
\end{pspicture*}
```

9.10 Weibull distribution

In probability theory and statistics, the Weibull distribution is a continuous probability distribution. The probability density function of a Weibull random variable x is:

$$P(x) = \alpha\beta^{-\alpha}x^{\alpha-1}e^{-\left(\frac{x}{\beta}\right)^{\alpha}} \tag{41}$$

$$D(x) = 1 - e^{-\left(\frac{x}{\beta}\right)^{\alpha}} \tag{42}$$

or slightly different as

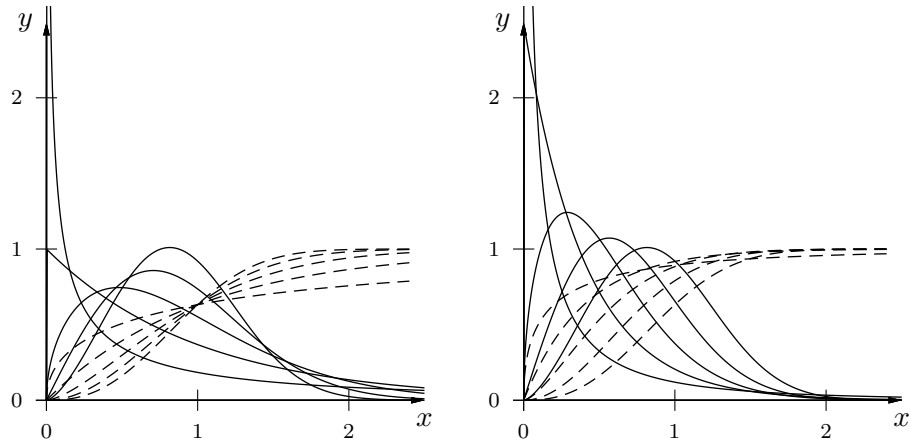
$$P(x) = \frac{\alpha}{\beta}\,x^{\alpha-1}e^{-\frac{x^{\alpha}}{\beta}} \tag{43}$$

$$D(x) = 1 - e^{-\frac{x^{\alpha}}{\beta}} \tag{44}$$

always for $x \in [0;\infty)$, where $\alpha > 0$ is the shape parameter and $\beta > 0$ is the scale parameter of the distribution.

$D(x)$ is the cumulative distribution function of the Weibull distribution. The values for α and β are preset to 1, but can be changed in the usual way.

The Weibull distribution is related to a number of other probability distributions; in particular, it interpolates between the exponential distribution $(\alpha = 1)$ and the Rayleigh distribution $(\alpha = 2)$.




```
\psset{unit=2}
\begin{pspicture*}(-0.5,-0.5)(2.6,2.6)
\psaxes{->}(0,0)(2.5,2.5)[$x$,-90][$y$,180]
\multido{\rAlpha=0.5+0.5}{5}{%
  \psWeibull[alpha=\rAlpha]{0}{2.5}
  \psWeibullI[alpha=\rAlpha,linestyle=dashed]{0}{2.4}}
\end{pspicture*}
%
\begin{pspicture*}(-0.5,-0.5)(2.6,2.6)
\psaxes{->}(0,0)(2.5,2.5)[$x$,-90][$y$,180]
\multido{\rAlpha=0.5+0.5,\rBeta=0.2+0.2}{5}{%
  \psWeibull[alpha=\rAlpha,beta=\rBeta]{0}{2.5}
  \psWeibullI[alpha=\rAlpha,beta=\rBeta,linestyle=dashed]{0}{2.4}}
\end{pspicture*}
```

The starting value for x should always be 0 or greater, if it is less than 0 then the macro draws a line from (#1,0) to (0,0) and starts `\psWeinbull` with 0.

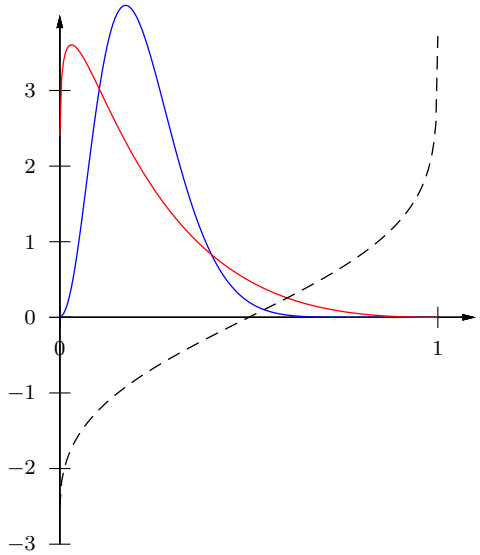
9.11 Vasicek distribution

For a homogenous portfolio of infinite granularity the portfolio loss distribution is given by

$$\mathbb{P}(L(P) < x) = 1 - \mathcal{N}\left(\frac{\mathcal{N}^{-1}(PD) - \sqrt{1-R2}\cdot\mathcal{N}^{-1}(x)}{R}\right)$$

$L(P)$ denotes the portfolio loss in percent, pd is the uniform default probability, and $R2$ is the uniform asset correlation.

They are preset to $pd = 0.22$ and $R2 = 0.11$ and can be overwritten in the usual way. The macro uses the PostScript function norminv from the package `pst-math` which is loaded by default and also shown in the following example.

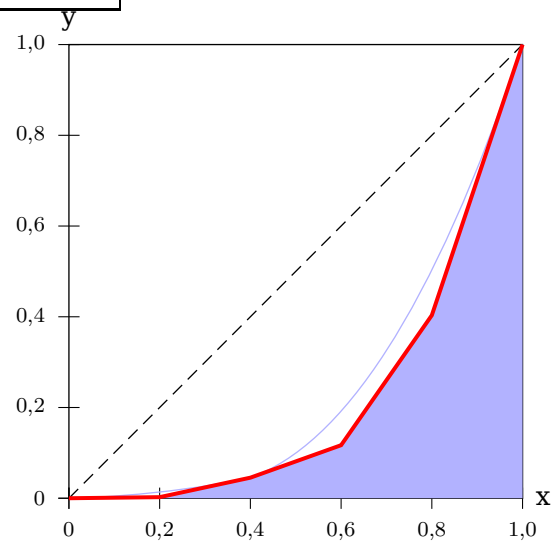



```
\psset{xunit=5}
\begin{pspicture}(-0.1,-3)(1.1,4)
\psaxes{->}(0,0)(0,-3)(1.1,4)
\psVasicek[plotpoints=200,linecolor=blue]{0}{0.9999}
\psVasicek[plotpoints=200,linecolor=red,pd=0.2,R2=0.3]{0}{0.9999}
\psplot[plotpoints=200,algebraic,linestyle=dashed]{0}{0.9999}{norminv(x)}
\end{pspicture}
```

10 The Lorenz curve

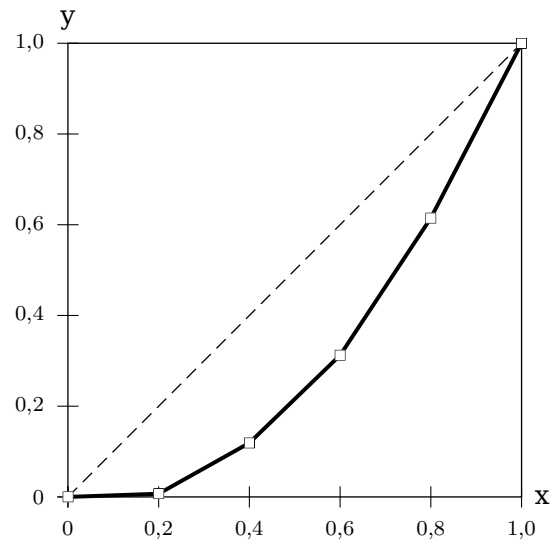
The so-called Lorenz curve is used in economics to describe inequality in wealth or size. The Lorenz curve is a function of the cumulative proportion of *ordered individuals* mapped onto the corresponding cumulative proportion of their size. Given a sample of n^{th} ordered individuals with x'_i the size of individual i and $x'_1 < x'_2 < \cdots < x'_n$, then the sample Lorenz curve is the *polygon* joining the points $(h/n, L_h/L_n)$, where $h = 0, 1, 2, \ldots n$, $L_0 = 0$ and $L_h = \sum_{i=1}^{h} x'_i$.

`\psLorenz*` `[Options]` {*data file*}




```
\psset{lly=-6mm,llx=-5mm}
\psgraph[Dx=0.2,Dy=0.2,axesstyle=frame](0,0)(1,1){6cm}{6cm}
\psline[linestyle=dashed](1,1)
\psLorenz*[linecolor=blue!30,linewidth=1.5pt]{0.50 0.10 0.3 0.09 0.01 }
\psLorenz[linecolor=blue!30,plotstyle=bezier]{0.50 0.10 0.3 0.09 0.01 }
\psLorenz[linecolor=red,linewidth=1.5pt]{0.50 0.10 0.3 0.09 0.01 }
\endpsgraph
```

There exists an optional argument `Gini` for the output of the Gini coefficient. It is by default set to `false`. With `true`the value is caculated and printed below the origin of the coordinate system.



Gini: 0,378712

```
\psset{lly=-13mm,llx=-5mm}
\psgraph[Dx=0.2,Dy=0.2,axesstyle=frame](0,0)(1,1){6cm}{6cm}
\psline[linestyle=dashed](1,1)
\psLorenz[linewidth=1.5pt,Gini]{0.025 0.275 0.2 0.270 0.230}
\psLorenz[plotstyle=dots,dotstyle=square,dotscale=1.5]{0.025 0.275 0.2 0.270 0.230}
\endpsgraph
```

11 \psLame – Lamé Curve, a superellipse

A superellipse is a curve with Cartesian equation

$$\left|\frac{x}{a}\right|^r + \left|\frac{y}{b}\right|^r = 1 \tag{45}$$

first discussed in 1818 by Gabriel Lamé (1795–1870)[3]. A superellipse may be described parametrically by

$$x = a \cdot \cos^{\frac{2}{r}} t \tag{46}$$

$$y = b \cdot \sin^{\frac{2}{r}} t \tag{47}$$

Superellipses with $a = b$ are also known as Lamé curves or Lamé ovals and the restriction to $r > 2$ is sometimes also made. The following table summarizes a few special cases. Piet Hein used $\frac{5}{2}$ with a number of different $\frac{a}{b}$ ratios for various of his projects. For example, he used $\frac{a}{b} = \frac{6}{5}$ for Sergels Torg (Sergel's Square) in Stockholm, and $\frac{a}{b} = \frac{3}{2}$ for his table.

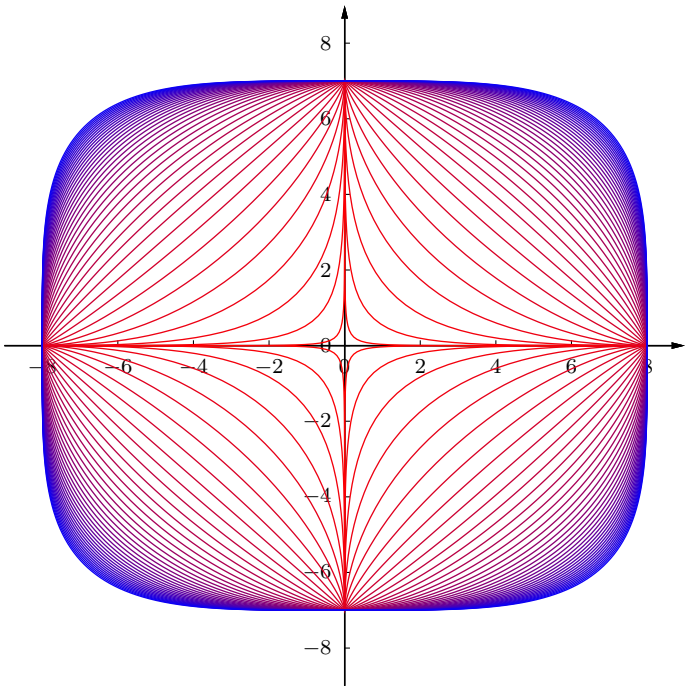
r	curve type	example
$\frac{2}{3}$	(squashed) astroid	
1	(squashed) diamond	
2	ellipse	
$\frac{5}{2}$	Piet Hein's „superellipse“	

If r is a rational, then a superellipse is algebraic. However, for irrational r, it is transcendental. For even integers $r = n$, the curve becomes closer to a rectangle as n increases. The syntax of the `\psLame` macro is:

```
\psLame[Options]{r}
```

It is internally plotted as a parametric plot with $0 \leq \alpha \leq 360$. Available keywords are `radiusA` and `radiusB`, both are preset to 1, but can have any valid value and unit.

3 Lamé worked on a wide variety of different topics. His work on differential geometry and contributions to Fermat's Last Theorem are important. He proved the theorem for $n = 7$ in 1839.




```
\definecolorseries{col}{rgb}{last}{red}{blue}
\resetcolorseries[41]{col}
\psset{unit=.5}
\pspicture(-9,-9)(9,9)
  \psaxes[Dx=2,Dy=2,tickstyle=bottom,ticksize=2pt]{->}(0,0)(-9,-9)(9,9)
  \multido{\rA=0.2+0.1,\iA=0+1}{40}{%
    \psLame[radiusA=8,radiusB=7,linecolor={col!![\iA]},linewidth=.5pt]{\rA}}
\endpspicture
```

12 \psThomae – the popcorn function

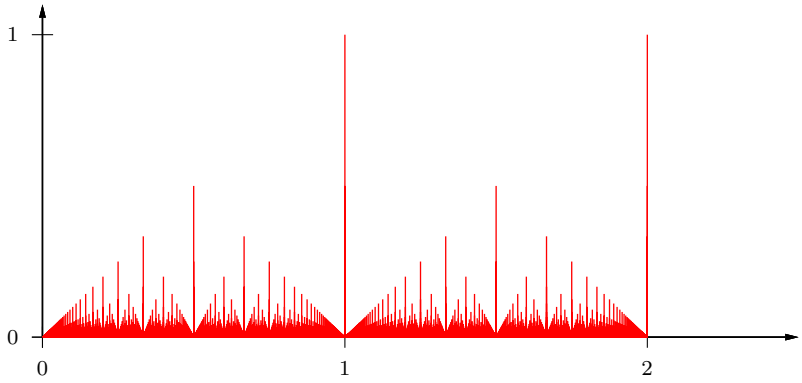
Thomae's function, also known as the popcorn function, the raindrop function, the ruler function or the Riemann function, is a modification of the Dirichlet function. This real-valued function $f(x)$ is defined as follows:

$$f(x) = \begin{cases} \frac{1}{q} \text{ if } x = \frac{p}{q} \text{ is a rational number} \\ 0 \text{ if } x \text{ is irrational} \end{cases}$$

It is assumed here that $gcd(p, q) = 1$ and $q > 0$ so that the function is well-defined and nonnegative. The syntax is:

```
\psThomae[Options](x0,x1){points}
```

`(x0,x1)` is the plotted interval, both values must be grater zero and $x_1 > x_0$. The plotted number of points is the third parameter.




```
\psset{unit=4cm}
\begin{pspicture}(-0.1,-0.2)(2.5,1.15)
   \psaxes{->}(0,0)(2.5,1.1)
   \psThomae[dotsize=2.5pt,linecolor=red](0,2){300}
\end{pspicture}
```

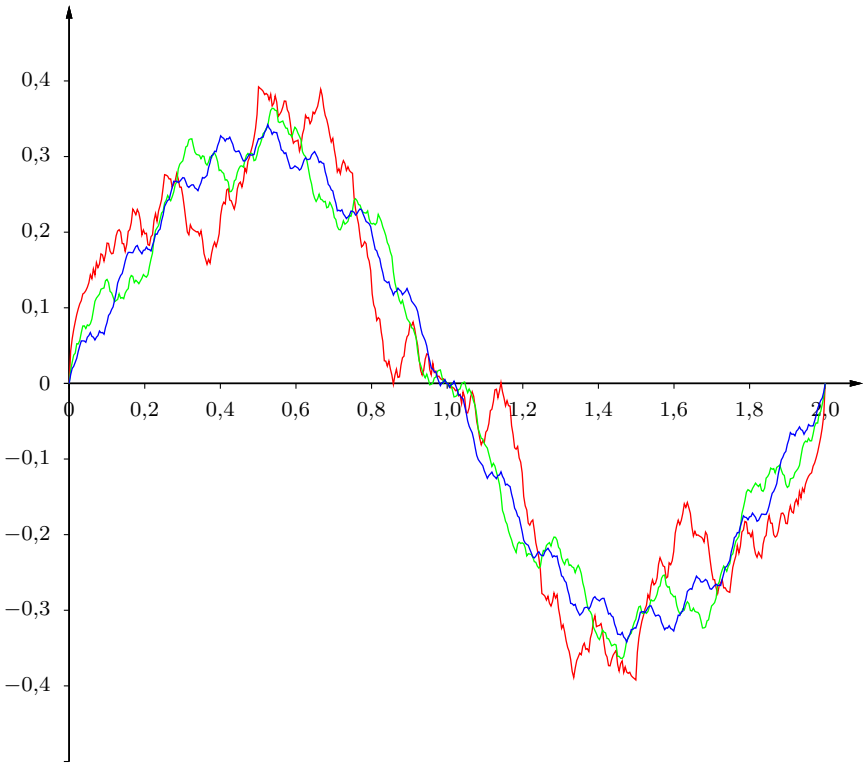
13 \psWeierstrass – a pathological function

The Weierstrass function is an example of a pathological real-valued function on the real line. The function has the property that it is continuous everywhere but differentiable nowhere.

$$f_a(x) = \sum_{k=1}^{\infty} \frac{\sin(\pi k^a x)}{\pi k^a}$$

`\psWeierstrass` `[Options]` (x_0, x_1) `[a]` {*a*/*b*}

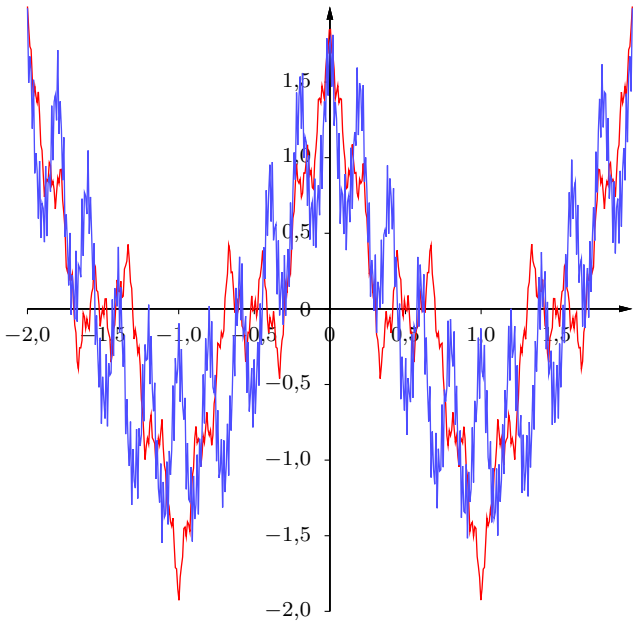
Without the optional argument the mandatory one is a, otherwise it is b and the optional one a. Without setting the optional argument `epsilon` the value of 1.e-8 will be used.




```
\psset{yunit=10,xunit=5}
\begin{pspicture}(-0.1,-0.5)(2.1,0.5)
\psaxes[Dx=0.2,Dy=0.1,ticksize=-2pt 0,
  labelFontSize=\scriptstyle]{->}(0,0)(0,-0.5)(2.1,0.5)
\psWeierstrass[linecolor=red](0,2){2}
\psWeierstrass[linecolor=green,epsilon=1.e-15](0,2){3}
\psWeierstrass[linecolor=blue,epsilon=1.e-5](0,2){4}
\end{pspicture}
```

The original Weierstraß function can be used with the optional argument:

$$f(x) = \sum_{n=0}^{\infty} a^n \cos(b^n \pi x)$$




```
\psset{unit=2cm,linewidth=0.5pt,plotpoints=5000}
\begin{pspicture}(-2.1,-2.1)(2.1,2.1)
\psaxes[Dx=0.5,Dy=0.5,ticksize=-2pt 0,
  labelFontSize=\scriptstyle]{->}(0,0)(-2,-2)(2,2)
\psWeierstrass[linecolor=red](-2,2)[0.5]{3}
\psWeierstrass[linecolor=blue!70](-2,2)[0.5]{10}
\end{pspicture}
```

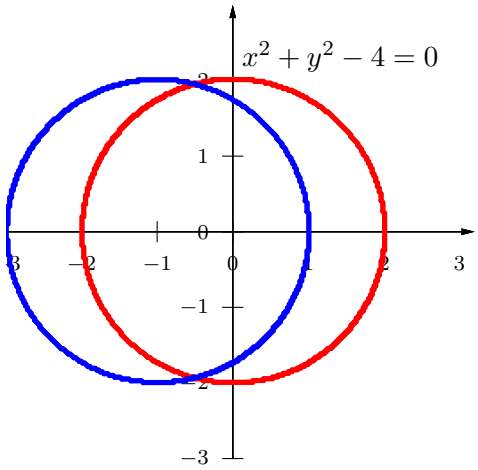
14 \psplotImp – plotting implicit defined functions

For a given area, the macro calculates in a first step row by row for every pixel (1pt) the function $f(x, y)$ and checks for a changing of the value from $f(x, y) < 0$ to $f(x, y) > 0$ or vice versa. If this happens, then the pixel must be part of the curve of the function $f(x, y) = 0$. In a second step the same is done column by column. This may take some time because an area of 400×300 pixel needs 120 thousand calculations of the function value. The user still defines this area in his own coordinates, the translation into pixel (pt) is done internally by the macro itself. The only special keyword is `stepFactor` which is preset to 0.67 and controls the horizontal and vertical step width.

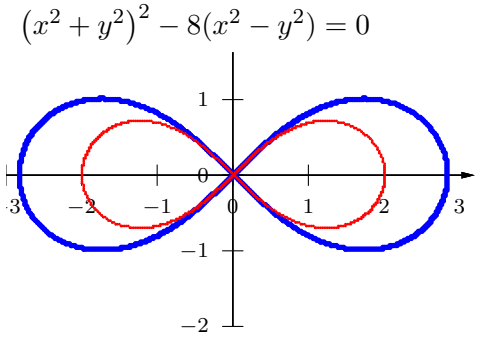
`\psplotImp` `[Options]` (*xMin,yMin*)(*xMax,yMax*) `[PS code]` {*function f(x,y)*}

The function must be of $f(x, y) = 0$ and described in `PostScript`code, or alternatively with the option `algebraic` (`pstricks-add`) in an algebraic form. No other value names than x and y are possible. In general, a starred `pspicture`$*$ environment maybe a good choice here.

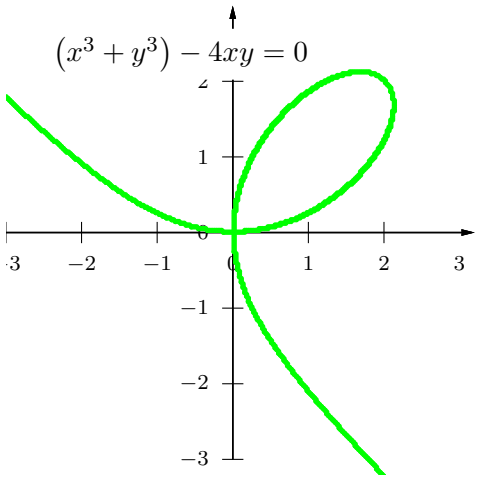
The given area for `\psplotImp` should be **greater** than the given `pspicture` area (see examples).



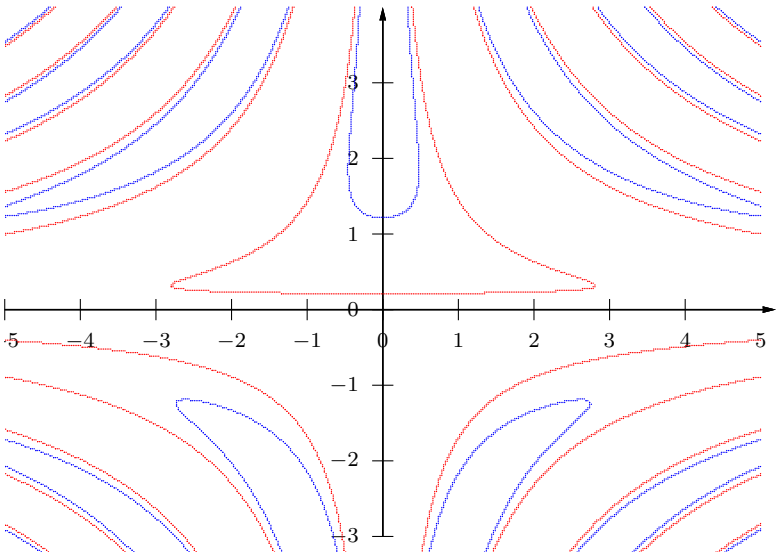

```
\begin{pspicture*}(-3,-3.2)(3.5,3.5)
\psaxes{->}(0,0)(-3,-3)(3.2,3)%
\psplotImp[linewidth=2pt,linecolor=red](-5,-2.1)(5,2.1){ x dup mul y dup mul add 4 sub }
\uput[45](0,2){$x^2+y^2-4=0$}
\psplotImp[linewidth=2pt,linecolor=blue,algebraic](-5,-3)(4,2.4){ (x+1)^2+y^2-4 }
\end{pspicture*}
```




```
\begin{pspicture*}(-3,-2.2)(3.5,2.5)
\psaxes{->}(0,0)(-3,-2)(3.2,2)%
\psplotImp[linewidth=2pt,linecolor=blue](-5,-2.2)(5,2.4){%
    /xqu x dup mul def
    /yqu y dup mul def
    xqu yqu add dup mul 2 dup add 2 mul xqu yqu sub mul sub }
\uput*[0](-3,2){$\left(x^2+y^2\right)^2-8(x^2-y^2)=0$}
\psplotImp[linewidth=1pt,linecolor=red,algebraic](-5,-2.2)(5,2.4){% Lemniskate a =2
   (x^2+y^2)^2-4*(x^2-y^2) }
\end{pspicture*}
```

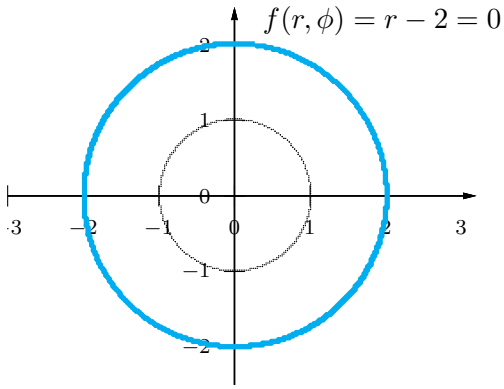



```
\begin{pspicture*}(-3,-3.2)(3.5,3.5)
\psaxes{->}(0,0)(-3,-3)(3.2,3)%
\psplotImp[linewidth=2pt,linecolor=green](-6,-6)(4,2.4){%
    x 3 exp y 3 exp add 4 x y mul mul sub }
\uput*[45](-2.5,2){$\left(x^3+y^3\right)-4xy=0$}
\end{pspicture*}
```

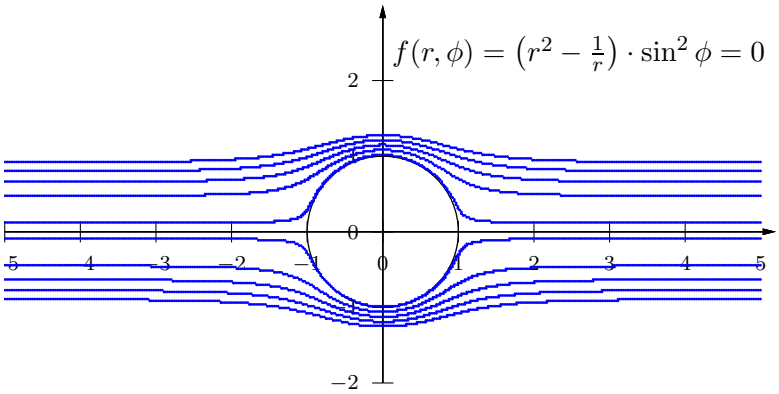



```
\begin{pspicture*}(-5,-3.2)(5.5,4.5)
\psaxes{->}(0,0)(-5,-3)(5.2,4)%
\psplotImp[algebraic,linecolor=red](-6,-4)(5,4){ y*cos(x*y)-0.2 }
\psplotImp[algebraic,linecolor=blue](-6,-4)(5,4){ y*cos(x*y)-1.2 }
\end{pspicture*}
```

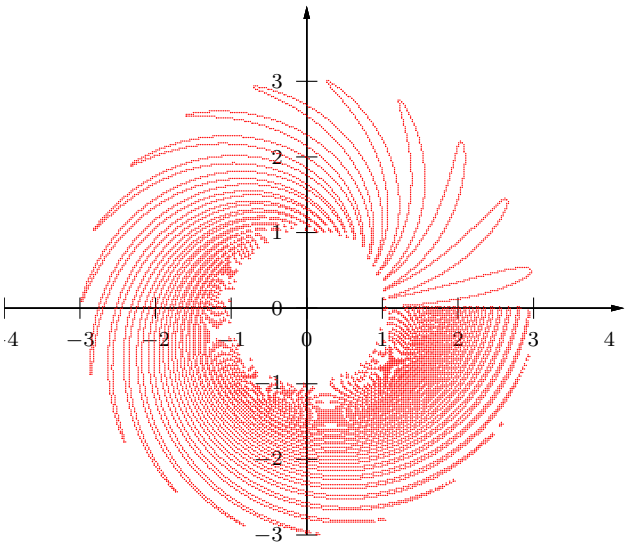
Using the `polarplot` option implies using the variables r and phi for describing the function, y and x are not respected in this case. Using the `algebraic` option for polar plots are also possible (see next example).




```
\begin{pspicture*}(-3,-2.5)(3.75,2.75)\psaxes{->}(0,0)(-3,-2.5)(3.2,2.5)%
\psplotImp[linewidth=2pt,linecolor=cyan,polarplot](-6,-3)(4,2.4){ r 2 sub }% circle r=2
\uput*[45](0.25,2){$f(r,\phi)=r-2=0$}
\psplotImp[polarplot,algebraic](-6,-3)(4,2.4){ r-1 }% circle r=1
\end{pspicture*}
```

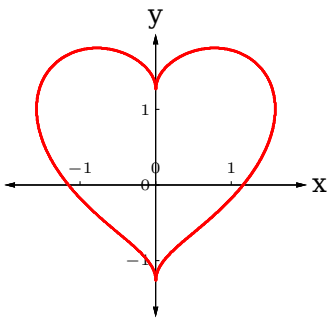



```
\begin{pspicture*}(-5,-2.2)(5.5,3.5)
\pscircle(0,0){1}%
\psaxes{->}(0,0)(-5,-2)(5.2,3)%
\multido{\rA=0.01+0.2}{5}{%
\psplotImp[linewidth=1pt,linecolor=blue,polarplot](-6,-6)(5,2.4){%
    r dup mul 1.0 r div sub phi sin dup mul mul \rA\space sub }}%
\uput*[45](0,2){$f(r,\phi)=\left(r^2-\frac{1}{r}\right)\cdot\sin^2\phi=0$}
\end{pspicture*}
```




```
\begin{pspicture*}(-4,-3.2)(4.5,4.5)
\psaxes{->}(0,0)(-4,-3)(4.2,4)%
\psplotImp[algebraic,polarplot,linecolor=red](-5,-4)(5,4){ r+cos(phi/r)-2 }
\end{pspicture*}
```

The data of an implicit plot can be written into an external file for further purposes. Use the optional argument [pstricks-add]saveData to write the $x|y$ values into the file `\jobname.data`. The file name can be changed with the keyword [pstricks-add]filename. When running a TEX file from within a GUI it may be possible that you get a writeaccess error from GhostScript, because it prevents writing into a file when called from another program. In this case run GhostScript on the `PostScript`-output from the command line.




```
\begin{pspicture*}(-3,-3)(3,3)
  \psaxes[linewidth=0.25pt,
   xlabelPos=top,
   labelFontSize=\scriptscriptstyle,
   labelsep=2pt,
   ticksize=0.05]{<->}(0,0)(-2,-1.75)(2,2)[x,0][y,90]
  \psplotImp[linecolor=red,linewidth=1pt,stepFactor=0.2,saveData,
    algebraic](-2.5,-1.75)(2.5,2.5){x^2+(5*y/4-sqrt(abs(x)))^2-2.5}
\end{pspicture*}
```

The values are saved pairwise in an array, e. g.:

```
...
[
-1.53237 0.695058
-1.53237 1.29957
]
[
-1.52534 0.666941
-1.52534 1.32065
]
...
```

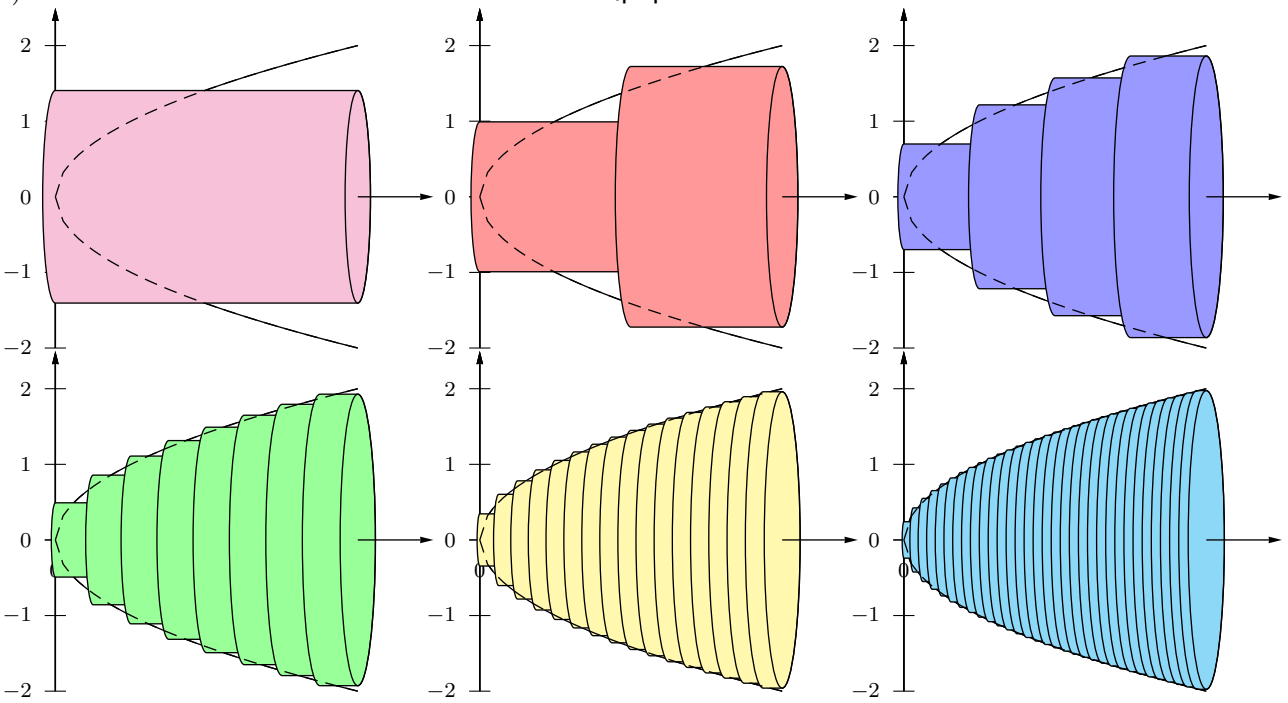
In one array all y values for the same x value are stored.

15 \psVolume – Rotating functions around the x-axis

This macro shows the behaviour of a rotated function around the x-axis.

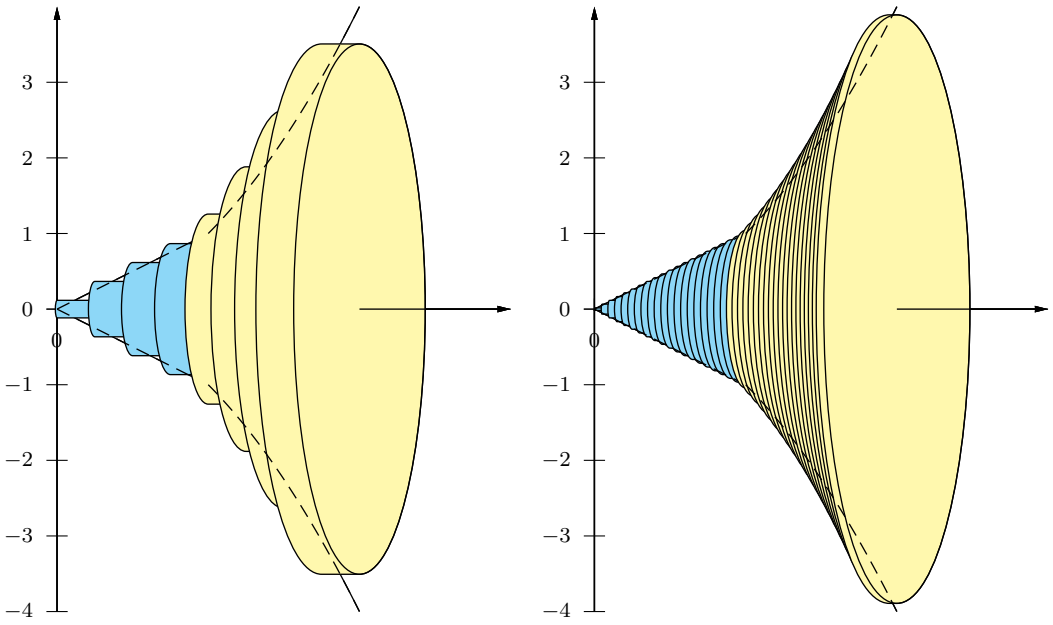
`\psVolume[Options]`(*xMin,xMax*){*steps*}{*function* $f(x)$}

$f(x)$ has to be described as usual for the macro `\psplot`.



```
\begin{pspicture}(-0.5,-2)(5,2.5)
\psaxes{->}(0,0)(0,-2)(3,2.5)
\psVolume[fillstyle=solid,fillcolor=magenta!30](0,4){1}{x sqrt}
\psline{->}(4,0)(5,0)
\end{pspicture}
%
\begin{pspicture}(-0.5,-2)(5,2.5)
\psaxes{->}(0,0)(0,-2)(3,2.5)
\psVolume[fillstyle=solid,fillcolor=red!40](0,4){2}{x sqrt}
\psline{->}(4,0)(5,0)
\end{pspicture}
%
\begin{pspicture}(-0.5,-2)(5,2.5)
\psaxes{->}(0,0)(0,-2)(3,2.5)
\psVolume[fillstyle=solid,fillcolor=blue!40](0,4){4}{x sqrt}
\psline{->}(4,0)(5,0)
\end{pspicture}

\begin{pspicture}(-0.5,-2)(5,2.5)
\psaxes{->}(0,0)(0,-2)(3,2.5)
\psVolume[fillstyle=solid,fillcolor=green!40](0,4){8}{x sqrt}
\psline{->}(4,0)(5,0)
\end{pspicture}
%
\begin{pspicture}(-0.5,-2)(5,2.5)
\psaxes{->}(0,0)(0,-2)(3,2.5)
\psVolume[fillstyle=solid,fillcolor=yellow!40](0,4){16}{x sqrt}
\psline{->}(4,0)(5,0)
\end{pspicture}
```

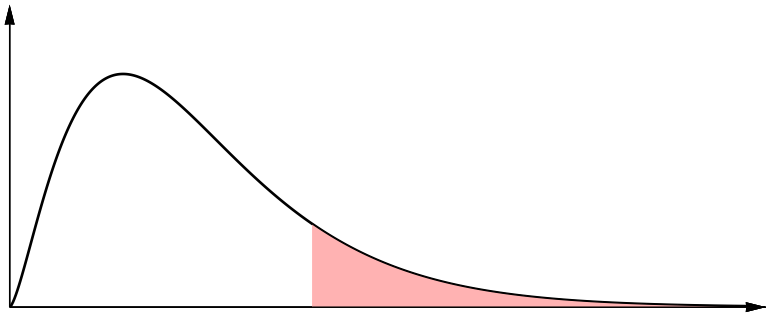
```
%
\begin{pspicture}(-0.5,-2)(5,2.5)
\psaxes{->}(0,0)(0,-2)(3,2.5)
\psVolume[fillstyle=solid,fillcolor=cyan!40](0,4){32}{x sqrt}
\psline{->}(4,0)(5,0)
\end{pspicture}
```




```
\psset{xunit=2}
\begin{pspicture}(-0.5,-4)(3,4)
  \psaxes{->}(0,0)(0,-4)(3,4)
  \psVolume[fillstyle=solid,fillcolor=cyan!40](0,1){4}{x}
  \psVolume[fillstyle=solid,fillcolor=yellow!40](1,2){4}{x dup mul}
  \psline(2,0)(3,0)
\end{pspicture}
%
\begin{pspicture}(-0.5,-4)(3,4)
  \psaxes{->}(0,0)(0,-4)(3,4)
  \psVolume[fillstyle=solid,fillcolor=cyan!40](0,1){20}{x}
  \psVolume[fillstyle=solid,fillcolor=yellow!40](1,2){20}{x dup mul}
  \psline(2,0)(3,0)
\end{pspicture}
```

16 Examples

16.1 Filling an area under a distribution curve



```
\psset{xunit=0.5cm,yunit=20cm,arrowscale=1.5}
\begin{pspicture}(-1,-0.1)(21,0.2)
\psChiIIDist[linewidth=1pt,nue=5]{0.01}{19.5}
\psaxes[labels=none,ticks=none]{->}(20,0.2)
\pscustom[fillstyle=solid,fillcolor=red!30]{%
  \psChiIIDist[linewidth=1pt,nue=5]{8}{19.5}%
  \psline(20,0)(8,0)}
\end{pspicture}
```

16.2 An animation of a distribution

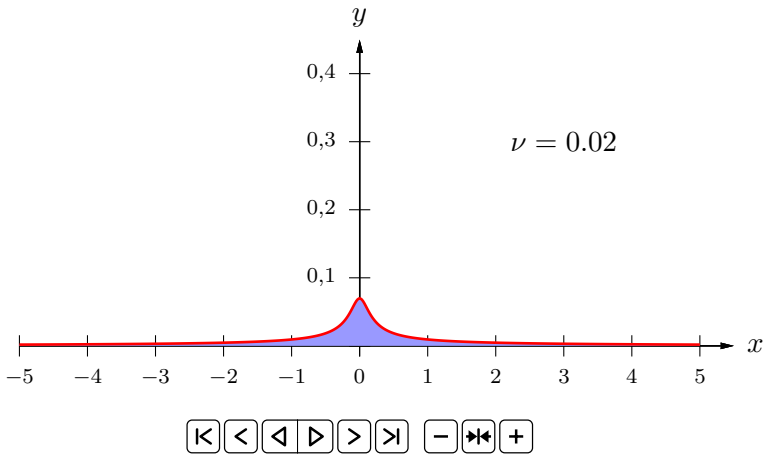


Figure 1: Student's t-distribution.

```
\psset{xunit=0.9cm,yunit=9cm}
\newcommand*\studentT[1]{%
 \begin{pspicture}(-6,-0.1)(6,0.5)
  \psaxes[Dy=0.1]{->}(0,0)(-5,0)(5.5,0.45)[$x$,0][$y$,90]
  \pscustom[fillstyle=solid,fillcolor=blue!40,opacity=0.4,linecolor=red,linestyle=none]{%
    \psline(0,0)(-5,0)
    \psTDist[nue=#1]{-5}{5}
    \psline(5,0)(0,0)
  }
  \psTDist[nue=#1,linecolor=red,linewidth=1pt]{-5}{5}
```

```
  \rput(3,0.3){$\nu = #1$}
 \end{pspicture}}

\begin{center}
 \begin{animateinline}[poster=first,controls,palindrome]{10}
  \multiframe{50}{rA=0.02+0.02}{\studentT{\rA}}
 \end{animateinline}
 \captionof{figure}{Student's $t$-distribution.}
\end{center}
```

17 List of all optional arguments for pst-func

Key	Type	Default
epsilon	ordinary	1.e-08
xShift	ordinary	0
cosCoeff	ordinary	0
sinCoeff	ordinary	1
coeff	ordinary	0 1
Derivation	ordinary	0
markZeros	boolean	true
epsZero	ordinary	0.1
dZero	ordinary	0.1
zeroLineTo	ordinary	-1
zeroLineColor	ordinary	black
zeroLineWidth	ordinary	0.5\pslinewidth
zeroLineStyle	ordinary	dashed
constI	ordinary	1
constII	ordinary	0
n	ordinary	1
sigma	ordinary	0.5
mue	ordinary	0
nue	ordinary	1
Simpson	ordinary	5
PSfont	ordinary	NimbusRomNo9L-Regu
valuewidth	ordinary	10
fontscale	ordinary	10
decimals	ordinary	-1
round	boolean	true
science	boolean	true
PrintVLimit	ordinary	1e-6
Switch2Log	ordinary	80
printValue	boolean	true
LineEnding	boolean	true
VLines	boolean	true
barwidth	ordinary	1
rightEnd	ordinary	2
leftEnd	ordinary	2
labelangle	ordinary	90
xlabelsep	ordinary	0
radiusout	ordinary	2
radiusinL	ordinary	0
radiusinR	ordinary	1.5
LabelColor	ordinary	black
LineEndColorL	ordinary	green
LineEndColorR	ordinary	red
fillcolorA	ordinary	blue!40
fillcolorB	ordinary	red!40

Continued on next page

Continued from previous page

Key	Type	Default
vlinestyle	ordinary	solid
LeftClipX	ordinary	-1
RightClipX	ordinary	-1
alpha	ordinary	0.5
beta	ordinary	0.5
m	ordinary	0
b	ordinary	1
pd	ordinary	0.22
R2	ordinary	0.11
Gini	boolean	true
radiusA	ordinary	1
radiusB	ordinary	1
stepFactor	ordinary	0.67
envelope	boolean	true
xory	boolean	true
approx	boolean	true
Framed	boolean	true
Newton	boolean	true
PrintCoord	boolean	true
onlyNode	boolean	true
onlyYVal	boolean	true
originV	boolean	true
PointName	ordinary	I
ydecimals	ordinary	2
labeldistance	ordinary	0

References

[1] Denis Girou. “Présentation de PSTricks”. In: *Cahier GUTenberg* 16 (Apr. 1994), pp. 21–70.

[2] Michel Goosens et al. *The LaTeX Graphics Companion*. 2nd ed. Reading, Mass.: Addison-Wesley Publishing Company, 2007.

[3] Laura E. Jackson and Herbert Voß. “Die Plot-Funktionen von `pst-plot`”. In: *Die TeXnische Komödie* 2/02 (June 2002), pp. 27–34.

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

[5] Herbert Voß. “Die mathematischen Funktionen von PostScript”. In: *Die TeXnische Komödie* 1/02 (Mar. 2002).

[6] Herbert Voß. *`pst-tools` – Helper functions*. 2012. url: /graphics/pstricks/contrib/pst-tools.

[7] Herbert Voß. *`PSTricks` – Grafik für TeX und LaTeX*. 7th ed. Heidelberg and Berlin, 2016.

[8] Herbert Voß. *PSTricks – Graphics for TeX and LaTeX*. Cambridge: UIT, 2011.

[9] Herbert Voß. *LaTeX quick reference*. Cambridge: UIT, 2012.

[10] Herbert Voß. *Typesetting mathematics with LaTeX*. Cambridge: UIT, 2010.

[11] Eric Weisstein. *Wolfram MathWorld*. 2007. url: http://mathworld.wolfram.com.

[12] Timothy van Zandt. *multido.tex - a loop macro, that supports fixed-point addition*. 1997. url: /graphics/pstricks/generic/multido.tex.

[13] Timothy van Zandt. *PSTricks - PostScript macros for generic TEX*. 1993. url: http://www.tug.org/application/PSTricks.

[14] Timothy van Zandt and Denis Girou. “Inside PSTricks”. In: *TUGboat* 15 (Sept. 1994), pp. 239–246.

[15] Timothy van Zandt and Herbert Voß. *pst-plot: Plotting two dimensional functions and data*. 1999. url: /graphics/pstricks/generic/pst-plot.tex.

pst-func-doc