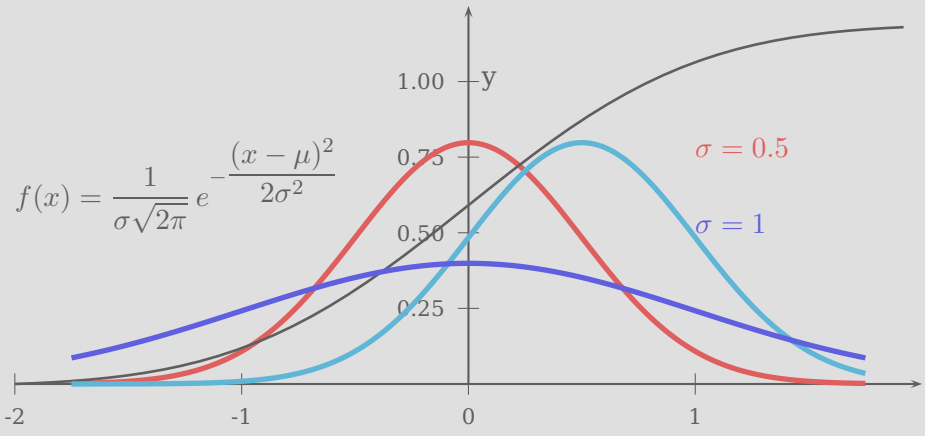
# PSTricks

## pst-func

Plotting special mathematical functions; v.0.60

October 8, 2009




Package author(s):
**Herbert Voß**

## Contents

**1 \psBezier# 4**

**2 Polynomials 6**
2.1 Chebyshev polynomials . . . . . 6
2.2 \psPolynomial . . . . . 10
2.3 \psBernstein . . . . . 16

**3 \psFourier 18**

**4 \psBessel 21**

**5 \psSi, \pssi and \psCi 24**

**6 \psIntegral, \psCumIntegral, and \psConv 26**

**7 Distributions 28**
7.1 Normal distribution (Gauss) . . . . . 29
7.2 Binomial distribution . . . . . 30
7.3 Poisson distribution . . . . . 36
7.4 Gamma distribution . . . . . 39
7.5 $\chi^2$-distribution . . . . . 40
7.6 Student's $t$-distribution . . . . . 41
7.7 $F$-distribution . . . . . 42
7.8 Beta distribution . . . . . 43

**8 \psLame – Lamé Curve, a superellipse 44**

**9 \psThomae – the popcorn function 46**

**10 \psplotImp – plotting implicit defined functions 47**

**11 \psVolume – Rotating functions around the x-axis 51**

**12 \psPrintValue 53**

**13 Examples 55**

**14 List of all optional arguments for pst-func 56**

**References 56**

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.

*If pstricks-add is loaded together with the package pst-func then the InsideArrow of the \psbezier macro doesn't work!*

Thanks to:
Rafal Bartczuk, Jean-Côme Charpentier, Martin Chicoine, Gerry Coombes, Denis Girou, John Frampton, Attila Gati, Horst Gierhardt, Christophe Jorssen, Lars Kotthoff, Buddy Ledger, Manuel Luque, 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 end-point. 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 $n = order + 1$ pairs of coordinates:

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

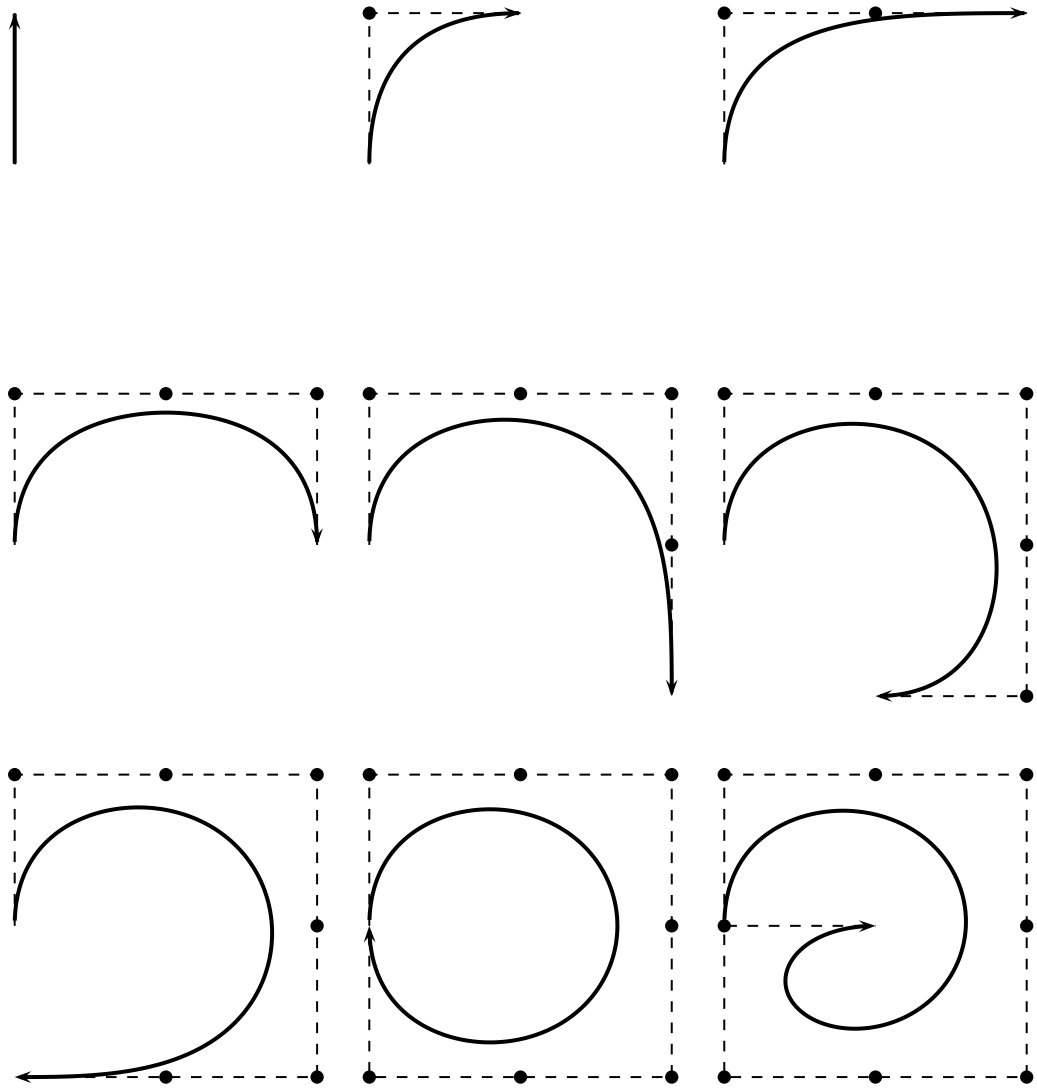
The number of steps between the first and last control points is given by the keyword `plotpoints` and preset to 200. It can be changed in the usual way.

```
\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

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

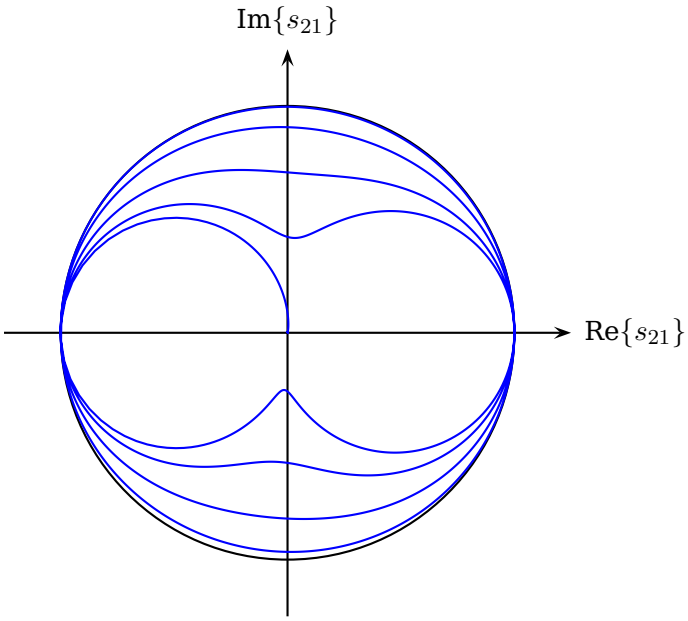
$$= \sum_{n=0}^{\infty} T_n(x)t^n \tag{5}$$

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

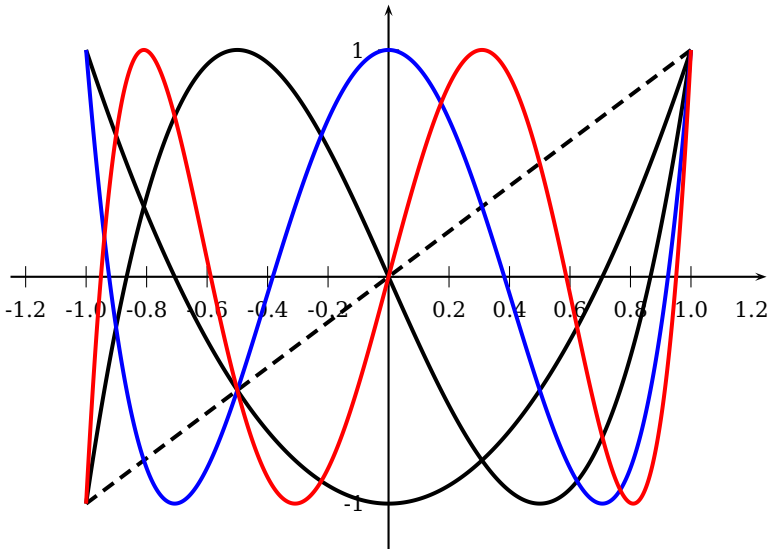
$$g(t,x) = \frac{1}{1-2xt+t^2} \tag{6}$$

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

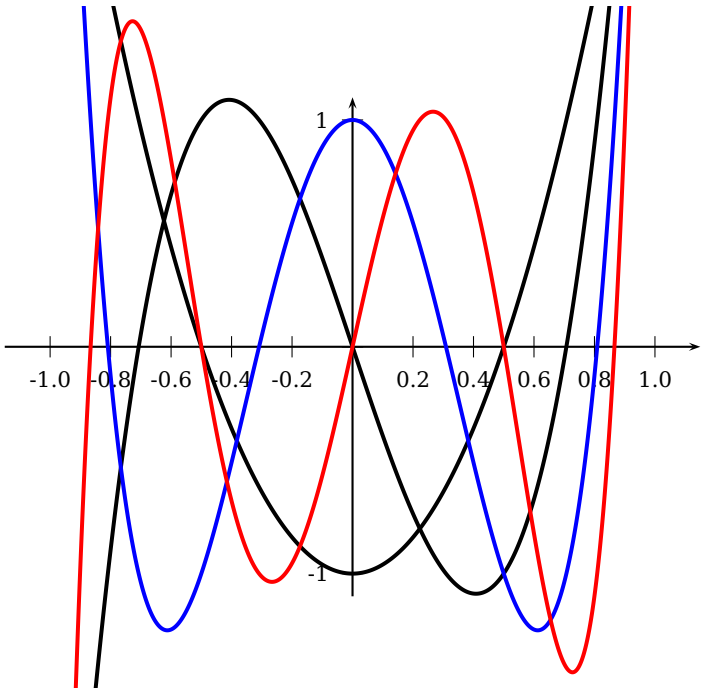
`pst-func` defines the TEX-macros `\ChebyshevT` for the first kind and `\ChebyshevU` for the second kind of Chebyshev polynomials. These TEX-macros cannot be used outside of PostScript, they are only wrappers for `tx@FuncDict begin ChebyshevT end` and the same for `\ChebyshevU`.



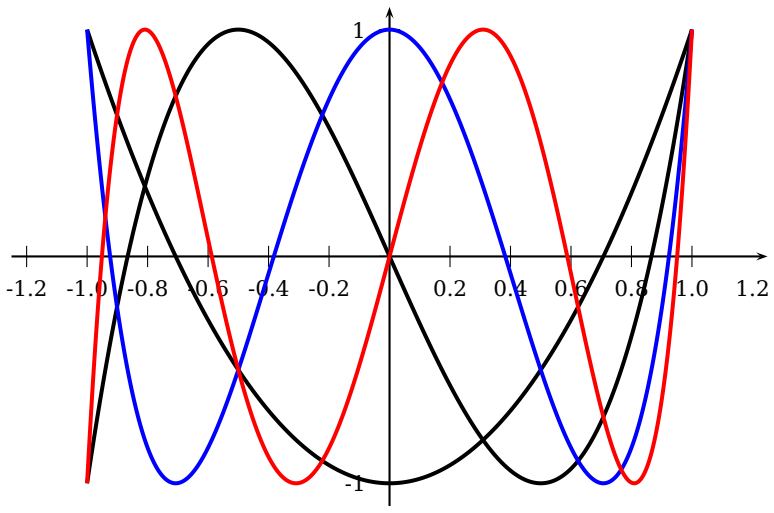

```
\psset{arrowscale=1.5,unit=3cm}
\begin{pspicture}(-1.5,-1.5)(1.5,1.5)
  \psaxes[ticks=none,labels=none]{->}(0,0)(-1.25,-1.25)(1.25,1.25)%
    [Re$\{s_{21}\}$,0][Im$\{s_{21}\}$,90]
  \pscircle(0,0){1}
  \parametricplot[linecolor=blue,plotpoints=10000]{0}{1.5}{
    /N 9 def
    /x 2 N mul t \ChebyshevT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

$$f(x) = a_0 + a_1x + a_2x^2 + a_3x^3 + \ldots + a_{n-1}x^{n-1} + a_nx^n \tag{8}$$

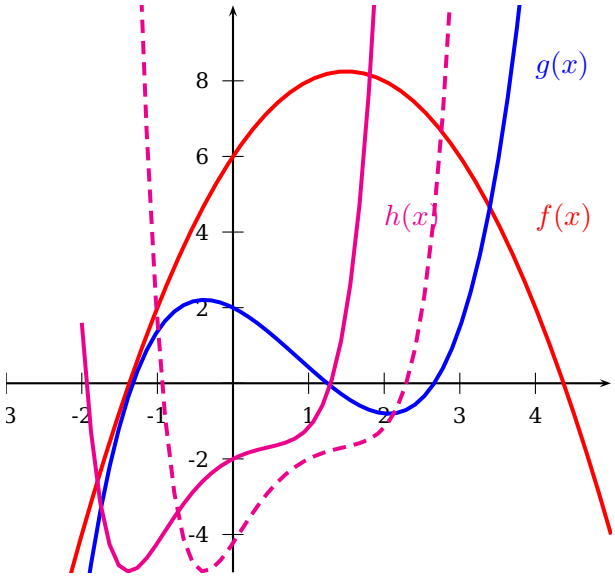
$$f'(x) = a_1 + 2a_2x + 3a_3x^2 + \ldots + (n-1)a_{n-1}x^{n-2} + na_nx^{n-1} \tag{9}$$

$$f''(x) = 2a_2 + 6a_3x + \ldots + (n-1)(n-2)a_{n-1}x^{n-3} + n(n-1)a_nx^{n-2} \tag{10}$$

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 $x - x\text{Shift}$; `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:

$$f(x) = 6 + 3x - x^2 \tag{11}$$

$$g(x) = 2 - x - x^2 + 0.5x^3 - 0.1x^4 + 0.025x^5 \tag{12}$$

$$h(x) = -2 + x - x^2 + 0.5x^3 + 0.1x^4 + 0.025x^5 + 0.2x^6 \tag{13}$$

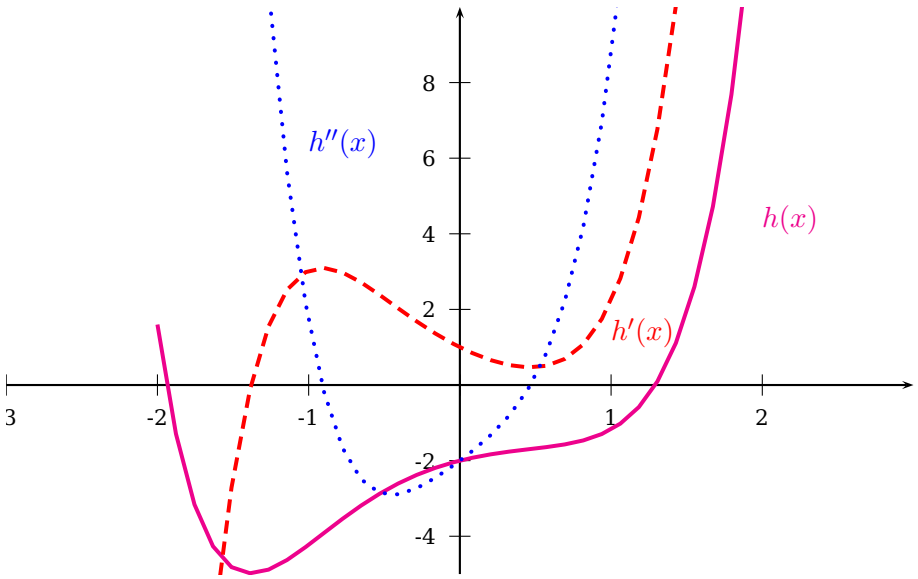
$$\begin{aligned} h^*(x) = &-2 + (x-1) - (x-1)^2 + 0.5(x-1)^3 + \\ &+ 0.1(x-1)^4 + 0.025(x-1)^5 + 0.2(x-1)^6 \end{aligned} \tag{14}$$

There are the following new options:

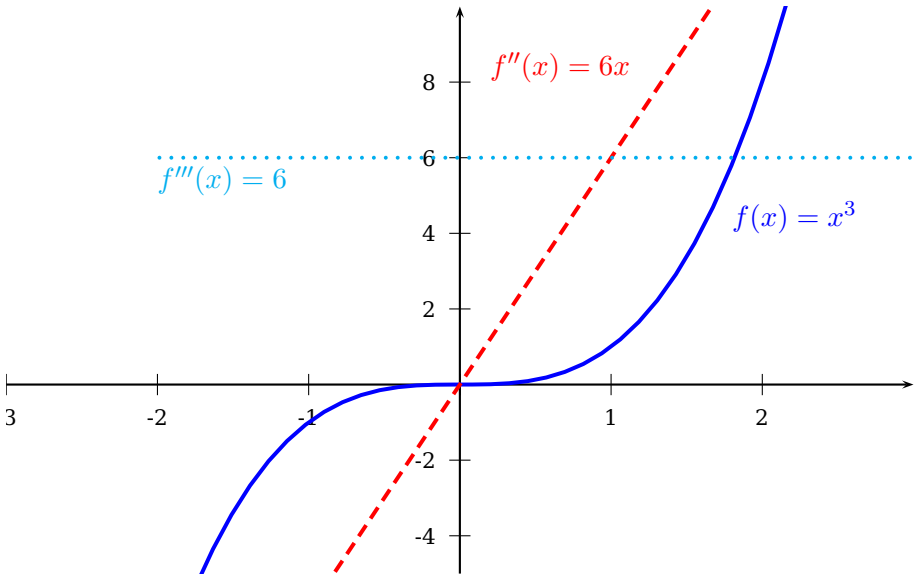
| Name | Value | Default | |
|---|---|---|---|
| `coeff` | `a0 a1 a2 ...` | `0 0 1` | The coefficients must have the order $a_0\ a_1\ a_2 \ldots$ and be separated by **spaces**. The number of coefficients is limited only by the memory of the computer ... The default value of the parameter `coeff` is `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

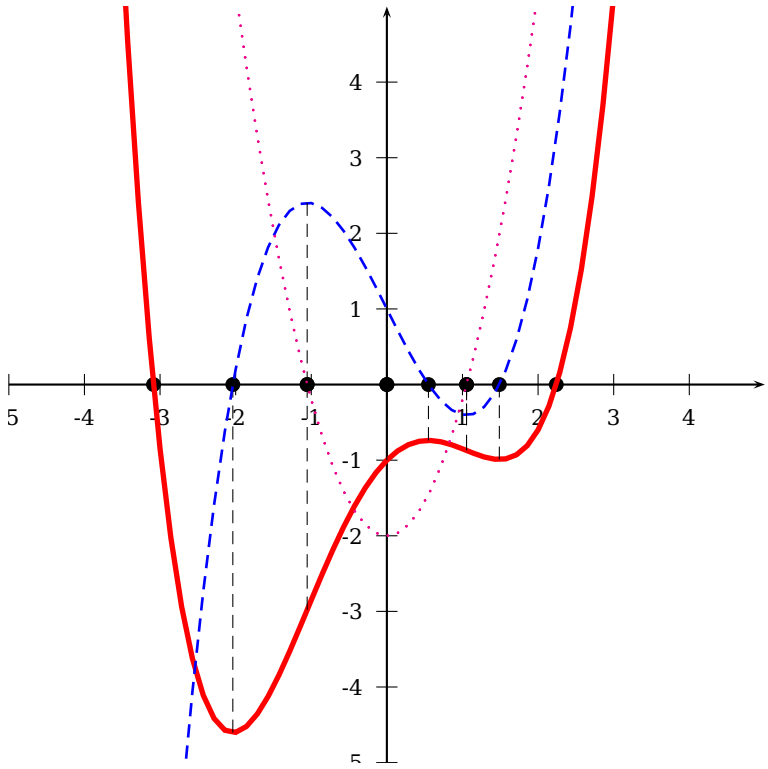
way with \psset.



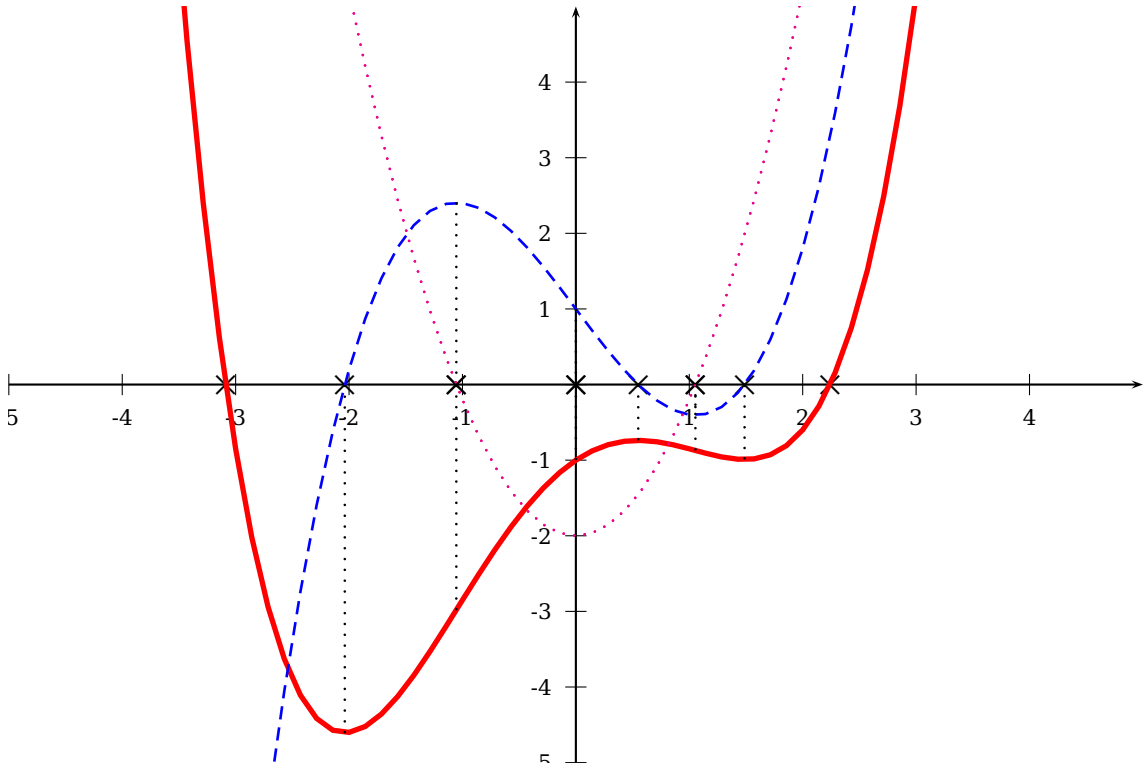

```
\psset{yunit=0.5cm,xunit=2cm}
\begin{pspicture*}(-3,-5)(3,10)
 \psaxes[Dy=2]{->}(0,0)(-3,-5)(3,10)
 \psset{linewidth=1.5pt}
 \psPolynomial[coeff=-2 1 -1 .5 .1 .025 .2 ,linecolor=magenta]{-2}{4}
 \psPolynomial[coeff=-2 1 -1 .5 .1 .025 .2 ,linecolor=red,%
   linestyle=dashed,Derivation=1]{-2}{4}
 \psPolynomial[coeff=-2 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

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

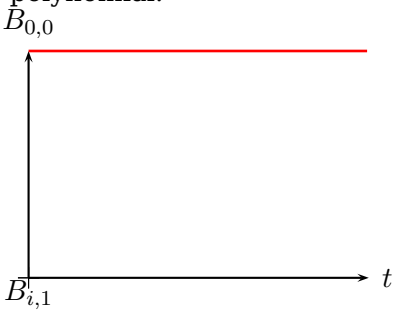
occurring at $t = \frac{i}{n}$. The envelope $f_n(x)$ of the Bernstein polynomials $B_{i,n}(x)$ for $i = 0, 1, \ldots, n$ is given by

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

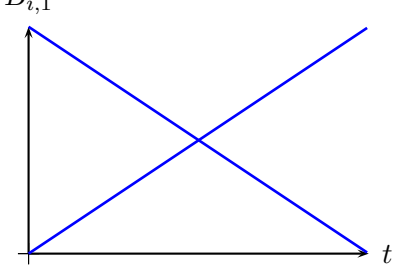
illustrated below for $n = 20$.

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

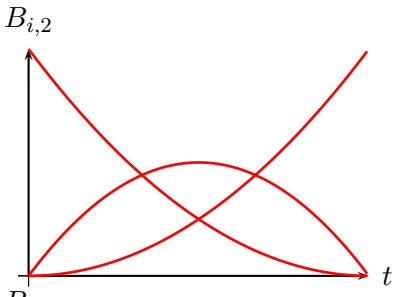
The (`tStart`, `tEnd`) are *optional* and preset by `(0,1)`. The only new optional argument is the boolean key `envelope`, which plots the envelope curve instead of the Bernstein polynomial.



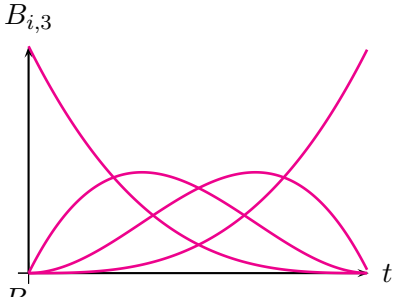

```
\psset{xunit=4.5cm,yunit=3cm}
\begin{pspicture}(1,1.1)
  \psaxes{->}(0,0)(1,1)[$t$,0][$B_{0,0}$,90]
  \psBernstein[linecolor=red,linewidth=1pt](0,0)
\end{pspicture}
```



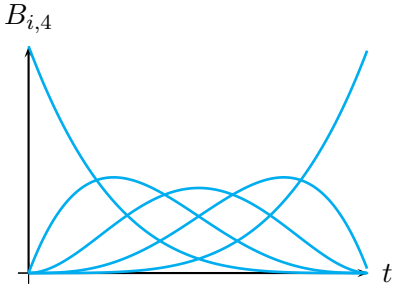

```
\psset{xunit=4.5cm,yunit=3cm}
\begin{pspicture}(1,1.1)
  \psaxes{->}(0,0)(1,1)[$t$,0][$B_{i,1}$,90]
  \psBernstein[linecolor=blue,linewidth=1pt](0,1)
  \psBernstein[linecolor=blue,linewidth=1pt](1,1)
\end{pspicture}
```



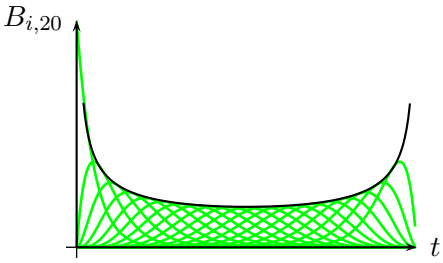

```
\psset{xunit=4.5cm,yunit=3cm}
\begin{pspicture}(1,1.1)
  \psaxes{->}(0,0)(1,1)[$t$,0][$B_{i,2}$,90]
  \multido{\i=0+1}{3}{\psBernstein[linecolor=red,
    linewidth=1pt](\i,2)}
\end{pspicture}
```



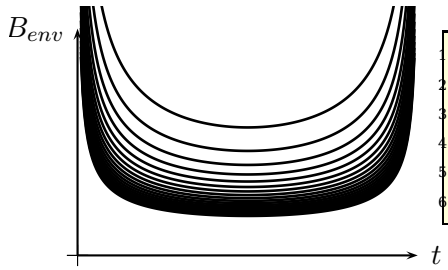

```
\psset{xunit=4.5cm,yunit=3cm}
\begin{pspicture}(1,1.1)
  \psaxes{->}(0,0)(1,1)[$t$,0][$B_{i,3}$,90]
  \multido{\i=0+1}{4}{\psBernstein[linecolor=magenta,
    linewidth=1pt](\i,3)}
\end{pspicture}
```




```
\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*}
```

## 3 \psFourier

A Fourier sum has the form:

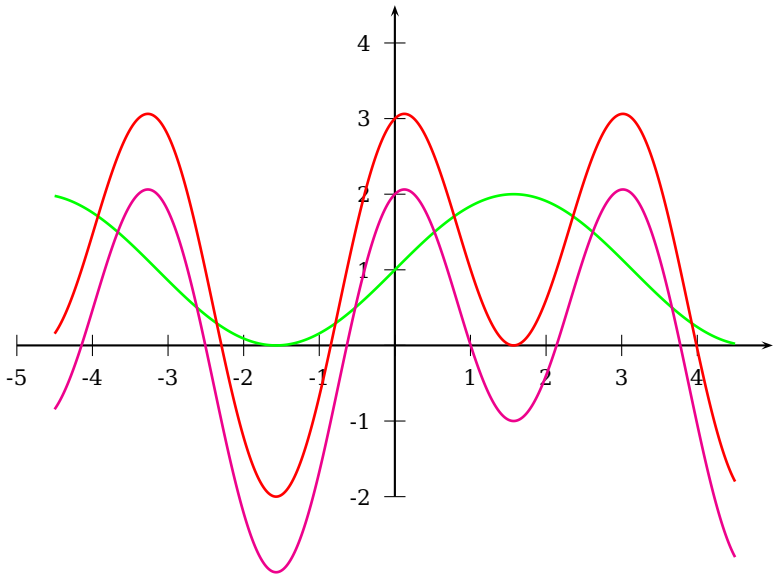
$$s(x) = \frac{a_0}{2} + a_1 \cos \omega x + a_2 \cos 2\omega x + a_3 \cos 3\omega x + \ldots + a_n \cos n\omega x \quad (15)$$

$$+ b_1 \sin \omega x + b_2 \sin 2\omega x + b_3 \sin 3\omega x + \ldots + b_m \sin m\omega x \quad (16)$$

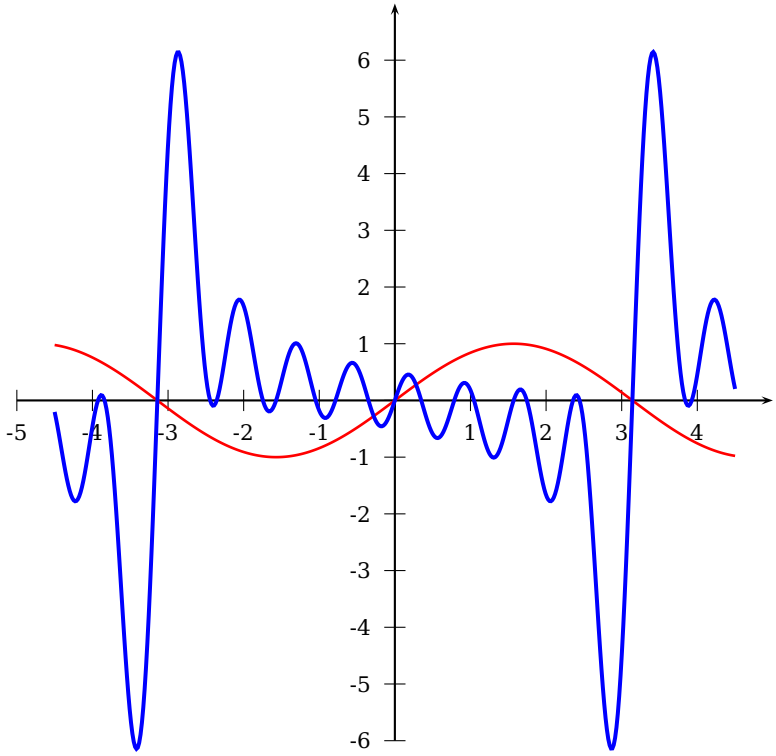
The macro `\psFourier` plots Fourier sums. The syntax is similiar to `\psPolynomial`, except that there are two kinds of coefficients:

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

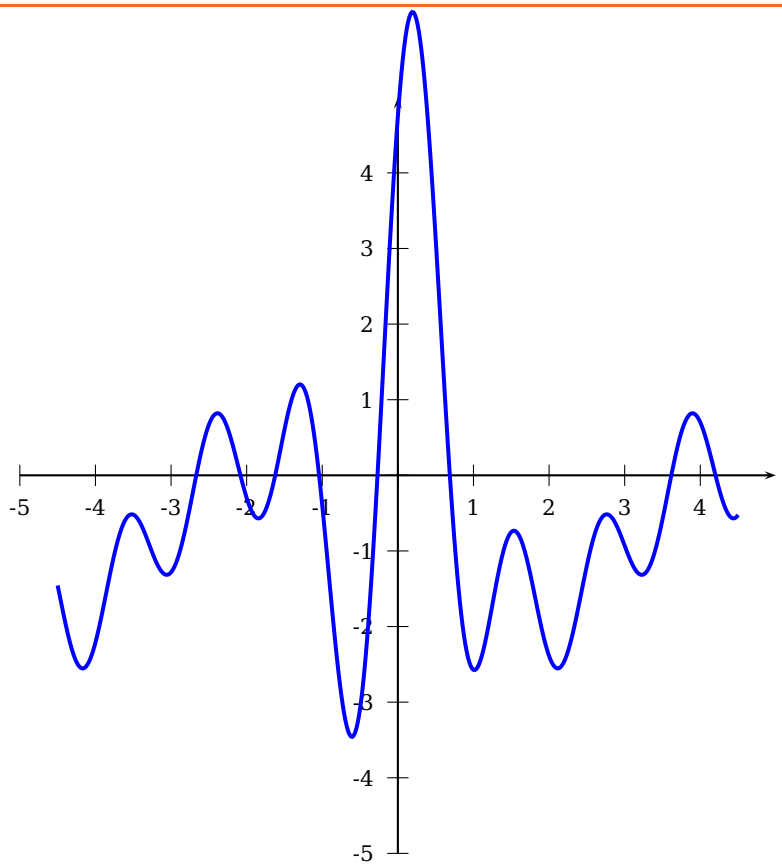
The coefficients must have the orders $cosCoeff = a_0\ a_1\ a_2\ \ldots$ and $sinCoeff = b_1\ b_2\ b_3\ \ldots$ and be separated by **spaces**. The default is `cosCoeff=0,sinCoeff=1`, which gives the standard `sin` function. Note that the constant value can only be set with `cosCoeff=a0`.




```
\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}
```

## 4 \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:

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

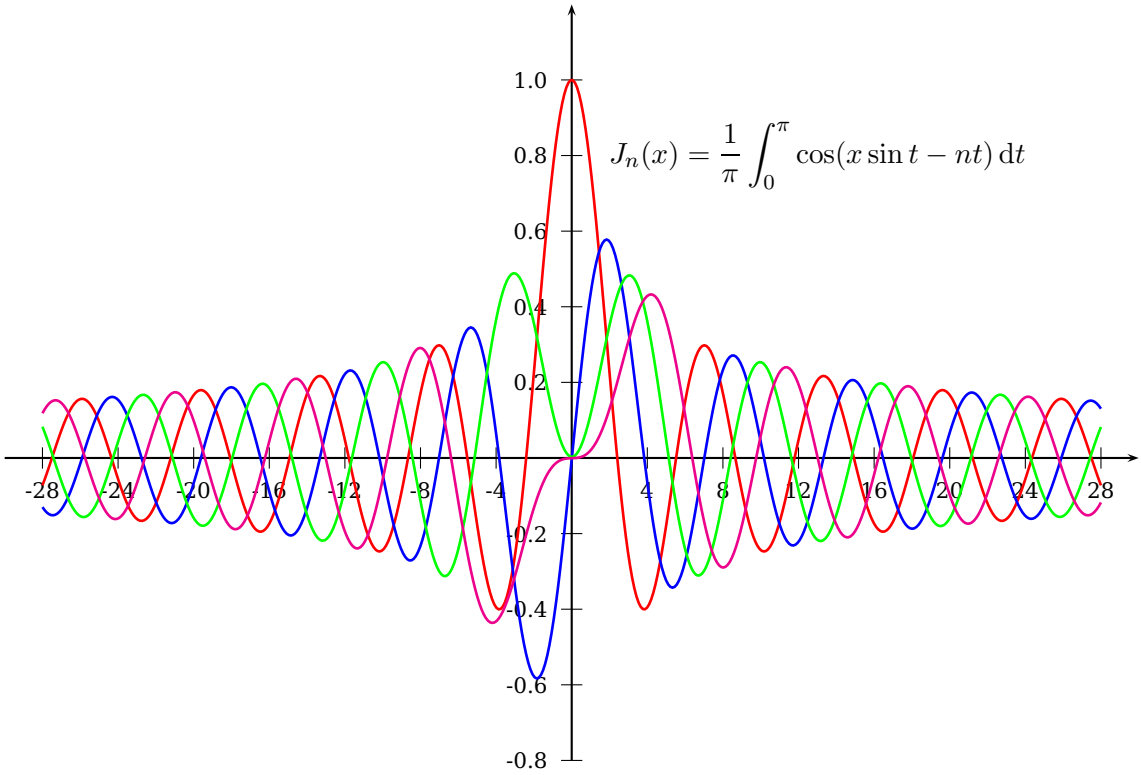
where constI and constII must be real PostScript expressions, e.g.:

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

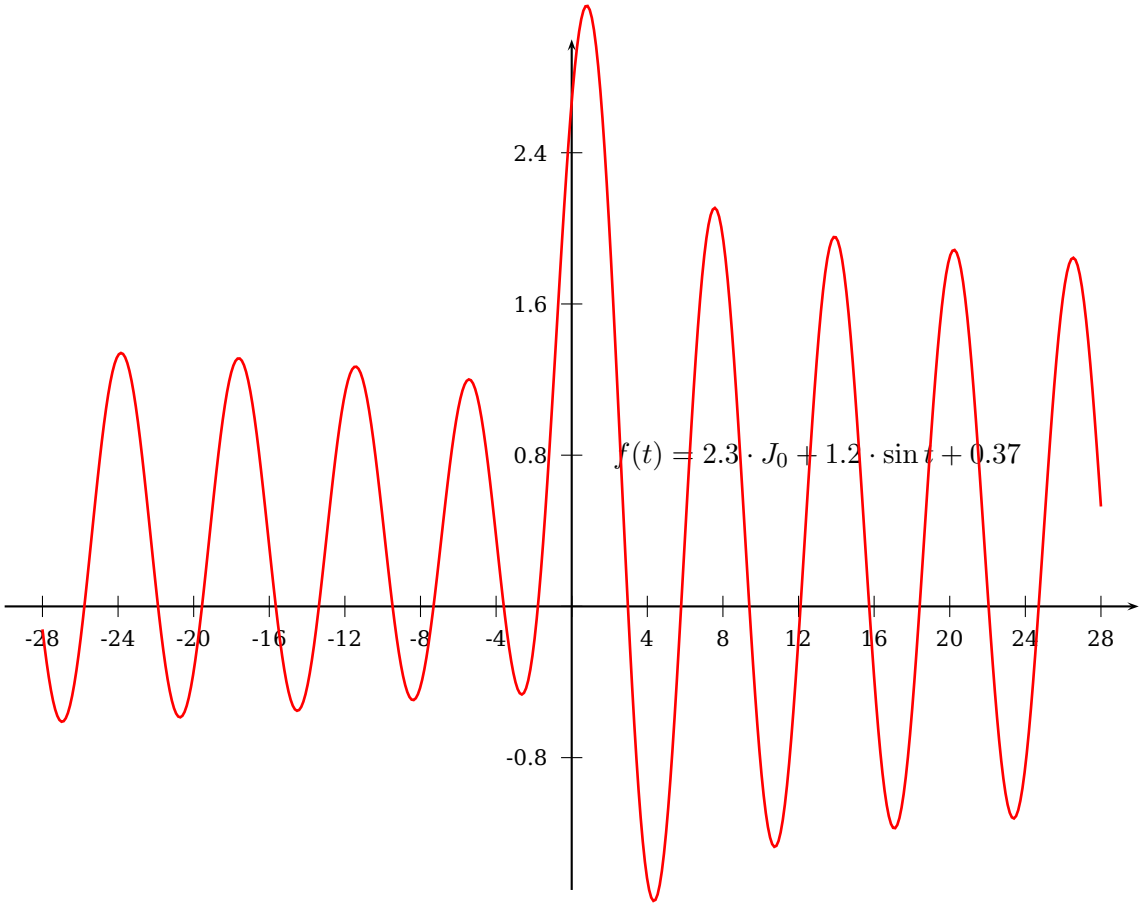
The Bessel function is plotted with the parametricplot macro, this is the reason why the variable is named t. The internal procedure k converts the value t from radian into degrees. The above setting is the same as

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

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}
}
```

## 5 \psSi, \pssi and \psCi

The integral sin and cosin are defined as

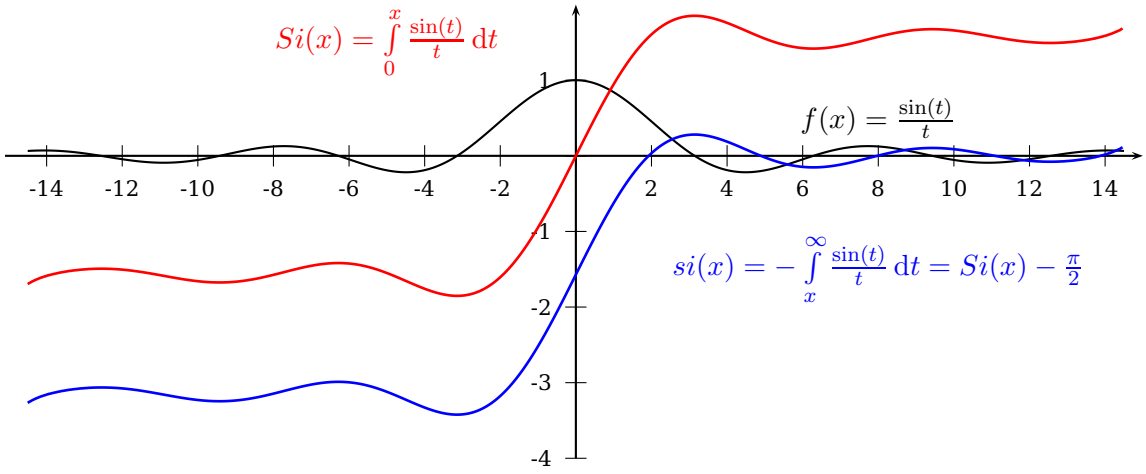
$$\mathrm{Si}(x) = \int_0^x \frac{\sin t}{t}\,\mathrm{d}t \tag{19}$$

$$\mathrm{si}(x) = -\int_x^\infty \frac{\sin t}{t}\,\mathrm{d}t = \mathrm{Si}(x) - \frac{\pi}{2} \tag{20}$$

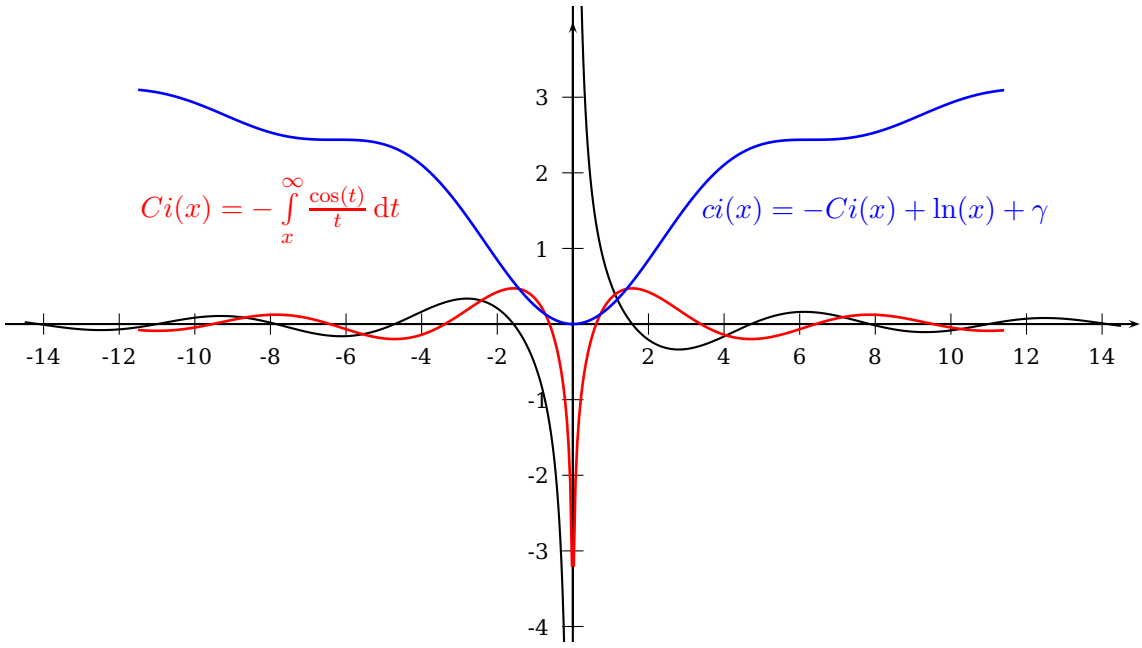
$$\mathrm{Ci}(x) = -\int_x^\infty \frac{\cos t}{t}\,\mathrm{d}t = \gamma + \ln x + \int_0^x \frac{\cos t - 1}{t}\,\mathrm{d}t \tag{21}$$

The syntax of the macros is

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




```
\def\pshlabel#1{\footnotesize#1} \def\psvlabel#1{\footnotesize#1}
\psset{xunit=0.5}
\begin{pspicture}(-15,-4.5)(15,2)
  \psaxes[dx=1cm,Dx=2]{->}(0,0)(-15.1,-4)(15,2)
  \psplot[plotpoints=1000]{-14.5}{14.5}{ x RadtoDeg sin x div }
  \psSi[plotpoints=1500,linecolor=red,linewidth=1pt]{-14.5}{14.5}
  \pssi[plotpoints=1500,linecolor=blue,linewidth=1pt]{-14.5}{14.5}
  \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}
```




```
\def\pshlabel#1{\footnotesize#1} \def\psvlabel#1{\footnotesize#1}
\psset{xunit=0.5}
\begin{pspicture*}(-15,-4.2)(15,4.2)
  \psaxes[dx=1cm,Dx=2]{->}(0,0)(-15.1,-4)(15,4)
  \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*}
```

## 6 \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:

$$\texttt{\textbackslash psIntegral}(x) = \int_a^b f(x,t)\mathrm{d}t \tag{22}$$

$$\texttt{\textbackslash psCumIntegral}(x) = \int_{\mathrm{xStart}}^{x} f(t)\mathrm{d}t \tag{23}$$

$$\texttt{\textbackslash psConv}(x) = \int_a^b f(t)g(x-t)\mathrm{d}t \tag{24}$$

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 $b$.

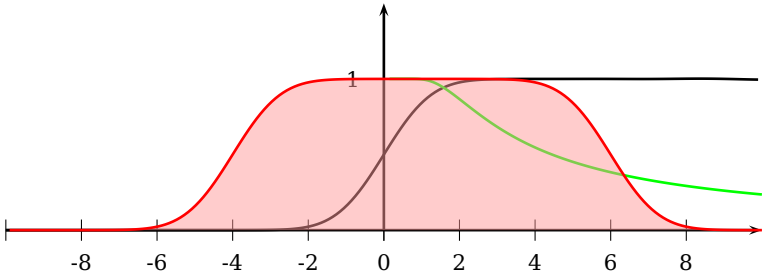
The syntax of these macros is:

\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.

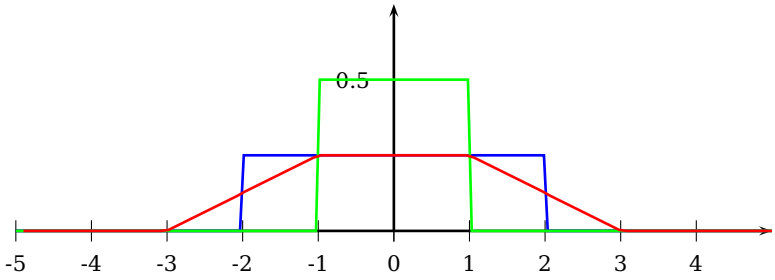
There are no new parameters for these functions. The two most important ones are plotpoints, which controls the number of points of the plot (number of divisions on $x$ for the plot) and Simpson, which controls the precision of the integration (a larger number means a smallest step). The precision and 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 .1 to 10.

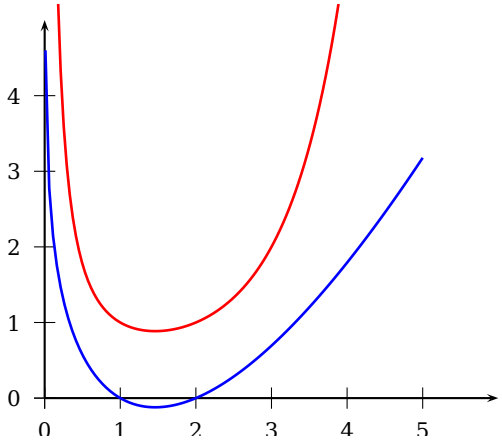



```
\psset{xunit=1cm,yunit=4cm}
\begin{pspicture}[linewidth=1pt](-5,-.2)(5,0.75)
  \psaxes[dx=1cm,Dx=1,Dy=0.5]{->}(0,0)(-5,0)(5,0.75)
  \psplot[linecolor=blue,plotpoints=200]{-5}{5}{x abs 2 le {0.25}{0} ifelse}
  \psplot[linecolor=green,plotpoints=200]{-5}{5}{x abs 1 le {.5}{0} ifelse}
  \psConv[plotpoints=100,Simpson=1000,linecolor=red]{-5}{5}(-10,10)%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

## 7 Distributions

All distributions which use the $\Gamma$- 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 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*}
```

## 7.1 Normal distribution (Gauss)

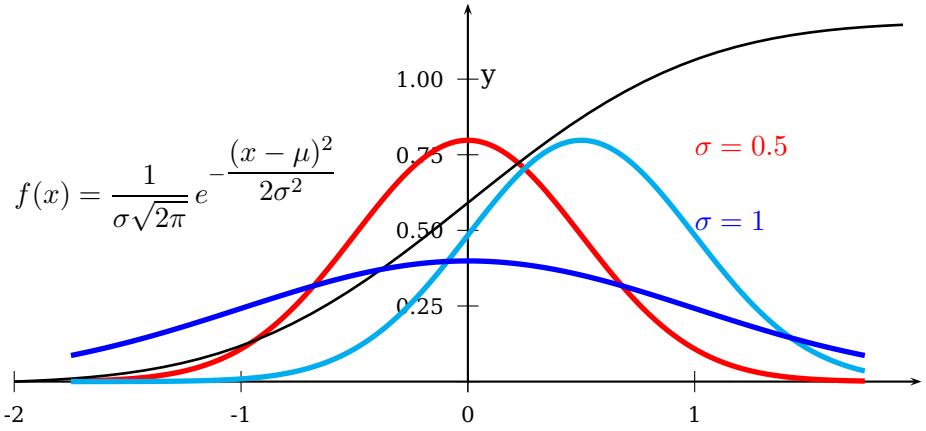
The Gauss function is defined as

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}}\, e^{-\frac{(x-\mu)^2}{2\sigma^2}} \tag{25}$$

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 intervalls 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,yunit=0.75]{-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}
```

## 7.2 Binomial distribution

These two macros plot binomial distribution, `\psBinomialN` the normalized one. It is always done in the $x$-Intervall $[0;1]$. Rescaling to another one can be done by setting the `xunit` option to any other value.

The binomial distribution gives the discrete probability distribution $P_p(n|N)$ of obtaining exactly $n$ 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

$$P_p(n|N) = \binom{N}{n} p^n q^{N-n} \tag{26}$$

$$= \frac{N!}{n!(N-n)!} p^n (1-p)^{N-n}, \tag{27}$$

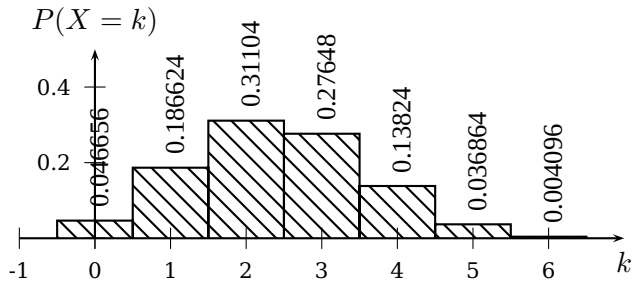
where $(N;n)$ is a binomial coefficient and $P$ the probability.

The syntax is quite easy:

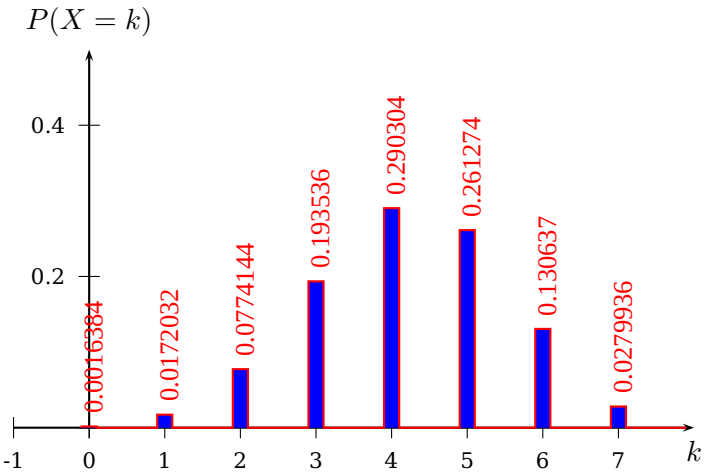
```
\psBinomial [Options] {N}{probability p}
\psBinomial [Options] {m,N}{probability p}
\psBinomial [Options] {m,n,N}{probability p}
\psBinomialN [Options] {N}{probability p}
```

- 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

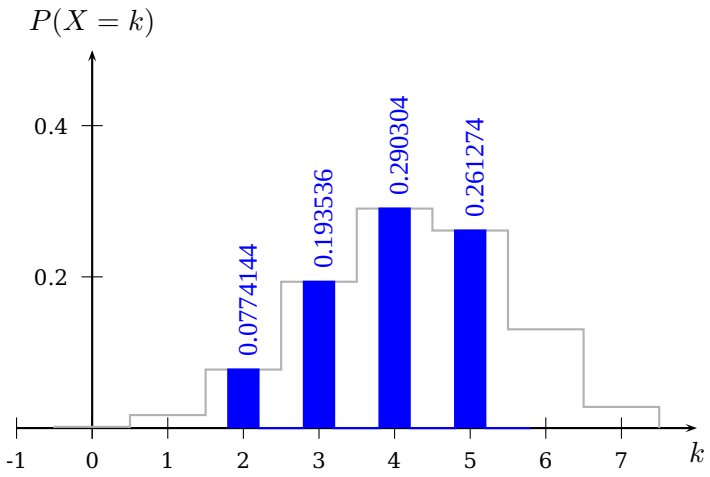
There is a restriction in using the value for N. It depends to the probability, but in general one should expect problems with $N > 100$. PostScript cannot handle such small values and there will be no graph printed. This happens on PostScript side, so TEX doesn't report any problem in the log file. The valid options for the macros are `markZeros` to draw rectangles instead of a continous line and `printValue` for printing the $y$-values on top of the lines, rotated by 90°. For this option all other options from section 12 for the macro `\psPrintValue` are valid, too. The only special option is `barwidth`, which is a factor (no dimension) and set by default to 1. This option is only valid for the macro `\psBinomial` and not for the normalized one!



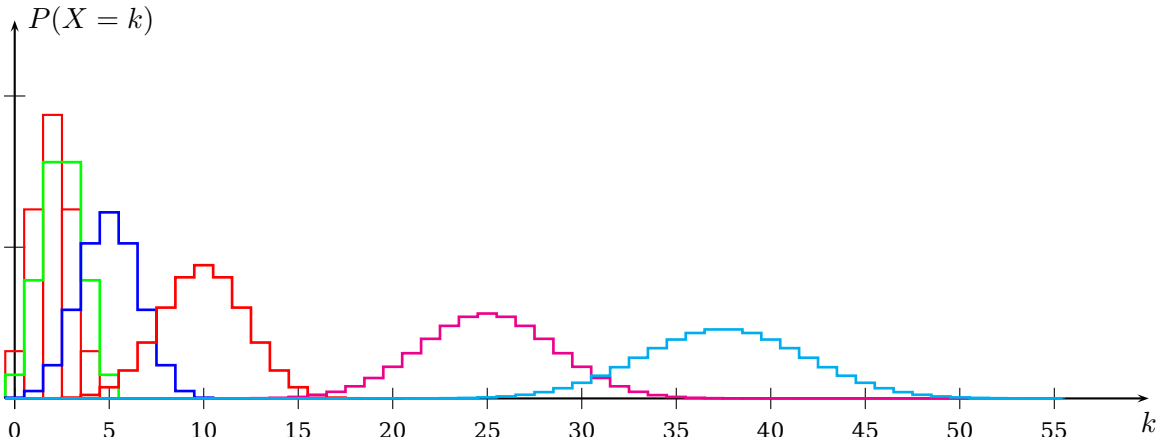

```
\psset{xunit=1cm,yunit=5cm}%
\begin{pspicture}(-1,-0.15)(7,0.55)%
\psaxes[Dy=0.2,dy=0.2\psyunit]{->}(0,0)(-1,0)(7,0.5)
\uput[-90](7,0){$k$} \uput[90](0,0.5){$P(X=k)$}
\psBinomial[markZeros,printValue,fillstyle=vlines]{6}{0.4}
\end{pspicture}
```




```
\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){$P(X=k)$}
\psBinomial[linecolor=red,markZeros,printValue,fillstyle=solid,
   fillcolor=blue,barwidth=0.2]{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}
\end{pspicture}
```

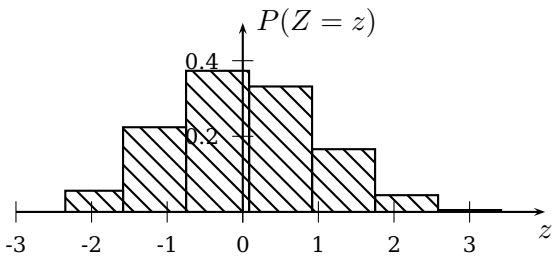



```
\psset{xunit=0.25cm,yunit=10cm}
\begin{pspicture*}(-1,-0.05)(61,0.52)
\psaxes[Dx=5,dx=5\psxunit,Dy=0.2,dy=0.2\psyunit]{->}(60,0.5)
\uput[-90](60,0){$k$} \uput[0](0,0.5){$P(X=k)$}
\psBinomial[markZeros,linecolor=red]{4}{.5}
\psset{linewidth=1pt}
\psBinomial[linecolor=green]{5}{.5} \psBinomial[linecolor=blue]{10}{.5}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

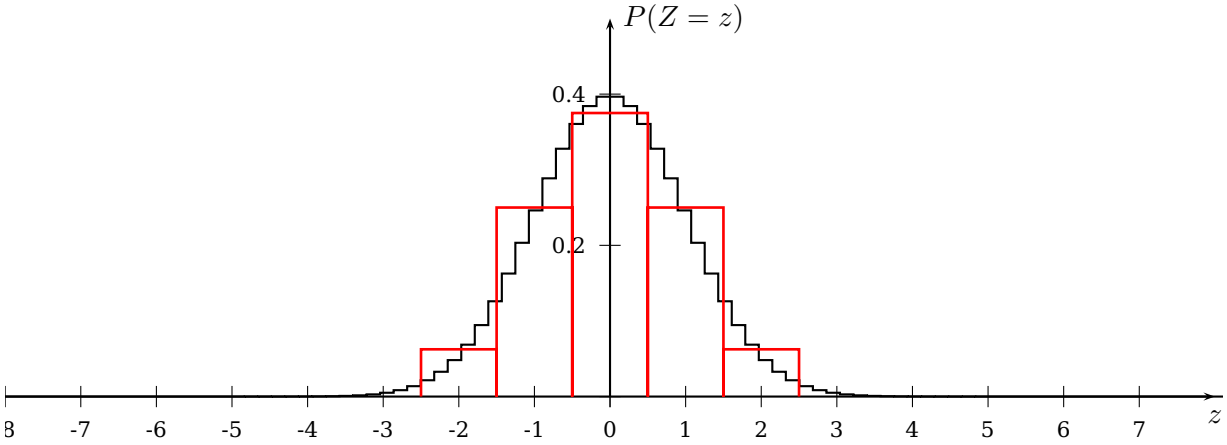
The default binomial distribution has the mean of $\mu = E(X) = N \cdot p$ and a variant of $\sigma^2 = \mu \cdot (1-p)$. The normalized distribution has a mean of 0. Instead of $P(X=k)$ we use $P(Z=z)$ with $Z = \dfrac{X - E(X)}{\sigma(X)}$ and $P \leftarrow P \cdot \sigma$. The macros use the rekursive

definition of the binomial distribution:

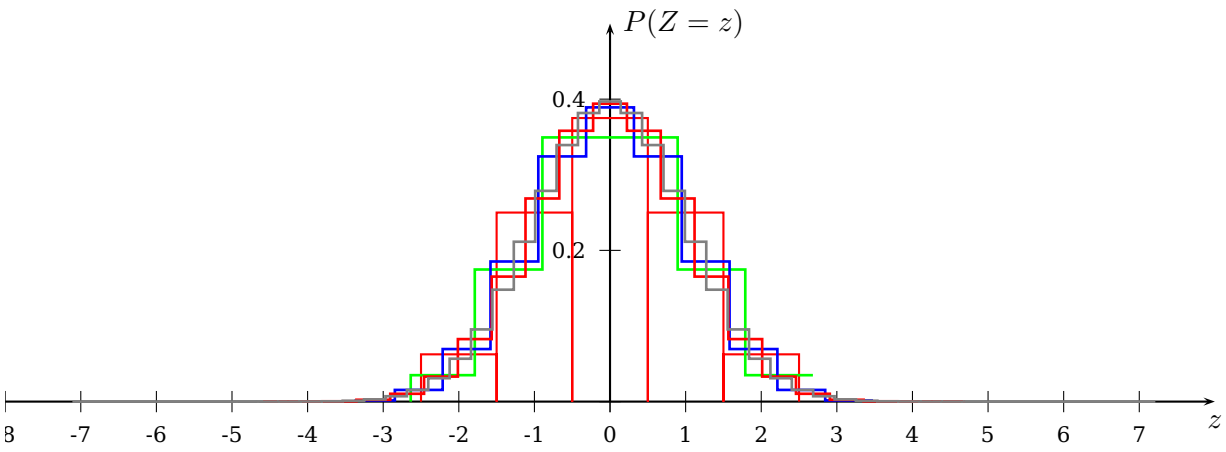
$$P(k) = P(k-1) \cdot \frac{N-k+1}{k} \cdot \frac{p}{1-p} \tag{28}$$




```
\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}
```

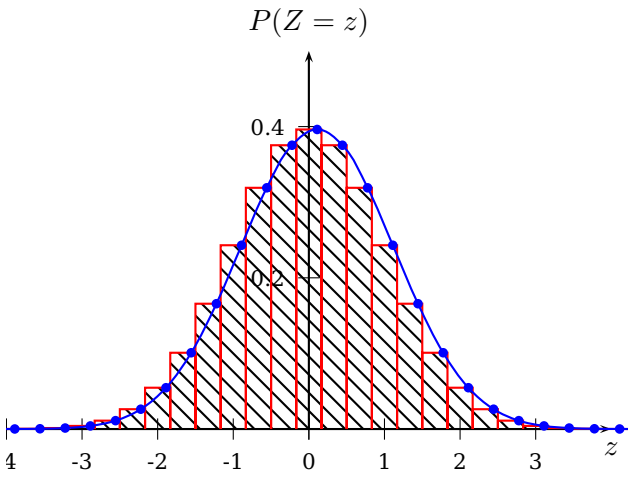



```
\psset{yunit=10}
\begin{pspicture*}(-8,-0.07)(8.1,0.55)
\psaxes[Dy=0.2,dy=0.2\psyunit]{->}(0,0)(-8,0)(8,0.5)
\uput[-90](8,0){$z$} \uput[0](0,0.5){$P(Z=z)$}
\psBinomialN{125}{.5}
\psBinomialN[markZeros,linewidth=1pt,linecolor=red]{4}{.5}
\end{pspicture*}
```

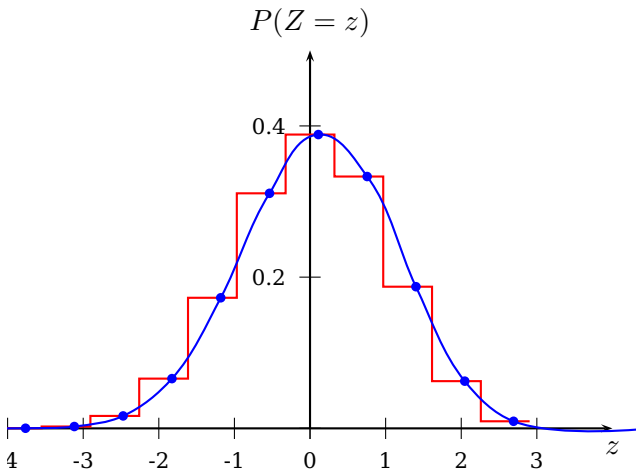



```
\psset{yunit=10}
\begin{pspicture*}(-8,-0.07)(8.1,0.52)
\psaxes[Dy=0.2,dy=0.2\psyunit]{->}(0,0)(-8,0)(8,0.5)
\uput[-90](8,0){$z$} \uput[0](0,0.5){$P(Z=z)$}
\psBinomialN[markZeros,linecolor=red]{4}{.5}
\psset{linewidth=1pt}
\psBinomialN[linecolor=green]{5}{.5}\psBinomialN[linecolor=blue]{10}{.5}
\psBinomialN[linecolor=red]{20}{.5}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

## 7.3 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 7.2)

$$P_p(n|N) = \frac{N!}{n!(N-n)!} \cdot p^n (1-p)^{N-n} \tag{29}$$

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

$$\lambda = n \cdot p \tag{30}$$

instead of the sample size $N$ for fixed $p$, equation (2) then becomes eq. 29

$$P_{\frac{\lambda}{n}}(n|N) = \frac{N!}{n!(N-n)!} \frac{\lambda}{N}^n 1 - \frac{\lambda}{N}^{N-n} \tag{31}$$

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 $p = \frac{\lambda}{n}$)

$$\lim_{n\to\infty} P(X = k) = \lim_{n\to\infty} \frac{n!}{(n-k)!\,k!} \left(\frac{\lambda}{n}\right)^k \left(1 - \frac{\lambda}{n}\right)^{n-k} \tag{32}$$

$$= \lim_{n\to\infty} \left(\frac{(n-k)! \cdot (n-k+1) \cdots (n-2)(n-1)n}{(n-k)!\,n^k}\right) \cdot \tag{33}$$

$$\left(\frac{\lambda^k}{k!}\right) \left(1 - \frac{\lambda}{n}\right)^n \left(1 - \frac{\lambda}{n}\right)^{-k} \tag{34}$$

$$= \frac{\lambda^k}{k!} \cdot \lim_{n\to\infty} \underbrace{\left(\frac{n}{n} \cdot \frac{n-1}{n} \cdot \frac{n-2}{n} \cdot \ldots \cdot \frac{n-k+1}{n}\right)}_{\to 1} \cdot \tag{35}$$

$$\underbrace{\left(1 - \frac{\lambda}{n}\right)^n}_{\to e^{-\lambda}} \underbrace{\left(1 - \frac{\lambda}{n}\right)^{-k}}_{\to 1} \tag{36}$$

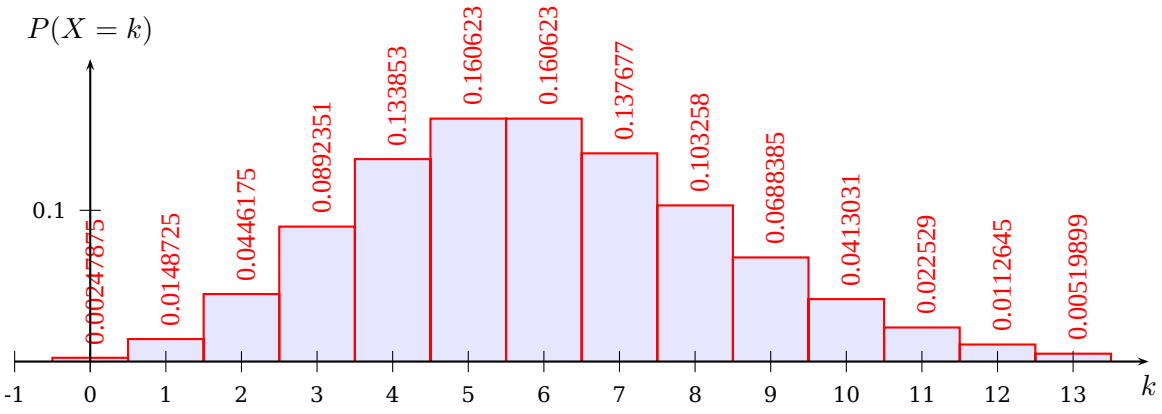
$$= \lambda^k e^{\frac{-\lambda}{k!}} \tag{37}$$

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

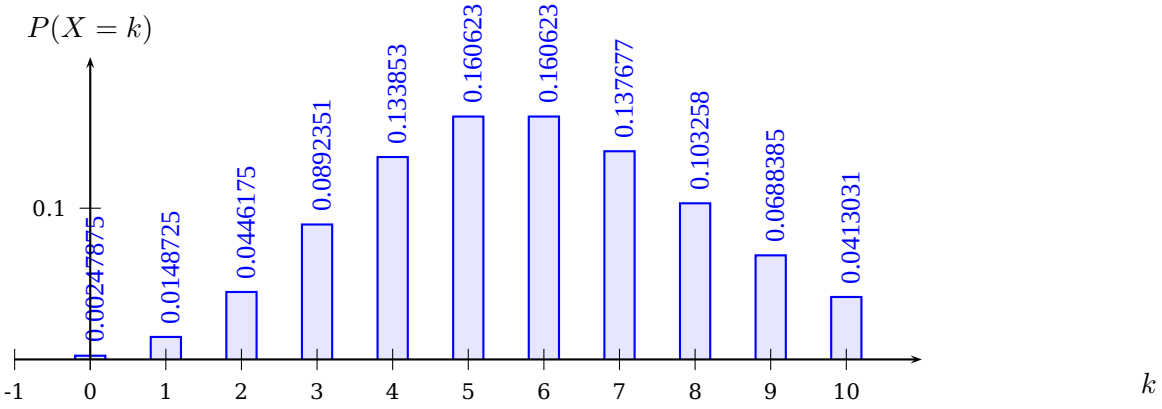
```
\psPoisson [Options] {N}{lambda}
\psPoisson [Options] {M,N}{lambda}
```

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

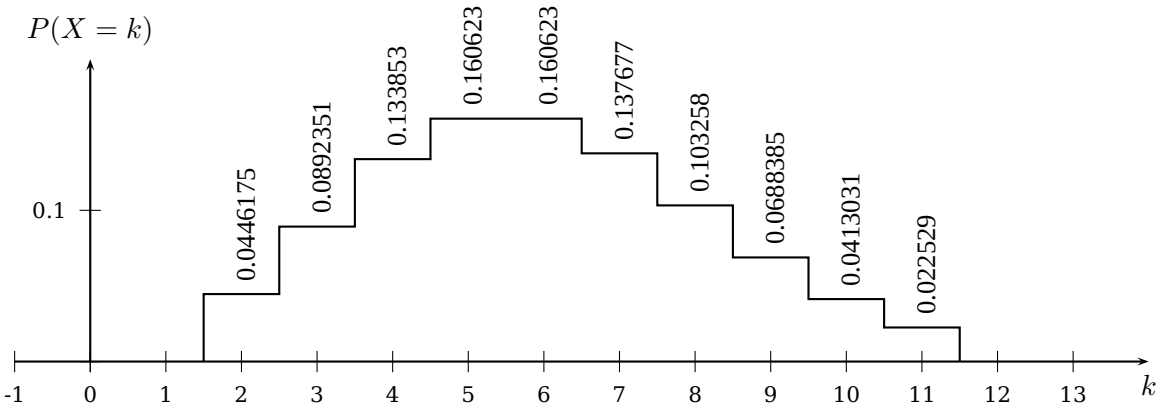
2 http://mathworld.wolfram.com/PoissonProcess.html




```
\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}
```

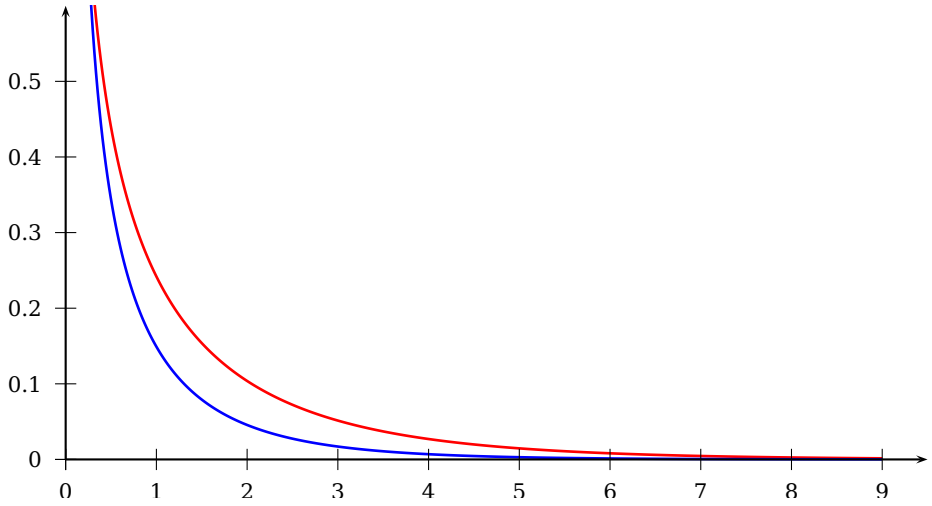
## 7.4 Gamma distribution

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 $alpha$ and $beta$. 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}
```



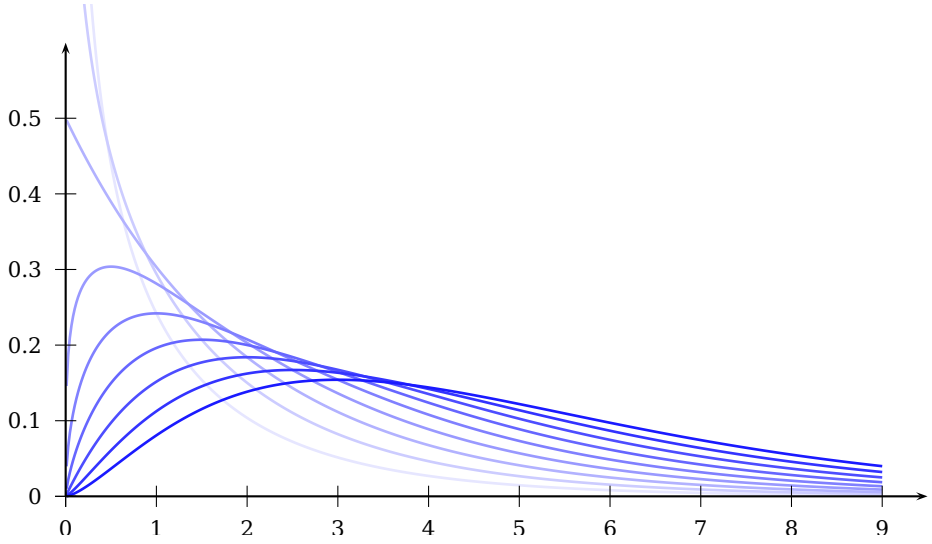

```
\psset{xunit=1.2cm,yunit=10cm,plotpoints=200}
\begin{pspicture*}(-0.75,-0.05)(9.5,0.6)
 \psGammaDist[linewidth=1pt,linecolor=red]{0.01}{9}
 \psGammaDist[linewidth=1pt,linecolor=blue,alpha=0.3,beta=0.7]{0.01}{9}
 \psaxes[Dy=0.1]{->}(0,0)(9.5,.6)
\end{pspicture*}
```

## 7.5 $\chi^2$-distribution

The $\chi^2$-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 $\chi^2$-distribution.

The $\chi^2$ with parameter $\nu$ 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*}
```

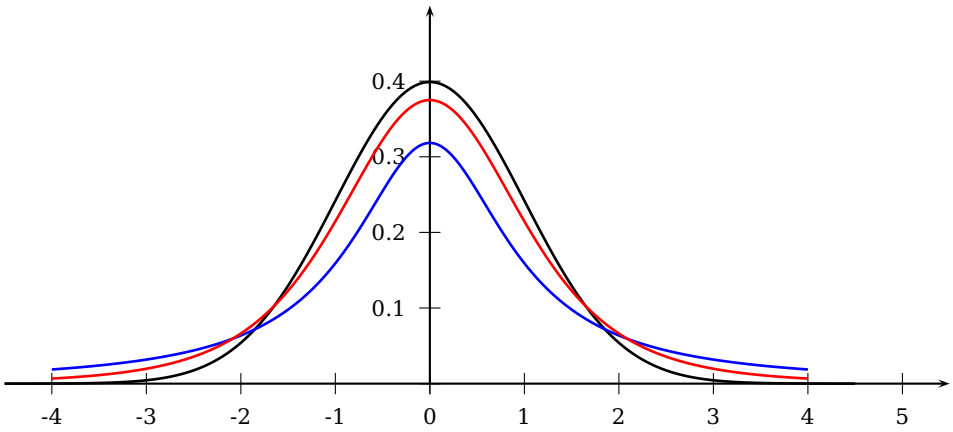
## 7.6 Student's $t$-distribution

A statistical distribution published by William Gosset in 1908 under his pseudonym „Student". The $t$-distribution with parameter $\nu$ 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}
```

## 7.7 $F$-distribution

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

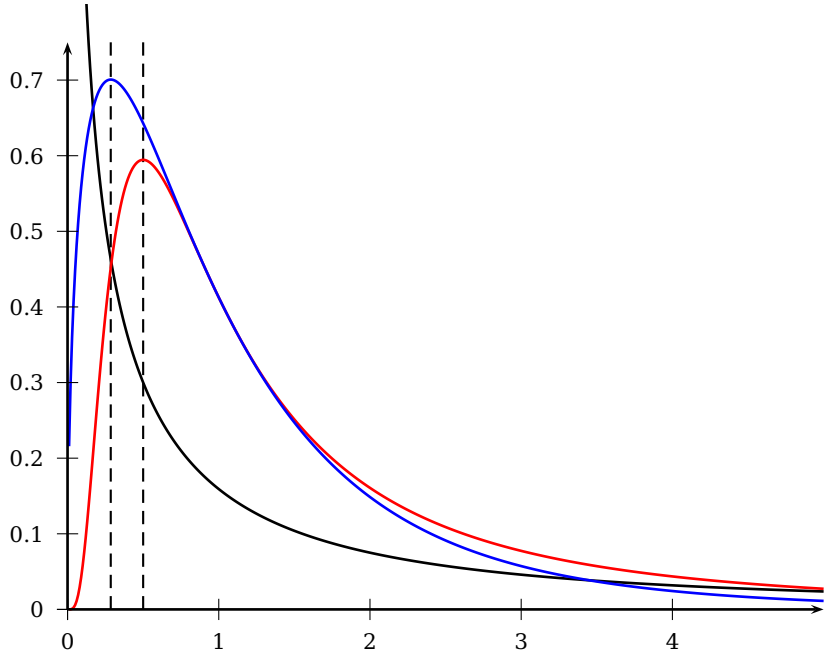
The F-distribution with parameters $\mu$ and $\nu$ has the probability function

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

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*}
```

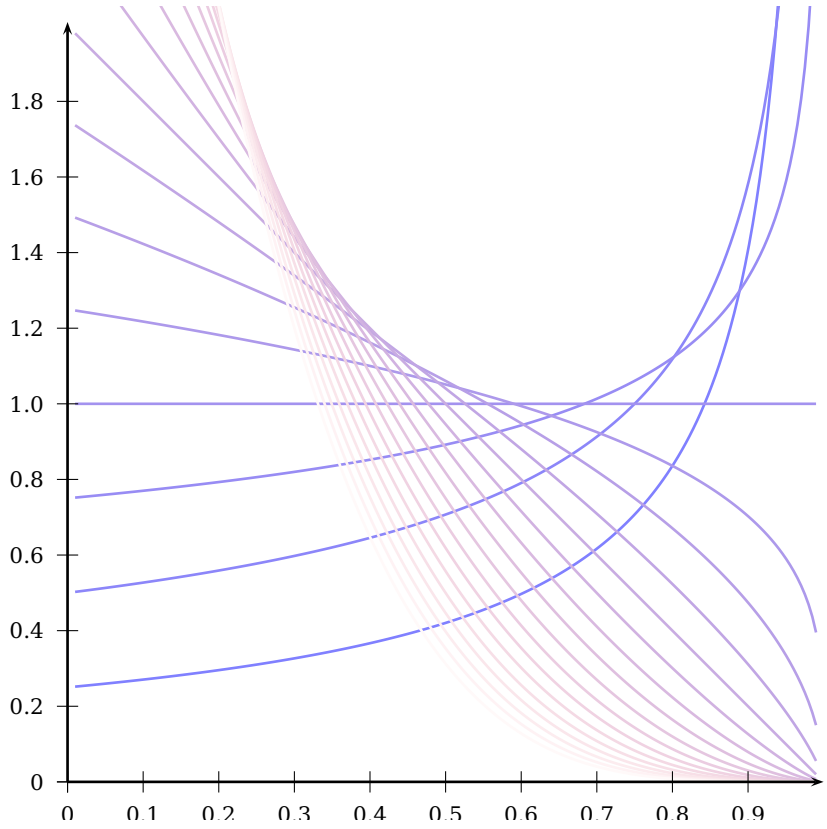
## 7.8 Beta distribution

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 $\alpha$ and $\beta$, 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

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

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*}
```

## 8 \psLame – Lamé Curve, a superellipse

A superellipse is a curve with Cartesian equation

$$\left|\frac{x}{a}\right|^r + \left|\frac{y}{b}\right|^r = 1 \tag{38}$$

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

$$x = a \cdot \cos^{\frac{2}{r}} t \tag{39}$$

$$y = b \cdot \sin^{\frac{2}{r}} t \tag{40}$$

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.

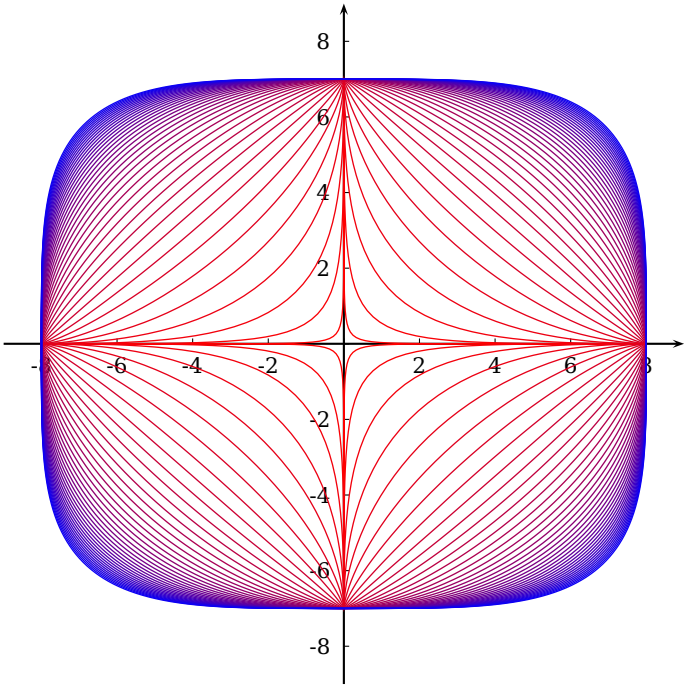
| r | curve type | example |
|---|---|---|
| $\frac{2}{3}$ | (squashed) astroid | |
| 1 | (squashed) diamond | |
| 2 | ellipse | |
| $\frac{5}{2}$ | Piet Hein's „superellipse“ | |

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
```

## 9 \psThomae – the popcorn function

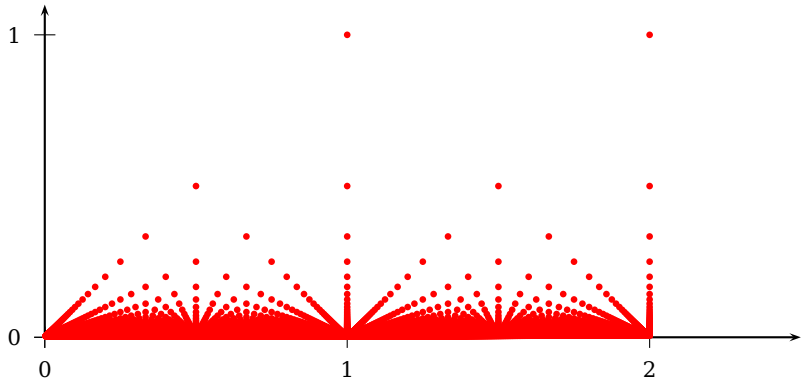
Thomae's function, also known as the popcorn function, the raindrop function, the ruler function or the Riemann function, is a modification of the Dirichlet function. This real-valued function $f(x)$ is defined as follows:

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

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}
```

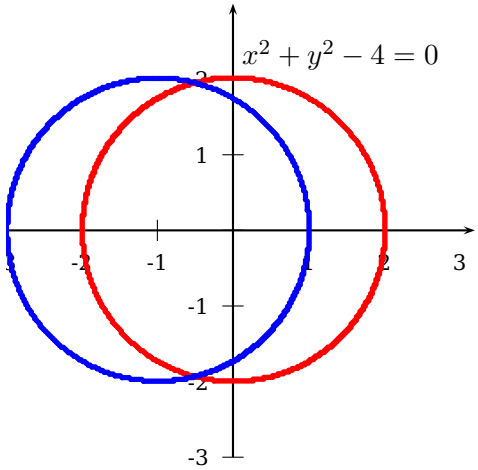
## 10 \psplotImp – plotting implicit defined functions

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 $400 \times 300$ 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.

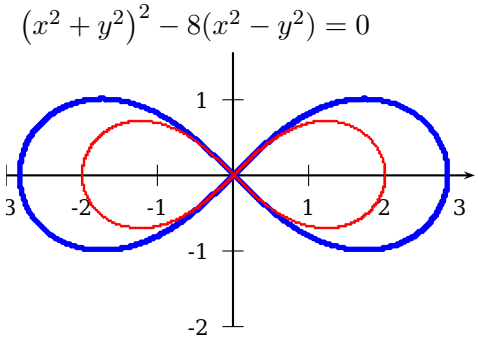
`\psplotImp` `[Options]` (*xMin,yMin*)(*xMax,yMax*){*function f(x,y)*}

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.

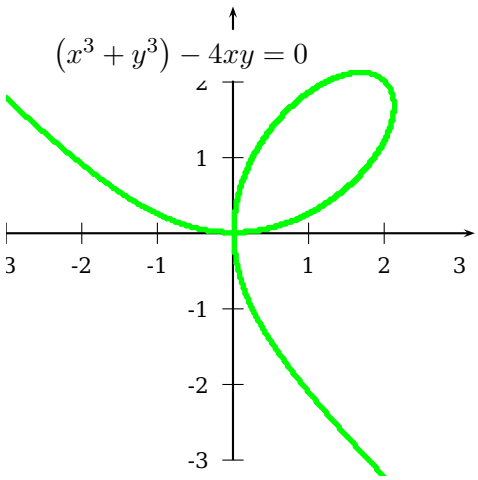
The given area for `\psplotImp` should be **greater** than the given `pspicture` area (see examples).



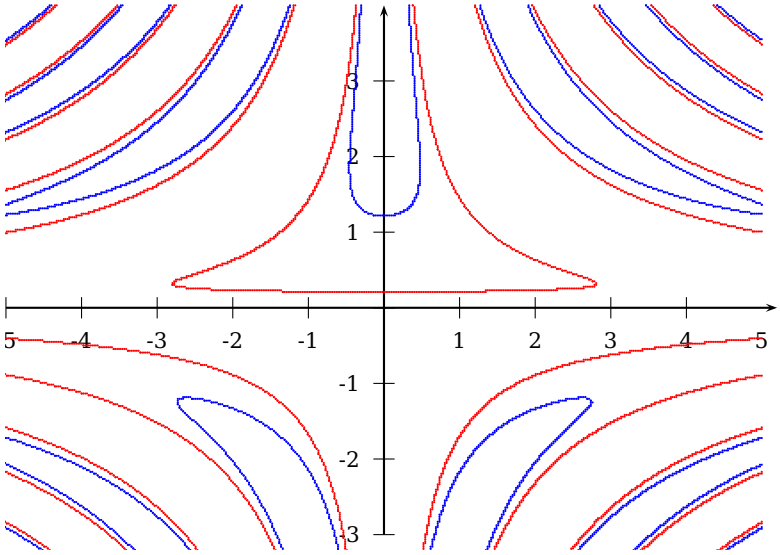

```
\begin{pspicture*}(-3,-3.2)(3.5,3.5)
\psaxes{->}(0,0)(-3,-3)(3.2,3)%
\psplotImp[linewidth=2pt,linecolor=red](-5,-2.1)(5,2.1){ x dup mul y dup mul add 4 sub }
\uput[45](0,2){$x^2+y^2-4=0$}
\psplotImp[linewidth=2pt,linecolor=blue,algebraic](-5,-3)(4,2.4){ (x+1)^2+y^2-4 }
\end{pspicture*}
```




```
\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*}
```

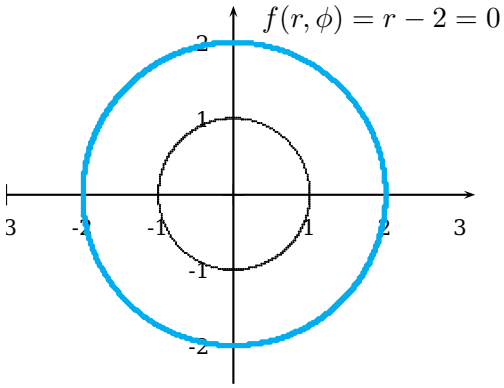



```
\begin{pspicture*}(-3,-3.2)(3.5,3.5)
\psaxes{->}(0,0)(-3,-3)(3.2,3)%
\psplotImp[linewidth=2pt,linecolor=green](-6,-6)(4,2.4){%
    x 3 exp y 3 exp add 4 x y mul mul sub }
\uput*[45](-2.5,2){$\left(x^3+y^3\right)-4xy=0$}
\end{pspicture*}
```

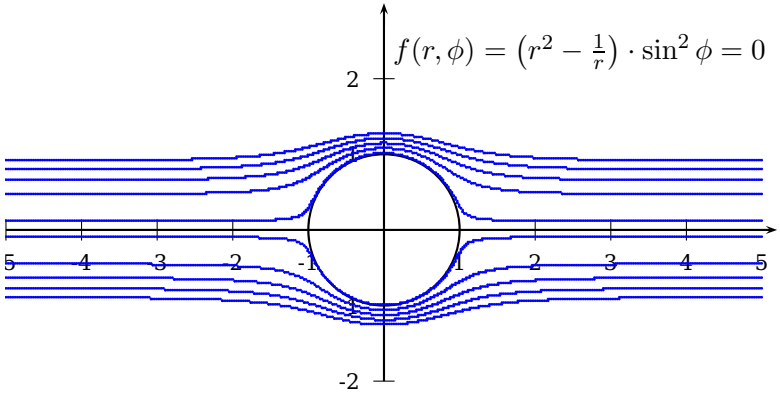



```
\begin{pspicture*}(-5,-3.2)(5.5,4.5)
\psaxes{->}(0,0)(-5,-3)(5.2,4)%
\psplotImp[algebraic,linecolor=red](-6,-4)(5,4){ y*cos(x*y)-0.2 }
\psplotImp[algebraic,linecolor=blue](-6,-4)(5,4){ y*cos(x*y)-1.2 }
\end{pspicture*}
```

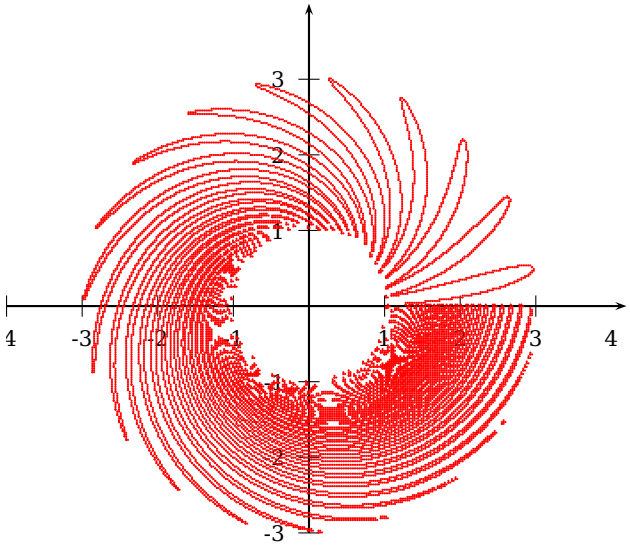
Using the `polarplot` option implies using the variables $r$ and $phi$ for describing the function, $y$ and $x$ are not respected in this case. Using the `algebraic` option for polar plots are also possible (see next example).




```
\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*}
```



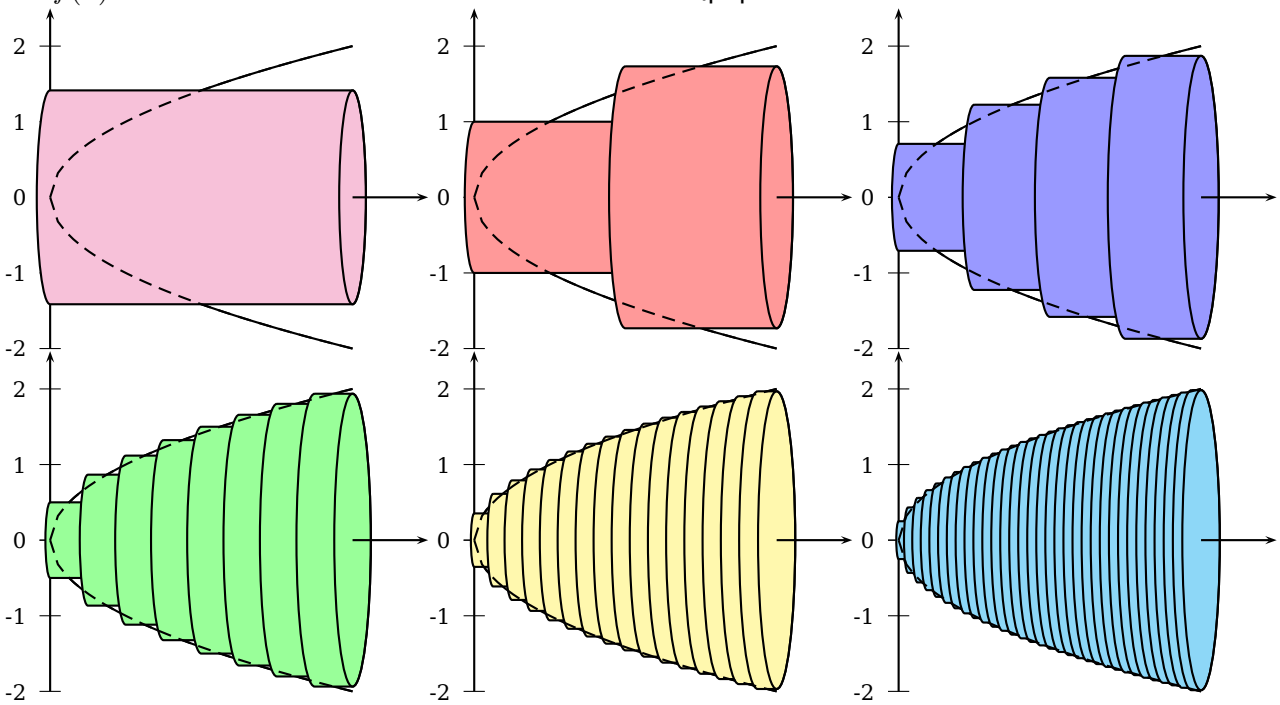

```
\begin{pspicture*}(-4,-3.2)(4.5,4.5)
\psaxes{->}(0,0)(-4,-3)(4.2,4)%
\psplotImp[algebraic,polarplot,linecolor=red](-5,-4)(5,4){ r+cos(phi/r)-2 }
\end{pspicture*}
```

## 11 \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* $f(x)$}

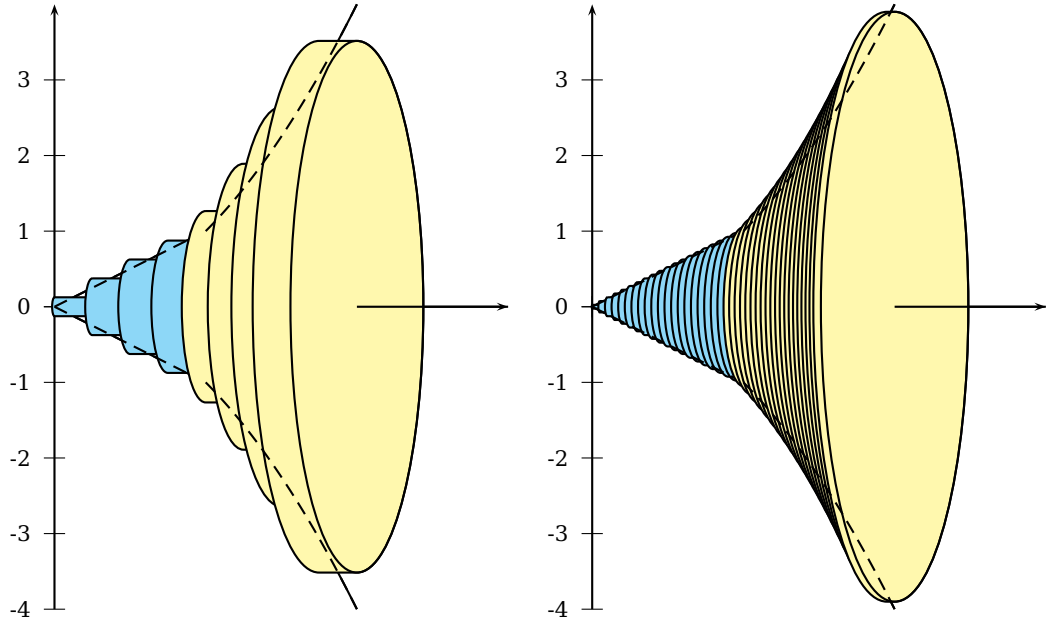
$f(x)$ has to be described as usual for the macro \psplot.




```
\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}
```

## 12 \psPrintValue

This new macro allows to print single values of a math function. It has the syntax

```
\psPrintValue [Options] {PostScript code}
\psPrintValue [algebraic,...] {x value, algebraic code}
```

Important is the fact, that \psPrintValue works on PostScript side. For TEX it is only a box of zero dimension. This is the reason why you have to put it into a box, which reserves horizontal space.

There are the following valid options for \psPrintValue:

| name | value | default | |
|---|---|---|---|
| PSfont | PS font name | Times | only valid PostScript font names are possible, e.g. Times-Roman, Helvetica, Courier, AvantGard, Bookman |
| fontscale | <number> | 10 | the font scale in pt |
| valuewidth | <number> | 10 | the width of the string for the converted real number; if it is too small, no value is printed |
| decimals | <number> | -1 | the number of printed decimals, a negative value prints all possible digits. |
| xShift | <number> | 0 | the x shift in pt for the output, relative to the current point. |
| algebraic | <boolean> | false | function in algebraic notation. |

| x(deg) | $\sin x$ | $\cos x$ | $\sqrt{x}$ | $\sin x + \cos x \sin^2 x + \cos^2 x$ | |
|---|---|---|---|---|---|
| 0 | 0.0 | 1.0 | 0,0 | *1.0* | 1.0 |
| 10 | 0.173648 | 0.984 | 3,16228 | *1.15846* | 1.0 |
| 20 | 0.34202 | 0.939 | 4,47214 | *1.28171* | 1.0 |
| 30 | 0.5 | 0.866 | 5,47723 | *1.36603* | 1.0 |
| 40 | 0.642788 | 0.766 | 6,32456 | *1.40883* | 1.0 |
| 50 | 0.766044 | 0.642 | 7,07107 | *1.40883* | 1.0 |
| 60 | 0.866025 | 0.5 | 7,74597 | *1.36603* | 1.0 |
| 70 | 0.939693 | 0.342 | 8,3666 | *1.28171* | 1.0 |
| 80 | 0.984808 | 0.173 | 8,94427 | *1.15846* | 1.0 |
| 90 | 1.0 | 0.0 | 9,48683 | *1.0* | 1.0 |
| 100 | 0.984808 | -0.173 | 10,0 | *0.81116* | 1.0 |
| 110 | 0.939693 | -0.342 | 10,4881 | *0.597672* | 1.0 |
| 120 | 0.866025 | -0.5 | 10,9545 | *0.366025* | 1.0 |
| 130 | 0.766044 | -0.642 | 11,4018 | *0.123257* | 1.0 |
| 140 | 0.642788 | -0.766 | 11,8322 | *-0.123257* | 1.0 |
| 150 | 0.5 | -0.866 | 12,2474 | *-0.366025* | 1.0 |
| 160 | 0.34202 | -0.939 | 12,6491 | *-0.597672* | 1.0 |
| 170 | 0.173648 | -0.984 | 13,0384 | *-0.81116* | 1.0 |

```
\psset{fontscale=12}
\makebox[2em]{x(deg)} \makebox[5em]{$\sin x$} \makebox[4em]{$\cos x$}\hspace{1em}
```

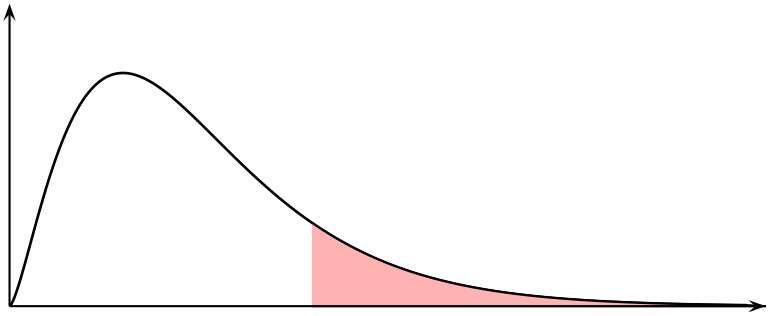
```
\makebox[5em]{$\sqrt x$}\makebox[7em]{$\sin x+\cos x$}\makebox[6em]{$\sin^2 x+\cos^2
   x$}\\[3pt]
\multido{\iA=0+10}{18}{
 \makebox[1em]{\iA}
 \makebox[5em]{\psPrintValue[PSfont=Helvetica,xShift=-10]{\iA\space sin}}
 \makebox[4em][r]{\psPrintValue[PSfont=Courier,fontscale=10,decimals=3,xShift=-20]{\
   iA\space cos}}\hspace{1em}
 \makebox[5em]{\psPrintValue[comma,valuewidth=15,linecolor=blue,PSfont=AvantGarde]{\
   iA\space sqrt}}
 \makebox[7em]{\psPrintValue[PSfont=Times-Italic]{\iA\space dup sin exch cos add}}
 \makebox[6em]{\psPrintValue[PSfont=Palatino-Roman]{\iA\space dup sin dup mul exch
   cos dup mul add}}\\}
```

With enabled `algebraic` option there must be two arguments, separated by a comma. The first one is the x value as a number, which can also be PostScript code, which leaves a number on the stack. The second part is the function described in algebraic notation. Pay attention, in algebraic notation angles must be in radian and not degrees.

| x(deg) | $\sin x$ | $\cos x$ | $\sqrt{x}$ | $\sin x + \cos x \sin^2 x + \cos^2 x$ | |
|---|---|---|---|---|---|
| 0.0 | 0.0 | 1.0 | 0,0 | *1.0* | 1.0 |
| 0.1 | 0.0998334 | 0.995 | 0,316228 | *1.09484* | 1.0 |
| 0.20001 | 0.198679 | 0.98 | 0,447225 | *1.17874* | 1.0 |
| 0.30002 | 0.295539 | 0.955 | 0,547741 | *1.25087* | 1.0 |
| 0.40002 | 0.389437 | 0.921 | 0,632471 | *1.31049* | 1.0 |
| 0.50003 | 0.479452 | 0.877 | 0,707128 | *1.35702* | 1.0 |
| 0.60004 | 0.564675 | 0.825 | 0,774622 | *1.38999* | 1.0 |
| 0.70004 | 0.644248 | 0.764 | 0,836684 | *1.40906* | 1.0 |
| 0.80005 | 0.717391 | 0.696 | 0,894455 | *1.41406* | 1.0 |
| 0.90005 | 0.783358 | 0.621 | 0,94871 | *1.40493* | 1.0 |
| 1.00006 | 0.841503 | 0.54 | 1,00003 | *1.38176* | 1.0 |
| 1.10007 | 0.891239 | 0.453 | 1,04884 | *1.34477* | 1.0 |
| 1.20007 | 0.932064 | 0.362 | 1,09548 | *1.29436* | 1.0 |
| 1.30008 | 0.96358 | 0.267 | 1,14021 | *1.231* | 1.0 |
| 1.40009 | 0.985465 | 0.169 | 1,18325 | *1.15534* | 1.0 |
| 1.50009 | 0.997501 | 0.07 | 1,22478 | *1.06815* | 1.0 |
| 1.6001 | 0.999571 | -0.029 | 1,26495 | *0.970271* | 1.0 |
| 1.7001 | 0.991652 | -0.128 | 1,30388 | *0.862708* | 1.0 |

```
\psset{algebraic, fontscale=12}% All functions now in algebraic notation
\makebox[2em]{x(deg)} \makebox[5em]{$\sin x$} \makebox[4em]{$\cos x$}\hspace{1em}
\makebox[5em]{$\sqrt x$}\makebox[7em]{$\sin x+\cos x$}\makebox[6em]{$\sin^2 x+\cos^2
   x$}\\[3pt]
\multido{\rA=0+0.1}{18}{\makebox[1em]{\rA}
 \makebox[5em]{\psPrintValue[PSfont=Helvetica,xShift=-10]{\rA, sin(x)}}
 \makebox[4em][r]{\psPrintValue[PSfont=Courier,fontscale=10,decimals=3,xShift=-20]{\
   rA,cos(x)}}\hspace{1em}
 \makebox[5em]{\psPrintValue[comma,valuewidth=15,linecolor=blue,PSfont=AvantGarde]{\
   rA,sqrt(x)}}
 \makebox[7em]{\psPrintValue[PSfont=Times-Italic]{\rA,sin(x)+cos(x)}}
 \makebox[6em]{\psPrintValue[PSfont=Palatino-Roman]{\rA,sin(x)^2+cos(x)^2}}\\}
```

## 13 Examples



```
\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}
```

## 14 List of all optional arguments for `pst-func`

| Key | Type | Default |
|---|---|---|
| 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 |
| sigma | ordinary | 0.5 |
| mue | ordinary | 0 |
| nue | ordinary | 1 |
| Simpson | ordinary | 5 |
| PSfont | ordinary | Times-Roman |
| valuewidth | ordinary | 10 |
| fontscale | ordinary | 10 |
| decimals | ordinary | -1 |
| round | boolean | true |
| science | boolean | true |
| printValue | boolean | true |
| barwidth | ordinary | 1 |
| alpha | ordinary | 0.5 |
| beta | ordinary | 0.5 |
| radiusA | ordinary | 1 |
| radiusB | ordinary | 1 |
| stepFactor | ordinary | 0.67 |
| envelope | boolean | true |

## References

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

[2] Michel Goosens, Frank Mittelbach, Sebastian Rahtz, Denis Roegel, and Herbert Voß. *The LaTeX Graphics Companion*. Addison-Wesley Publishing Company, Reading, Mass., 2007.

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

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

[5] Herbert Voß. *Chaos und Fraktale selbst programmieren: von Mandelbrotmengen über Farbmanipulationen zur perfekten Darstellung*. Franzis Verlag, Poing, 1994.

[6] Herbert Voß. Die mathematischen Funktionen von PostScript. *Die TEXnische Komödie*, 1/02, March 2002.

[7] Herbert Voß. *LATEX in Mathematik und Naturwissenschaften*. Franzis-Verlag, Poing, 2006.

[8] Herbert Voß. *PSTricks – Grafik für TEX und LATEX*. DANTE – Lehmanns, Heidelberg/Hamburg, 5. edition, 2008.

[9] Eric Weisstein. *Wolfram MathWorld*. http://mathworld.wolfram.com, 2007.

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

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

[12] Timothy van Zandt. *pst-plot: Plotting two dimensional functions and data*. CTAN:graphics/pstricks/generic/pst-plot.tex, 1999.

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

## Index


**A**
algebraic, 47, 49, 53, 54
AvantGard, 53

**B**
Bézier, 4
Bézier curve, 4
barwidth, 30
Bayesian analysis, 43
Bernstein-Bézier, 4
black, 11
Bookman, 53

**C**
Chebyshev polynomials, 6
ChebyshevT, 6
\ChebyshevT, 6
ChebyshevU, 6
\ChebyshevU, 6
coeff, 11
coefficients, 10
constI, 21
constII, 21
cosCoeff, 18
Courier, 53
curve, 34

**D**
dashed, 11
decimals, 53
density function, 41
Derivation, 11
Dirichlet, 46
dZero, 11

**E**
envelope, 16
Environment
– pspicture, 11, 47
– pspicture*, 47
epsZero, 11

**F**
File
– pst-math.pro, 28
fontscale, 53
Fourier sums, 18

**G**
GAMMA, 28
GAMMALN, 28

**H**
Helvetica, 53

**I**
InsideArrow, 3

**K**
Keyvalue
– black, 11
– curve, 34
– dashed, 11
Keyword
– algebraic, 47, 49, 53, 54
– barwidth, 30
– coeff, 11
– constI, 21
– constII, 21
– cosCoeff, 18
– decimals, 53
– Derivation, 11
– dZero, 11
– envelope, 16
– epsZero, 11
– fontscale, 53
– InsideArrow, 3
– markZeros, 11, 30
– mue, 29
– plotpoints, 4, 21, 26
– plotstyle, 34
– polarplot, 49
– printValue, 30
– PSfont, 53
– radiusA, 44
– radiusB, 44
– showpoints, 34
– sigma, 29
– Simpson, 26, 29
– sinCoeff, 18

– stepFactor, 47
– tEnd, 16
– tStart, 16
– valuewidth, 53
– xShift, 10, 11, 53
– xStart, 26
– xunit, 30
– zeroLineColor, 11
– zeroLineStyle, 11
– zeroLineTo, 11
– zeroLineWidth, 11

**L**
Lamé, 44

**M**
Macro
– \ChebyshevT, 6
– \ChebyshevU, 6
– \psBernstein, 16
– \psBessel, 21
– \psBetaDist, 43
– \psBezier, 4
– \psbezier, 3
– \psBinomial, 30
– \psBinomialN, 30, 34
– \psChiIIDist, 40
– \psCi, 24
– \psConv, 26
– \psCumIngegral, 26
– \psCumIntegral, 26
– \psFDist, 42
– \psFourier, 18
– \psGammaDist, 39
– \psGauss, 29
– \psGaussI, 26, 29
– \psIntegral, 26
– \psLame, 44
– \psplot, 51
– \psplotImp, 47
– \psPoisson, 36
– \psPolynomial, 10, 11, 18
– \psPrintValue, 30, 53
– \psset, 12, 29
– \psSi, 24
– \pssi, 24
– \psTDist, 41
– \psThomae, 46
– \psVolume, 51
markZeros, 11, 30
mue, 29

**P**
Package
– pst-func, 3, 6, 10, 28
– pst-math, 3, 28
– pst-plot, 3, 21
– pst-xkey, 3
– pstricks, 3, 21
– pstricks-add, 3, 47
parametric plot, 44
Piet Hein, 44
plotpoints, 4, 21, 26
plotstyle, 34
polarplot, 49
polynomial function, 10
popcorn function, 46
PostScript
– AvantGard, 53
– Bookman, 53
– ChebyshevT, 6
– ChebyshevU, 6
– Courier, 53
– GAMMA, 28
– GAMMALN, 28
– Helvetica, 53
– Times-Roman, 53
print, 53
printValue, 30
\psBernstein, 16
\psBessel, 21
\psBetaDist, 43
\psBezier, 4
\psbezier, 3
\psBinomial, 30
\psBinomialN, 30, 34
\psChiIIDist, 40
\psCi, 24
\psConv, 26
\psCumIngegral, 26
\psCumIntegral, 26
\psFDist, 42
PSfont, 53
\psFourier, 18

\psGammaDist, 39
\psGauss, 29
\psGaussI, 26, 29
\psIntegral, 26
\psLame, 44
pspicture, 11, 47
pspicture*, 47
\psplot, 51
\psplotImp, 47
\psPoisson, 36
\psPolynomial, 10, 11, 18
\psPrintValue, 30, 53
\psset, 12, 29
\psSi, 24
\pssi, 24
pst-func, 3, 6, 10, 28
pst-math, 3, 28
pst-math.pro, 28
pst-plot, 3, 21
pst-xkey, 3
\psTDist, 41
\psThomae, 46
pstricks, 3, 21
pstricks-add, 3, 47
\psVolume, 51

**R**
radiusA, 44
radiusB, 44
raindrop function, 46
Riemann function, 46
rotated function, 51
ruler function, 46

**S**
showpoints, 34
sigma, 29
Simpson, 26, 29
sinCoeff, 18
statistical distribution, 41
stepFactor, 47
superellipse, 44
Superellipses, 44

**T**
tEnd, 16
Thomae's function, 46
Times-Roman, 53
trapezoid function, 27
tStart, 16

**V**
Value
– curve, 34
valuewidth, 53

**W**
William Gosset, 41

**X**
xShift, 10, 11, 53
xStart, 26
xunit, 30

**Z**
zeroLineColor, 11
zeroLineStyle, 11
zeroLineTo, 11
zeroLineWidth, 11