

pst-plot — plotting functions in “pure” L^AT_EX

Commonly, one wants to simply plot a function as a part of a L^AT_EX document. Using some postscript tricks, you can make a graph of an arbitrary function in one variable including implicitly defined functions.

The commands described on this worksheet require that the following lines appear in your document header (before `\begin{document}`).

```
\usepackage{pst-plot}
\usepackage{pstricks}
```

The full `pstricks` manual (including `pst-plot` documentation) is available at:¹
<http://www.risc.uni-linz.ac.at/institute/systems/documentation/TeX/tex/>

A good page for showing the power of what you can do with `pstricks` is: <http://www.pstricks.de/>.²

Reverse Polish Notation (postfix notation) Reverse polish notation (RPN) is a modification of polish notation which was a creation of the logician Jan Lukasiewicz (advisor of Alfred Tarski). The advantage of these notations is that they do not require parentheses to control the order of operations in an expression. The advantage of this in a computer is that parsing a RPN expression is trivial, whereas parsing an expression in standard notation can take quite a bit of computation.

At the most basic level, an expression such as $1 + 2$ becomes `1 2 +`. For more complicated expressions, the concept of a stack must be introduced. The stack is just the list of all numbers which have not been used yet. When an operation takes place, the result of that operation is left on the stack. Thus, we could write the sum of all integers from 1 to 10 as either,

$$1\ 2\ +\ 3\ +\ 4\ +\ 5\ +\ 6\ +\ 7\ +\ 8\ +\ 9\ +\ 10\ +$$

or

$$1\ 2\ 3\ 4\ 5\ 6\ 7\ 8\ 9\ 10\ +\ +\ +\ +\ +\ +\ +\ +\ +$$

In both cases the result 55 is left on the stack. In the first example the numbers are added from smallest to largest, however in the second example the numbers are added from largest to smallest. Here are some sample conversions. Try to do the conversions yourself before looking at the answers. (Note: there is frequently more than one correct answer)

$3(1 + 2)$	<code>1 2 + 3 *</code>
$3x^5 - 1$	<code>3 x 5 ^ 1 -</code>
$\frac{3x^2 - 2x}{15x - 2}$	<code>3 x 2 ^ * 2 x * - 15 x * 2 - /</code>

Postscript functions Postscript uses reverse polish notation, but does not use the standard mathematical symbols `+ - * / ^`. Instead, Postscript uses the following commands³.

The simple functions are:

operation:	<code>+</code>	<code>-</code>	<code>*</code>	<code>/</code>	<code>^</code>	<code>√</code>	<code>ln</code>	<code>log</code>	<code>abs</code>
PS command:	<code>add</code>	<code>sub</code>	<code>mul</code>	<code>div</code>	<code>exp</code>	<code>sqrt</code>	<code>ln</code>	<code>log</code>	<code>abs</code>

Additionally, the `neg` command negates the top number. `idiv` is integer division, it divides two integers and returns the integer part of their quotient. `rand` returns a random integer in the interval $[0, 230]$. `mod` returns the remainder of dividing the two numbers, that is $5\ 3\ \text{mod} \rightarrow 2$, but $-5\ 3\ \text{mod}\ 2 \rightarrow -2$.

¹If this site disappears, then a search for “teTeX A Documentation Guide” will find a mirror of this page.

²Including an animation of the construction of a regular 17-gon, all in pure L^AT_EX and `pstricks`!

³A complete list of commands can be found at <http://mdwconsulting.mdwserver.net/postscript/postscript-operators/>

The trig functions `sin`, `cos`, and `tan` take units in degrees. `atan` takes two arguments a and b and returns the angle (in degrees between 0 and 360) whose tangent is $\frac{a}{b}$. Examples: `0 1 atan` \rightarrow 0.0; `1 0 atan` \rightarrow 90.0; `0 -1 atan` \rightarrow 180.0; `-1 0 atan` \rightarrow 270.0

There are some stack manipulators which may be useful. `dup` duplicates the top element. `exch` swaps the top two elements.

The integerizing commands `ceiling`, `floor`, `round`, and `truncate` (round towards zero) are available.

The examples from above become:

```

3(1 + 2)      1 2 add 3 mul
3x5 - 1     3 x 5 exp 1 sub
 $\frac{3x^2-2x}{15x-2}$   3 x 2 exp mul 2 x mul sub 15 x mul 2 sub div

```

Postscript plotting Plots should occur within the `pspicture` environment. When you `\begin{pspicture}` the next two things should be the coordinates of the bottom left and top right corners of the picture. The values may be negative. Inside the `pspicture` you have access to many commands for drawing objects (which are described in detail in the `pstricks` documentation). Some of the most common things you will use are `\psline`, `\psplot`, `\rput`, `\pscicle`, `\psgrid`, and `\pscurve`. Here are some common arguments given to these commands.

plotstyle=style Legal styles are `dots`, `line`, `polygon` (a plot closed by a line), `curve`, `ecurve`, `ccurve` (a plot closed by a cubic splice). (default: `line`)

linecolor=color `color` is one of `black`, `darkgray`, `gray`, `lightgray`, `white`, `red`, `green`, `blue`, `cyan`, `magenta`, `yellow`, or one defined by `\new...color`. (default: `black`)

fillcolor=color see above.

linewidth=width `width` must include units. (default: `.8pt`)

plotpoints=num Number of points to interpolate from. Alter to increase smoothness, or decrease compilation time. (default: `50`)

fillstyle=style Valid fill styles are `none`, `solid`, `vlines`, `hlines`, `crosshatch` (default: `none`)

Units There are three unit definitions that can be made. They are `xunit`, `yunit`, and `unit` (which sets both the x and y units). These units need to be set before you begin a `pspicture`, so you should set them using `\psset` just before you `\begin{pspicture}`. (Note: generally it works best if `\psset` is called within some other environment so that your temporary values don't leak out to other pictures.)

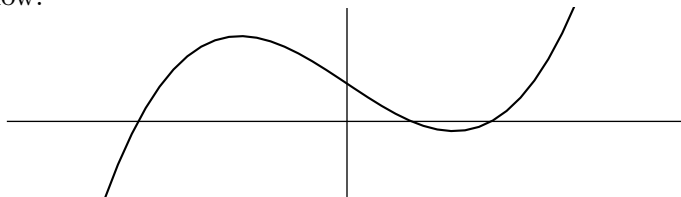
Example 1 My favorite test question:

```

\begin{center}
\psset{xunit=3cm,yunit=1cm}
\begin{pspicture*}(-1.5,-1)(1.5,1.5)
  \psline[linewidth=.5pt](-1.5,0)(1.5,0) % x-axis
  \psline[linewidth=.5pt](0,-1)(0,1.5)   % y-axis
  \psplot{-1.5}{1.5}{3 x x x mul mul mul 2 x mul sub .5 add } % 3x3 - 2x + 1/2
\end{pspicture*}
\end{center} % xunit and yunit are reset when we leave the \end{center}

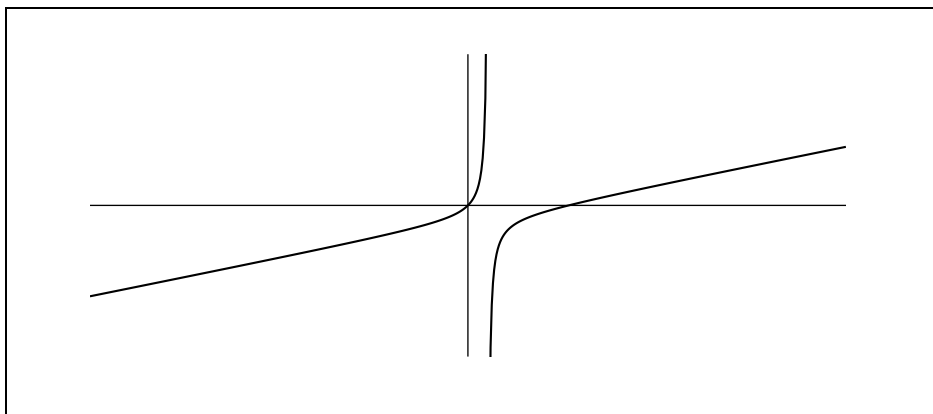
```

1. Draw the graph of a function whose derivative is given below.



Example 2 Using the `*` form of `pspicture` chops off singularities (We also used it in Example 1). PSTricks also insists upon creating continuous functions, so sometimes we need to split our graph into pieces to avoid running over discontinuities. (Try this without the `*` and without splitting the plot to see the difference.)

```
\psset{unit=2cm}
\begin{pspicture*}(-2.5,-1)(2.5,1)
  \psline[linewidth=.5pt](-2.5,0)(2.55,0) % x-axis
  \psline[linewidth=.5pt](0,-1)(0,1)      % y-axis
  \psplot[plotpoints=300]{-2.5}{.133}
    {3 x 2 exp mul 2 x mul sub 15 x mul 2 sub div}
  \psplot[plotpoints=300]{.134}{2.5}
    {3 x 2 exp mul 2 x mul sub 15 x mul 2 sub div}
\end{pspicture*}
```



Note: You can also define temporary commands so that you don't have to re-type your function:

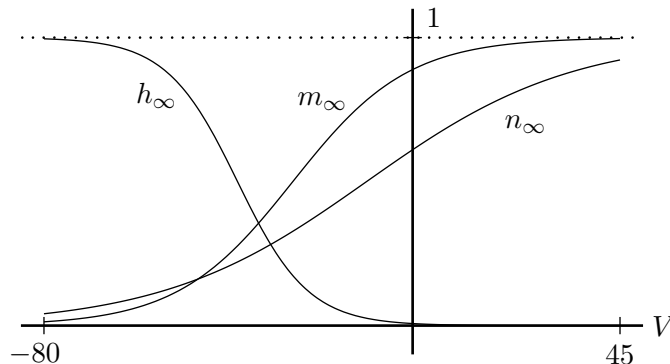
```
\psset{unit=2cm}
\begin{pspicture*}(-2.5,-1)(2.5,1)
  \def\F{3 x 2 exp mul 2 x mul sub 15 x mul 2 sub div}
  \psline[linewidth=.5pt](-2.5,0)(2.55,0) % x-axis
  \psline[linewidth=.5pt](0,-1)(0,1)      % y-axis
  \psplot[plotpoints=300]{-2.5}{.133}{\F}
  \psplot[plotpoints=300]{.134}{2.5}{\F}
\end{pspicture*}      % \F is forgotten when we leave pspicture
```

Example 3 Asymptotic behavior of Sodium and Potassium gates in the Hodgkin-Huxley model.

```

\begin{center}
  \psset{xunit=0.024in,yunit=1.5in}           % sizing ratio
  \begin{pspicture}(-80,-0.1)(45,1.1)         % bounding box
    \psline[linewidth=1pt]{-}(-85,0)(50,0)   % x-axis
    \psline[linewidth=1pt]{-}(0,-0.1)(0,1.1) % y-axis
    \psline[linewidth=0.5pt]{-}(-2,1)(2,1)   % tick mark
    \psline[linewidth=0.5pt]{-}(-80,-0.03)(-80,0.03) % tick mark
    \psline[linewidth=0.5pt]{-}(45,-0.03)(45,0.03) % tick mark
    \psline[linewidth=1pt,linestyle=dotted]{-}(-85,1)(50,1) % dotted line
    \rput[l](52,0){$V$}                       % label for x-axis and location
    \rput[l](-25,0.78){$m_{\infty}$}         % label
    \rput[l](-60,0.8){$h_{\infty}$}         % label
    \rput[l](20,0.7){$n_{\infty}$}         % label
    \rput[l](3,1.07){$1$}                    % label
    \rput[c](-82,-0.1){$-80$}                % label
    \rput[c](45,-0.1){$45$}                  % label
    \psplot[plotstyle=curve,linewidth=.5pt]{-80}{45}{1 1
      2.71828182846 -0.08 x 26 add mul exp add div} % m infinity
    \psplot[plotstyle=curve,linewidth=.5pt]{-80}{45}{1 1
      2.71828182846 0.13 x 38 add mul exp add div} % h infinity
    \psplot[plotstyle=curve,linewidth=.5pt]{-80}{45}{1 1
      2.71828182846 -0.045 x 10 add mul exp add div} % n infinity
  \end{pspicture}
\end{center}

```



Example 4 Shading the region between two curves.

```

2.\ \ Compute the area of the shaded region
\begin{center}
  \def\ff{\psplot{-0.1}{5}{x 180 mul 3.1415 div sin}}
  \def\flabel{$y=\sin x$}
  \def\gf{\psplot{-0.1}{5}{x 180 mul 3.1415 div cos}}
  \def\glabel{$y=\cos x$}
  \psset{unit=1.5cm}
  \begin{pspicture}(-.1,-1.2)(5,1.2)

```

Since we have to use each function twice (once to plot it, and once to define the region), we make temporary definition (they will go away at the `\end{center}`) so that it is easier to keep the two copies in sync. We also set a unit length, and start the picture environment.

```

\psclip{
  \pscustom[linestyle=none]{
    \moveto(0,-1)\f\lineto(5,-1)}
  \pscustom[linestyle=none]{
    \moveto(0,1)\g\lineto(5,1)}
  }
\psframe*[linecolor=gray](0,-1)(4,1)
\endpsclip

```

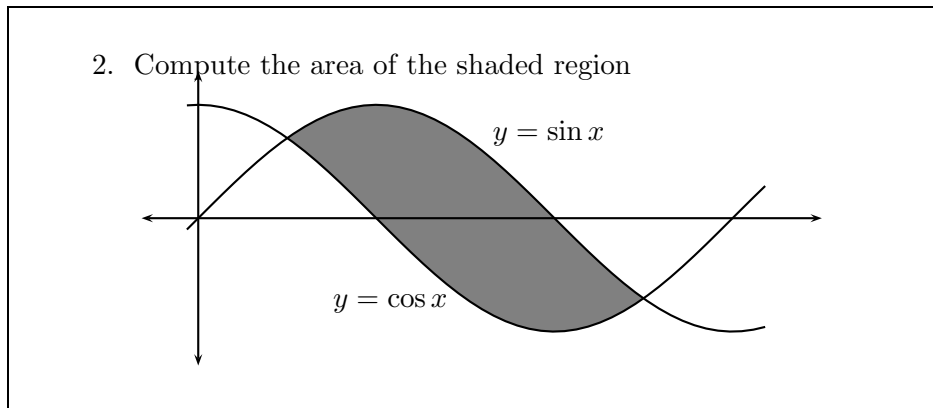
Here we only do the shading. The `\psframe` command creates a solid block of shaded background (whose bottom left coordinate is $(0, -1)$ and top right coordinate is $(4, 1)$). The things in the braces of the `\psclip` select the parts of the gray background that we want to keep. The first `\pscustom` cuts away everything except for the area below $\sin x$ (the `\moveto` and `\lineto` extend `\f` down to the bottom of the image). Similarly, the second `\pscustom` cuts away everything except for the area above $\cos x$. All that is left is the area between the curves.

```

\psline{<->}(0,-1.3)(0,1.3)
\psline{<->}(-.5,0)(5.5,0)
\f\rput[1](2.6,.75){\flabel}
\g\rput[r](2.2,-.75){\glabel}
\end{pspicture}
\end{center}

```

Finally, we actually plot the axes and functions and place the labels. The `[r]` and `[1]` place the labels to the left and right of the specified point respectively.



Grids The `\psgrid` command creates grid lines easily. The format is:

```

\psgrid( $l_x, l_y$ )( $min_x, min_y$ )( $max_x, max_y$ )

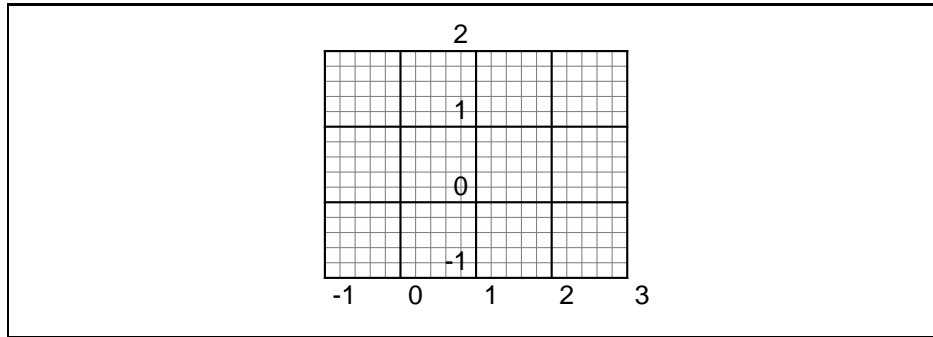
```

The *min* and *max* numbers are pretty clear, the *l* numbers are specify where the labels should be placed. The *x*-labels are placed near the line $y = l_y$ and the *y*-labels are placed near the line $x = l_x$. Unfortunately, all of these numbers need to be integers (non-integer values tend to give undesirable results, but you may get lucky). An example:

```

\begin{center}
\begin{pspicture}(-1,-1)(3,2)
  \psgrid(1,-1)(-1,-1)(3,2)
\end{pspicture}
\end{center}

```



Most of the time you will want $(l_x, l_y) = (0, 0)$ or $(l_x, l_y) = (min_x, min_y)$ (which is the same thing as leaving off (l_x, l_y)). Some of the useful grid options (in addition to the standard options):

gridwidth=*size* width of the grid lines (default: .8pt)

griddots=*num* dotted grid lines with *num* dots per division. 0 gives solid lines. (default: 0)

gridlabels=*size* size of number labels (default: 10pt)

gridcolor=*color* the grid color (default: black)

subgriddiv=*int* number of grid subdivisions (default: 5)

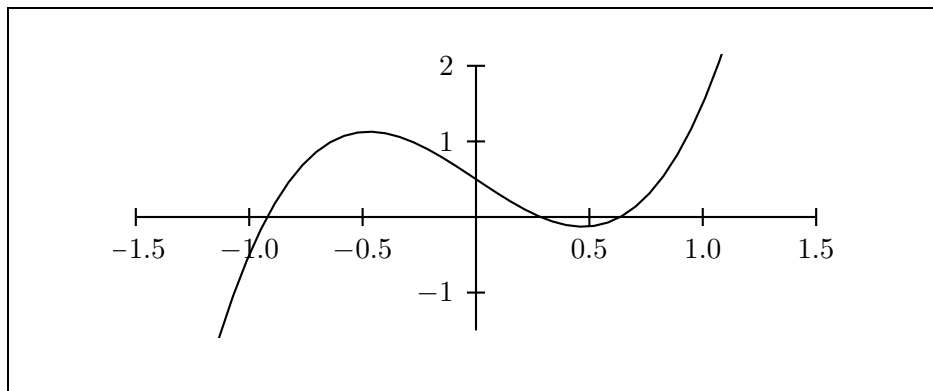
subgridwidth=*size* width of the sub-grid lines (default: .4pt)

subgriddots=*num* dotted sub-grid lines with *num* dots per division. (default: 0)

subgridcolor=*color* the grid color (default: gray)

Example 5 If, however, all you want is simple automatic labeling, use the `\psaxes` command which has the same syntax as `\psgrid`. (Notice that I had to expand the bounds of the picture so that the labels did not get chopped off!⁴)

```
\begin{center}
\psset{xunit=3cm,yunit=1cm}
\begin{pspicture*}(-1.6,-1.6)(1.6,2.15)
  \psaxes[Dx=.5](0,0)(-1.5,-1.5)(1.5,2)
  \psplot[-1.5]{1.5}{3 x x x mul mul mul 2 x mul sub .5 add } % 3x^3 - 2x + 1/2
\end{pspicture*}
\end{center}
```



⁴ $(-1.6, -1.6)(1.6, 2.15)$ instead of $(-1.5, -1.5)(1.5, 2)$ as we did before