Examples of embedding Sage in LaTeX

March 12, 2008

1 Inline Sage, code blocks

This is an example $2 + 2 = 4$. If you raise the current year mod 100 (8) to the power of the current day (12), you get 68719476736. Also, 2008 modulo 42 is 34.

Code block which uses a variable `s` to store the solutions:

```
var('a,b,c')
eqn = [a+b*c==1, b-a*c==0, a+b==5]
s = solve(eqn, a,b,c)
```

Solutions of eqn $= [bc + a = 1, b - ac = 0, b + a = 5]$:

$$\left[a = \frac{25\sqrt{79}i + 25}{6\sqrt{79}i - 34}, b = \frac{5\sqrt{79}i + 5}{\sqrt{79}i + 11}, c = \frac{\sqrt{79}i + 1}{10}\right]$$

$$\left[a = \frac{25\sqrt{79}i - 25}{6\sqrt{79}i + 34}, b = \frac{5\sqrt{79}i - 5}{\sqrt{79}i - 11}, c = \frac{1 - \sqrt{79}i}{10}\right]$$

Now we evaluate the following block:

```
E = EllipticCurve("37a")
```

You can't do assignment inside `\sage` macros, since Sage doesn't know how to typeset the output of such a thing. So you have to use a code block. The elliptic curve E given by $y^2 + y = x^3 - x$ has discriminant 37.

You can do anything in a code block that you can do in Sage and/or Python. Here we save an elliptic curve into a file.

```
try:
    E = load('E2')
except IOError:
    E = EllipticCurve([1,2,3,4,5])
    E.anlist(100000)
    E.save('E2')
```

The 9999th Fourier coefficient of $y^2 + xy + 3y = x^3 + 2x^2 + 4x + 5$ is -27.

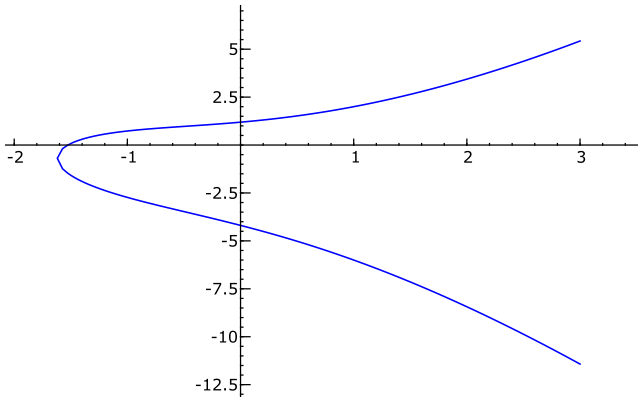
The following code block doesn't appear in the typeset file... but we can refer to whatever we did in that code block: $e = 7$.

```
var('x')
f = log(sin(x)/x)
```

The Taylor Series of f is: $\frac{-x^2}{6} - \frac{x^4}{180} - \frac{x^6}{2835} - \frac{x^8}{37800} - \frac{x^{10}}{467775}$.

2 Plotting

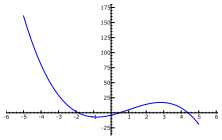
Here's a plot of the elliptic curve E.



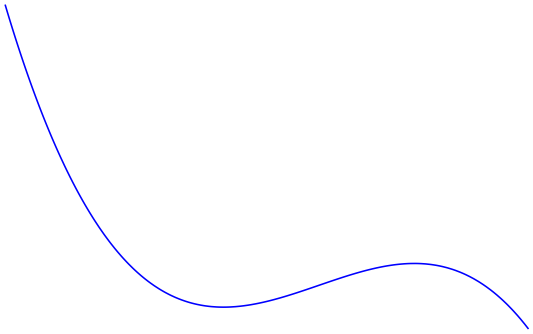
You can use variables to hold plot objects and do stuff with them.

```
p = plot(f, x, -5, 5)
```

Here's a small plot of f from -5 to 5, which I've centered:



On second thought, use the default size of 3/4 the `\textwidth` and don't use axes:

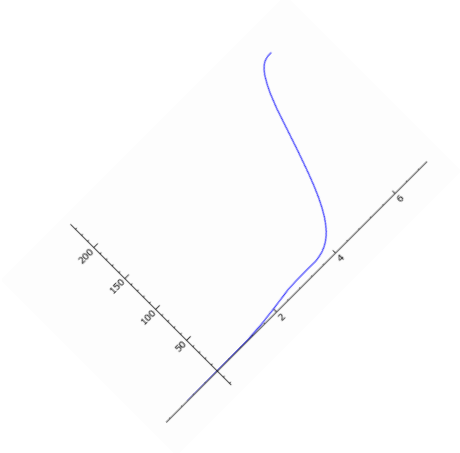


Remember, you're using Sage, and can therefore call upon any of the software packages Sage is built out of.

```
f = maxima('sin(x)^2*exp(x)')
g = f.integrate('x')
```

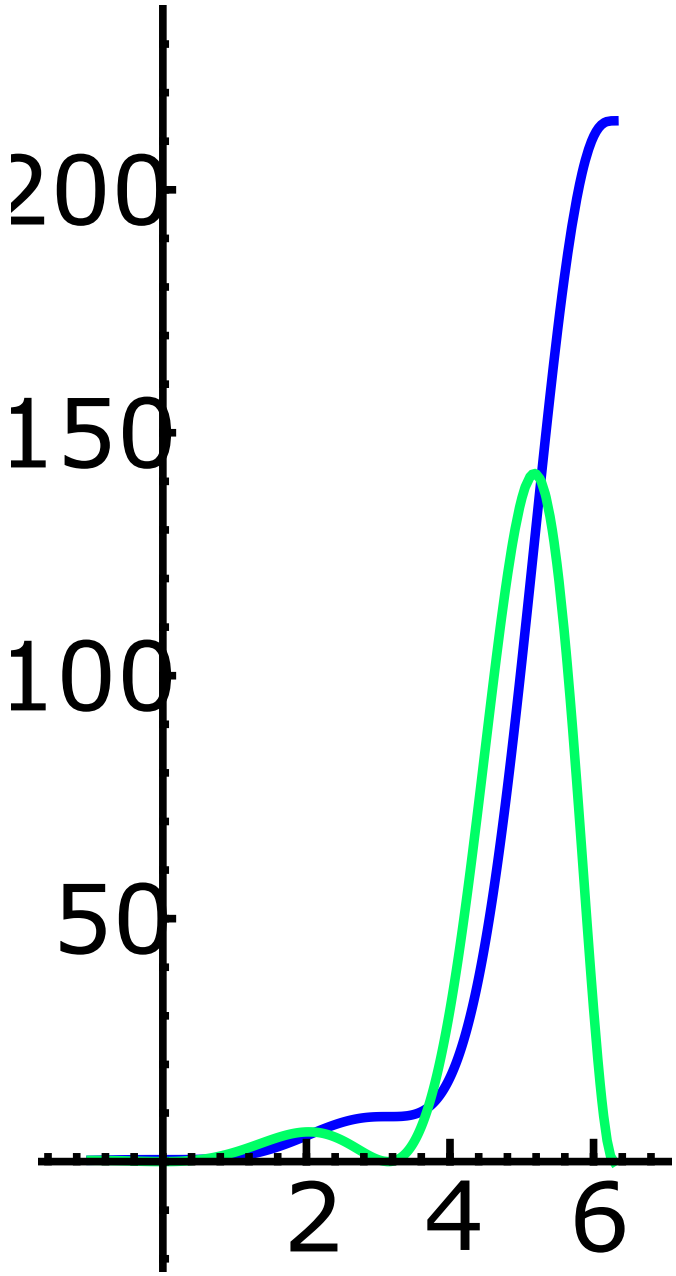
Plot $g(x)$, but don't typeset it.

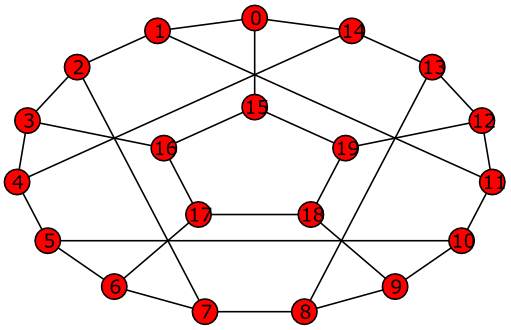
You can specify a file format and options for `includegraphics`. The default is for EPS and PDF files, which are the best choice in almost all situations. (Although see the section on 3D plotting.)



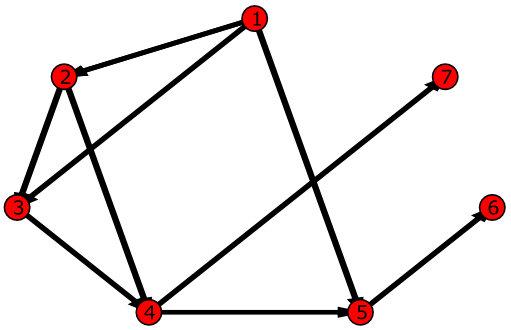
If you use regular `latex` to make a DVI file, you'll see a box, beause DVI files can't include PNG files. If you use `pdflatex` that will work. See the documentation for details.

When using `\sageplot`, you can pass in just about anything that Sage can call `.save()` on to produce a graphics file:


200
150
100
50
2
4
6




```
G4 = DiGraph({1:[2,2,3,5], 2:[3,4], 3:[4], 4:[5,7], 5:[6]},\
             multiedges=True)
G4plot = G4.plot(layout='circular')
```

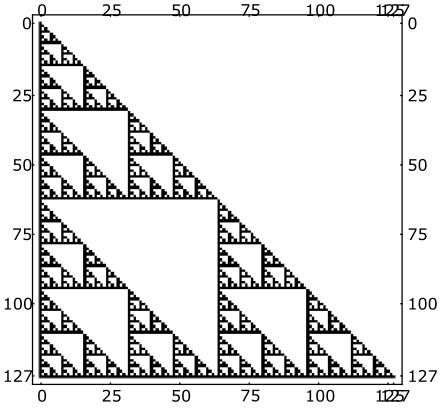


Indentation and so on works fine.

```
s     = 7
s2    = 2^s
P.<x> = GF(2)[]
M     = matrix(parent(x),s2)
for i in range(s2):
   p  = (1+x)^i
   pc = p.coeffs()
   a  = pc.count(1)
   for j in range(a):
       idx        = pc.index(1)
       M[i,idx+j] = pc.pop(idx)

matrixprogram = matrix_plot(M,cmap='Greys')
```

And here's the picture:



2.1 3D plotting

3D plotting right now is problematic because there's no convenient way to produce vector graphics. We can make PNGs, though, and since the `sageplot` command defaults to EPS and PDF, *you must specify a valid format for 3D plotting*. Sage right now (version 2.10.3) can't produce EPS or PDF files from plot3d objects, so if you don't specify a valid format, things will go badly. You can specify the "`imagemagick`" option, which will use the Imagemagick `convert` utility to make EPS files. See the documentation for details.

Here's the famous Sage cube graph:

```
G = graphs.CubeGraph(5)
```