2016-04-27

William A. Stein

4/27/2016

Contents

1 Math 480: Open Source Mathematical Software

1.0.1 2016-04-27

1.0.2 William Stein

1.1 Lectures 14: Symbolic Calculus (part 2/3)

1.2 Topics

(reminder: screencast) 1. finding roots: symbolic, numerical 1. numerical approximation of a symbolic expression 1. more about 2d plotting 1. more about 3d plots

1.3 Finding Roots: symbolic

You can use the solve command to solve for zeroes of a function.

```
x^2 + 3 == 5
```

```
x^2 + 3 == 5
```

```
eqn = x^2 + 3 == 5
show(eqn)
```

$$x^2 + 3 = 5$$

```
eqn.add_to_both_sides(-3)
```

```
x^2 == 2
```

```
pi == pi
```

```
pi == pi
```

```
bool(pi == pi)
```

```
True
```

```
solve(x^4 + 2*x + 3 == 0, x)
```

```
[x == -1/2*sqrt(((2*I*sqrt(15) + 2)^(2/3) + 4)/(2*I*sqrt(15) + 2)^(1/3)) -
1/2*sqrt(-(2*I*sqrt(15) + 2)^(1/3) - 4/(2*I*sqrt(15) + 2)^(1/3) + 4/sqrt(((2*I*sqrt(15) +
2)^(2/3) + 4)/(2*I*sqrt(15) + 2)^(1/3))), x == -1/2*sqrt(((2*I*sqrt(15) + 2)^(2/3) +
4)/(2*I*sqrt(15) + 2)^(1/3)) + 1/2*sqrt(-(2*I*sqrt(15) + 2)^(1/3) - 4/(2*I*sqrt(15) +
2)^(1/3) + 4/sqrt(((2*I*sqrt(15) + 2)^(2/3) + 4)/(2*I*sqrt(15) + 2)^(1/3))), x ==
1/2*sqrt(((2*I*sqrt(15) + 2)^(2/3) + 4)/(2*I*sqrt(15) + 2)^(1/3)) -
1/2*sqrt(-(2*I*sqrt(15) + 2)^(1/3) - 4/(2*I*sqrt(15) + 2)^(1/3) - 4/sqrt(((2*I*sqrt(15) +
2)^(2/3) + 4)/(2*I*sqrt(15) + 2)^(1/3))), x == 1/2*sqrt(((2*I*sqrt(15) + 2)^(2/3) +
4)/(2*I*sqrt(15) + 2)^(1/3)) + 1/2*sqrt(-(2*I*sqrt(15) + 2)^(1/3) - 4/(2*I*sqrt(15) +
2)^(1/3) - 4/sqrt(((2*I*sqrt(15) + 2)^(2/3) + 4)/(2*I*sqrt(15) + 2)^(1/3)))]
```

```
show(solve(x^4 + 2*x + 3 == 0, x)[0])
```

$$x = -\frac{1}{2}\sqrt{\frac{\left(2i\sqrt{15}+2\right)^{\frac{2}{3}}+4}{\left(2i\sqrt{15}+2\right)^{\frac{1}{3}}}} - \frac{1}{2}\sqrt{-\left(2i\sqrt{15}+2\right)^{\frac{1}{3}} - \frac{4}{\left(2i\sqrt{15}+2\right)^{\frac{1}{3}}} + \frac{4}{\sqrt{\frac{\left(2i\sqrt{15}+2\right)^{\frac{2}{3}}+4}{\left(2i\sqrt{15}+2\right)^{\frac{1}{3}}}}}}$$

```
solve(sqrt(x) == 2, x)
```

```
[x == 4]
```

```
solve(sin(x) == 0, x)
```

```
[x == 0]
```

```
solve(e^(3*x) == 5, x)
```

```
[x == log(1/2*I*5^(1/3)*sqrt(3) - 1/2*5^(1/3)), x == log(-1/2*I*5^(1/3)*sqrt(3) -
1/2*5^(1/3)), x == 1/3*log(5)]
```

```
v = solve(e^(3*x) == 5, x, solution_dict=True); v
```

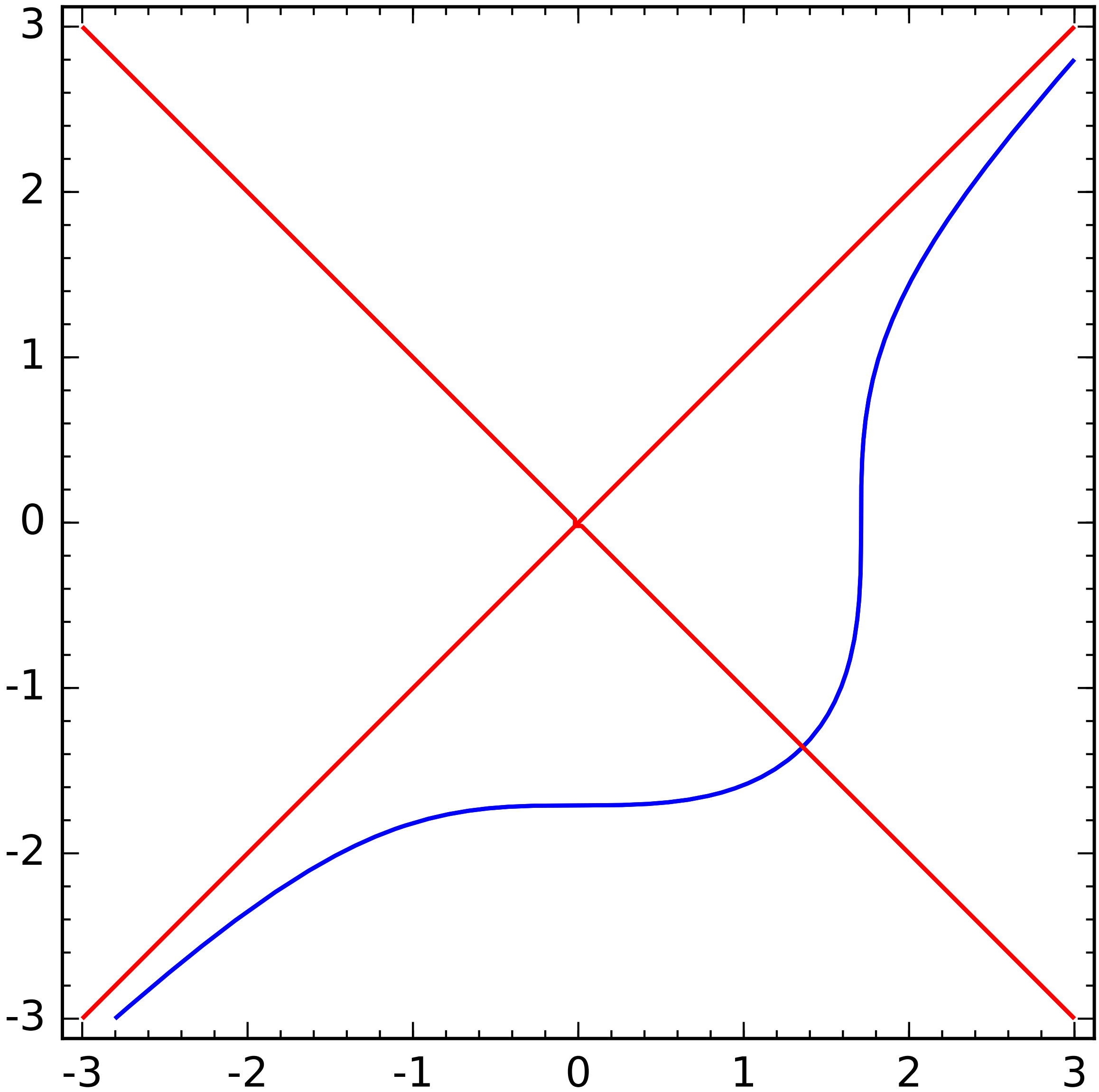
```
[{x: log(1/2*I*5^(1/3)*sqrt(3) - 1/2*5^(1/3))}, {x: log(-1/2*I*5^(1/3)*sqrt(3) -
1/2*5^(1/3))}, {x: 1/3*log(5)}]
```

```
%var x, y
solve([x^2 == y^2, x^3 == y^3 + 5], [x,y])
```

```
[[x == 1.35720887245841, y == -1.35720887245841], [x == (-0.6786044041487247 +
1.175377306225595*I), y == (0.6786044041487285 - 1.175377306225602*I)], [x ==
```

```
(-0.6786044041487247 - 1.175377306225595*I), y == (0.6786044041487285 +
1.175377306225602*I)]]
```

```
g = implicit_plot(x^3 == y^3 + 5, (x, -3, 3), (y,-3,3))
g += implicit_plot(x^2 == y^2, (x, -3, 3), (y,-3,3),color='red')
g
```




```
show(v[0][x])
```

$$\log\left(\frac{1}{2} i \cdot 5^{\frac{1}{3}} \sqrt{3} - \frac{1}{2} \cdot 5^{\frac{1}{3}}\right)$$

```
(e^(v[0][x]*3)).simplify_full()
5
```

```
# or use roots:
v = (e^(3*x) - 5).roots()
v
```

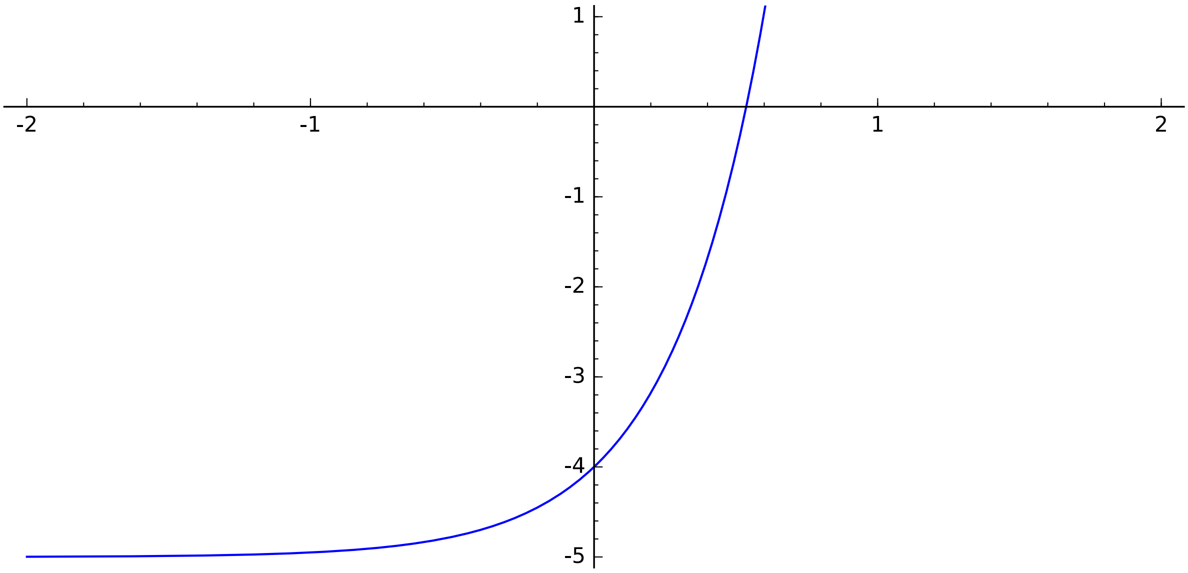
```
[(log(1/2*I*5^(1/3)*sqrt(3) - 1/2*5^(1/3)), 1), (log(-1/2*I*5^(1/3)*sqrt(3) -
1/2*5^(1/3)), 1), (1/3*log(5), 1)]
```

```
show(v[0][0])
```

$$\log\left(\frac{1}{2}i \cdot 5^{\frac{1}{3}}\sqrt{3} - \frac{1}{2} \cdot 5^{\frac{1}{3}}\right)$$

1.4 Finding A SINGLE Root in an interval: numerical

```
f(x) = e^(3*x) - 5
plot(f, -2, 2, ymax=1)
```




```
f.find_root(1, 2)
```

```
Error in lines 1-1
Traceback (most recent call last):
  File ‘‘/projects/sage/sage-6.10/local/lib/python2.7/site-
packages/smc_sagews/sage_server.py’’, line 905, in execute
    exec compile(block+'\n', '', 'single') in namespace, locals
  File ‘‘’’, line 1, in <module>
  File ‘‘sage/symbolic/expression.pyx’’, line 10840, in
sage.symbolic.expression.Expression.find_root
(/projects/sage/sage-6.10/src/build/cythonized/sage/symbolic/expression.cpp:57550)
    return find_root(f, a=a, b=b, xtol=xtol,
  File ‘‘/projects/sage/sage-6.10/local/lib/python2.7/site-
```

```
packages/sage/numerical/optimize.py'', line 94, in find_root
    raise RuntimeError(``f appears to have no zero on the interval'')
RuntimeError: f appears to have no zero on the interval
```

1.5 Numerical approximation of a symbolic expression

```
alpha = log(1/2*I*5^(1/3)*sqrt(3) - 1/2*5^(1/3))
show(alpha)
```

$$\log\left(\frac{1}{2}i \cdot 5^{\frac{1}{3}}\sqrt{3} - \frac{1}{2} \cdot 5^{\frac{1}{3}}\right)$$

```
f = pi/10^30 + pi - e/10^30 - acos(0)
f
```

```
500000000000000000000000000001/1000000000000000000000000000000*pi -
1/1000000000000000000000000000000*e
```

```
numerical_approx(f)
```

```
1.57079632679490
```

```
numerical_approx(alpha)
```

```
0.536479304144700 + 2.09439510239320*I
```

```
numerical_approx(alpha, prec=200)
```

```
0.53647930414470012486691977742062546508533784756172573970 88 +
2.0943951023931954923084289221863352561314462662500705473166*I
```

```
numerical_approx(alpha, digits=20)
```

```
0.53647930414470012487 + 2.0943951023931954923*I
```

```
# This is the number of digits used in computing the result, NOT the\
    number of correct digits in output!
numerical_approx(alpha, digits=3)
```

```
0.536 + 2.09*I
```

```
# Interval arithmetic --
# Every displayed digit except last in the output is definitely \
   right:
ComplexIntervalField(20)(alpha)
```

```
0.53648? + 2.09440?*I
```

```
N is numerical_approx
```

```
True
```

```
alpha.N()
```

```
0.536479304144700 + 2.09439510239320*I
```

```
alpha.n()
```

```
0.536479304144700 + 2.09439510239320*I
```

```
alpha.numerical_approx()
```

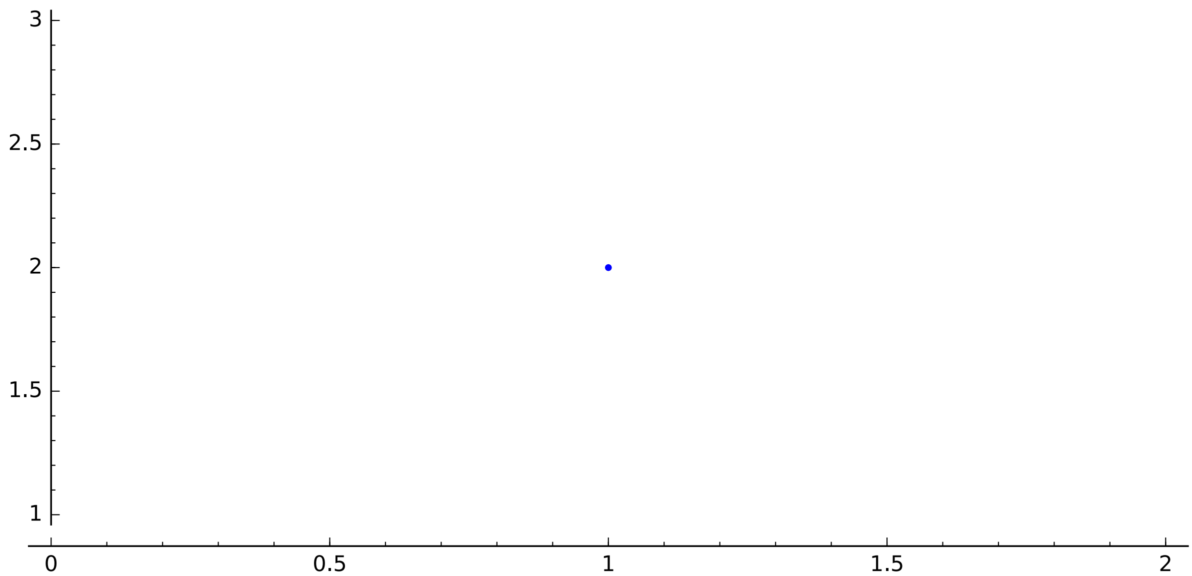
```
0.536479304144700 + 2.09439510239320*I
```

1.6 More about 2d plots

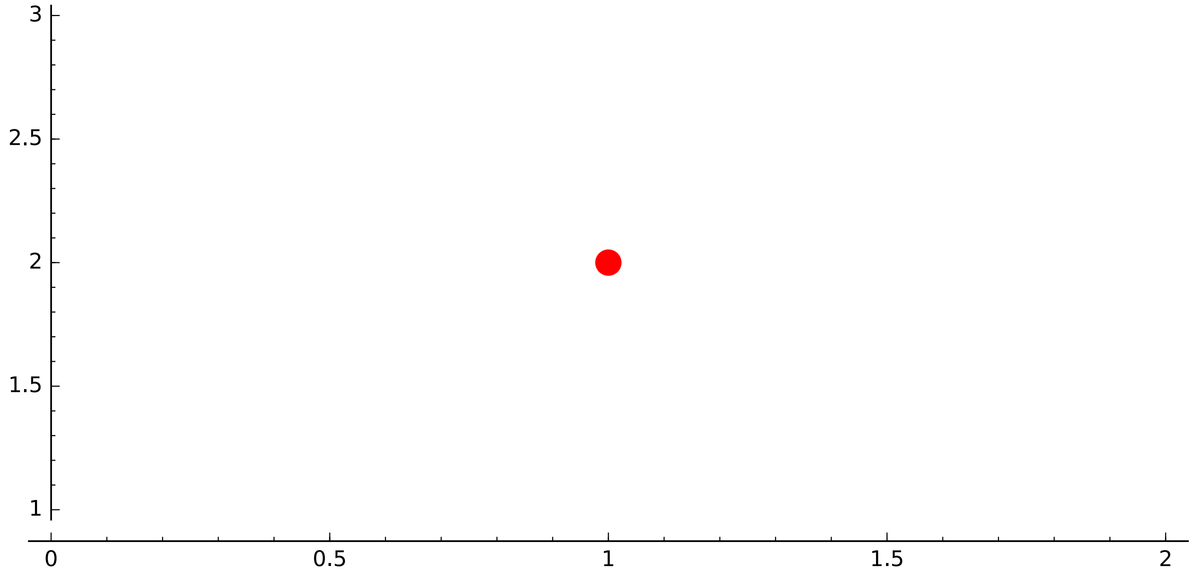
You can do much more than just plot(function...). E.g.,

- a point
- a bunch of points
- text
- "line" through a bunch of points
- polygon
- ellipse
- implicit plot
- contour plot
- vector field

```
point((1,2))
```




```
point2d((1,2), pointsize=150, color='red')
```




```
point([(random(), random()) for i in range(100)])
```

tmp_Q7VW0n.pdf

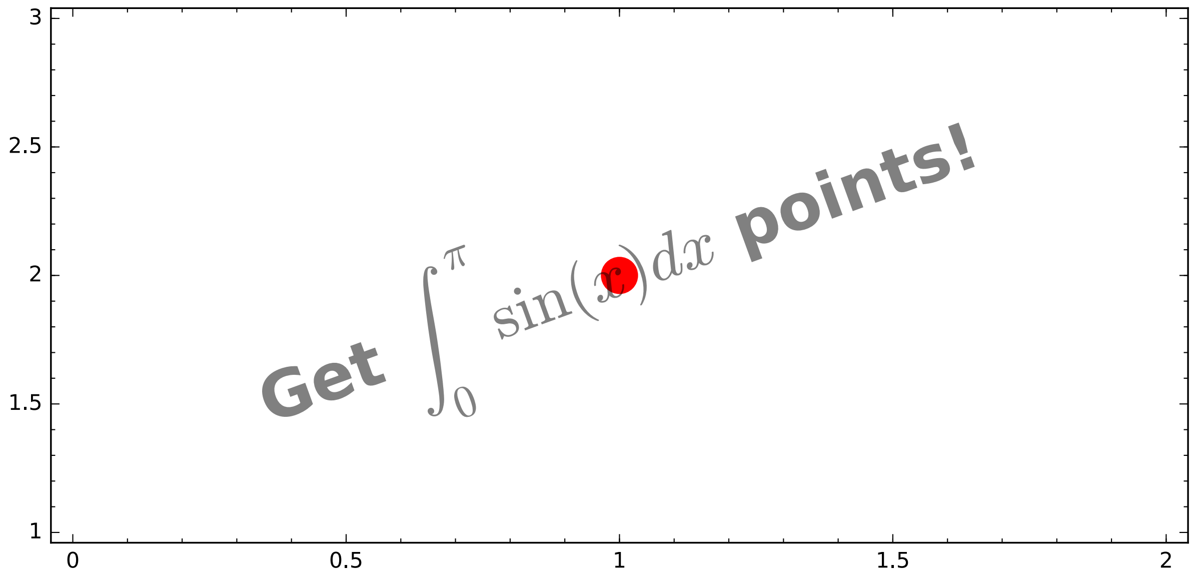
```
print r"adlkjf\nlksjdfljs"
```

adlkjf\nlksjdfljs

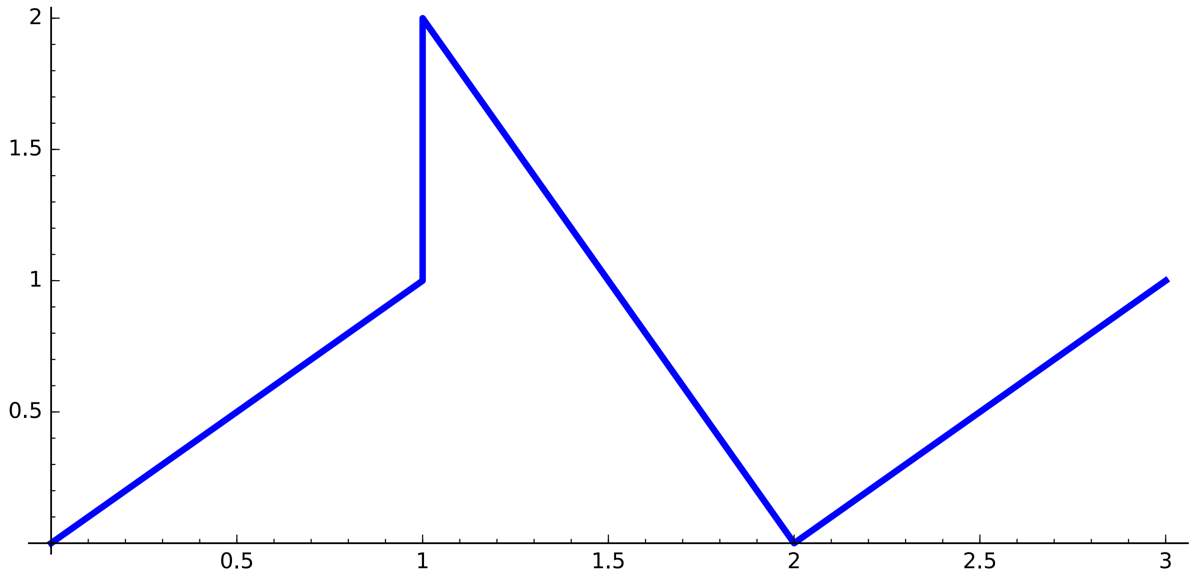
```
print "adlkjf\\lksjdfljs"
```

adlkjf\lksjdfljs

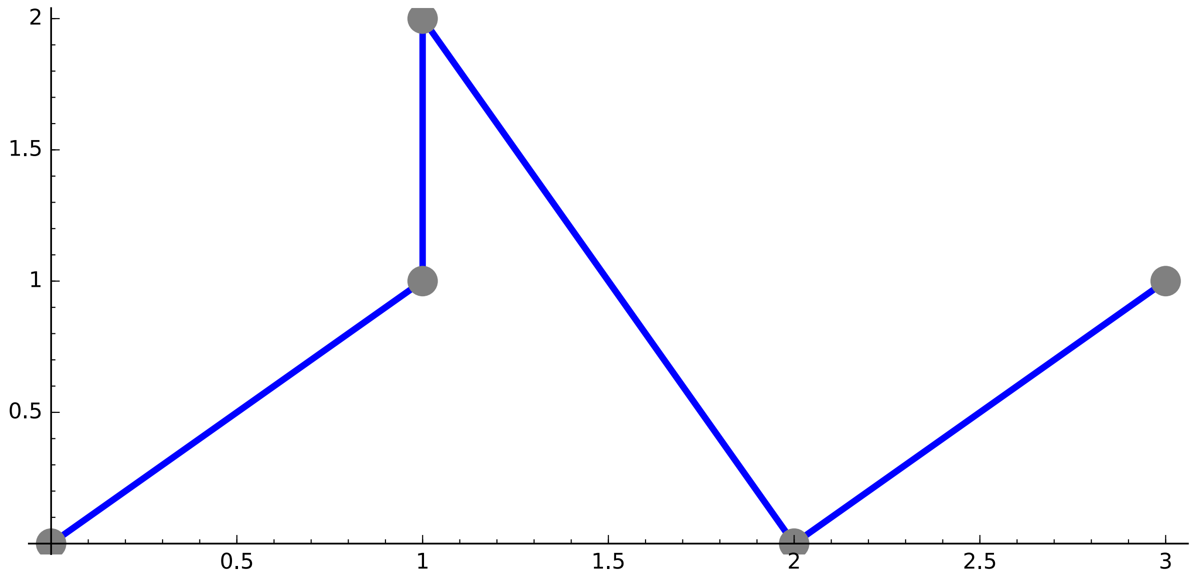
```
g =  point((1,2), pointsize=300, color='red')
g += text(r"Get $\int_0^{\pi} \sin(x) dx$ points!", (1,2), alpha\
   =0.5, fontsize=30, fontweight='bold', color='black', rotation=20,\
    zorder=1)
g.show(frame=True, axes=False)
```



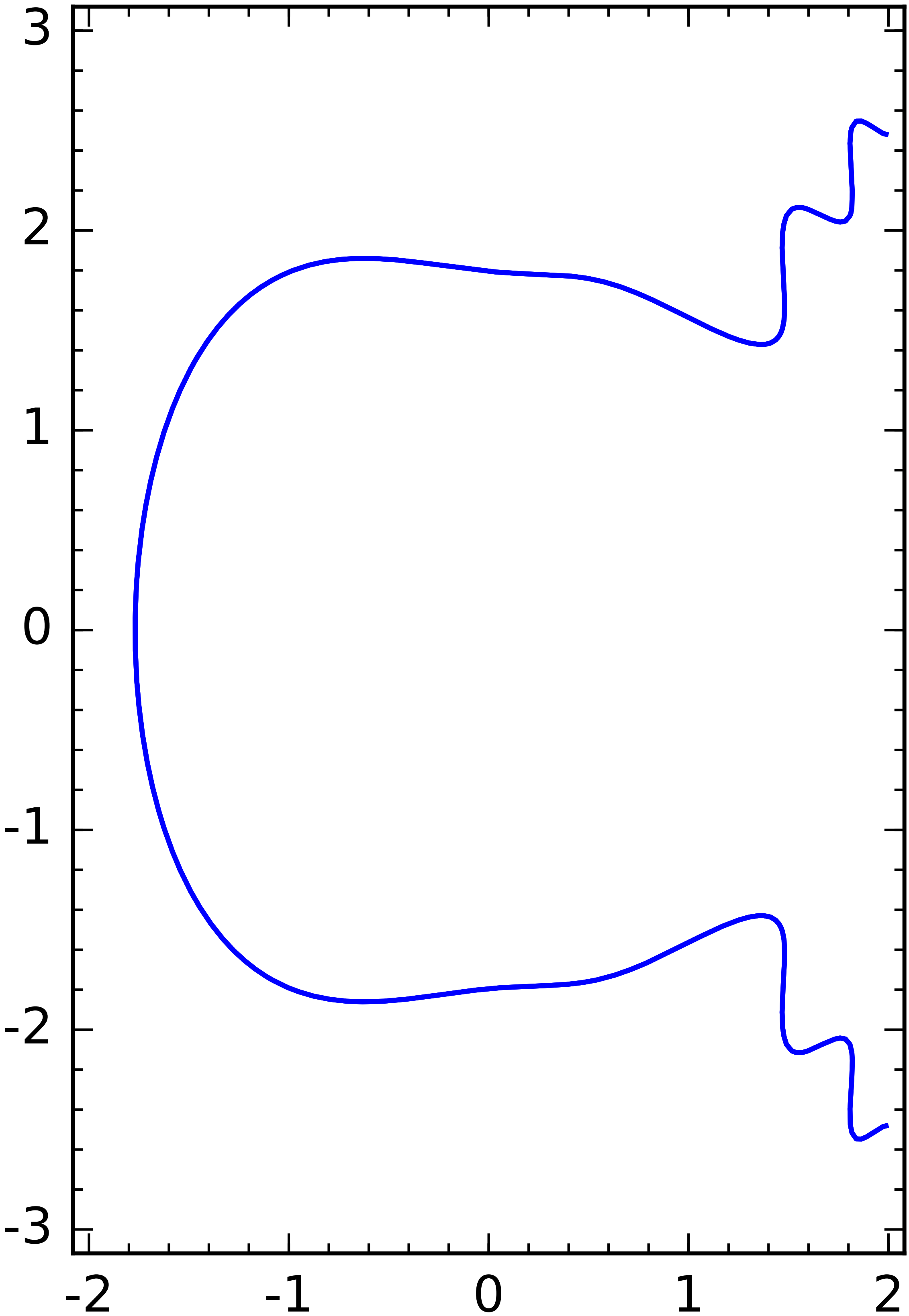

```
line([(0,0), (1,1), (1,2), (2,0), (3,1)], thickness=3, color='blue')
```




```
v = [(0,0), (1,1), (1,2), (2,0), (3,1)]
line(v, thickness=3, color='blue', zorder=-1) + points(v, pointsize\
   =200, color='grey', zorder=1)
```




```
%var x, y
implicit_plot(y^2 + cos(y*e^x) == x^3 - 2*x + 3, (x,-2,2), (y,-3,3))
```


3
2
1
0
-1
-2
-3
-2
-1
0
1
2

```
oo
```

```
+Infinity
```

```
-oo
```

```
-Infinity
```

```
show(plot(x^2, -100, 100), ymax=3, ymin=2)
```