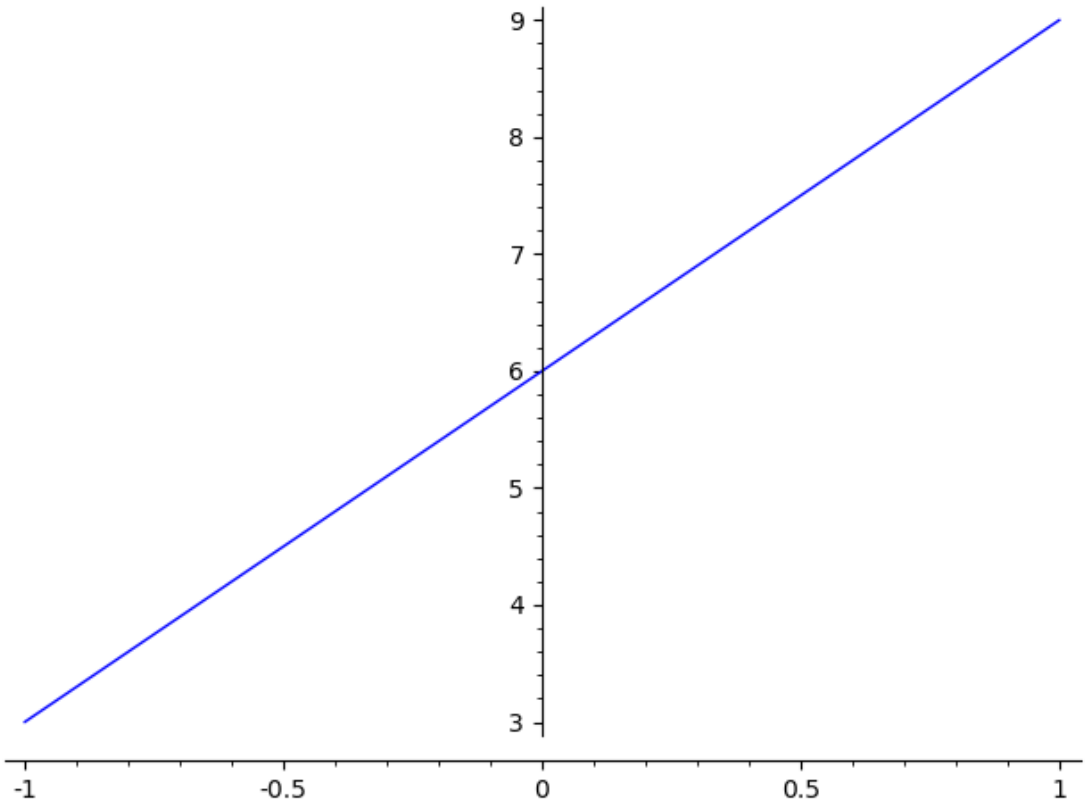
MATH1110H-B-lab-2023-09-19-F02

September 26, 2023

[ ]:
```
# MATH 1110H-B F02 Lab 2023-09-19
#
# Wherein we ring the changes on the several plotting commands in
# SageMath, without getting into the fancy stuff like labelling
# points, curves, or axes.
```

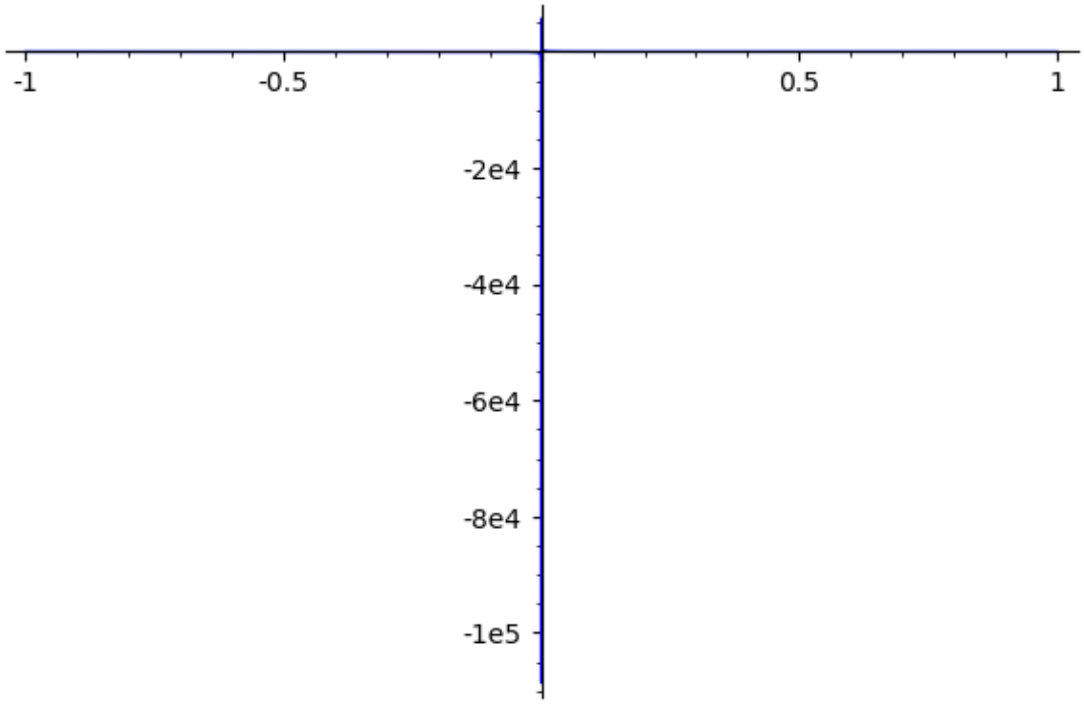
[1]:
```
plot(3*x+6)  # The basic plot command, which plots the function
             # for x between -1 and 1 by default. Note that the
             # scale on the vertical and horizontal axes is not
             # the same.
```

[1]:



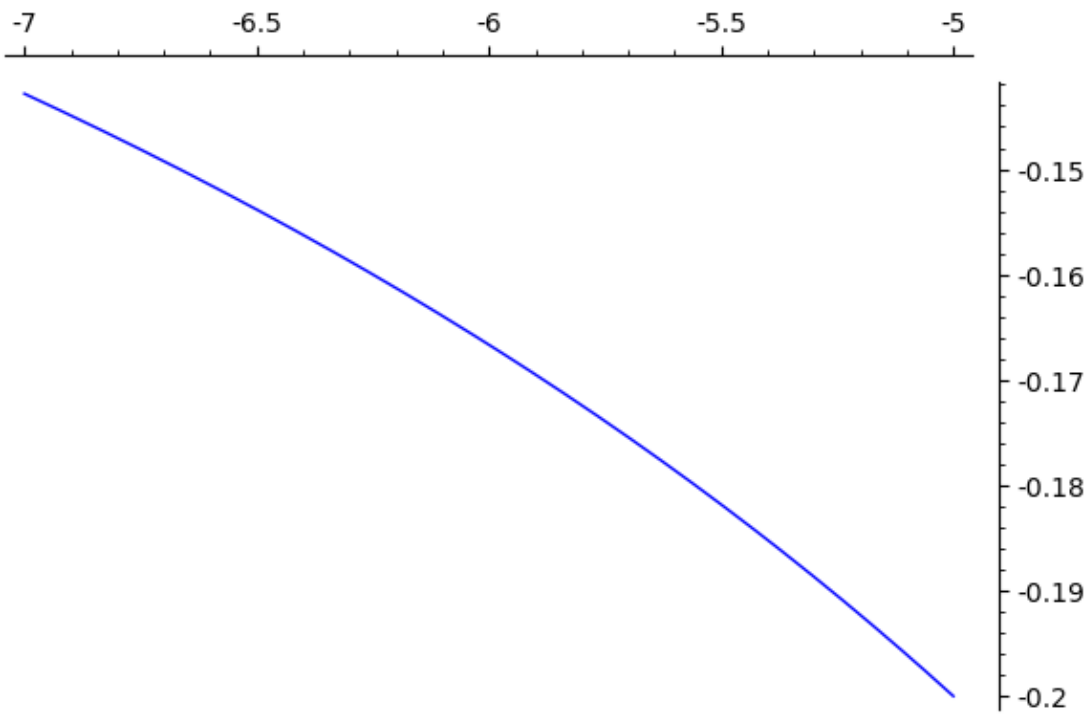
[2]:
```
plot(1/x)  # The scale problem gets worse when a function has an
           # asymptote.
```

[2]:



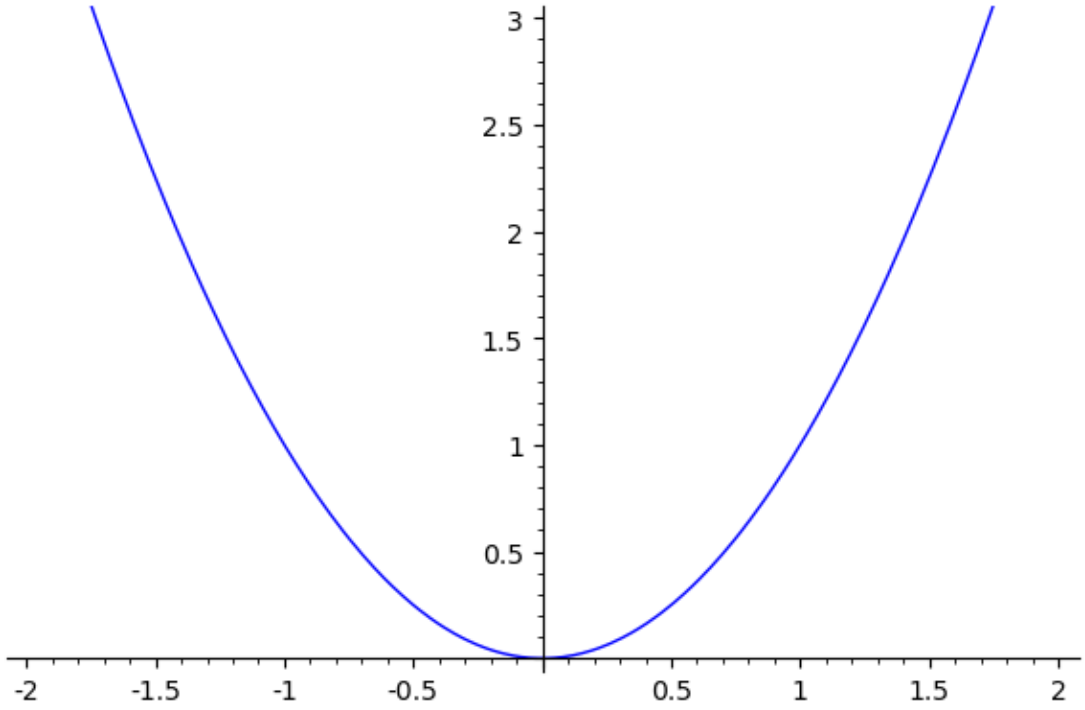
[3]:
```
plot(1/x,-7,-5)  # One can change the x-values used in the plot.
```

[3]:



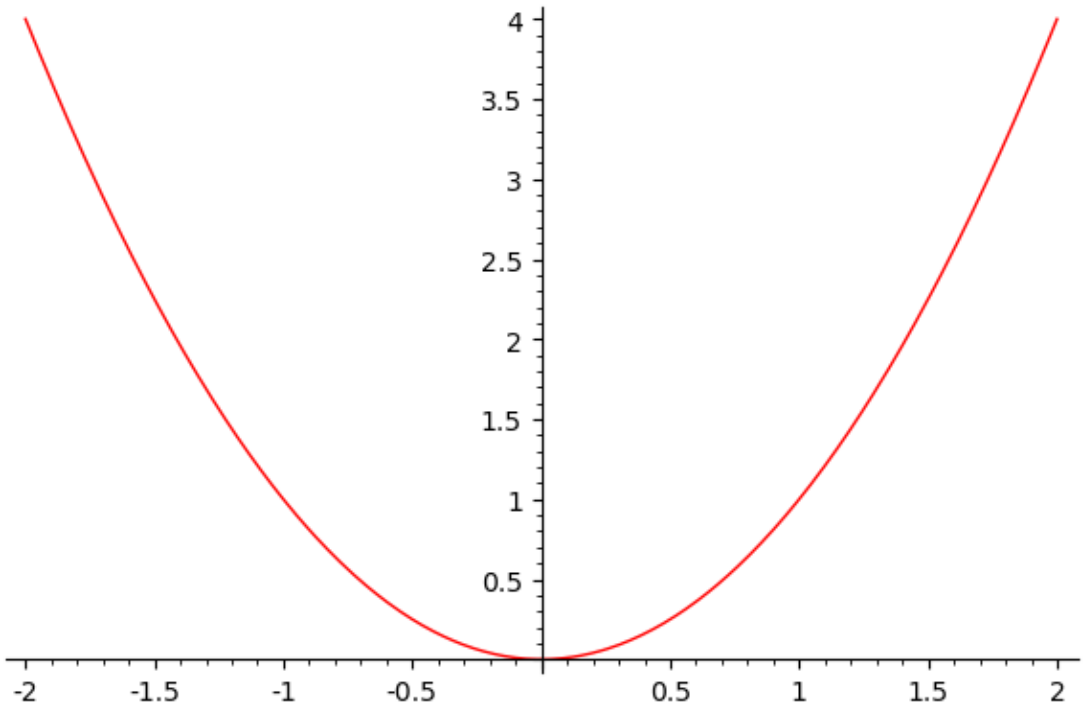
[4]:
```
plot(x^2,-2,2,ymin=-0,ymax=3)  # One can also limit the y-values.
```

[4]:



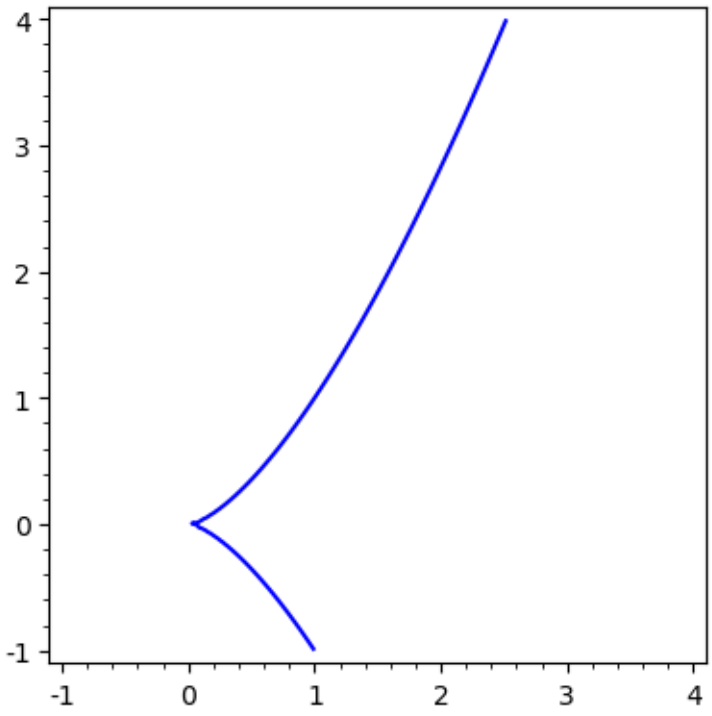
[5]:
```
# One can also change the color of the plotted curve.
plot(x^2,-2,2,color='red')
```

[5]:



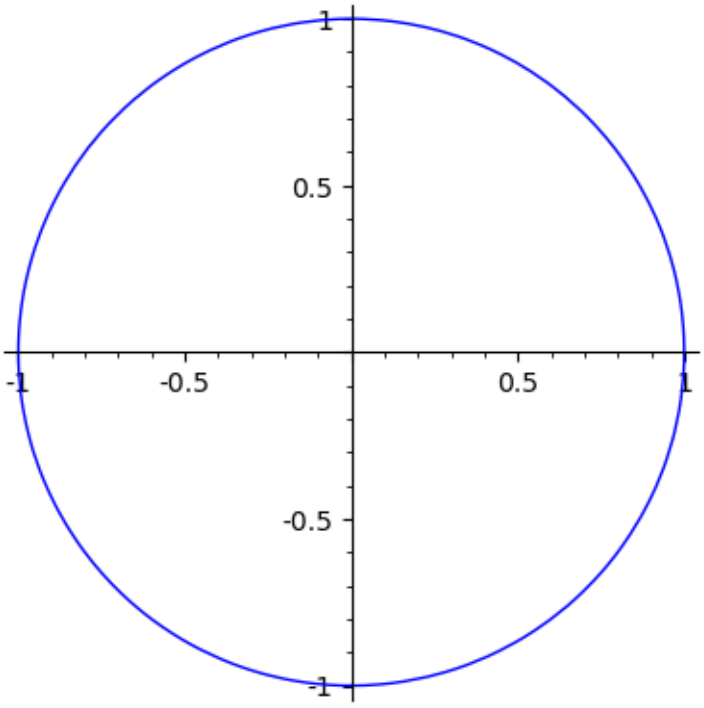
[8]:
```
var('y')  # If you want anything other than x to be considered as a
          # variable, you need to specify as such.
implicit_plot(y^2==x^3,(x,-1,4),(y,-1,4))  # Graphing curves
          # defined implicitly by an equation has its own command.
          # Note the use of == for equality in the given equation
          # and the need to specify the range for each variable
          # separately. (With a format a rather different from how
          # one does it in the basic plot command.)
```

[8]:



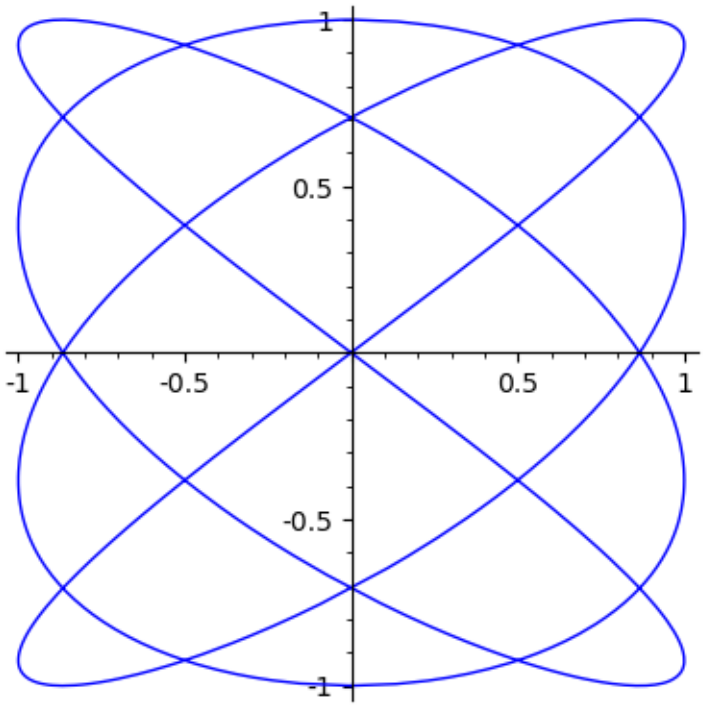
[9]:
```
var('w')  # Once again,  to use t as a variable,  we need to tell
          # SageMath this before actually using it.
parametric_plot((sin(w),cos(w)),(w,-pi,pi))  # This is the
          # specialized command for plotting parametric curves, in
          # which the x and y coordinates are controlled by a third
          # variable (the parameter), i.e. x = f(t) and y = g(t)
          # for some functions f(t) and g(t). (See Section 10.4 of
          # the textbook.) Note that the x and y coordinates are
          # specified in an ordered pair and that the range of t to
          # be used is given in the same format as ranges in the
          # implicit_plot command are.
```

[9]:



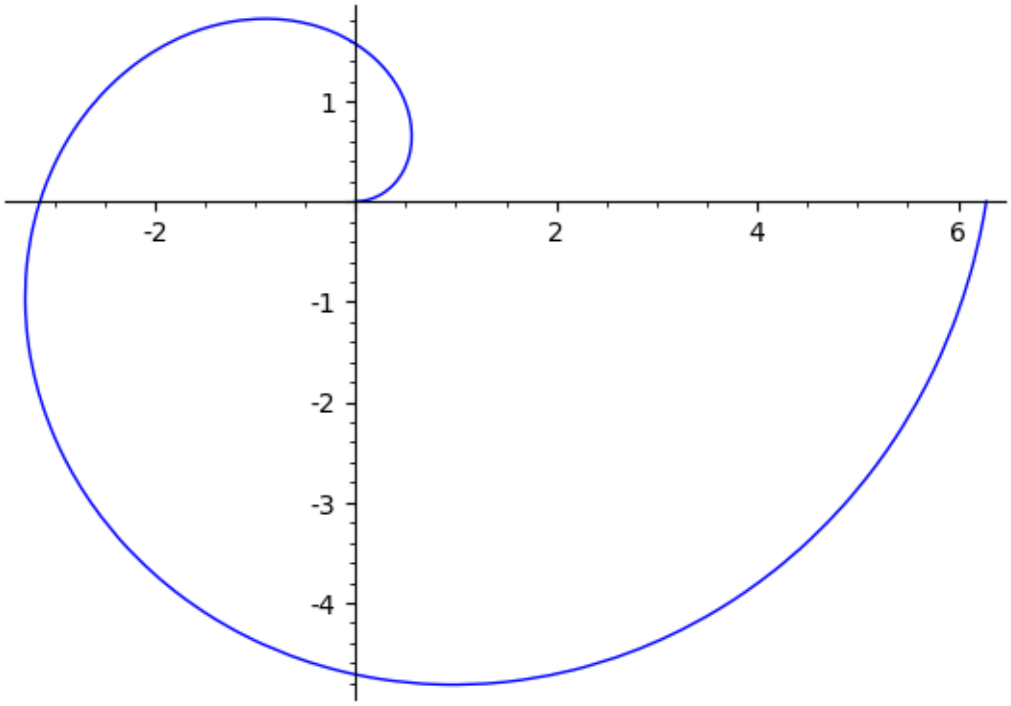
[10]:
```
parametric_plot((sin(4*w),cos(3*w)),(w,-pi,pi))
# Another example of a parametric plot;  this is a Lissajous curve.
```

[10]:



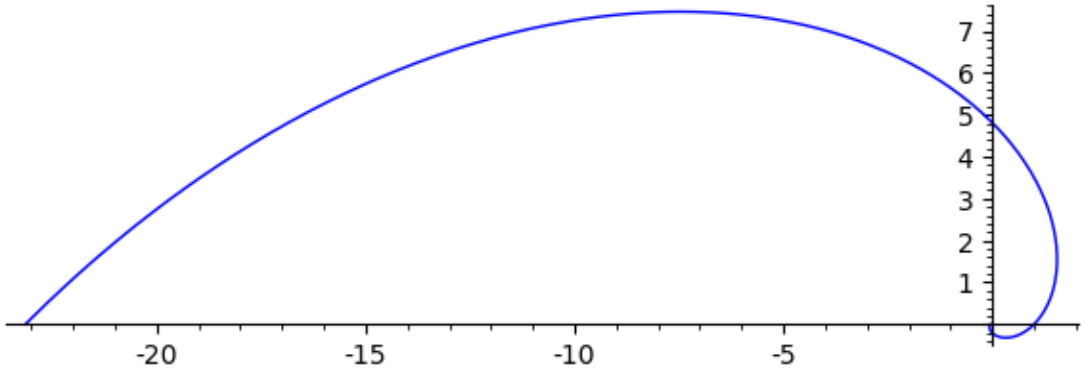
[11]:
```
var('theta')  # Again,  to use theta as a variable,  we need to tell
              # SageMath this before actually using it.
polar_plot(theta,(theta,0,2*pi))  # This is the specialized
              # command for plotting in polar coordinates. Theta
              # gives the direction of a point, i.e. the angle that
              # the line joining the origin to the point makes with
              # the positive x-axis, measured counterclockwise,
              # and r = f(theta) gives the distance the point is
              # from the origin. (See Section 10.1 of the textbook.)
              # Note that the range of theta to be used is given in
              # the same format as ranges in the implicit_plot and
              # parametric_plot commands are.
```

[11]:



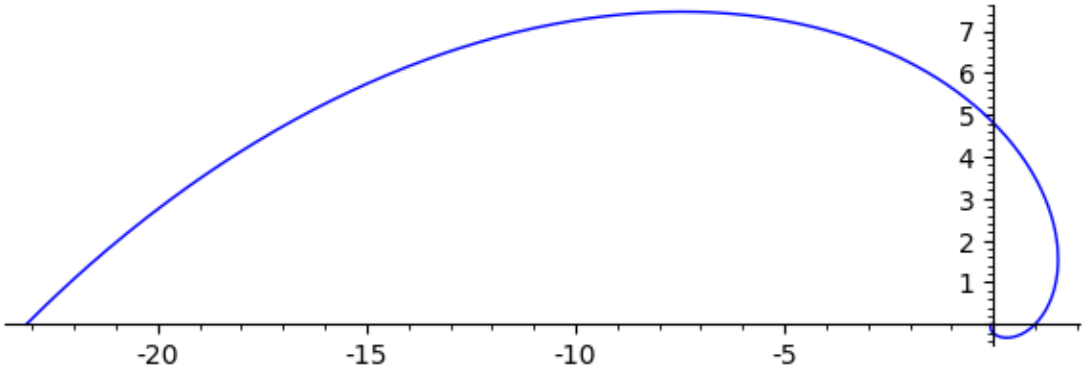
[13]:
```
polar_plot(e^theta,(theta,-pi,pi))
# An exponential spiral in polar coordinates.
```

[13]:



[14]:
```
f = e^theta  # The same spiral,  but with the function defined and
             # named outside the polar_plot command.  This is handy
             # if you want to use a function repeatedly or if you
             # want to avoid overly long commands.
polar_plot(f,(theta,-pi,pi))
```

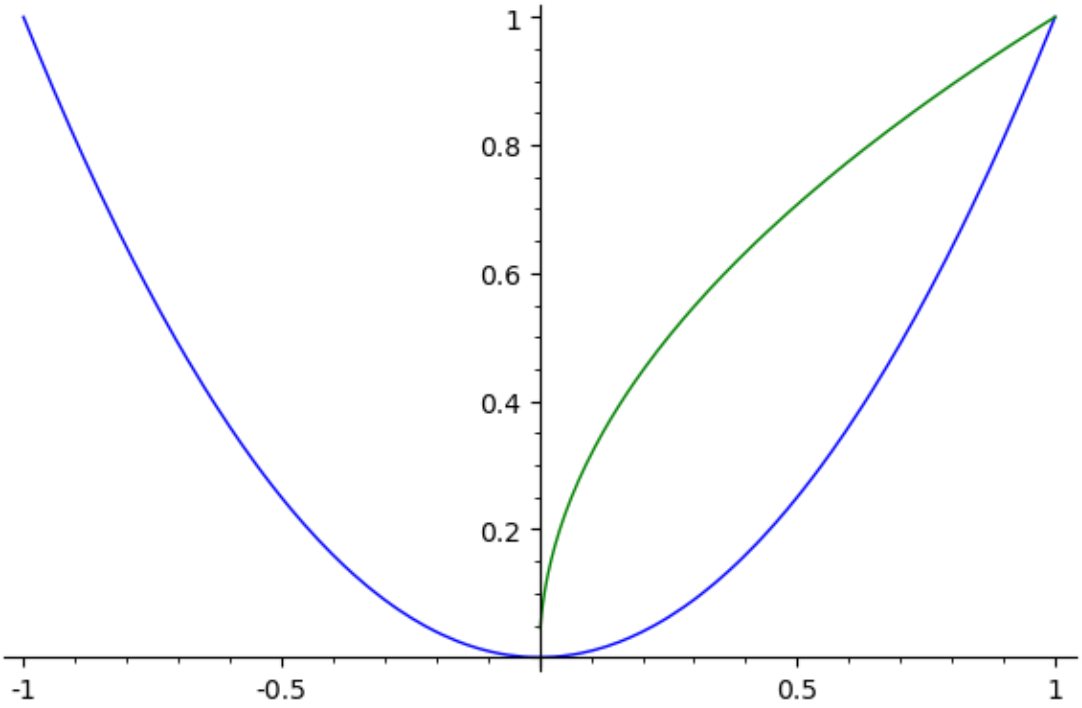
[14]:



[15]:
```
p1 = plot(x^2)  # You can also give plots names and refer to them
                # by name later...
p2 = plot(sqrt(x),color='green')
p1+p2           # ... and superimpose plots by adding them!
```

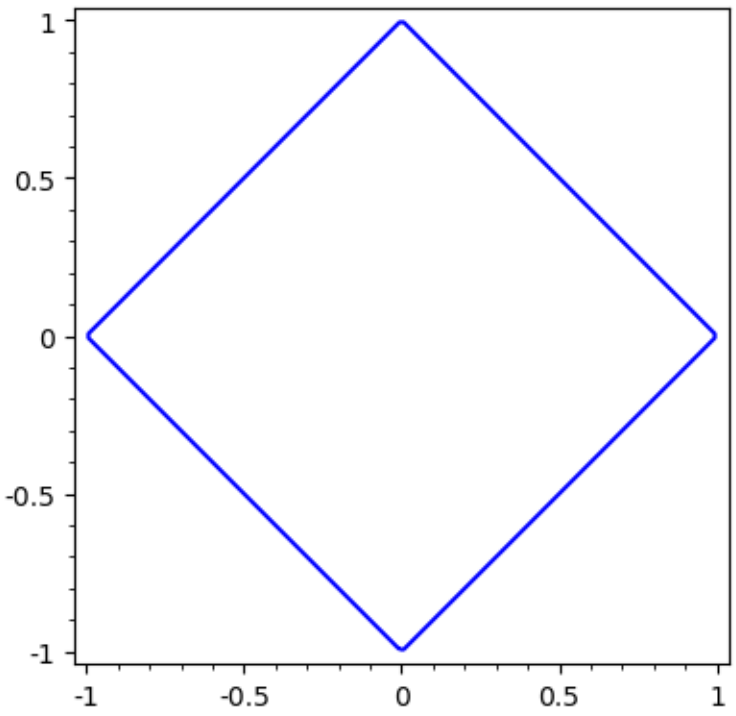
```
verbose 0 (3791: plot.py, generate_plot_points) WARNING: When plotting, failed
to evaluate function at 100 points.
verbose 0 (3791: plot.py, generate_plot_points) Last error message: 'math domain
error'
```

[15]:



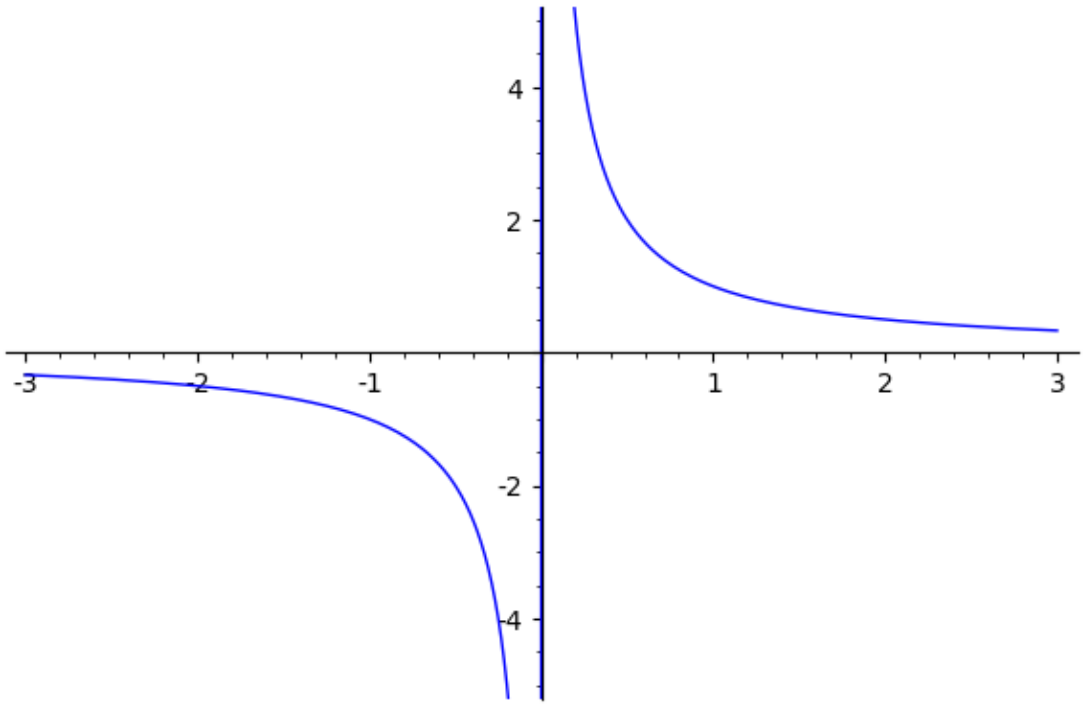
[16]:
```
implicit_plot(abs(x) + abs(y) == 1, (x,-1,1), (y,-1,1))
# SageMath use abs for the absolute value function.
```

[16]:



[19]:
```
plot(1/x,-3,3,ymin=-5,ymax=5)  # Here we revisit out very first
                               # plot with limits on x and y that
                               # give us a plot that gives a good
                               # idea of how the function behaves.
```

[19]:



[ ]:
```
# That's all for this time! :-)
```