5

July 23, 2021

1 Criando Gráficos

O uso de gráficos na matemática, principalmente no Cálculo e na Geometria Analítica, é de extrema importância. Embora esse não seja o foco do `SymPy`, é possível criar os chamados *plots* com certa facilidade. Inclusive, como é utilizado o `matplotlib` para criar os gráficos, há uma grande margem para alterações. Isso permite gráficos quase totalmente customizáveis.

Os tipos de plots abordados nesse capítulos serão os criados pelas funções:

- plot()
- plot_implicit()
- plot3d()

O seu uso é facilitado mais ainda pelo ambiente Jupyter, e podemos trabalhar com eles de forma semelhante ao que fora abordado no último capítulo. Ou seja, podemos criar um objeto para trabalharmos com ele, e depois ver o resultado. Ou, podemos simplesmente criar um plot.

Antes de começarmos, vamos a importação e as definições essenciais:

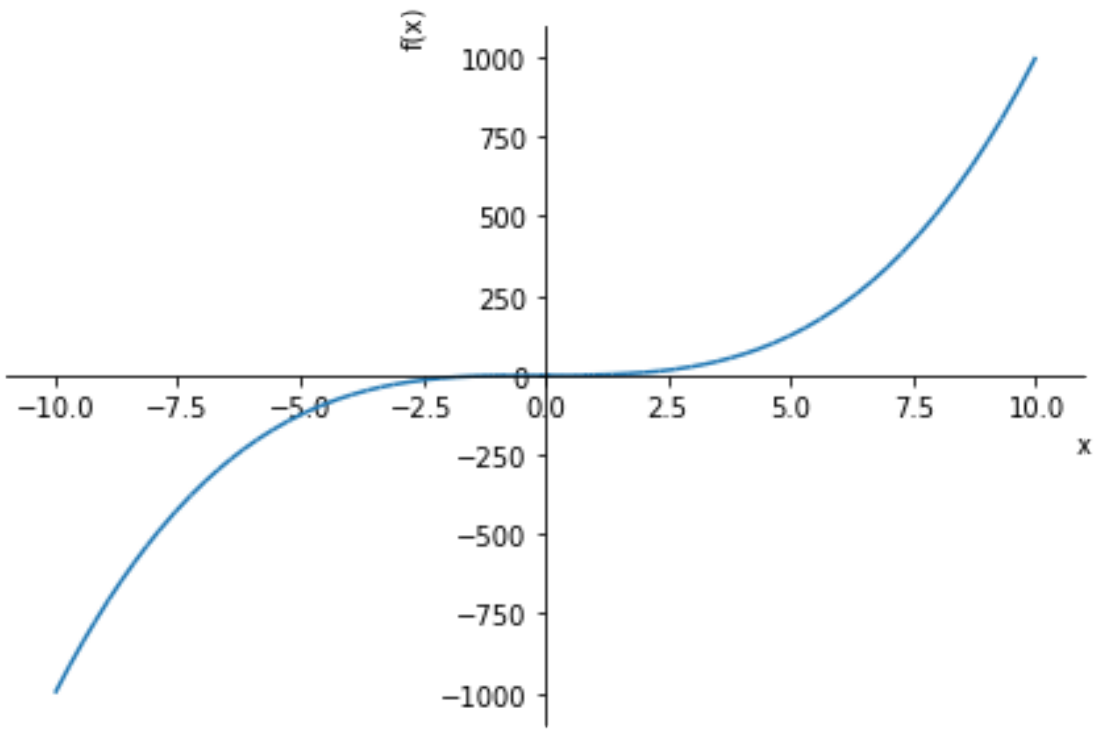
[1]:
```
from sympy import *
x, y, z = symbols('x y z')
init_printing(use_unicode=True, use_latex='mathjax')
```

1.1 Plot 2D

As duas primeiras funções que trabalharemos criam plots em 2D. Para fazer um plot de uma função, basta utilizar a função `plot()`. Confira abaixo:

1.1.1 Funções

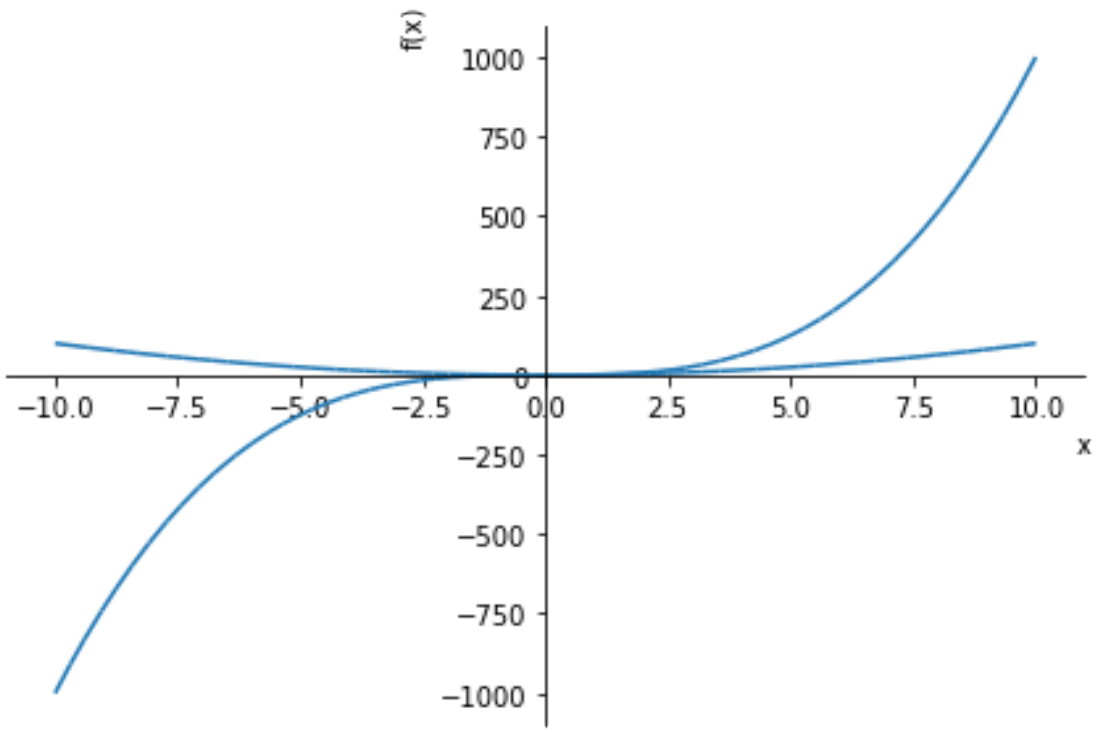
[2]:
```
plot(x**3)
```



[2]: `<sympy.plotting.plot.Plot at 0x7f2220cd9710>`

Podemos, inclusive, fazer vários plots unidos.

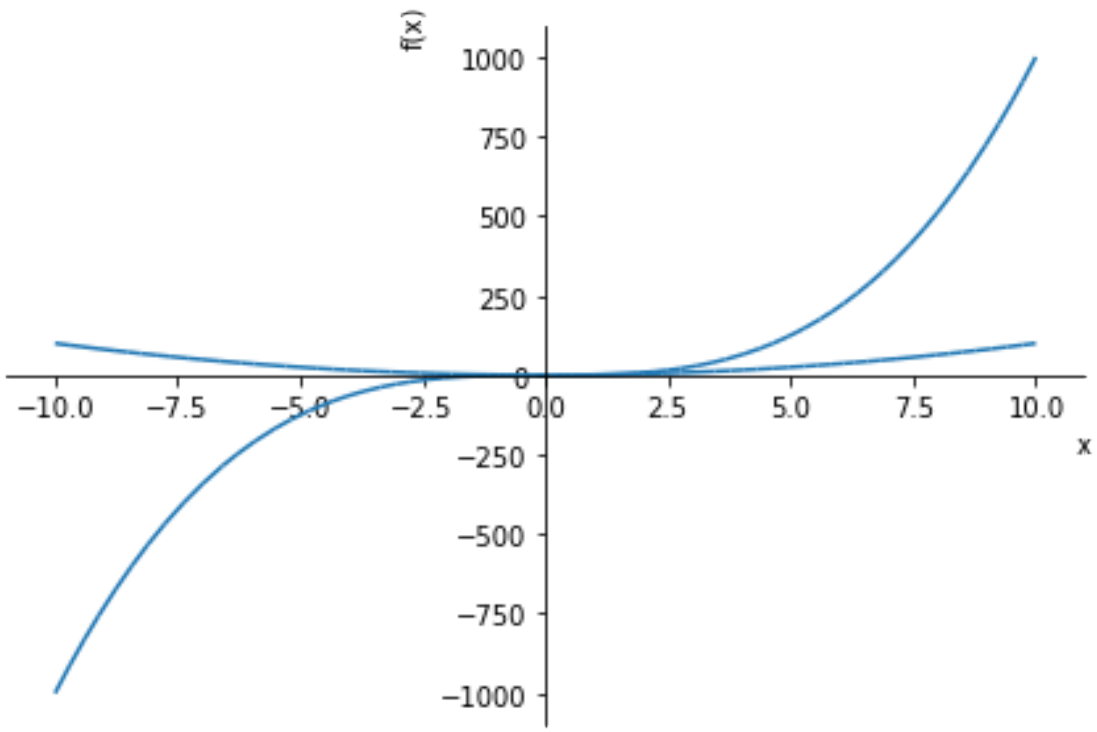
[3]:
```
plot(x**3, x**2)
```



[3]: <sympy.plotting.plot.Plot at 0x7f21f6b30ad0>

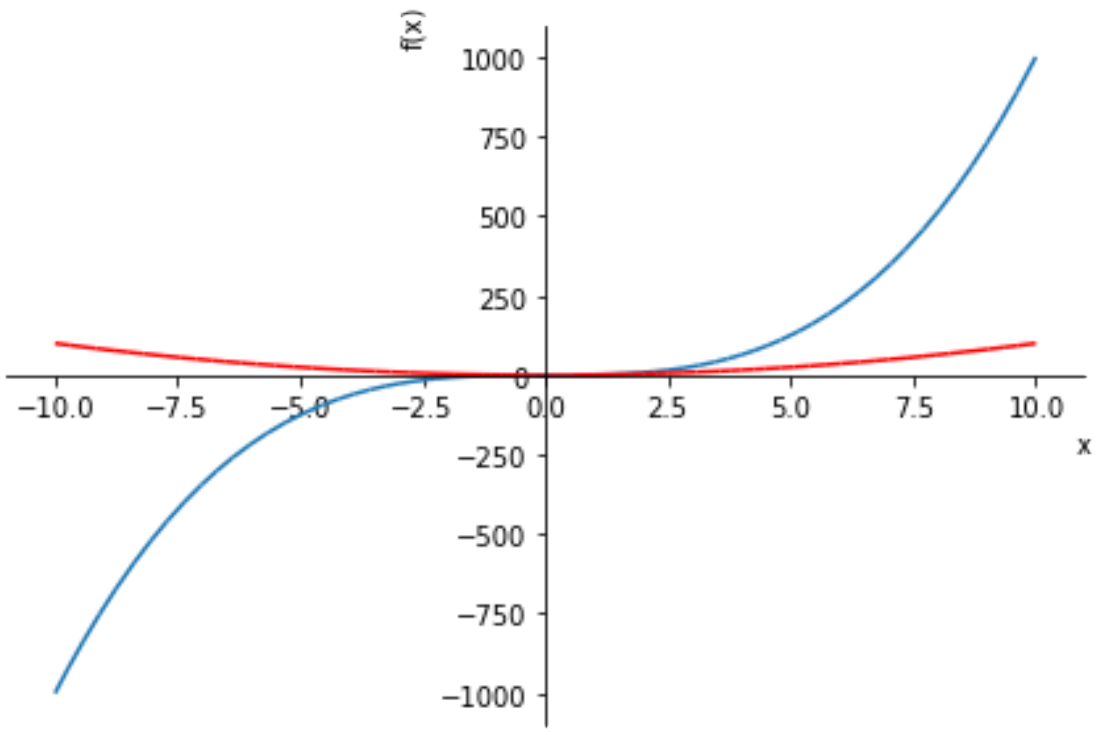
Obviamente, o gráfico acima está longe de ser o melhor possível. Mas com apenas uma função fizemos algo relevante. Mas, se armazenarmos, note o que podemos fazer.

[4]:
```
my_plot = plot(x**3, x**2, show=False) # show=False para não plotar.
my_plot.show() # .show() para exibir
```



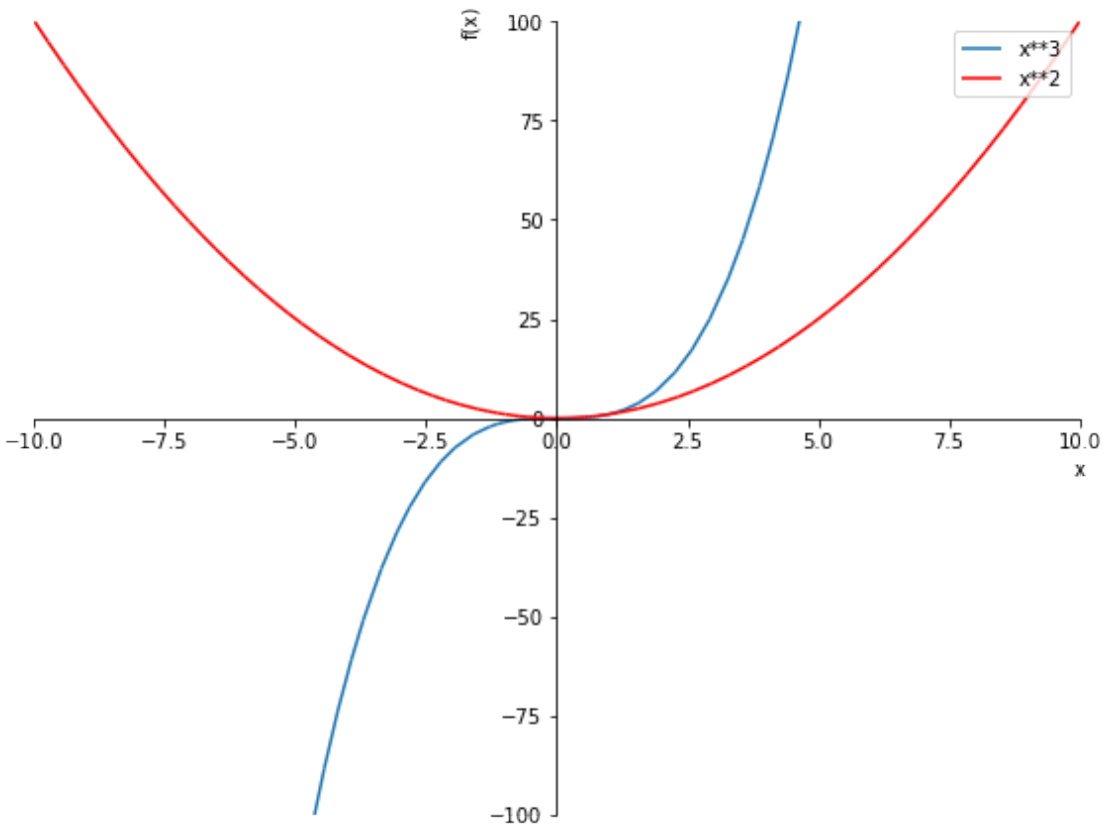
Por enquanto, nada mudou. Mas podemos ir trabalhando em seus atributos assim.

[5]:
```
my_plot[1].line_color = 'red' # x²
my_plot.show()
```



Estamos melhorando. Veja que `my_plot` armazena as funções em sequência. Trabalhamos em x^2 individualmente quando usamos `my_plot[1]`.

[6]:
```
my_plot.legend = True # Legenda
my_plot.xlim = (-10,10) # Esse é o máximo por padrão, veremos como alterar isso␣
↪mais tarde
my_plot.ylim = (-100,100)
my_plot.size = (8,6) # Tamanho da figura
my_plot.show()
```



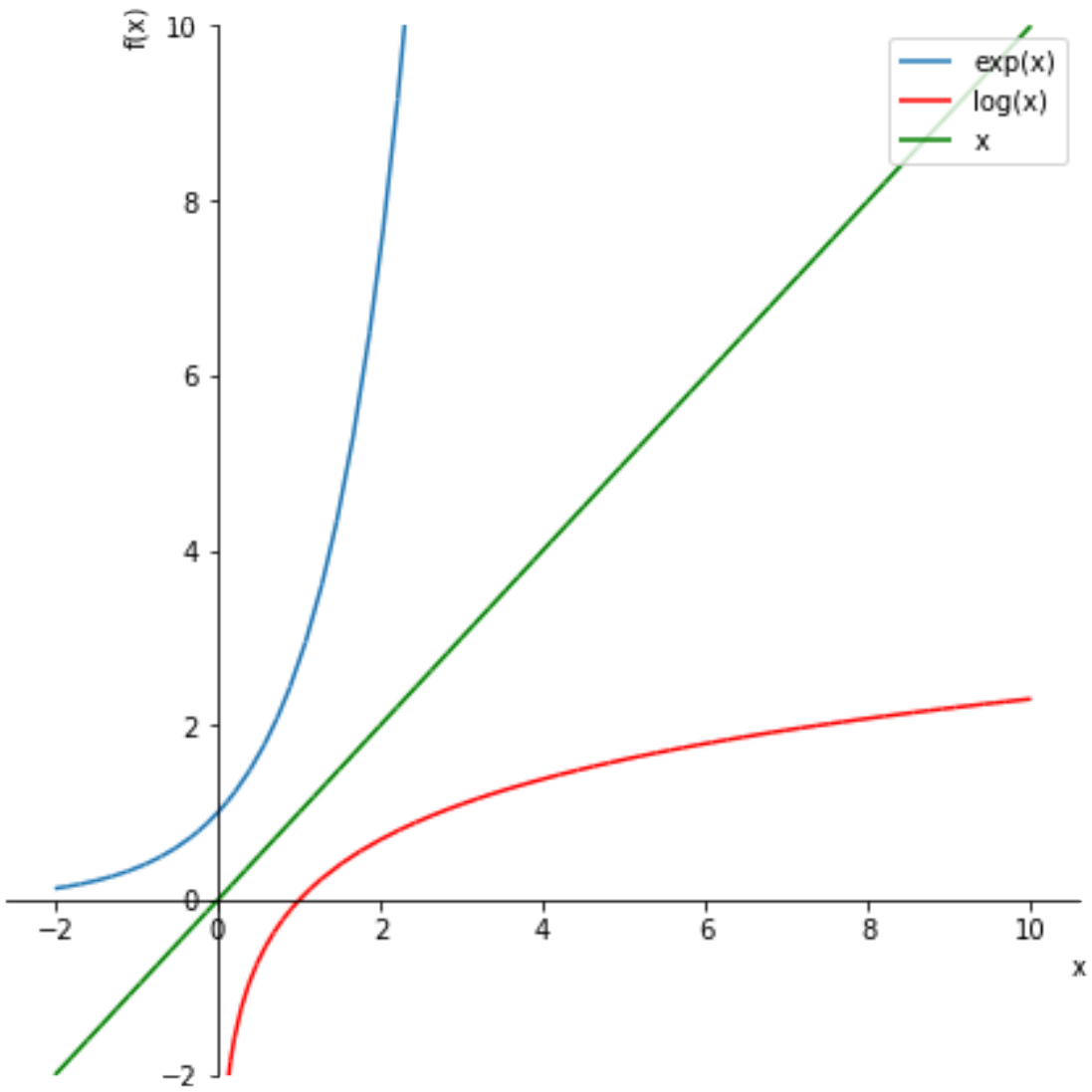
Acho que conseguimos um bom resultado. Contudo, você concorda que é um pouco cansativo acessar cada um desses valores e ir alterando, certo? E, se não definíssemos os limites em x para $[-10, 10]$, você veria que a função não continuaria. Isso ocorre pois não definimos o alcance de nosso plot ao criá-lo.

Aliado a isso, é possível criar dois plots diferentes e uní-los com o método `.extend()`.

Com o que aprendemos podemos fazer, com um único comando, um bom plot de uma única função. Caso queiramos mais de uma (como fizemos acima), basta alterar as cores de suas linhas.

Caso você conheça `matplotlib`, é possível utilizar todos os parâmetros para anotações e etc. Não abordarei isso aqui pois acredito que foge do nosso objetivo, mas é algo interessante.

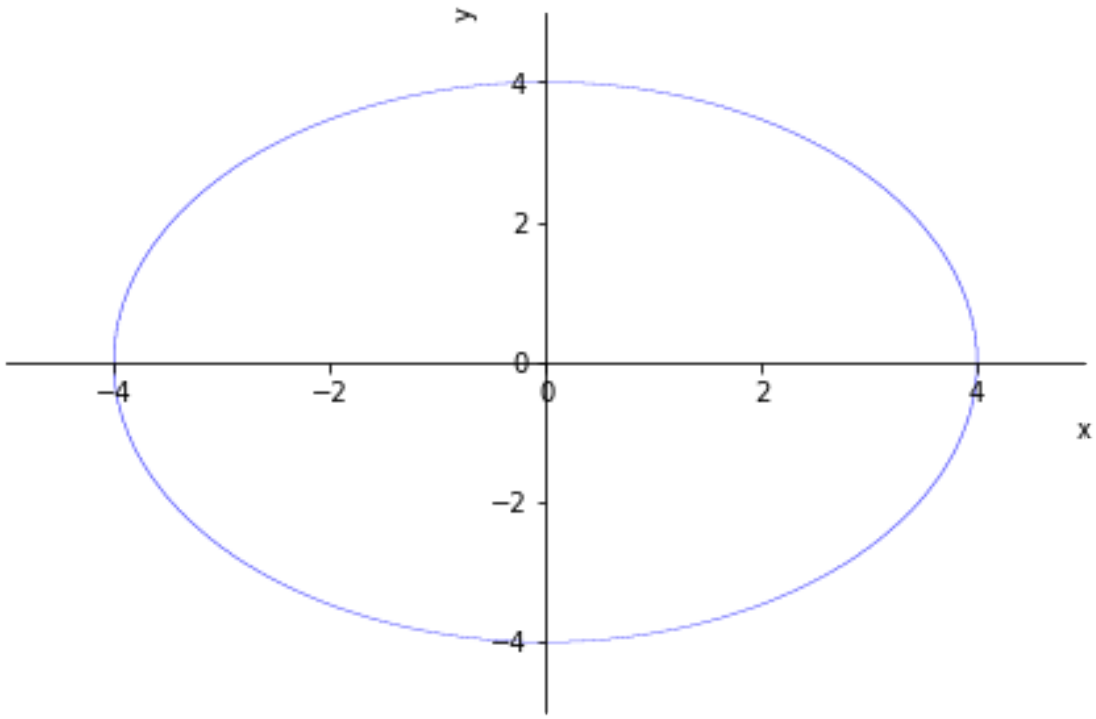
[7]:
```
p1 = plot(exp(x), log(x),x, (x, -2, 10), ylim = (-2,10), legend = True, size =␣
↪(6,6), show=False)
p1[1].line_color = 'red'
p1[2].line_color = 'green'
p1.show()
```



1.1.2 Equações Implícitas

Quando temos uma equação com variável implícita, utilizamos a `plot_implicit()`. Ela segue a mesma lógica de `plot()` em sua construção. Veja uma circunferência.

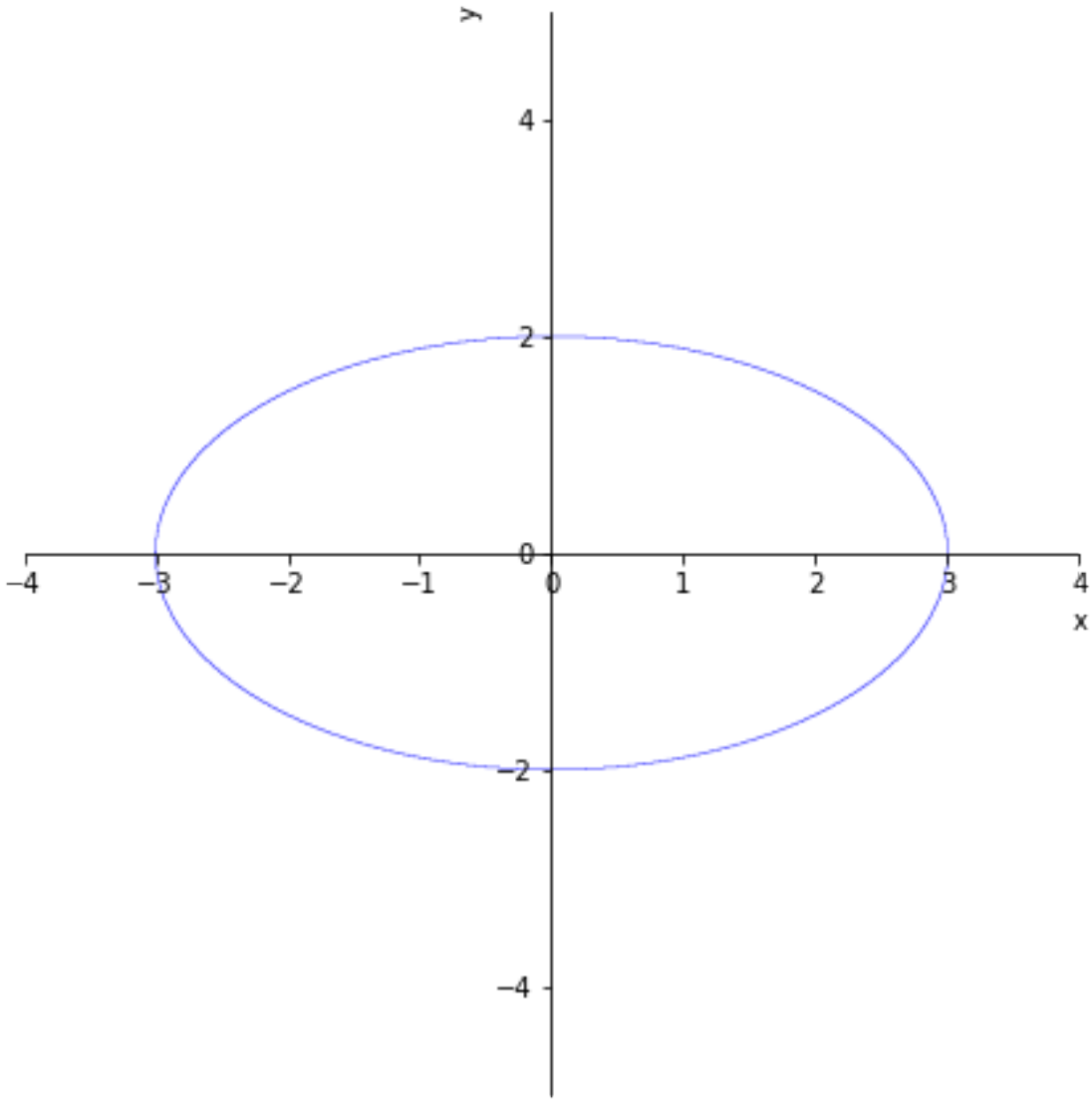
[8]:
```
cir = Eq(x**2 + y**2, 16) # R = 2
plot_implicit(cir)
```



[8]:
```
<sympy.plotting.plot.Plot at 0x7f21f4425ad0>
```

Também podemos elaborá-lo.

[9]:
```
plot_implicit(Eq(x**2/9 + y**2/4, 1), (x, -4, 4), ylim = (-4,4), size = (6,6))␣
↪# Elípse
# axis = False, deixa sem os eixos
```

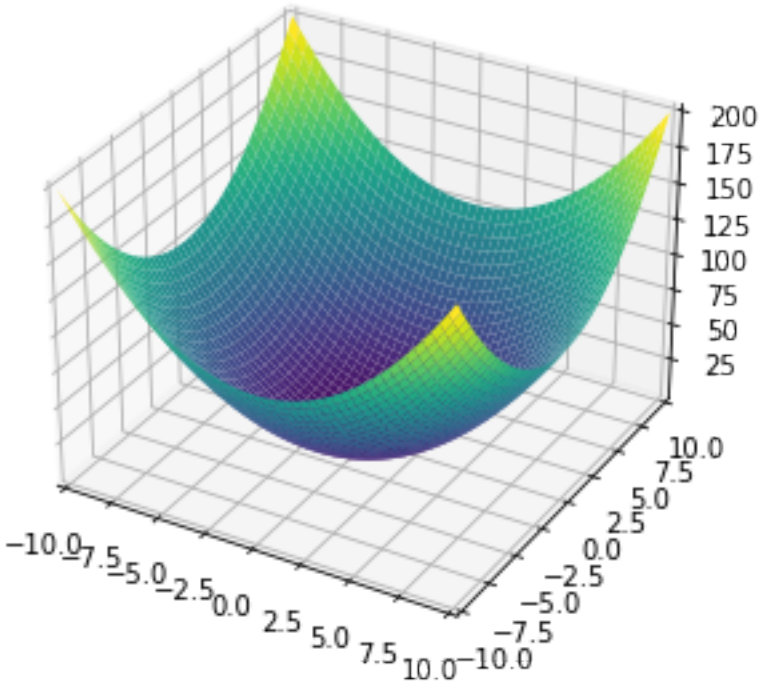


[9]: <sympy.plotting.plot.Plot at 0x7f21f4553b50>

1.2 Plot 3D

Utilizando funções de duas variáveis, nós conseguimos fazer plots em 3d com a função `plot3d()`.

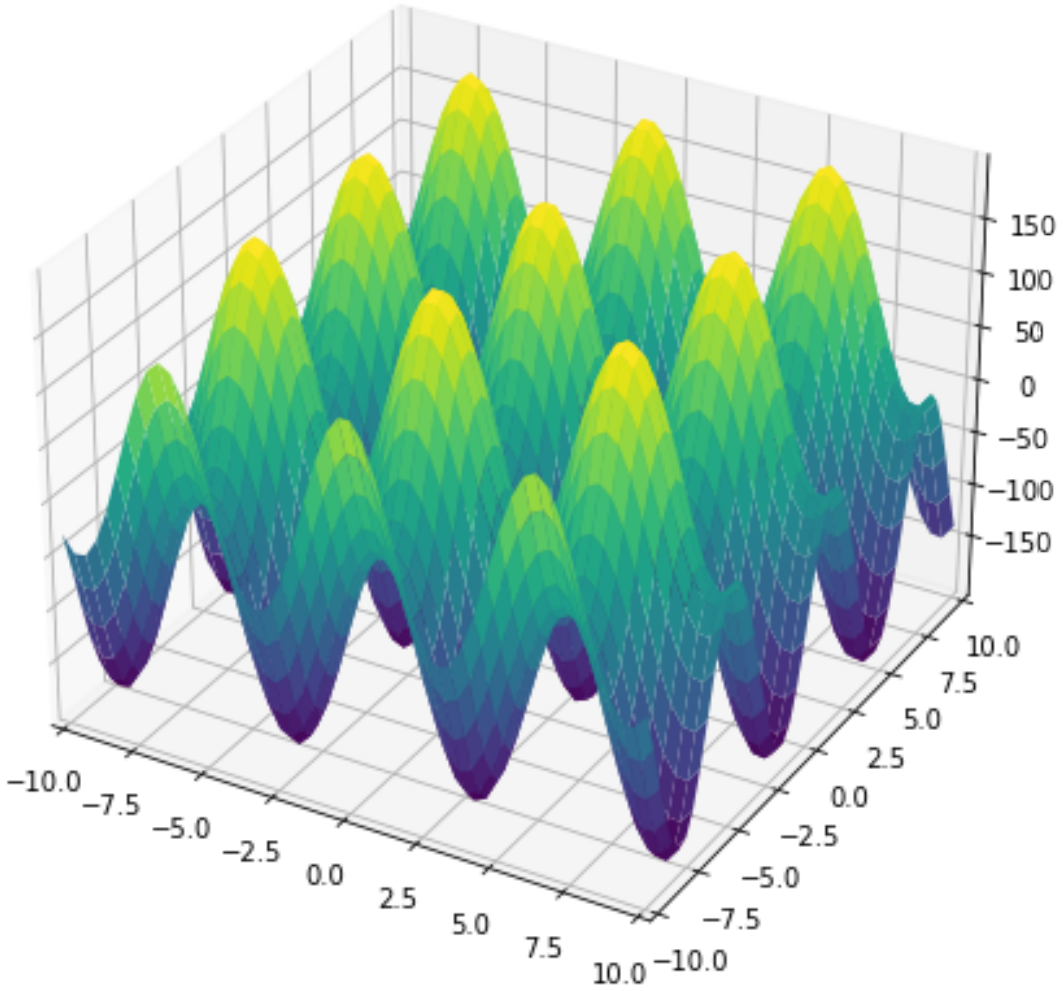
[10]:
```
from sympy.plotting import plot3d # Caso queira importá-la diretamente
plot3d(x**2 + y**2)
```



[10]:
```
<sympy.plotting.plot.Plot at 0x7f21f43d7510>
```

Os parâmetros são muito parecidos com a função `plot()`

[11]:
```
plot3d(100*(sin(x) + cos(y)), size = (6,6))
```



[11]: `<sympy.plotting.plot.Plot at 0x7f21f41bd8d0>`

Além desses plots, há os plots paramétricos. Recomendo que dê uma olhada na documentação. No mais, é realmente simples criar plots no Sympy.

1.3 Exercícios

1. Faça o plot das seguintes funções, escolhendo os melhores valores para os parâmetros:

$$f(x) = 4x^2 - 3x + 26$$

$$g(x) = \log(x^2 + 10)$$

$$h(x) = 10x^4 + 7x^3 - 10x + 20$$

$$p(x) = \sin(x^2 - x \cdot \pi) + \cos\left(x + \frac{\pi}{6}\right)$$

2. Faça o plot das seguintes equações, escolhendo os melhores valores para os parâmetros:

$$2x - 5y = 20$$

$$x^2 + y^2 = 60$$

$$\frac{x^2}{10} - \frac{y^2}{8} = 1$$

$$2x - 5y^2 = 10$$