lektion7

January 29, 2018

Table of Contents

1 Lektion 7

1.1 Gleichungen mit Parameter

In [1]:
```
from sympy import *
init_printing()
import matplotlib.pyplot as plt
import numpy as np
%matplotlib notebook
#%matplotlib inline
x,y,z,a,b,c,d = symbols('x y z a b c d')
```

In [2]:
```
solve(Eq(exp(-a*x),3*x**a),x)
```

Out[2]:

$$\left[\text{LambertW}\left(3^{-\frac{1}{a}}\right)\right]$$

In [3]:
```
sol = solve(x*exp(x)-a,x)
sol
```

Out[3]:

$$[\text{LambertW}\,(a)]$$

In [4]:
```
sol[0].subs(a,1).n()
```

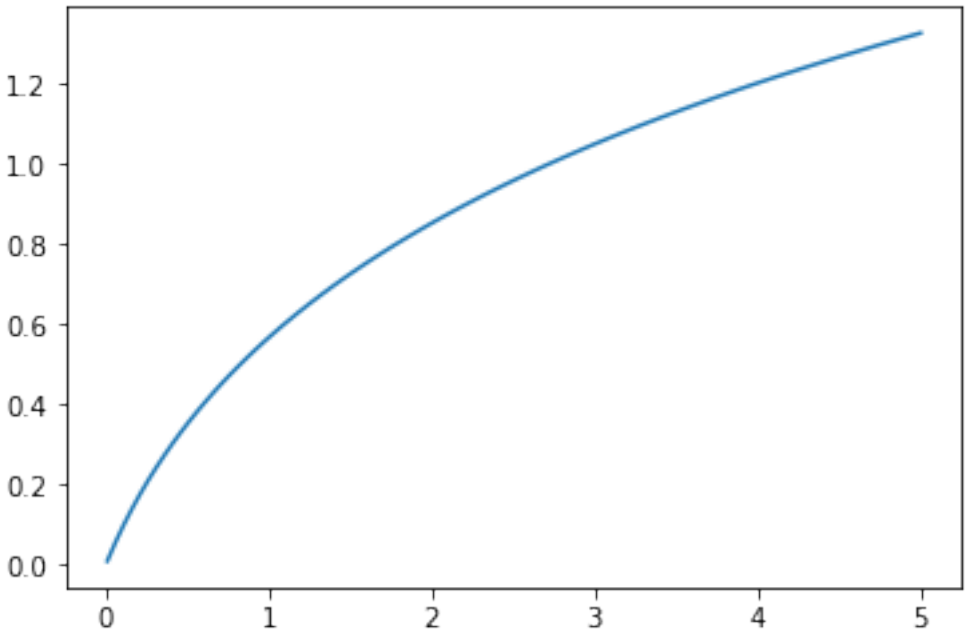
Out[4]:

0.567143290409784

```
In [5]: from scipy.special.lambertw import lambertw
        soln = lambdify(a,sol[0],modules={"LambertW" : lambertw})
        soln(1)
```

```
Out[5]: (0.5671432904097838+0j)
```

```
In [6]: xn = np.linspace(0.01,5,100)
        yn = soln(xn)
        plt.plot(xn,np.real(yn));
```



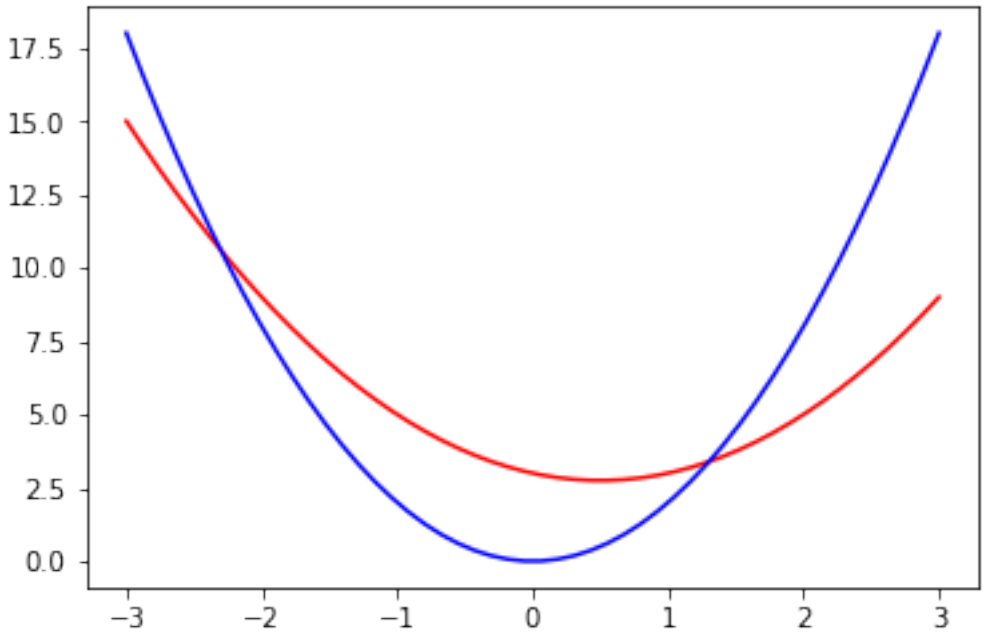
1.2 Ungleichungen

```
In [7]: f = x**2 - x + 3
        g = 2*x**2
        solve(f>g)
```

Out[7]:

$$x < -\frac{1}{2} + \frac{\sqrt{13}}{2} \wedge -\frac{\sqrt{13}}{2} - \frac{1}{2} < x$$

```
In [8]: fig = plt.figure()
        ax = fig.gca()
        xn = np.linspace(-3,3)
        ax.plot(xn,lambdify(x,f)(xn),'r')
        ax.plot(xn,lambdify(x,g)(xn),'b');
```




```
In [9]: solveset(sin(x)>cos(x),x,domain=S.Reals)
```

Out[9]:

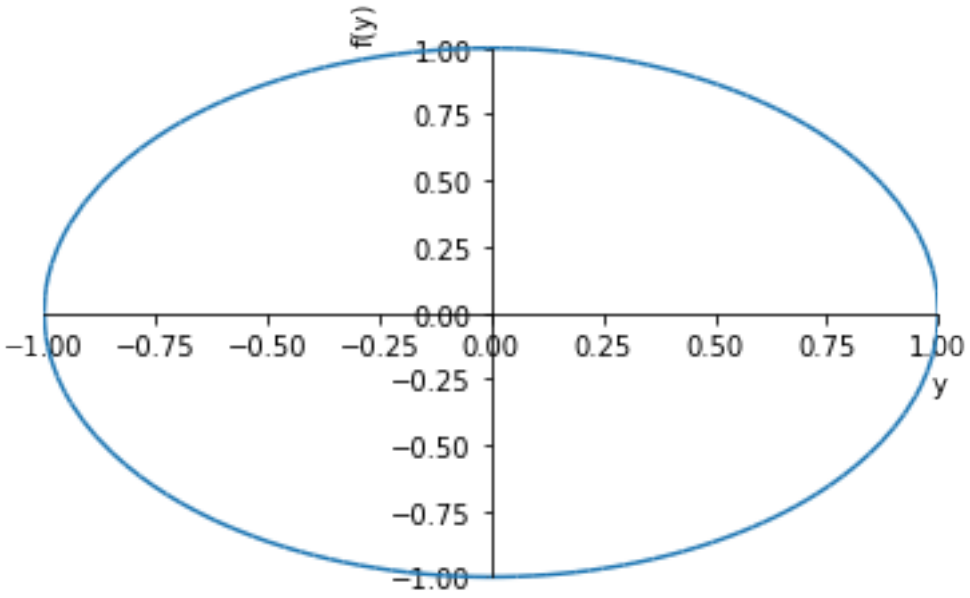
$$\left(\frac{\pi}{4}, \frac{5\pi}{4}\right)$$

```
In [10]: solve(sin(x)>cos(x),x)
```

Out[10]:

$$\frac{\pi}{4} < x \wedge x < \frac{5\pi}{4}$$

```
In [11]: gl = Eq(x**2+y**2,1)
         sol = solve(gl)
         fig3 = plot(sol[0][x],sol[1][x],(y,-1,1))
         fig3._backend.ax.set_aspect('equal')
```



In [12]:
```
gl = Eq(x*y,0)
sol = solve(gl)
sol
```

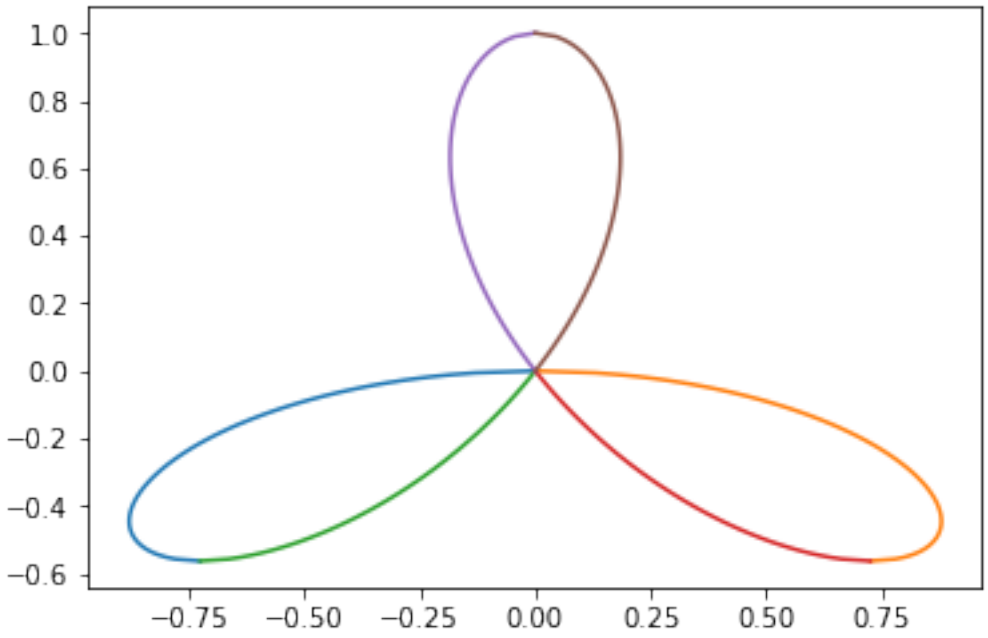
Out[12]:

$$[\{x : 0\}, \quad \{y : 0\}]$$

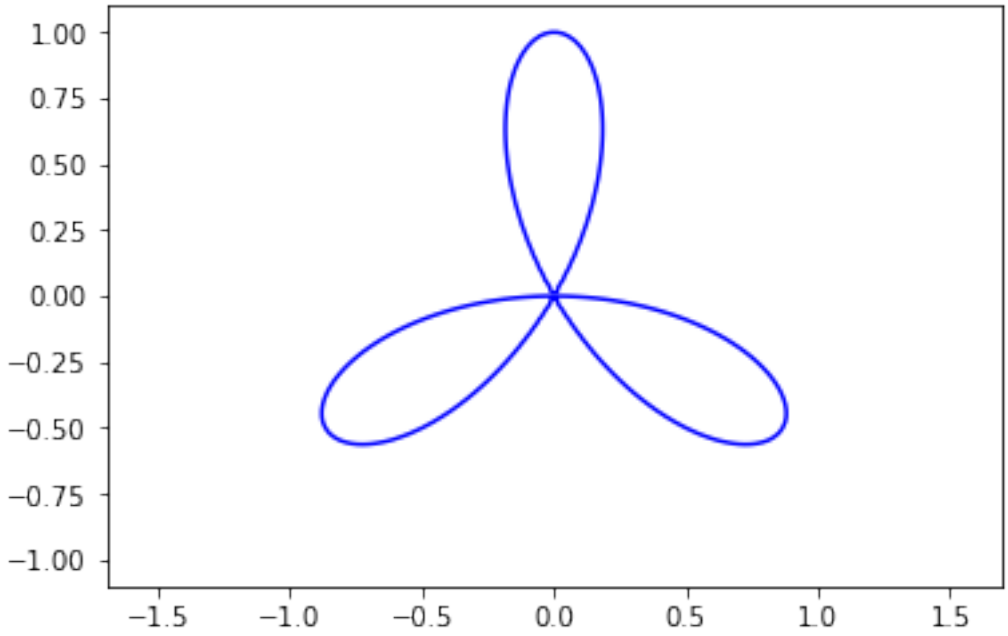
In [13]:
```
gl = Eq((x**2 + y**2)**2 + 3*x**2*y - y**3)
sols = solve(gl)
fig = plt.figure()
ax = fig.gca()

yn = np.linspace(-9/16,0,100)
for sol in sols:
    ax.plot(lambdify(y,sol[x])(yn),yn)

yn = np.linspace(0,1,100)
ax.plot(lambdify(y,sols[2][x])(yn),yn)
ax.plot(lambdify(y,sols[3][x])(yn),yn);
```



In [14]:
```
fig = plt.figure()
ax = fig.gca()
xn = np.linspace(-1.1,1.1,100)
X,Y = np.meshgrid(xn,xn)
ax.contour(X,Y,lambdify((x,y),gl.lhs)(X,Y),[0],colors='blue')
ax.axis('equal');
```



1.3 Gleichungssysteme

```
In [15]: glnS = {Eq(x+y,a),Eq(2*x-b*y,3)}
         gln = (Eq(x+y,a),Eq(2*x-b*y,3))
         gln
```

Out[15]:

$$(x + y = a, \quad -by + 2x = 3)$$

```
In [16]: sol = solve(gln,(x,y))
         sol
```

Out[16]:

$$\left\{x : \frac{ab+3}{b+2}, \quad y : \frac{2a-3}{b+2}\right\}$$

```
In [17]: gln = {Eq(x**2+y**2-1,0),Eq(x-y,0)}
         gln
```

Out[17]:

$$\{x - y = 0, x^2 + y^2 - 1 = 0\}$$

```
In [18]: lsg = solve(gln,{x,y})
         lsg
```

Out[18]:

$$\left[\left\{x : -\frac{\sqrt{2}}{2}, \quad y : -\frac{\sqrt{2}}{2}\right\}, \quad \left\{x : \frac{\sqrt{2}}{2}, \quad y : \frac{\sqrt{2}}{2}\right\}\right]$$

```
In [19]: [gl.subs(l) for l in lsg for gl in gln ]
```

Out[19]:

$$[\text{True}, \quad \text{True}, \quad \text{True}, \quad \text{True}]$$

```
In [20]: list(gln)[0].lhs , list(gln)[1]
```

Out[20]:

$$(x - y, \quad x^2 + y^2 - 1 = 0)$$

```
In [21]: [gl.lhs for gl in gln]
```

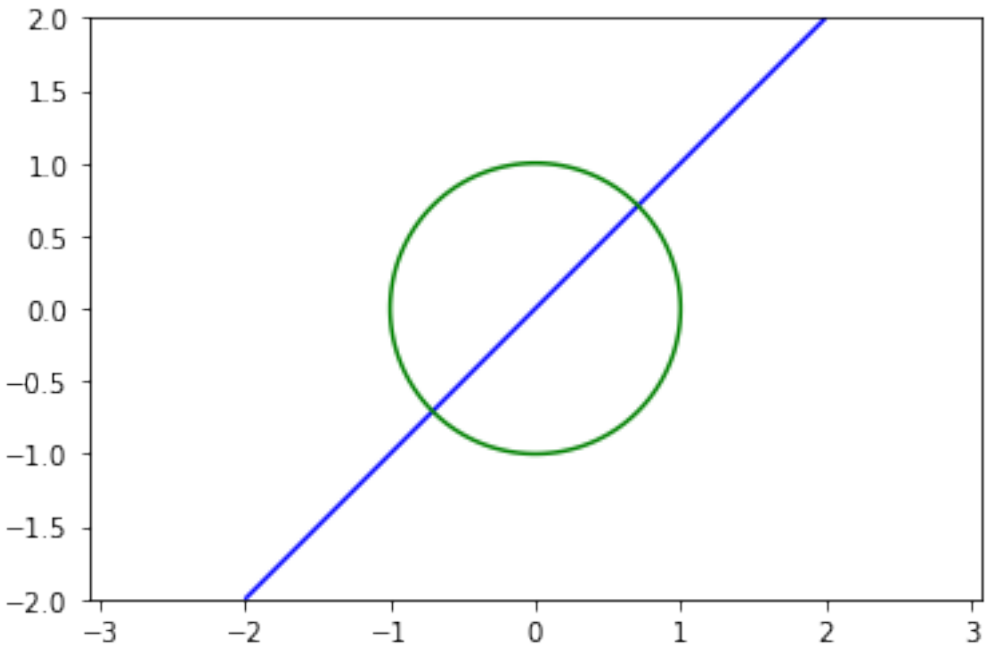
Out[21]:

$$[x - y, \quad x^2 + y^2 - 1]$$

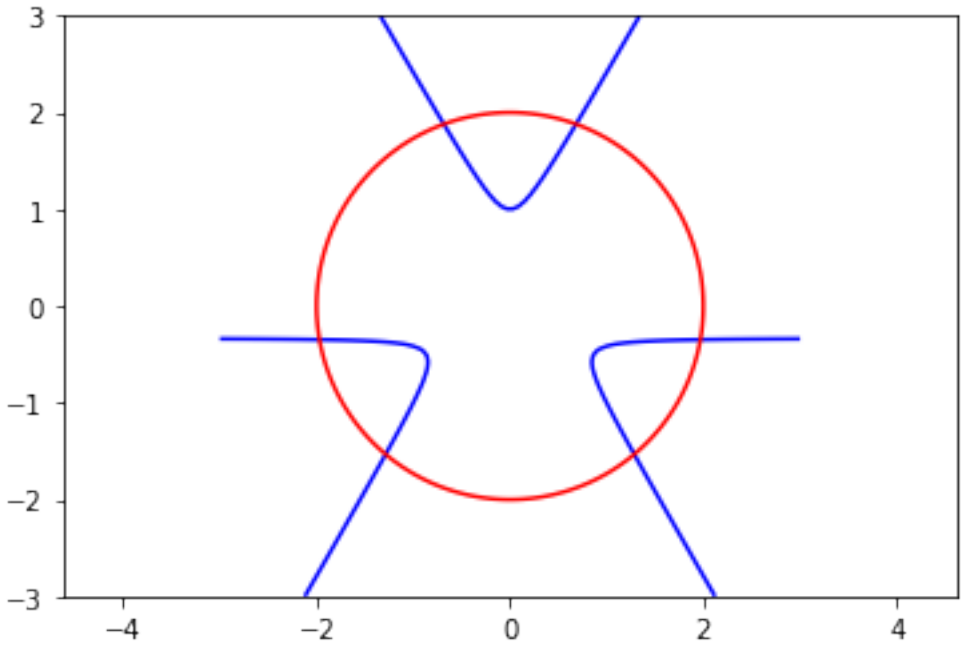
In [22]:
```
xn = np.linspace(-2,2,100)
X,Y = np.meshgrid(xn,xn)
fig = plt.figure()
ax = fig.gca()
ax.contour(X,Y,lambdify((x,y),list(gln)[0].lhs)(X,Y),[0],colors='blue')
ax.contour(X,Y,lambdify((x,y),list(gln)[1].lhs)(X,Y),[0],colors='green')
ax.axis('equal')
```

Out[22]:

$$(-2.0, \quad 2.0, \quad -2.0, \quad 2.0)$$



In [23]:
```
f = x**2+y**2+3*x**2*y-y**3
g = x**2+y**2
xn = np.linspace(-3,3,100)
X,Y = np.meshgrid(xn,xn)
fig = plt.figure()
ax = fig.gca()
ax.contour(X,Y,lambdify((x,y),f)(X,Y),[0],colors='blue')
ax.contour(X,Y,lambdify((x,y),g)(X,Y),[4],colors='red')
ax.axis('equal');
```



In [24]:
```
sol = solve({f, x**2+y**2-4})
x0 = sol[0]
x0
```

Out[24]:

$$\left\{x : -2^{\frac{2}{3}}\sqrt{\frac{1}{\left(1+\sqrt{3}i\right)^{\frac{10}{3}}}\left(2-2\sqrt{3}i-4\cdot 2^{\frac{2}{3}}\sqrt[3]{1+\sqrt{3}i}+\sqrt[3]{2}\left(1+\sqrt{3}i\right)^{\frac{2}{3}}+\sqrt[3]{2}\sqrt{3}i\left(1+\sqrt{3}i\right)^{\frac{2}{3}}\right)},\quad y : \frac{}{\left(-\frac{1}{2}+\frac{\sqrt{3}i}{2}\right.}\right.$$

In [25]:
```
for l in sol:
    print('x {0: 12.9f} + {1: 12.9f} i \t'.format(float(re(l[x]).n()), float(im(l[x]).n
          ' y {0: 12.9f} + {1: 12.9f} i'.format(float(re(l[y]).n()), float(im(l[y]).n()
```

```
x -1.285575219 + -0.000000000 i 	 y -1.532088886 + -0.000000000 i
x  1.285575219 +  0.000000000 i 	 y -1.532088886 + -0.000000000 i
x -1.969615506 + -0.000000000 i 	 y -0.347296355 + -0.000000000 i
x  1.969615506 +  0.000000000 i 	 y -0.347296355 + -0.000000000 i
x -0.684040287 + -0.000000000 i 	 y  1.879385242 +  0.000000000 i
x  0.684040287 + -0.000000000 i 	 y  1.879385242 +  0.000000000 i
```

In [26]:
```
im(sol[4][y]).n()
```

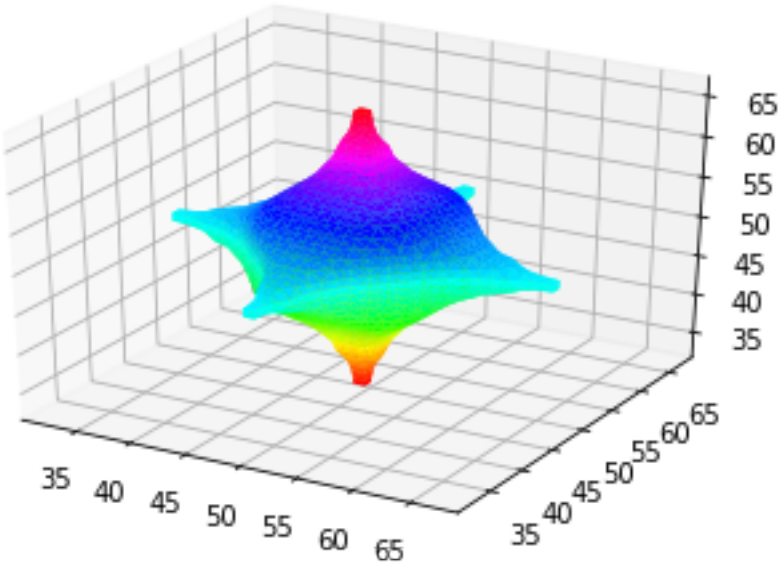
Out[26]:

$$0$$

1.3.1 Eine Isoflaeche

```
In [27]: from skimage import measure
         from mpl_toolkits.mplot3d import Axes3D
         xn=np.linspace(-5,5,100)
         X,Y,Z = np.meshgrid(xn,xn,xn)
         vol = X**4+Y**4+Z**4+1000*(X**4+Y**4)*(X**4+Z**4)*(Y**4+Z**4)-10
         #vol = X**2+Y**2+Z**2-25
         verts, faces, _, __ = measure.marching_cubes(vol, 0)
         fig = plt.figure()
         ax = fig.add_subplot(111, projection='3d')
         ax.plot_trisurf(verts[:,0],verts[:,1],faces,verts[:,2],cmap='hsv')
```

```
Out[27]: <mpl_toolkits.mplot3d.art3d.Poly3DCollection at 0x7f1d80a21d30>
```



1.4 Python rechnet komplex

```
In [28]: z = x + I*y
         z
```

Out[28]:

$$x + iy$$

```
In [29]: z**4
```

Out[29]:

$$(x + iy)^4$$

In [30]:
```
re(z**4).expand()
```

Out[30]:

$$(\Re(x))^4 - 4(\Re(x))^3 \Im y - 6(\Re(x))^2 (\Re(y))^2 - 12(\Re(x))^2 \Re(y)\Im x - 6(\Re(x))^2 (\Im x)^2 + 6(\Re(x))^2 (\Im y)^2 + 12\Re(x) (\Re$$

In [31]:
```
abs(z)
```

Out[31]:

$$|x + iy|$$

In [32]:
```
re(z)
```

Out[32]:

$$\Re(x) - \Im y$$

In [33]:
```
im(z)
```

Out[33]:

$$\Re(y) + \Im x$$

In [34]:
```
x, y = symbols('x y',real=True)
z = x + I*y
```

In [35]:
```
re(z**4).expand()
```

Out[35]:

$$x^4 - 6x^2y^2 + y^4$$

In [36]:
```
abs(z)
```

Out[36]:

$$\sqrt{x^2 + y^2}$$