Производная

Вычесление производных

In [3]:

```
from sympy import *
```

In [2]:

```
# Пример 1
x = symbols('x')
y = x * cos(x)
diff(x*cos(x), x)
```

Out[2]:

$$-x \sin(x) + \cos(x)$$

In [3]:

```
# Пример 2
diff(log(x), x, 3)
```

Out[3]:

$$\frac{2}{x^3}$$

In [5]:

```
diff(log(x), x, x, x)
```

Out[5]:

$$\frac{2}{x^3}$$

In [6]:

```
# Пример 3
y = log(x**3, 10)**3
diff(y, x, 2).subs(x, 10)
```

Out[6]:

$$-\frac{9\,(-6 + \log(1000))\log(1000)}{100\log(10)^3}$$

In [7]:

```
# Пример 4
y = (x**2+x-6)/(x**2-10*x+25)
z = diff(y, x)
z
```

Out[7]:

$$\frac{(10-2x)\left(x^2+x-6\right)}{\left(x^2-10x+25\right)^2}+\frac{2x+1}{x^2-10x+25}$$

In [8]:

```
solve(z, x)
```

Out[8]:

```
[7/11]
```

Производная неявной функции

In [10]:

```
# Пример 5
x = symbols('x')
y = symbols('y')
f = x**2 + y**2 - 4
idiff(f, y, x)
```

Out[10]:

$$-\frac{x}{y}$$

In [11]:

```
f = x**2 + y**2 - 4
idiff(f, y, x, 2)
```

Out[11]:

$$-\frac{\frac{x^2}{y^2}+1}{y}$$

In [12]:

```
idiff(f, y, x, 2).simplify()
```

Out[12]:

$$\frac{-x^2-y^2}{y^3}$$

Производная функции, заданной в параметрической форме

In [14]:

```
# Пример 6
t = symbols('t')
x = t - sin(t)
y = 1 - cos(t)

y_diff = diff(y, t)/diff(x, t)
y_diff
```

Out[14]:

$$\frac{\sin(t)}{1-\cos(t)}$$

In [15]:

```
y_2diff = diff(y_diff, t)/diff(x, t)
y_2diff.simplify()
```

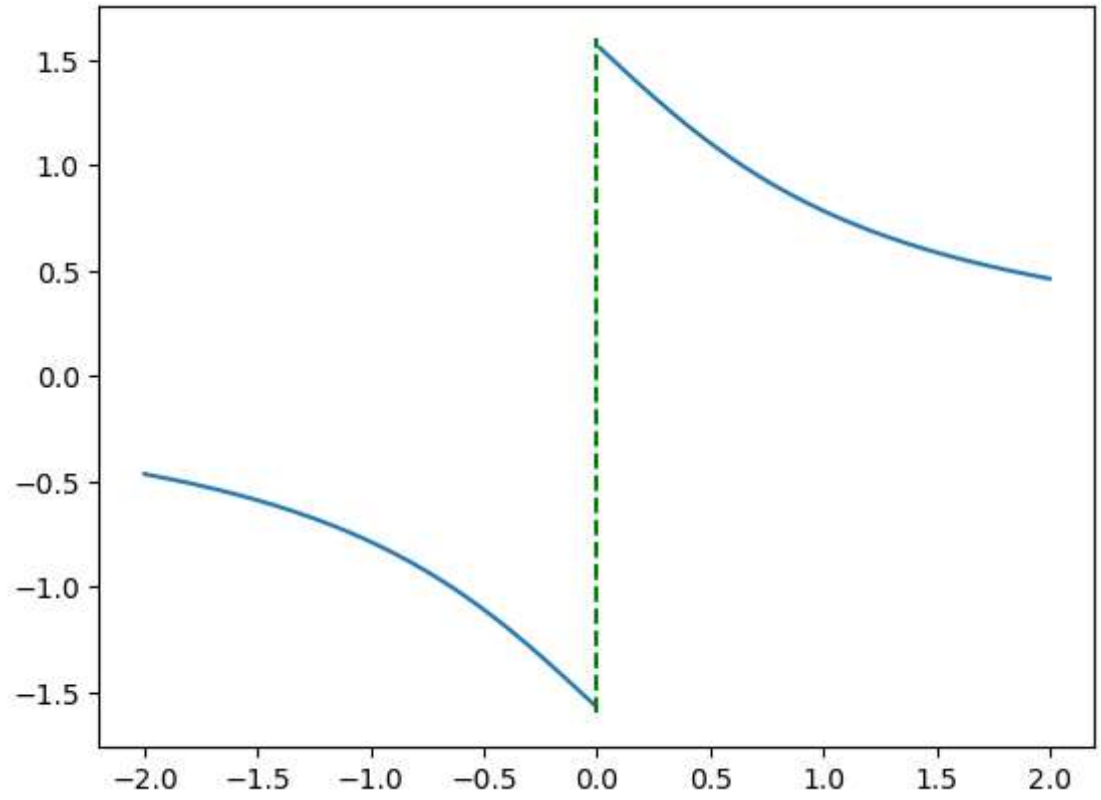
Out[15]:

$$-\frac{1}{(\cos(t)-1)^2}$$

Односторонняя производная

In [37]:

```
# Пример 7
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
x = np.linspace(-2, 2, 500)

x[(x>-0.01)&(x < 0.01)] = np.nan
y = np.arctan(1/x)
plt.plot(x, y)
plt.vlines(0, -1.6, 1.6, color='g', linestyles='dashed')
plt.show()
```



In [39]:

```
x = symbols('x')
y = atan(1/x)
z = diff(y,x)
limit(z, x, 0, dir='+')
```

Out[39]:

$$-1$$

In [40]:

```
limit(z, x, 0, dir='-')
```

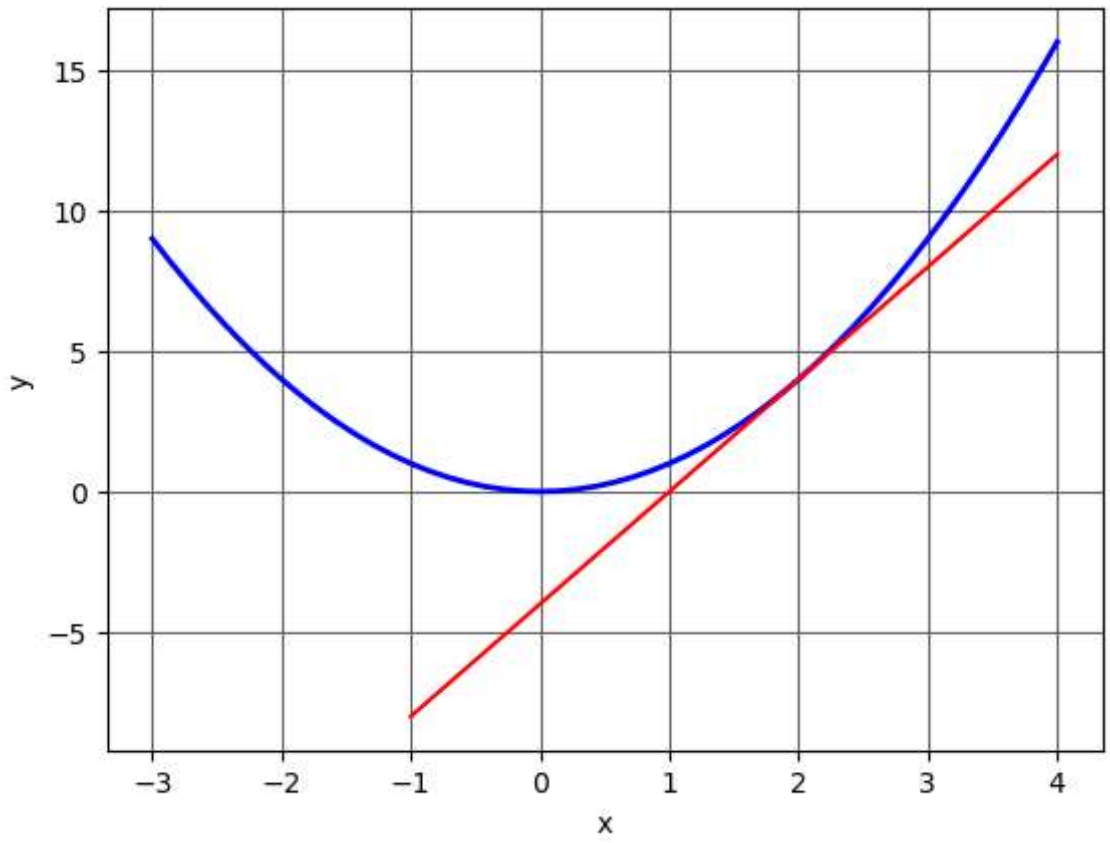
Out[40]:

$$-1$$

Применение производной при исследовании функции. Уравнение касательной

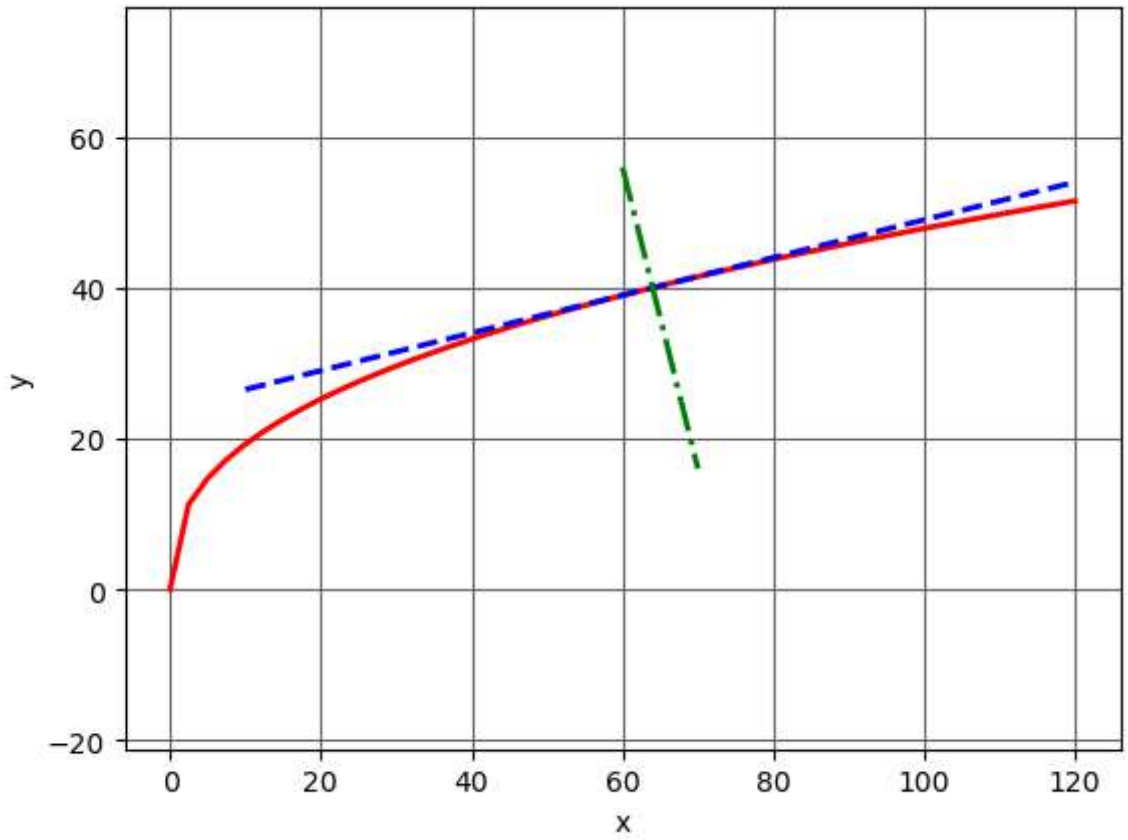
In [41]:

```
# Пример 8
x = np.linspace(-3,4,50)
y1 = x**2
plt.plot(x,y1,lw=2,c='b')
x = np.linspace(-1,4,50)
y2 = 4*x - 4
plt.plot(x,y2,c='r')
plt.xlabel('x')
plt.ylabel('y')
plt.grid(True, linestyle='-', color='0.4')
plt.show()
```



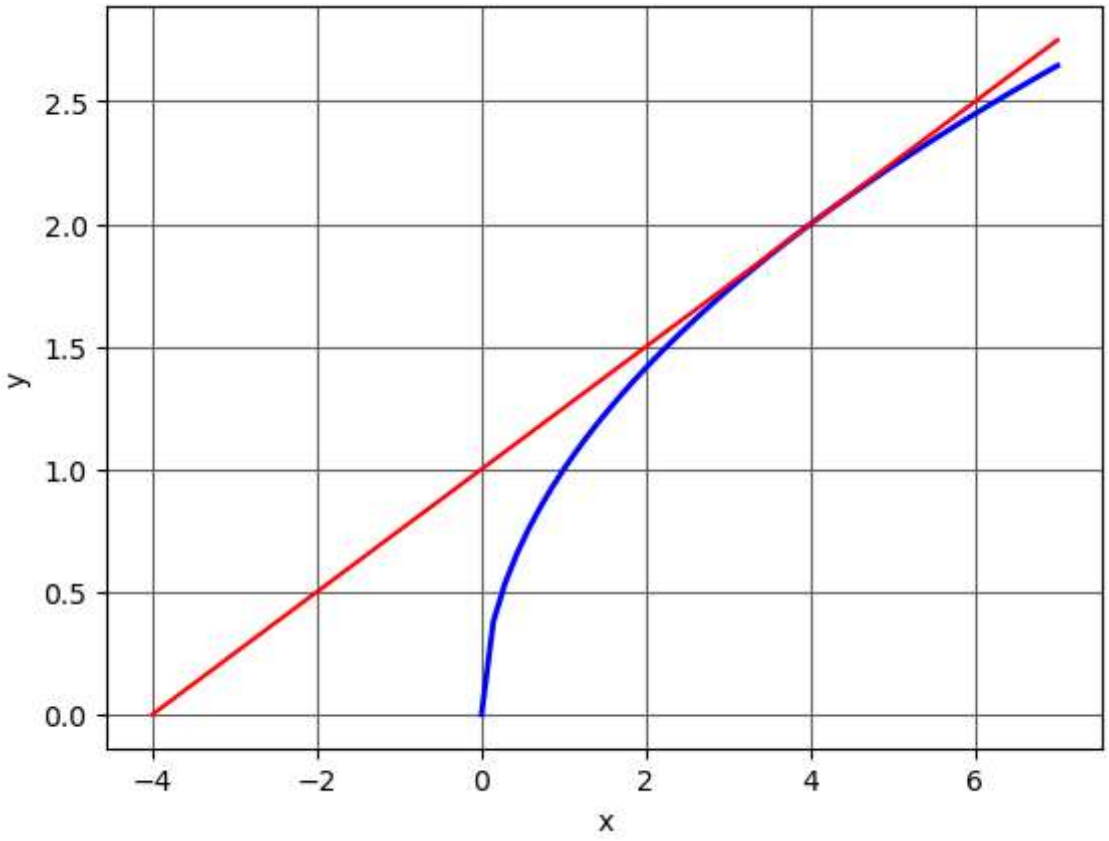
In [42]:

```
# Пример 9
x = np.linspace(0,120,50)
y1 = 6*x**(1/3) + 2*x**(1/2)
plt.plot(x,y1,lw=2,c='r')
x = np.linspace(10,120,50)
y2 = x/4 + 24
plt.plot(x,y2,'--',lw=2,c='b')
x = np.linspace(60,70,50)
y3 = 296 - 4*x
plt.plot(x,y3,'-.',lw=2,c='g')
plt.xlabel('x')
plt.ylabel('y')
plt.grid(True, linestyle='-', color='0.4')
plt.axis('equal')
plt.show()
```



In [43]:

```
# Пример 10
x = np.linspace(0,7,50)
y1 = np.sqrt(x)
plt.plot(x,y1,lw=2,c='b')
x = np.linspace(-4,7,50)
y2 = x/4 + 1
plt.plot(x,y2,c='r')
plt.xlabel('x')
plt.ylabel('y')
plt.grid (True, linestyle='-', color='0.4')
plt.show()
```



Исследование функции

In [44]:

```
# Пример 11
x, y = symbols('x y')
solve(x**2 < 3)
```

Out[44]:

$$-\sqrt{3} < x \wedge x < \sqrt{3}$$

In [45]:

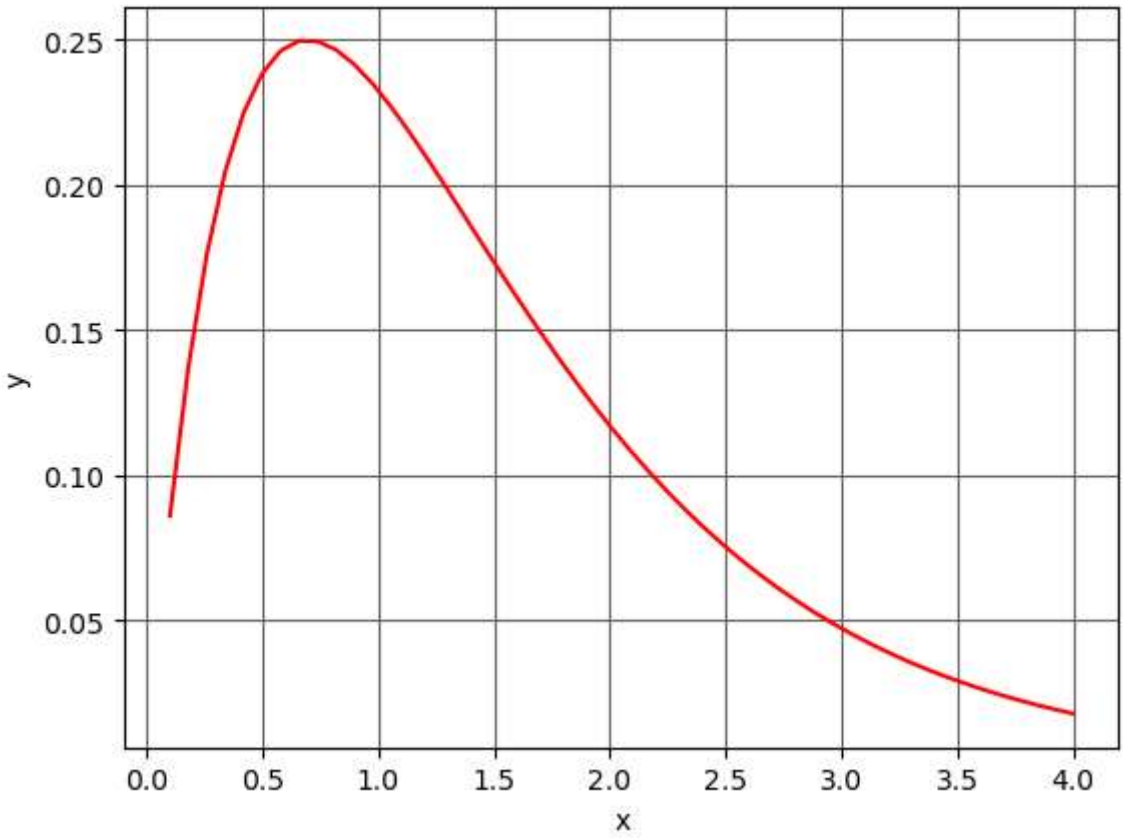
```
# Пример 12
solve(x**2 - y**2, x)
```

Out[45]:

```
[-y, y]
```

In [46]:

```
# Пример 13
f = lambda x: np.exp(-x) - np.exp(-2*x)
x = np.linspace(0.1,4,50)
plt.plot(x, f(x), 'r')
plt.xlabel('x')
plt.ylabel('y')
plt.grid(True, linestyle='-', color='0.4')
plt.show()
```



In [47]:

```
from scipy.optimize import minimize

f_max = lambda x: -(np.exp(-x) - np.exp(-2*x))
res = minimize(f_max, -2)
res
```

Out[47]:

```
  message: Optimization terminated successfully.
  success: True
   status: 0
      fun: -0.249999999999441
        x: [ 6.931e-01]
      nit: 12
      jac: [-7.413e-07]
 hess_inv: [[ 1.986e+00]]
     nfev: 26
     njev: 13
```

In [48]:

```
# Пример 14
x = symbols('x')
y = x**3
x0 = solve(diff(y,x))[0]
print('x0: %.3f y(x0): %.3f' % (x0, y.subs(x, x0)))
```

```
x0: 0.000 y(x0): 0.000
```

In [49]:

```
diff(y,x,2).subs(x,x0)
```

Out[49]:

$$0$$

In [51]:

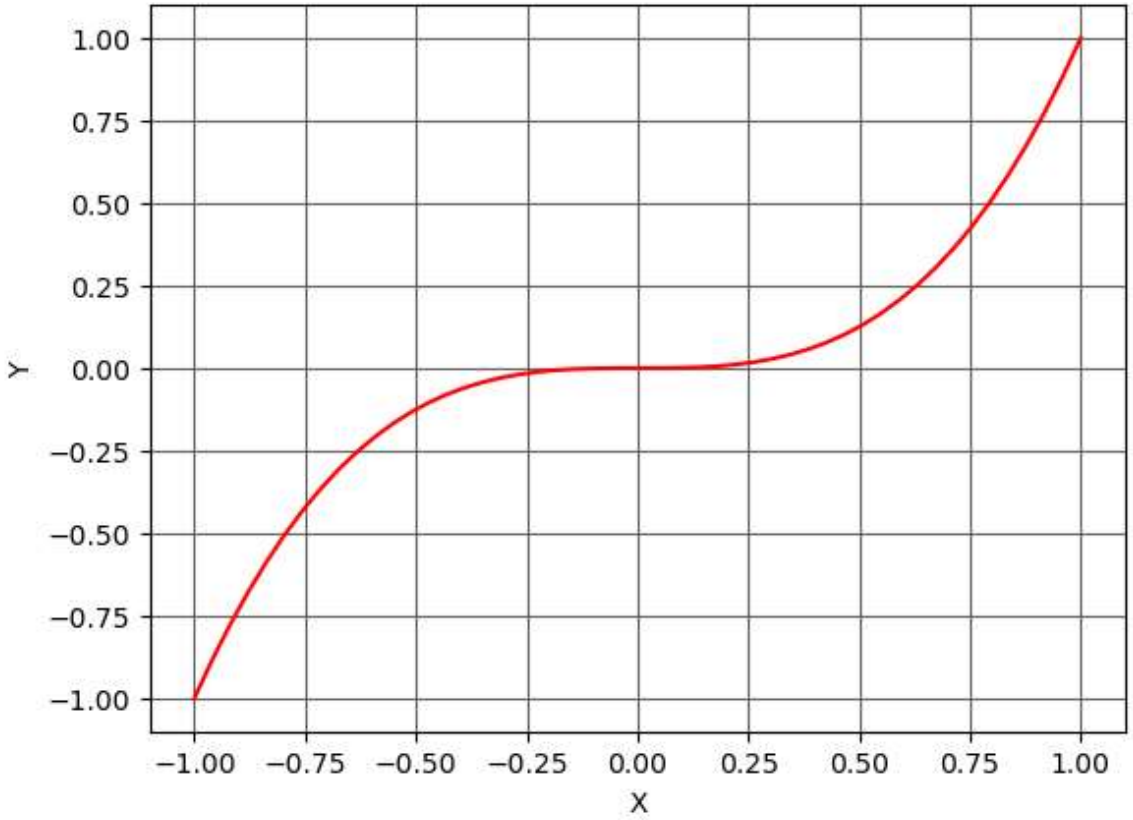
```
diff(y,x,3).subs(x,x0)
```

Out[51]:

$$6$$

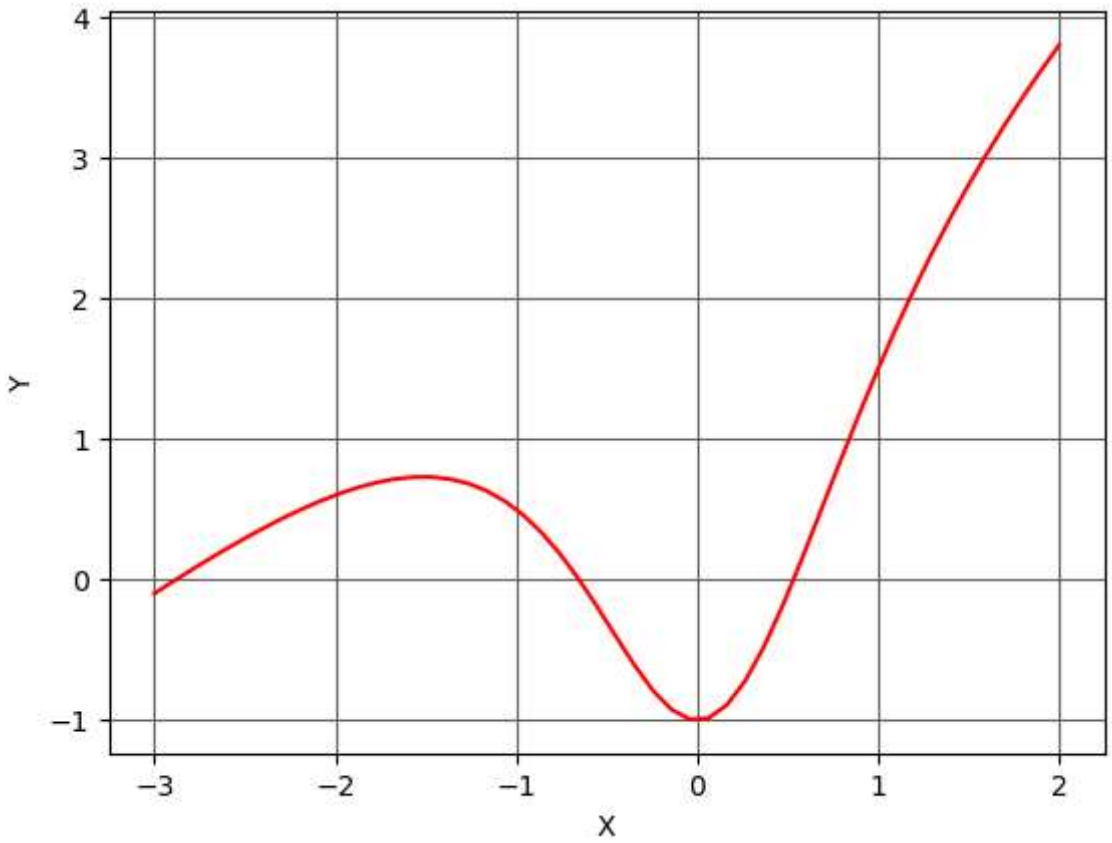
In [52]:

```
x = np.linspace(-1,1,50)
plt.plot(x, x**3, 'r')
plt.xlabel('X')
plt.ylabel('Y')
plt.grid(True, linestyle='-', color='0.4')
plt.show()
```



In [53]:

```
# Пример 15
f = lambda x: (x**3+3*x**2-1) / (x**2+1)
x = np.linspace(-3,2,50)
plt.plot(x, f(x), 'r')
plt.xlabel('X')
plt.ylabel('Y')
plt.grid(True, linestyle='-', color='0.4')
plt.show()
```



In [54]:

```
res = minimize(f, 1)
print('x_min: %.3f fmin: %.3f' % (res.x, f(res.x)))
```

```
x_min: 0.000 fmin: -1.000
```

In [55]:

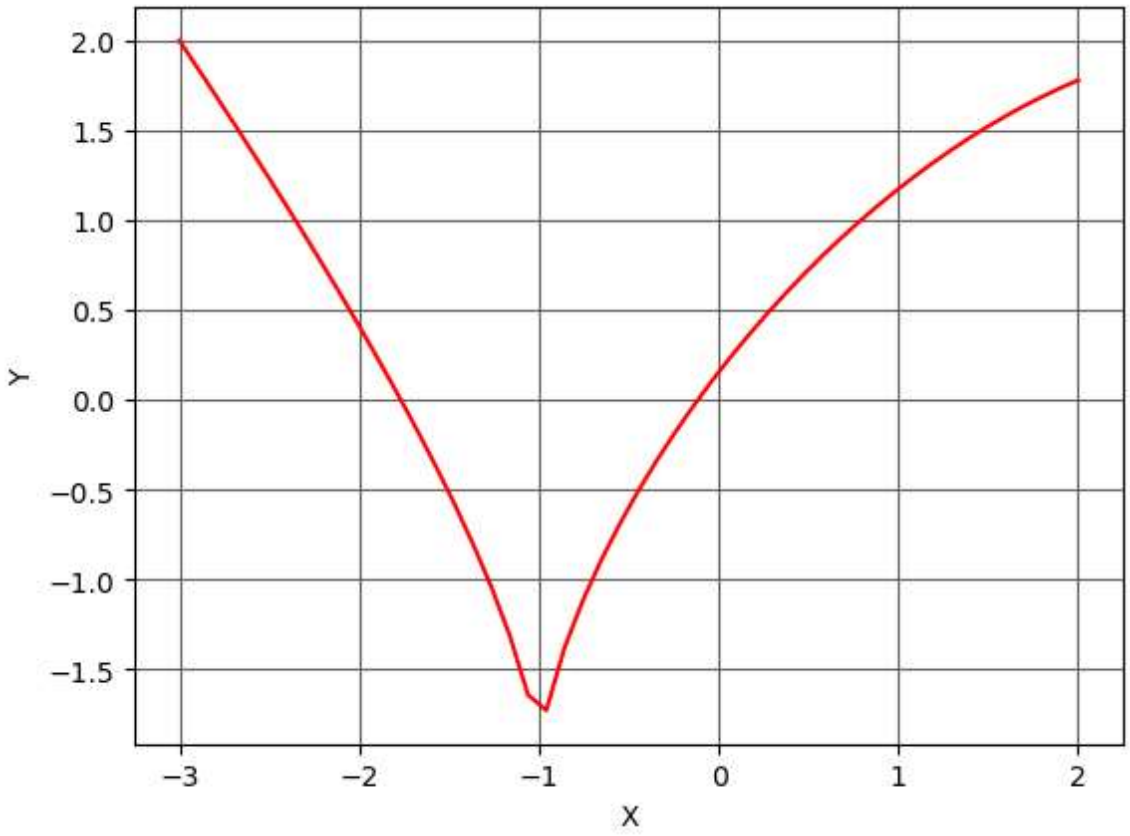
```
f_max = lambda x: -(x**3+3*x**2-1) / (x**2+1)
res = minimize(f_max, -2)
print('x_max: %.3f f max: %.3f' % (res.x, f(res.x)))
```

```
x_max: -1.513 f max: 0.731
```

Наибольшее и наименьшее значения на отрезке

In [56]:

```
fun = lambda x: np.cbrt(2*(x+1)**2*(5-x)) - 2
x = np.linspace(-3, 3, 100)
plt.xlabel('X')
plt.ylabel('Y')
plt.plot(x, fun(x), 'r')
plt.grid(True, linestyle='-', color='0.4')
plt.show()
```



In [57]:

```
res = minimize(fun, -1.5)
print('x_min: %.3f' % res.x)
```

```
x_min: -0.490
```

In [58]:

```
res = minimize(fun, -1.001)
print('xmin: %.3f' % res.x)
```

```
xmin: -1.001
```

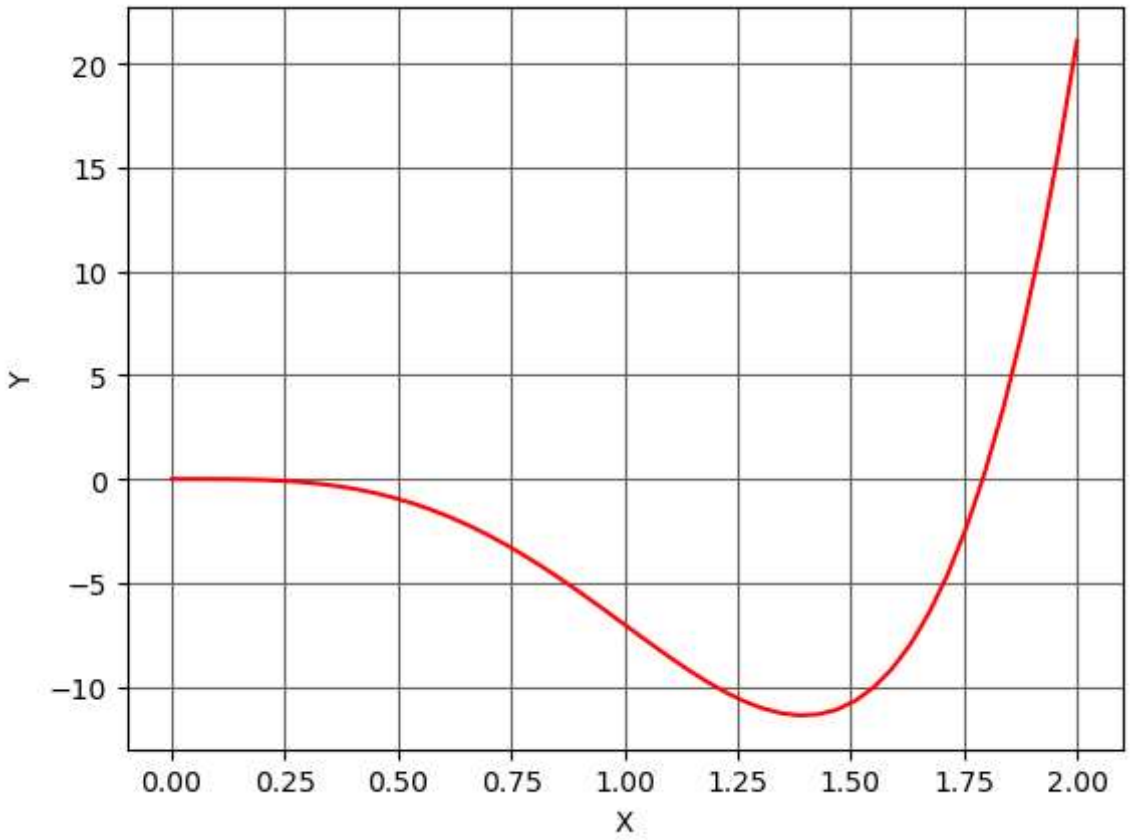
In [59]:

```
print('y(-3): %.3f y(3): %.3f' % (fun(-3), fun(3)))
```

```
y(-3): 2.000 y(3): 2.000
```

In [60]:

```
# Пример 17
f = lambda x: x**4 * (12*np.log(x) - 7)
x = np.linspace(0.001,2,50)
plt.plot(x, f(x), 'r')
plt.xlabel('X')
plt.ylabel('Y')
plt.grid(True, linestyle='-', color='0.4')
plt.show()
```



In [61]:

```
x = symbols('x')
y = x**4 * (12*log(x) - 7)
y_2deriv = diff(y,x,2)
y_2deriv
```

Out[61]:

$$144x^2 \log(x)$$

In [62]:

```
x_inflex = solve(y_2deriv, x)
x_inflex
```

Out[62]:

```
[0, 1]
```

In [63]:

```
diff(y,x,3).subs(x, 1)
```

Out[63]:

144

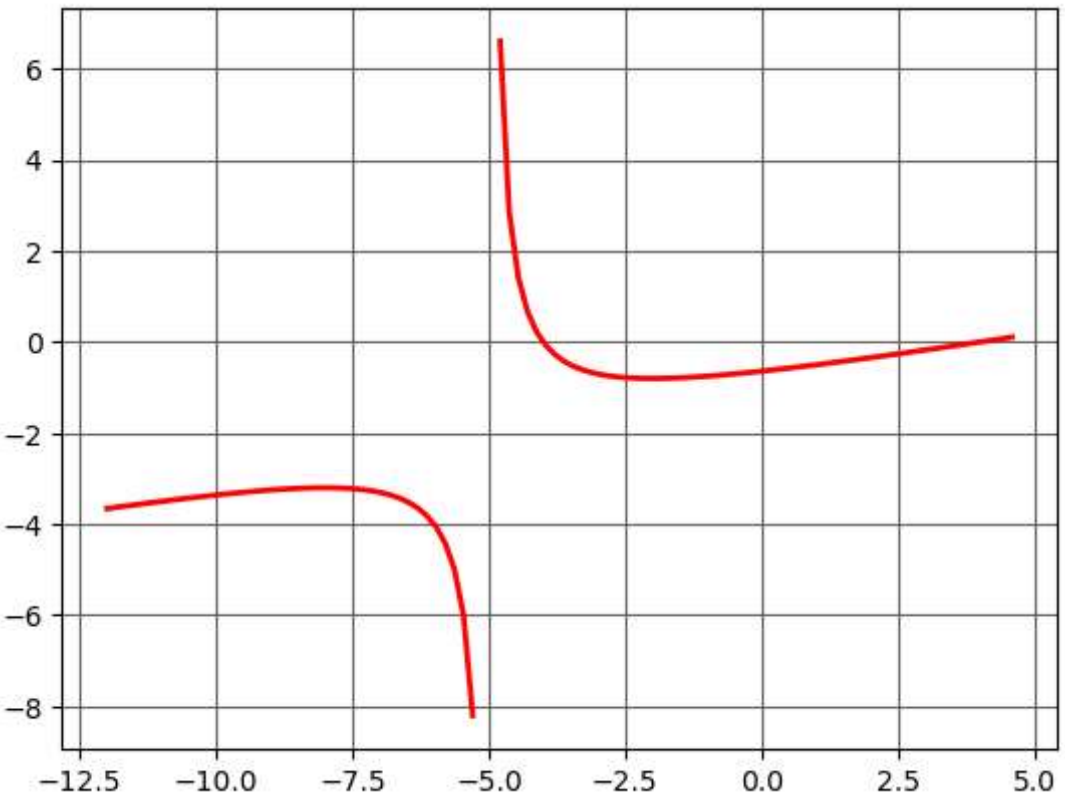
In [64]:

```
print('слева: %.1f справа: %.1f' % (y_2deriv.subs(x,0.9), y_2deriv.subs(x,1.1)))
```

```
слева: -12.3 справа: 16.6
```

In [65]:

```
# Пример 18
x = symbols('x')
y = (x**2-16)/(5*(x+5))
f = lambda x: (x**2-16)/(5*(x+5))
x = np.linspace(-12,4.6,100)
x[(x>-5.2) & (x < -4.8)] = np.nan
y = f(x)
plt.plot(x,y,lw=2,color='red')
plt.grid(True, linestyle='-', color='0.4')
plt.show()
```



In [66]:

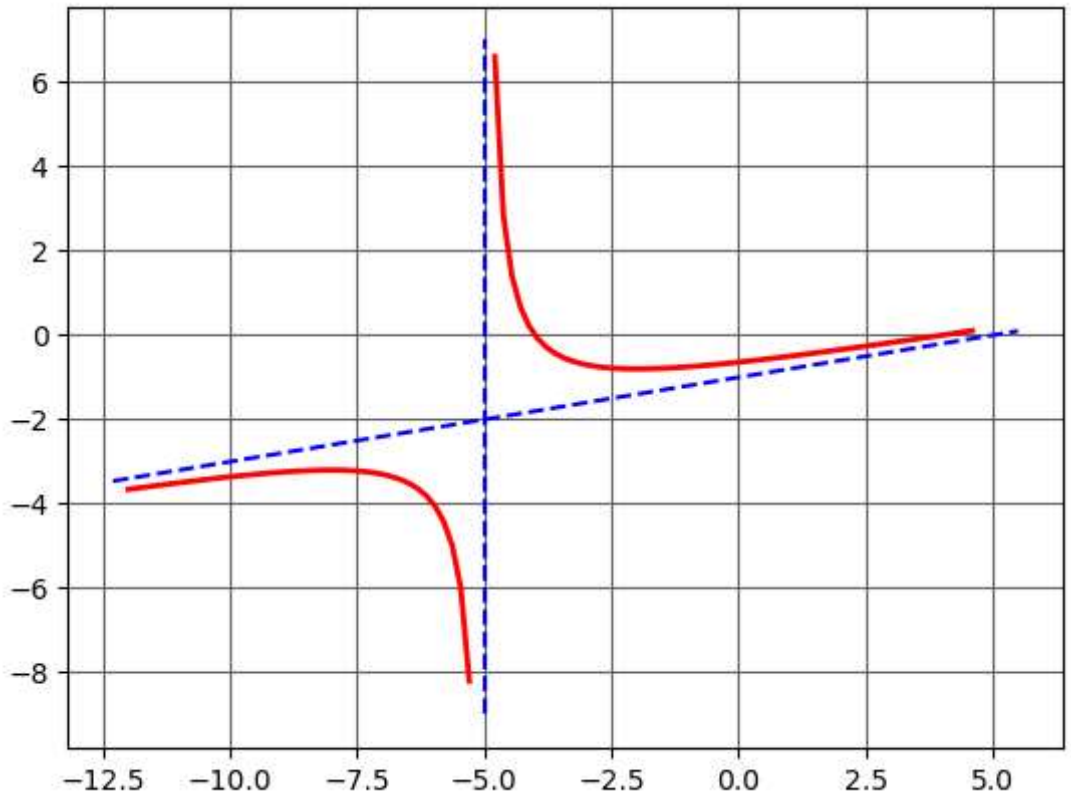
```
k = limit(y/x, x, oo)
k
```

Out[66]:

$$\left[0.304761904761905 \quad 0.306779839677906 \quad 0.308883012171189 \quad 0.3110772901\right.$$

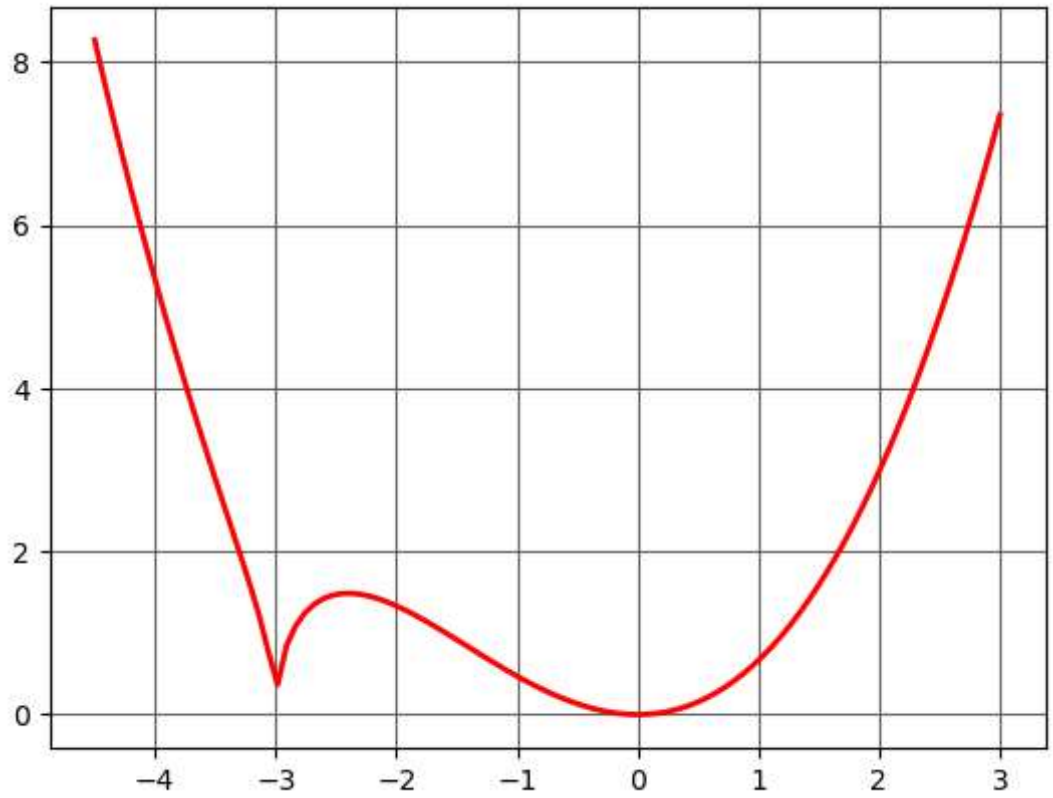
In [67]:

```
f = lambda x: (x**2-16)/(5*(x+5))
x = np.linspace(-12,4.6,100)
x[(x>-5.2) & (x < -4.8)] = np.nan
y = f(x)
plt.plot(x,y,lw=2,color='red')
x = np.linspace(-12.3,5.5,100)
y = x/5 - 1
plt.plot(x,y,'--',color='b')
plt.plot([-5,-5],[-9,7],'--',color='b')
plt.grid(True, linestyle='-', color='0.4')
plt.show()
```



In [68]:

```
# Пример 19
x = symbols('x')
y = x**2*sqrt(abs(x+3))/3
f = lambda x: x**2*np.sqrt(abs(x+3))/3
x = np.linspace(-4.5,3,100)
f_x = f(x)
plt.plot(x,f_x,lw=2,color='red')
plt.grid(True, linestyle='-', color='0.4')
plt.show()
```



In [69]:

```
y.subs(x,0)
```

Out[69]:

$$\frac{x^2\sqrt{|x+3|}}{3}$$

In [70]:

```
limit(y, x, oo)
```

Out[70]:

$$\frac{x^2\sqrt{|x+3|}}{3}$$

In [71]:

```
k = limit(y/x, x, oo)
k
```

Out[71]:

$$\left[-0.0740740740740741x^2\sqrt{|x+3|} \quad -0.0753424657534247x^2\sqrt{|x+3|} \quad -0.0766\right.$$

In [72]:

```
x = symbols('x')
y1 = x**2*sqrt(-x-3)/3
y1_ = diff(y1,x). simplify()
y1_
```

Out[72]:

$$\frac{x(-5x-12)}{6\sqrt{-x-3}}$$

In [73]:

```
y2 = x**2*sqrt(x+3)/3
y12_ = diff(y1,x,2).simplify()
y12_
```

Out[73]:

$$\frac{\sqrt{-x-3}\cdot\left(5x^2+24x+24\right)}{4\left(x^2+6x+9\right)}$$

In [74]:

```
y22_ = diff(y2,x,2).simplify()
y22_
```

Out[74]:

$$\frac{5x^2+24x+24}{4(x+3)^{\frac{3}{2}}}$$

In [75]:

```
xp1 = solve(y12_)
xp1
```

Out[75]:

```
[-12/5 - 2*sqrt(6)/5, -12/5 + 2*sqrt(6)/5]
```

In [76]:

```
diff(y1, x, 3).subs(x,xp1[0]).evalf(5)
```

Out[76]:

$$-10.465$$

In [77]:

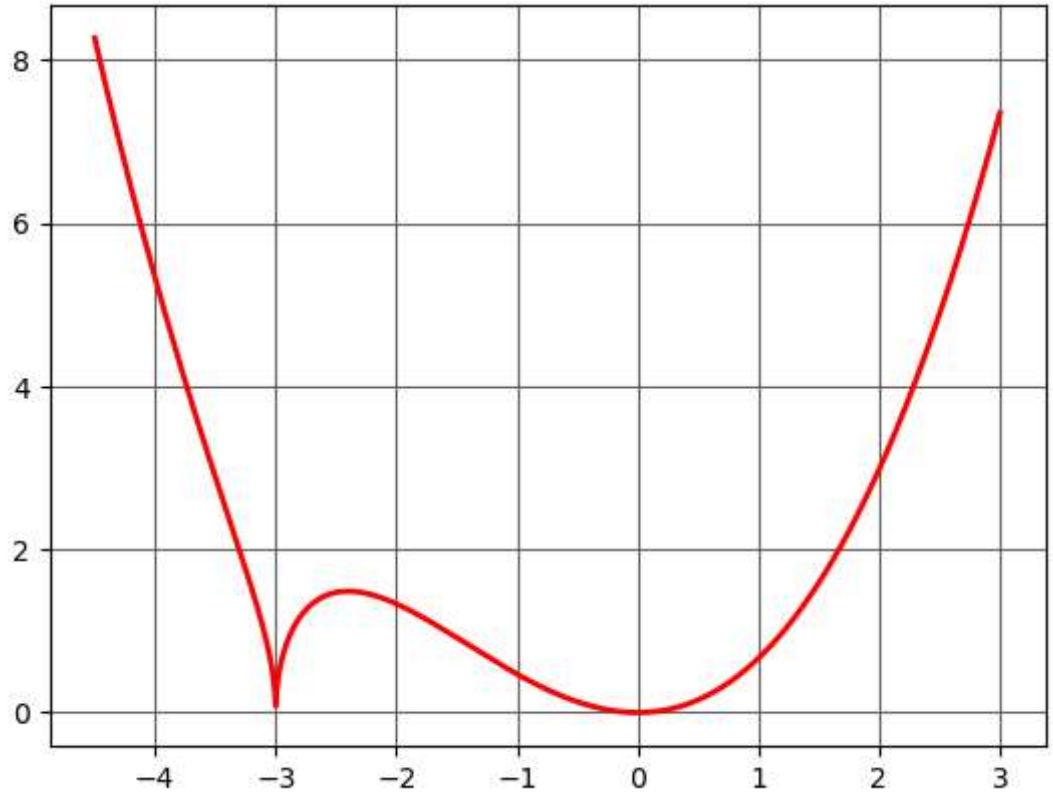
```
diff(y1, x, 3).subs(x,xp1[1]).evalf(4)
```

Out[77]:

$1.234i$

In [78]:

```
f = lambda x: x**2*np.sqrt(abs(x+3))/3
x = np.linspace(-4.5,3,2000)
f_x = f(x)
plt.plot(x,f_x,lw=2,color='red')
plt.grid(True, linestyle='-', color='0.4')
plt.show()
```



Частные производные. Функция diff(f, x, k, y, m, ...)

In [79]:

```
# Пример 20
x, y = symbols("x y")
z = x*y**2 + exp(-x)
```

In [80]:

```
diff(z, x, 2)
```

Out[80]:

e^{-x}

In [81]:

```
diff(z, y, 2)
```

Out[81]:

$2x$

In [82]:

```
# Пример 21
x, y = symbols('x y')
z = sin(x)*cos(y)
diff(z, x, 2, y)
```

Out[82]:

$\sin(y)\sin(x)$

Градиент

In [83]:

```
# Пример 22
x,y = symbols('x y')
z = 5*log(x**2 + y**2)
z_x = diff(z,x).subs({x:1, y:2})
z_y = diff(z,y).subs({x:1, y:2})
grad_f = (z_x, z_y)
grad_f
```

Out[83]:

```
(2, 4)
```

In [84]:

```
# Пример 23
x,y = symbols('x y')
z = x**2 + x*y +7
z_x = diff(z,x).subs({x:1, y:-1})
z_y = diff(z,y).subs({x:1, y:-1})
grad_f = (z_x, z_y)
grad_f
```

Out[84]:

```
(0, 0)
```

Производная по направлению

In [85]:

```
# Пример 24
l = Point(3,4)
l_n = l.distance(Point(0,0))
cos_a = l.x/l_n
cos_b = l.y/l_n
x,y = symbols('x y')
z = x**2 + y**2
z_x = diff(z,x).subs({x:1, y:1})
z_y = diff(z,y).subs({x:1, y:1})
z_l = z_x*cos_a + z_y*cos_b
z_l
```

Out[85]:

$$\frac{14}{5}$$

Касательная плоскость

In [86]:

```
# Пример 25
def tangent_plane(F,M):
    F_diff_x = diff(F,x).subs({x:M.x,y:M.y,z:M.z})
    F_diff_y = diff(F,y).subs({x:M.x,y:M.y,z:M.z})
    F_diff_z = diff(F,z).subs({x:M.x,y:M.y,z:M.z})

    n = Point(F_diff_x, F_diff_y, F_diff_z)

    p = Plane(M, normal_vector=n).equation()

    K = Point(M.x+n.x, M.y+n.y, M.z+n.z)
    l_n = Line(M, K).arbitrary_point()
    return p, ln
```

In [87]:

```
x, y, z = symbols('x y z')
F = x**2 + y**2 + z**2 - 9
M = Point(1,1,1)
p, l_n = tangent_plane(F,M)
```

In [88]:

```
p
```

Out[88]:

$$2x + 2y + 2z - 6$$

Экстремум функции многих переменных

In [89]:

```
# Пример 26
z = lambda w: (w[0]-1)**2 + (w[1]-3)**4
res = minimize(z, (0, 0))
res.x
```

Out[89]:

```
array([0.99999999, 2.98725136])
```

In [90]:

```
z((1,3)) < z((0.999,3.001))
```

Out[90]:

```
True
```

In [91]:

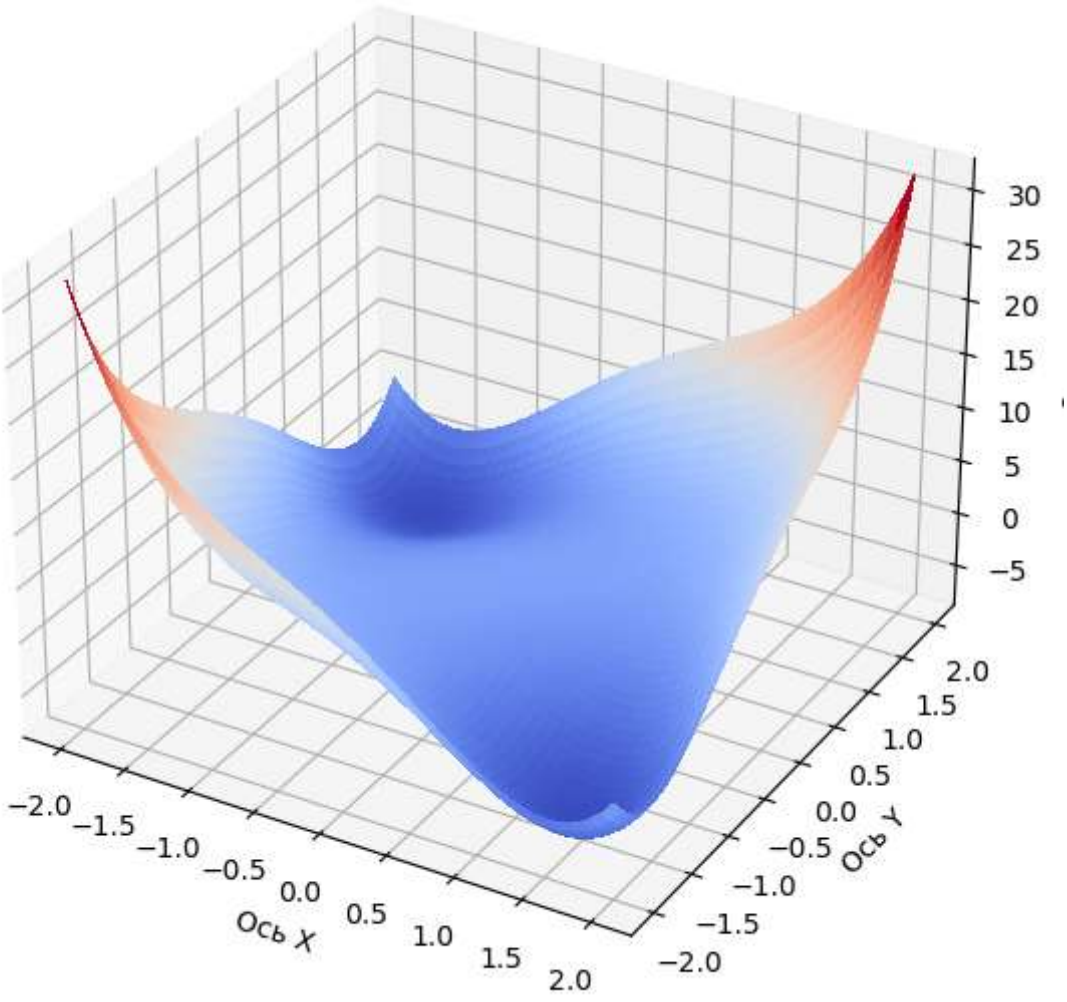
```
z((1,3))
```

Out[91]:

```
0
```

In [92]:

```
# Пример 27
z = lambda w: w[0]**4 + w[1]**4 - 2*w[0]**2 +4*w[0]*w[1] - 2*w[1]**2

fig = plt.figure(figsize=(7,7))
axes = fig.add_subplot(projection='3d')
y = x = np.linspace(-2, 2, 50)
x, y = np.meshgrid(x, y)
Z = z((x,y))
surf = axes.plot_surface(x, y, Z, cmap='coolwarm',linewidth=0, antialiased=False)

axes.set_xlabel('Ось X')
axes.set_ylabel('Ось Y')
axes.set_zlabel('Ось Z')
plt.show()
```



In [93]:

```
res = minimize(z, (1, -1))
res.x
```

Out[93]:

```
array([ 1.41421356, -1.41421356])
```

In [94]:

```
z(res.x)
```

Out[94]:

```
-8.0
```

In [95]:

```
res = minimize(z, (-1, 1))
res.x
```

Out[95]:

```
array([-1.41421357,  1.41421357])
```

In [96]:

```
z(res.x)
```

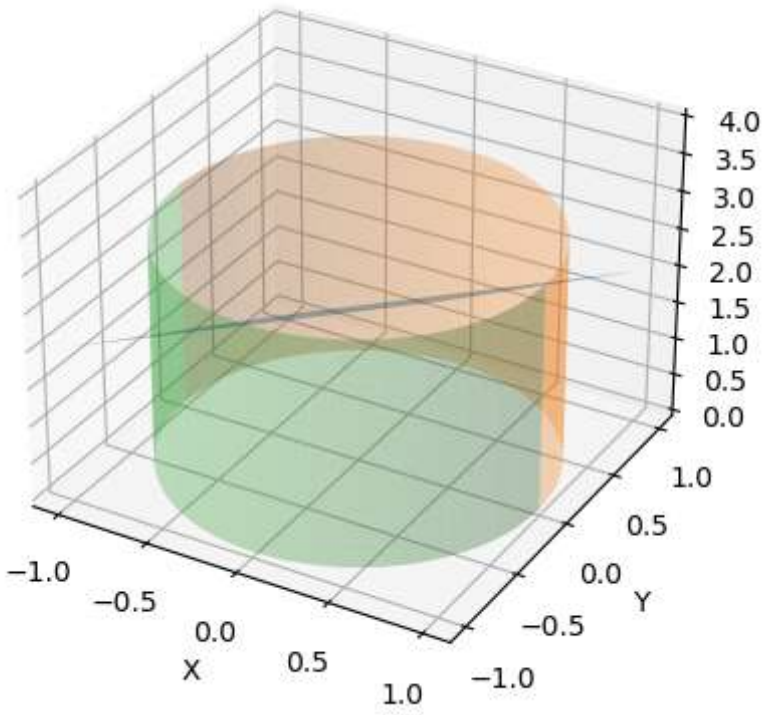
Out[96]:

```
-7.999999999999997
```

Поиск экстремума с использованием функции solve() библиотеки sympy

In [97]:

```
# Пример 28
f = lambda w: w[0] - w[1] + 2
fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')

x = np.linspace(-1, 1, 50)
y = np.linspace(-1, 1, 50)

x, y = np.meshgrid(x, y)
z1 = f((x,y))
ax.plot_surface(x, y, z1, alpha=0.4)

x = np.linspace(-1, 1, 100)
z = np.linspace(0, 3, 100)
xc, zc = np.meshgrid(x, z)
yc = np.sqrt(1-xc**2)
ax.plot_surface(xc, yc, zc, alpha=0.3)
ax.plot_surface(xc, -yc, zc, alpha=0.3)
ax.set_xlabel("X")
ax.set_ylabel("Y")
ax.set_zlabel("Z")
plt.show()
```



In [98]:

```
cons = ({'type': 'eq', 'fun': lambda w: w[0]**2 + w[1]**2 - 1})

bnds = ((None, None), (None, None))

res = minimize(f, (-0.5, 0.5), bounds=bnds, constraints=cons)

res.x
```

Out[98]:

```
array([-0.70710679,  0.70710677])
```

In [99]:

```
f_max = lambda w: -(1.5*w[0] - w[1] + 1)
cons = ({'type': 'eq', 'fun': lambda w: w[0]**2 + w[1]**2 - 1})
bnds = ((None, None), (None, None))
res = minimize(f_max, (0.5, -0.5), bounds=bnds,constraints=cons)

res.x
```

Out[99]:

```
array([ 0.83205051, -0.55469991])
```

Применения производной в экономике

In [100]:

```
# Пример 29
x,y = symbols('x y')
z = 4.5*x**(0.33) * y**(0.66)
z_x = diff(z, x)
z_y = diff(z, y)

E_x = (x/z)*z_x
E_y = (y/z)*z_y
print('E_x: %.2f E_y: %.2f' % (E_x, E_y))
```

```
E_x: 0.33 E_y: 0.66
```

In [101]:

```
# Пример 30
K,V0 = symbols('K V0')
V = V0*log(5+K**2)

Vprim2 = diff(V,K,2)

Vprim3 = diff(V,K,3)

s = solve(Vprim2,K)
s
```

Out[101]:

```
[-sqrt(5), sqrt(5)]
```

In [102]:

```
Vprim3.subs(K,s[1])
```

Out[102]:

$$-\frac{\sqrt{5}V_0}{25}$$

Индивидуальное задание

Автомобиль движется по криволинейной дороге, заданной уравнением $y = f(x)$, где $f(x)$ — дифференцируемая функция. В определенной точке кривой автомобиль имеет ускорение $10 м/с^2$ в направлении положительной оси x и ускорение $6 м/с^2$ в направлении положительной оси y. Скорость автомобиля в этот момент равна $20 м/с$. Каков радиус кривизны кривой в этой точке? Визуализируйте на графике этот радиус.

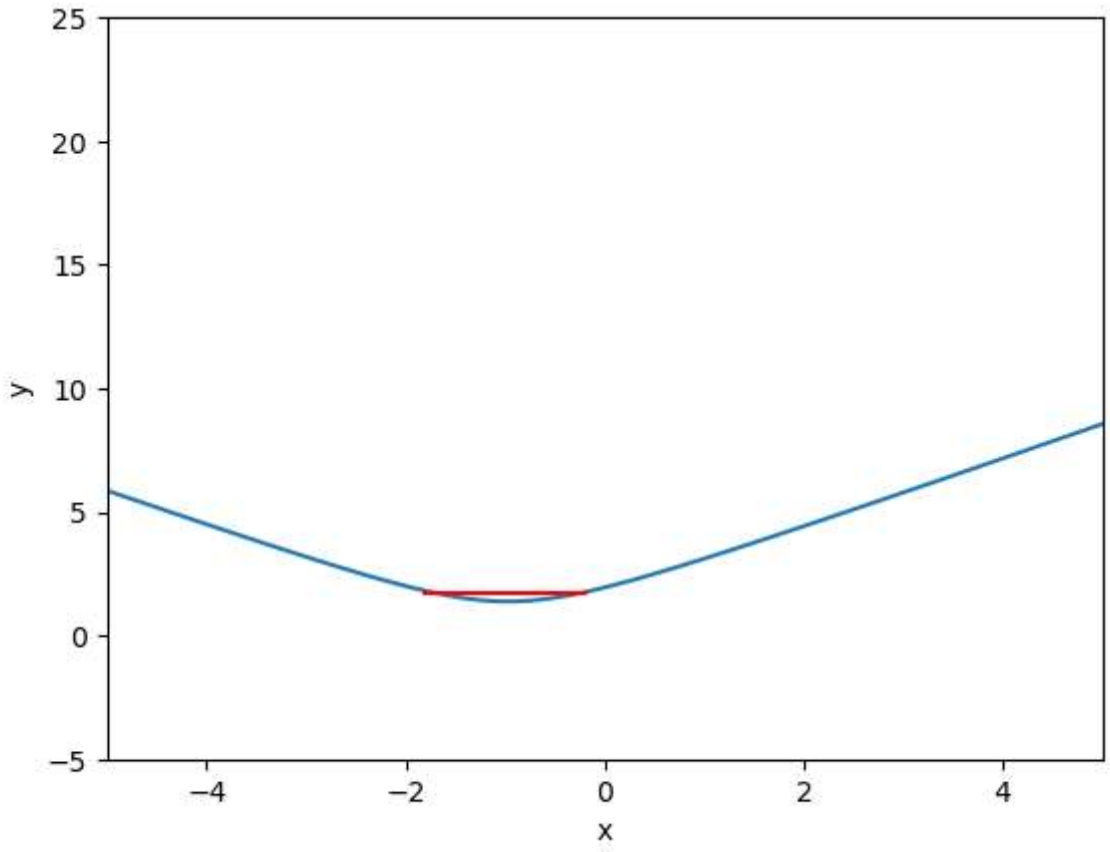
In [16]:

```
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline

x, y = symbols('x y')
f = Function('f')(x)

v_x = f.diff(x)
v_y = f.diff(y)

a_x = vx.diff(x)
a_y = vy.diff(y)

ax_val = 10
ay_val = 6

a = sqrt(ax_val**2 + ay_val)

v_val = 20
R = float(v_val / a)

print('Радиус кривизны в данной точке равен:', R)
```

```
Радиус кривизны в данной точке равен: 1.9425717247145284
```

In [34]:

```
x = np.linspace(-5, 5, 100)

y = []

for elem in x:
    y.append(sqrt(np.power(elem, 2) + np.power(elem + R, 2)))


# define point of interest
x0 = -1.8
y0 = 1.7

# define radius endpoints
x1 = -0.2
y1 = 1.7

fig, ax = plt.subplots()
ax.plot(x, y)

ax.plot([x0, x1], [y0, y1], color='r')

ax.set_xlim([-5, 5])
ax.set_ylim([-5, 25])
ax.set_xlabel('x')
ax.set_ylabel('y')

plt.show()
```



In [ ]:

```

```