Производная

In [1]:
```
from sympy import *
import numpy as np
from scipy.optimize import minimize
from scipy.stats import norm
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
import warnings
warnings.filterwarnings('ignore')
%matplotlib inline
```

In [2]:
```
def tangent(y, x0):
    y0 = y.subs(x, x0)
    x1 = x0 + 1
    k = diff(y,x).subs(x,x0)
    y1 = y0 + k
    return Line((x0,y0), (x1,y1))
```

Пример 1

Производная функции $y = x\cos x$

In [3]:
```
x = symbols('x')
y = x*cos(x)
diff(x*cos(x), x)
```

Out[3]: $-x\sin(x) + \cos(x)$

Пример 2

Производная 3-го порядка функции $y = \ln x$.

In [4]:
```
diff(log(x), x, 3)
```

Out[4]: $\frac{2}{x^3}$

Пример 3

Найти значение $y''(10)$ для функции $y = \lg^3(x^3)$.

In [5]:
```
y = log(x**3,10)**3
diff(y,x,2).subs(x,10)
```

Out[5]: $-\frac{9(-6 + \log(1000))\log(1000)}{100\log(10)^3}$

In [6]:
```
diff(y,x,2).subs(x,10).simplify()
```

Out[6]: $$\frac{81\left(2-\log\left(10\right)\right)}{100\log\left(10\right)^{2}}$$

Пример 4

Решить уравнение $y'=0$, где $y(x)=\frac{x^2+x-6}{x^2-10x+25}$

In [7]:
```
y = (x**2+x-6)/(x**2-10*x+25)
z = diff(y,x)
z
```

Out[7]: $$\frac{9\log\left(x^{3}\right)^{2}}{x\log\left(10\right)^{3}}$$

In [8]:
```
solve(z, x)
```

Out[8]:
```
[1, -1/2 - sqrt(3)*I/2, -1/2 + sqrt(3)*I/2]
```

Пример 5

Найти первую производную $\frac{dy}{dx}$ и вторую производную $\frac{d^2y}{dx^2}$ для функции, заданной неявно в виде уравнения: $x^2+y^2=4$.

In [9]:
```
x = symbols('x')
y = symbols('y')
f = x**2 + y**2 - 4
idiff(f, y, x)
```

Out[9]: $$-\frac{x}{y}$$

In [10]:
```
f = x**2 + y**2 - 4
idiff(f, y, x, 2)
```

Out[10]: $$\frac{-\frac{x^{2}}{y}-y}{y^{2}}$$

In [11]:
```
idiff(f, y, x, 2).simplify()
```

Out[11]: $$-\frac{x^{2}+y^{2}}{y^{3}}$$

Пример 6

Найти y'_x и y''_{xx} для функции, заданной в параметрической форме:

$$\begin{cases} x=t-\sin t \\ y=1-\cos t \end{cases}$$

In [12]:
```
t = symbols('t')
x = t - sin(t)
y = 1 - cos(t)
y_diff = diff(y,t)/diff(x,t)
y_diff
```

Out[12]:

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

In [13]:
```
y_2diff = diff(y_diff,t)/diff(x,t)
y_2diff.simplify()
```

Out[13]:

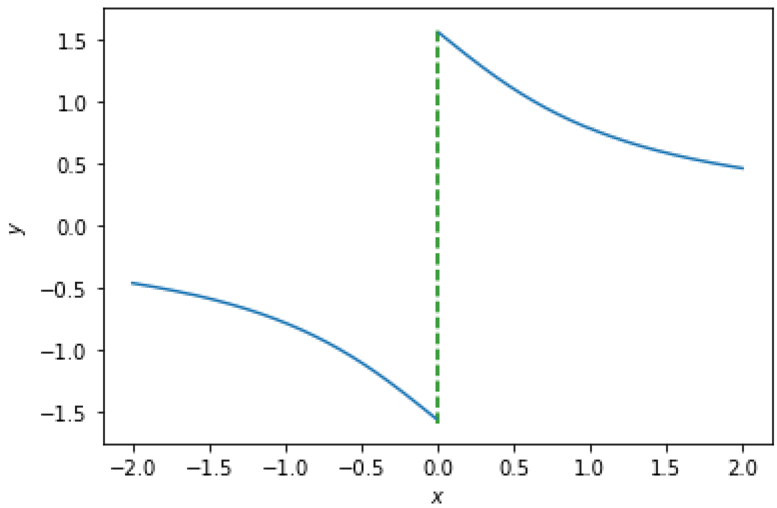
$$-\frac{1}{(\cos(t)-1)^2}$$

Пример 7

Найти левую и правую производные функции $f(x)$ в точке ее разрыва, если

$$f(x)=\begin{cases}\operatorname{arctg}\left(\frac{1}{x}\right), x\neq 0\\ -\frac{\pi}{2}, x=0\end{cases}$$

In [14]:
```
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.xlabel('$x$')
plt.ylabel('$y$')
plt.vlines(0, -1.6, 1.6, color='g', linestyles='dashed')
plt.show()
```



In [15]:
```
x = symbols('x')
y = atan(1/x)
z = diff(y,x)
limit(z, x, 0, dir='+')
```

Out[15]: -1

In [16]:
```
limit(z, x, 0, dir='-')
```

Out[16]: -1

Пример 8

Провести касательную к графику функции

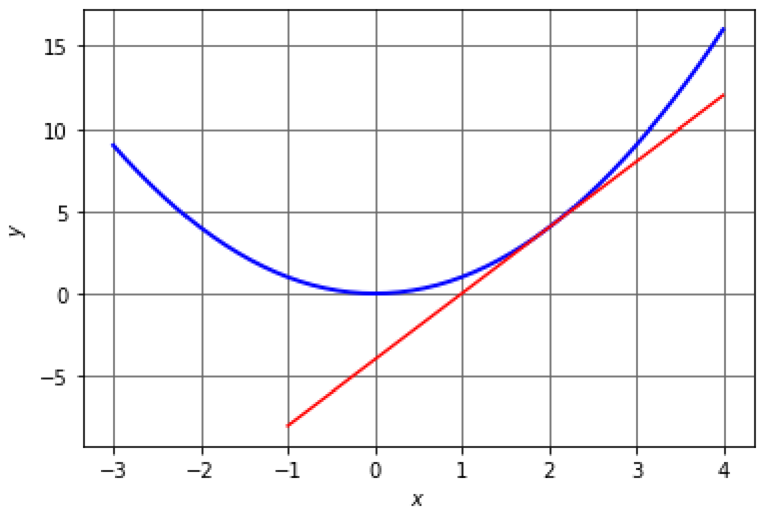
$$y = x^2$$

в точке с абсциссой

$$x_0 = 2$$

.

In [17]:
```
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()
```

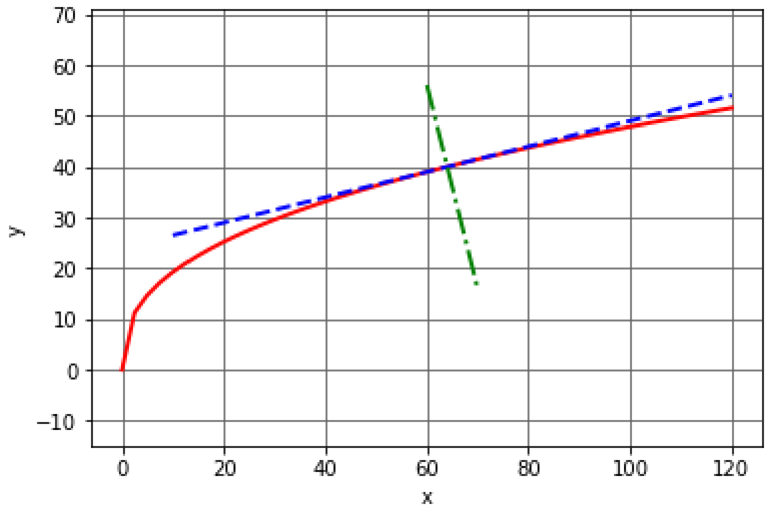


Пример 9

Найти уравнение касательной и нормали к графику функции $y = 6\sqrt[3]{x} + 2\sqrt{x}$ в точке с абсциссой $x_0 = 64$.

In [18]:
```
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()
```

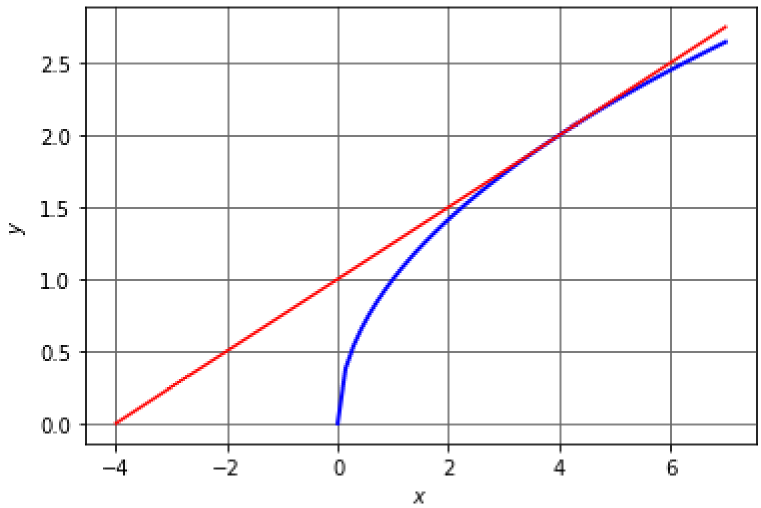


Пример 10

Из точки A(—4;0) провести касательную к кривой $y = \sqrt{x}$

In [19]:

```
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()
```

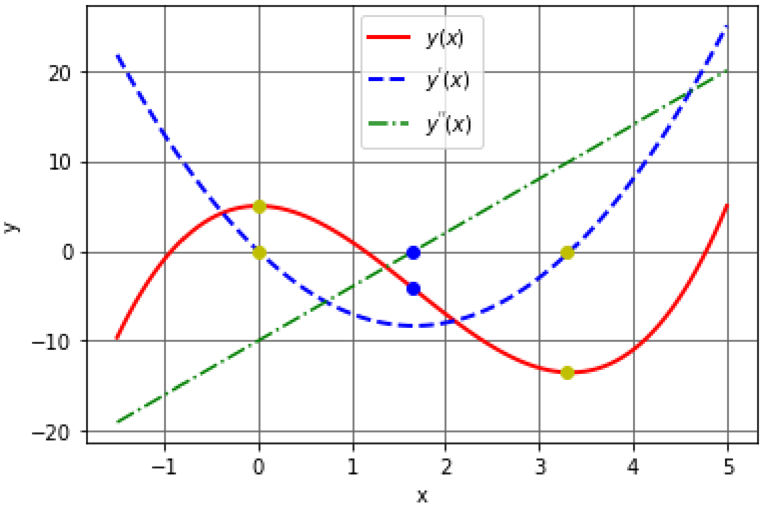


Исследование функции

In [20]:

```
t = np.linspace(-1.5, 5, 100)
f = t**3 - 5*t**2 + 5
fd = 3*t**2 - 10*t
```

```
fdd = 6*t - 10
plt.plot(t,f,lw=2,color='red',label = "$y(x)$")
plt.plot(t,fd,'--',lw=2,color='b',label = "$y^{'}(x)$")
plt.plot(t,fdd, '-.', color='g',label = "$y^{''}(x)$")
plt.plot([0], [0], 'o', color='y')
plt.plot([0], [5], 'o', color='y')
plt.plot([3.3], [0], 'o', color='y')
plt.plot([3.3], [-13.4], 'o', color='y')
plt.plot([1.65], [0], 'o', color='b')
plt.plot([1.65], [-4], 'o', color='b')
plt.grid(True, linestyle='-', color='0.4')
plt.xlabel('x')
plt.ylabel('y')
plt.legend()
plt.show()
```



Пример 11

Решить неравенство $x^2 < 3$.

In [21]:
```
x, y = symbols('x y')
solve(x**2 < 3)
```

Out[21]: $-\sqrt{3} < x \wedge x < \sqrt{3}$

Пример 12

Решить уравнение $x^2 — y^2 = 0$ относительно переменной x.

In [22]:
```
solve(x**2 - y**2, x)
```

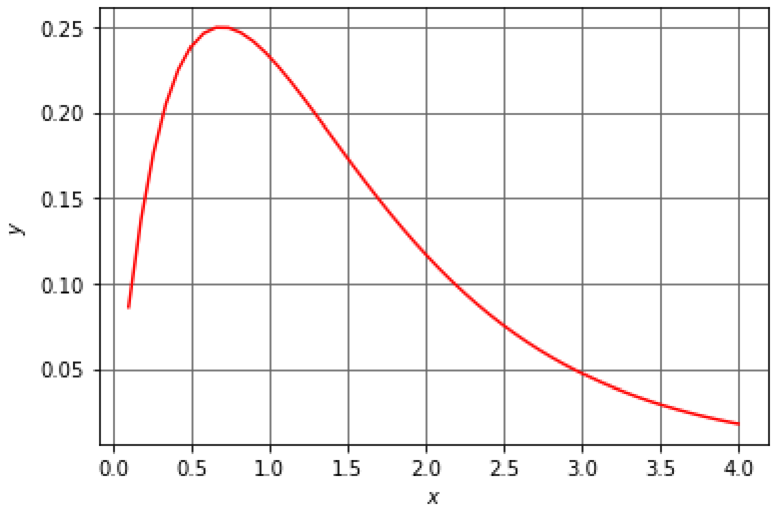
Out[22]:
```
[-y, y]
```

Пример 13

Исследовать на экстремум функцию $y = e^{-x} - e^{-2x}$.

In [23]:
```
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 [24]:
```
f_max = lambda x: -(np.exp(-x) - np.exp(-2*x))
res = minimize(f_max, -2)
res
```

Out[24]:
```
      fun: -0.24999999999945666
 hess_inv: array([[1.98553383]])
      jac: array([-7.26431608e-07])
  message: 'Optimization terminated successfully.'
     nfev: 26
      nit: 12
     njev: 13
   status: 0
  success: True
        x: array([0.69314571])
```

Пример 14

Исследовать на экстремум функцию $y = x^3$.

In [25]:
```
x = symbols('x')
y = x**3
x0 = solve(diff(y,x))[0]
print(f'x0: {x0} y(x0): {y.subs(x, x0)}')
```

```
x0: 0 y(x0): 0
```

In [26]:
```
diff(y,x,2).subs(x,x0)
```

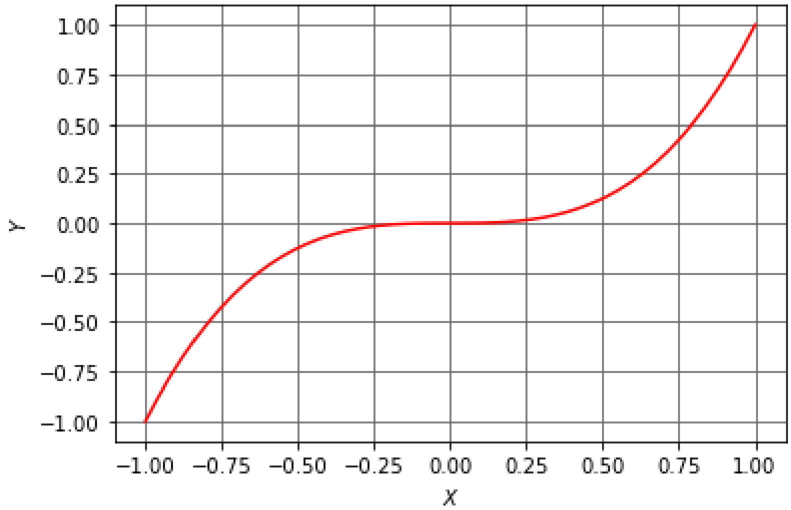
Out[26]: 0

In [27]:
```
diff(y,x,3).subs(x,x0)
```

Out[27]: $\frac{246}{3125}$

In [28]:
```
x = np.linspace(-1,1,50)
plt.plot(x, x**3, 'r')
plt.xlabel('$X$')
plt.ylabel('$Y$')
```

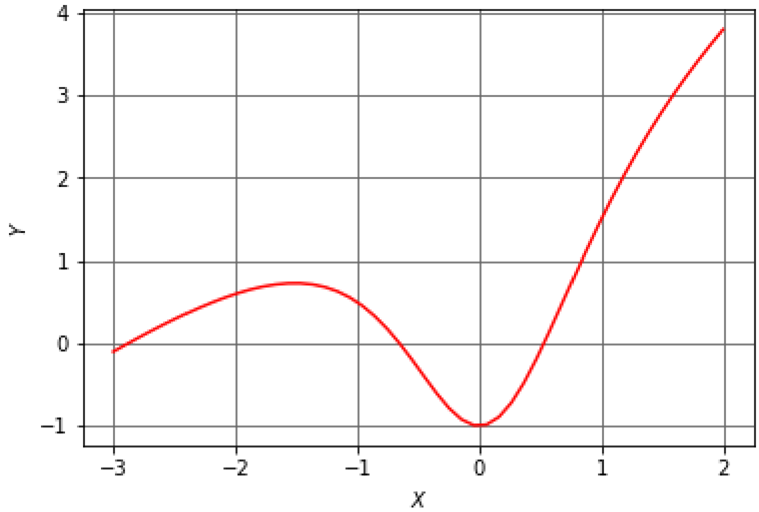
```
plt.grid(True, linestyle='-', color='0.4')
plt.show()
```



Пример 15

Найти экстремум функции $y = \frac{x^3+3x^2-1}{x^2+1}$.

In [29]:
```
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 [30]:
```
res = minimize(f, 1)
print(f'x_min: {res.x[0]:.4f} f_min: {f(res.x)[0]:.4f}')
```

```
x_min: 0.0000 f_min: -1.0000
```

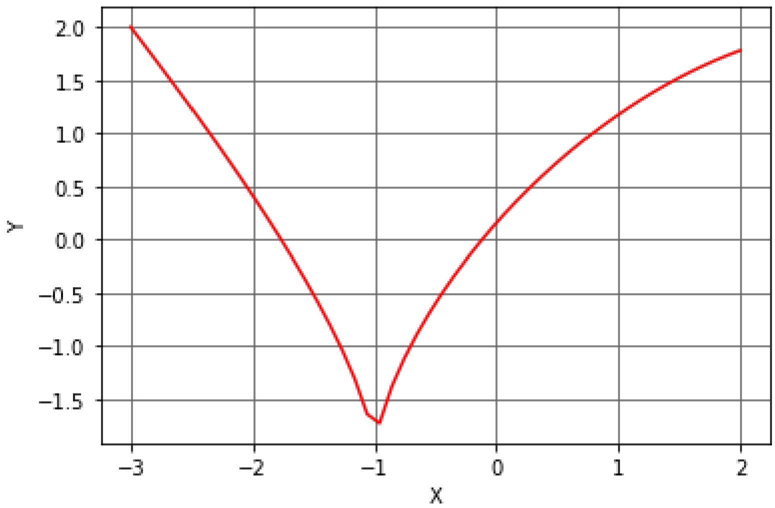
In [31]:
```
f_max = lambda x: -(x**3+3*x**2-1) / (x**2+1)
res = minimize(f_max, -2)
print(f'x_max: {res.x[0]:.4f} f_max: {f(res.x)[0]:.4f}')
```

```
x_max: -1.5127 f_max: 0.7309
```

Пример 16

Найти наибольшее и наименьшее значения функции $f(x)$ на отрезке:
$f(x) = \sqrt[3]{2(x+1)^2(5-x)} - 2; x \in [-3; 3]$

In [32]:
```
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 [33]:
```
res = minimize(fun, -1.5)
print(f'x_min: {res.x[0]:.3f}')
```

```
x_min: -0.490
```

In [34]:
```
res = minimize(fun, -1.001)
print(f'x_min: {res.x[0]:.3f}')
```

```
x_min: -1.001
```

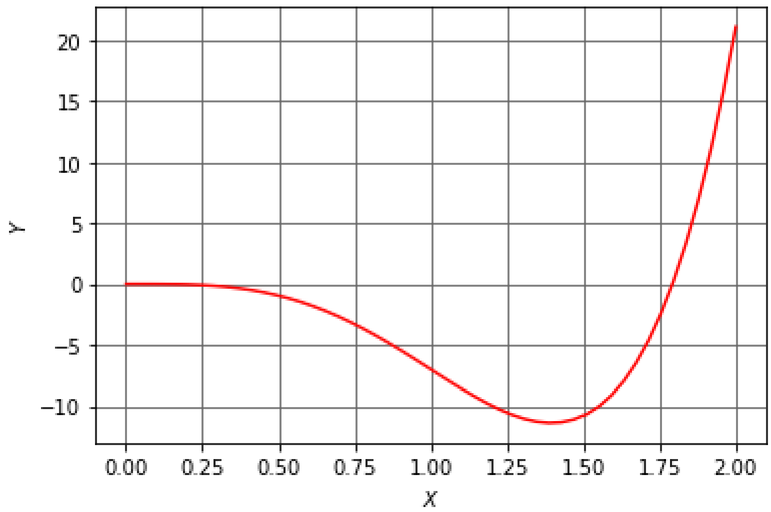
In [35]:
```
print(f'y(-3): {fun(-3):.3f} y(3): {fun(-3):.3f}')
```

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

Пример 17

Найти точки перегиба и исследовать характер выпуклости кривой $y = x^4(12\ln x - 7)$.

In [36]:
```
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 [37]:
```
x = symbols('x')
y = x**4 * (12*log(x) - 7)
y_2deriv = diff(y,x,2)
y_2deriv
```

Out[37]: $144x^2 \log(x)$

In [38]:
```
x_inflex = solve(y_2deriv, x)
x_inflex
```

Out[38]:
```
[0, 1]
```

In [39]:
```
diff(y,x,3).subs(x, 1)
```

Out[39]: 144

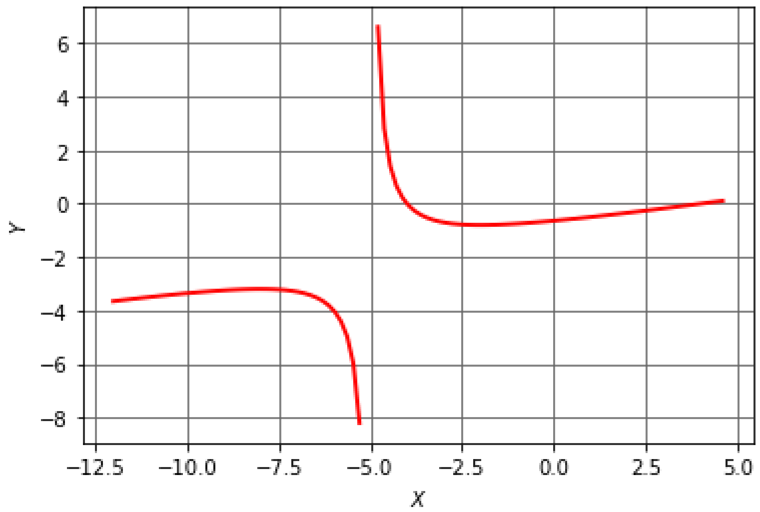
In [40]:
```
print(f'Слева: {y_2deriv.subs(x,0.9):.3f} Справа: {y_2deriv.subs(x,1.1):.3f}')
```

```
Слева: -12.289 Справа: 16.607
```

Пример 18

Провести полное исследование функции $y = \frac{x^2-16}{5(x+5)}$.

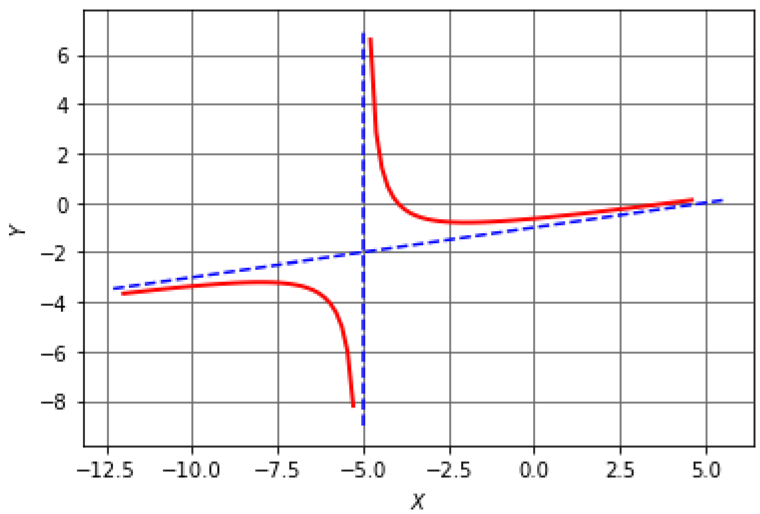
In [41]:
```
x = symbols('x')
y = (x**2-16)/(5*(x+5))
f = lambda x: (x**2-16)/(5*(x+5))
x = np.linspace(-12,4.6,100)
x[(x>-5.2) & (x < -4.8)] = np.nan
y = f(x)
plt.plot(x,y,lw=2,color='red')
plt.xlabel('$X$')
plt.ylabel('$Y$')
plt.grid(True, linestyle='-', color='0.4')
plt.show()
```



In [42]:
```
k = limit(y/x, x, oo)
k
```

Out[42]: [.304761904761905 0.306779839677906 0.308883012171189 0.311077290143904 (

In [43]:
```
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.xlabel('$X$')
plt.ylabel('$Y$')
plt.grid(True, linestyle='-', color='0.4')
plt.show()
```

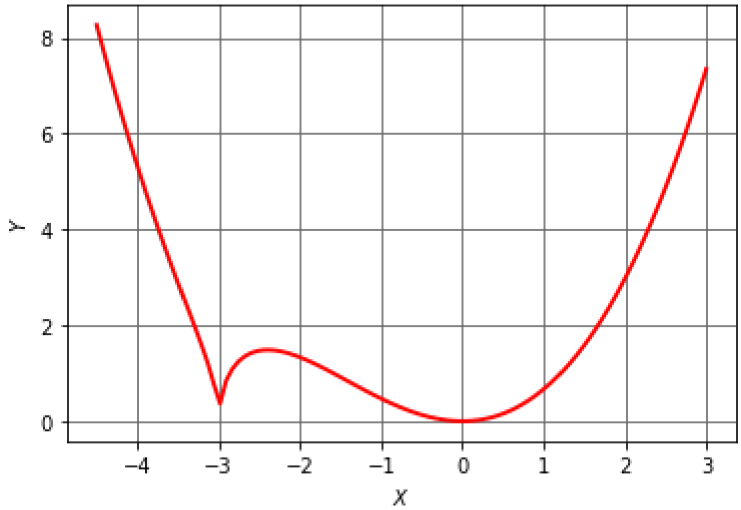


Пример 19

Провести полное исследование функции$y = \frac{1}{3}x^2\sqrt{|x+3|}$.

In [44]:
```
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.xlabel('$X$')
plt.ylabel('$Y$')
plt.grid(True, linestyle='-', color='0.4')
plt.show()
```



In [45]:
```
y.subs(x,0)
```

Out[45]:

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

In [46]:
```
limit(y, x, oo)
```

Out[46]:

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

In [47]:
```
k = limit(y/x, x, oo)
k
```

Out[47]:

$$\left[0.0740740740740741x^2\sqrt{|x+3|} \quad -0.0753424657534247x^2\sqrt{|x+3|} \quad -0.07665505226\right.$$

In [48]:
```
x = symbols('x')
y1 = x**2*sqrt(-x-3)/3
y1_ = diff(y1,x). simplify()
y1_
```

Out[48]:

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

In [49]:
```
y2 = x**2*sqrt(x+3)/3
y12_ = diff(y1,x,2).simplify()
y12_
```

Out[49]:

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

In [50]:
```
y22_ = diff(y2,x,2).simplify()
y22_
```

Out[50]:

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

In [51]:
```
xp1 = solve(y12_)
xp1
```

Out[51]:
```
[-12/5 - 2*sqrt(6)/5, -12/5 + 2*sqrt(6)/5]
```

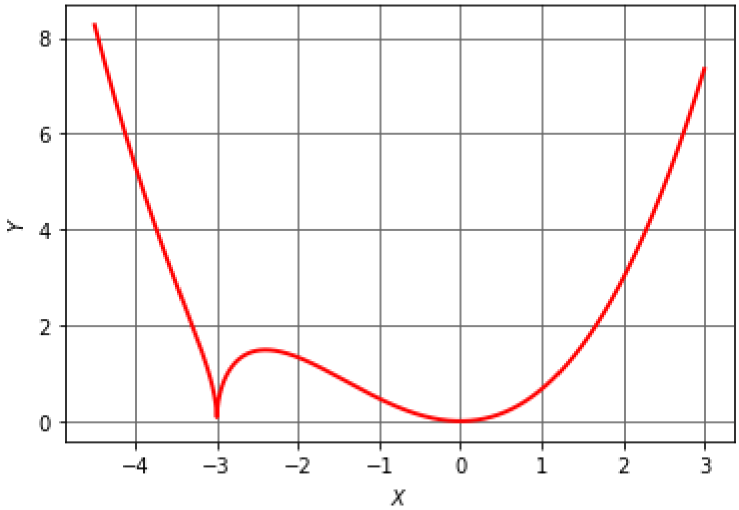
In [52]:
```
diff(y1, x, 3).subs(x,xp1[0]).evalf(5)
```

Out[52]: -10.465

In [53]:
```
diff(y1, x, 3).subs(x,xp1[1]).evalf(4)
```

Out[53]: $1.234i$

In [54]:
```
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.xlabel('$X$')
plt.ylabel('$Y$')
plt.grid(True, linestyle='-', color='0.4')
plt.show()
```



Пример 20

Вычислить частные производные $\frac{\partial^2 z}{\partial x^2}$ и $\frac{\partial^2 z}{\partial y^2}$ функции $z = xy^2 + e^{-x}$

In [55]:
```
x, y = symbols("x y")
z = x*y**2 + exp(-x)
```

In [56]:
```
diff(z, x, 2)
```

Out[56]: e^{-x}

In [57]:
```
diff(z, y, 2)
```
Out[57]: $2x$

Пример 21

Вычислить частную производную $\frac{\partial^3 z}{\partial x^2 \partial y}$ функции $z = \sin x \cdot \cos y$

In [58]:
```
x, y = symbols('x y')
z = sin(x)*cos(y)
diff(z, x, 2, y)
```
Out[58]: $\sin(y)\sin(x)$

Пример 22

Вычислить градиент функции $z = 5\ln(x^2 + y^2)$

In [59]:
```
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[59]:
```
(2, 4)
```

Пример 23

Определить направление l быстрейшего возрастания функции $z = x^2 + xy + 7$ в точке $M(1; -1)$.

In [60]:
```
x,x = 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[60]:
```
(1, 1)
```

Пример 24

Найти производную функции $z = x^2 + y^2$ в точке $M(1; 1)$ по направлению вектора $\boldsymbol{l} = 3\boldsymbol{i} + 4\boldsymbol{j}$.

In [61]:
```
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[61]: $\frac{14}{5}$

Пример 25

Провести касательную плоскость и нормаль к сфере $x^2 + y^2 + z^2 = 3$ в точке $M(1;1;1)$.

In [62]:
```
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, In
```

In [63]:
```
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 [64]:
```
p
```

Out[64]: $2x + 2y + 2z - 6$

Пример 26

Найти минимум функции двух переменных $z = (x-1)^2 + (y-3)^4$.

In [65]:
```
z = lambda w: (w[0]-1)**2 + (w[1]-3)**4
res = minimize(z, (0, 0))
res.x
```

Out[65]:
```
array([0.99999999, 2.98725136])
```

In [66]:
```
z((1,3)) < z((0.999,3.001))
```

Out[66]:
```
True
```

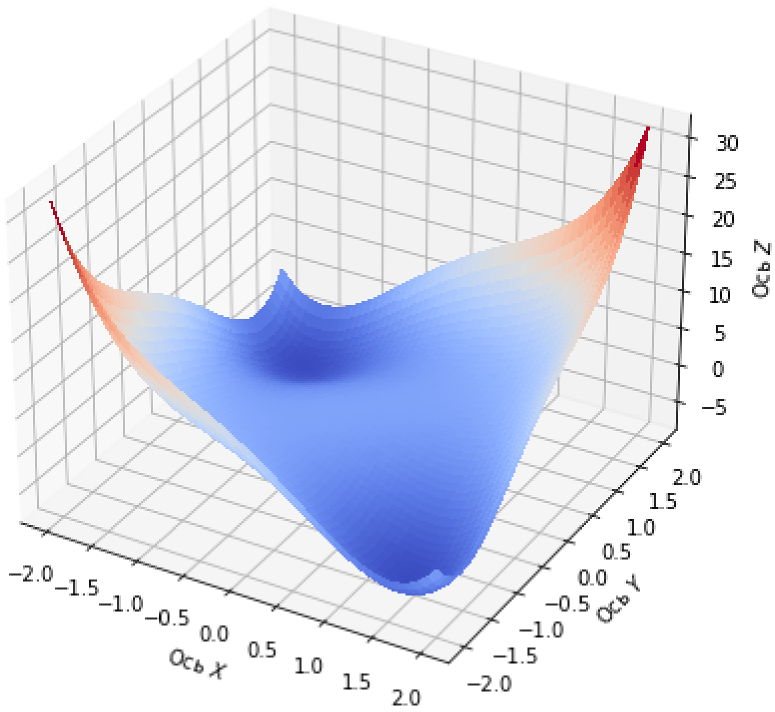
In [67]:
```
z((1,3))
```

Out[67]:
```
0
```

Пример 27

Исследовать функцию на экстремум: $z = x^4 + y^4 - 2x^2 + 4xy - 2y^2$

In [68]:
```
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.gca(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 [69]:
```
res = minimize(z, (1, -1))
res.x
```

Out[69]:
```
array([ 1.41421356, -1.41421356])
```

In [70]:
```
z(res.x)
```

Out[70]:
```
-8.0
```

In [71]:
```
res = minimize(z, (-1, 1))
res.x
```

Out[71]:
```
array([-1.41421356,  1.41421356])
```

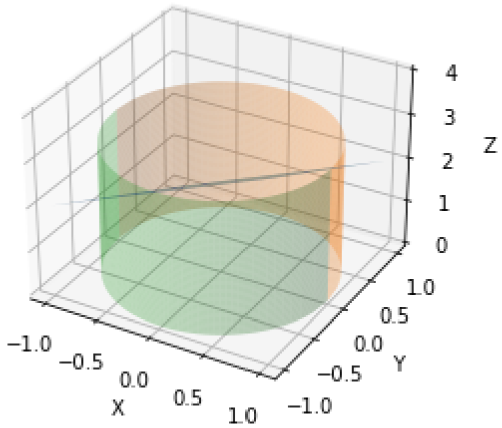
In [72]:
```
z(res.x)
```

Out[72]:
```
-8.0
```

Пример 28

Найти экстремумы функции $z = x - y + 2$ при ограничении: $x^2 + y^2 = 1$.

In [73]:
```
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 [74]:
```
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[74]:
```
array([-0.70710679,  0.70710677])
```

In [75]:
```
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[75]:
```
array([ 0.83205051, -0.55469991])
```

Пример 29

Определить коэффициенты эластичности производственной функции Кобба-Дугласа $z = 4,5x^{0,33}y^{0,66}$.

In [76]:
```
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(f'E_x: {E_x:.3f} E_y: {E_y:.3f}')
```

```
E_x: 0.330 E_y: 0.660
```

Пример 30

Зависимость объема выпуска продукции V от капитальных затрат K определяется функцией $V = V_0 \ln(5 + K^2)$. Найти интервал изменения K, на котором увеличение капитальных затрат неэффективно.

In [77]:
```
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[77]:
```
[-sqrt(5), sqrt(5)]
```

In [78]:
```
Vprim3.subs(K,s[1])
```

Out[78]:

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

Примеры решения задач

Вычислить y' для функции $x\cos\left(\log(x)\right) + \sin\left(\log(x)\right)$

In [79]:
```
x = symbols('x')
y = x*(cos(log(x))+sin(log(x)))
diff(y,x)
```

Out[79]:

$$\mathrm{x}\left(-\frac{\sin\left(\log\left(\mathrm{x}\right)\right)}{\mathrm{x}} + \frac{\cos\left(\log\left(\mathrm{x}\right)\right)}{\mathrm{x}}\right) + \sin\left(\log\left(\mathrm{x}\right)\right) + \cos\left(\log\left(\mathrm{x}\right)\right)$$

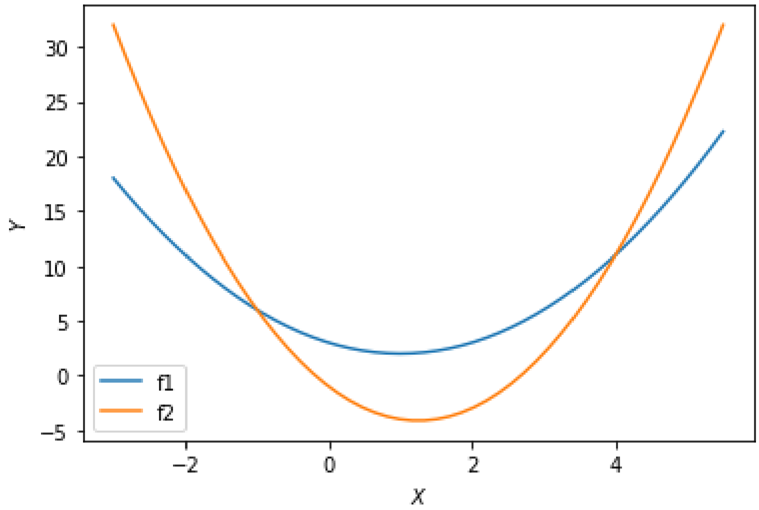
In [80]:
```
diff(y,x).simplify()
```

Out[80]:

$$2\cos\left(\log\left(\mathrm{x}\right)\right)$$

Решить уравнение $y'(x) = 0$, где $y(x) = max\{x^2 - 2x + 3; 2x^2 - 5x - 1\}$

In [81]:
```
f1 = lambda x: x**2-2*x+3
f2 = lambda x: 2*x**2-5*x-1
x = np.linspace(-3, 5.5, 50)
y1 = f1(x)
plt.plot(x,y1, label = "f1")
y2 = f2(x)
plt.plot(x,y2, label = "f2")
plt.xlabel('$X$')
plt.ylabel('$Y$')
plt.legend()
plt.show()
```



In [82]:
```
f1(-1) == f2(-1)
```

Out[82]: True

In [83]:
```
f1(4) == f2(4)
```

Out[83]: True

In [84]:
```
x = symbols('x')
f1 = x**2-2*x+3
f2 = 2*x**2-5*x-1

y_diff1 = diff(f2,x)
y_diff1
```

Out[84]: $4x - 5$

In [85]:
```
y_diff2 = diff(f1,x)
y_diff2
```

Out[85]: $2x - 2$

Показать, что функция $y = x\sin\ x$ удовлетворяет уравнению $\frac{y'}{\cos\ x} - x = \tan x$

In [86]:
```
x, y = symbols('x y')
y = x*sin(x)
yprim = diff(y, x)
f = yprim/cos(x) - x
f.simplify()
```

Out[86]: $\tan(x)$

Написать уравнения касательных к графику функции $y = (x^2 + 1)(x - 2)$ в точках её пересечения с осями координат.

```
In [87]: x = symbols('x', real=True)
         y = (x**2 + 1)*(x - 2)
```

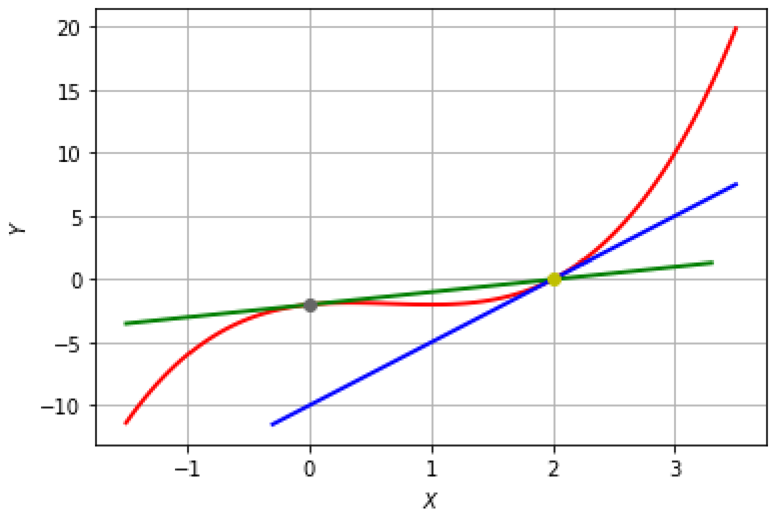
```
In [88]: tangent(y, 0).equation()
```

Out[88]: $-x + y + 2$

```
In [89]: xp = solve(y, x)
         tangent(y, xp[0]).equation()
```

Out[89]: $-5x + y + 10$

```
In [90]: x = np.linspace(-1.5,3.5,100)
         y = (x**2 + 1)*(x - 2)
         plt.plot(x, y, lw=2, color='red')
         x = np.linspace(-1.5,3.3,100)
         y1 = x - 2
         plt.plot(x, y1, lw=2, color='green')

         x = np.linspace(-0.3,3.5,100)
         y2 = 5*x - 10
         plt.plot(x, y2, lw=2, color='blue')
         plt.plot([0], [-2], 'o', color='0.4')
         plt.plot([2], [0], 'o', color='y')
         plt.xlabel('$X$')
         plt.ylabel('$Y$')
         plt.grid(True)
         plt.show()
```



При каком значении параметра a парабола $y = ax^2$ касается кривой $y = lnx$?

```
In [91]: x, a, x0 = symbols('x a x0')

         y1 = a*x**2
         y2 = log(x)

         y1_diff = diff(y1,x).subs(x,x0)
```

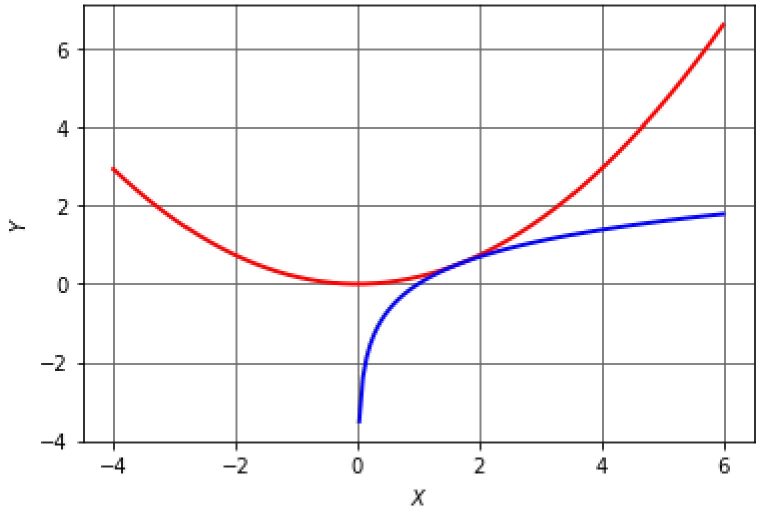
```
y2_diff = diff(y2,x).subs(x,x0)

y1_0 = y1.subs(x,x0)
y2_0 = y2.subs(x,x0)

solve([y1_0-y2_0, y1_diff-y2_diff], [x0, a])
```

Out[91]:

```
[(exp(1/2), exp(-1)/2)]
```

In [92]:

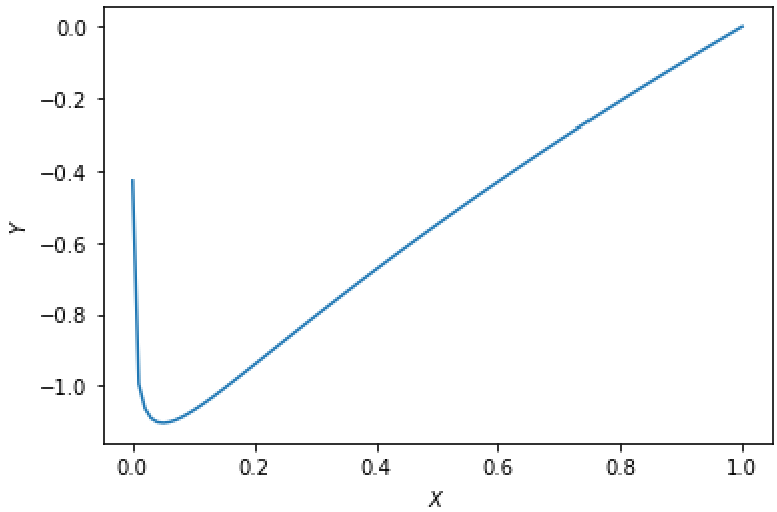
```
x = np.linspace(-4, 6, 500)
y = x**2/(2*np.exp(1))
plt.plot(x, y, lw=2, c='r')
x = np.linspace(0.03, 6, 100)
y = np.log(x)
plt.plot(x, y, lw=2, c='b')
plt.xlabel('$X$')
plt.ylabel('$Y$')
plt.grid(True, linestyle='-', color='0.4')
plt.show()
```



Исследовать на экстремум функцию $y = \sqrt[3]{x} \times lnx$.

In [93]:

```
f = lambda x: (x**(1/3))*np.log(x)

x = np.linspace(0.0001,1,100)
y = f(x)
plt.plot(x, y)
plt.xlabel('$X$')
plt.ylabel('$Y$')
plt.show()
```



In [94]:
```
res = minimize(f, 0.01)
print(f'xmin: {res.x[0]:.4f} y(x_min): {f(res.x)[0]:.4f}')
```

```
xmin: 0.0498 y(x_min): -1.1036
```

In [95]:
```
x = symbols('x')
y = x**(1/3) * log(x)
x_min = solve(diff(y,x))[0]
print(f'x_min: {x_min:.4f} y(x_min): {y.subs(x, x_min):.3f}')
```

```
x_min: 0.0498 y(x_min): -1.104
```

Найти производную функции $w = \frac{x^2}{2} + \frac{y^2}{9} - z^2$ в точке $A(1;2)$ по направлению радиус-вектора этой точки.

In [96]:
```
l = Point(2,3,1)

l_n = l.distance(Point(0,0,0))

cos_a = l.x/l_n
cos_b = l.y/l_n
cos_c = l.z/l_n
```

In [97]:
```
x,y,z = symbols('x y z')
w = x**2/2 + y**2/9 - z**2

w_x = diff(w,x).subs({x:2, y:3, z:1})
w_y = diff(w,y).subs({x:2, y:3, z:1})
w_z = diff(w,z).subs({x:2, y:3, z:1})
```

In [98]:
```
w_x = diff(w,x).subs([(x,2),(y,3),(z,1)])
w_y = diff(w,y).subs([(x,2),(y,3),(z,1)])
w_z = diff(w,z).subs([(x,2),(y,3),(z,1)])

w_l = w_x*cos_a + w_y*cos_b + w_z*cos_c
w_l
```

Out[98]:

$$\frac{2\sqrt{14}}{7}$$

Найти экстремумы функции $z = x^2 - 4xy - 2y^2 + 8x$

In [99]:
```
def critical_points(z):
```

```python
    z_x = diff(z,x)
    z_y = diff(z,y)

    cr_point = solve([z_x, z_y], [x, y], dict=True)

    A = diff(z,x,2)
    B = diff(z,x,y)
    C = diff(z,y,2)

    D = A*C - B**2
    return cr_point, A, D
```

In [100…

```python
def suff_indic(A, D, cr_point):
    A0 = A.subs(cr_point)
    D0 = D.subs(cr_point)
    return D0, A0
```

In [101…

```python
x, y = symbols('x y')
z = x**2 - 4*x*y - 2*y**2 + 8*x
cr_point, A, D = critical_points(z)
cr_point
```

Out[101]:

```
[{x: -4/3, y: 4/3}]
```

In [102…

```python
D0, A0 = suff_indic(A, D, cr_point[0])
D0, A0
```

Out[102]:

```
(-24, 2)
```

Зависимость между себестоимостью продукции C и объёмом её производства Q выражается формулой $C(Q) = 80 - 0,38Q$. Определить эластичность себестоимости при выпуске продукции $Q = 20$ ден. ед.

In [103…

```python
Q = symbols('Q')
c = 80 - 0.38*Q
Dprim = diff(c,Q)
E = (Q*Dprim/c).subs(Q,20)
S(E).n(3)
```

Out[103]: -0.105

Функция спроса D и предложения S от цены p имеют вид: $D(p) = 40 - 1,3p$, $S(p) = 20 + 1,2p$. Найти эластичность спроса в точке равновенсой цены.

In [104…

```python
p = symbols('p')
D = 40 - 1.3*p
S = 20 + 1.2*p
p0 = solve(D-S,p)
p0[0].n(2)
```

Out[104]: 8.0

In [105…

```python
Dprim = diff(D,p)
E = (p*Dprim/D).subs(p,p0[0])
E.n(3)
```

Out[105]: -0.351

Решение собственной задачи с использованием производных

Модель Блэка-Шоулза. Уравнения Блэка-Шоулза произвели революцию в ценообразовании опционов, когда в 1973 году Мрайон Шоулз и Фишер Блэк опубликовали свою работу.

При рассмотрении приведенных ниже формул необходимо учитывать ряд важных допущений:

1. Процентная ставка известна и постоянна во времени.
2. Акции следуют случайному блужданию в непрерывном времени, дисперсия путей цены акции следует логнормальному распределению.
3. Волатильность постоянна
4. Акции не выплачивают дивиденды (однако их можно модифицировать для включения дивидендов).
5. Опцион может быть исполнен только по истечении срока действия, т.е. это европейский тип опциона.
6. Отсутствие транзакционных издержек, т.е. комиссий за продажу в короткую позицию и т.д.
7. Возможна дробная торговля, т.е. мы можем купить/продать 0,х любой акции.

Формулы Блэка-Шоулза для акций, не выплачивающих дивиденды

$$Call = S_0 N(d_1) - N(d_2) K e^{-rT}$$

$$Put = N(-d_2) K e^{-rT} - N(-d_1) S_0$$

$$d_1 = \frac{ln(\frac{S}{K}) + (r + \frac{\sigma^2}{2})T}{\sigma\sqrt{T}}$$
$$d_2 = d_1 - \sigma\sqrt{T}$$

S : текущая цена актива

K: цена исполнения опциона

r: безрисковая ставка

T : время до истечения срока опциона

σ: годовая волатильность доходности актива

N(x): кумулятивная функция распределения для стандартного нормального распределения, показанного ниже.

$$N(x) = \int_{-\infty}^{x} \frac{e^{-x^2/2}}{\sqrt{2\pi}}$$

In [106…

```
N = norm.cdf
```

```
def BS_CALL(S, K, T, r, sigma):
    d1 = (np.log(S/K) + (r + sigma**2/2)*T) / (sigma*np.sqrt(T))
    d2 = d1 - sigma * np.sqrt(T)
    return S * N(d1) - K * np.exp(-r*T)* N(d2)

def BS_PUT(S, K, T, r, sigma):
    d1 = (np.log(S/K) + (r + sigma**2/2)*T) / (sigma*np.sqrt(T))
    d2 = d1 - sigma* np.sqrt(T)
    return K*np.exp(-r*T)*N(-d2) - S*N(-d1)
```

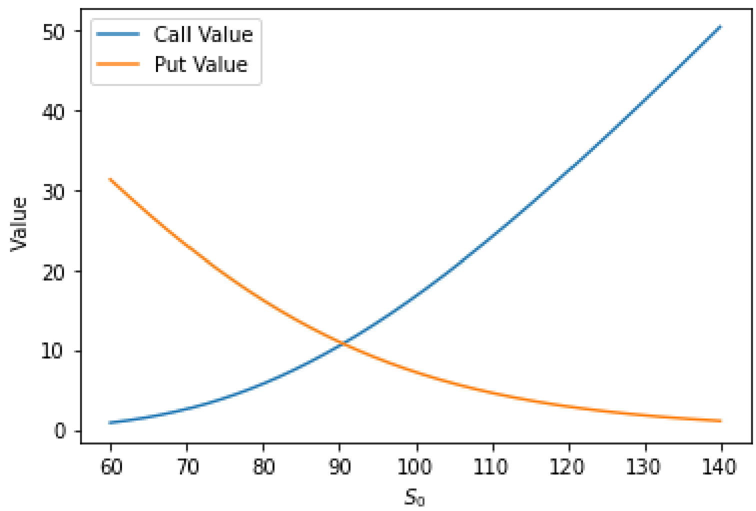
В этом разделе мы рассмотрим влияние изменения входных параметров на стоимость контрактов и опционов.

Влияние S на стоимость опциона. Здесь мы будем поддерживать постоянными все переменные, кроме текущей цены акций S, и рассмотрим, как меняется стоимость контрактов и опционов.

In [107…

```
K = 100
r = 0.1
T = 1
sigma = 0.3

S = np.arange(60,140,0.1)

calls = [BS_CALL(s, K, T, r, sigma) for s in S]
puts = [BS_PUT(s, K, T, r, sigma) for s in S]
plt.plot(S, calls, label='Call Value')
plt.plot(S, puts, label='Put Value')
plt.xlabel('$S_0$')
plt.ylabel(' Value')
plt.legend()
```

Out[107]: <matplotlib.legend.Legend at 0x1ccc4fc8d00>



Влияние σ на стоимость по Блэку-Шоулзу

Как и следовало ожидать, при постоянстве других переменных и увеличении параметра волатильности стоимость контрактов и опционов увеличивается линейно, как показано ниже.

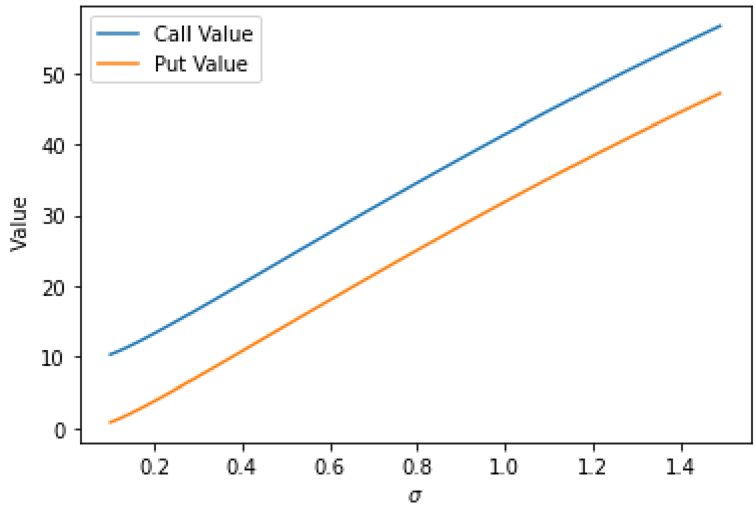
Чтобы понять, почему стоимость контрактов строго больше стоимости опционов в зависимости от волатильности, изменим процентную ставку r до 0 и заметим, что кривые точно совпадают. Вместо того чтобы строить графики влияния на процентные ставки, мы можем сделать вывод, что увеличение процентных ставок увеличивает стоимость контрактов и уменьшает стоимость опционов.

In [108…

```
K = 100
r = 0.1
T = 1
Sigmas = np.arange(0.1, 1.5, 0.01)
S = 100

calls = [BS_CALL(S, K, T, r, sig) for sig in Sigmas]
puts = [BS_PUT(S, K, T, r, sig) for sig in Sigmas]
plt.plot(Sigmas, calls, label='Call Value')
plt.plot(Sigmas, puts, label='Put Value')
plt.xlabel('$\sigma$')
plt.ylabel(' Value')
plt.legend()
```

Out[108]:

```
<matplotlib.legend.Legend at 0x1ccc501edc0>
```



Влияние времени на цену по Блэку-Шоулзу

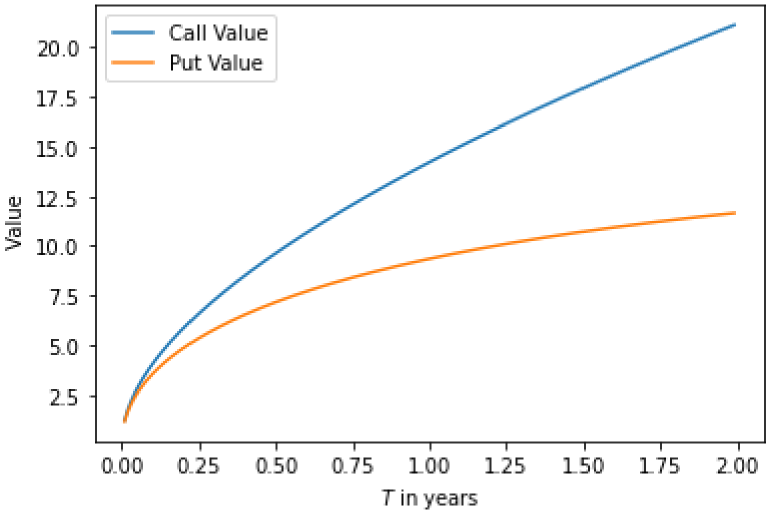
С увеличением времени увеличивается неопределенность относительно будущей цены. Поскольку неопределенность идет на пользу держателю опциона, цена опциона растет со временем. Снова попробуем установить процентную ставку на ноль, чтобы увидеть, что разница между опционами и контрактами исчезает.

In [109…

```
K = 100
r = 0.05
T = np.arange(0, 2, 0.01)
sigma = 0.3
S = 100

calls = [BS_CALL(S, K, t, r, sigma) for t in T]
puts = [BS_PUT(S, K, t, r, sigma) for t in T]
plt.plot(T, calls, label='Call Value')
plt.plot(T, puts, label='Put Value')
plt.xlabel('$T$ in years')
plt.ylabel(' Value')
plt.legend()
```

Out[109]:

```
<matplotlib.legend.Legend at 0x1ccc5082490>
```



Основная проблема подхода Блэка Шоулза заключается в том, что она не может адекватно работать при условии изменчивой волатильности.