Аналитическая геометрия

Пример 1

Ввод [1]:

```
import numpy as np
from numpy.linalg import norm

from sympy import *
from sympy.abc import x, y

import matplotlib.pyplot as plt
from matplotlib import cm
from mpl_toolkits.mplot3d import Axes3D


import math

%matplotlib inline

import warnings
warnings.filterwarnings('ignore')
```

Ввод [2]:

```
a = np.array([1,2,3,4,5])
a
```

Out[2]:

```
array([1, 2, 3, 4, 5])
```

Ввод [3]:

```
a[1]
```

Out[3]:

```
2
```

Ввод [4]:

```
a[1:4]
```

Out[4]:

```
array([2, 3, 4])
```

Ввод [5]:

```
a[-1]
```

Out[5]:

```
5
```

Ввод [6]:

```
a = np.array([1,2,3,4])
b = np.array([11,12,13,14])
c = a+b
d = a-b
e = 4*a
print('a+b: %s, a-b: %s, 4a: %s' % (c, d, e))
```

```
a+b: [12 14 16 18], a-b: [-10 -10 -10 -10], 4a: [ 4  8 12 16]
```

Пример 2

Ввод [7]:

```
a = np.array([3,4])
norm(a)
```

Out[7]:

```
5.0
```

Пример 3

Ввод [8]:

```
f = np.array([3,1,4])
g = np.array([0,-2,1])
np.dot(f, g)
```

Out[8]:

```
2
```

Пример 4

Ввод [9]:

```
a = np.array([1,2])
b = np.array([3,4])
np.dot(a,b)/norm(b)
```

Out[9]:

```
2.2
```

Пример 5

Ввод [10]:

```
f = np.array([3,1,4])
g = np.array([0,-2,1])
np.cross(f,g)
```

Out[10]:

```
array([ 9, -3, -6])
```

Ввод [11]:

```
a = np.array([1,2,3])
b = np.array([4,0,-1])
c = np.array([0,5,-2])
np.dot(a,np.cross(b,c))
```

Out[11]:

```
81
```

Ввод [12]:

```
A = np.array([a,b,c])
np.linalg.det(A)
```

Out[12]:

```
80.99999999999996
```

Ввод [13]:

```
P1P = np.array([8-2,5-2])
P2P = np.array([8-10,5-6])
np.cross(P1P,P2P)
```

Out[13]:

```
array(0)
```

Пример 6

Ввод [14]:

```
a = Matrix([[1,2,3], [0,-1, 1]])
a
```

Out[14]:

$\displaystyle \left[\begin{matrix}1 & 2 & 3\\0 & -1 & 1\end{matrix}\right]$

Ввод [15]:

```
A = Matrix([[1,2,3]])
A
```

Out[15]:

$\displaystyle \left[\begin{matrix}1 & 2 & 3\end{matrix}\right]$

Ввод [16]:

```
A = Matrix([[1],[2],[3]])
A
```

Out[16]:

$\displaystyle \left[\begin{matrix}1\\2\\3\end{matrix}\right]$

Ввод [17]:

```
A = Matrix([1,2,3])
A
```

Out[17]:

$\displaystyle \left[\begin{matrix}1\\2\\3\end{matrix}\right]$

Точки

Ввод [18]:

```
Point(1, 2, 3)
```

Out[18]:

$\displaystyle \operatorname{Point3D}\left(1, 2, 3\right)$

Ввод [19]:

```
Point([1, 2])
```

Out[19]:

$\displaystyle \operatorname{Point2D}\left(1, 2\right)$

Ввод [20]:

```
p = Point(dim=4)
```

Ввод [21]:

```
p1, p2 = Point(1, 1), Point(4, 5)
p1.distance(p2)
```

Out[21]:

$\displaystyle 5$

Ввод [22]:

```
p3 = Point(x, y)
p3.distance(Point(0, 0))
```

Out[22]:

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

Ввод [23]:

```
p1, p2 = Point(1, 1), Point(13, 5)
p1.midpoint(p2)
```

Out[23]:

$\displaystyle \operatorname{Point2D}\left(7, 3\right)$

Ввод [24]:

```
p1 = Point3D(1, 2, 2)
p2 = Point3D(2, 7, 2)
p3 = Point3D(0, 0, 2)
p4 = Point3D(1, 1, 2)
Point3D.are_coplanar(p1, p2, p3, p4)
```

Out[24]:

```
True
```

Ввод [25]:

```
p1, p2 = Point(0, 0), Point(1, 1)
p3, p4, p5 = Point(2, 2), Point(x, x), Point(1, 2)
Point.is_collinear(p1, p2, p3, p4)
```

Out[25]:

```
False
```

Ввод [26]:

```
pl, p2, p3, P4 = Point(1, 0), (0, 1), (-1, 0), (0, -1)
pl.is_concyclic()
```

Out[26]:

```
True
```

Прямые на плоскости и в пространстве

Ввод [27]:

```
Line((0, 0), (1, 1))
```

Out[27]:

$\displaystyle \operatorname{Line2D}\left(\operatorname{Point2D}\left(0, 0\right), \operatorname{Point2D}\left(1, 1\right)\right)$

Ввод [28]:

```
pl, p2 = Point(1, 1), Point(3, 0)
Line(p1, p2)
```

Out[28]:

$\displaystyle \operatorname{Line2D}\left(\operatorname{Point2D}\left(0, 0\right), \operatorname{Point2D}\left(3, 0\right)\right)$

Ввод [29]:

```
p1, p2 = Point(1,0), Point(5,3)
l1 = Line(p1, p2)
l1.equation()
```

Out[29]:

$\displaystyle - 3 x + 4 y + 3$

Ввод [30]:

```
l1.coefficients
```

Out[30]:

```
(-3, 4, 3)
```

Ввод [31]:

```
p1, p2 = Point(1,0,0), Point(5,3,2)
l2 = Line(p1, p2)
l2.equation()
```

Out[31]:

$\displaystyle \left(- 3 x + 4 y + 3, \ - x + 2 z + 1\right)$

Ввод [32]:

```
p1, p2 = Point(1, 0), Point(5, 3)
l1 = Line(p1, p2)
l1.arbitrary_point()
```

Out[32]:

$\displaystyle \operatorname{Point2D}\left(4 t + 1, 3 t\right)$

Ввод [33]:

```
p1, p2 = Point3D(1, 0, 0), Point3D(5, 3, 1)
l1 = Line3D(p1, p2)
l1.arbitrary_point()
```

Out[33]:

$\displaystyle \operatorname{Point3D}\left(4 t + 1, 3 t, t\right)$

Ввод [34]:

```
A = Point(-2,3)
k = 2
l = Line(A, slope = k)
l.equation()
```

Out[34]:

$\displaystyle - 2 x + y - 7$

Ввод [35]:

```
A = Point(-2,3,0)
l = Line(A, direction_ratio=[1,2,0])
l.equation()
```

Out[35]:

$\displaystyle \left(- 2 x + y - 7, \ z\right)$

Ввод [36]:

```
a, b = (0,0), (3,3)
Line(a, b).direction
```

Out[36]:

$\displaystyle \operatorname{Point2D}\left(3, 3\right)$

Ввод [37]:

```
Line(a, b).direction.unit
```

Out[37]:

$\displaystyle \operatorname{Point2D}\left(\frac{\sqrt{2}}{2}, \frac{\sqrt{2}}{2}\right)$

Ввод [38]:

```
p1, p2 = Point(0, 0, 0), Point(1, 1, 1)
s = Line(p1, p2)
s.distance(Point(-1, 1, 1))
```

Out[38]:

$\displaystyle \frac{2 \sqrt{6}}{3}$

Ввод [39]:

```
e = Line((0, 0), (1, 0))
sw = Line((1, 1), (0, 0))
sw.angle_between(e)
```

Out[39]:

$\displaystyle \frac{3 \pi}{4}$

Ввод [40]:

```
sw.smallest_angle_between(e)
```

Out[40]:

$\displaystyle \frac{\pi}{4}$

Ввод [41]:

```
p1, p2 = Point(0, 0), Point(3, 5)
p3, p4 = Point(-2, -2), Point(0, 2)
l1, l2, l3 = Line(p1, p2), Line(p1, p3), Line(p1, p4)
Line.are_concurrent(l1, l2, l3)
```

Out[41]:

```
True
```

Ввод [42]:

```
l1 = Line3D(Point3D(4,19,12), Point3D(5,25,17))
l2 = Line3D(Point3D(-3, -15, -19), direction_ratio=[2,8,8])
l1.intersection(l2)
```

Out[42]:

```
[Point3D(1, 1, -3)]
```

Ввод [43]:

```
p1, p2 = Point3D(0, 0, 0), Point3D(3, 4, 5)
p3, p4 = Point3D(2, 1, 1), Point3D(8, 9, 11)
l1, l2 = Line3D(p1, p2), Line3D(p3, p4)
Line3D.is_parallel(l1, l2)
```

Out[43]:

```
True
```

Ввод [44]:

```
p1, p2, p3 = Point(0, 1), Point(3, 4), Point(2, 3)
l1 = Line(p1, p2)
l2 = Line(p1, p3)
l1.is_similar(l2)
```

Out[44]:

```
True
```

Ввод [45]:

```
p1, p2, p3 = Point(0, 0), Point(2,3), Point(-2, 2)
l1 = Line(p1, p2)
l2 = l1.parallel_line(p3)
p3 in l2
```

Out[45]:

```
True
```

Ввод [46]:

```
p1, p2, p3 = Point(0, 0), Point(2, 3), Point(-2, 2)
l1 = Line(p1, p2)
l2 = l1.perpendicular_line(p3)
p3 in l2
```

Out[46]:

```
True
```

Ввод [47]:

```
p1, p2, p3 = Point(0, 0), Point(1, 1), Point(Rational(1, 2), 0)
l1 = Line(p1, p2)
l1.projection(p3)
```

Out[47]:

$\displaystyle \operatorname{Point2D}\left(\frac{54}{25}, \frac{72}{25}\right)$

Плоскости

Ввод [48]:

```
a = Plane(Point3D(1, 1, 2), Point3D(2, 4, 7), Point3D(3, 5, 1))
a.equation()
```

Out[48]:

$\displaystyle - 23 x + 11 y - 2 z + 16$

Ввод [49]:

```
a = Plane(Point3D(1, 4, 2), normal_vector=(6, 6, 6))
a.equation()
```

Out[49]:

$\displaystyle 6 x + 6 y + 6 z - 42$

Пример 7

Ввод [50]:

```
Plane((1, 1, 1), (1, 4, 7))
```

Out[50]:

$\displaystyle \operatorname{Plane}\left(\operatorname{Point3D}\left(1, 1, 1\right), \left(1, \ 4, \ 7\right)\right)$

Ввод [51]:

```
a = Plane((1, 1, 1), (0, 0, 1))
p = a.p1
p
```

Out[51]:

$\displaystyle \operatorname{Point3D}\left(1, 1, 1\right)$

Ввод [52]:

```
a = Plane(Point3D(1, 1, 2), Point3D(2, 4, 7), Point3D(3, 5, 1))
a.equation()
```

Out[52]:

$\displaystyle - 23 x + 11 y - 2 z + 16$

Ввод [53]:

```
a = Plane(Point3D(1, 1, 1), Point3D(2, 3, 4), Point3D(2, 2, 2))
a.normal_vector
```

Out[53]:

$\displaystyle \left(-1, \ 2, \ -1\right)$

Ввод [54]:

```
a = Plane(Point3D(1, 4, 6), normal_vector=(2, 4, 6))
a.parallel_plane(Point3D(2, 3, 5))
```

Out[54]:

$\displaystyle \operatorname{Plane}\left(\operatorname{Point3D}\left(2, 3, 5\right), \left(2, \ 4, \ 6\right)\right)$

Ввод [55]:

```
a, b = Point3D(0, 0, 0), Point3D(0, 1, 0)
Z = (0, 0, 1)
p = Plane(a, normal_vector=Z)
p.perpendicular_plane(a, b)
```

Out[55]:

$\displaystyle \operatorname{Plane}\left(\operatorname{Point3D}\left(0, 0, 0\right), \left(1, \ 0, \ 0\right)\right)$

Ввод [56]:

```
a = Plane(Point3D(1,4,6), normal_vector=(2, 4, 6))
a.perpendicular_line(Point3D(9, 8, 7))
```

Out[56]:

$\displaystyle \operatorname{Line3D}\left(\operatorname{Point3D}\left(9, 8, 7\right), \operatorname{Point3D}\left(11, 12, 13\right)\right)$

Ввод [57]:

```
a = Plane(Point3D(1, 1, 1), normal_vector=(1, 1, 1))
b = Point3D(1, 2, 3)
a.distance(b)
```

Out[57]:

$\displaystyle \frac{13 \sqrt{14}}{14}$

Ввод [58]:

```
a = Plane(Point3D(1, 2, 2), normal_vector=(1, 2, 3))
b = Line3D(Point3D(1, 3, 4), Point3D(2, 2, 2))
a.angle_between(b)
```

Out[58]:

$\displaystyle - \operatorname{asin}{\left(\frac{\sqrt{21}}{6} \right)}$

Ввод [59]:

```
a = Plane(Point3D(1, 2, 3), normal_vector=(1, 1, 1))
l = Line((-1, 2, 0), (5, -3, 0))
a.intersection(l)
```

Out[59]:

```
[Point3D(-37, 32, 0)]
```

Ввод [60]:

```
d = Plane(Point3D(6, 0, 0), normal_vector=(1, 1, 1))
e = Plane(Point3D(2, 0, 0), normal_vector=(3, 4, -3))
d.intersection(e)
```

Out[60]:

```
[Line3D(Point3D(18, -12, 0), Point3D(11, -6, 1))]
```

Ввод [61]:

```
a = Plane(Point3D(5, 0, 0), normal_vector=(1, -1, 1))
b = Plane(Point3D(0, -2, 0), normal_vector=(3, 1, 1))
Plane.are_concurrent(a, b)
```

Out[61]:

```
True
```

Ввод [62]:

```
a = Plane(Point3D(1, 2, 3), normal_vector=(1, 1, 1))
b = Plane(Point3D(1, 2, 3), normal_vector=(2, 2, 2))
a.equals(b)
```

Out[62]:

```
False
```

Ввод [63]:

```
a = Plane(Point3D(1,4,6), normal_vector=(2, 4, 6))
b = Plane(Point3D(3,1,3), normal_vector=(4, 8, 12))
a.is_parallel(b)
```

Out[63]:

```
False
```

Ввод [64]:

```
a = Plane((1,4,6), (2, 4, 6))
l = Line(Point(1, 3, 4), Point(2, 2, 2))
a.is_parallel(l)
```

Out[64]:

```
True
```

Ввод [65]:

```
a = Plane(Point3D(1,4,6), normal_vector=(2, 4, 6))
b = Plane(Point3D(2, 2, 2), normal_vector=(-1, 2, -1))
a.is_perpendicular(b)
```

Out[65]:

```
False
```

Ввод [66]:

```
a = Plane(Point3D(1,4,6), normal_vector=(2, 4, 6))
l = Line3D(Point3D(1, 3, 4), Point3D(2, 2, 2))
a.is_perpendicular(l)
```

Out[66]:

```
False
```

Ввод [67]:

```
a = Plane(Point3D(1, 1, 1), normal_vector=(1, 1, 1))
c = Line3D(Point3D(1, 1, 1), Point3D(2, 2, 2))
a.projection_line(c)
```

Out[67]:

$\displaystyle \operatorname{Point3D}\left(1, 1, 1\right)$

Ввод [68]:

```
a = Plane(Point3D(1, 1, 1), normal_vector=(1, 1, 1))
c = Line3D(Point3D(0, 0, 0), Point3D(1, 0, 1))
a.projection_line(c)
```

Out[68]:

$\displaystyle \operatorname{Line3D}\left(\operatorname{Point3D}\left(1, 1, 1\right), \operatorname{Point3D}\left(\frac{4}{3}, \frac{1}{3}, \frac{4}{3}\right)\right)$

Ввод [69]:

```
a = Plane(Point3D(1, 1, 2), normal_vector=(1, 1, 1))
b = Point3D(0, 1, 2)
a.projection(b)
```

Out[69]:

$\displaystyle \operatorname{Point3D}\left(\frac{1}{3}, \frac{4}{3}, \frac{7}{3}\right)$

Пример 8

Ввод [70]:

```
def distance_line(A,B,M,N):
    AB = Point(B.x-A.x,B.y-A.y,B.z-A.z)
    l1 = Line(A,B)
    l2 = Line(M,N)
    p = Plane(A, normal_vector=AB)
    pl2 = p.projection_line(l2)
    d = pl2.distance(A)
    return(d)
```

Ввод [71]:

```
A = Point(0,0,0)
D = Point(1,1,0)
E = Point(0,1,0)
B1 = Point(1,0,1)

distance_line(A,E,D, B1)
```

Out[71]:

$\displaystyle 1$

Отрезок

Ввод [72]:

```
s = Segment((1,0), (4,4))
s
```

Out[72]:

$\displaystyle \operatorname{Segment2D}\left(\operatorname{Point2D}\left(1, 0\right), \operatorname{Point2D}\left(4, 4\right)\right)$

Ввод [73]:

```
s.points
```

Out[73]:

```
(Point2D(1, 0), Point2D(4, 4))
```

Ввод [74]:

```
s.slope
```

Out[74]:

$\displaystyle \frac{4}{3}$

Ввод [75]:

```
s.length
```

Out[75]:

$\displaystyle 5$

Ввод [76]:

```
s.midpoint
```

Out[76]:

$\displaystyle \operatorname{Point2D}\left(\frac{5}{2}, 2\right)$

Ввод [77]:

```
A = Point(0,2)
s.distance(A)
```

Out[77]:

$\displaystyle 2$

Ввод [78]:

```
s = Segment((0,2), (2,0))
s.perpendicular_bisector()
```

Out[78]:

$\displaystyle \operatorname{Line2D}\left(\operatorname{Point2D}\left(1, 1\right), \operatorname{Point2D}\left(3, 3\right)\right)$

Ввод [79]:

```
Line(s)
```

Out[79]:

$\displaystyle \operatorname{Line2D}\left(\operatorname{Point2D}\left(0, 2\right), \operatorname{Point2D}\left(2, 0\right)\right)$

Ввод [80]:

```
Line(s).equation()
```

Out[80]:

$\displaystyle 2 x + 2 y - 4$

Пример 9

Ввод [81]:

```
def Point_oneside_L(A,B,l):
    s = Segment(A,B)
    return not(Line.are_concurrent(l, s))
```

Ввод [82]:

```
l = Line((2,0), (0,2))
A = Point(1,0)
B = Point(0,1)
D = Point(1,2)
Point_oneside_L(A,B,l)
```

Out[82]:

```
True
```

Пример 10

Ввод [83]:

```
Point_oneside_L(A,D,l)
```

Out[83]:

```
False
```

Пример 11

Ввод [84]:

```
def Point_oneside_P(A,B,P):
    s = Segment(A,B)
    p = P.intersection(s)
    if len(p) == 0:
        return True
    else:
        return not(s.contains(p[0]))
```

Ввод [85]:

```
P = Plane((3,0,0), (0,3,0), (0,0,3))
A = Point(0,0,0)
B = Point(5,5,5)
D = Point(1,0,0)
Point_oneside_P(A,D,P)
```

Out[85]:

```
True
```

Ввод [86]:

```
Point_oneside_P(A,B,P)
```

Out[86]:

```
False
```

Пример 12

Ввод [87]:

```
def Point_opposite_l(A,l):
    A0 = l.projection(A)
    x = 2*A0.x - A.x
    y = 2*A0.y - A.y
    if len(A) == 2:
        return Point(x,y)
    elif len(A) == 3:
        z = 2*A0.z - A.z
        return Point(x,y,z)
```

Ввод [88]:

```
A = Point(0,7)
l = Line((0,-3),(2,1))
Point_opposite_l(A,l)
```

Out[88]:

$\displaystyle \operatorname{Point2D}\left(8, 3\right)$

Пример 13

Ввод [89]:

```
def Point_opposite_P(A,P):
    A0 = P.projection(A)
    x =2*A0.x - A.x
    y =2*A0.y - A.y
    z =2*A0.z - A.z
    return Point(x,y,z)
```

Ввод [90]:

```
A = Point(0,0,0)
P = Plane((3,0,0),(0,3,0),(0,0,3))
Point_opposite_P(A,l)
```

Out[90]:

$\displaystyle \operatorname{Point3D}\left(\frac{12}{5}, - \frac{6}{5}, 0\right)$

Луч

Ввод [91]:

```
Ray((0,0,0), (1,1,1))
```

Out[91]:

$\displaystyle \operatorname{Ray3D}\left(\operatorname{Point3D}\left(0, 0, 0\right), \operatorname{Point3D}\left(1, 1, 1\right)\right)$

Ввод [92]:

```
r = Ray((0,0), (1,0))
r.rotate(-pi/2)
```

Out[92]:

$\displaystyle \operatorname{Ray2D}\left(\operatorname{Point2D}\left(0, 0\right), \operatorname{Point2D}\left(0, -1\right)\right)$

Треугольник

Ввод [93]:

```
A = Point(0,0)
B = Point(0,4)
D = Point(3,0)
Triangle(A,B,D)
```

Out[93]:

$$\operatorname{Triangle}(\operatorname{Point2D}(0, 0), \operatorname{Point2D}(0, 4), \operatorname{Point2D}(3, 0))$$

Ввод [94]:

```
Triangle(sss=(1,1,1))
```

Out[94]:

$$\operatorname{Triangle}\left(\operatorname{Point2D}(0, 0), \operatorname{Point2D}(1, 0), \operatorname{Point2D}\left(\frac{1}{2}, \frac{\sqrt{3}}{2}\right)\right)$$

Ввод [95]:

```
Triangle(sas=(1,45,2))
```

Out[95]:

$$\operatorname{Triangle}\left(\operatorname{Point2D}(0, 0), \operatorname{Point2D}(2, 0), \operatorname{Point2D}\left(\frac{\sqrt{2}}{2}, \frac{\sqrt{2}}{2}\right)\right)$$

Ввод [96]:

```
Triangle(asa=(90, 1, 30))
```

Out[96]:

$$\mathrm{Triangle}\left(\mathrm{Point2D}(0, 0), \mathrm{Point2D}(1, 0), \mathrm{Point2D}\left(0, \frac{\sqrt{3}}{3}\right)\right)$$

Ввод [97]:

```
T = Triangle(sss=(3,4,5))
A, B, D = T.vertices
B
```

Out[97]:

$$\mathrm{Point2D}(3, 0)$$

Ввод [98]:

```
T.vertices[2]
```

Out[98]:

$$\mathrm{Point2D}(3, 4)$$

Ввод [99]:

```
T.altitudes[T.vertices[0]]
```

Out[99]:

$$\mathrm{Segment2D}(\mathrm{Point2D}(0, 0), \mathrm{Point2D}(3, 0))$$

Ввод [100]:

```
T.orthocenter
```

Out[100]:

$$\mathrm{Point2D}(3, 0)$$

Ввод [101]:

```
T.medians[T.vertices[2]]
```

Out[101]:

$$\mathrm{Segment2D}\left(\mathrm{Point2D}(3, 4), \mathrm{Point2D}\left(\frac{3}{2}, 0\right)\right)$$

Ввод [102]:

```
T.circumcircle
T.circumcenter
T.circumradius
```

Out[102]:

$$\frac{5}{2}$$

Ввод [103]:

```
T.incircle
T. incenter
T.inradius
```

Out[103]:

$$1$$

Ввод [104]:

```
T.medial
```

Out[104]:

$$\text{Triangle}\left(\text{Point2D}\left(\frac{3}{2}, 0\right), \text{Point2D}(3, 2), \text{Point2D}\left(\frac{3}{2}, 2\right)\right)$$

Пример 14

Ввод [105]:

```
T = Triangle(sas=(3,90,4))
O1 = T.incenter
O2 = T.circumcenter
O1.distance(O2)
```

Out[105]:

$$\frac{\sqrt{5}}{2}$$

Многоугольник

Ввод [106]:

```
pl, p2, pz, p4 = [(0, 0), (4, 0), (5, 4), (0, 2)]
Polygon(pl, p2, p3, p4)
```

Out[106]:

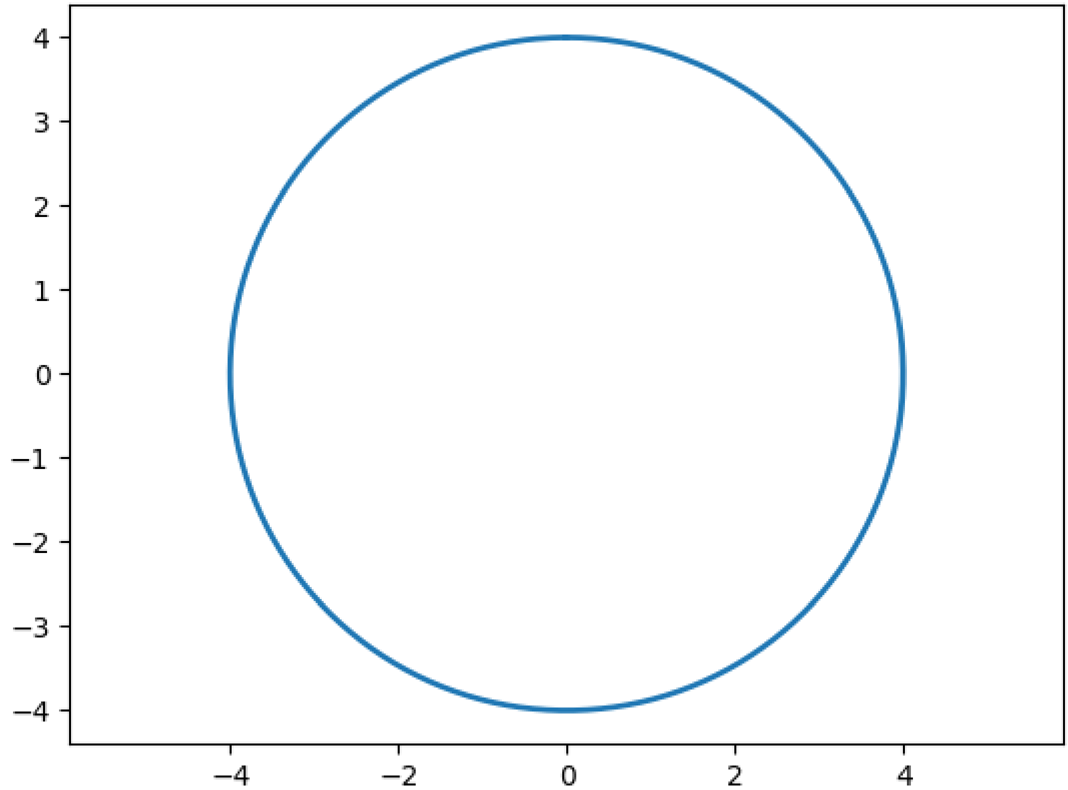
$$\text{Polygon}(\text{Point2D}(0, 0), \text{Point2D}(4, 0), \text{Point2D}(2, 3), \text{Point2D}(0, 2))$$

Кривые второго порядка

Окружность

Ввод [107]:

```
r = 4
t = np.arange(0, 2*np.pi, 0.01)
x = r*np.sin(t)
y = r*np.cos(t)
plt.plot(x, y, lw=2)
plt.axis('equal')
plt.show()
```



Ввод [108]:

```
Circle(Point(0,0), 5)
```

Out[108]:

Circle(Point2D(0, 0), 5)

Ввод [109]:

```
Circle(Point(0,0), Point(0,1), Point(1,0))
```

Out[109]:

$$\mathrm{Circle}\left(\mathrm{Point2D}\left(\frac{1}{2}, \frac{1}{2}\right), \frac{\sqrt{2}}{2}\right)$$

Ввод [110]:

```
c3 = Circle(Point(0, 0), 5)
c3.equation()
```

Out[110]:

$$x^2 + y^2 - 25$$

Ввод [111]:

```
c = Circle(Point(0, 0), Point(1, 1), Point(1, 0))
c.hradius, c.vradius, c.radius
```

Out[111]:

```
(sqrt(2)/2, sqrt(2)/2, sqrt(2)/2)
```

Ввод [112]:

```
c.center
```

Out[112]:

$$\mathrm{Point2D}\left(\frac{1}{2}, \frac{1}{2}\right)$$

Ввод [113]:

```
s = c3.area
s
```

Out[113]:

$$25\pi$$

Ввод [114]:

```
c4 = Circle(Point(3, 4), 6)
c4.circumference
```

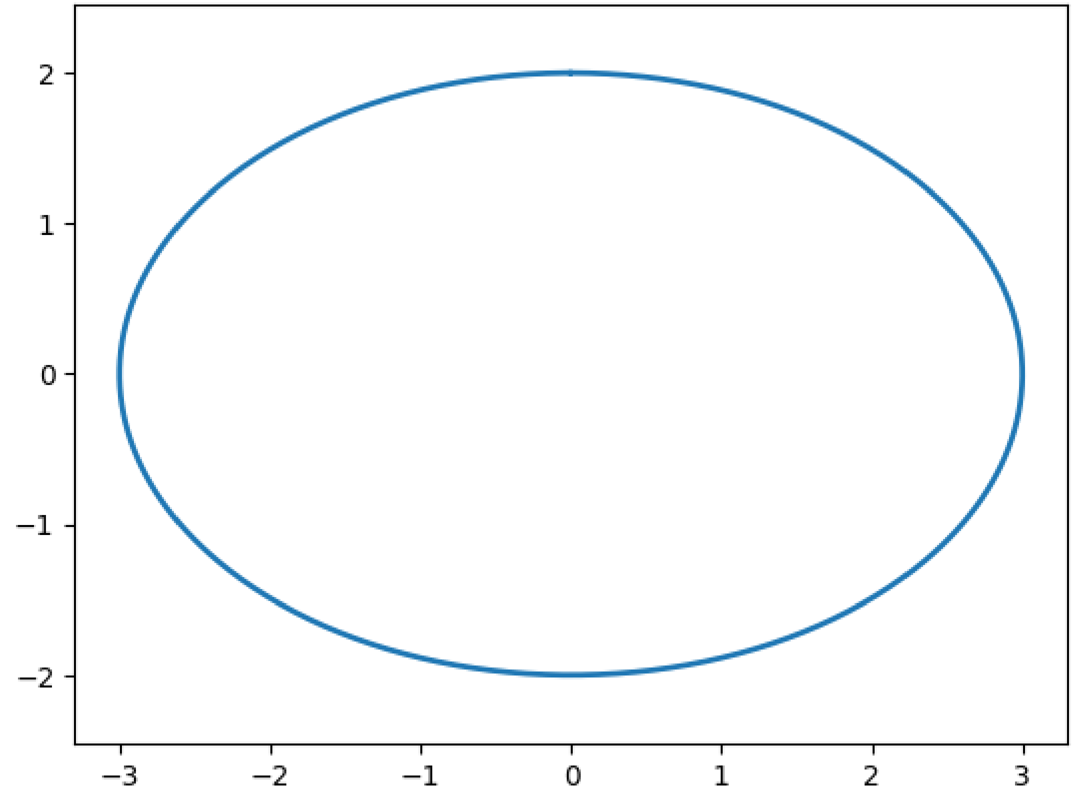
Out[114]:

$$12\pi$$

Элипс

Ввод [115]:

```
a = 3
b = 2
t = np.arange(0, 2*np.pi, 0.01)
x = a*np.sin(t)
y = b*np.cos(t)
plt.plot(x, y, lw=2)
plt.axis('equal')
plt.show()
```



Ввод [116]:

```
e1 = Ellipse(Point(0, 0), 5, 2)
e1
```

Out[116]:

$$\mathrm{Ellipse}(\mathrm{Point2D}(0, 0), 5, 2)$$

Ввод [117]:

```
e2 = Ellipse(Point(3, 1), hradius=3, eccentricity=Rational(4, 5))

e2
```

Out[117]:

$$\mathrm{Ellipse}\left(\mathrm{Point2D}(3, 1), 3, \frac{9}{5}\right)$$

Ввод [118]:

```
e1 = Ellipse(Point(1, 0), 3, 2)
e1.equation()
```

Out[118]:

$$\frac{y^2}{4} + \left(\frac{x}{3} - \frac{1}{3}\right)^2 - 1$$

Ввод [119]:

```
el = Ellipse(Point(0, 0), 3, 2)
el.arbitrary_point()
```

Out[119]:

$$\mathrm{Point2D}(3\cos(t), 2\sin(t))$$

Ввод [120]:

```
p1 = Point(0, 0)
e1 = Ellipse(p1, 3, 1)
e1.area
```

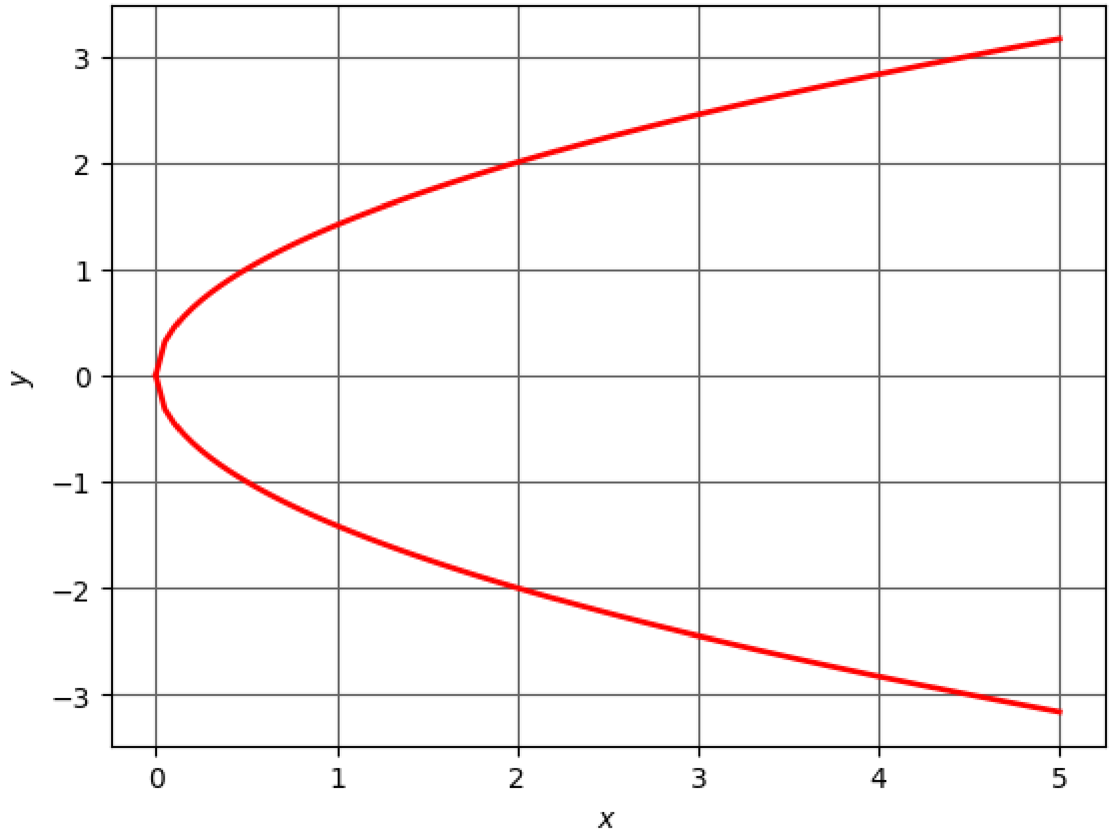
Out[120]:

$$3\pi$$

Парабола

Ввод [121]:

```
x = np.linspace(0,5,100)
y1 = np.sqrt(2*x)
y2 = -np.sqrt(2*x)

plt.plot(x, y1, lw=2, color='r')
plt.plot(x, y2, lw=2, color='r')

plt .grid(True, linestyle='-', color='0.4')
plt.ylabel('$y$')
plt.xlabel('$x$')
plt.show()
```



Ввод [122]:

```
pl = Parabola(Point(0, 0), Line(Point(5, 8), Point(7,8)))
```

Ввод [123]:

```
pl = Parabola(Point(0, 0), Line(Point(5, 8), Point(7, 8)))
pl.equation()
```

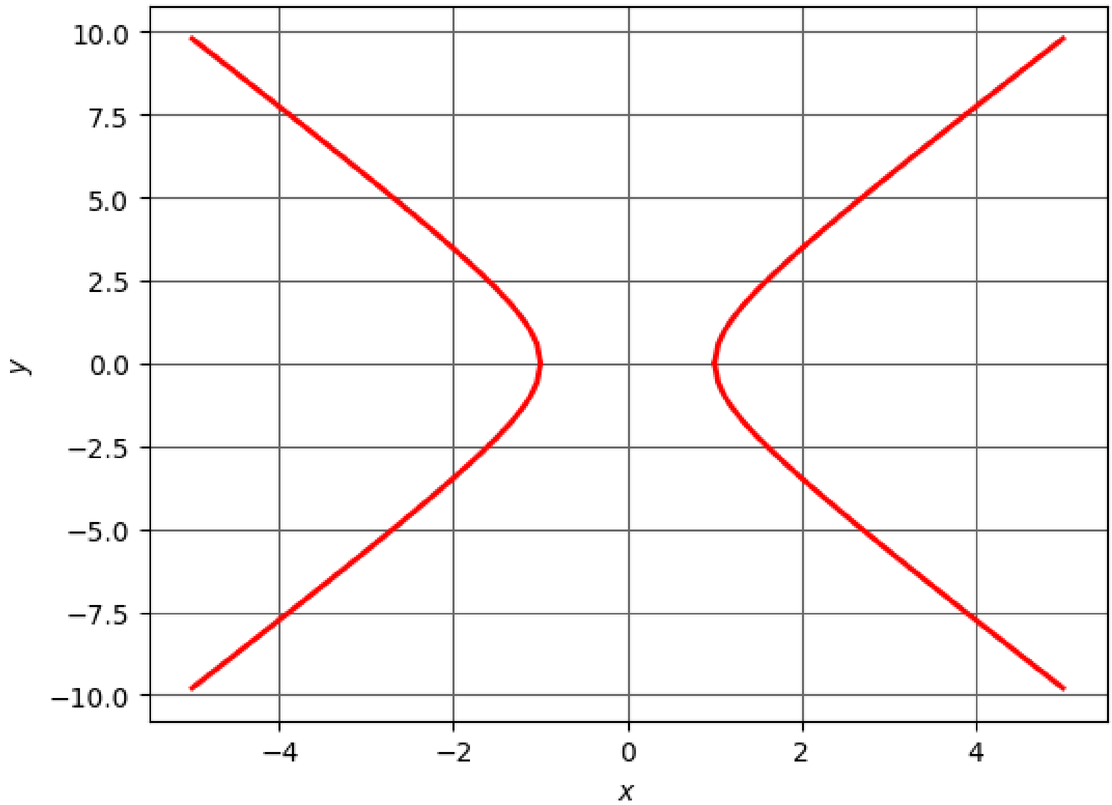
Out[123]:

$$-x^2 - 16y + 64$$

Гипербола

Ввод [124]:

```
x1 = np.linspace(-5,-1,100)
y1 = 2*np.sqrt(x1**2-1)
y2 = -2*np.sqrt(x1**2-1)
x2 = np.linspace(1,5,100)
y3 = 2*np.sqrt(x2**2-1)
y4 = -2*np.sqrt(x2**2-1)
plt.plot(x1, y1, lw=2, color='r')
plt.plot(x1, y2, lw=2, color='r')
plt.plot(x2, y3, lw=2, color='r')
plt.plot(x2, y4, lw=2, color='r')
plt.grid(True, linestyle='-', color='0.4')
plt.ylabel('$y$')
plt.xlabel('$x$')
plt.show()
```



Примедение кривой второго порядка к каноническому виду

Пример 15

Ввод [125]:

```
x = Symbol('x')
y = Symbol('y')

M = Matrix([[x**2,x*y,y**2,x,y,1],
            [1**2,1*1,1**2,1,1,1],
            [0**2,0*1,1**2,0,1,1],
            [1**2,1*0,0**2,1,0,1],
            [2**2,2*3,3**2,2,3,1],
            [3**2,3*2,2**2,3,2,1]])

det(M)
```

Out[125]:

$$-8x^2 + 32xy - 24x - 8y^2 - 24y + 32$$

Пример 16

Ввод [126]:

```
def conic_curve(A,a,f_transform=0):
    if (A.shape != (2,2)) or (len(a) != 3):
        raise ValueError('Invalid size of Aya matrices')

    a11 = A[0,0]; a12 = A[0,1]; a22 = A[1,1]
    a1 = a[0]; a2 = a[1]; a0 = a[2]
    D = det(A)
    Delta = det(Matrix([[a11,a12,a1],
    [a12,a22,a2],
    [a1, a2, a0]]))

    I = a11+a22
    B = det(Matrix([[a11,a1],
                    [a1, a0]])) + \
        det(Matrix([[a22,a2],
                    [a2, a0]]))

    if (Delta*I < 0) and (D > 0):
        print('ellipse')
    if (Delta != 0) and (D < 0):
        print('Hyperbola')
    if (Delta != 0) and (D == 0):
        print('Parabola')
    if (Delta == 0) and (D < 0):
        print(' pair of intersecting spindles')
    if (Delta == 0) and (D == 0) and (c < 0):
        print(' pair of parallel strands')
    if (Delta == 0) and (D == 0) and (B == 0):
        print(' Poyai')
    if (Delta == 0) and (D > e) and (B == e):
        print(' Point')
    if (Delta*I > 0) and (D > e):
        print(' min ellipse')
    if (Delta == 0) and (D == 0) and (B > e):
        print(' pair of imaginary parallel lines')

    T, _ = A.diagonalize()
    T1 = T.inv()
    n1 = sqrt(T1[0,0]**2+T1[1,0]**2)
    n2 = sqrt(T1[0,1]**2+T1[1,1]**2)
    x,y,x1,y1 = symbols('x y x1 y1')

    Q0 = a11*x**2 + 2*a12*x*y + a22*y**2 + \
        2*a1*x + 2*a2*y + a0
    x0 = (T1[0,0]/n1)*x1+(T1[1,0]/n1)*y1
    y0 = (T1[0,1]/n2)*x1+(T1[1,1]/n2)*y1

    Q = Q0.subs({x: x0, y: y0}).simplify()
    if (f_transform == 0):
        print('Equation: %s' % Q)
    else:
        print('Equation: %s' % Q)
        print('transition equation:')
        print('x = %s' % x0)
        print('y = %s' % y0)
```

Ввод [127]:

```
A = Matrix([[1,0],
[0,-4]])
a = Matrix([-2,0,-8])
conic_curve(A,a)
```

```
Hyperbola
Equation: -4*x1**2 + y1**2 - 4*y1 - 8
```

Пример 17

Ввод [128]:

```
A = Matrix([[1,1],
            [1,1]])
a = Matrix([2,0,1])
conic_curve(A,a,f_transform=1)
```

```
Parabola
Equation: -2*sqrt(2)*x1 + 2*y1**2 + 2*sqrt(2)*y1 + 1
transition equation:
x = -sqrt(2)*x1/2 + sqrt(2)*y1/2
y = sqrt(2)*x1/2 + sqrt(2)*y1/2
```

Кривая в пространстве

Ввод [129]:

```
t = symbols('t')
C = Curve((sin(t), cos(t)), (t, -pi, pi))
C.functions
```

Out[129]:

$(\sin(t),\ \cos(t))$

Ввод [130]:

```
C.limits
```

Out[130]:

$(t,\ -\pi,\ \pi)$

Ввод [131]:

```
C.parameter
```

Out[131]:

t

Длина кривой

Ввод [132]:

```
x = symbols('x')
Curve((x, x**2), (x, 0, 1)).length
```

Out[132]:

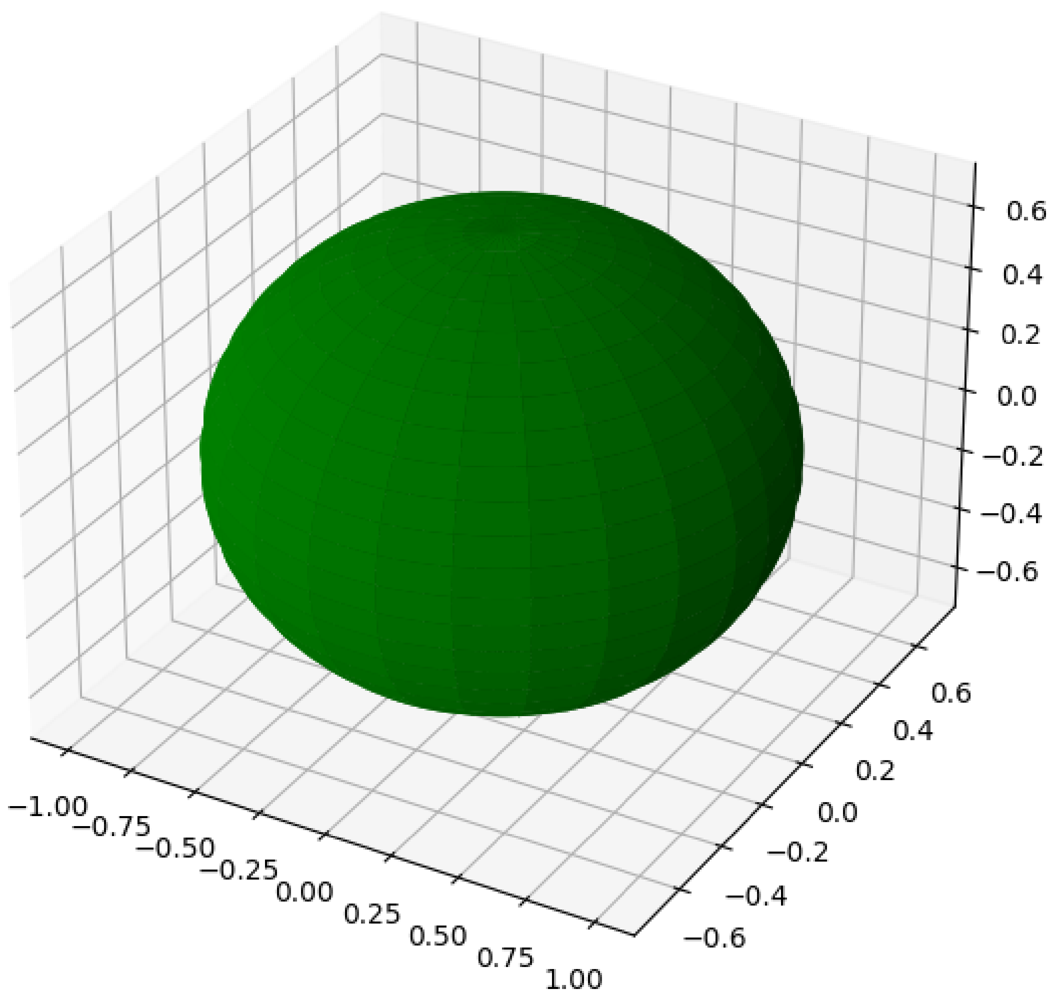
$$\frac{\operatorname{asinh}(2)}{4} + \frac{\sqrt{5}}{2}$$

Поверхности второго порядка

Элипсоид

Ввод [133]:

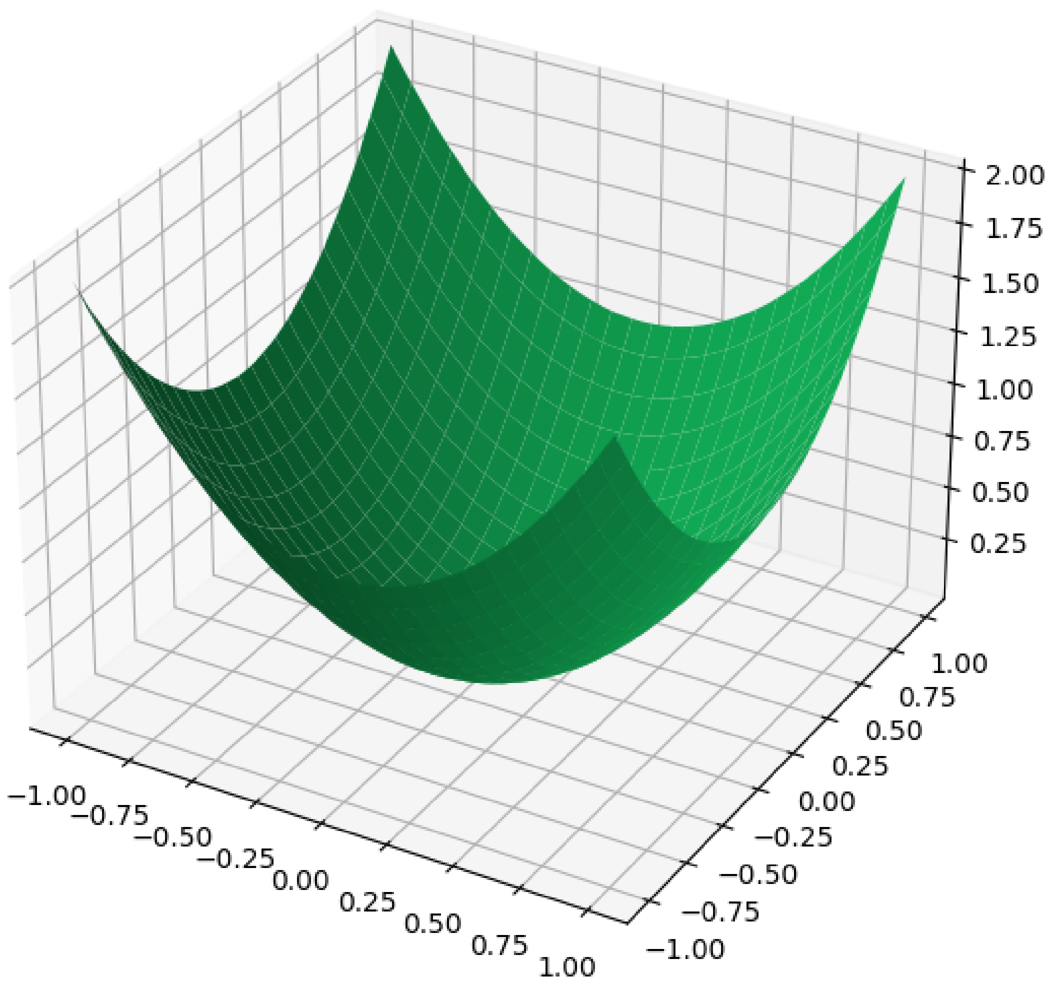
```
fig = plt.figure(figsize=(7,7))
ax = fig.add_subplot(projection='3d')

coefs = (1, 2, 2)

rx, ry, rz = 1/np.sqrt(coefs)

u = np.linspace(0, 2 * np.pi, 100)
v = np.linspace(0, np.pi, 100)

x = rx * np.outer(np.cos(u), np.sin(v))
y = ry * np.outer(np.sin(u), np.sin(v))
z = rz * np.outer(np.ones_like(u), np.cos(v))

ax.plot_surface(x, y, z, rstride=4, cstride=4, color='g')

plt.show()
```



Элиптический параболоид

Ввод [134]:

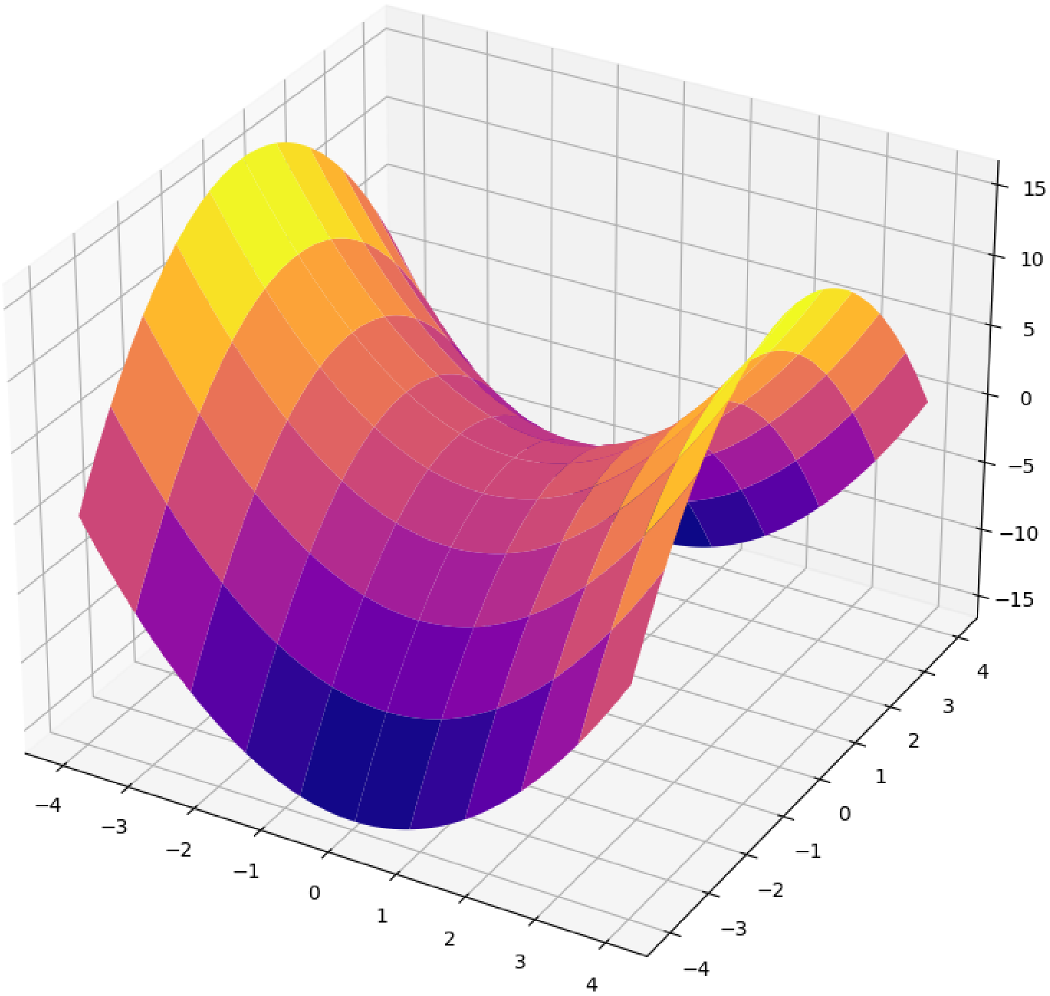
```
fig = plt.figure(figsize=(7,7))
ax = fig.add_subplot(projection='3d')
x = np.linspace(-1,1,100);
y = np.linspace(-1,1,100);
[x,y] = np.meshgrid(x,y);
z = lambda w: w[0]**2 + w[1]**2
Z = z((x,y))

ax.plot_surface(x,y, Z, rstride=4, cstride=4, color='#11aa55')
plt.show()
```



Гиперболический параболоид

Ввод [135]:

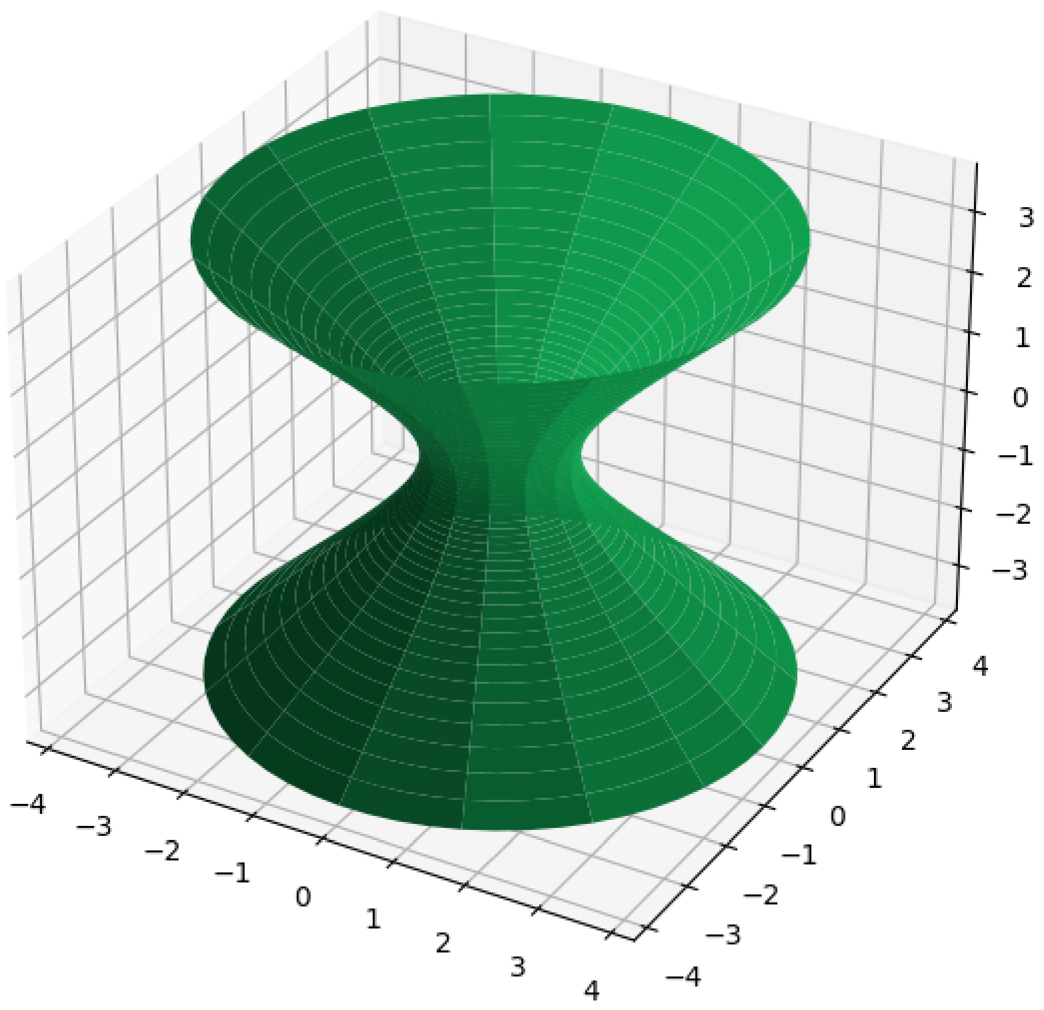
```
f = lambda x, y: x ** 2 - y ** 2

fig = plt.figure(figsize = (10, 10))
ax = fig.add_subplot(1, 1, 1, projection = '3d')
xval = np.linspace(-4, 4, 100)
yval = np.linspace(-4, 4, 100)
x, y = np.meshgrid(xval, yval)
z = f(x, y)

surf = ax.plot_surface(x, y, z, rstride = 10,\
                    cstride = 10,cmap = cm.plasma)

plt.show()
```



Однополосный гиперболоид

Ввод [136]:

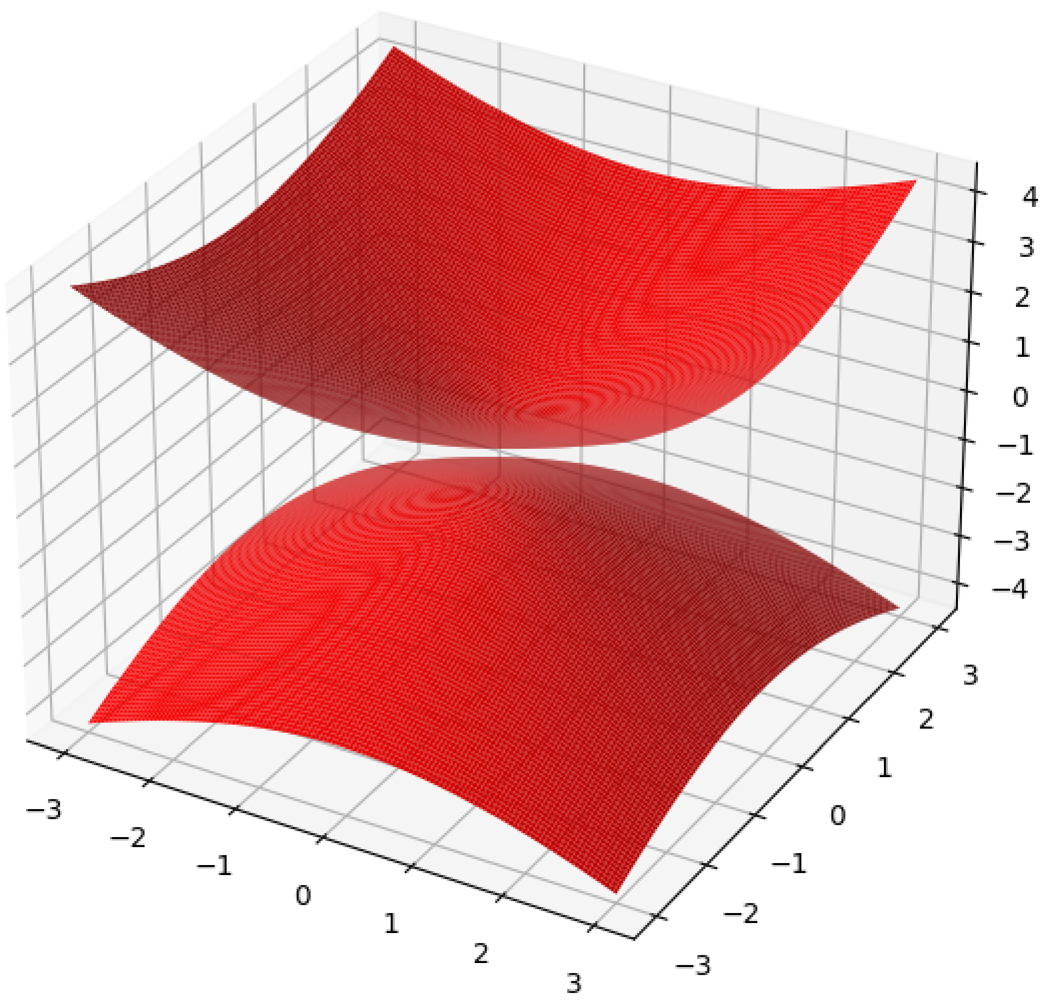
```
fig = plt.figure(figsize=(7,7))
ax = fig.add_subplot(projection='3d')

u=np.linspace(-2,2,200);
v=np.linspace(0,2*np.pi,60);
[u,v]=np.meshgrid(u,v);

x = np.cosh(u)*np.cos(v)
y = np.cosh(u)*np.sin(v)
z = np.sinh(u)

ax.plot_surface(x, y, z, rstride=4, cstride=4, color='#11aa55')

plt.show()
```



Двухполостный гиперболоид

Ввод [137]:

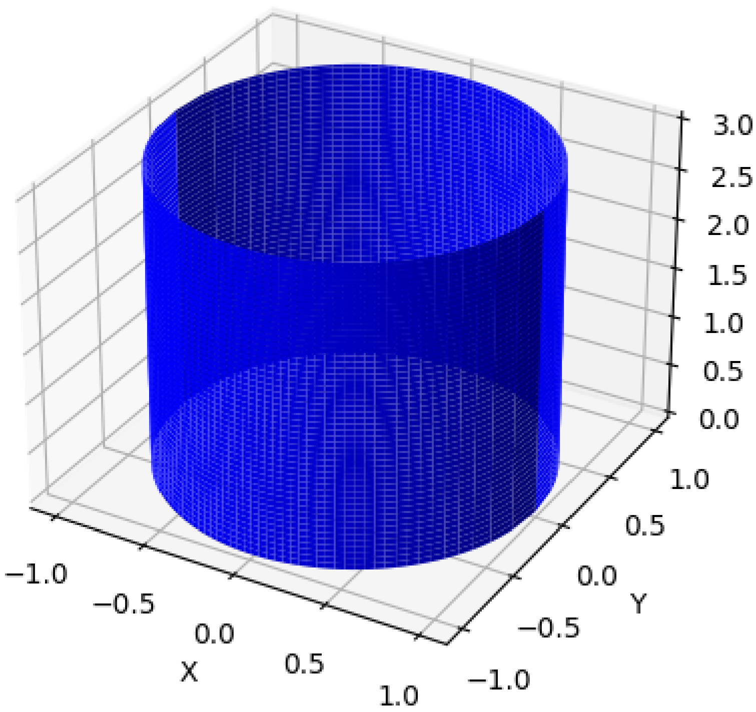
```
fig = plt.figure(figsize=(7,7))
ax = fig.add_subplot(projection='3d')
x = np.linspace(-3,3,500);
y = np.linspace(-3,3,500);
[x,y] = np.meshgrid(x,y);
z1 = lambda w: np.sqrt(w[0]**2 + w[1]**2 + 1)
Z1 = z1((x,y))
z2 = lambda w: -np.sqrt(w[0]**2 + w[1]**2 + 1)
Z2 = z2((x,y))
ax.plot_surface(x, y, Z1, rstride=4, cstride=4, color='r')
ax.plot_surface(x, y, Z2, rstride=4, cstride=4, color='r')
plt.show()
```



Элиптический цилиндр

Ввод [138]:

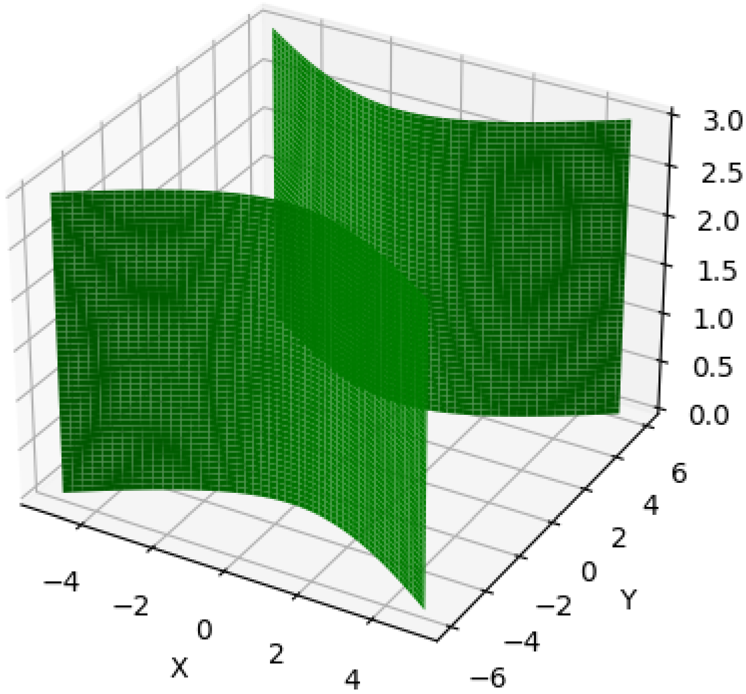
```
fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')

x = np.linspace(-1, 1, 100)
z = np.linspace(0, 3, 100)
x, z = np.meshgrid(x, z)

y = np.sqrt(1-x**2)
ax.plot_surface(x, y, z, color='b')
ax.plot_surface(x, -y, z, color='b')

ax.set_xlabel("X")
ax.set_ylabel("Y")
ax.set_zlabel("Z")

plt.show()
```



Гиперболический цилиндр

Ввод [139]:

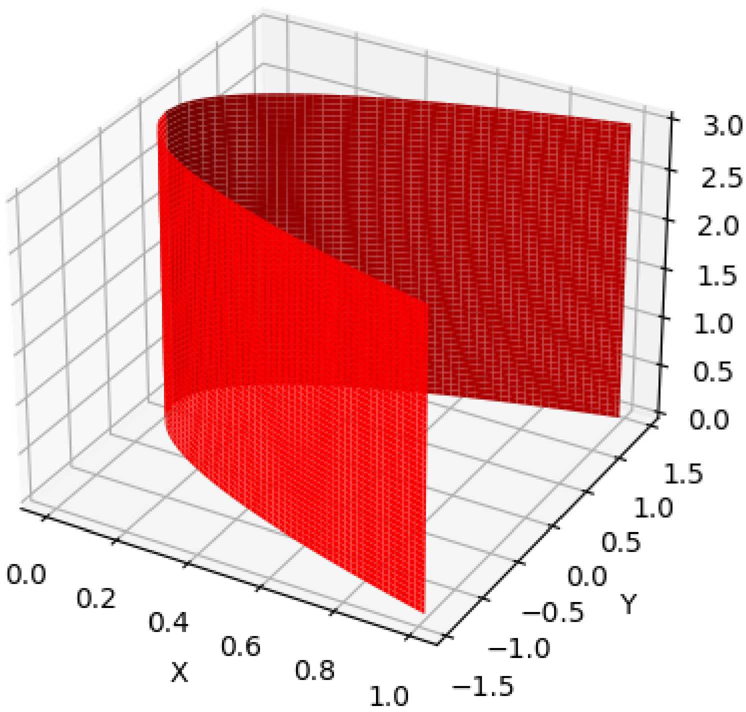
```
fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
x = np.linspace(-5, 5, 100)
z = np.linspace(0, 3, 100)
x, z = np.meshgrid(x, z)
y = np.sqrt(10+x**2)
ax.plot_surface(x, y, z, color='g')
ax.plot_surface(x, -y, z, color='g')
ax.set_xlabel("X")
ax.set_ylabel("Y")
ax.set_zlabel("Z")
plt.show()
```



Параболический цилиндр

Ввод [140]:

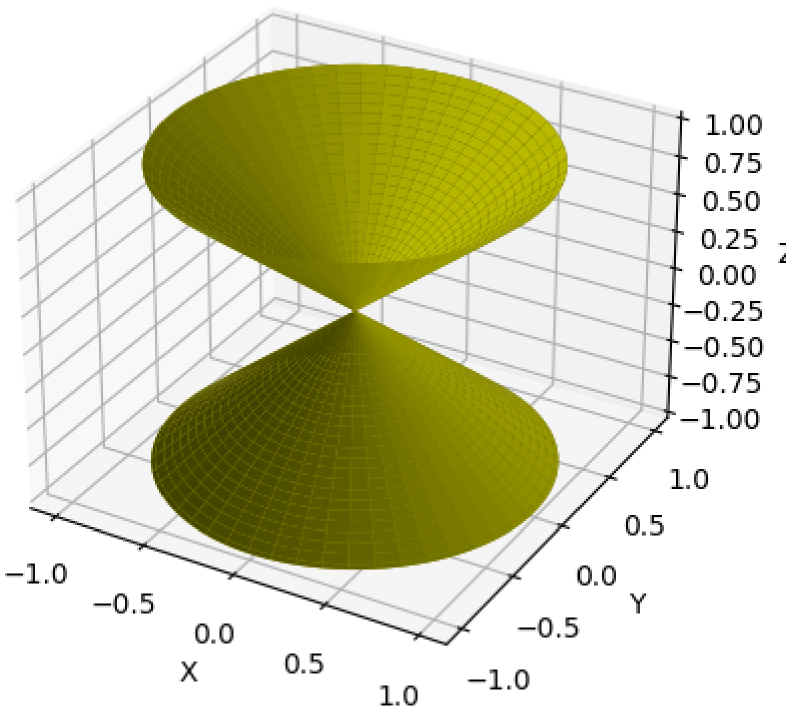
```
fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
x = np.linspace(0, 1, 100)
z = np.linspace(0, 3, 100)
x, z = np.meshgrid(x, z)
y = np.sqrt(2*x)
ax.plot_surface(x, y, z, color='r')
ax.plot_surface(x, -y, z, color='r')
ax.set_xlabel("X")
ax.set_ylabel("Y")
ax.set_zlabel("Z")
plt.show()
```



Конус

Ввод [141]:

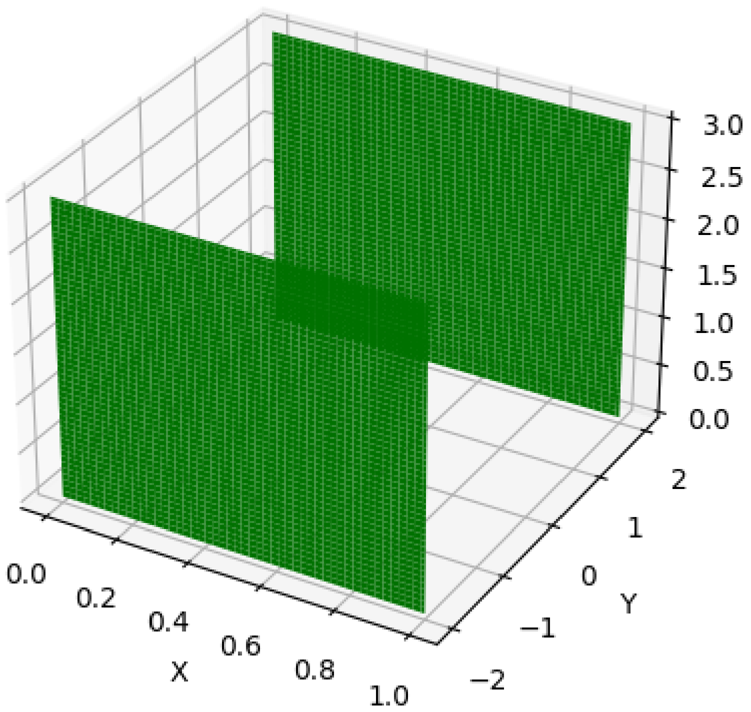
```
fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
theta = np.linspace(0, 2*np.pi, 100)
r = np.linspace(-1, 1, 100)
t, R = np.meshgrid(theta, r)
x = R*np.cos(t)
y = R*np.sin(t)
z1 = R
z2 = -R
ax.plot_surface(x, y, z1, color='y')
ax.plot_surface(x, y, z2, color='y')
ax.set_xlabel("X")
ax.set_ylabel("Y")
ax.set_zlabel("Z")
plt.show()
```



Пара параллельных плоскостей

Ввод [142]:

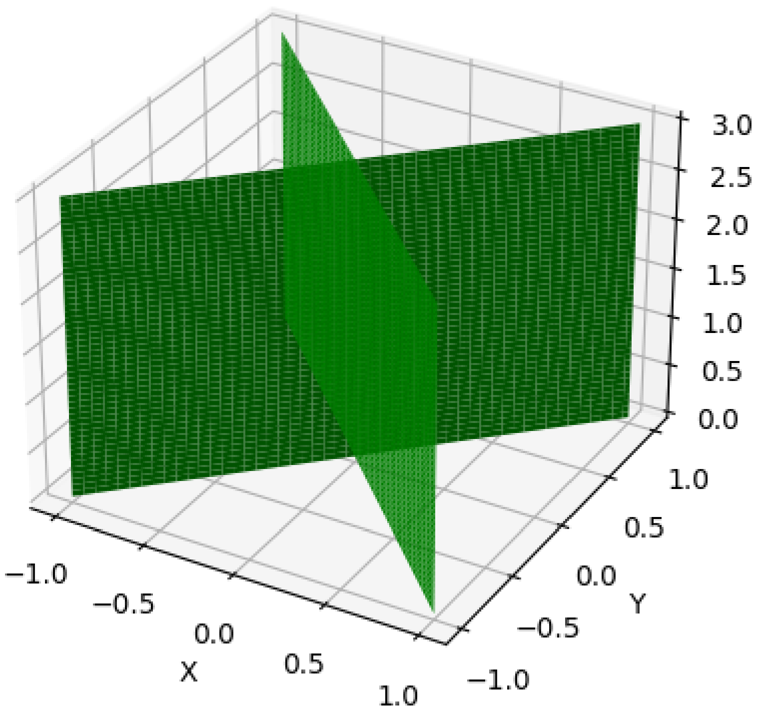
```
fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')

x = np.linspace(0, 1, 100)
z = np.linspace(0, 3, 100)
x, z = np.meshgrid(x, z)

y = 2

ax.plot_surface(x, y, z, color='g')
ax.plot_surface(x, -y, z, color='g')

ax.set_xlabel("X")
ax.set_ylabel("Y")
ax.set_zlabel("Z")

plt.show()
```



Пара пересекающихся плоскостей

Ввод [143]:

```
fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')

x = np.linspace(-1, 1, 100)
z = np.linspace(0, 3, 100)
x, z = np.meshgrid(x, z)

y = x

ax.plot_surface(x, y, z, color='g')
ax.plot_surface(x, -y, z, color='g')

ax.set_xlabel("X")
ax.set_ylabel("Y")
ax.set_zlabel("Z")

plt.show()
```



Примеры решения задач

Пример 1

Даны векторы

$$a = (2, -3, 4, 1), b = (-6, 9, -12, -3), p = (3, 2, -1, 4).$$

Вычислить:

$$x = 2(a, b) \cdot p + 3b(p, p) - |b| \cdot b.$$

Ввод [144]:

```
a = np.array([2,-3,4,1])
b = np.array([-6,9,-12,-3])
p = np.array([3,2,-1,4])
x = 2 * np.dot(a,b) * p + 3 * b * np.dot(p,p) - norm(b) * b
x
```

Out[144]:

```
array([-981.40993965,  302.11490947, -702.8198793 , -940.70496982])
```

Пример 2

Определить, какой угол между векторами

$$a = (2, -3, 4, 1), b = (-6, 9, -12, -3) :$$

острый, тупой или прямой.

Ввод [145]:

```
np.dot(a,b)
```

Out[145]:

```
-90
```

Пример 3

Являются ли ортогональными векторы

$$a = (2, 0, -3), b = (6, 1, 4)?$$

Ввод [146]:

```
a = (2, 0, -3)
b = (6, 1, 4)
np.dot(a,b)
```

Out[146]:

```
0
```

Пример 4

Прямую l, заданную общим уравнением

$$4x - 5y + 20 = 0,$$

записать в параметрическом виде.

Ввод [147]:

```
s = Line((2,8), (-3,4))
s.equation()
```

Out[147]:

$4x - 5y + 32$

Пример 5

Найти направляющий вектор прямой, заданной уравнением

$$4x - 7y - 14 = 0.$$

Ввод [148]:

```
l = Line((0,4), (-5,0))
l.arbitrary_point()
```

Out[148]:

$\mathrm{Point2D}(-5t, 4 - 4t)$

Пример 6

Найти направляющий вектор прямой, заданной уравнением

$$4x - 7y - 14 = 0$$

Ввод [149]:

```
s = Line((3.5,0), (0,-2))
s.direction
```

Out[149]:

$\mathrm{Point2D}\left(-\frac{7}{2}, -2\right)$

Пример 7

Прямую s, представленную общим уравнением:

$$5x - 2y + 10 = 0$$

Ввод [150]:

```
s = Line((-2,0), (0,5))
p = s.direction
p
```

Out[150]:

$\mathrm{Plane}(\mathrm{Point3D}(3, 0, 0), (9,\ 9,\ 9))$

Пример 8

Прямую s, представленную общим уравнением:

$$x + y - 2 = 0,$$

записать в параметрическом виде.

Ввод [151]:

```
s = Line((2,0), (0,2))
s.arbitrary_point()
```

Out[151]:

$\mathrm{Point2D}(2 - 2t, 2t)$

Пример 9

Вычислить расстояние от точки $\mathrm{M}(5; 4)$ до прямой, проходящей через точки $A(1; -2)$ и $B(0; 3)$.

Ввод [152]:

```
M = Point(5,4)
s = Line((1,-2), (0,3))
s.distance(M)
```

Out[152]:

$\displaystyle \sqrt{26}$

Пример 10

Написать уравнение прямой, проходящей через точку M и параллельной прямой l, если $M(-2; 1), l : 3x - 2y + 12 = 0$.

Ввод [153]:

```
M = Point(-2,1)
l = Line((0,6), (-4,0))
l1 = l.parallel_line(M)
l1.equation()
```

Out[153]:

$\displaystyle 6 x - 4 y - 14$

Пример 11

Написать уравнение прямой, проходящей через точку M и перпендикулярной прямой l, если $M(3; -3), l : x + 2y - 4 = 0$.

Ввод [154]:

```
M = Point(3,-3)
l = Line((0,2), (4,0))
l1 = l.perpendicular_line(M)
l1.equation()
```

Out[154]:

$\displaystyle - 4 x + 2 y + 18$

Пример 12

Найти точку пересечения прямых

$$s_1\ : 2x - 3y + 12 = 0 \text{ и } s_2\ : x + y - 2 = 0.$$

Ввод [155]:

```
s1 = Line((0,4), (-6,0))
s2 = Line((0,2), (2,0))
s1.intersection(s2)
```

Out[155]:

```
[Point2D(-6/5, 16/5)]
```

Пример 13

Написать уравнение прямой, проходящей через точку (M) и точку пересечения прямых l_1 и l_2, если $M(2;0), l_1\ : 2x - y - 1 = 0, l_2\ : x + 3y - 4 = 0$

Ввод [156]:

```
M = Point(2,0)
l1 = Line((0,-1), (0.5,0))
l2 = Line((1,1), (4,0))
A = l1.intersection(l2)
s = Line(M,A[0])
s.equation()
```

Out[156]:

$\displaystyle - x - y + 2$

Пример 14

Найти расстояние от точки $P(2;2)$ до прямой l, заданной в каноническом виде: $\frac{x-2}{3} = \frac{y+1}{2}$.

Ввод [157]:

```
P = Point(2,2)
l = Line((2,-1), (5,1))
l.distance(P)
```

Out[157]:

$\displaystyle \frac{9 \sqrt{13}}{13}$

Пример 15

Найти расстояние от точки $P(-2;2)$ до прямой l, записанной в параметрической форме:

$$x = 2t - 3, y = t + 2.$$

Ввод [158]:

```
p = Point(-2,2)
l = Line((-3,2), (-1,3))
l. distance(P)
```

Out[158]:

$\displaystyle \sqrt{5}$

Пример 16

Найти угол пересечения прямой $x - 2y + 4 = 0$ с осью Ox.

Ввод [159]:

```
l1 = Line((0,2), (-4,0))
l2 = Line((0,0), (1,0))
l1. smallest_angle_between(l2)
```

Out[159]:

$\displaystyle \operatorname{acos}{\left(\frac{2 \sqrt{5}}{5} \right)}$

Пример 17

Найти координаты основания перпендикуляра, опущенного из начала координат на прямую l, заданную уравнением:

$$x - y - 17 = 0$$

Ввод [160]:

```
l = Line((17,0), (0,-17))
s = l.perpendicular_line((0,0))
l.intersection(s)
```

Out[160]:

```
[Point2D(17/2, -17/2)]
```

Пример 18

Найти расстояние между параллельными прямыми l_1, l_2, заданными уравнениями $3x - 4y - 2 = 0$ и $3x - 4y + 1 = 0$.

Ввод [161]:

```
l1 = Line((2,1), (6,4))
l2 = Line((1,1), (5,4))
l2.distance((2,1))
```

Out[161]:

$\displaystyle \frac{3}{5}$

Пример 19

Дан куб $ABDEA_1B_1D_1E_1$ со стороной, равной 1. Найти угол между диагоналями AD_1 и B_1E.

Ввод [162]:

```
A = Point(0,0,0)
D1 = Point(1,1,1)
B1 = Point(1,1,0)
E = Point(0,1,0)
AD1 = Line(A,D1)
B1E = Line(B1,E)
AD1.angle_between(B1E)
```

Out[162]:

$\displaystyle \operatorname{acos}{\left(- \frac{\sqrt{3}}{3} \right)}$

Пример 20

Найти проекцию точки $A(-1; 1)$ на прямую $s : 2x + 3y = 6$.

Ввод [163]:

```
A = Point(-1,1)
s = Line((3,0), (0,2))
s.projection(A)
```

Out[163]:

$\displaystyle \operatorname{Point2D}\left(- \frac{3}{13}, \frac{28}{13}\right)$

Пример 21

Написать уравнение медианы и высоты, проведенных из вершины A треугольника ABC, если заданы вершины:

$$A(-1; -5), B(3; -1), C(1; -2)$$

Ввод [164]:

```
A, B, C = Point(-3, -2), Point(0, 4), Point(6, 0)
M = B.midpoint(C)
s = Line(A, M)
s.equation()
```

Out[164]:

$\displaystyle - 4 x + 6 y$

Ввод [165]:

```
s = Line(B, C)
l2 = s.perpendicular_line(A)
l2
```

Out[165]:

$\displaystyle \operatorname{Line2D}\left(\operatorname{Point2D}\left(-3, -2\right), \operatorname{Point2D}\left(1, 4\right)\right)$

Ввод [166]:

```
l2.equation()
```

Out[166]:

$\displaystyle - 6 x + 4 y - 10$

Пример 22

Найти уравнение прямой l, являющейся пересечением плоскостей p_1 и p_2, если p_1 проходит через точки $(0; 1; 2)$, $(2; 1; 3)$ и $2; -2; 5)$, а p2 проходит через точки $(-1; 3; -2)$, $(4; 0; 1)$ и $(5; 1; 0)$.

Ввод [167]:

```
p1 = Plane((0,1,2), (2,1,3), (2,-2,5))
p2 = Plane((-1,3,-2), (4,0,1), (5,1,0))
l = p1.intersection(p2)
l
```

Out[167]:

```
[Line3D(Point3D(-4, 1, 0), Point3D(12, -23, 24))]
```

Ввод [168]:

```
s = l[0]
s.equation()
```

Out[168]:

$\displaystyle \left(3 x + 2 y + 10, \ - 3 x + 2 z - 12\right)$

Ввод [169]:

```
p1.equation()
```

Out[169]:

$\displaystyle - x^{2} - 16 y + 64$

Ввод [170]:

```
p2.equation()
```

Out[170]:

$\displaystyle 8 y + 8 z - 8$

Пример 23

Выяснить характер расположения прямых AB и CD (пересекаются, параллельны или скрещиваются), где $A(1;1;1)$, $B(3;-2;0)$, $C(1;0;1)$, $D(2;1;0)$

Ввод [171]:

```
s1 = Line3D((1,1,1), (3,-2,0))
S2 = Line3D((1,0,1), (2,1,0))
Line.are_concurrent(s1, s2)
```

Out[171]:

```
False
```

Ввод [172]:

```
Line.is_parallel(s1, s2)
```

Out[172]:

```
False
```

Ввод [173]:

```
s1.is_similar(s2)
```

Out[173]:

```
False
```

Пример 24

Доказать, что три медианы треугольника пересекаются в одной точке.

Ввод [174]:

```
a1, a2, b1, b2, d1, d2 = symbols('al a2 y b2 dl d2')

A = Point(a1,a2)
B = Point(b1,b2)
D = Point(d1,d2)

M = B.midpoint(D)
N = D.midpoint(A)
K = A.midpoint(B)
```

Ввод [175]:

```
AM = Line(A, M)
BN = Line(B, N)
DK = Line(D, K)

Line.are_concurrent(AM, BN, DK)
```

Out[175]:

```
True
```

Пример 25

Определить тип кривой второго порядка

$$13x^2 + 18xy + 37y^2 - 26x - 18y - 27 = 0$$

Ввод [176]:

```
A = Matrix([[13,9],
            [9, 37]])
a = Matrix([-13,-9,-27])
conic_curve(A,a,f_transform=1)
```

```
ellipse
Equation: 10*x1**2 + 6*sqrt(10)*x1 + 40*y1**2 - 8*sqrt(10)*y1 - 27
transition equation:
x = -3*sqrt(10)*x1/10 + sqrt(10)*y1/10
y = sqrt(10)*x1/10 + 3*sqrt(10)*y1/10
```

Индивидуальное задание

Анализ оптимальной траектории движения транспортного средства по криволинейной траектории

Описание задачи: Определить оптимальную траекторию движения автомобиля по криволинейной траектории и проанализировать соответствующие параметры.

Исходные условия:
Длина пути: 200 метров
Радиус поворота: 10 метров
Масса автомобиля: 975 кг
Колесная база: 2,46 метра
Распределение веса: 60% спереди, 40% сзади
Максимальный коэффициент сцепления шин с дорогой: 1,2
Максимальное боковое усилие на шинах: 12000 Н
Ширина дороги: 5 метров

Задачи:

1. Рассчитать максимальную боковую силу шин.
2. Вычислить скорость входа в поворот.
3. Определить положение апекса и радиус поворота.
4. Рассчитать скорость выхода из поворота.
5. Сгенерировать точки траектории вдоль кривой.
6. Построить кривую траектории, а также левую и правую границы трассы.

Заданные параметры пути и транспортного средства

Ввод [177]:

```
track_length = 200  # метров
turn_radius = 10  # метров
mass = 975  # кг
wheelbase = 2.46  # метров
weight_distribution = 0.60  # распределение веса между передней и задней частями тела
mu = 1.2  # максимальный коэффициент сцепления шины с дорогой
f_max = 12000  # максимальная боковая сила шины (Н)
road_width = 5  # метров
```

Рассчет максимальной боковой силы шины

Ввод [178]:

```
f_lat_max = f_max * weight_distribution
f_lat_max
```

Out[178]:

```
7200.0
```

Расчет скорости входа в поворот

Ввод [179]:

```
entry_speed = sqrt(f_lat_max * turn_radius / (mass * (1 - weight_distribution)))
entry_speed
```

Out[179]:

$\displaystyle 13.5873244097351$

Вычисление положение вершины и радиус апекса

Ввод [180]:

```
apex_radius = turn_radius / sqrt(1 - weight_distribution)
apex_radius
```

Out[180]:

$\displaystyle 15.8113883008419$

Ввод [181]:

```
apex_position = (track_length / 2) - apex_radius
apex_position
```

Out[181]:

$\displaystyle 84.1886116991581$

Расчет скорости выхода из поворота

Ввод [182]:

```
exit_speed = sqrt((2 * f_lat_max * apex_radius) / (mass * (1 - weight_distribution)))
exit_speed
```

Out[182]:

$\displaystyle 24.1620592353513$

Генерирация точек траектории

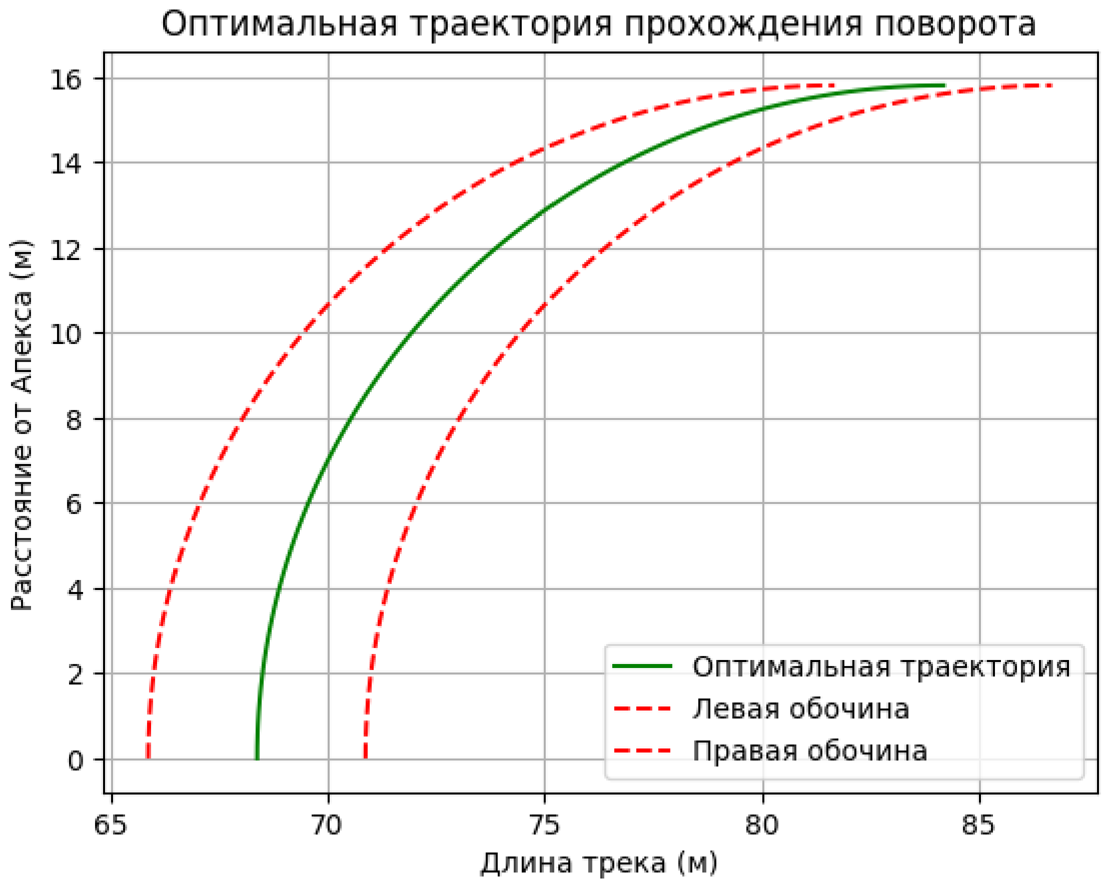
Ввод [183]:

```
theta = rad(180)  # Начальный угол (180 градусов)
delta_theta = rad(1)  # Приращение угла
trajectory_points = []
left_border_points = []
right_border_points = []

while theta >= rad(90):
    x = apex_position + apex_radius * cos(theta)
    y = apex_radius * sin(theta)
    trajectory_points.append((x, y))
    left_border_points.append((x - (road_width / 2), y))
    right_border_points.append((x + (road_width / 2), y))
    theta -= delta_theta
```

Построение кривой траектории и обочины

Ввод [184]:

```
x_vals, y_vals = zip(*trajectory_points)
left_x_vals, left_y_vals = zip(*left_border_points)
right_x_vals, right_y_vals = zip(*right_border_points)

plt.plot(x_vals, y_vals, label='Оптимальная траектория', color="green")
plt.plot(left_x_vals, left_y_vals, '--', label='Левая обочина', color="red")
plt.plot(right_x_vals, right_y_vals, '--', label='Правая обочина', color="red")
plt.xlabel('Длина трека (м)')
plt.ylabel('Расстояние от Апекса (м)')
plt.title('Оптимальная траектория прохождения поворота')
plt.grid(True)
plt.legend()
plt.show()
```



Вывед рассчитанных значений

Ввод [185]:

```
print(f"Максимальное боковое усилие на шинах: {f_lat_max:.2f} Н")
print(f"Скорость входа: {entry_speed.evalf() * 3.6:.2f} км/ч")
print(f"Апекс-позиция: {apex_position:.2f} м")
print(f"Скорость выхода: {exit_speed.evalf() * 3.6:.2f} км/ч")
```

```
Максимальное боковое усилие на шинах: 7200.00 Н
Скорость входа: 48.91 км/ч
Апекс-позиция: 84.19 м
Скорость выхода: 86.98 км/ч
```

Ввод [ ]:

```

```