# Лабораторный практикум 8. Поверхности второго порядка

Загрузите необходимые библиотеки:

import numpy as np

import matplotlib.pyplot as plt

from mpl\_toolkits import mplot3d

from sympy import \*

%matplotlib widget

## Построение поверхностей второго порядка

Метод построения трехмерных графиков описан в практикуме 5: для начала нужно подготовить прямоугольную сетку на плоскости *Оxy*, в узлах которой будут рассчитываться значения отображаемой функции (значение по оси *Z*). Для создания такой сетки можно воспользоваться функцией *np.meshgrid*().

В курсе аналитической геометрии использовался метод сечений, сущность которого состоит в следующем. Пусть поверхность задана в прямоугольной системе координат уравнением . Поверхность пересекаем плоскостями, параллельными координатным плоскостям, и находим линии пересечения поверхности с этими плоскостями. По виду этих линий и выносится суждение о форме поверхности. В Python 3D-графики поверхностей, построенные с помощью Matplotlib, можно проецировать на 2D-поверхности. В результате получается иллюстрация метода сечений.

**Для построения графика непрерывной поверхности используется *ax*.**plt\_surface()**, каркасной поверхности – *ax.plt\_wireframe*()**, контурного **графика – *ax.contourf*()**.

**Пример 1.** Построить эллиптический параболоид . Так как переменная *z* явно выражена через *x* и *y*, то будем строить эту поверхность с помощью команды *meshgrid*. Приведенный ниже код создает 3D-график и визуализирует его проекции на 2D-контурные графики:

x = np.linspace(-1,1,200)

y = np.linspace(-10,1,200)

[x,y] = np.meshgrid(x,y)

z = lambda x,y: x\*\*2 + 2\*y\*\*2 # Уравнение параболоида

Z = z(x,y)

fig = plt.figure(figsize =(10, 10))

ax = plt.axes(projection ='3d')

my\_cmap = plt.get\_cmap('magma') # Создание цветовой карты

# Создание графиков

surf = ax.plot\_surface(x,y,Z,rstride = 8,cstride = 8,alpha = 0.8,cmap = my\_cmap)

cset = ax.contourf(x,y,Z,zdir ='z',offset = np.min(Z),cmap=my\_cmap)

cset = ax.contourf(x,y,Z,zdir ='x',offset =-2,cmap = my\_cmap)

cset = ax.contourf(x,y,Z,zdir ='y',offset = 2,cmap = my\_cmap)

fig.colorbar(surf,ax = ax,shrink = 0.5,aspect = 5)

ax.set\_xlabel('X-axis')

ax.set\_xlim(-2, 2)

ax.set\_ylabel('Y-axis')

ax.set\_ylim(-2, 2)

ax.set\_zlabel('Z-axis')

ax.set\_zlim(np.min(Z), np.max(Z))

ax.set\_title('График эллиптического параболоида $x^2+2y^2=z$ с проекциями')

plt.show()

На графике хорошо видно, что проекции, параллельные координатным плоскостям *Oxz* и *Oyz* – это параболы, а параллельные плоскости *Oxy* – эллипсы. Уравнения этих проекций соответственно: , , .

Наряду с цветовой картой *magma* существуют и другие цветовые карты, такие как *viridis*, *plasma*, *rainbow*, *gnuplot* и т.д. Полный список цветовых карт есть на сайте Matplotlib.

**Упражнение 8.1.** Построить поверхность . Написать уравнения проекций данной поверхности на координатные плоскости.

Во некоторых случаях, когда ни одна из переменных в уравнении поверхности не выражена в явном виде, удобно использовать её параметрические уравнения.

**Пример 2.** Построить эллипсоид .

Будем использовать его параметрические уравнения ():

Проверьте, подставив *x*, *y*, *z* в каноническое уравнение эллипсоида. Параметр *φ* – угол на плоскости, параметр *θ* регулирует высоту фигуры вдоль оси *Oz*.

Данные координаты называются ***эллиптическими координатами***.

fig = plt.figure(figsize =(10, 10))

ax = plt.axes(projection ='3d')

#Коэффициенты уравнения

coefs = (9, 1, 1)

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

u = np.linspace(0, 2\*np.pi, 200) # Угол на плоскости

v = np.linspace(-np.pi/2, np.pi/2, 200) # Угол в пространстве

#Уравнение эллипсоида

x = rx \* np.outer(np.cos(u), np.cos(v))

y = ry \* np.outer(np.sin(u), np.cos(v))

z = rz \* np.outer(np.ones\_like(u), np.sin(v))

my\_cmap = plt.get\_cmap('rainbow')

surf = ax.plot\_surface(x,y,z,rstride = 10,cstride = 10,alpha = 0.6,cmap = my\_cmap)

cset = ax.contourf(x,y,z,zdir ='z',offset = -2,cmap = my\_cmap)

cset = ax.contourf(x,y,z,zdir ='x',offset =-4,cmap = my\_cmap)

cset = ax.contourf(x,y,z,zdir ='y',offset = 2,cmap = my\_cmap)

ax.set\_xlabel('X-axis'); ax.set\_xlim(-4, 4)

ax.set\_ylabel('Y-axis'); ax.set\_ylim(-2, 2)

ax.set\_zlabel('Z-axis'); ax.set\_zlim(-2, 2)

ax.set\_title('График эллипсоида $x^2/9+y^2+z^2=1$ с проекциями')

plt.show()

**С помощью команды *help* выясните, что делает функция *outer*()!Это Векторное умножение**

**Упражнение 8.2.** Построить сферу, используя ее параметрические уравнения:

(Проверьте, подставив *x*, *y*, *z* в каноническое уравнение сферы).

Данные координаты называются ***сферическими координатами***.

**Упражнение 8.3.** Построить однополостный гиперболоид, используя его параметрические уравнения:

(Проверьте, подставив *x*, *y*, *z* в каноническое уравнение ). Коэффициенты выбрать самостоятельно. Написать уравнения проекций данной поверхности на координатные плоскости.

**Пример 3.** Построить эллиптический цилиндр .

Для построения можно использовать как декартовы, так и ***цилиндрические координаты***:

Построим каркасную поверхность для обоих случаев. В декартовых координатах:

fig = plt.figure(figsize =(7, 7))

ax = plt.axes(projection ='3d')

x = np.linspace(-1, 1, 30)

z = np.linspace(0, 3, 30)

x, z = np.meshgrid(x, z)

y = np.sqrt(1-x\*\*2)

ax.plot\_wireframe(x, y, z,color='b')

ax.plot\_wireframe(x, -y, z, color='b')

ax.set\_title('Эллиптический цилиндр $x^2+y^2=1$')

plt.show()

В цилиндрических координатах:

fig = plt.figure(figsize =(7, 7))

ax = plt.axes(projection ='3d')

phi = np.linspace(0, 2\*np.pi, 30)

H=np.linspace(0, 3, 30)

R = 1 # радиус цилиндра

x = R\*np.outer(np.cos(phi),np.ones(len(H)))

y = R\*np.outer(np.sin(phi),np.ones(len(H)))

z = np.outer(np.ones(len(phi)),H)

surf = ax.plot\_wireframe(x,y,z,color='r')

ax.set\_title('Эллиптический цилиндр в цилиндрических координатах')

plt.show()

**Упражнение 8.4.** Построить параболический цилиндр . Изобразить проекцию данной поверхности на плоскость *Oxz* (используйте функцию *ax.view\_init*(0, ±90)). Найти координаты вершины данной проекции?

**Пример 4.** Найти точки пересечения поверхности и прямой, проходящей через точки (0,0, –2) и (4, –3,2). Сделать рисунок. Нанести на него найденные точки.

Найдем параметрические уравнения прямой, подставим их в уравнение цилиндра и определим значения параметра *t*. Для нахождения общих точек подставим эти значения обратно в параметрические уравнения прямой.

# уравнение прямой, проходящей через две точки р1 и р2

p1, р2 = Point(0,0,-2), Point(4,-3,2)

L = Line(p1, р2)

L=L.arbitrary\_point()

print('Параметрические уравнения прямой:',L)

x,y,t=symbols('x,y,t')

P=x\*\*2+y\*\*2-1

PP=P.subs([(x,L[0]),(y,L[1])])

print(PP) # уравнение относительно параметра t

X=solve(PP)

print('Значения параметра t=',X)

t=X[0]

M1=[4\*t, -3\*t, 4\*t - 2]

t=X[1]

M2=[4\*t, -3\*t, 4\*t - 2]

print('M1 =',M1)

print('M2 =',M2)

Сделаем рисунок и посмотрим на проекцию на плоскость *Oxz*:

# Построение цилиндра

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

z = np.linspace(-3, 0, 5)

x, z = np.meshgrid(x, z)

y = np.sqrt(1-x\*\*2)

fig = plt.figure(figsize =(12, 7))

ax = plt.axes(projection ='3d')

ax.plot\_wireframe(x, y, z, color='b')

ax.plot\_wireframe(x, -y, z, color='b')

ax.set\_xlabel('X')

ax.set\_ylabel('Y')

ax.set\_zlabel('Z')

# Построение прямой по двум точкам (при t=-1/2 и t=1/2)

x1 = np.array([2, -2])

y1 = np.array([-3/2, 3/2])

z1 = np.array([0, -4])

ax.plot(x1, y1, z1,color='r')

ax.plot(-4/5, 3/5, -14/5,'or',markersize=12,label='точка M1')

ax.plot(4/5, -3/5, -6/5,'ok',markersize=12,label='точка M2')

ax.legend()

ax.view\_init(90, 0)

plt.show()

По рисунку видно, что найденные точки *М*1 и *М*2 являются точками пересечения цилиндра и прямой.

**Упражнение 8.5.** Найти точки пересечения поверхности и прямой, проходящей через точки (8, –1, –3) и (–4,3,3). Сделать рисунок. Нанести на него найденные точки.

## Дополнительное задание

Постройте поверхности и придумайте им названия: