Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт перспективной инженерии
Департамент цифровых, робототехнических систем и электроники

ОТЧЕТ
ПО ПРАКТИЧЕСКОЙ РАБОТЕ №3
дисциплины
«Искусственный интеллект и машинное обучение»
Вариант 15

Выполнил:
Степанов Артем Сергеевич
2 курс, группа ИВТ-б-о-23-2,
09.03.01 «Информатика и вычислительная техника»,
направленность (профиль)
«Программное обеспечение средств вычислительной техники и автоматизированных систем», очная форма обучения

_____
(подпись)

Проверил:
Доцент департамента цифровых, робототехнических систем и электроники института перспективной инженерии
Воронкин Роман Александрович

_____
(подпись)

Отчет защищен с оценкой _____ Дата защиты_____

Ставрополь, 2025 г

Тема: Основы работы с библиотекой matplotlib

Цель: исследовать базовые возможности библиотеки matplotlib языка программирования Python

1. Выполнил задания из практической работы.

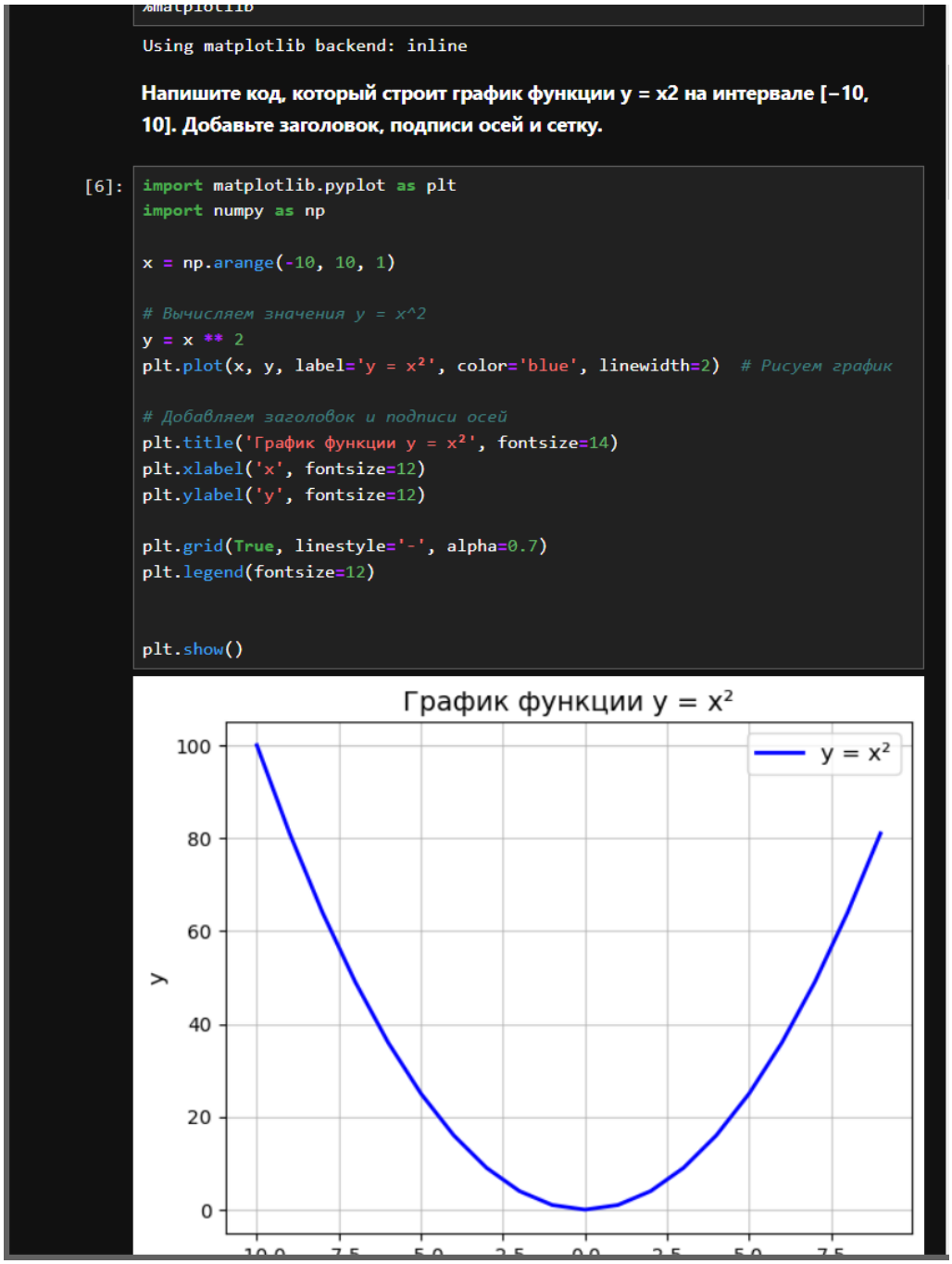

Рисунок 1 – Решение задания №1

Постройте три линии на одном графике: y = x (синяя, пунктирная линия), y = x2 (зеленая, штрих-пунктирная линия), y = x3 (красная, сплошная линия). Добавьте легенду и сделайте оси одинакового масштаба.

[28]:
```
import matplotlib.pyplot as plt
import numpy as np

x = np.linspace(-10, 10, 10)
y1 = x
y2 = x * 2
y3 = x * 3  # Это не ошибка, сделал намеренно, иначе ничего нормально не было

plt.plot(x, y1, 'b--', label='y = x', linewidth=2)
plt.plot(x, y2, 'g-.', label='y = x²', linewidth=2)
plt.plot(x, y3, 'r-', label='y = x³', linewidth=2)

plt.title('Графики функций')
plt.xlabel('x')
plt.ylabel('y')
plt.grid(True, linestyle=':', alpha=1)
plt.legend()
plt.axis('equal')

plt.show()
```

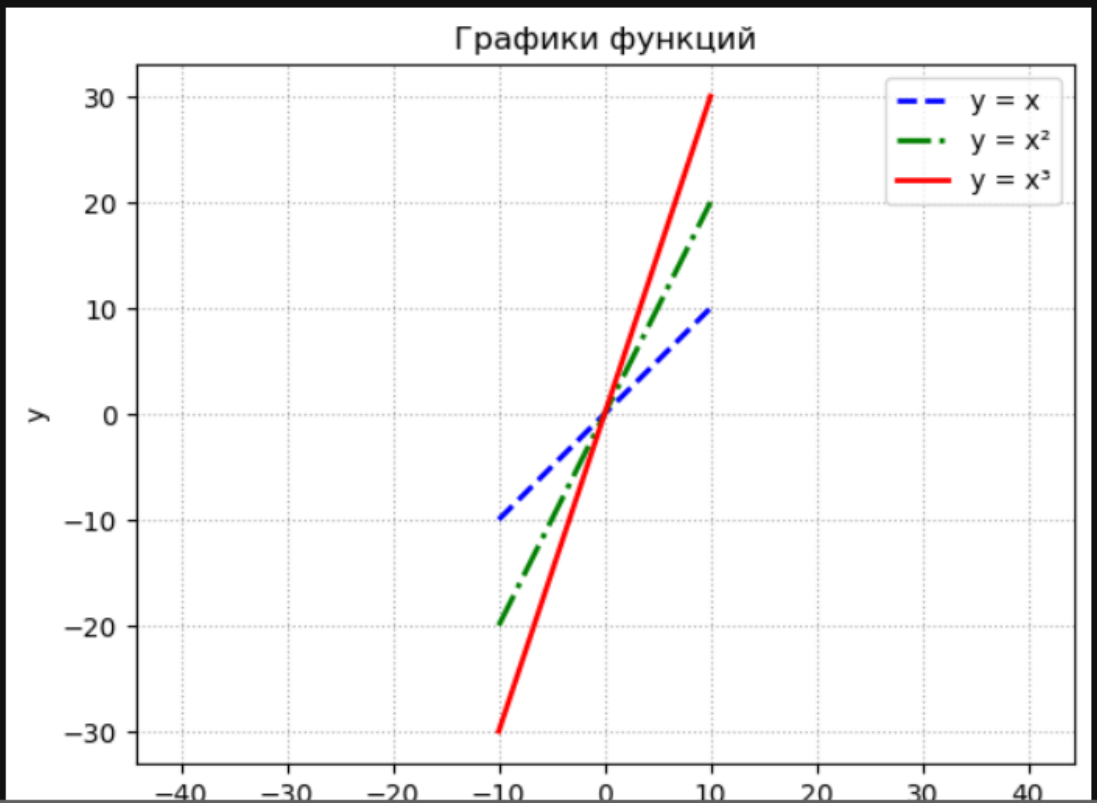

Рисунок 2 – Решение задания №2

Сгенерируйте 50 случайных точек и постройте диаграмму рассеяния (scatter plot), где цвет точек зависит от их координаты по оси x, а размер точек зависит от координаты по оси y

[37]:

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

np.random.seed(42)
x = np.random.uniform(0, 10, 50)
y = np.random.uniform(0, 10, 50)

colors = x
sizes = y * 10  # Увеличил немного размеры

# Создаем scatter plot

scatter = plt.scatter(x, y, c=colors, s=sizes, alpha=0.7, cmap='viridis')


cbar = plt.colorbar(scatter) # Тут создана шкала с правой стороны, мне понрав
cbar.set_label('Значение X (цвет)')

# Подписи и оформление
plt.title('Диаграмма рассеяния: цвет=X, размер=Y', fontsize=14)
plt.xlabel('Координата X', fontsize=12)
plt.ylabel('Координата Y', fontsize=12)
plt.grid(True, linestyle='--', alpha=0.5)

plt.show()
```

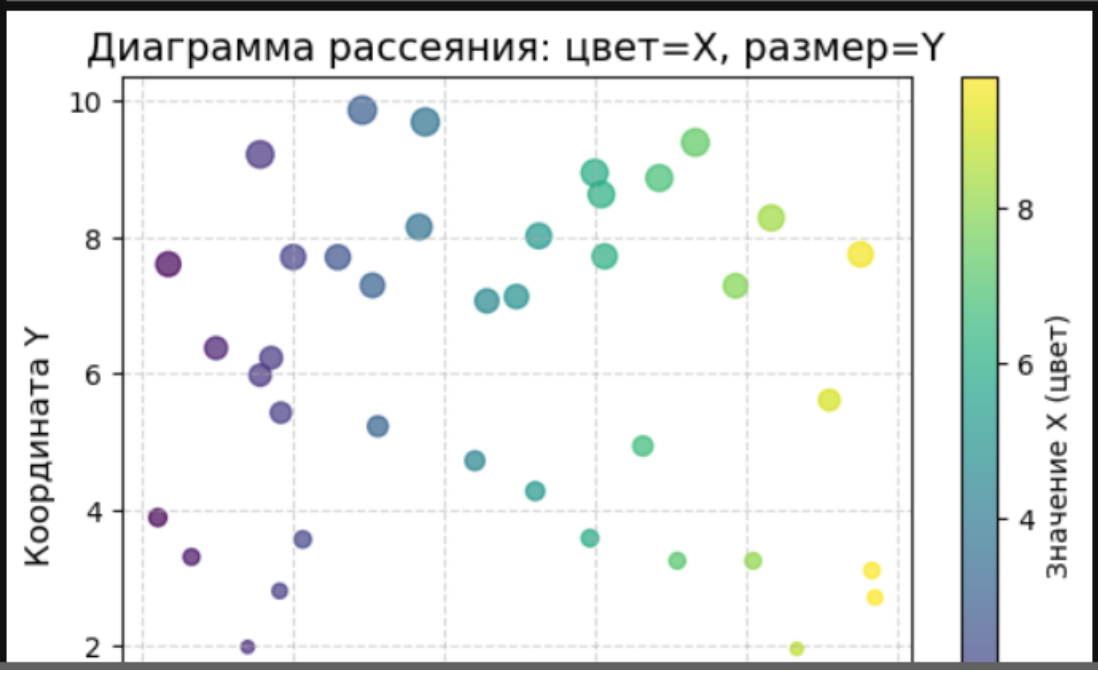

Рисунок 3 – Решение задания №3

Сгенерируйте 1000 случайных чисел из нормального распределения с параметрами μ = 0, σ = 1 и постройте их гистограмму с 30 бинами. Добавьте вертикальную линию в среднем значении.

[52]:
```
import numpy as np
import matplotlib.pyplot as plt


np.random.seed(42)
data = np.random.normal(loc=0, scale=1, size=1000)

# Построение гистограммы
plt.hist(data, bins=30, alpha=0.7, color='steelblue', edgecolor='white')

# Добавление вертикальной линии среднего
mean = np.mean(data)
plt.axvline(mean, color='red', linestyle='--', linewidth=2, label=f'Среднее')


plt.xlabel('Значение', fontsize=12)
plt.ylabel('Плотность вероятности', fontsize=12)
plt.legend(fontsize=12)

plt.show()
```

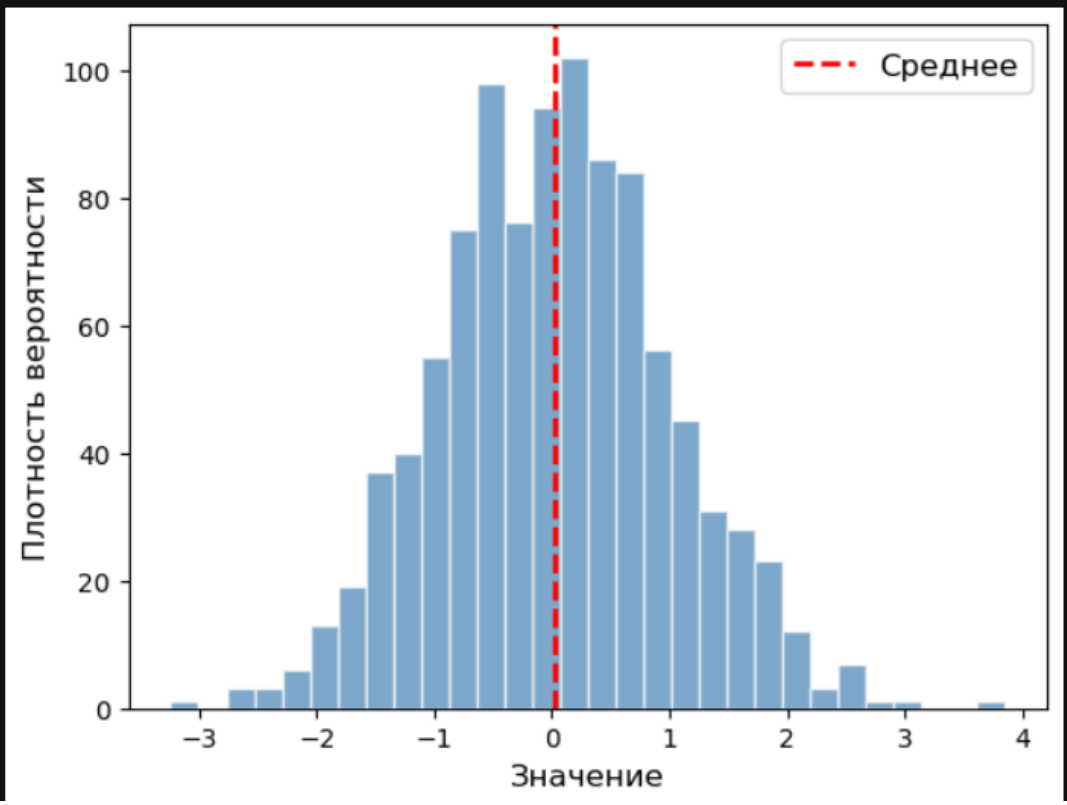

Рисунок 4 – Решение задания №4

Создайте столбчатую диаграмму, которая показывает количество студентов, получивших оценки: "Отлично" — 20 человек, "Хорошо" — 35 человек, "Удовлетворительно" — 30 человек, "Неудовлетворительно" — 15 человек. Добавьте подписи к осям и заголовок.

[55]:
```python
import matplotlib.pyplot as plt

grades = ['Отлично', 'Хорошо', 'Удовлетворительно', 'Неудовлетворительно']
students = [20, 35, 30, 15]

bars = plt.bar(grades, students, edgecolor='black', linewidth=1)

# Добавляем значения над столбцами
for bar in bars:
    height = bar.get_height()
    plt.text(bar.get_x() + bar.get_width()/2., height,
             f'{int(height)}',
             ha='center', va='bottom', fontsize=12)


plt.title('Распределение студентов по оценкам', fontsize=16, pad=20)
plt.xlabel('Оценки', fontsize=14, labelpad=10)
plt.ylabel('Количество студентов', fontsize=14, labelpad=10)
plt.ylim(0, max(students) + 5)

plt.show()
```

Рисунок 5 – Решение задания №5

Используя данные предыдущей задачи, постройте круговую диаграмму с процентными подписями секторов.

[58]:
```
import matplotlib.pyplot as plt


grades = ['Отлично', 'Хорошо', 'Удовлетворительно', 'Неудовлетворительно']
students = [20, 35, 30, 15]
colors = ['green', 'lightgreen', 'yellow', 'red']


plt.pie(students, labels=grades, colors=colors, autopct='%1.1f%%')
plt.title('Распределение студентов по оценкам')

plt.show()
```

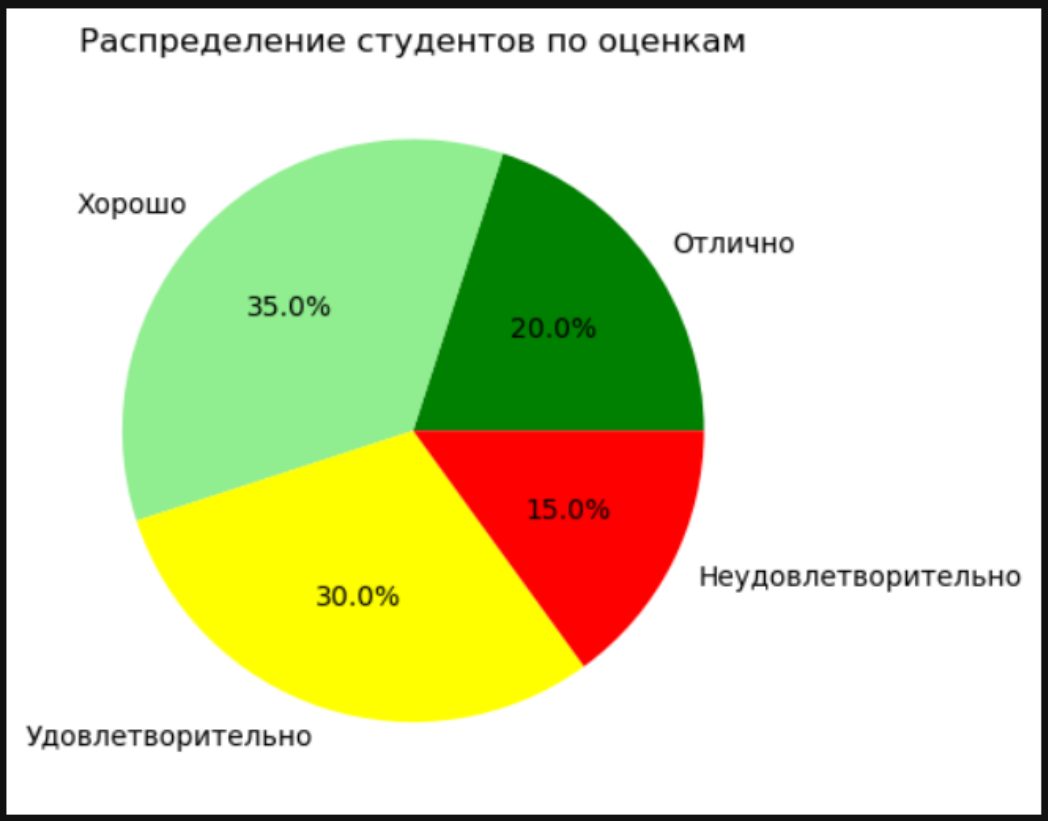

Рисунок 6 – Решение задания №6

Используя mpl_toolkits.mplot3d , постройте 3D-график функции z = sin(√x2 + y2) на сетке значений x, y в диапазоне [−5, 5]

[61]:

```
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D

# Сетка значений
x = np.linspace(-5, 5, 50)
y = np.linspace(-5, 5, 50)
X, Y = np.meshgrid(x, y)
Z = np.sin(np.sqrt(X**2 + Y**2))

fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
ax.plot_surface(X, Y, Z)

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

plt.show()
```

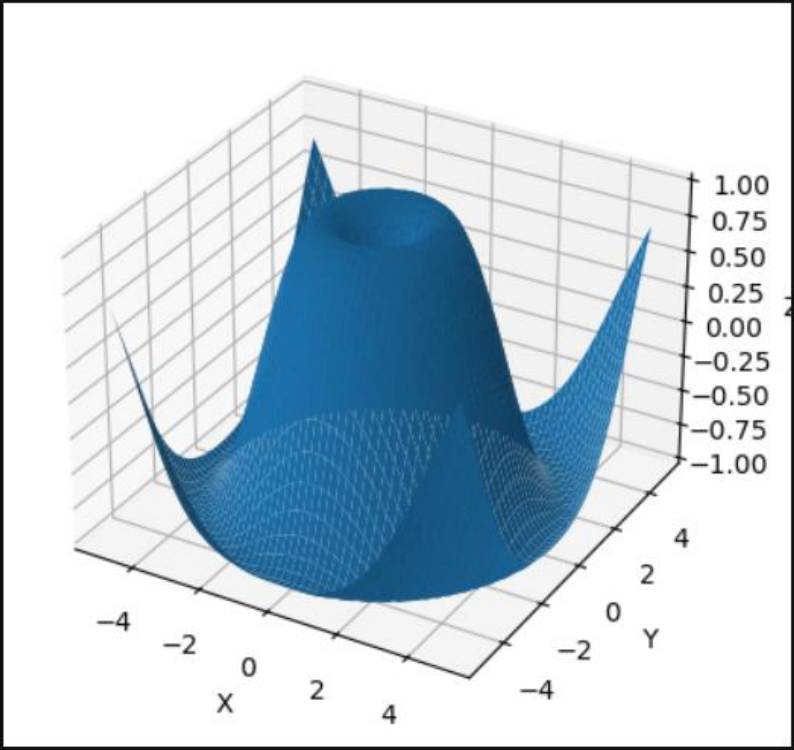

Рисунок 7 – Решение задания №7

Постройте четыре графика в одной фигуре (2 × 2 сетка):Линейный график y = x. Парабола y = x2.Синус y = sin(x).Косинус y = cos(x).Добавьте заголовки к каждому подграфику

[68]:

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


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

# Создаем фигуру с 4 подграфиками
fig, axs = plt.subplots(2, 2, figsize=(10, 8))

# График 1: y = x
axs[0, 0].plot(x, x)
axs[0, 0].set_title('Линейная функция: y = x')

# График 2: y = x^2
axs[0, 1].plot(x, x**2)
axs[0, 1].set_title('Парабола: y = x²')

# График 3: y = sin(x)
axs[1, 0].plot(x, np.sin(x))
axs[1, 0].set_title('Синусоида: y = sin(x)')

# График 4: y = cos(x)
axs[1, 1].plot(x, np.cos(x))
axs[1, 1].set_title('Косинусоида: y = cos(x)')


plt.tight_layout()
plt.show()
```

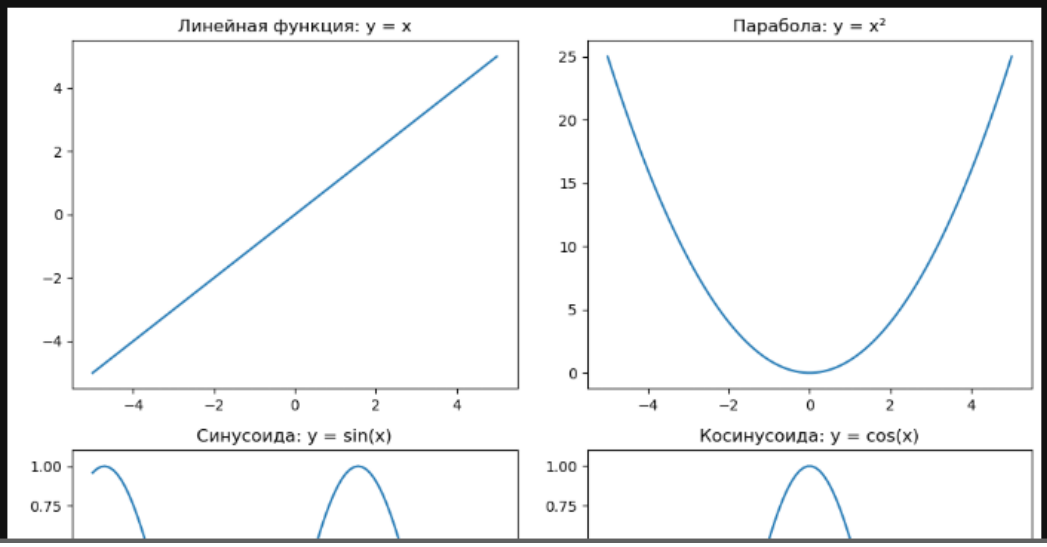

Рисунок 8 – Решение задания №8

Создайте случайную матрицу 10 × 10 с элементами от 0 до 1 и визуализируйте её как тепловую карту с цветовой шкалой.

[71]:
```
import numpy as np
import matplotlib.pyplot as plt

matrix = np.random.rand(10, 10)
plt.imshow(matrix, cmap='viridis')
# Добавляем цветовую шкалу
plt.colorbar()
plt.title('Тепловая карта случайной матрицы 10×10')
plt.xlabel('Столбцы')
plt.ylabel('Строки')

plt.show()
```

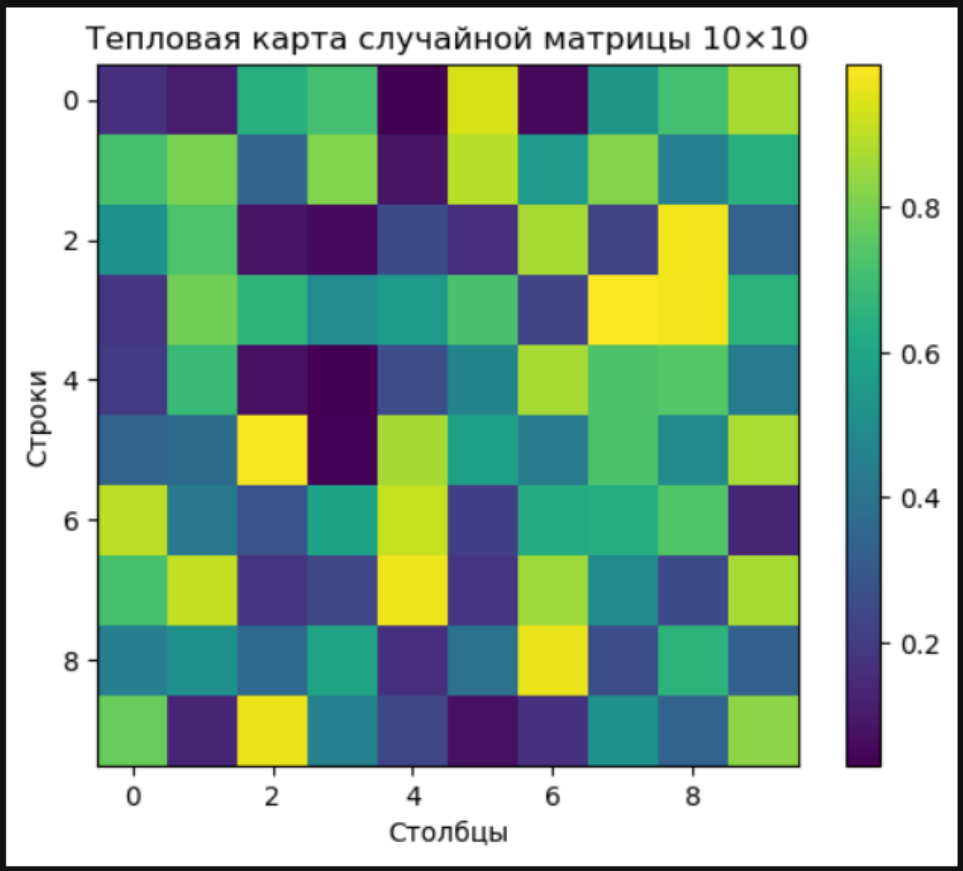

Рисунок 9 – Решение задания №9

1. Индивидуальное задание

Заряд батареи уменьшался в зависимости от времени использования:

Время работы (часы): [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10] Заряд (%) : [100, 92, 85, 75, 60, 50, 40, 28, 15, 5, 0] **Постройте график с красной линией и квадратными маркерами, добавьте подпись в точке разрядки**

[4]:
```
import matplotlib.pyplot as plt

hours = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
charge = [100, 92, 85, 75, 60, 50, 40, 28, 15, 5, 0]


plt.plot(hours, charge, 'r-s')  # Красная линия с квадратными маркерами
# Подпись точки разрядки
plt.annotate('Разрядка', (10, 0))
plt.title('Разрядка батареи')
plt.xlabel('Время работы (часы)')
plt.ylabel('Заряд (%)')

plt.show()
```

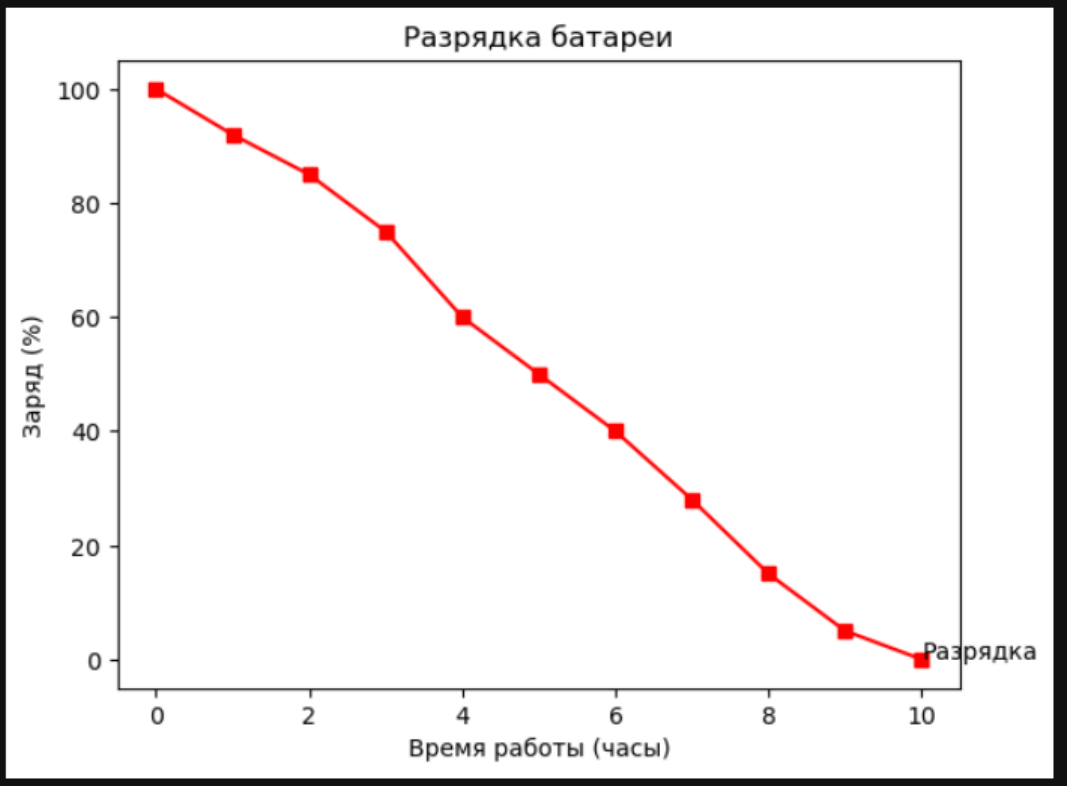

Рисунок 10 – График работы батареи от 100 до 0

Число выпущенных автомобилей (в млн):Компании: ['Toyota', 'Volkswagen', 'Ford', 'Honda', 'BMW'] Выпущено (млн): [10, 9, 6, 5, 3]

Используйте цветовую шкалу от светлого к темному в зависимости от количества выпущенных машин.

[8]:
```
import matplotlib.pyplot as plt

# Данные
companies = ['Toyota', 'Volkswagen', 'Ford', 'Honda', 'BMW']
production = [10, 9, 6, 5, 3]
colors = [0.9, 0.7, 0.5, 0.3, 0.1]  # От светлого к темному (0.1 - самый

plt.bar(companies, production, color=plt.cm.Greys(colors))
plt.title('Выпуск автомобилей по компаниям')
plt.xlabel('Компании')
plt.ylabel('Выпущено (млн)')

plt.show()
```

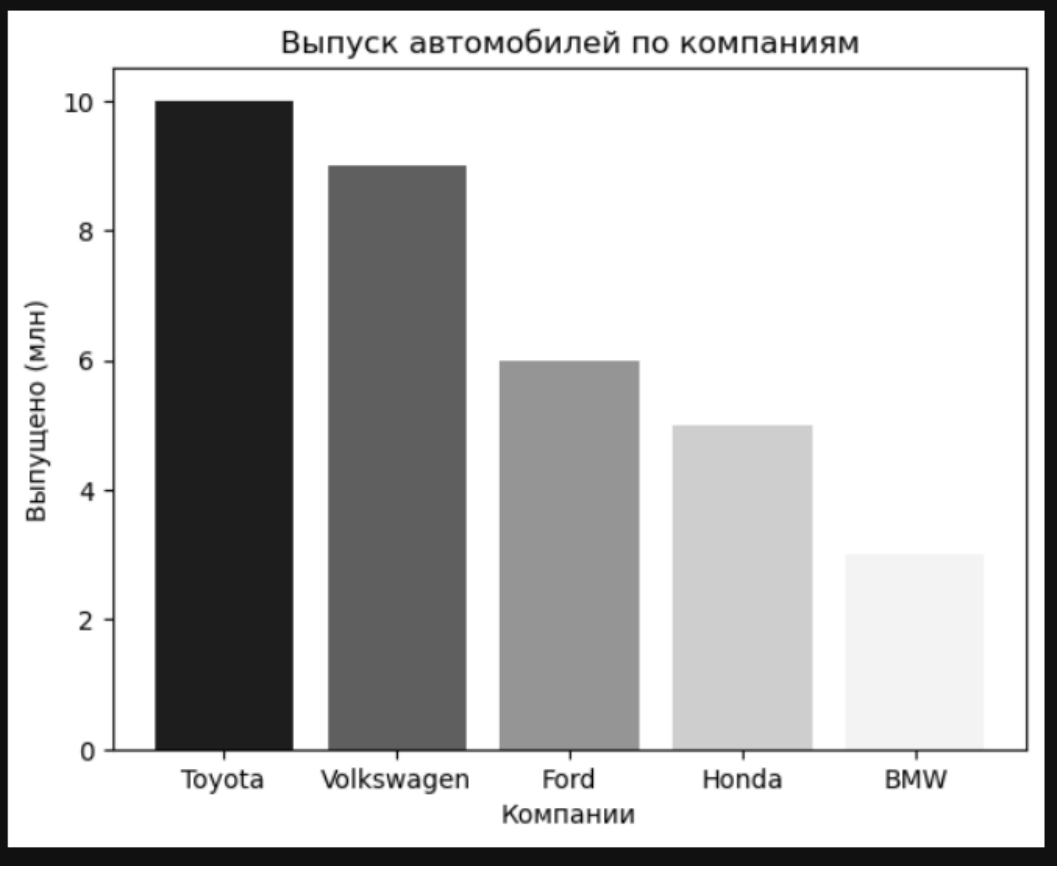

Рисунок 11 – Количество выпущенных автомобилей по кампаниям

В каждой задаче требуется:Построить график подинтегральной функции. 2. Вычислить площадь под кривой на заданном отрезке как значение определенного интеграла. 3. Закрасить область под графиком, чтобы визуализировать интеграл. Найдите площадь под: f(x) = |x – 1| на интервале [0, 2]

[23]:

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.integrate import quad

# Функция
def f(x):
    return np.abs(x - 1)
# Вычисление интеграла
integral_value, error = quad(f, 0, 2)


x = np.linspace(0, 2, 500)
y = f(x)
plt.plot(x, y, 'b-', linewidth=2)

# Закрашивание области под кривой
plt.fill_between(x, y, color='red', alpha=0.5)
plt.title(f'График функции f(x) = |x - 1|\nПлощадь под кривой: {int(integral_value)}')
plt.xlabel('x')
plt.ylabel('f(x)')
plt.grid(True)

plt.show()
```

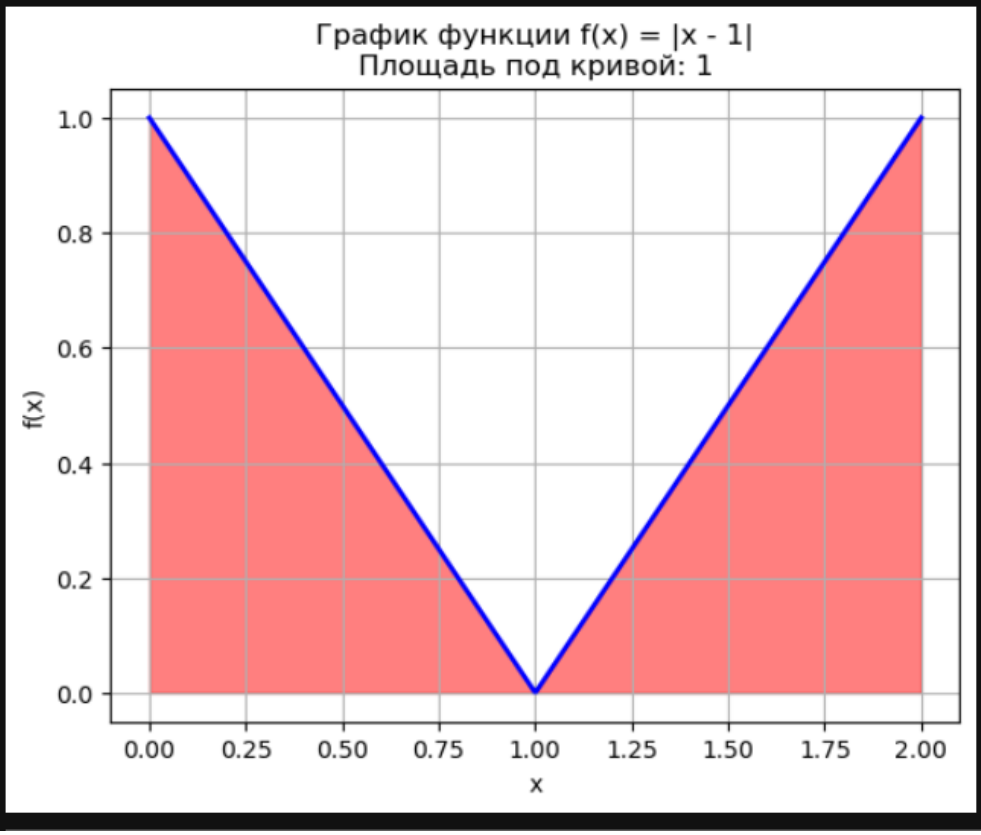

Рисунок 12 – Вычисление площади образованной фигуры

[31]:

```python
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D

x = np.linspace(-5, 5, 100)
y = np.linspace(-5, 5, 100)
X, Y = np.meshgrid(x, y)

# Вычисляем функцию
Z = np.log(X**2 + Y**2 + 1)


fig = plt.figure(figsize=(10, 7))
ax = fig.add_subplot(111, projection='3d')

poverh = ax.plot_surface(X, Y, Z, cmap='viridis')
fig.colorbar(poverh, shrink=0.5, aspect=5)
ax.set_title('График функции f(x, y) = ln(x² + y² + 1)')
ax.set_xlabel('X')
ax.set_ylabel('Y')
ax.set_zlabel('Z')

plt.tight_layout()
plt.show()
```

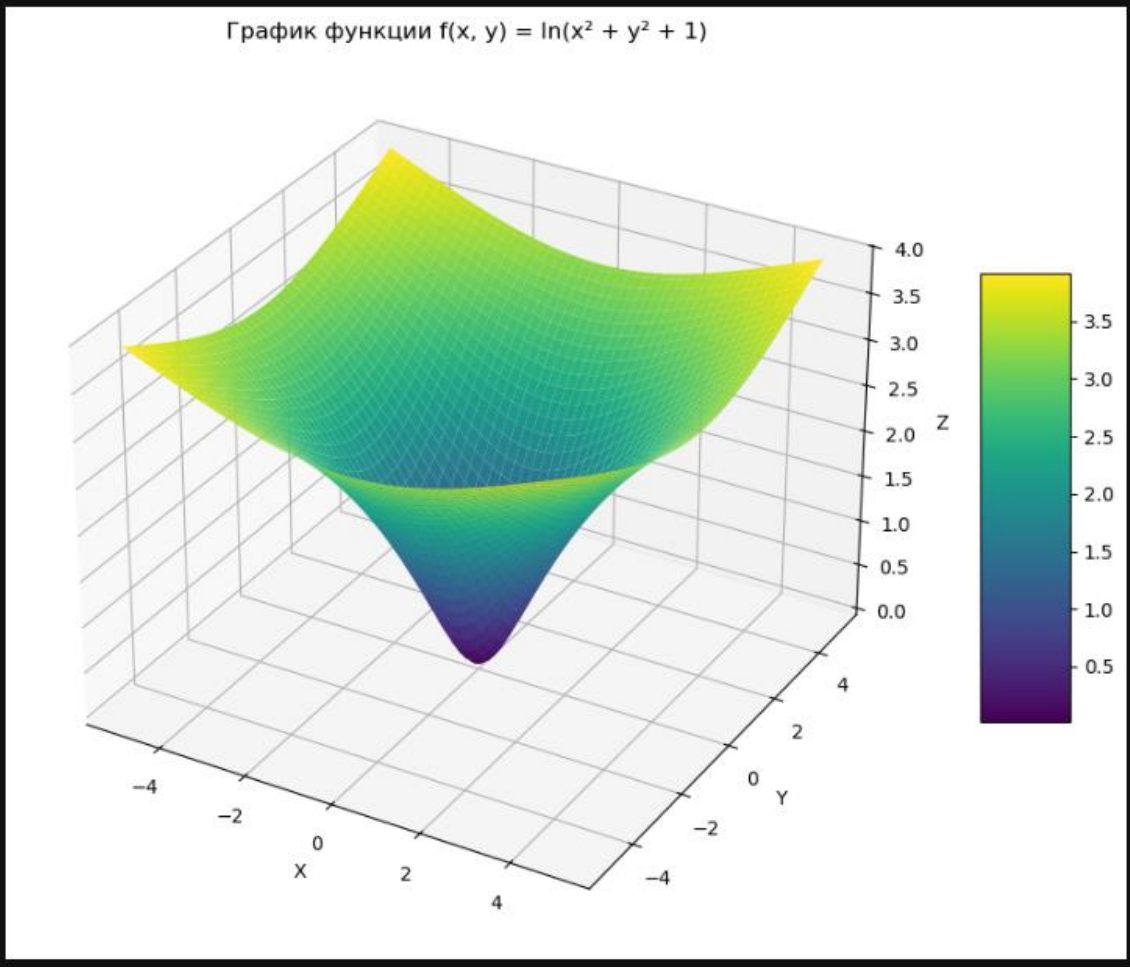

Рисунок 13 – Построение 3D-графика

Ссылка на репозиторий GitHub:

https://github.com/ArtemStepanovNkey/AI-university/tree/main

Ответы на контрольные вопросы:

1. Установка matplotlib

pip install matplotlib

2. Магическая команда Jupyter

%matplotlib inline # для статичных графиков

3. Отображение графика plot

import matplotlib.pyplot as plt

plt.plot([1, 2, 3], [4, 5, 1])

plt.show()

4. Несколько графиков на одном поле

plt.plot([1, 2, 3], [4, 5, 1], label='Line 1')

plt.plot([1, 2, 3], [2, 3, 4], label='Line 2')

plt.legend()

plt.show()

5. Диаграммы категориальных данных

categories = ['A', 'B', 'C']

values = [10, 20, 15]

plt.bar(categories, values) # Столбчатая диаграмма

plt.show()

6. Основные элементы графика

- plt.title() – заголовок
- plt.xlabel(), plt.ylabel() – подписи осей
- plt.grid() – сетка
- plt.legend() – легенда

7. Управление текстовыми надписями

```
plt.text(x=2, y=3, s='Текст')  # Координаты (x, y)

plt.annotate('Важно!', xy=(2, 3), xytext=(3, 4), arrowprops=dict(arrowstyle='->'))
```

8. Управление легендой

```
plt.plot([1, 2], label='Линия')

plt.legend(loc='upper right', fontsize=10)  # Позиция и размер шрифта
```

9. Цвет и стиль линий

```
plt.plot([1, 2], [1, 2], color='red', linestyle='--', linewidth=2)  # Красный пунктир
```

10. Размещение в разных полях (subplots)

```
fig, axs = plt.subplots(2, 2)  # 2x2 сетка

axs[0, 0].plot([1, 2], [1, 2])  # Первый график

plt.show()
```

11. Линейный график

```
plt.plot([1, 2, 3], [4, 5, 1], marker='o')  # С маркерами

plt.show()
```

12. Заливка областей

```
# Между графиком и осью X

plt.fill_between(x, y, color='blue', alpha=0.2)

# Между двумя графиками

plt.fill_between(x, y1, y2, color='gray', alpha=0.5)
```

13. Выборочная заливка по условию

```
plt.fill_between(x, y, where=(y > 0), color='green', alpha=0.3)
```

14. Двухцветная заливка

```
plt.fill_between(x, y, 0, where=(y > 0), color='green', alpha=0.3)

plt.fill_between(x, y, 0, where=(y <= 0), color='red', alpha=0.3)
```

15. Маркировка графиков

plt.plot([1, 2], [1, 2], marker='o', markersize=8, label='Маркеры')

plt.legend()

16. Обрезка графиков

plt.xlim(0, 5) # Ограничение по X

plt.ylim(0, 10) # Ограничение по Y

17. Ступенчатый график

plt.step([1, 2, 3], [4, 5, 1], where='post') # Особенность: постоянные значения между точками

plt.show()

18. Стековый график

plt.stackplot([1, 2, 3], [4, 5, 1], [2, 3, 4], labels=['A', 'B']) # Особенность: накопление значений

plt.legend()

plt.show()

19. Stem-график (дискретные значения)

plt.stem([1, 2, 3], [4, 5, 1]) # Особенность: вертикальные линии от оси до точки

plt.show()

20. Точечный график (scatter plot)

plt.scatter([1, 2, 3], [4, 5, 1], color='red', s=100) # Особенность: отдельные точки

plt.show()

21. Столбчатые диаграммы

plt.bar(['A', 'B', 'C'], [10, 20, 15], color=['red', 'green', 'blue'])

plt.show()

22. Групповая столбчатая диаграмма и errorbar

```
# Групповая
plt.bar([1, 2, 3], [10, 20, 15], width=0.4, label='Group 1')
plt.bar([1.4, 2.4, 3.4], [5, 10, 7], width=0.4, label='Group 2')
plt.legend()
# C errorbar
plt.errorbar([1, 2, 3], [10, 20, 15], yerr=[1, 2, 3], fmt='o')
plt.show()
```

23. Круговая диаграмма

```
plt.pie([10, 20, 30], labels=['A', 'B', 'C'], autopct='%1.1f%%')
plt.show()
```

24. Цветовые карты (colormaps)

```
plt.imshow([[1, 2], [3, 4]], cmap='viridis')
plt.colorbar()
plt.show()
```

25. Отображение изображения

```
img = plt.imread('image.png')
plt.imshow(img)
plt.axis('off')  # Скрыть оси
plt.show()
```

26. Тепловая карта

```
import numpy as np
data = np.random.rand(10, 10)
plt.imshow(data, cmap='hot')
plt.colorbar()
plt.show()
```

27. Линейный 3D-график

```
from mpl_toolkits.mplot3d import Axes3D
fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
ax.plot([1, 2, 3], [4, 5, 6], [7, 8, 9])
plt.show()
```

28. Точечный 3D-график

```
ax.scatter([1, 2, 3], [4, 5, 6], [7, 8, 9], c='red', s=100)
plt.show()
```

29. Каркасная поверхность (wireframe)

```
X, Y = np.meshgrid(np.linspace(-5, 5, 50), np.linspace(-5, 5, 50))
Z = np.sin(np.sqrt(X**2 + Y**2))
ax.plot_wireframe(X, Y, Z, color='black')
plt.show()
```

30. Трехмерная поверхность (surface)

```
ax.plot_surface(X, Y, Z, cmap='viridis')
plt.colorbar()
plt.show()
```