МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение высшего образования
«СЕВЕРОКАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Кафедра

инфокоммуникаций

Институт цифрового

развития

ОТЧЁТ

по лабораторной работе №3.5

Дисциплина: «Построение 3D графиков. Работа с mplot3d Toolkit»

Выполнила: студентка 2 курса группы Пиж-б-о-21-1

Рязанцев Матвей Денисович

Ставрополь 2023

Цель работы: исследовать базовые возможности визуализации данных в трехмерном пространстве средствами библиотеки matplotlib языка программирования Python.

1. Проработайте примеры лабораторной работы в отдельном ноутбуке.

Рисунок 1 – Выполненные примеры

2. Создать ноутбук, в котором выполнить решение вычислительной задачи (например, задачи из области физики, экономики, математики, статистики и т. д.) требующей построения трехмерного графика, условие которой предварительно необходимо согласовать с преподавателем.

Рисунок 2 – Выполненное индивидуальное задание

Вопросы для защиты работы

1. Как выполнить построение линейного 3D-графика с помощью matplotlib?

Matplotlib позволяет строить 3D графики. Для этого импортируем необходимые модули для работы с 3D:

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

Для построения линейного графика используется функция plot().

```
Axes3D.plot(self, xs, ys, *args, zdir='z', **kwargs)
```

где:

– xs: 1D-массив - x координаты.

– y : 1D-массив - y координаты.

s

– zs: скалярное значение или 1D-массив - z координаты. Если передан скаляр, то он будет присвоен всем точкам графика.

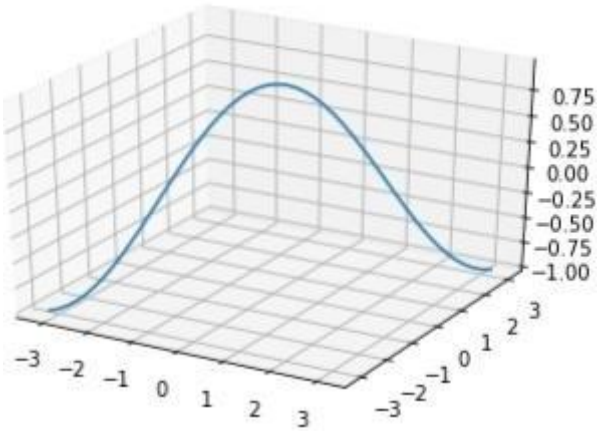
– zdir: {‘x’, ‘y’, ‘z’} - определяет ось, которая будет принята за z направление, значение по умолчанию: ‘z’.

– **kwargs - дополнительные аргументы, аналогичные тем, что используются в функции plot() для построения двумерных графиков.

Пример:

```
x = np.linspace(-np.pi, np.pi, 50)
y = x
z = np.cos(x)
```

```
fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
ax.plot(x, y, z, label='parametric curve')
```



2. Как выполнить построение точечного 3D-графика с помощью matplotlib?

Для построения точечного графика используется функция scatter().

Axes3D.scatter(self, xs, ys, zs=0, zdir='z', s=20, c=None, depthshade=True, *args, **kwargs)

– xs, ys: массив - координаты точек по осям x и y.

– zs: float или массив, optional - координаты точек по оси z. Если передан скаляр, то он будет присвоен всем точкам графика. Значение по умолчанию: 0.

– zdir: {'x', 'y', 'z', '-x', '-y', '-z'}, optional - определяет ось, которая будет принята за z направление, значение по умолчанию: 'z'

– s: скаляр или массив, optional - размер маркера. Значение по умолчанию: 20.

– c: color, массив, массив значений цвета, optional - цвет маркера.

– depthshade: bool, optional - затенение маркеров для придания эффекта глубины.

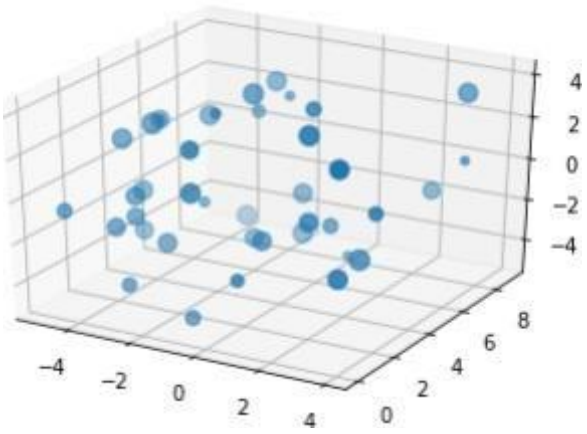
– **kwargs - дополнительные аргументы, аналогичные тем, что используются в функции scatter() для построения двумерных графиков.

Пример:

```
np.random.seed(123)
x = np.random.randint(-5, 5, 40)
y = np.random.randint(0, 10, 40)
z = np.random.randint(-5, 5, 40)
s = np.random.randint(10, 100, 20)

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

ax.scatter(x, y, z, s=s)
```



3. Как выполнить построение каркасной поверхности с помощью matplotlib?

Для построения каркасной поверхности используется функция plot_wireframe()

– plot_wireframe(self, X, Y, Z, *args, **kwargs)

– X, Y, Z: 2D-массивы - данные для построения поверхности.

– rcount, ccount: int - максимальное количество элементов каркаса, которое будет использовано в каждом из направлений. Значение по умолчанию: 50.

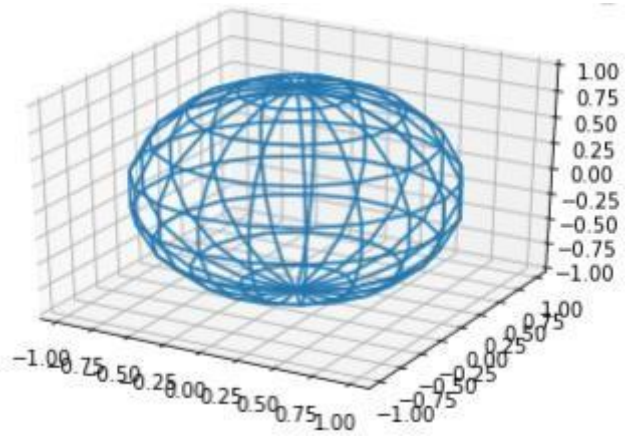
– rstride, cstride: int - параметры определяют величину шага, с которым будут браться элементы строки / столбца из переданных массивов. Параметры rstride, cstride и rcount, ccount являются взаимоисключающими.

– **kwargs - дополнительные аргументы, определяемые Line3DCollection

Пример:

```
u, v = np.mgrid[0:2*np.pi:20j, 0:np.pi:10j]
x = np.cos(u)*np.sin(v)
y = np.sin(u)*np.sin(v)
z = np.cos(v)

fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
ax.plot_wireframe(x, y, z)
ax.legend()
```



4. Как выполнить построение трехмерной поверхности с помощью matplotlib?

Для построения поверхности используйте функцию plot_surface().

plot_surface(self, X, Y, Z, *args, norm=None, vmin=None, vmax=None, lightsource=None, **kwargs)

– X, Y, Z : 2D-массивы - данные для построения поверхности.

– rcount, ccount : int - см. rcount, ccount в “Каркасная поверхность“.

– rstride, cstride : int - см.rstride, cstride в “Каркасная поверхность“.

– color: color - цвет для элементов поверхности.

– cmap: Colormap - Colormap для элементов поверхности.

– facecolors: массив элементов color - индивидуальный цвет для каждого элемента поверхности.

– norm: Normalize - нормализация для colormap.

– vmin, vmax: float - границы нормализации.

– shade: bool - использование тени для facecolors. Значение поумолчанию: True.

– lightsource: LightSource - объект класса LightSource – определяет источник света, используется, только если shade = True.

– **kwargs - дополнительные аргументы, определяемыеPoly3DCollection