МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение высшего образования

«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Кафедра инфокоммуникаций

Отчет по лабораторной работе №3.6

Построение 3D графиков. Работа с mplot3d Toolkit.

по дисциплине «Анализ данных»

Выполнил студент группы ИВТ-б-о-21-1

Лысенко И.А. « »_____20__г.

Подпись студента_____

Работа защищена « »_____20__г.

Проверил Воронкин Р.А. _____

(подпись)

Ставрополь 2023

Цель работы: исследовать базовые возможности интерактивных оболочек IPython и Jupyter Notebook для языка программирования Python.

Ход работы:

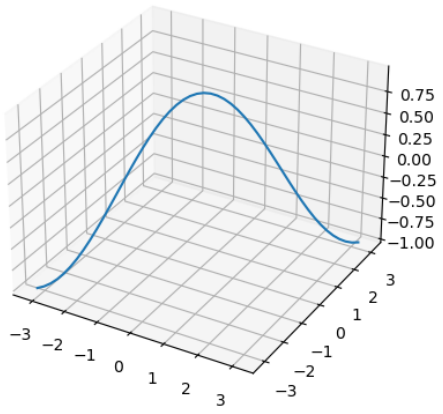
1. Создал репозиторий на GitHub: https://github.com/IsSveshuD/lab_3.6.git .

2. Проработал примеры:

Линейный график

Axes3D.plot(xs, ys, *args, zdir='z', **kwargs) xs: 1D-массив - x координаты. ys: 1D-массив - y координаты. 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.

In [2]:
```
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');
```



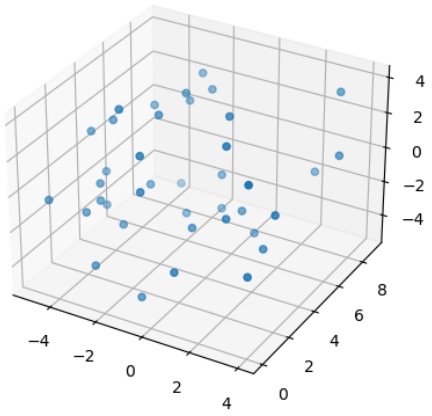
Точечный график

Для построения точечного графика используется функция scatter(). 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. Возможные значения: Строковое значение цвета для всех маркеров. Массив строковых значений цвета. Массив чисел, которые могут быть отображены в цвета через функции cmap и norm. 2D массив, элементами которого являются RGB или RGBA. depthshade: bool, optional - затенение маркеров для придания эффекта глубины. **kwargs - дополнительные аргументы, аналогичные тем, что используются в функции scatter() для построения двумерных графиков. Axes3D.scatter(self, xs, ys, zs=0, zdir='z', s=20, c=None, depthshade=True, *args, **kwargs)

In [3]:
```
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)
```

In [4]:
```
fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
ax.scatter(x, y, z, s);
```

```
C:\Users\user\anaconda3\lib\site-packages\mpl_toolkits\mplot3d\art3d.py:1104: FutureWarning: elementwise comparison failed; ret
urning scalar instead, but in the future will perform elementwise comparison
  if zdir == 'x':
C:\Users\user\anaconda3\lib\site-packages\mpl_toolkits\mplot3d\art3d.py:1106: FutureWarning: elementwise comparison failed; ret
urning scalar instead, but in the future will perform elementwise comparison
  elif zdir == 'y':
```



Каркасная поверхность

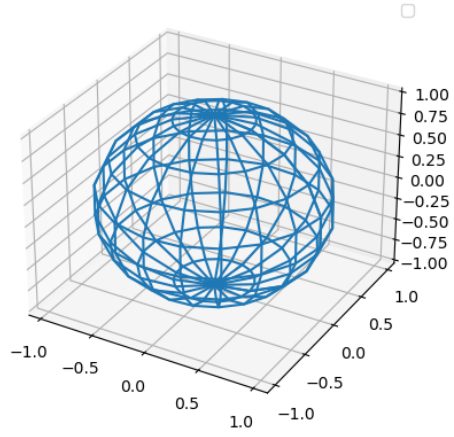
Для построения каркасной поверхности используется функция plot_wireframe(). plot_wireframe(self, X, Y, Z, *args, **kwargs)

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.

In [5]:
```
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)
```

In [6]:
```
fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
ax.plot_wireframe(x, y, z)
ax.legend();
```

```
No artists with labels found to put in legend.  Note that artists whose label start with an underscore are ignored when legend
() is called with no argument.
```



Поверхность

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 по умолчанию: True. lightsource: LightSource - объект класса LightSource – определяет источник света, используется, только если shade = True. **kwargs - дополнительные аргументы, определяемые Poly3DCollection.

In [7]:
```
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)
```

In [8]:
```
fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
ax.plot_surface(x, y, z, cmap='inferno')
ax.legend();
```

```
No artists with labels found to put in legend.  Note that artists whose label start with an underscore are ign
() is called with no argument.
```

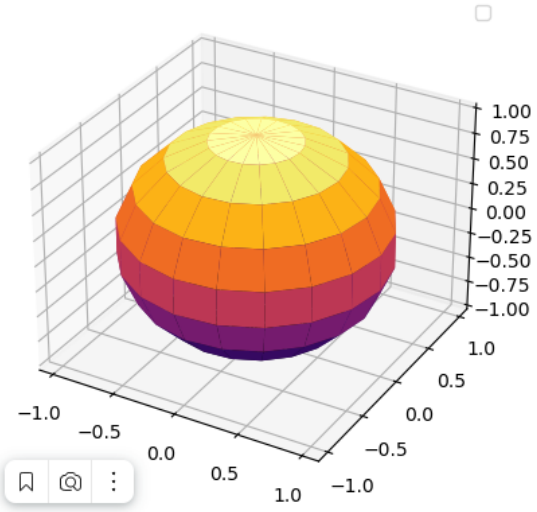


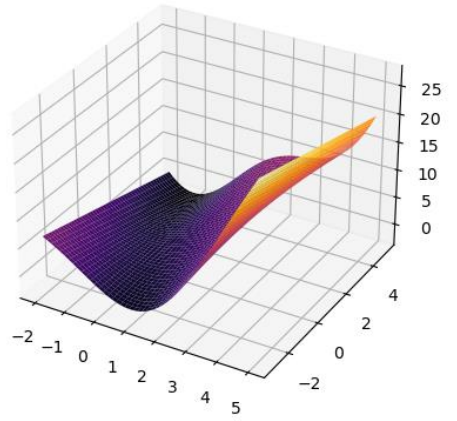
Рисунок 1 – Примеры

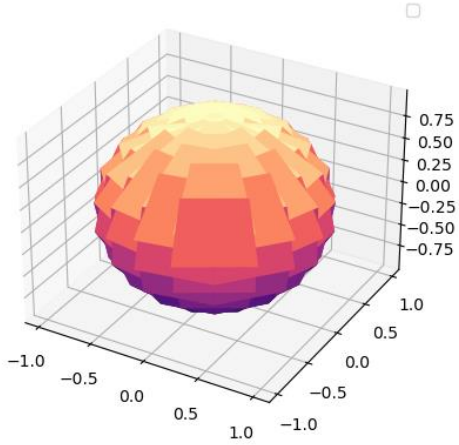
3. Выполнил индивидуальное задание.

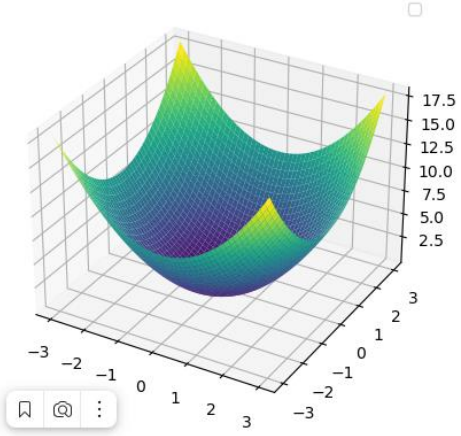
Создать ноутбук, в котором выполнить решение вычислительной задачи (например, задачи из области физики, экономики, математики, статистики и т. д.) требующей построения трехмерного графика, условие которой предварительно необходимо согласовать с преподавателем.

Построить три любых трехмерных графика.

In [57]:
```
import matplotlib.pyplot as plt
import numpy as np
from mpl_toolkits.mplot3d import Axes3D


x1, y1 = np.mgrid[-2:5:50j, -3:5:50j]
z = np.power(x1,2)+y1*np.sin(x1)

u, v = np.mgrid[0:4*np.pi:20j, 0:2*np.pi:10j]
x2 = np.cos(u)*np.cos(v)
y2 = np.cos(u)*np.sin(v)
z2 = np.sin(u)

x3, y3 = np.mgrid[-3:3:50j, -3:3:50j]
z3 = np.power(x3,2)+np.power(y3,2)

fig = plt.figure()
figs = plt.figure()
figr = plt.figure()

ax = fig.add_subplot(111, projection='3d')
axs = figs.add_subplot(111, projection='3d')
axr = figr.add_subplot(111, projection='3d')

ax.plot_surface(x1, y1, z, cmap='inferno')
axs.plot_surface(x2, y2, z2, cmap='magma')
axr.plot_surface(x3, y3, z3, cmap='viridis')

ax.legend()
axs.legend()
axr.legend()
```







In [61]:
```
u, v = np.mgrid[0:100:100j, 0:100:100j]
x = 0.01*u - 0.05
y = 0.01*u - 0.05
z = np.sin(u)*np.sin(v)


fig = plt.figure()


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


ax.plot_surface(x, y, z, cmap='inferno')

ax.legend()
```

```
No artists with labels found to put in legend.  Note that artists whose label start with an underscore are ignored when legend
() is called with no argument.
```

Out[61]: `<matplotlib.legend.Legend at 0x2951b5759c0>`

```
Out[61]: <matplotlib.legend.Legend at 0x2951b5759c0>
```

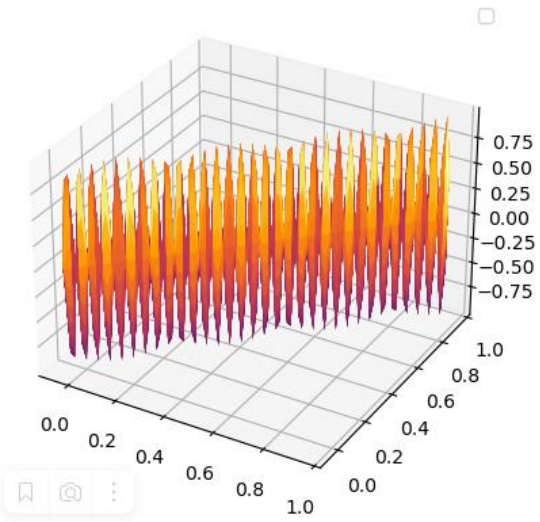


Рисунок 2 – Индивидуальное задание

Ответы на вопросы:

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

Линейный график

Для построения линейного графика используется функция *plot()*.

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

- xs: 1D-массив - *x* координаты.
- ys: 1D-массив - *y* координаты.
- 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.

```
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?

Точечный график

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. Возможные значения:
  - Строковое значение цвета для всех маркеров.
  - Массив строковых значений цвета.
  - Массив чисел, которые могут быть отображены в цвета через функции *cmap* и *norm*.
  - *2D* массив, элементами которого являются *RGB* или *RGBA*.
- depthshade: bool, optional - затенение маркеров для придания эффекта глубины.
- **kwargs - дополнительные аргументы, аналогичные тем, что используются в функции *scatter()* для построения двумерных графиков.

```
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?

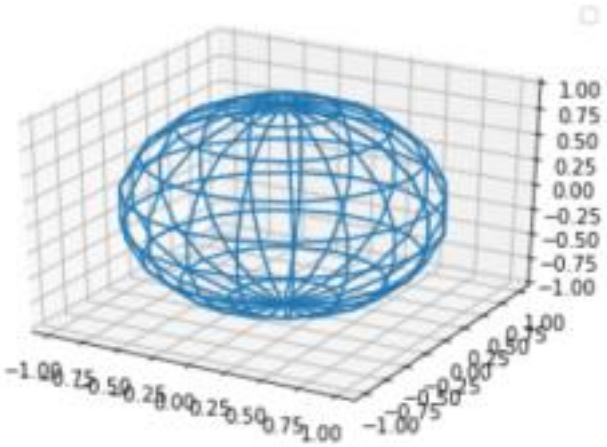
Каркасная поверхность

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*(https://matplotlib.org/api/_as_gen/mpl_toolkits.mplot3d.art3d.Line3DCollection.html#mpl_toolkits.mplot3d.art3d.Line3DCollection).

```
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?

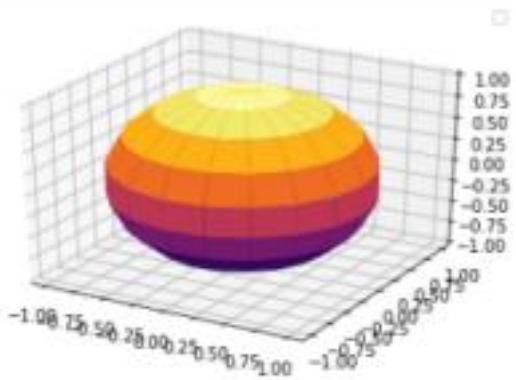
Поверхность

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 (https://devpractice.ru/matplotlib-lesson-5-1-mplot3d-toolkit/#p3)".
- rstride, cstride : int - см.*rstride*, *cstride* в "Каркасная поверхность (https://devpractice.ru/matplotlib-lesson-5-1-mplot3d-toolkit/#p3)".
- 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 по умолчанию: *True*.
- lightsource: *LightSource* - объект класса *LightSource* – определяет источник света, используется, только если *shade* = *True*.
- **kwargs - дополнительные аргументы, определяемые *Poly3DCollection*(https://matplotlib.org/api/_as_gen/mpl_toolkits.mplot3d.art3d.Poly3DCollection.html#mpl_toolkits.mplot3d.art3d.Poly3DCollection).

```
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_surface(x, y, z, cmap='inferno')
ax.legend()
```



Вывод: в результате выполнения лабораторной работы были получены практические навыки и теоретические сведения необходимые для работы с базовыми возможностями визуализации данных в трехмерном пространстве средствами библиотеки matplotlib языка программирования Python.