

Лабораторна Робота 12 – Візуалізація даних в Matplotlib:

Побудова 3D графіків

Мета роботи – вивчити та засвоїти навички роботи з базовим функціоналом модуля Matplotlib та mplot3d Toolkit. Ознайомитися з принципами та особливостями візуалізації даних у тривимірних графіках та засвоїти практичні навички їх використання.

1.1 Побудова 3D графіків

До цього моменту всі графіки, які ми будували були двовимірні, Matplotlib дозволяє будувати 3D графіки.

- Лінійний 3D-графік
- Точковий 3D-графік
- Каркасна поверхня
- Поверхня

Імпортуємо необхідні модулі для роботи з 3D:

```
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D as ax3
```

У бібліотеці доступні інструменти для побудови різних типів графіків. Розглянемо деякі з них більш детально.

1.2 Лінійний графік

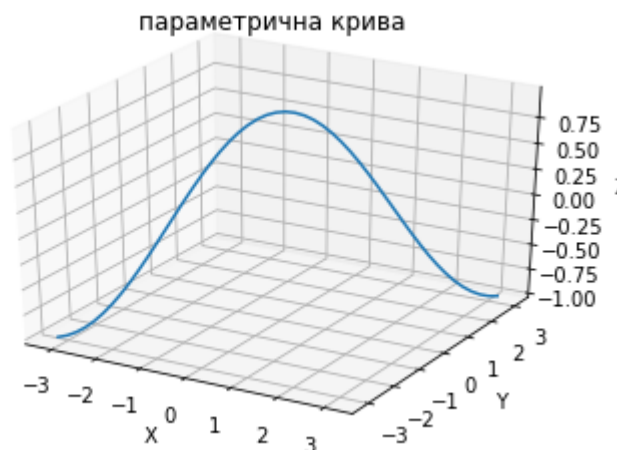
Для побудови лінійного графіка використовується функція `plot()`.

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

- `xs`: 1D масив (x координати).
- `ys`: 1D масив (y координати).

- `zs`: скалярне значення або 1D масив (z координати). Якщо переданий скаляр, то він буде присвоєний всім точкам графіка.
- `zdir`: {'x', 'y', 'z'} - Визначає вісь, яка буде прийнята за z напрямком, значення за замовчуванням: 'z'.
- `**kwargs` - Додаткові аргументи, аналогічні тим, що використовуються в функції `plot()` для побудови двовимірних графіків.

```
[26]: 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)
      ax.set_title("параметрична крива")
      ax.set_xlabel("X")
      ax.set_ylabel("Y")
      ax.set_zlabel("Z")
      fig.savefig("MyFile.png", dpi=800)
```



1.3 Точковий графік

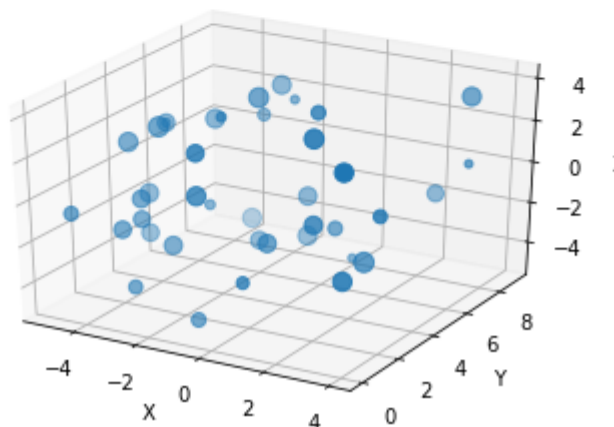
Для побудови точкового графіка використовується функція `scatter()`.

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

- `xs, ys`: масив. Координати точок по осях x і y .
- `zs`: float або масив, optional. Координати точок по осі z . Якщо переданий скаляр, то він буде присвоєний всім точкам графіка. Значення за замовчуванням: 0.

- `zdir`: { 'x', 'y', 'z', '-x', '-y', '-z' }, optional. Визначає вісь, яка буде прийнята за z напрямом, значення за замовчуванням: 'z'
- `s`: скаляр або масив, optional. Розмір маркера. Значення за замовчуванням: 20.
- `c`: color, масив, масив значень кольору, optional. Колір маркера.
Можливі значення:
 - Строкове значення кольору для всіх маркерів.
 - Масив строкових значень кольору.
 - Масив чисел, які можуть бути відображені як кольори через функції `cm` і `norm`.
 - 2D масив, елементами якого є RGB або RGBA.
- `**kwargs` - Додаткові аргументи, аналогічні тим, що використовуються в функції `scatter()` для побудови двовимірних графіків.

```
[31]: 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)
ax.set_xlabel("X")
ax.set_ylabel("Y")
ax.set_zlabel("Z")
fig.savefig("MyFile.png", dpi=800)
```



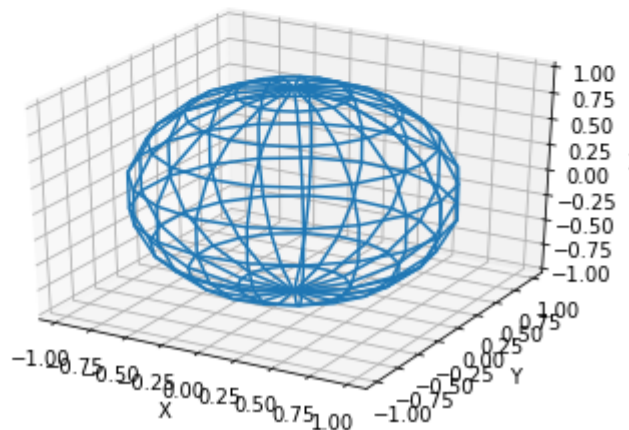
1.4 Каркасна поверхня

Для побудови каркасної поверхні використовується функція `plot_wireframe()`.

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

- `X, Y, Z`: 2D масиви. Дані для побудови поверхні.
- `rcount, ccount: int` - максимальна кількість елементів каркаса, яке буде використано в кожному з напрямків. Значення за замовчуванням: 50.
- `rstride, cstride: int` - ці установки впливають на величину кроку, з яким будуть братися елементи рядка/стовпця з переданих масивів. Параметри `rstride, cstride` і `rcount, ccount` є взаємовиключними.
- `**kwargs` - додаткові аргументи, які визначаються `Line3DCollection`.

```
[34]: 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.set_xlabel("X")
      ax.set_ylabel("Y")
      ax.set_zlabel("Z")
      fig.savefig("MyFile.png", dpi=800)
```



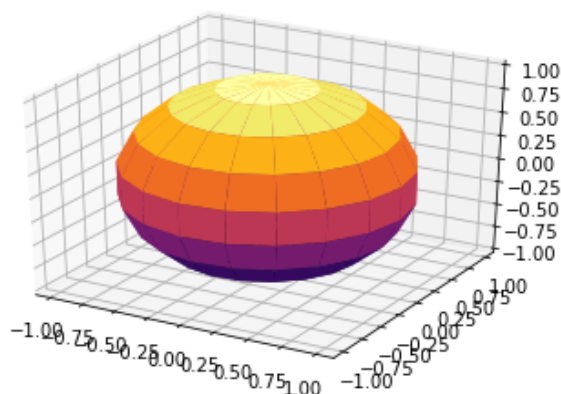
1.5 Поверхня

Для побудови поверхні використовуйте функцію `plot_surface()`.

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

- `X, Y, Z`: 2D масиви. Дані для побудови поверхні.
- `rcount, ccount`: `int` - див. `rcount, ccount` в "Каркасна поверхня".
- `rstride, cstride`: `int` - див. `rstride, cstride` в "Каркасна поверхня".
- `color`: `color` - колір для елементів поверхні.
- `cmap`: `Colormap` - `Colormap` для елементів поверхні.
- `facecolors`: масив елементів `color` - індивідуальний колір для кожного елемента поверхні.
- `norm`: `Normalize` - нормалізація для `colormap`.
- `vmin, vmax`: `float` - межі нормалізації.
- `shade`: `bool` - використання тіні для `facecolors`. Значення за замовчуванням: `True`.
- `lightsources`: `LightSource` - об'єкт класу `LightSource` - визначає джерело світла, використовується, тільки якщо `shade = True`.
- `**kwargs` - додаткові аргументи, які визначаються `Poly3DCollection`.

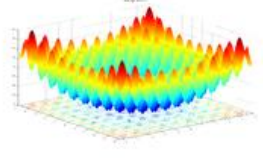
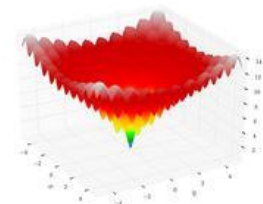
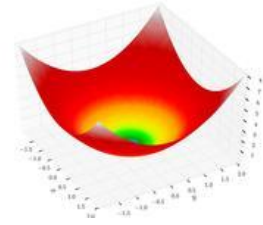
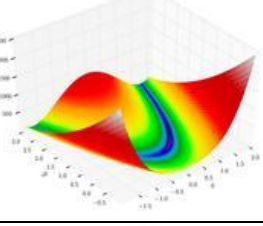
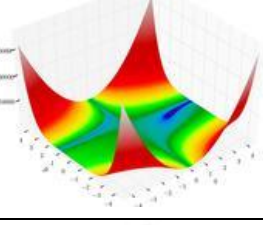
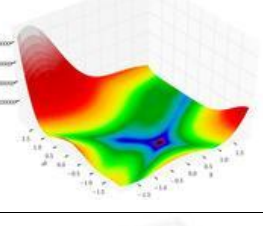
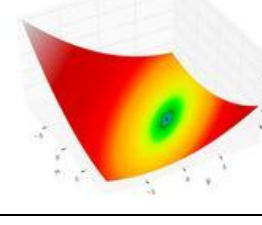
```
[36]: u, v = np.mgrid[0:2*np.pi:20j, 0:np.pi:10j]
      x = np.cos(u)*np.sin(v)
      y = np.sin(u)*np.sin(v)
      z = np.cos(v)
      fig = plt.figure()
      ax = fig.add_subplot(111, projection='3d')
      ax.plot_surface(x, y, z, cmap='inferno')
      fig.savefig("MyFile.png", dpi=800)
```

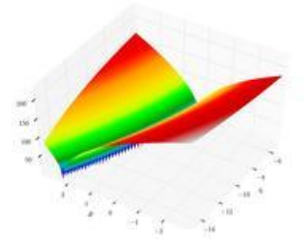
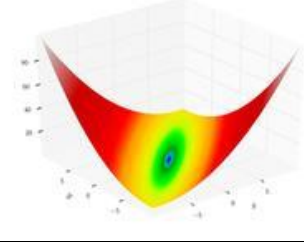
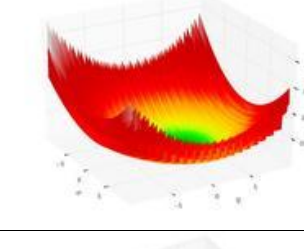
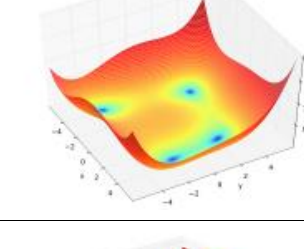
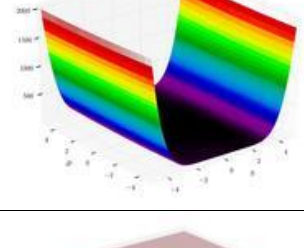
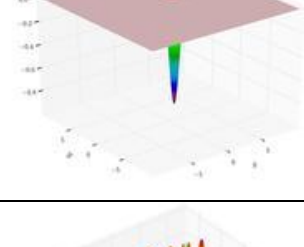
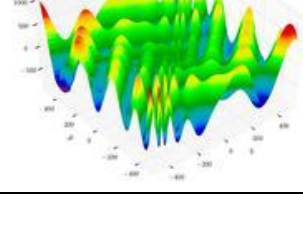


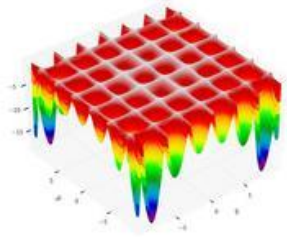
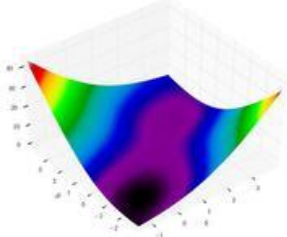
2. Варіанти завдання

Використовуючи засоби matplotlib побудувати графіки 3D відповідно до таблиці варіантів (таб. 12.1).

Таблиця 12.1 – Варіанти завдань

	Назва	Формула	Малюнок
1.	Функція Растрігіна	$f(\mathbf{x}) = An + \sum_{i=1}^n [x_i^2 - A \cos(2\pi x_i)]$ <p>where: $A = 10$</p>	
2.	Функція Еклі	$f(x, y) = -20 \exp \left[-0.2 \sqrt{0.5 (x^2 + y^2)} \right] - \exp [0.5 (\cos 2\pi x + \cos 2\pi y)] + e + 20$	
3.	Функція сфери	$f(\mathbf{x}) = \sum_{i=1}^n x_i^2$	
4.	Функція Розенброка	$f(\mathbf{x}) = \sum_{i=1}^{n-1} \left[100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2 \right]$	
5.	Функція Біла	$f(x, y) = (1.5 - x + xy)^2 + (2.25 - x + xy^2)^2 + (2.625 - x + xy^3)^2$	
6.	Функція Гольдман-Прайса	$f(x, y) = \left[1 + (x + y + 1)^2 (19 - 14x + 3x^2 - 14y + 6xy + 3y^2) \right] \left[30 + (2x - 3y)^2 (18 - 32x + 12x^2 + 48y - 36xy + 27y^2) \right]$	
7.	Функція Бута	$f(x, y) = (x + 2y - 7)^2 + (2x + y - 5)^2$	

8.	Функція Букіна N6	$f(x, y) = 100\sqrt{ y - 0.01x^2 } + 0.01 x + 10 .$	
9.	Функція Матьяса	$f(x, y) = 0.26(x^2 + y^2) - 0.48xy$	
10.	Функція Леві N 13	$f(x, y) = \sin^2 3\pi x + (x - 1)^2 (1 + \sin^2 3\pi y) + (y - 1)^2 (1 + \sin^2 2\pi y)$	
11.	Функція Гіммельблау	$f(x, y) = (x^2 + y - 11)^2 + (x + y^2 - 7)^2.$	
12.	Функція трехгорбого верблюда	$f(x, y) = 2x^2 - 1.05x^4 + \frac{x^6}{6} + xy + y^2$	
13.	Функція Ізома	$f(x, y) = -\cos(x) \cos(y) \exp\left(-\left((x - \pi)^2 + (y - \pi)^2\right)\right)$	
14.	Функція "підставка для яєць" (Eggholder function)	$f(x, y) = -(y + 47) \sin \sqrt{\left \frac{x}{2} + (y + 47)\right } - x \sin \sqrt{ x - (y + 47) }$	

15.	Таблична функція Хольдера	$f(x, y) = - \left \sin x \cos y \exp \left(1 - \frac{\sqrt{x^2 + y^2}}{\pi} \right) \right $	
16.	Функція МакКорміка	$f(x, y) = \sin(x + y) + (x - y)^2 - 1.5x + 2.5y + 1$	

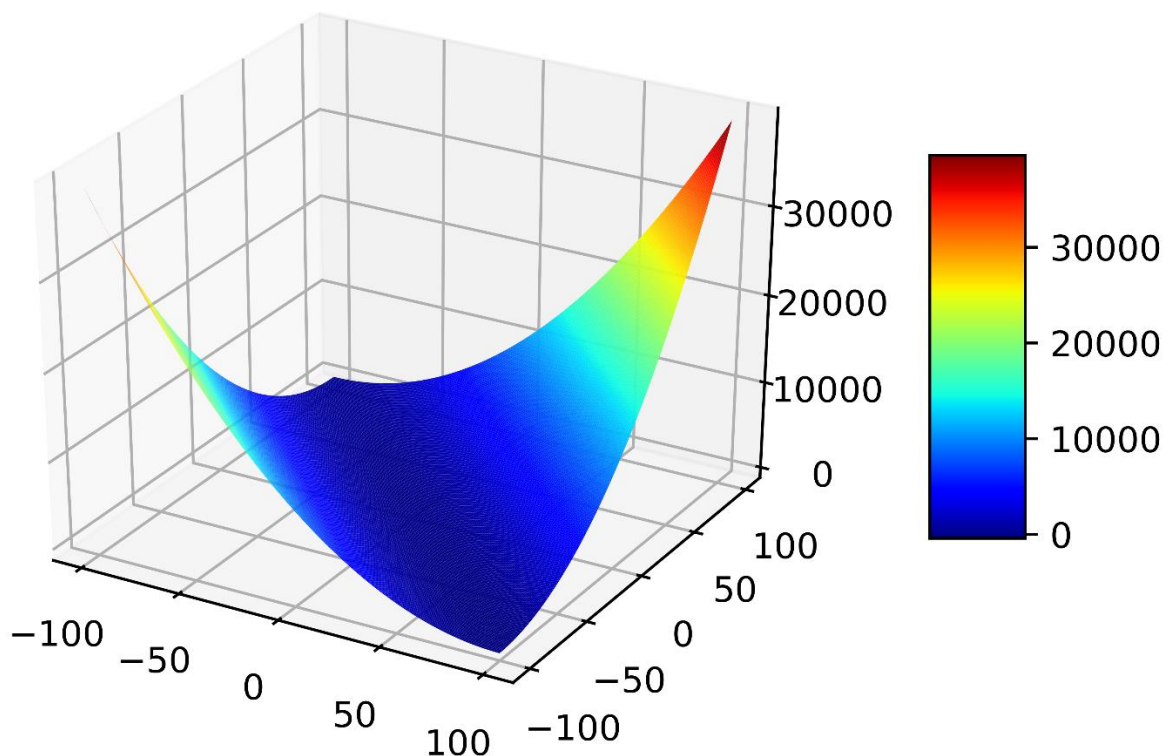
3. Приклад

Варіант -16.

Файл програми

```
[4]: from mpl_toolkits.mplot3d import Axes3D
import matplotlib.pyplot as plt
from matplotlib import cm
import numpy as np
fig = plt.figure()
ax = fig.gca(projection='3d')
# Make data.
X = np.arange(-100, 100, 0.2)
Y = np.arange(-100, 100, 0.2)
X, Y = np.meshgrid(X, Y)
Z = np.sin(X+Y)+(X+Y)**2-(1.5*X)+(2.5*Y)+1
# Plot the surface.
surf = ax.plot_surface(X, Y, Z, rstride=5, cstride=5, cmap = cm.jet)
# Add a color bar which maps values to colors.
fig.colorbar(surf, shrink=0.5, aspect=4)
plt.show()
fig.savefig("MyFile.png",dpi=800)
```

Результат роботи програми



4. Контрольні запитання

1. Модуль `matplotlib`. Призначення та базові принципи застосування. Наведіть приклади.
2. Наведіть приклади існуючих у `matplotlib` видів тривимірних графіків.
3. Опишіть що таке поверхня у `matplotlib` та чим вона відрізняється від каркасної поверхні. Наведіть приклади .
4. Опишіть існуючі види налаштування відображення графіків.
5. Які існують відмінності та переваги застосування тривимірних графіків перед двовимірними? Наведіть приклади.