Міністерство освіти і науки України
Львівський національний університет імені Івана Франка
Факультет електроніки та комп’ютерних технологій

Звіт
про виконання лабораторної роботи №10
З курсу “Методи обчислень”
на тему **:**

# «Метод Хука-Дживса багатовимірної оптимізації.»

Виконав
студент групи ФеС-21
Шавало Андрій

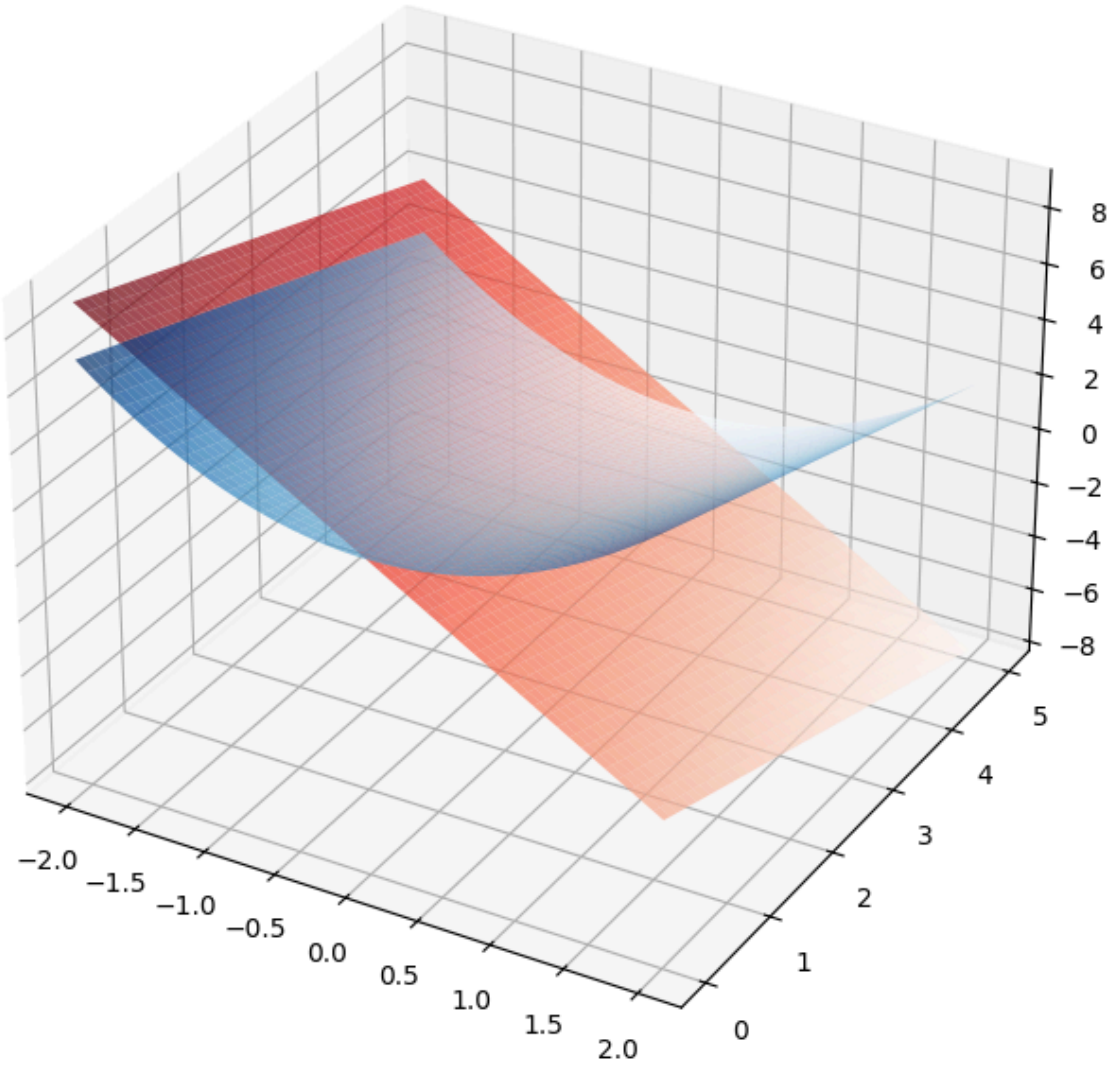
Львів 2025 р.

## Хід роботи

### 1. Задання системи нелінійних рівнянь та графіки

Я задав систему нелінійних рівнянь при m, p. Я побудував графіки обох рівнянь у 3D для візуалізації їхнього перетину. Також за допомогою функції fsolve я знайшов точний розв’язок цієї системи.

```
Розв'язок системи: x1 = 0.0, x2 = 3.0
Перевірка рівнянь у точці розв'язку:
Перше рівняння: 0.0
Друге рівняння: 0.0
Кількість ітерацій: 6
```



2. **Реалізація методу Хука–Дживса для мінімізації**

Я реалізував метод Хука–Дживса для пошуку мінімуму цільової функції. Створив функцію hooke_jeeves з допоміжною exploratory_search.
Для тестування я знайшов мінімум функції:

```
start_point = [-1,10]

solution, value, iter_count = hooke_jeeves(objective_function, start_point)

print(f"Знайдений мінімум у точці: x1 = {solution[0]}, x2 = {solution[1]}")
print(f"Кількість ітерацій: {iter_count}")

Знайдений мінімум у точці: x1 = 0, x2 = 3
Кількість ітерацій: 23
```

3. **Мінімізація функції Φ(x) — розв'язання системи**

Для пошуку розв'язку системи я сформував цільову функцію як суму квадратів: Φ(x)=[f1(x)]^2+[f2(x)]^2 , де f1,f2 - рівняння системи.Я застосував метод Хука–Дживса до функції Φ(x) та знайшов її мінімум, тобто розв'язок системи.

```
def phi(x):
    f1 = x[0]**2 + (p - 5) - x[1]
    f2 = (p - 5) * (1 - x[0]) - x[1]
    return f1**2 + f2**2
start_point = [1.0, 1.0]

solution, value, iter_count = hooke_jeeves(phi, start_point)

print(f"Розв'язок системи (мінімум Φ): x1 = {solution[0]}, x2 = {solution[1]}")
print(f"Значення Φ(x): {value}")
print(f"Кількість ітерацій: {iter_count}")

Розв'язок системи (мінімум Φ): x1 = 0.0, x2 = 3.0
Значення Φ(x): 0.0
Кількість ітерацій: 20
```

4. **Побудова траєкторії спуску, критерії зупинки**

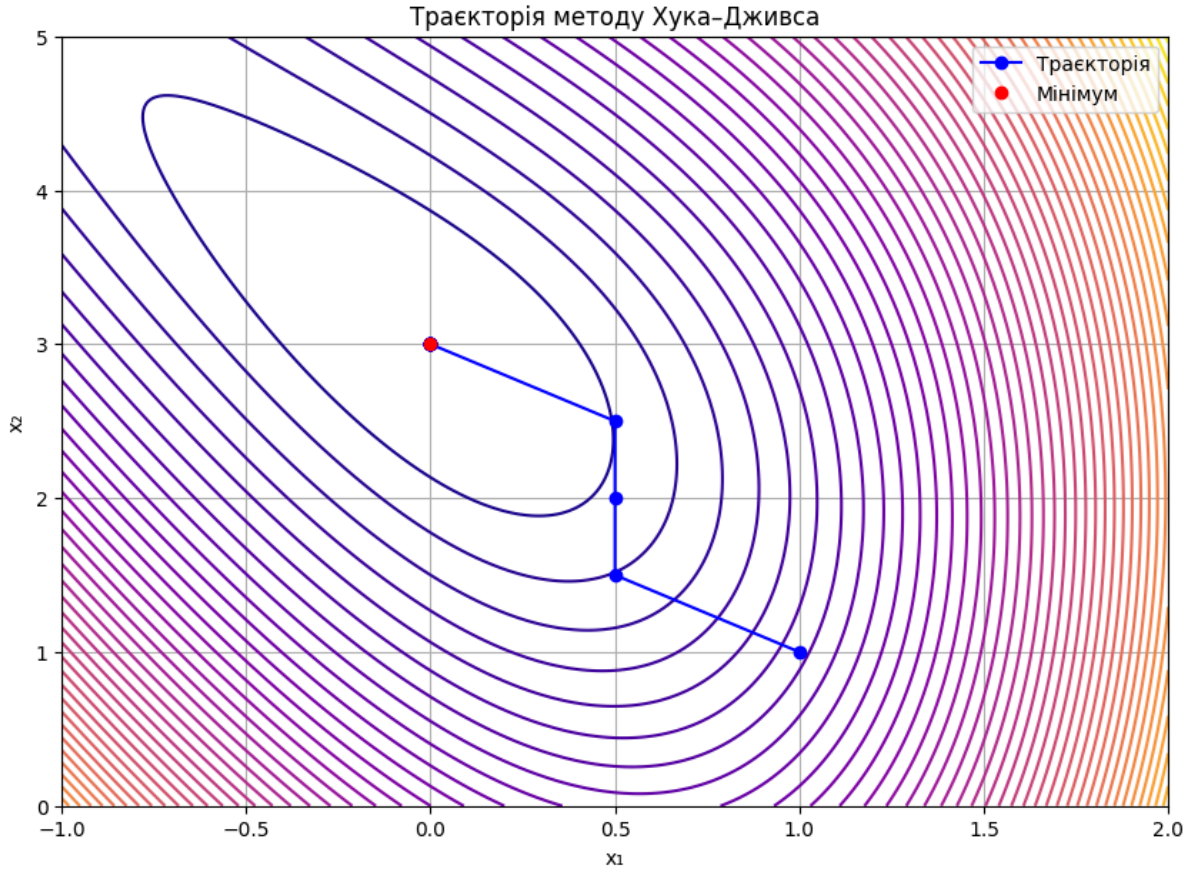
Я реалізував модифікований варіант методу hooke_jeeves_traced, який зберігає всі точки траєкторії. Критерій зупинки: коли крок менше ніж ε=1e−5, або кількість ітерацій перевищує 500.

```
solution, value, iter_count, trajectory = hooke_jeeves_traced(phi, [1.0, 1.0])

with open("trajectory.txt", "w") as file:
    for point in trajectory:
        file.write(f"{point[0]:.8f} {point[1]:.8f}\n")

print(f"Траєкторію збережено, кількість кроків: {len(trajectory)}")
```

```
Траєкторію збережено, кількість кроків: 21
```

5. **Збереження траєкторії у файл, побудова графіка**

Я зберіг траєкторію точок у файл trajectory.txt і побудував контурний графік функції Φ(x) з нанесеною траєкторією пошуку та точкою мінімуму.



**Висновок:** У ході лабораторної роботи я реалізував метод Хука–Дживса для розв’язання системи нелінійних рівнянь. Я задав систему з двох рівнянь, побудував графіки обох функцій та знайшов точний розв’язок системи за допомогою методу fsolve.Я протестував метод Хука–Дживса на простій цільовій функції та переконався в його працездатності. Після цього я сформулював цільову функцію як суму квадратів відхилень системи рівнянь і застосував до неї метод Хука–Дживса.Я задав початкову точку, крок, коефіцієнт зменшення кроку та критерії зупинки. Я знайшов розв’язок системи як точку мінімуму цільової функції та оцінив кількість ітерацій, необхідних для досягнення розв’язку з заданою точністю.Також я зберіг координати точок траєкторії пошуку в текстовий файл і побудував графік траєкторії спуску на рівнях цільової функції.