Національний технічний університет України «КПІ ім. Ігоря Сікорського»
Факультет Інформатики та Обчислювальної Техніки
Кафедра інформаційних систем та технологій

Лабораторна робота № 7
з дисципліни «Методи та технології штучного інтелекту»
На тему:
«Знаходження мінімуму та максимуму функцій за допомогою генетичних алгоритмів»

Виконала:
студентка групи ІС-12.
Мельникова К.О.

Перевірив:
Шимкович В. М.

Київ – 2023

Мета: Знайти мінімум (мінімізація) і максимум (максимізація) функцій одно- і двох змінних за допомогою генетичних алгоритмів.

Варіант:

11.	$y = \sin((x-2)/2) \cdot x \cdot \sin(x-3)$ $z = \cos(y + x/2)$	11.	20.	9.

Завдання:

1) Знайти мінімум функції однієї змінної за варіантом з таблиці Б.

2) Знайти максимум функції двох змінних за варіантом з таблиці 5.

3) Оформити та здати звіт по лабораторній.

Хід роботи:

Для виконання даної лабораторної роботи спершу створимо клас ***GeneticAlgorithm***, що міститиме методи:

- ***generate_population*** для генерації початкової популяції
- ***calculate_y_values*** для обчислення значень функції y
- ***select*** для вибора індивіда для спарювання
- ***crossover*** для створення нащадків
- ***mutate*** для створення випадкової мутації з невеликою ймовірністю
- ***main_cycle*** для виконання основного циклу генетичного алгоритму

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

class GeneticAlgorithm:
    def __init__(self, function, variables_number, bounds, max_search=True, generations_number=100, individuals_number=10):
        self.function = function
        self.variables = variables_number
        self.bounds = bounds
        self.max_search = max_search
        self.generations_number = generations_number
        self.individuals_number = individuals_number

    def generate_population(self):
        return [[random.uniform(*bound) for bound in self.bounds] for _ in range(self.individuals_number)]

    def calculate_y_values(self, individual):
        return self.function(*individual)

    def select(self, population):
        selected_y = [self.calculate_y_values(individual) for individual in population]
        if self.max_search:
            return max(zip(population, selected_y), key=lambda x: x[1])[0]
        else:
            return min(zip(population, selected_y), key=lambda x: x[1])[0]

    def crossover(self, parent_one, parent_two):
        child = []
        for i in range(self.variables):
            if random.random() < 0.5:
                child.append(parent_one[i])
            else:
                child.append(parent_two[i])
        return child
```

```python
    def mutate(self, chromosome):
        for i in range(self.variables):
            if random.random() < 0.01:
                chromosome[i] = random.uniform(*self.bounds[i])
        return chromosome

    def main_cycle(self):
        population = self.generate_population()
        for _ in range(self.generations_number):
            offspring = []
            for _ in range(self.individuals_number):
                parent_one = self.select(population)
                parent_two = self.select(population)
                child = self.crossover(parent_one, parent_two)
                child = self.mutate(child)
                offspring.append(child)
            population = offspring
        return self.select(population)
```

Тепер створимо функції та виведемо для них відповідні максимаьні і мінімальні значення:

```python
def one_variable_function(x):
    return np.sin((x-2)/2) * x * np.sin(x-3)

def two_variable_function(x, y):
    return np.cos(y + x / 2)

x = np.linspace(-10, 10, 400) # one_variable_function
y_values = one_variable_function(x)

y = np.linspace(-10, 10, 400) # two_variable_function
X, Y = np.meshgrid(x, y)
Z = two_variable_function(X, Y)

minFunc1 = np.min(y_values) # використаємо дані значення для порівняння з отриманими у ході роботи генетичного алгоритму
maxFunc2 = np.max(Z)

# Знайдемо мінімум для першої функції
genetic1 = GeneticAlgorithm(one_variable_function, variables_number=1, bounds=[(-10, 10)], max_search=False)
findMin = genetic1.main_cycle()
print(f"Мінімум для першої функції: (x = {findMin}, y = {one_variable_function(*findMin)})")
print(f"Реальне значення y: {minFunc1}")

# Знайдемо максимум для другої функції
genetic2 = GeneticAlgorithm(two_variable_function, variables_number=2, bounds=[(-10, 10), (-10, 10)])
findMax = genetic2.main_cycle()
print(f"Максимум для другої функції (x = {findMax}, z = {two_variable_function(*findMax)})")
print(f"Реальне значення Z: {maxFunc2}")
```

[26] ✓ 0.0s Python

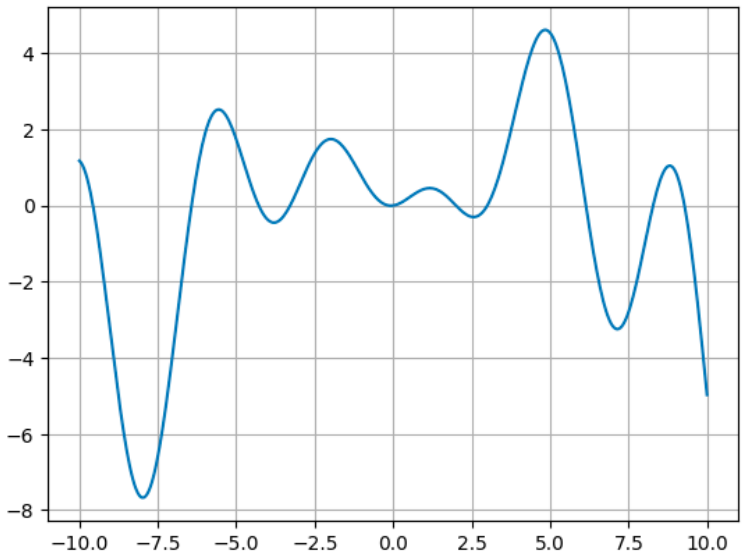
```
Мінімум для першої функції: (x = [-8.96542040631741], y = -3.6381606906970134)
Реальне значення y: -7.672246052870277
Максимум для другої функції (x = [3.887743838983603, -1.932834344175534], z = 0.9999390865839863)
Реальне значення Z: 0.9999967444372527
```

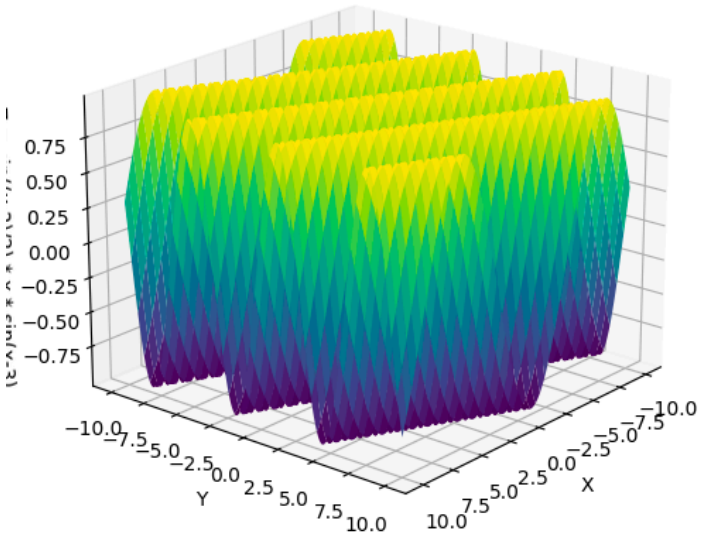
Для візуалізації результатів також побудуємо графіки:

```
plt.plot(x, one_variable_function(x))
plt.grid()
plt.show()

ax = plt.subplot(projection='3d')
ax.plot_surface(X, Y, Z, cmap='viridis', edgecolor='none')
ax.set_xlabel('X')
ax.set_ylabel('Y')
ax.set_zlabel('Z = sin((x-2)/2) * x * sin(x-3)')
ax.view_init(azim=40, elev=20)
plt.tight_layout()
plt.show()
```

✓ 0.3s





Висновок:

Отже, у результаті виконання цієї лабораторної роботі було знайдено мінімум і максимум функцій однієї та двох змінних за допомогою генетичних алгоритмів. У результаті лабораторної роботи було отримано алгоритм, що коректно знаходить шукані значення. Перевірку було здійснено як шляхом обчислень, так і графічно. Розбіжність отриманих та реальних значень є незначною, завдяки чому можна стверджувати, що даний алгоритм є оптимальним і підходить для виконання подібних задач.