**Міністерство освіти і науки України**

**Національний університет «Запорізька Політехніка»**

Кафедра програмних засобів

**ЗВІТ**

з лабораторної роботи №2

з дисципліни «Методи Оптимізації та Дослідження Операцій» на тему:

«Одновимірний пошук оптимуму, Методи оптимізації з виключенням інтервалів»

**Виконав:**

Студент групи КНТ-122 О. А. Онищенко

**Прийняли:**

Викладач: Л. Ю. Дейнега

2024

Одновимірний пошук оптимуму, Методи оптимізації з виключенням інтервалів

Мета роботи

Вивчити одновимірні методи оптимізації з виключенням інтервалів; навчитися застосовувати методи оптимізації для аналізу й обробки інформації.

Постановка задачі

Побудувати графік заданої функції на заданому інтервалі за допомогою пакету matplotlib. Розробити программну реалізацію процедури зменшення інтервалу пошуку з використанням обох вивчених методів одновимірного пошуку.

Функція:

Інтервал: [-1, 3]

Результати виконання

Відокремимо нашу функцію для полегшення звернення до неї:

def f(x):

    return (x - 2) \*\* 2

Далі намалюємо графік нашої функції використовуючи інструменти пакету matplotlib.pyplot:

x = np.linspace(-1, 3, 400)

y = f(x)

plt.figure(figsize=(10, 6))

plt.plot(x, y, label="(x-2)^2")

plt.title("Plot of (x-2)^2")

plt.xlabel("x")

plt.ylabel("y")

plt.legend()

plt.grid(True)

Визначимо функцію, яка буде використовувати метод золотого перетину для пошуку оптимуму:

def goldenSearch(f: callable = f, a: float = -1, b: float = 3, tol: float = 1e-5):

    ratio = (5\*\*0.5 - 1) / 2

    c = b - ratio \* (b - a)

    d = a + ratio \* (b - a)

    while abs(c - d) > tol:

        if f(c) < f(d):

            b = d

        else:

            a = c

        c = b - ratio \* (b - a)

        d = a + ratio \* (b - a)

    return (a + b) / 2

Визначимо функцію, що застосовуватиме метод розподілу інтервалу навпіл для визначення оптимуму:

def bisectionSearch(f: callable = f, a: float = -1, b: float = 3, tol: float = 1e-5):

    while (b - a) >= tol:

        c = (a + b) / 2

        if f(c) == 0.0:

            break

        if f(a) \* f(c) < 0:

            b = c

        else:

            a = c

    return c

Висновки

Таким чином, ми вивчили одновимірні методи оптимізації з виключенням інтервалів, а також навчилися застосовувати методи оптимізації для аналізу й обробки інформації.

Контрольні питання

**Наведіть приклад використання ЛП. Складіть математичну модель задачі**

Завдання - приготувати їжу та напої для пікніка. При обмеженому бюджеті та за наявності серед гостей вегетаріанців, необхідно максимізувати задоволення від пікніка серед усіх гостей.

Математична модель задачі виглядатиме наступним чином:

- - їжа, а - напої.

- та це персональні вподобання кожного гостя щодо різних продуктів.

- - кількість гостей.

Обмеження задачі:

- - вартість їжи,

- - вартість напоїв,

- - бюджет,

- - частка вегетаріанської їжі серед типу їжі,

- - мінімальна необхідна кількість вегетаріанської їжі,

- - максимальна доступна кількість їжі у магазині,

- - максимальна доступна кількість напоїв у магазині.

**Сформулюйте загальну задачу ЛП**

Загальна задача ЛП - визначення максимального або мінімального значення цільової функції при наявних обмеженнях (які є майже завжди, особливо в задачах з реального світу).

Вигляд загальної задачі Лінійного Програмування є наступний:

Із наступними обмеженнями:

- - Задані константи, де це коефіцієнти при змінних у обмеженнях, - значення після знаку нерівності у обмеженнях, а - коефіцієнти при змінних цільової функцією,

- - Змінні (або значення, які необхідно з'ясувати задля розв'язання задачі оптимізації нашої цільової функції),

- , - індекси змінних або констант,

- - кількість змінних,

- - кількість обмежень,

**Дайте визначення стандартної (симетричної) і основної (канонічної) задачі ЛП**

Стандартною або симетричною задачею ЛП називають таку задачу, метою якої є максимізація (або мінімізація) цільової функції за наявних максимальних (верхніх) обмежень, а змінні не можуть бути негативними.

Основною або канонічною задачею ЛП називають таку задачу, метою якої є максимізація (або мінімізація) цільової функцією за наявних мінімальних (нижніх) обмежень, а змінні можуть бути негативними.

Додаток А – Повний вихідний код програми

from rich.console import Console

from rich.traceback import install

import matplotlib.pyplot as plt

import numpy as np

from scipy import optimize

install()

console = Console()

def f(x):

    return (x - 2) \*\* 2

x = np.linspace(-1, 3, 400)

y = f(x)

plt.figure(figsize=(10, 6))

plt.plot(x, y, label="(x-2)^2")

plt.title("Plot of (x-2)^2")

plt.xlabel("x")

plt.ylabel("y")

plt.legend()

plt.grid(True)

def goldenSearch(f: callable = f, a: float = -1, b: float = 3, tol: float = 1e-5):

    ratio = (5\*\*0.5 - 1) / 2

    c = b - ratio \* (b - a)

    d = a + ratio \* (b - a)

    while abs(c - d) > tol:

        if f(c) < f(d):

            b = d

        else:

            a = c

        c = b - ratio \* (b - a)

        d = a + ratio \* (b - a)

    return (a + b) / 2

def bisectionSearch(f: callable = f, a: float = -1, b: float = 3, tol: float = 1e-5):

    while (b - a) >= tol:

        c = (a + b) / 2

        if f(c) == 0.0:

            break

        if f(a) \* f(c) < 0:

            b = c

        else:

            a = c

    return c

with console.status("Оптимізуємо...", spinner="point"):

    # limiting floating point precision to 2 digits after comma

    resGolden: float = f"{goldenSearch():.2f}"

    testGolden: float = f"{optimize.golden(f):.2f}"

    resBisection: float = f"{bisectionSearch():.2f}"

    testBisection: float = f"{optimize.bisect(f, -1, 2):.2f}"

    # scipy bisect не працює для [-1, 3], але чомусь працює для [-1, 2]. результат правильний видає

console.print(f"Golden Section Method: {resGolden}")

console.print(f"Golden Section Method (from scipy): {testGolden}")

console.print(f"Bisection Method: {resBisection}")

console.print(f"Bisection Method (from scipy): {testBisection}")

console.print()

console.print(

    "[green bold]Correct answer found![/green bold]"

    if resGolden == testGolden

    and resBisection == testBisection

    and resGolden == testGolden == resBisection == testBisection

    else "[red bold]Doesn't match![/red bold]"

)

plt.show()