КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ТАРАСА ШЕВЧЕНКА

Кафедра інтелектуальних та інформаційних систем

Лабораторна робота № 2
з дисципліни
“Методи синтезу та оптимізації”

Виконав студент
групи КН- 31
Пашковський Павло Володимирович

Київ-2020

Завдання: Знайти мінімум функції:

$$\frac{x^2}{k_1^2} + \frac{y^2}{k_2^2} = 2z$$
$$k_1, k_2 = \overline{0,5}$$

Номер в списку =26, отже k1, k2 =1.

Функція матиме наступний вигляд: **2z = (x^2) + (y^2) .**

Точність: **e=10^(-6).**

Методи:

- Метод Нелдера-Міда
- Метод найшвидшого спуску

Побудуємо графік функції

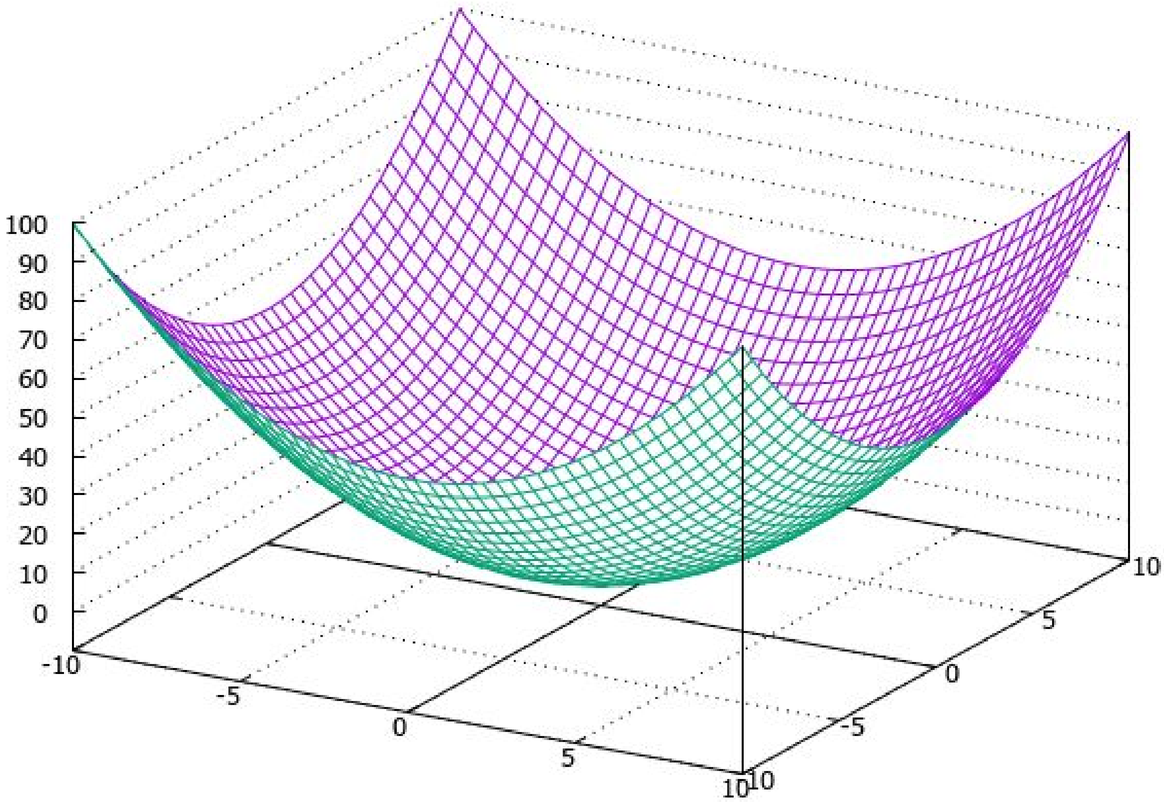

Рис. 1. Графік

Визначимо початкові точки: a = (-1; 1) , b = (1; 1), c = (1; -1).

I Метод Нелдера-Міда.

Алгоритм:

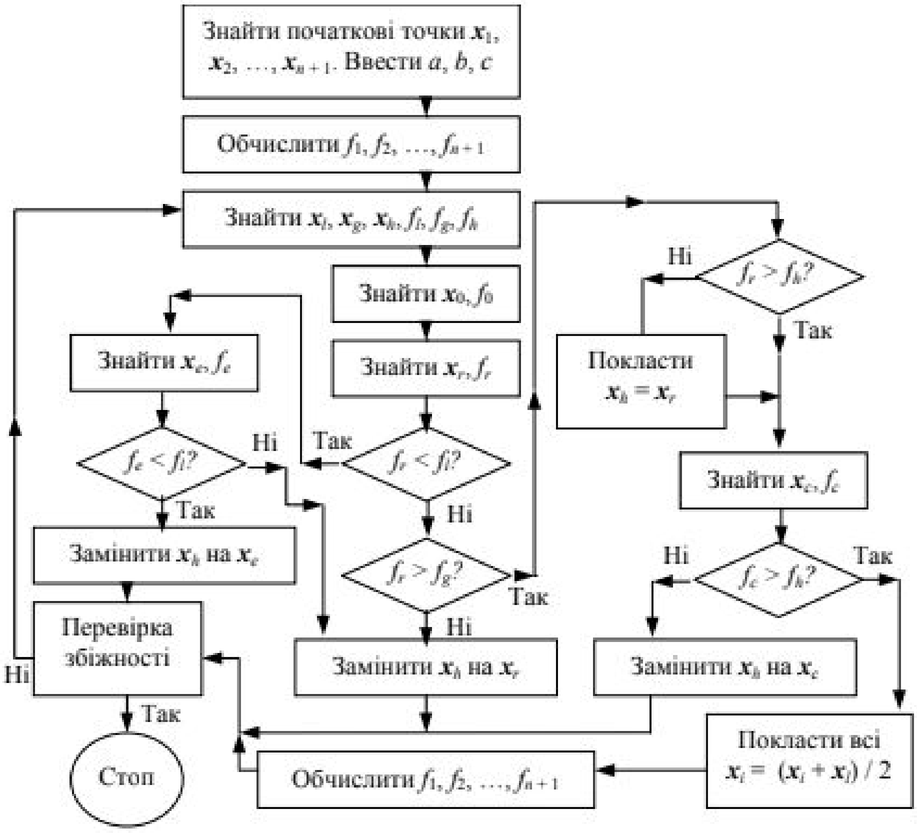

Рис. 2. Алгоритм роботи методу Нелдера-Міда

Результат роботи коду (перші 20 ітерацій):

Початкові точки: (-1, 1), (1, 1), (1, -1)
Ітерація: 1, Точки: (-1, 1), (1, 1), (0.5, 0.0)
Ітерація: 2, Точки: (0.5, 0.0), (1, 1), (0.3125, 0.625)
Ітерація: 3, Точки: (0.5, 0.0), (0.3125, 0.625), (-0.1875, -0.375)
Ітерація: 4, Точки: (-0.1875, -0.375), (0.5, 0.0), (0.1953125, 0.015625)
Ітерація: 5, Точки: (0.1953125, 0.015625), (-0.1875, -0.375), (0.1279296875, -0.134765625)
Ітерація: 6, Точки: (0.1279296875, -0.134765625), (0.1953125, 0.015625), (0.0743408203125, -0.138427734375)
Ітерація: 7, Точки: (0.0743408203125, -0.138427734375), (0.1279296875, -0.134765625), (0.1246795654296875, -0.098541259765625)
Ітерація: 8, Точки: (0.0743408203125, -0.138427734375), (0.1246795654296875, -0.098541259765625), (0.04267120361328125, -0.0859222412109375)
Ітерація: 9, Точки: (0.04267120361328125, -0.0859222412109375), (0.0743408203125, -0.138427734375), (-0.00766754150390625, -0.1258087158203125)

Ітерація: 10, Точки: (0.04267120361328125, -0.0859222412109375), (-0.00766754150390625, -0.1258087158203125), (-0.039337158203125, -0.07330322265625)
Ітерація: 11, Точки: (-0.039337158203125, -0.07330322265625), (0.04267120361328125, -0.0859222412109375), (0.020336151123046875, 0.01277923583984375)
Ітерація: 12, Точки: (0.020336151123046875, 0.01277923583984375), (-0.039337158203125, -0.07330322265625), (-0.061672210693359375, 0.02539825439453125)
Ітерація: 13, Точки: (0.020336151123046875, 0.01277923583984375), (-0.061672210693359375, 0.02539825439453125), (-0.025335311889648438, -0.004009246826171875)
Ітерація: 14, Точки: (0.020336151123046875, 0.01277923583984375), (-0.025335311889648438, -0.004009246826171875), (0.012293577194213867, -0.0008683204650878906)
Ітерація: 15, Точки: (0.012293577194213867, -0.0008683204650878906), (0.020336151123046875, 0.01277923583984375), (-0.004510223865509033, 0.0009731054306030273)
Ітерація: 16, Точки: (-0.004510223865509033, 0.0009731054306030273), (0.012293577194213867, -0.0008683204650878906), (-0.0002194419503211975, -0.003129318356513977)
Ітерація: 17, Точки: (-0.0002194419503211975, -0.003129318356513977), (-0.004510223865509033, 0.0009731054306030273), (0.0012997696176171303, -0.0010256599634885788)
Ітерація: 18, Точки: (0.0012997696176171303, -0.0010256599634885788), (-0.0002194419503211975, -0.003129318356513977), (-0.0007224330911412835, -0.0013148405123502016)
Ітерація: 19, Точки: (-0.0007224330911412835, -0.0013148405123502016), (0.0012997696176171303, -0.0010256599634885788), (0.0007967784767970443, 0.0007888178806751966)
Ітерація: 20, Точки: (0.0007967784767970443, 0.0007888178806751966), (-0.0007224330911412835, -0.0013148405123502016), (-0.0012254242319613695, 0.0004996373318135738)
Точка методом Нелдера-Міла: (0.0007967784767970443, 0.0007888178806751966)

Ⅱ Метод найшвидшого спуску.

Алгоритм:

Рис. 3. Алгоритм роботи методу найшвидшого спуску

Результат роботи коду (перші 20 ітерацій):

Ітерація 0, Точка: -1.4999833589941132, 0.9999889059960755
Ітерація 1, Точка: -1.4999667179882263, 0.999977811992151
Ітерація 2, Точка: -1.4999500769823395, 0.9999667179882265
Ітерація 3, Точка: -1.4999334359764527, 0.9999556239843019
Ітерація 4, Точка: -1.4999167949705658, 0.9999445299803774
Ітерація 5, Точка: -1.499900153964679, 0.9999334359764529
Ітерація 6, Точка: -1.4998835129587922, 0.9999223419725284
Ітерація 7, Точка: -1.4998668719529054, 0.9999112479686039
Ітерація 8, Точка: -1.4998502309470185, 0.9999001539646794
Ітерація 9, Точка: -1.4998335899411317, 0.9998890599607548
Ітерація 10, Точка: -1.4998169489352449, 0.9998779659568303
Ітерація 11, Точка: -1.499800307929358, 0.9998668719529058
Ітерація 12, Точка: -1.4997836669234712, 0.9998557779489813
Ітерація 13, Точка: -1.4997670259175844, 0.9998446839450568
Ітерація 14, Точка: -1.4997503849116975, 0.9998335899411323
Ітерація 15, Точка: -1.4997337439058107, 0.9998224959372077
Ітерація 16, Точка: -1.4997171028999239, 0.9998114019332832
Ітерація 17, Точка: -1.499700461894037, 0.9998003079293587
Ітерація 18, Точка: -1.4996838208881502, 0.9997892139254342
Ітерація 19, Точка: -1.4996671798822634, 0.9997781199215097
Ітерація 20, Точка: -1.4996505388763766, 0.9997670259175852

Код програми:

```
def f(point):
    x, y = point
    return (x ** 2) / 2 + (y ** 2) / 2
def X(x):
    return x
def Y(y):
    return y
def norm(point):
    x, y = point
    return (x ** 2 + y ** 2) ** 0.5
def gradient_descent(h, iter, x0, y0):
    for i in range(iter):
        grad_x = X(x0)
        grad_y = Y(y0)
        iter_norm = norm((x0, y0))
        x_tmp = x0
        y_tmp = y0
        x0 = x0 - h * grad_x / iter_norm
        y0 = y0 - h * grad_y / iter_norm
        if abs(f((x0, y0))) > abs(f((x_tmp, y_tmp))):
            break
        print(f'Ітерація {i}, Точка: {x0}, {y0}')
    return x0, y0

class Vector(object):
    def __init__(self, x, y):
        self.x = x
        self.y = y

    def __repr__(self):
        return f'({self.x}, {self.y})'

    def __add__(self, other):
        x = self.x + other.x
        y = self.y + other.y
        return Vector(x, y)

    def __sub__(self, other):
        x = self.x - other.x
        y = self.y - other.y
        return Vector(x, y)

    def __rmul__(self, other):
        x = self.x * other
        y = self.y * other
        return Vector(x, y)

    def __truediv__(self, other):
```

```
        x = self.x / other
        y = self.y / other
        return Vector(x, y)

    def cords(self):
        return self.x, self.y

def nelder_mead(alpha=1, beta=0.5, gamma=2, iter=10):
    b = 0
    v1 = Vector(-1, 1)
    v2 = Vector(1, 1)
    v3 = Vector(1, -1)
    iteration = 0

    print(f'Початкові точки: {v1}, {v2}, {v3}')
    for i in range(iter):
        adict = {
            v1: f(v1.cords()),
            v2: f(v2.cords()),
            v3: f(v3.cords())
        }
        points = sorted(adict.items(), key=lambda x: x[1])
        b = points[0][0]
        g = points[1][0]
        w = points[2][0]
        mid = (g + b) / 2
        xr = mid + alpha * (mid - w)
        if f(xr.cords()) < f(g.cords()):
            w = xr
        else:
            if f(xr.cords()) < f(w.cords()):
                w = xr
            c = (w + mid) / 2
            if f(c.cords()) < f(w.cords()):
                w = c
        if f(xr.cords()) < f(b.cords()):

            xe = mid + gamma * (xr - mid)
            if f(xe.cords()) < f(xr.cords()):
                w = xe
            else:
                w = xr
        if f(xr.cords()) > f(g.cords()):
            xc = mid + beta * (w - mid)
            if f(xc.cords()) < f(w.cords()):
                w = xc
        v1 = w
        v2 = g
        v3 = b
        iteration += 1
```

```
      print(f'Ітерація: {iteration}, Точки: {b}, {g}, {w}')

  return b
res = nelder_mead()
print(f'Точка методом Нелдера-Міла: {res}')
res = gradient_descent(0.00002, 100, -1.5, 1)
print(f'Точка методом найшвидшого спуску: {res}')
```

Висновки: в ході другої лабораторної роботи були досліджені різні методи пошуку для функції n змінних, а саме: метод найшвидшого спуску та метод Нелдера-Міда. Метод найшвидшого спуску виявився простішим в реалізації, але менш ефективним в порівнянні з методом Нелдера-Міда, оскільки він виявився менш точним та обчислення тривало більшу кількість ітерацій.