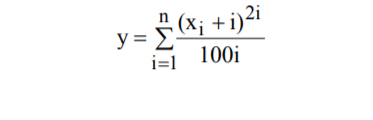
**Завдання**

Розв’язати задачу оптимізації методом найшвидшого спуску

Функція:

**

Тобто **f = (x1 + 1)2/100 + (x2 + 2)4/200**

Початкове наближення:

Х1 = (0.5, 1)

Точність:

е = 0.01

**Аналітичний розв’язок**

*Очевидно, що функція може набувати лише невід’ємних значень, а отже мінімум цієї функції буде 0, при значеннях х1 = -1, х2 = -2. Метод найшвидшого спуску є ітеративним і результат буде залежати від заданої точності. У нашому випадку вона дорівнює 0, 01, значить результат має вийти 0 з точністю до другого знаку після коми.*

**Крок 1**

Шукаємо значення функції у початковому наближенні, щоб спостерігати, як зменшується це значення у процесі алгоритму

f(X1) = 0,0675

Шукаємо загальний випадок напрямку пошуку у просторі керованих змінних х1 та х2.

∇f(X) = (0,02(x1 + 1),

0,02(x2 + 2)3)

Підставляємо значення початкового наближення у цю формулу

∇f(X1) = (0,03,

0,54)

Шукаємо модуль цього вектора, щоб перевірити, чи продовжувати алгоритм (якщо модуль більший за значення e – точність не досягнута, якщо менший чи рівний – досягнута)

|∇f(X1)| ≈ 0,5 > 0.01

Отже, шукаємо X2.

**Крок 2**

За формулою,

X2 = X1 – λ∇f(X1) = (0,5; 1)Т – λ(0,03; 0,54)Т = (0,5 – 0,03λ; 1 – 0,54λ)Т

Шукаємо значення функції у поточному наближенні:

f(X2) = (0,5 – 0,03λ)2/100 + (1 – 0,54λ)4/200

Шукаємо похідну цієї функції і прирівнюємо її до 0:

f `(X2) = -0,0006(0,5 – 0,03λ) – 0,0108(1 – 0,54λ)3= 0

Розв’язуємо рівняння, щоб знайти λ

-0,0003 + 0,000018λ - 0,0108(1 - 3λ + 3λ2 – λ3)= 0

-0,0003 + 0,000018λ - 0,0108 + 0,0324λ – 0,0324λ2 + 0,0108λ3= 0

0,0108λ3 – 0,0324λ2 + 0,032418λ – 0,0111 = 0

λ = 6.47814966555761

Підставляємо значення λ у вектор Х2:

X2 = (0,3056555100332717; -2,498200819401109)Т

Шукаємо градієнт у цьому значенні:

∇f(X2) = (0,0261131102006654; -0,0024731092860607)Т

|∇f(X2)| ≈ 0.02 > 0.01

Отже, шукаємо Х3.

**Крок 3**

X3 = X2 – λ∇f(X2) = (0,3056555100332717; -2,498200819401109)Т – λ(0,0261131102006654; -0,0024731092860607)Т = (0,3056555100332717–0,0261131102006654λ; -2,498200819401109 + 0,0024731092860607λ)Т

Шукаємо значення функції у поточному наближенні:

f(X3) = (0,3056555100332717–0,0261131102006654λ)2/100 + (-2,498200819401109 + 0,0024731092860607λ)4/200

Шукаємо похідну цієї функції і прирівнюємо її до 0:

f `(X3) = -0,0006(0,3056555100332717–0,0261131102006654λ) + 0,0108(-2,498200819401109 + 0,0024731092860607λ)3= 0

Розв’язуємо рівняння:

λ = 56,2673756839549

Підставляємо значення λ у вектор Х3:

X3 = (-1,163660671904083; -2,359045450094854)Т

Шукаємо градієнт у цьому значенні:

∇f(X3) = (-0,0032732134380817; -0.000925717083717798)Т

|∇f(X3)| ≈ 0.003 < 0.01

**Отже f(X3) = 0.000350941782026955 – шуканий розв’язок.**

**Програмна реалізація + візуалізація**

import matplotlib.pyplot as plt

import autograd.numpy as np

from mpl\_toolkits.mplot3d import Axes3D

from matplotlib.colors import LogNorm

from autograd import elementwise\_grad, value\_and\_grad

from sympy.solvers import solve

from sympy import Symbol

from sympy import sqrt

import numpy as np

f = lambda x1, x2: ((x1 + 1)\*\*2)/100 + ((x2 + 2)\*\*4)/200

def gradient(x1, x2):

gradf = [(2/100)\*(x1 + 1), (4/200)\*(x2 + 2)\*\*3]

return gradf

def modgrad(gradf):

return sqrt(gradf[0]\*\*2 + gradf[1]\*\*2)

def deriv(gradf, x\_old, lambd):

lambd = Symbol('l', real=True)

return (-2/100)\*gradf[0]\*(x\_old[0] - lambd\*gradf[0] + 1)\*\*3 - (4/200)\*gradf[1]\*(x\_old[1] - lambd\*gradf[1] + 2)\*\*3

eps = float(input("Eps: "))

mod = 1000

it = 0

while True:

if it == 0:

x\_old = np.array([0.5, 1]) #pochatkove nablyzhennia

grad = np.array(gradient(x\_old[0], x\_old[1]))

mod = modgrad(grad)

if mod < eps:

break

l = Symbol('l', real=True)

e = deriv(grad, x\_old, l)

lamd = solve(e, l)

x\_new = x\_old - lamd\*grad

x\_old = x\_new

it+=1

xmin, xmax, xstep = -10, 10, .1

ymin, ymax, ystep = -10, 10, .1

x1, x2 = np.meshgrid(np.arange(xmin, xmax + xstep, xstep), np.arange(ymin, ymax + ystep, ystep))

z = f(x1, x2)

minima = np.array([x\_new[0], x\_new[1]])

minima\_ = minima.reshape(-1, 1)

fig = plt.figure(figsize=(8, 5))

ax = plt.axes(projection='3d', elev=50, azim=-50)

ax.plot\_surface(x1, x2, z, norm=LogNorm(), rstride=1, cstride=1,

edgecolor='none', alpha=.8, cmap=plt.cm.jet)

ax.plot(\*minima\_, f(\*minima\_), 'r\*', markersize=10)

ax.set\_xlabel('$x$')

ax.set\_ylabel('$y$')

ax.set\_zlabel('$z$')

ax.set\_xlim((xmin, xmax))

ax.set\_ylim((ymin, ymax))

plt.show()

dz\_dx1 = elementwise\_grad(f, argnum=0)(x1, x2)

dz\_dx2 = elementwise\_grad(f, argnum=1)(x1, x2)

fig, ax = plt.subplots(figsize=(10, 6))

ax.contour(x1, x2, z, levels=np.logspace(0, 5, 35), norm=LogNorm(), cmap=plt.cm.jet)

ax.quiver(x1, x2, x1 - dz\_dx1, x2 - dz\_dx2, alpha=.5)

ax.plot(\*minima\_, 'r\*', markersize=18)

ax.set\_xlabel('$x$')

ax.set\_ylabel('$y$')

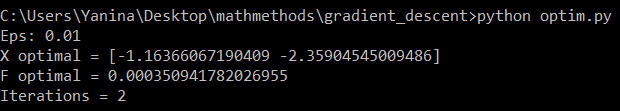
ax.set\_xlim((xmin, xmax))

ax.set\_ylim((ymin, ymax))

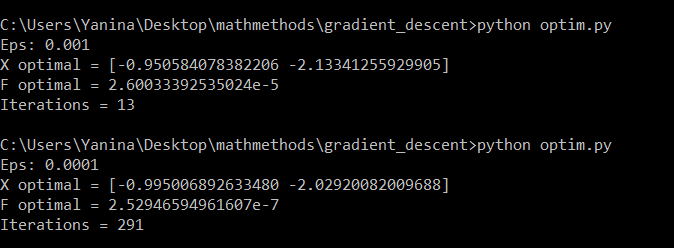
plt.show()

print("X optimal = " + str(x\_new))

print("F optimal = " + str(f(x\_new[0], x\_new[1])))

print("Iterations = " + str(it))

**Збільшуючи точність, значення функції буде наближатися до нульового,а значення х1, х2 до -1 та -2 відповідно.**

****

