НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ

ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

Факультет прикладної математики

Кафедра прикладної математики

Курсова робота

із дисципліни «Дослідження операцій»

на тему: «Модифікований метод Ньютона»

Студентки групи КМ-81

Фрумович Альони

Керівник:

Норкін Б.В.

Київ — 2021

ВСТУП

В ході курсової роботи планується проведення досліджень роботи методу Ньютона мінімізації функцій, а саме - як різні параметри згідно постановки задачі впливають на збіжність методу до теоретичного мінімуму.

Метою проведення досліджень у роботі є отримання набору параметрів, при якому метод оптимізації дає найкращий результат.

Завдання — провести дослідження збіжності методу, варіюючи при цьому схеми обчислення І-ої та ІІ-ої похідних, величину кроку в різнецевих схемах та 2 типів критеріїв закінчення.

Об’єктом дослідження є метод Ньютона мінімізації функцій, а предметом — функція Розенброка та усі параметри, які впливають на роботу методу.

Метод роботи — пошук та систематизація інформації щодо роботи методу Ньютона, написання програми для спрощення та прискорення проведення обрахунків.

ПОСТАНОВКА ЗАДАЧІ

Дослідити збіжність модифікованого методу Ньютона при мінімізації функції

Розенброка в залежності від:

1. Величини кроку h при обчисленні першої та другої похідних.

2. Схеми обчислення першої та другої похідних.

3. Вигляду критерію закінчення.

Використати метод штрафних функцій (метод зовнішньої точки) для умовної

оптимізації в залежності від:

1. Розташування локального мінімума (всередині/поза допустимою

областю).

2. Виду допустимої області (випукла/невипукла).

ТЕОРЕТИЧНІ ОСНОВИ МЕТОДУ НЬЮТОНА

Розкладемо цільову функцію в ряд Тейлора, при цьому всі члени порядку 3 і вище відкидаємо — отримаємо квадратичну апроксимацію f(x):

f(x)=f(x(k))+∇f(x(k))TΔx+1/2ΔxT∇2f(x(k))Δx,

де f(x) — апроксимуюча функція змінної x в точці x(k).

Δx(k)=x(k+1)−x(k)

Якщо Δx(k) — напрямок пошуку в методі Ньютона, тоді

f(x(k+1))=f(x(k))+∇f(x(k))TΔx(k)+1/2Δx(k)T∇2f(x(k))Δx(k)

Мінімум функції f(x) за напрямком визначається диференціюванням f(x) за кожною із компонент, після чого отриманий вираз прирівнюється до 0

∇f(x)=∇f(x(k))+∇2f(x(k))Δx=0.

Звідки

Δx=−[∇2f(x(k))]−1∗∇f(x(k)),

де [∇2f(x(k))]−1 — матриця, обернена до матриці Гессе H(x(k)) в точці x(k).

Перехід з точки в точку у методі Ньютона здійснюється у вказаний далі спосіб:

x(k+1)=x(k)−[∇2f(x(k))]−1∗∇f(x(k)).

Напрямок і величина кроку точно визначені. У задачі пошуку мінімуму довільної квадратичної функції з додатньо визначеною матрицею Гессе метод Ньютона дає рішення за одну ітерацію, вибір початкової точки ролі не грає. У разі загальної нелінійної цільової функції метод Ньютона зійдеться до шуканої точки із квадратичною швидкістю при наступних умовах: матриця Гессе функції, що мінімізується в точці повинна бути додатньо визначена, точка початку ітераційного процесу повинна знаходитися досить близько до x. Так як метод Ньютона базується на квадратичній апроксимації, він має квадратичну швидкість збіжності (згадану вище), тобто виконується нерівність

∥ε(k+1)∥≤C∥ε(k)∥2,

де стала С пов'язана з обумовленістю матриці Гессе. Алгоритму не властиве зменшення значень цільової функції від ітерації до ітерації. Для того, щоб напрямок пошуку був напрямком спуску, повинна виконуватись нерівність

∇f(x(k))T∗S(x(k))<0.

Припустимо, що поточне наближення x(k) не є стаціонарною точкою (тобто ∇f(x(k))≠0), і знайдемо проекцію напрямку пошуку за методом Ньютона на напрям, що задається градієнтом в точці x(k)

−∇f(x(k))T∗∇2f(x(k))−1∗∇f(x(k)<0.

У випадку коли матриця Гессе ∇2f(x(k)) додатньо визначена в точці x(k), наведена умова виконується, отже, напрямок пошуку за методом Ньютона буде напрямком спуску. Якщо в деякій точці ∇2f(x(k)) від’ємно визначена, то зазначений напрямок є напрямком підйому. У разі невизначеності матриці Гессе до однозначного висновку не приходять. За мінімізації, коли всі власні значення додатні ∇2f(x(k)) (матриця додатньо визначена), локальна квадратична апроксимація відповідає круговій чи еліптичній впадині, у якій наявний мінімум. Якщо пара власних значень має протилежні знаки (матриця невизначена), квадратична апроксимація являє собою сідло, що не має локального мінімуму. У цьому випадку рух у напрямку пошуку по методу Ньютона приведе у сідлову точку. Критерій, що гарантує збіжність методу Ньютона в припущенні, що функція f(x) двічі диференційовна, полягає в тому, що матриця, обернена матриці Гессе цільової функції, повинна бути позитивно визначеною.

[∇2f(x(k))]−1≡H(−1)(x(k))>0

ТЕОРЕТИЧНІ ОСНОВИ МЕТОДІВ ШТРАФНИХ ФУНКЦІЙ. МЕТОД ЗОВНІШНЬОЇ ТОЧКИ

Методи штрафних функцій використовуються для вирішення завдань нелінійного програмування. Загальна задача нелінійного програмування виглядає наступним чином:

Мінімізувати f(x);xϵR з обмеженнями

gj(x)≥0;j=1,…,J

hk(x)=0;k=1,…,K

x(l)i≤xi≤x(u)i;i=1,…,n

Передбачається, що відоме початкове наближення x(0), можливо недопустиме. Будується послідовність x(t);t=1,…,T, яка починається з &x^{(0)}& і закінчується точкою x(t), що дає найкраще наближення серед усіх точок побудованої послідовності.

У якості x(t);t=1,…,T беруть стаціонарні точки штрафної функції. Штрафна функція — цільова функція вихідної задачі безумовної оптимізації. За допомогою штрафної функції вихідна задача умовної оптимізації перетворюється в послідовність задач безумовної оптимізації. Методи, засновані на даному підході, визначаються видом штрафної функції, а також правилами, за якими проводиться перерахунок штрафних параметрів на кожному циклі безумовної оптимізації. Штрафна функція, що дозволяє обмежитися рішенням тільки одного завдання безумовної оптимізації, називається точною. При використанні методів штрафних функцій виходить максимальний оптимізуючий ефект за рахунок постійного компромісу між необхідністю виконання обмежень і процесом мінімізації цільової функції f(x), який досягається шляхом присвоєння належних вагів цільової функції і функціям, що задають обмеження. Вплив компонент штрафної функції в процесі оптимізації слабшає, а в границі повністю зникає, так що послідовність проміжних значень штрафної функції сходиться до того ж значення, що і послідовність значень f(x(k)) і, отже, їх екстремуми однакові.

Штрафна функція визначається наступним виразом:

P(x,R)=f(x)+Ф(R,g(x),h(x)),

де R — набір штрафних параметрів; Ф — штраф, функція від R і функцій, які задають обмеження. Функція Ф визначається таким чином, щоб допустимі точки мали перевагу перед недопустимими щодо безумовної оптимізації штрафної функції. Методи, засновані на перетворенні задач умовної оптимізації в задачі безумовної оптимізації повинні відповідати таким критеріям:

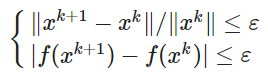
1) Рішення підзадач повинні збігатися до вирішення вихідної задачі.

limt→T<∞x(t)=x∗

2) Складність мінімізації P(x,R) повинна бути того ж порядку, що і мінімізації функції f(x).

3) Правило перерахунку штрафних параметрів R(t+1)=F(R(t)) повинно бути простим (іноді складність перерахунку може бути виправдана за наявності складних функцій f(x)).

Дослідження збіжності функції використовуючи різні критерії закінчення:

1. 

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Точність | h | Різницева схема | Знайдена точка | К-ть обчислень ф-ї | К-ть ітерацій |
| 0.001 | 0.01 | Центральна | (0.98038106, 0.9611467 ) | 179 | 6 |
| 0.001 | 0.001 | Центральна | (0.99979945, 0.99959894) | 149 | 5 |
| 0.001 | 0.0001 | Центральна | (0.99999755, 0.9999951) | 149 | 5 |
| 0.001 | 0.01 | Права | (0.36115249, 0.12543102) | 584 | 20 |
| 0.001 | 0.001 | Права | (0.36115249, 0.12543102) | 265 | 9 |
| 0.001 | 0.0001 | Права | (0.97135571, 0.94348187) | 178 | 6 |
| 0.001 | 0.001 | Ліва | (1.99677774, 3.98762113) | 1234 | 42 |
| 0.001 | 0.0001 | Ліва | (1.03164666, 1.06434482) | 178 | 6 |
| 0.001 | 0.00001 | Ліва | (1.00302238, 1.0060589) | 149 | 5 |

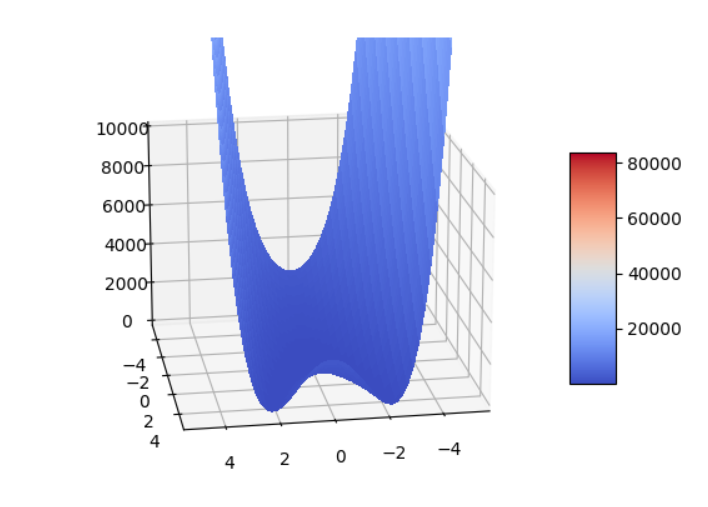
Як видно з вищенаведеної таблиці, центральна схема працює значно краще за ліву та праву. Зі зменшенням кроку h, зменшується кількість обчислень функції та ітерацій і підвищується точність знаходження точки мінімуму. Ліва схема працює гірше за все.

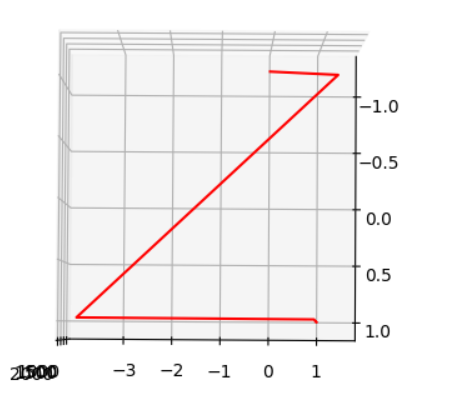
1. 

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Точність | h | Різницева схема | Знайдена точка | К-ть обчислень ф-ї | К-ть ітерацій |
| 0.001 | 0.01 | Центральна | (0.97981775, 0.96004236) | 166 | 5 |
| 0.001 | 0.001 | Центральна | (0.99979945, 0.99959894) | 166 | 5 |
| 0.001 | 0.0001 | Центральна | (0.99999755, 0.9999951) | 166 | 5 |
| 0.001 | 0.01 | Права | (0.36115249 0.12543102) | 646 | 20 |
| 0.001 | 0.001 | Права | (0.7940201, 0.62996732) | 262 | 8 |
| 0.001 | 0.0001 | Права | (0.97155362, 0.9438653) | 166 | 5 |
| 0.001 | 0.01 | Ліва | (1.98480724, 3.93995564) | 966 | 30 |
| 0.001 | 0.0001 | Ліва | (1.03157058, 1.06418733) | 166 | 5 |
| 0.001 | 0.00001 | Ліва | (1.00302238, 1.0060589) | 166 | 5 |

Для цього критерія закінчення, вплив вибору схеми та довжини кроку не змінюється, порівняно з минулим критерієм закінчення.

При виборі центральної схеми, кількість обчислень функції менше для другого критерія закінчення.

Графік функції Розенброка

Траєкторія пошуку точки мінімуму

ВИСНОВКИ

Провівши дослідження, ми з’ясували, що для знаходження точки мінімуму функції Розенброка краще працює центральна різницева схема. При зменшенні кроку h, кількість обчислень функції також зменшується. Критерій закінчення, що базується на нормі градієнта функції, потребує меншої кількості ітерацій.

ДОДАТОК

import math

import numpy as np

import matplotlib.pyplot as plt

from matplotlib import cm

def f(x):

global num

num += 1

return 100 \* (x[0]\*\*2 - x[1])\*\*2 + (x[0] - 1)\*\*2

def l\_r(x0, h):

x1 = [x0[0] - h, x0[1]]

df = []

df.append((f(x0) - f(x1))/h)

x2 = [x0[0], x0[1] - h]

df.append((f(x0) - f(x2))/h)

return df

def c\_r(x0, h):

x1 = [x0[0] + h, x0[1]]

x11 = [x0[0] - h, x0[1]]

df = []

df.append((f(x1) - f(x11))/(2\*h))

x2 = [x0[0], x0[1] + h]

x22 = [x0[0], x0[1] - h]

df.append((f(x2) - f(x22))/(2\*h))

return df

def r\_r(x0, h):

x1 = [x0[0] + h, x0[1]]

df = []

df.append((f(x1) - f(x0))/h)

x2 = [x0[0], x0[1] + h]

df.append((f(x2) - f(x0))/h)

return df

def hes(x0, h):

r = []

f2x1 = (f([x0[0] + 2\* h, x0[1]]) - 2\* f(x0) + f([x0[0] - 2\* h, x0[1]]))/(4 \* h\*\*2)

f2x1x2 = (f([x0[0] + h, x0[1] + h]) - f([x0[0] + h, x0[1] - h]) - f([x0[0] - h, x0[1] + h]) + f([x0[0] - h, x0[1] - h]))/(4 \* h\*\*2)

f2x2 = (f([x0[0], x0[1] + 2\* h]) - 2\* f(x0) + f([x0[0], x0[1] - 2\* h]))/(4 \* h\*\*2)

r.append([f2x1, f2x1x2])

r.append([f2x1x2, f2x2])

return r

def newton(x0, h, eps):

x = [x0]

global x1\_arr

x1\_arr = [x0[0]]

global y1\_arr

y1\_arr = [x0[1]]

global z\_arr

z\_arr = [f(x0)]

k = 0

while np.linalg.norm(l\_r(x[-1], h)) >= eps:

hes\_inv = np.linalg.inv(hes(x[-1], h))

s = hes\_inv.dot(l\_r(x[-1], h))

d = np.matrix(l\_r(x[-1], h))

l = -(d.dot(s))/((s.dot(hes(x[-1],h))).dot(s.transpose()))

l = float(l[0])

x1 = x[-1] - s

x.append(x1)

k += 1

print(x1, k)

def newton1(x0, h, eps):

global x1\_arr

x1\_arr = [x0[0]]

global y1\_arr

y1\_arr = [x0[1]]

global z\_arr

z\_arr = [f(x0)]

x = [x0]

k = 0

while True:

hes\_inv = np.linalg.inv(hes(x[-1], h))

s = hes\_inv.dot(r\_r(x[-1], h))

d = np.matrix(r\_r(x[-1], h))

l = -(d.dot(s))/((s.dot(hes(x[-1],h))).dot(s.transpose()))

l = float(l[0])

x1 = x[-1] - s

x.append(x1)

k += 1

x1\_arr.append(x1[0])

y1\_arr.append(x1[1])

z\_arr.append(f(x1))

if np.linalg.norm(x[-1] - x[-2])/np.linalg.norm(x[-1]) <= eps and abs(f(x[-1]) - f(x[-2])) <= eps:

break

print(x1, k)

num = 0

x0 = [-1.2, 0]

y0 = f(x0)

S1 = [0, 1]

S2 = [1, 0]

eps = 10 \*\* (-3)

h = 10 \*\* (-5)

x1\_arr = []

y1\_arr = []

z\_arr = []

#newton(x0, h, eps)

newton1(x0, h, eps)

print(num)

fig = plt.figure()

ax = fig.gca(projection='3d')

X = np.arange(-5, 5, 0.2)

Y = np.arange(-5, 5, 0.2)

X, Y = np.meshgrid(X, Y)

Z = ((1-X)\*\*2) + (100 \* ((Y-X\*\*2)\*\*2))

surf = ax.plot\_surface(X, Y, Z, cmap=cm.coolwarm,

linewidth=0, antialiased=False)

ax.set\_zlim(-50, 10000)

fig.colorbar(surf, shrink=0.5, aspect=5)

plt.show()

fig = plt.figure()

ax = fig.gca(projection='3d')

plt.plot(x1\_arr, y1\_arr, z\_arr, 'r-')

plt.show()